



**For my parents**

# **Escaping the Bounds of Generality — Unbounded Bi-Objective Optimisation**

by

Adam Michael Berry, BSc. (Hons)

Submitted in fulfilment of the  
Requirements for the Degree of  
Doctor of Philosophy  
University of Tasmania  
(March, 2008)

This thesis contains no material which has been accepted for a degree or diploma by the University or any other institution, except by way of background information and duly acknowledged in the thesis, and to the best of my knowledge and belief no material previously published or written by another person except where due acknowledgement is made in the text of the thesis, nor does the thesis contain any material that infringes copyright. This thesis may be made available for loan and limited copying in accordance with the Copyright Act (1968).

---



# **I     ABSTRACT**

The vast majority of contemporary evolutionary multiobjective optimisation research is grounded in the ideals of generality — that is, the capacity to perform well irrespective of the number of objectives that exist in a given task. Despite this resolute focus, the fact remains that a large portion of the reported real-world applications and existing algorithmic studies are exclusively two-dimensional. By remaining fixated on generality, research has failed to explore the unique properties of these bi-objective tasks and, in so doing, has sacrificed potential performance gains in lieu of an often-unnecessary level of flexibility. As a response, this work focuses on the bi-objective domain, endeavouring to elucidate and then harness the special characteristics of non-dominated bi-objective sets to produce powerful and efficient specialist techniques.

Central to this work is the bi-objective specialisation of the powerful elite archiving mechanism. Where conventional modern systems limit the size of their archives to curb the inefficiencies of naive list constructs (despite the potential for degradation in both the quality of archival members and crowding estimates caused by such artificial thresholding), this thesis describes a new construct (the `Mak_Tree`) that releases artificial size bounds while maintaining high levels of efficiency. Indeed, both theoretical and empirical results illustrate that the `Mak_Tree` performs better than other generalist unbounded methodologies *and* is often more efficient than, or at least competitive with, tightly bound truncated techniques.

Moreover, this thesis indicates that the use of unbounded archives imparts a real practical performance benefit to the optimisation of bi-objective problems. The extension of modern evolutionary optimisers to incorporate the efficient unbounded `Mak_Tree` construct — be it in a passive, active or hybridised manner — results in significant performance improvements across a host of disparate test functions. The creation of novel algorithms designed to capitalise on the properties of the `Mak_Tree` further emphasises the power of specialisation, with significant improvements again noted when results are compared against those produced by the contemporary bounded approaches.

Outside of the improvement of optimiser efficacy, efficient access to unbounded elite sets also offers potential for the development of enhanced meta-processes. Specifically, this thesis proposes and then explores amongst the first fully-realised, intuitive, autonomous and reliable termination systems available to the field — a system that is simply infeasible with access only to bounded archival sets. Additionally, the work examines a new end-of-run presentation system that distills complete unbounded archives into well-distributed collections that are suitable for decision-maker analysis. Empirical results suggest that the proposed system produces significantly better distributions than pre-existing techniques *and* offers guarantees of solution quality that simply cannot be made when using bounded stores.

## **II      ACKNOWLEDGEMENTS**

It is absolutely true to say that this thesis would not be here if it were not for my friends and family. As such, the absolute least that I can do (and that is precisely how much I would like to do) is to provide you each with a couple of words about how special you are to me and how instrumental you have all been in shaping myself and, as a consequence, my work. To Ian, thank you for being a constant, though often terrifying, friend throughout this whole thing — you have elucidated so much to me and I look forward to future elucidation. To Phil and Tash, the cutest couple in the entire world and certainly the most tolerant. I look forward to third wheeling with you guys for years to come. To the Touchy Feelies, for being the single worst touch football team in the history of the world and for providing me with an opportunity for escape from typing and an avenue for the consumption of cheap beer. To the Cows, for giving me a chance to throw balls at heads, lose various pieces of clothing and equipment, and, most of all, for allowing me to get in touch with my inner bogan. To my various office mates, I am sorry for all of the mould and the strange smells emanating from various corners — I promise to clean it up tomorrow. To Ben, Chris, Cat, Steve, Sandra, Chloe and Nick, my cadre of mature friends who can still somehow manage to tolerate my impish charms, despite the various spillages, late arrivals and forgetfulness that comes as part of the imp package. I have never had a bad night with you people. To White Lightning, Black Thunder, Irish Luck and The Italian Stallion, you are my oldest friends and I miss you all greatly. Now this apparently endless thesis is done, we should hit the alleys again soon. Jegan, buddy, even though you fled the state, you are still one of my best mates and I am proud of you. To my baby cousin, Jason, for guaranteeing entertainment at any family function, for being there when I need you and for making the biggest chicken parmas known to man. If you were better looking I would spend more time with you. To my god-daughter Bella, thanks for being the cutest thing in the whole world. You don't know me yet, but now I am free of this thing, just be aware that you always have a godfather who is happy to hang with you any time. To that guy in Zoology who hated me, without you I would never have ended up here. To Rob and Nadia, for using your millions of years of experience to offer me sage advice on issues such as the love a man can have for cows and other dairy animals. You are great friends. To Sunny, for describing Korean octopuses in a detail far too

vivid than ought be allowed and for teaching me how to ensure the production of male babies. Best of luck with your marriage, it is an honour to be invited. To Andrew, Tristan and Tristan, my young whipper-snapper friends who have given me so many stories to tell. If only some of them included something other than vomit. To Claire, my old roomie, thanks for making me brave the real world and for tolerating a near-constant assault on your side-burns. To Sophie, my old buddie, thanks for being there when times were tough a few years back. To Dave and Pip, for very rarely coming to anything they are invited to, but for being fun enough to ensure they get invited again next time anyway. For my badminton team (variously, Pain Train Number Nine and The Cock Slappers), for letting me vent my frustration against bird-flu infested shuttles and for providing me with an outlet for high-fives. To Howard Bloom, for writing a book that blew my mind and started me in this direction. For Adam Hills (he liked my t-shirt!), go you big red fire engine! To Sam and Michael, for importing a variety of delicious American products and for getting me hooked on said products to ensure my ongoing friendship. Remember, good friends put other good friends in fashion shows. To Sam, for being sunshine in an otherwise crappy month, you are like so much cooler than elephants and tree-houses, but I would never tell you that. I can't wait for our adventure, thanks for agreeing to suffer my random moments of insanity over a full month (be afraid!). Oh, and sorry for beating you. What can I say? I'm da bomb. To the other post-grads here, each slacking away in a way that makes me feel like a proud dad watching his son get drunk for the first time. Good luck and don't be afraid to let deadlines slide on by. To my students, for laughing at me in that sympathetic, he-tried-his-best, kind of way. To Peter, for being an excellent supervisor, top-drawer friend and mediocre bowler. Your advice has always been excellent, even when provided from across Bass Strait. To Ray, for stepping into the breach, providing invaluable feedback and for waging war with me on em-dashes — thanks. To Jon-Jon, for having an entirely adorable name, for drinking copious amounts of alcohol with me in Mexico and for tolerating my near-endless badgering of you via e-mail. Hope to catch up with you again some time soon. To my grand parents, the wisest, and yet still amongst the coolest, people I know. Thank you for all of the wonderful stories, support and encouragement. To Arturo, Kalyanmoy, Joshua and Carlos, thank you for answering my many questions. Big shots like you need not have lowered yourself to help me out, but you did, and I appreciate it. To The Arctic Monkeys, Foo Fighters, Ben

Folds, Joy Division and about a million other bands that made me sing at embarrassingly high levels while driving to work every day. I tell ya, I own falsetto, baby! To Dave, my own personal tech support guru, for understanding that I don't really know anything about anything, but I would like it fixed now regardless. To Mark, for his furrowed eye-brows, angry expression and bulging muscles. You are a great guy. Seriously. Don't break me. To my aunties and uncles, for giving me the largest support network of anyone I know, for making me dance at family functions and for demanding an inappropriately large number of kisses whenever I arrive or leave (sadly yes, this does apply to both aunties *and* uncles). To Luke, the hardest worker I have ever seen. Congratulations on your success, don't let us lure you too far into the dark side. To all of the staff and lecturers here, this work is a direct by-product of your words, so thank you. To my near-endless ocean of cousins, for giving me someone to play cricket against at Christmas time and to throw food at around the dinner table. To my poker buddies, for never ever saying "burn and turn!", because that would make you wankers. To Pauline, for absolutely everything over the past three years. Your name is all over this thesis and you deserve it. Most of who I am today and a large part of what I have achieved is directly because of you and I will always remember that. Good luck in everything, you are a fantastic person and you deserve to go on to big wonderful happy things! Don't forget me when you get there, I will always be your friend. To mum and dad, the most tolerant people in the entire universe and amongst the most wonderful to spend time with. I hope this has done you proud, because that is all I have ever wanted. Thank you for being my heroes, my friends and my mentors — without you, this would be a blank piece of paper (literally and figuratively)!

### III TABLE OF CONTENTS

<b>1</b>	<b>INTRODUCTION.....</b>	<b>2</b>
1.1	RESEARCH QUESTIONS .....	3
1.2	KEY CONTRIBUTIONS.....	4
1.3	OUTLINE OF THE THESIS .....	5
<b>2</b>	<b>WHAT IS MULTIOBJECTIVE OPTIMISATION? .....</b>	<b>8</b>
2.1	INTRODUCING PARETO OPTIMALITY .....	10
2.2	FORMALLY DEFINING SOME KEY TERMS IN MULTIOBJECTIVE OPTIMISATION .....	12
2.2.1	<i>Definition of a Multiobjective Solution and Solution-space .....</i>	<i>12</i>
2.2.2	<i>Definition of a Multiobjective Problem .....</i>	<i>13</i>
2.2.3	<i>Definition of Objective-Space .....</i>	<i>13</i>
2.2.4	<i>Definition of a Conflicting Multiobjective Problem .....</i>	<i>13</i>
2.2.5	<i>Definition of Pareto Dominance .....</i>	<i>14</i>
2.2.6	<i>Definition of Equality.....</i>	<i>15</i>
2.2.7	<i>Definition of Incomparability .....</i>	<i>15</i>
2.2.8	<i>Definition of Pareto Fronts and Pareto Optimality.....</i>	<i>15</i>
2.3	AN ILLUSTRATIVE EXAMPLE.....	16
2.3.1	<i>Formula One Front Wing Design .....</i>	<i>16</i>
2.3.2	<i>Defining the Multiobjective Problem and Solution Structure.....</i>	<i>17</i>
2.3.3	<i>Examining Solution and Fitness Spaces .....</i>	<i>18</i>
2.3.4	<i>Moving towards the Pareto Optimal Front .....</i>	<i>19</i>
2.3.5	<i>Termination and Presentation .....</i>	<i>21</i>
2.3.6	<i>Concluding Remarks on the Illustrative Example .....</i>	<i>22</i>
2.4	MULTIOBJECTIVE AND SINGLE-OBJECTIVE OPTIMISATION .....	23
2.4.1	<i>The Argument for Single-Objective Optimisation.....</i>	<i>23</i>
2.4.2	<i>Rejecting the Argument for Single-Objective Optimisation.....</i>	<i>24</i>
2.4.2.1	A Priori requirements.....	24
2.4.2.2	A Posteriori Issues .....	25
2.4.2.3	Unstable User Needs.....	25
2.4.2.4	Descriptiveness .....	26
2.5	CONCLUSIONS .....	26
<b>3</b>	<b>HOW TO PERFORM MULTIOBJECTIVE OPTIMISATION .....</b>	<b>28</b>
3.1.1	<i>Defining a Basic Intelligent Iterative Search Algorithm .....</i>	<i>28</i>
3.1.2	<i>Analysing The Basic Search Algorithm .....</i>	<i>29</i>
3.1.3	<i>Constructing Example Generic Search Algorithms.....</i>	<i>30</i>
3.1.3.1	Random Search .....	30
3.1.3.2	Greedy Hill-Climbing Search .....	32
3.1.3.3	Non-Elitist Population-Based Search .....	32
3.1.3.4	Elitist Population-Based Search.....	32
3.1.4	<i>Solution Variation.....</i>	<i>33</i>
3.1.4.1	The Genetic Operator and Genetic Algorithms .....	33
3.1.5	<i>Defining Solution Utility.....</i>	<i>34</i>
3.1.6	<i>Existing Evolutionary Multiobjective Optimisers.....</i>	<i>35</i>
3.2	CONCLUSIONS .....	38

<b>4</b>	<b>REAL-VALUED MULTIOBJECTIVE PROBLEM CHARACTERISTICS</b>	<b>40</b>
4.1	SHAPE OF THE PARETO OPTIMAL FRONT.....	40
4.2	THE NATURE OF THE OBJECTIVE-SPACE.....	42
4.2.1	<i>Bias</i> .....	42
4.2.2	<i>Deception</i> .....	44
4.2.3	<i>Multiple Fronts</i> .....	44
4.2.4	<i>Noise</i> .....	45
4.2.5	<i>Collateral Noise</i> .....	46
4.2.6	<i>Dynamic Objective-Spaces</i> .....	46
4.2.7	<i>Dimensionality and Extent</i> .....	47
4.2.8	<i>Degenerative Fronts</i> .....	48
4.3	THE NATURE OF THE SEARCH SPACE.....	49
4.3.1	<i>Dimensionality and Extent</i> .....	49
4.3.2	<i>Separability</i> .....	49
4.3.3	<i>Redundancies</i> .....	50
4.3.4	<i>Optima Locality</i> .....	50
4.3.5	<i>Zero Utility-Gradients</i> .....	51
4.3.6	<i>Granularity</i> .....	52
4.4	CONSTRAINTS.....	52
4.5	SUMMARY.....	54
<b>5</b>	<b>A NEW MULTIOBJECTIVE TEST SUITE.....</b>	<b>57</b>
5.1	EXPLORING EXISTING TEST SUITES.....	57
5.1.1	<i>ZDT Functions</i> .....	58
5.1.1.1	Analysing the ZDT Suite.....	58
5.1.2	<i>CTP Functions</i> .....	60
5.1.2.1	Analysing The CTP Suite.....	60
5.1.3	<i>FDA Functions</i> .....	61
5.1.3.1	Analysing The FDA Suite.....	62
5.1.4	<i>WFG Functions</i> .....	62
5.1.4.1	Analysing The WFG Suite.....	63
5.2	DESCRIBING TEST FUNCTIONS.....	63
5.2.1	<i>Visualisations</i> .....	64
5.2.2	<i>Textual Descriptions</i> .....	66
5.2.3	<i>Mathematical Formulae</i> .....	66
5.3	THE NEW ALTERNATIVE PROBLEM SUITE.....	66
5.3.1	<i>Frontal Shape Problems — AP-1, AP-2 and AP-3</i> .....	66
5.3.1.1	AP-1 — Convex Pareto Optimal Front.....	67
5.3.1.2	AP-2 — Concave Pareto Optimal Front.....	67
5.3.1.3	AP-3 — Disconnected Convex Optimal Front.....	68
5.3.2	<i>A Multi-Modal Problem — AP-4</i> .....	69
5.3.3	<i>Biased Problem — AP-5</i> .....	69
5.3.4	<i>Noisy Problems — AP-6 and AP-7</i> .....	70
5.3.4.1	AP-6 — A Gaussian-Based Noisy Function.....	71
5.3.4.2	AP-7 — A Non-Gaussian-Based Noisy Function.....	71
5.3.5	<i>Problems with Side-Constraints — AP-8, AP-9 and AP-10</i> .....	72
5.3.5.1	AP-8.....	72
5.3.5.2	AP-9.....	73
5.3.5.3	AP-10.....	73

5.3.6	<i>Dynamic Problems — AP-11, AP-12 and AP-13</i> .....	74
5.3.6.1	AP-11 .....	74
5.3.6.2	AP-12 .....	75
5.3.6.3	AP-13 .....	76
5.3.6.4	<i>Non-Separable Problems — AP-14 and AP-15</i> .....	77
5.3.6.5	AP-14 .....	77
5.3.6.6	AP-15 .....	78
5.3.7	<i>A Problem with Zero-Utility Gradients — AP-16</i> .....	79
5.3.8	<i>A Deceptive Problem — AP-17</i> .....	80
5.3.9	<i>Problems with Numerous Objectives — AP-18 and AP-19</i> .....	81
5.3.9.1	AP-18 .....	82
5.3.9.2	AP-19 .....	82
5.3.10	<i>A Degenerative Problem — AP-20</i> .....	83
5.3.11	<i>A Multi-Faceted Problem — AP-21</i> .....	84
5.4	CONCLUSIONS .....	86
<b>6</b>	<b>BI-OBJECTIVE OPTIMISATION AND THE MAK_TREE</b> .....	<b>89</b>
6.1	PROPERTIES OF BI-OBJECTIVE PROBLEMS .....	90
6.1.1	<i>Defining an Ordered Non-dominated Bi-Objective List</i> .....	90
6.1.2	<i>Property One: Ordering</i> .....	91
6.1.3	<i>Property Two: Objective-Result Variance</i> .....	93
6.1.4	<i>Property Three: Dominated Sets</i> .....	94
6.1.5	<i>Property Four: Non-Dominance</i> .....	97
6.1.6	<i>Property Five: Dominance</i> .....	98
6.1.7	<i>Summarising The Key Bi-Objective Properties</i> .....	98
6.2	ASSESSING LIMITATIONS OF CONTEMPORARY GENERIC MULTIOBJECTIVE OPTIMISATION .....	98
6.2.1	<i>Elite Archiving</i> .....	99
6.2.1.1	Frontal Degradation .....	100
6.2.1.2	Loss of Expensive Regions .....	104
6.2.1.3	Inaccurate Crowding Estimation.....	105
6.2.1.4	Stopping Conditions.....	109
6.2.1.5	Summarising Deficiencies in Generic Elite Archiving.....	112
6.2.2	<i>Existing Unbounded Archiving Techniques</i> .....	113
6.2.2.1	Mostaghim Quad-Trees .....	114
6.2.2.2	Dominated Trees .....	115
6.2.2.3	Orthogonal Range Searching .....	118
6.3	INTRODUCING THE MAK_TREE .....	119
6.3.1	<i>The Mak_Tree Data Structure</i> .....	119
6.3.2	<i>Red-Black Trees</i> .....	120
6.3.2.1	A Red-Black Mak_Tree .....	121
6.3.3	<i>Updating The Mak_Tree</i> .....	122
6.3.3.1	Verifying Non-Dominance .....	122
6.3.3.2	Locating Dominated Nodes .....	124
6.4	PERFORMANCE OF THE MAK_TREE.....	126
6.4.1	<i>Complexity Analysis</i> .....	126
6.4.1.1	Space Complexity .....	126
6.4.1.2	Run-Time Complexity .....	126
6.4.2	<i>Empirical Performance</i> .....	131
6.4.2.1	The Test Problems .....	131



6.4.2.2	The Implementation of Alternative Elite Storage Techniques ....	133
6.4.2.3	Empirical Results and Discussion .....	137
6.5	CONCLUSIONS AND DISCUSSIONS .....	146
<b>7</b>	<b>HARNESSING THE MAK_TREE .....</b>	<b>148</b>
7.1	STOPPING CONDITIONS .....	148
7.1.1	<i>Termination Using Unbounded Archives .....</i>	<i>149</i>
7.1.2	<i>Key Properties of Termination Conditions That Use</i> <i>Unbounded Elite Sets .....</i>	<i>150</i>
7.1.2.1	Simplicity .....	150
7.1.2.2	Efficiency .....	150
7.1.2.3	Generality .....	151
7.1.2.4	Correctness .....	151
7.1.2.5	Autonomy .....	151
7.1.3	<i>A New Unbounded Approach to Termination in</i> <i>Bi-Objective Domains — Introducing the Mak_Terminator.....</i>	<i>151</i>
7.1.3.1	Defining Solution Utility .....	152
7.1.3.2	Maintaining the Objective-Space Grid .....	152
7.1.3.3	Mak_Terminator Parameters .....	154
7.1.3.4	The Mak_Terminator in Practice .....	154
7.1.3.5	Empirical Results .....	155
7.1.3.6	Limitations and Extensions .....	164
7.2	THE PRESENTATION OF AN UNBOUNDED MAK_TREE .....	165
7.2.1	<i>Empirical Analysis of Presentational Algorithms .....</i>	<i>170</i>
7.3	CONCLUDING REMARKS .....	178
<b>8</b>	<b>EXTENDING THE MAK_TREE.....</b>	<b>180</b>
8.1	MAINTAINING CROWDING INFORMATION.....	180
8.1.1	<i>Generic Crowding Metrics .....</i>	<i>180</i>
8.1.1.1	Fitness Sharing and Distance-Based Clustering.....	180
8.1.1.2	Cell-Based Clustering.....	181
8.1.1.3	$k^{\text{th}}$ Nearest-Neighbour .....	183
8.1.1.4	Cuboid Nearest-Neighbour Crowding.....	184
8.1.1.5	Conclusions About Existing Generic Crowding Metrics .....	185
8.1.2	<i>Single-Nearest-Neighbour Crowding in Mak_Trees.....</i>	<i>186</i>
8.1.3	<i>Cuboid Crowding in Mak_Trees .....</i>	<i>188</i>
8.1.4	$\kappa$ Nearest-Neighbour Crowding in Mak_Trees .....	189
8.1.4.2	Updating $\kappa$ nearest-neighbour Crowds After Deletion.....	192
8.1.4.3	The Complexity of Maintaining $\kappa$ nearest-neighbours.....	193
8.1.4.4	Empirical Results .....	196
8.1.5	<i>Locating Isolated Solutions in the Archive.....</i>	<i>201</i>
8.1.5.1	Annotating The Mak_Tree .....	202
8.1.5.2	Locating A Set of Isolated Nodes.....	207
8.2	THE COMPLEXITY OF THE FULLY-FEATURED EXTENDED MAK_TREE.....	209
8.2.1	<i>Amortised Cost of the Fully Featured Mak_Tree.....</i>	<i>210</i>
8.2.1.1	Amortised Cost when $\kappa > \eta > 0$ .....	210
8.2.1.2	Amortised Cost when $0 < \kappa \leq \eta$ .....	211
8.2.1.3	Amortised Cost when $\eta = 0$ .....	211
8.2.1.4	Overall Amortised Cost of Using the Fully Featured Mak_Tree .....	212
8.2.2	<i>Empirical Results .....</i>	<i>212</i>

8.3	CELL-BASED CROWDING IN MAK_TREES.....	220
8.3.1	<i>Using a Mak_Tree for Cell-Based Density Estimation.....</i>	220
8.3.2	<i>Annotating and Selecting Cells.....</i>	221
8.3.3	<i>Merging Density Procedures.....</i>	221
8.3.4	<i>Empirical Results and Discussion.....</i>	222
8.3.5	<i>Limitations in the Cell-Based Approach.....</i>	229
8.3.6	<i>An Accurate Cell-Based Approach.....</i>	233
8.4	CONCLUSIONS.....	235
<b>9</b>	<b>LIBERATING BOUNDED OPTIMISERS.....</b>	<b>237</b>
9.1	PERFORMANCE ANALYSIS.....	237
9.1.1	<i>Selected Performance Metrics.....</i>	240
9.1.1.1	The Hypervolume Indicator.....	240
9.1.1.2	The Additive Epsilon Indicator.....	241
9.1.1.3	50% Attainment Surfaces.....	241
9.1.1.4	Frequency Matrices.....	242
9.1.2	<i>Methodology.....</i>	243
9.1.2.1	Data Acquisition.....	244
9.1.2.2	Output Representation.....	244
9.2	MAK_TREE INTEGRATION.....	245
9.2.1	<i>The Pareto Archived Evolutionary Algorithm.....</i>	246
9.2.1.1	Describing The PAES Algorithm.....	246
9.2.1.2	Integrating The Mak_Tree Into PAES.....	247
9.2.1.3	Empirical Analysis Methodology.....	247
9.2.1.4	Empirical Analysis.....	248
9.2.2	<i>The Pareto Envelope Selection Algorithm.....</i>	261
9.2.2.1	Describing PESA.....	261
9.2.2.2	Integrating The Mak_Tree Into PESA.....	262
9.2.2.3	Empirical Analysis Methodology.....	263
9.2.2.4	Empirical Analysis.....	263
9.2.3	<i>The Elitist Non-dominated Sorting Genetic Algorithm (NSGA-II) ..</i>	278
9.2.3.1	Describing NSGA-II.....	278
9.2.3.2	Integrating The Mak_Tree Into NSGA-II.....	278
9.2.3.3	Empirical Analysis Methodology.....	280
9.2.3.4	Empirical Analysis.....	280
9.2.4	<i>The Improved Strength Pareto Evolutionary Algorithm (SPEA2)...</i>	295
9.2.4.1	Describing SPEA2.....	295
9.2.4.2	Integrating The Mak_Tree Into SPEA2.....	296
9.2.4.3	Empirical Analysis Methodology.....	297
9.2.4.4	Empirical Analysis.....	298
9.2.5	<i>Conclusions.....</i>	315
<b>10</b>	<b>ANALYSING CONTEMPORARY OPTIMISERS.....</b>	<b>317</b>
10.1	INDICATOR BASED EVOLUTIONARY ALGORITHMS.....	318
10.2	DESCRIBING IBEA.....	318
10.3	EMPIRICAL ANALYSIS METHODOLOGY.....	320
10.4	EMPIRICAL ANALYSIS.....	320
10.4.1	<i>PAES Performance.....</i>	321
10.4.2	<i>Performance of Multi-Member Systems.....</i>	323
10.4.2.1	Diversity-Related Performance.....	323
10.4.2.2	Performance Against Complex Problem Spaces.....	326

10.4.2.3	General Performance .....	328
10.5	CONCLUSIONS .....	329
<b>11</b>	<b>NOVEL UNBOUNDED OPTIMISERS .....</b>	<b>347</b>
11.1	INTRODUCING DIVERSITY_PAES .....	347
11.1.1	Describing Diversity_PAES .....	347
11.2	INTRODUCING MAK_OS .....	349
11.2.1	Describing Mak_OS .....	349
11.3	EMPIRICAL ANALYSIS METHODOLOGY .....	351
11.4	EMPIRICAL ANALYSIS .....	351
11.4.1	Diversity_PAES Performance .....	351
11.4.2	Mak_OS Versus Mak_OS_KNN .....	354
11.4.3	Mak_OS_KNN Performance .....	355
11.4.4	Summarising Key Findings .....	358
11.5	CONCLUSIONS .....	359
<b>12</b>	<b>COMPLETING THE ANALYSES .....</b>	<b>375</b>
12.1	EMPIRICAL ANALYSIS METHODOLOGY .....	375
12.2	EMPIRICAL PERFORMANCE .....	376
12.3	CONCLUSIONS .....	380
<b>13</b>	<b>CONCLUSIONS AND FUTURE WORK .....</b>	<b>400</b>
13.1	CONCLUSIONS .....	400
13.2	FUTURE WORK .....	403
13.3	FINAL THOUGHTS .....	409
<b>14</b>	<b>REFERENCES .....</b>	<b>411</b>
<b>A</b>	<b>TEST PROBLEM APPENDIX .....</b>	<b>430</b>
A.1	EXAMINING AP-1 .....	430
A.2	EXAMINING AP-2 .....	431
A.3	EXAMINING AP-3 .....	432
A.4	EXAMINING AP-4 .....	433
A.5	EXAMINING AP-5 .....	434
A.6	EXAMINING AP-6 .....	435
A.7	EXAMINING AP-7 .....	437
A.8	EXAMINING AP-8 .....	439
A.9	EXAMINING AP-9 .....	440
A.10	EXAMINING AP-10 .....	441
A.11	EXAMINING AP-11 .....	442
A.12	EXAMINING AP-12 .....	444
A.13	EXAMINING AP-13 .....	446
A.14	EXAMINING AP-14 .....	449
A.15	EXAMINING AP-15 .....	451
A.16	EXAMINING AP-16 .....	452
A.17	EXAMINING AP-17 .....	454
A.18	EXAMINING AP-18 .....	456
A.19	EXAMINING AP-19 .....	458
A.20	EXAMINING AP-20 .....	462
A.21	EXAMINING AP-21 .....	464
A.22	AN AUXILIARY TEST PROBLEM — <i>PI</i> .....	466

A.23	AUXILIARY FUNCTIONS FOR TEST PROBLEMS.....	468
<b>B</b>	<b>ALGORITHM PARTICULARS.....</b>	<b>474</b>
B.1	NSGA-II SETTINGS .....	474
B.2	MUTATION OPERATOR USED FOR ALL OPTIMISERS .....	475
<b>C</b>	<b>SUPPLEMENTAL RESULTS .....</b>	<b>477</b>
C.1	THE EFFECT OF CLEANING DOMINATED TREES .....	477
<b>D</b>	<b>NOMENCLATURE.....</b>	<b>481</b>
D.1	STYLE GUIDE .....	481
D.2	SPECIAL NOTATIONS.....	481
D.3	TERMS AND OPERATORS .....	482
D.4	MAK_TREE NODE PROPERTIES .....	483
D.5	ACRONYMS .....	484
<b>E</b>	<b>FULL LISTING OF EPSILON AND HYPERVOLUME STATISTICAL INFERENCES.....</b>	<b>487</b>
E.1	HYPERVOLUME INFERENCES.....	487
E.2	EPSILON INFERENCES.....	490
<b>F</b>	<b>MISCELLANEOUS .....</b>	<b>494</b>
F.1	AVAILABILITY OF SOURCE CODE FOR RED-BLACK TREE IMPLEMENTATIONS 494	
F.2	FREQUENCY MATRIX SETTINGS.....	495
F.3	FURTHER ILLUSTRATIONS .....	495

## IV LIST OF FIGURES

FIGURE 1 — DIFFERENT COMPROMISE SOLUTIONS.....	9
FIGURE 2 — DOMINANCE AND INCOMPARABILITY .....	11
FIGURE 3 — PARETO FRONTS .....	11
FIGURE 4 — AN ILLUSTRATION OF THE FRONT QUARTER OF THE SAUBER BMW F107 .....	16
FIGURE 5 — DECISION-SPACE FOR A SIMPLIFIED F1 WING.....	18
FIGURE 6 — HYPOTHETICAL FITNESS SPACE FOR A SIMPLIFIED FORMULA 1 WING ...	19
FIGURE 7 — END-OF-RUN SETS.....	21
FIGURE 8 — SEXUAL CROSSOVER .....	34
FIGURE 9 — EXAMPLE PARETO OPTIMAL FRONTS. ....	41
FIGURE 10 — EXAMPLE BIASED FITNESS SPACES .....	43
FIGURE 11 — AN EXAMPLE DECEPTIVE FITNESS SPACE .....	44
FIGURE 12 — AN EXAMPLE MULTI-FRONTAL FITNESS SPACE .....	45
FIGURE 13 — AN EXAMPLE NOISY SOLUTION THE UTILITY OF THE SOLUTION CHANGES DEPENDING ON WHEN IT IS OBSERVED.....	46
FIGURE 14 — AN EXAMPLE DYNAMIC FITNESS SPACE.....	47
FIGURE 15 — AN EXAMPLE DEGENERATIVE PROBLEM .....	49
FIGURE 16 — AN EXAMPLE NON-SEPARABLE SOLUTION.....	50
FIGURE 17 — EXAMPLE EXTREME AND MEDIAL OPTIMAL VALUES IN SOLUTION-SPACE .....	51
FIGURE 18 — AN EXAMPLE ZERO-UTILITY GRADIENT SPACE .....	51
FIGURE 19 — EXAMPLE SEARCH-SPACE GRANULARITIES.....	53
FIGURE 20 — EXAMPLE CONSTRAINED SPACES .....	54
FIGURE 21 — ILLUSTRATING COLLATERAL NOISE IN <i>ZDT1</i> .....	59
FIGURE 22 — EXAMPLE SPECTRAL FREQUENCY GRAPHS.....	65
FIGURE 23 - EXAMPLE ORDERED NON-DOMINATED LISTS.....	91
FIGURE 24 - ILLUSTRATING DOMINATED SETS.....	95
FIGURE 25 — AN EXAMPLE OF DOMINATED SETS WITH DUPLICATES .....	96
FIGURE 26 — FRONTAL DEGRADATION IN OBJECTIVE-SPACE .....	101
FIGURE 27 — THE PERCENTAGE OF WEAK SOLUTIONS IN NSGA-II ARCHIVES THAT ARE INCORRECTLY LABELLED AS NON-DOMINATED .....	103
FIGURE 28 — NUMBER OF SOLUTIONS TRUNCATED FROM THE ELITE ( $\mathcal{A}$ ) PORTION OF THE NSGA-II ARCHIVE.....	104
FIGURE 29 - LOSING A VALUABLE REGION .....	105
FIGURE 30 — MISLEADING OBJECTIVE-SPACE CROWDING INFORMATION .....	106
FIGURE 31 — CROWDING INACCURACIES IN TRUNCATED ELITE SETS.....	107
FIGURE 32 — DANGERS OF GLOBAL CROWDING MEASURES IN A TRUNCATED ENVIRONMENT .....	109
FIGURE 33 — STOPPING CONDITION PERFORMANCE IN TRUNCATED SETS.....	111
FIGURE 34 — AN EXAMPLE MOSTAGHIM QUAD-TREE FOR UNBOUNDED ELITE ARCHIVING.....	114
FIGURE 35 — EXAMPLE DOMINATED AND NON-DOMINATED TREES .....	116
FIGURE 36 — THE COST OF REBUILDING DOMINATED TREES .....	117
FIGURE 37 — AN EXAMPLE DYNAMIC RANGE TREE .....	118
FIGURE 38 — UNBALANCED AND BALANCED BINARY SEARCH TREES .....	120
FIGURE 39 — AN EXTREMELY UNBALANCED RED-BLACK TREE .....	121
FIGURE 40 — AN EXAMPLE RED-BLACK MAK <sub>2</sub> TREE .....	122

FIGURE 41 — ILLUSTRATING THE RANGE PROPERTIES OF BINARY SEARCH TREES..	124
FIGURE 42 — ILLUSTRATING THE RANGE PROPERTIES OF MAK_TREES.....	124
FIGURE 43 — LOCATING DOMINATED NODES IN AN EXAMPLE MAK_TREE.....	125
FIGURE 44 — ELITE UNBOUNDED ARCHIVE SIZES .....	132
FIGURE 45 — ACCEPTANCE, REJECTION AND REMOVAL RATIOS.....	134
FIGURE 46 — AVERAGE CUMULATIVE TIME COSTS OF DIFFERING UNBOUNDED ARCHIVING TECHNIQUES.....	139
FIGURE 47 — AVERAGE CUMULATIVE TIME COSTS OF DIFFERING UNBOUNDED ARCHIVING TECHNIQUES.....	140
FIGURE 48 — AVERAGE CUMULATIVE TIME COSTS OF DIFFERING UNBOUNDED ARCHIVING TECHNIQUES.....	141
FIGURE 49 — AVERAGE CUMULATIVE TIME COSTS OF UNBOUNDED AND BOUNDED ARCHIVING TECHNIQUES.....	143
FIGURE 50 — AVERAGE CUMULATIVE TIME COSTS OF UNBOUNDED AND BOUNDED ARCHIVING TECHNIQUES.....	144
FIGURE 51 —AVERAGE CUMULATIVE TIME COSTS OF DIFFERING UNBOUNDED ARCHIVING TECHNIQUES.....	145
FIGURE 52 — SOLUTION UTILITY IN OBJECTIVE-SPACE GRID .....	152
FIGURE 53 — AVERAGE $R$ -SCORES FOR VARIOUS PRECISION LEVELS .....	158
FIGURE 54 — AVERAGE $R$ -SCORES FOR VARIOUS PRECISION LEVELS .....	159
FIGURE 55 — AVERAGE $R$ -SCORES FOR VARIOUS PRECISION LEVELS .....	160
FIGURE 56 — MEDIAN NUMBER OF EVALUATIONS BEFORE CONVERGENCE TERMINATION .....	160
FIGURE 57 — MEDIAN FRONTS AS AT MAK_TERMINATION.....	161
FIGURE 58 — MEDIAN FRONTS AS AT MAK_TERMINATION.....	162
FIGURE 59 — MEDIAN FRONTS AS AT MAK_TERMINATION.....	163
FIGURE 60 — EXAMPLE FRONTAL PROGRESSION ON $AP$ -2 WITH $P = 0.05$ .....	163
FIGURE 61 — ARCHIVAL TRUNCATION LEADING TO POORLY DISTRIBUTED PRESENTATION SETS.....	166
FIGURE 62 — ARCHIVAL TRUNCATION LEADING TO APPROPRIATELY DISTRIBUTED PRESENTATION SETS.....	167
FIGURE 63 — NEAREST NEIGHBOURS AND SUCCESSOR DISTANCES IN DETERMINING PRESENTATION SETS .....	172
FIGURE 64 — $Q$ -VALUE PERFORMANCE OF PRESENTATIONAL ALGORITHMS (BOX PLOTS).....	173
FIGURE 65 — EXAMPLE PRESENTATION SETS ( $r=20$ ).....	174
FIGURE 66 — EXAMPLE PRESENTATION SETS ( $r=20$ ).....	175
FIGURE 67 — EXAMPLE PRESENTATION SETS ( $r=20$ ).....	176
FIGURE 68 — EXAMPLE PRESENTATION SETS ( $r=20$ ).....	177
FIGURE 69 — CELL-BASED CROWDING INACCURACIES .....	182
FIGURE 70 — $K^{TH}$ NEAREST-NEIGHBOUR INACCURACIES .....	184
FIGURE 71 — CUBOID CROWDING INACCURACIES .....	185
FIGURE 72 — UPDATING NEAREST NEIGHBOURS AFTER INSERTION AND DELETION.....	188
FIGURE 73 — UPDATING THE INSERTED NODE FOR $K$ NEAREST-NEIGHBOURS CROWDING.....	190
FIGURE 74 — UPDATING THE NEIGHBOURHOOD FOR $K$ NEAREST-NEIGHBOURS CROWDING AFTER INSERTION .....	192
FIGURE 75 — UPDATING NEIGHBOURHOODS FOR $K$ NEAREST-NEIGHBOURS CROWDING AFTER DELETION.....	195

FIGURE 76 — AVERAGE CUMULATIVE TIME COSTS OF UNBOUNDED AND BOUNDED ARCHIVING TECHNIQUES WITH COMPLETE DENSITY ESTIMATES ( $k = 20$ ) .....	198
FIGURE 77 — AVERAGE CUMULATIVE TIME COSTS OF UNBOUNDED AND BOUNDED ARCHIVING TECHNIQUES WITH COMPLETE DENSITY ESTIMATES ( $k = 20$ ) .....	199
FIGURE 78 — AVERAGE CUMULATIVE TIME COSTS OF UNBOUNDED AND BOUNDED ARCHIVING TECHNIQUES WITH COMPLETE DENSITY ESTIMATES ( $k = 20$ ) .....	200
FIGURE 79 — EXAMPLE TREE ANNOTATIONS.....	202
FIGURE 80 — AN EXAMPLE ANNOTATION UPDATE AFTER INSERTION .....	204
FIGURE 81 — EXAMPLE ANNOTATION UPDATES WITH DELETION .....	206
FIGURE 82 — AN EXAMPLE OF SELECTING MULTIPLE UNCROWDED NODES FROM THE MAK_TREE.....	208
FIGURE 83 — AVERAGE CUMULATIVE TIME COSTS OF UNBOUNDED AND BOUNDED ARCHIVING TECHNIQUES WITH COMPLETE DENSITY ESTIMATES, ANNOTATIONS AND SELECTION ( $k = 20, \phi = 50$ ).....	216
FIGURE 84 — AVERAGE CUMULATIVE TIME COSTS OF UNBOUNDED AND BOUNDED ARCHIVING TECHNIQUES WITH COMPLETE DENSITY ESTIMATES, ANNOTATIONS AND SELECTION ( $k = 20, \phi = 50$ ).....	217
FIGURE 85 — AVERAGE CUMULATIVE TIME COSTS OF UNBOUNDED AND BOUNDED ARCHIVING TECHNIQUES WITH COMPLETE DENSITY ESTIMATES, ANNOTATIONS AND SELECTION ( $k = 20, \phi = 50$ ).....	218
FIGURE 86 — AVERAGE CUMULATIVE TIME COSTS OF CELL-BASED ARCHIVING TECHNIQUES AND PRECISE ARCHIVING APPROACHES ( $B = 0.001, k = 20, \phi = 50$ ).....	223
FIGURE 87 — AVERAGE CUMULATIVE TIME COSTS OF CELL-BASED ARCHIVING TECHNIQUES AND PRECISE ARCHIVING APPROACHES ( $B = 0.001, k = 20, \phi = 50$ ).....	224
FIGURE 88 — AVERAGE CUMULATIVE TIME COSTS OF CELL-BASED ARCHIVING TECHNIQUES AND PRECISE ARCHIVING APPROACHES ( $B = 0.001, k = 20, \phi = 50$ ).....	225
FIGURE 89 — AVERAGE NUMBER OF NODES STORED IN VARIOUS MAK_TREE.....	227
FIGURE 90 — AVERAGE NUMBER OF NODES STORED IN VARIOUS MAK_TREE.....	228
FIGURE 91 — AVERAGE NUMBER OF NODES STORED IN VARIOUS MAK_TREE.....	229
FIGURE 92 — AN EXAMPLE OF THE LIMITATIONS OF CELL-BASED MAK_TREE.....	229
FIGURE 93 — AVERAGE NUMBER OF SOLUTIONS STORED IN VARIOUS MAK_TREE .....	231
FIGURE 94 — AVERAGE NUMBER OF SOLUTIONS STORED IN VARIOUS MAK_TREE .....	232
FIGURE 95 — AVERAGE NUMBER OF SOLUTIONS STORED IN VARIOUS MAK_TREE .....	233
FIGURE 96 — AN EXAMPLE RUN-TIME PERFORMANCE ANALYSIS.....	239
FIGURE 97 — AN EXAMPLE HYPERVOLUME.....	240
FIGURE 98 — AN EXAMPLE 50% ATTAINMENT SURFACE .....	242
FIGURE 99 - AN EXAMPLE FREQUENCY MATRIX .....	243
FIGURE 100 — PROGRESSIVE HYPERVOLUME AVERAGES .....	250
FIGURE 101 — PROGRESSIVE HYPERVOLUME AVERAGES .....	251
FIGURE 102 — END-OF-RUN HYPERVOLUME BOX-PLOTS.....	252
FIGURE 103 — PROGRESSIVE EPSILON AVERAGES .....	253

FIGURE 104 — PROGRESSIVE EPSILON AVERAGES .....	254
FIGURE 105 — END-OF-RUN EPSILON BOX-PLOTS .....	255
FIGURE 106 — END-OF-RUN FREQUENCY MATRICES .....	257
FIGURE 107 — END-OF-RUN FREQUENCY MATRICES .....	258
FIGURE 108 — 50% ATTAINMENT SURFACES .....	259
FIGURE 109 — 50% ATTAINMENT SURFACES .....	260
FIGURE 110 — PROGRESSIVE HYPERVOLUME AVERAGES .....	266
FIGURE 111 — PROGRESSIVE HYPERVOLUME AVERAGES .....	267
FIGURE 112 — END-OF-RUN HYPERVOLUME BOX-PLOTS .....	268
FIGURE 113 — PROGRESSIVE EPSILON AVERAGES .....	270
FIGURE 114 — PROGRESSIVE EPSILON AVERAGES .....	271
FIGURE 115 — END-OF-RUN EPSILON BOX-PLOTS .....	272
FIGURE 116 — END-OF-RUN FREQUENCY MATRICES .....	274
FIGURE 117 — END-OF-RUN FREQUENCY MATRICES .....	275
FIGURE 118 — 50% ATTAINMENT SURFACES .....	276
FIGURE 119 — 50% ATTAINMENT SURFACES .....	277
FIGURE 120 — PROGRESSIVE HYPERVOLUME AVERAGES .....	284
FIGURE 121 — PROGRESSIVE HYPERVOLUME AVERAGES .....	285
FIGURE 122 — END-OF-RUN HYPERVOLUME BOX-PLOTS .....	286
FIGURE 123 — PROGRESSIVE EPSILON AVERAGES .....	287
FIGURE 124 — PROGRESSIVE EPSILON AVERAGES .....	288
FIGURE 125 — END-OF-RUN EPSILON BOX-PLOTS .....	289
FIGURE 126 — END-OF-RUN FREQUENCY MATRICES .....	291
FIGURE 127 — END-OF-RUN FREQUENCY MATRICES .....	292
FIGURE 128 — 50% ATTAINMENT SURFACES .....	293
FIGURE 129 — 50% ATTAINMENT SURFACES .....	294
FIGURE 130 - THE PROGRESSIVE SIZE OF THE MAK_SPEA2_KNN UNBOUNDED SETS .....	302
FIGURE 131 — EXAMPLE SELECTIONS WITH AVERAGED $k$ NEAREST-NEIGHBOURS .....	302
FIGURE 132 — PROGRESSIVE HYPERVOLUME AVERAGES .....	303
FIGURE 133 — PROGRESSIVE HYPERVOLUME AVERAGES .....	304
FIGURE 134 — END-OF-RUN HYPERVOLUME BOX-PLOTS .....	305
FIGURE 135 — PROGRESSIVE EPSILON AVERAGES .....	307
FIGURE 136 — PROGRESSIVE EPSILON AVERAGES .....	308
FIGURE 137 — END-OF-RUN EPSILON BOX-PLOTS .....	309
FIGURE 138 — END-OF-RUN FREQUENCY MATRICES .....	311
FIGURE 139 — END-OF-RUN FREQUENCY MATRICES .....	312
FIGURE 140 — 50% ATTAINMENT SURFACES .....	313
FIGURE 141 — 50% ATTAINMENT SURFACES .....	314
FIGURE 142 — PROGRESSIVE HYPERVOLUME AVERAGES .....	331
FIGURE 143 — PROGRESSIVE HYPERVOLUME AVERAGES .....	332
FIGURE 144 — PROGRESSIVE HYPERVOLUME AVERAGES .....	333
FIGURE 145 — END-OF-RUN HYPERVOLUME BOX-PLOTS .....	334
FIGURE 146 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN HYPERVOLUME RESULTS .....	335
FIGURE 147 — PROGRESSIVE EPSILON AVERAGES .....	336
FIGURE 148 — PROGRESSIVE EPSILON AVERAGES .....	337
FIGURE 149 — PROGRESSIVE EPSILON AVERAGES .....	338
FIGURE 150 — END-OF-RUN EPSILON BOX-PLOTS .....	339



FIGURE 151 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN EPSILON RESULTS.....	340
FIGURE 152 —BEST AND WORST MEDIAN END-OF-RUN 50% ATTAINMENT SURFACES FOR EACH PROBLEM .....	341
FIGURE 153 —BEST AND WORST END-OF-RUN 50% ATTAINMENT SURFACES FOR EACH PROBLEM .....	342
FIGURE 154 — END-OF-RUN FREQUENCY MATRICES.....	343
FIGURE 155 — END-OF-RUN FREQUENCY MATRICES.....	344
FIGURE 156 —AVERAGE SIZE OF POST-TRUNCATION CELLS USING PESA.....	345
FIGURE 157 — PROGRESSIVE HYPERVOLUME AVERAGES .....	361
FIGURE 158 — PROGRESSIVE HYPERVOLUME AVERAGES .....	362
FIGURE 159 — PROGRESSIVE HYPERVOLUME AVERAGES .....	363
FIGURE 160 — END-OF-RUN HYPERVOLUME BOX-PLOTS.....	364
FIGURE 161 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN HYPERVOLUME RESULTS.....	365
FIGURE 162 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN HYPERVOLUME RESULTS.....	366
FIGURE 163 — PROGRESSIVE EPSILON AVERAGES .....	367
FIGURE 164 — PROGRESSIVE EPSILON AVERAGES .....	368
FIGURE 165 — PROGRESSIVE EPSILON AVERAGES .....	369
FIGURE 166 — END-OF-RUN EPSILON BOX-PLOTS.....	370
FIGURE 167 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN EPSILON RESULTS.....	371
FIGURE 168 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN EPSILON RESULTS.....	372
FIGURE 169 — AVERAGE NUMBER OF SOLUTIONS EXTRACTED FROM THE ARCHIVE IN MAK_OS_KNN.....	373
FIGURE 170 — ILLUSTRATING RANKED BOX-PLOTS .....	376
FIGURE 171 — PROGRESSIVE HYPERVOLUME AVERAGES .....	383
FIGURE 172 — PROGRESSIVE HYPERVOLUME AVERAGES .....	384
FIGURE 173 — END-OF-RUN HYPERVOLUME BOX PLOTS WITH STATISTICAL RANKS .....	385
FIGURE 174 — END-OF-RUN HYPERVOLUME BOX PLOTS WITH STATISTICAL RANKS .....	386
FIGURE 175 — END-OF-RUN HYPERVOLUME BOX-PLOTS WITH STATISTICAL RANKS AND SUMMARY HYPERVOLUME GRAPHS (NORMALISED) .....	387
FIGURE 176 — END-OF-RUN HYPERVOLUME MEDIAN AND AVERAGES .....	388
FIGURE 177 — END-OF-RUN HYPERVOLUME MEDIAN AND AVERAGES .....	389
FIGURE 178 — END-OF-RUN HYPERVOLUME MEDIAN AND AVERAGES .....	390
FIGURE 179 — PROGRESSIVE EPSILON AVERAGES .....	391
FIGURE 180 — PROGRESSIVE EPSILON AVERAGES .....	392
FIGURE 181 — END-OF-RUN EPSILON BOX PLOTS WITH STATISTICAL RANKS.....	393
FIGURE 182 — END-OF-RUN EPSILON BOX PLOTS WITH STATISTICAL RANKS.....	394
FIGURE 183 — END-OF-RUN EPSILON BOX PLOTS WITH STATISTICAL RANKS (AP-17, AP-21) AND SUMMARY EPSILON GRAPHS (NORMALISED) .....	395
FIGURE 184 — END-OF-RUN EPSILON MEDIAN AND AVERAGES .....	396
FIGURE 185 — END-OF-RUN EPSILON MEDIAN AND AVERAGES .....	397
FIGURE 186 — END-OF-RUN EPSILON MEDIAN AND AVERAGES .....	398
FIGURE A1 — PROBLEM CHARACTERISTICS FOR AP-1 .....	430
FIGURE A2 — PROBLEM CHARACTERISTICS FOR AP-2 .....	431

FIGURE A3 — PROBLEM CHARACTERISTICS FOR <i>AP-3</i> .....	432
FIGURE A4 — PROBLEM CHARACTERISTICS FOR <i>AP-4</i> .....	433
FIGURE A5 — PROBLEM CHARACTERISTICS FOR <i>AP-5</i> .....	434
FIGURE A6 — PROBLEM CHARACTERISTICS FOR <i>AP-6</i> .....	435
FIGURE A7 — PROBLEM CHARACTERISTICS FOR <i>AP-6</i> .....	436
FIGURE A8 — PROBLEM CHARACTERISTICS FOR <i>AP-7</i> .....	437
FIGURE A9 — PROBLEM CHARACTERISTICS FOR <i>AP-7</i> .....	438
FIGURE A10 — PROBLEM CHARACTERISTICS FOR <i>AP-8</i> .....	439
FIGURE A11 — PROBLEM CHARACTERISTICS FOR <i>AP-9</i> .....	440
FIGURE A12 — PROBLEM CHARACTERISTICS FOR <i>AP-10</i> .....	441
FIGURE A13 — PROBLEM CHARACTERISTICS FOR <i>AP-11</i> .....	442
FIGURE A14 — PROBLEM CHARACTERISTICS FOR <i>AP-11</i> .....	443
FIGURE A15 — PROBLEM CHARACTERISTICS FOR <i>AP-12</i> .....	444
FIGURE A16 — PROBLEM CHARACTERISTICS FOR <i>AP-12</i> .....	445
FIGURE A17 — PROBLEM CHARACTERISTICS FOR <i>AP-13</i> .....	446
FIGURE A18 — PROBLEM CHARACTERISTICS FOR <i>AP-13</i> .....	447
FIGURE A19 — PROBLEM CHARACTERISTICS FOR <i>AP-13</i> .....	448
FIGURE A20 — PROBLEM CHARACTERISTICS FOR <i>AP-14</i> .....	449
FIGURE A21 — PROBLEM CHARACTERISTICS FOR <i>AP-14</i> .....	450
FIGURE A22 — PROBLEM CHARACTERISTICS FOR <i>AP-15</i> .....	451
FIGURE A23 — PROBLEM CHARACTERISTICS FOR <i>AP-16</i> .....	452
FIGURE A24 — PROBLEM CHARACTERISTICS FOR <i>AP-16</i> .....	453
FIGURE A25 — PROBLEM CHARACTERISTICS FOR <i>AP-17</i> .....	454
FIGURE A26 — PROBLEM CHARACTERISTICS FOR <i>AP-17</i> .....	455
FIGURE A27 — PROBLEM CHARACTERISTICS FOR <i>AP-18</i> .....	456
FIGURE A28 — PROBLEM CHARACTERISTICS FOR <i>AP-18</i> .....	457
FIGURE A29 — PROBLEM CHARACTERISTICS FOR <i>AP-19</i> .....	458
FIGURE A30 — PROBLEM CHARACTERISTICS FOR <i>AP-19</i> .....	459
FIGURE A31 — PROBLEM CHARACTERISTICS FOR <i>AP-19</i> .....	460
FIGURE A32 — PROBLEM CHARACTERISTICS FOR <i>AP-19</i> .....	461
FIGURE A33 — PROBLEM CHARACTERISTICS FOR <i>AP-20</i> .....	462
FIGURE A34 — PROBLEM CHARACTERISTICS FOR <i>AP-20</i> .....	463
FIGURE A35 — PROBLEM CHARACTERISTICS FOR <i>AP-21</i> .....	464
FIGURE A36 — PROBLEM CHARACTERISTICS FOR <i>AP-21</i> .....	465
FIGURE A37 — PROBLEM CHARACTERISTICS FOR AUXILIARY PROBLEM <i>PI</i> .....	467
FIGURE A38 — ILLUSTRATING THE <i>S_LINEAR</i> FUNCTION .....	468
FIGURE A39 — ILLUSTRATING THE <i>R_FLAT</i> FUNCTION .....	468
FIGURE A40 — ILLUSTRATING DEPENDENCIES CAUSED BY THE <i>INTERDEPEND</i> FUNCTION .....	469
FIGURE A41 — ILLUSTRATING THE <i>S_DECEPT</i> FUNCTION .....	469
FIGURE A42 — ILLUSTRATING EXAMPLE DEPENDENCIES CAUSED BY THE <i>NONSEP_SPECIAL</i> FUNCTION .....	470
FIGURE A43 — ILLUSTRATING AN EXAMPLE OF USING <i>S_MULTI</i> .....	471
FIGURE A44 — ILLUSTRATING <i>B_PARAM</i> .....	472
FIGURE A45 — COMPARATIVE CUMULATIVE TIME COSTS FOR DOMINATED TREES WITH AND WITHOUT CLEANING .....	477
FIGURE A46 — COMPARATIVE CUMULATIVE TIME COSTS FOR DOMINATED TREES WITH AND WITHOUT CLEANING .....	478
FIGURE A47 — COMPARATIVE CUMULATIVE TIME COSTS FOR DOMINATED TREES WITH AND WITHOUT CLEANING .....	479

---

FIGURE A48 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN HYPERVOLUME RESULTS .....	487
FIGURE A49 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN HYPERVOLUME RESULTS .....	488
FIGURE A50 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN HYPERVOLUME RESULTS .....	489
FIGURE A51 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN EPSILON RESULTS .....	490
FIGURE A52 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN EPSILON RESULTS .....	491
FIGURE A53 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN EPSILON RESULTS .....	492
FIGURE A54 — LOCATING DOMINATED NODES IN AN EXAMPLE MAK_TREE .....	495

## V LIST OF TABLES

TABLE 1 — TRADITIONAL PARETO-BASED APPROACHES TO EVOLUTIONARY MULTIOBJECTIVE OPTIMISATION .....	36
TABLE 2 - CONTEMPORARY APPROACHES TO PARETO-BASED EVOLUTIONARY MULTIOBJECTIVE EVOLUTIONARY ALGORITHMS .....	37
TABLE 3 — SUMMARISING THE NEW <i>AP</i> TEST SUITE.....	86
TABLE 4 — SPECIAL PROPERTIES OF ORDERED NON-DOMINATED BI-OBJECTIVE LISTS .....	99
TABLE 5 - INCORRECTLY LABELLED NON-DOMINATED SOLUTIONS .....	102
TABLE 6 — CROWDING INACCURACIES IN TRUNCATED ELITE SETS .....	107
TABLE 7 — STOPPING CONDITION PERFORMANCE IN TRUNCATED SETS.....	111
TABLE 8 — BIG-OH SPACE AND PERFORMANCE COMPLEXITY FOR EXAMINED DATA STRUCTURES .....	131
TABLE 9 — UNBOUNDED ELITE SET CHARACTERISTICS .....	133
TABLE 10 — AVERAGE TOTAL CUMULATIVE TIMES FOR ELITE ARCHIVING TECHNIQUES .....	141
TABLE 11 — AVERAGE TOTAL CUMULATIVE TIMES FOR ELITE ARCHIVING TECHNIQUES .....	145
TABLE 12 — AVERAGE TOTAL CUMULATIVE TIMES FOR ELITE ARCHIVING TECHNIQUES AFTER 150,000 EVALUATIONS .....	145
TABLE 13 — PRESENTATION SET SIZES USING THE BASIC TECHNIQUE .....	168
TABLE 14 — PRESENTATION SET SIZES USING THE MAK_PRESENTATION ALGORITHM .....	170
TABLE 15 — AVERAGE <i>Q</i> -VALUE PERFORMANCE OF PRESENTATIONAL ALGORITHMS .....	174
TABLE 16 — TIME-BASED PERFORMANCE OF ARCHIVES WITH COMPLETE DENSITY ESTIMATES.....	200
TABLE 17 — TIME-BASED PERFORMANCE OF ARCHIVES WITH COMPLETE DENSITY ESTIMATES.....	201
TABLE 18 — EMPIRICAL PERFORMANCE OF THE ANNOTATED MAK_TREE.....	215
TABLE 19 — EMPIRICAL PERFORMANCE OF THE FULLY-FEATURED MAK_TREE AND THE TRUNCATED CUBOID AFTER 10,000 NSGA-II EVALUATIONS.....	218
TABLE 20 — EMPIRICAL PERFORMANCE OF THE FULLY-FEATURED MAK_TREE AND THE TRUNCATED CUBOID AFTER 50,000 NSGA-II EVALUATIONS.....	219
TABLE 21 — EMPIRICAL PERFORMANCE OF THE FULLY-FEATURED MAK_TREE AND THE TRUNCATED CUBOID AFTER 150,000 NSGA-II EVALUATIONS.....	219
TABLE 22 — EMPIRICAL PERFORMANCE OF CELL-BASED MAK_TREES AND THE TRUNCATED CUBOID AFTER 10,000 NSGA-II EVALUATIONS .....	225
TABLE 23 — EMPIRICAL PERFORMANCE OF CELL-BASED MAK_TREES AND THE TRUNCATED CUBOID AFTER 50,000 NSGA-II EVALUATIONS .....	226
TABLE 24 — EMPIRICAL PERFORMANCE OF CELL-BASED MAK_TREES AND THE TRUNCATED CUBOID AFTER 150,000 NSGA-II EVALUATIONS .....	226
TABLE 25 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN HYPERVOLUME RESULTS .....	256
TABLE 26 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN EPSILON RESULTS .....	256
TABLE 27 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN HYPERVOLUME RESULTS .....	269

TABLE 28 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN EPSILON RESULTS.....	273
TABLE 29 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN HYPERVOLUME RESULTS.....	290
TABLE 30 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN EPSILON RESULTS.....	290
TABLE 31 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN HYPERVOLUME RESULTS.....	306
TABLE 32 — TWO-TAILED KRUSKAL-WALLIS TESTS ON END-OF-RUN EPSILON RESULTS.....	310
TABLE 33 - SUMMARISING KEY FINDINGS IN THE COMPARATIVE STUDY OF CONTEMPORARY EVOLUTIONARY ALGORITHMS.....	330
TABLE 34 - SUMMARISING KEY FINDINGS IN THE COMPARATIVE STUDY OF NOVEL AND CONTEMPORARY EVOLUTIONARY ALGORITHMS.....	359
TABLE 34 — NORMALISED HYPERVOLUME RANKS FOR EACH EXAMINED PROBLEM .....	382
TABLE 35 — NORMALISED EPSILON RANKS FOR EACH EXAMINED PROBLEM .....	382
TABLE A1 — KEY PARAMETER SETTINGS FOR NSGA-II .....	474
TABLE A2 — KEY PARAMETER SETTINGS FOR NSGA-II .....	474
TABLE A3 — STYLE GUIDE LISTING .....	481
TABLE A4 — SPECIAL NOTATIONS LISTING .....	481
TABLE A5 — TERMS AND OPERATORS LISTING .....	482
TABLE A6 — MAK_TREE PROPERTIES LISTING.....	483
TABLE A7 — EXTENDED MAK_TREE PROPERTIES LISTING .....	483
TABLE A8 — ACRONYMS (PART ONE) .....	484
TABLE A9 — ACRONYMS (PART TWO).....	485
TABLE A10 - ONLINE SOURCES FOR RED-BLACK TREE IMPLEMENTATIONS.....	494
TABLE A11 - FREQUENCY MATRIX DIMENSIONS.....	495

## VI LIST OF ALGORITHMS

ALGORITHM 1 — A BASIC SEARCH ALGORITHM .....	28
ALGORITHM 2 — INSERTION INTO THE MAK_TREE.....	123
ALGORITHM 3 — HANDLE_DOMINANCE_HELPER .....	123
ALGORITHM 4 — FORMING A BASIC PRESENTATION SET .....	168
ALGORITHM 5 — THE MAK_PRESENTATION ALGORITHM .....	169
ALGORITHM 6 — UPDATING NEAREST-NEIGHBOUR SCORES AFTER INSERTION.....	187
ALGORITHM 7 — UPDATING NEAREST-NEIGHBOUR SCORES AFTER DELETION .....	187
ALGORITHM 8 — UPDATING CUBOID SCORES AFTER INSERTION .....	188
ALGORITHM 9 — UPDATING CUBOID SCORES AFTER DELETION .....	189
ALGORITHM 10 — $\kappa$ NEAREST-NEIGHBOURS INSERTION: UPDATING THE NEW NODE.....	190
ALGORITHM 11 — $\kappa$ NEAREST-NEIGHBOURS INSERTION: UPDATING NEIGHBOURS.....	191
ALGORITHM 12 — $\kappa$ NEAREST-NEIGHBOURS DELETION .....	193
ALGORITHM 13 — $\kappa$ NEAREST-NEIGHBOURS DELETEFROMRIGHT (HELPER ALGORITHM).....	194
ALGORITHM 14 — UPDATING CROWDING ANNOTATIONS AFTER INSERTION .....	203
ALGORITHM 15 — ANNOTATE SUB-TREE (RECURSIVE HELPER FUNCTION).....	204
ALGORITHM 16 — ANNOTATION UPDATES AFTER DELETION .....	205
ALGORITHM 17 — THE BASIC PAES ALGORITHM.....	247
ALGORITHM 18 — THE PARETO ENVELOPE SELECTION ALGORITHM .....	261
ALGORITHM 19 — EVENLY SUB-DIVIDING THE OBJECTIVE-SPACE .....	262
ALGORITHM 20 — THE NSGA-II ALGORITHM .....	279
ALGORITHM 21 — THE SPEA2 ALGORITHM .....	296
ALGORITHM 22 — THE IBEA ALGORITHM.....	319
ALGORITHM 23 — THE DIVERSITY_PAES ALGORITHM.....	348
ALGORITHM 24 — THE MAK_OS ALGORITHM.....	350



# Chapter



## Introduction

*“I forget, I ever forget, that I  
have no wings to fly, that I  
am bound in this spot  
evermore.”*

*Rabindranath Tagore —  
Author*

*“There is no such thing as a  
long piece of work, except  
one that you dare not start.”*

*Charles Baudelaire — Poet*





# **1 INTRODUCTION**

There is something to be said about there being truth in clichés. Increasingly, the notion that “most real-world problems are multiobjective in nature” [1] has become a clichéd introduction into the power of multiobjective optimisation (see [2-10] for variations on the theme), not just because it is an impressive catch phrase — the notion that a field of study is directly applicable to *real* problems that affect *real* people is appealing indeed — but also because the research has borne out the truth of the sentiment. Consider the comprehensive study of Coello, Veldhuizen and Lamont [11]: they note over four hundred practical applications that were published prior to 2002. An investigation of more recent literature illustrates that the practical benefits of multiobjective optimisation are being realised in fields as diverse as airfoil design [12-18], water management [19-22], robotics [23-25], image processing [26-28], neural network optimisation [29-33], development of combinatorial and integrated circuits [34-36], cancer research [37, 38] and bicycle design [39].

As multiobjective optimisation has become a more powerful player in real-world design and optimisation tasks, the need for specialisation has come to the fore. In particular, outside of the crisply defined and controllable ivory tower of theoretical research, the need to address dynamic problem spaces, noisy function approximations and constrained domains has led to a host of specialist techniques. Such approaches endeavour to augment the core ideas of contemporary multiobjective research to better match the demands of the job which they are required to do (see, for instance, [10, 40-50]).

Interestingly however, despite the presence of single objective optimisation as a clearly delineated field and the fact that the “overwhelming majority” of studies focus on the bi-objective domain [51] (see [12, 13, 15, 18, 20, 21, 30, 47, 52-74] for contemporary examples), work relating to the specialisation of multiobjective research based on the number of objectives has been reasonably slim. This is particularly surprising given that recent results have shown that there is a marked variation in performance between low- and high-dimensional objective spaces [75-78].

There is therefore a valuable research question that has thus far been ignored. Given the prevalence of bi-objective optimisation in practical applications, the centrality of real-world problem solving to the field as a whole and the notion that the number of objectives strongly influences the performance of existing optimisers, does specialisation into the bi-objective domain represent a potential avenue for improving performance? Specifically, are there special properties that exist in the bi-objective domain that may be exploited and, if so, are such properties sufficiently powerful to make specialisation worthwhile?

Given the emergence of elite archiving as *the* common defining component of contemporary optimisers (as evidenced by their use in [79-91]) and the power that such elitism imparts [92-94], the exploitation of bi-objective properties to improve the efficiency and efficacy of such archiving offers an interesting way to assess the value of bi-objective specialisation. In particular, the naïve list-based techniques that are the implied standard for conventional elite archiving [95] and the generalist extensions that have been proposed as alternatives [95-97] are both insufficiently efficient to facilitate unbounded archives in practice, while the artificial binding of archive sizes can have severe ramifications with respect to decreasing quality in archival members [97, 98]. If bi-objective specialisation can facilitate efficient unbounded archiving and capitalise upon this to produce significantly better results than their bounded alternatives, then such work is surely a valuable endeavour. It is the investigation of this issue that will form the centrepiece of this thesis.

## 1.1 RESEARCH QUESTIONS

The primary research questions that this thesis seeks to address are:

- Is it feasible to create unbounded bi-objective archives?
- Does such archiving offer tangible performance improvements in the optimisation of bi-objective problems?
- Does access to an accurate and unbounded elite set impart benefits outside of the optimisation process itself?

## **1.2 KEY CONTRIBUTIONS**

This work offers the following key contributions to the field of multiobjective optimisation:

- The development and detailed analysis of a rich suite of test functions that explore a wide-range of important multiobjective problem characteristics. Each function is deliberately narrow in focus, concentrating on a particular problem feature (Chapter 5).
- Elucidation of unique properties in the special bi-objective domain, particularly relating to the characteristics of non-dominated sets (Chapter 6).
- Exploitation of the bi-objective properties to produce an efficient unbounded archiving strategy known as the Mak\_Tree. This includes a thorough description, theoretical analysis and empirical investigation of the methodology and comparisons with both common and contemporary bounded and unbounded alternatives (Chapter 6).
- Empirical investigation into the limitations of truncated archiving, specifically with respect to the quality of the stored solutions and the accuracy of diversity estimates (Chapter 6).
- The development of guidelines for autonomous and intuitive stopping criteria and the use of the basic Mak\_Tree in the creation of such criteria. Includes empirical analysis of the performance of the newly proposed Mak\_Terminator system (Chapter 7).
- The creation of presentational algorithms that are tasked with sampling the unbounded archive and creating small, evenly-distributed, sets that are appropriate for decision-maker analysis. Features empirical investigations of the new Mak\_Presentation technique and the more common bounded and unbounded clustering methodologies (Chapter 7).
- The extension of the basic Mak\_Tree to incorporate efficient diversity estimation, selection and the storage of cells. This includes an analysis of the

performance of the extended Mak\_Trees and comparison with the behaviour of contemporary tightly-bound archival systems (Chapter 8).

- The integration of unbounded Mak\_Tree archives into existing contemporary optimisation algorithms and a thorough analysis of performance across a range of test functions. Extended algorithms include the contemporary PAES, PESA, NSGA-II and SPEA2 systems, with active, hybrid and referential integration methodologies described (Chapter 9).
- A comprehensive, and statistically rigorous, analysis of a rich set of contemporary optimisers operating in a diverse set of well-defined bi-objective domains. Results suggest a general hierarchy of performance and indicate characteristics that maximise performance on individual problem features (Chapter 10).
- The development and investigation of novel unbounded bi-objective optimisation algorithms, each designed to address the limitations and weaknesses of existing techniques. This features rich analysis against existing contemporary truncated approaches and the newly developed unbounded extensions (Chapter 11 and Chapter 12).

### **1.3 OUTLINE OF THE THESIS**

The thesis is structured as follows:

- Chapter Two introduces multiobjective optimisation via description, example and strict mathematical definition, with a view to establishing the central concepts, motivations and ideas of the field.
- Chapter Three explores the problem characteristics that a real-valued multiobjective optimiser may encounter during a given run.
- Chapter Four develops a test suite (the *AP* collection) that effectively captures each of these characteristics and offers an in-depth analysis of both objective and decision spaces.
- Chapter Five defines the special properties of the bi-objective domain, examines the limitations of traditional truncated archiving and offers a new specialist

unbounded approach (the Mak\_Tree) that leverages the attributes of bi-objective non-dominated sets. The chapter also features comprehensive theoretical and empirical analyses of both unbounded and bounded techniques across many of the *AP* functions proposed in Chapter Four.

- Chapter Six capitalises on the attributes of the basic Mak\_Tree for the development of effective stopping criteria and set presentation mechanisms.
- Chapter Seven extends the basic Mak\_Tree to include efficient diversity estimation and selection procedures, with thorough timing analyses indicating the costs associated with such extension.
- Chapter Eight integrates the extended Mak\_Tree into a host of contemporary truncated systems, with differentiation in performance elucidated by the use of Pareto compliant indicators and visualisation tools.
- Chapter Nine offers a thorough analysis of contemporary truncated approaches in static real-valued bi-objective domains with a view to establishing which algorithmic characteristics are best suited to particular domain types.
- Chapter Ten addresses the limitations of existing techniques by developing novel unbounded algorithms and explores their performance against truncated contemporaries.
- Chapter Eleven draws together the results of the preceding three chapters to establish the best general performer of the examined optimisers.
- Chapter Twelve emphasises the key conclusions of this thesis and offers a summary of the many avenues of future work that exist for those interested in bi-objective optimisation.

# Chapter



## Introducing Evolutionary Multiobjective Optimisation

*“Compromise, if not the spice of life, is its solidity. It is what makes nations great and marriages happy.”*

*Phyllis McGinley — Author*

*“All government, indeed every human benefit and enjoyment, every virtue, and every prudent act is founded on compromise...”*

*Edmund Burke — British Statesmen*

*“If you limit your choices only to what seems possible or reasonable, you disconnect yourself from what you truly want, and all that is left is a compromise.”*

*Robert Fritz — Composer, Film Maker and Author*

*“Compromise is the Devil.”*

*Anonymous*

## **2 WHAT IS MULTIOBJECTIVE OPTIMISATION?**

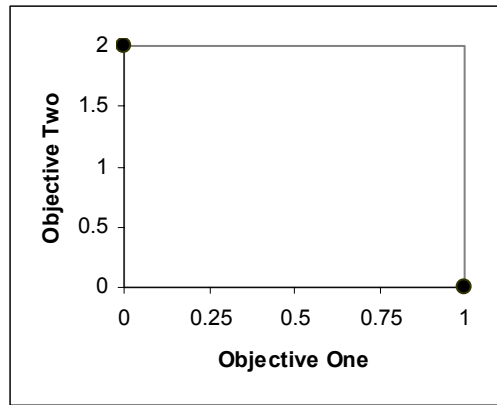
At its core, multiobjective optimisation is fundamentally about compromise. Where single objective approaches such as traditional Genetic Algorithms (GAs) are charged with finding the single best available solution, multiobjective optimisers strive to develop well-distributed sets of ideas that represent the best possible trade-offs. Indeed, in a multiobjective context, finding only one solution, even if it is considered optimal, is viewed as a failure<sup>1</sup> — the driving concern lies not just with utility, but also with the breadth of choice. Such principles are an indelible side effect of addressing conflicting goals where, as in life, it is rarely the case that a single approach can be viewed as ideal when considered from multiple perspectives. It is also for this reason that multiobjective optimisation has been drawing increasing attention from designers and engineers (see [11] for an impressive survey) — they deal in multi-faceted real-world problems that, in all likelihood, lack a grand unifying solution. Of more value to such work is the creation of a rich palette of ideas that represent a more accurate picture of the best choices available, upon which an understanding of the problem at hand can be developed.

Thus, compromise can be considered as the key distinguishing and driving factor of multiobjective optimisation — but what does compromise mean from a theoretical standpoint? Based on practical knowledge of the word, compromise infers a balancing of distinct objectives — typically with the goal of finding points where all parties involved are satisfied (at least to some extent). In multiobjective optimisation however, the concern is not just with balanced compromises such as these, but with finding a full range of trade-offs, incorporating those that give rise to mutual satisfaction, those that are heavily biased and most-everything in-between. As an example, consider Figure 1. Figure 1a represents the type of biased, extrema solutions that could be expected from performing two single-objective runs — they describe little about the interaction between objectives, but offer some insight into the optimal performance for each case. Figure 1b offers only balanced-compromise solutions — they speak volumes about how objectives can be integrated for mutual

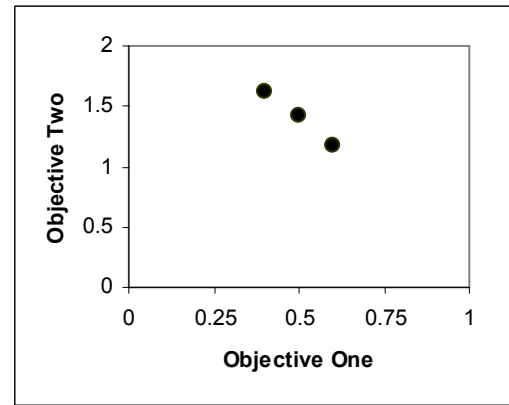
---

<sup>1</sup> Assuming the objectives are in conflict and non-degenerative (Section 2.2.4 and Section 4.2.8).

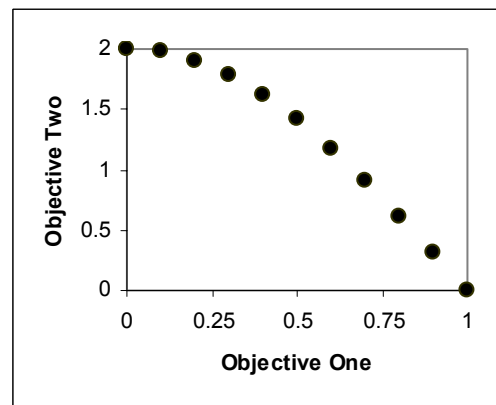




(a) Extrema Solutions



(b) Mutually Beneficial Solutions



(c) A Rich Set of Compromises

**Figure 1 — Different Compromise Solutions**

(a) Extrema solutions — typical of two distinct single-objective runs. (b) Mutually beneficial compromise solutions. (c) Well-distributed compromise solutions, typical of an effective multiobjective optimisation run.

gain, but fail to elucidate more general properties about the interaction between the problems, particularly where one objective is given greater priority than another. Figure 1c is representative of the set of compromises that multiobjective optimisation ultimately strives for — it is rich in the sense that the solutions provided are not only diverse, but illustrate more about the nature of the problem at hand than the extrema or balanced solutions alone could ever hope to.

Of course, having a well-distributed and wide-ranging set of compromises means little if the solutions are not considered to be *good compromises*. It may be argued that removing the landing-gear from an aeroplane is a good way to compromise between construction costs and safety (with a business-minded preference for cost), but it is likely that there exist better alternatives — ones which result in craft that are both less costly and safer. As such, it is important that distinctions can be drawn

between those solutions that are clearly better or worse than one another, lest the optimisation process be driven by diversity alone. Fortunately, the utility of compromises can be deduced via the principal of Pareto optimality.

## 2.1 INTRODUCING PARETO OPTIMALITY

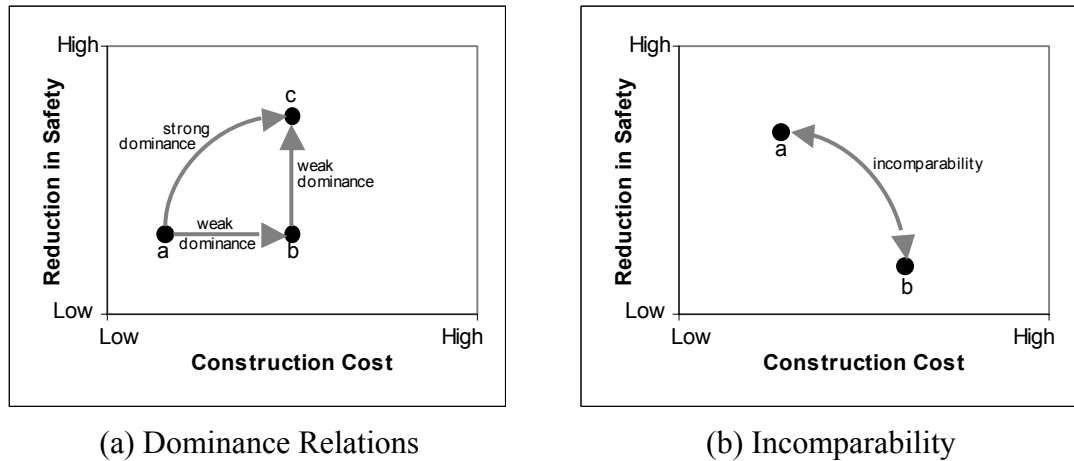
While the intricacies of Pareto optimality will be given a more formal treatment later (see Section 2.2), this section will introduce some of the key terms and concepts central to the field of multiobjective optimisation without recourse to strict mathematical definition. It is important to first gain an understanding of the motivation for the ideas, structures and dependencies inherent in Pareto optimality prior to elucidating the finer points of putting those ideas into practice.

Pareto optimality<sup>2</sup> [99] is predicated on the notion of Pareto dominance, whereby two solutions can be differentiated according to apparent worth. In this case, a solution is said to *strongly (strictly) dominate* another solution if it is better in all objectives of the problem, while *weak dominance* relaxes the condition to allow equality in those objectives where a solution is not strictly better, so long as it *is* better in at least one objective (see Figure 2 for examples). Using these simple definitions alone, it is now possible to distinguish between a poor compromise solution and an improved one. Returning to the wheelless aeroplane example, it can now be said that any alternative solution that offers both a less-expensive and safer option is fundamentally a better choice, since it will be strongly dominant with respect to our poorly designed craft (see solutions *a* and *c* in Figure 2a).

While the dominance relation provides a fundamentally important step towards measuring optimality in a multiobjective environment, it alone is not enough. Consider an alternative to the rough-landing aeroplane that features improved safety, but only at a significantly increased cost (see Figure 2b). In this case, neither solution dominates the other and it is not necessarily clear which alternative is preferable. These solutions are *incomparable* — they each form distinct trade-offs

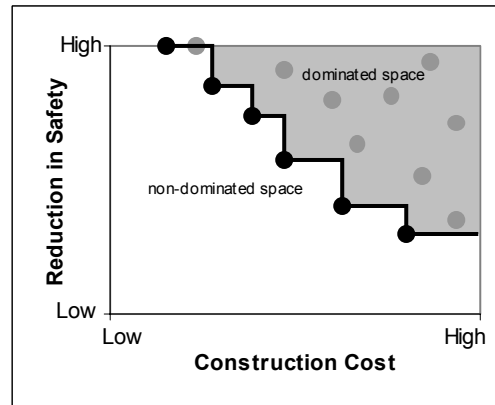
---

<sup>2</sup> Vilfredo Pareto (1848-1923) was an Italian economist, engineer and sociologist. In time, Pareto optimality became synonymous in economics with a societal equilibrium where no single person or entity can be made more wealthy without sacrificing the well-being of another. More recently, the term has gained popularity in game theory, operations research and multiobjective optimisation.



**Figure 2 — Dominance and Incomparability**

(a) Solution *a* weakly dominates solution *b* and strongly (strictly) dominates solution *c*; solution *b* weakly dominates solution *c*. (b) Solutions *a* and *b* are incomparable with each other. All graphs assume objectives are to be minimised.



**Figure 3 — Pareto Fronts**

The black circles represent the best compromise proposals in the population — they are members of the best estimate of the Pareto front. If the black circles represent the best possible solutions, then they are members of the true Pareto optimal front. The shaded region represents space that is dominated by the front — proposals residing in this region are considered sub-optimal.

which satisfy a different balance of the objectives being addressed. Given that multiobjective optimisation is tasked with finding good, well-distributed, compromises, it seems beneficial to view non-dominated incomparable solutions as useful in a multiobjective context.

Thus, using only the dominance and incomparability relations, it is possible to take a population of solutions and extract the best compromises — those that (at least) weakly dominate, or are incomparable with, every other member of the population (see Figure 3). In multiobjective literature, this set of solutions is generally referred to as the *non-dominated front* and is considered optimal with respect to the

population of solutions from which the set is extracted. If the population includes every possible solution, then the extracted set is considered globally optimal and forms the *Pareto optimal front* — notionally the goal of all multiobjective optimisers. In practice, however, where most optimal sets are very large (tending towards infinity for continuous problems), such a goal is often infeasible [100]. Thus, the practical target for all multiobjective optimisers is the formation of a good approximation of the Pareto optimal set — one that satisfies the need for a rich population of solutions that lie near to the optimal front, but are also well distributed along that front [101-104].

## 2.2 FORMALLY DEFINING SOME KEY TERMS IN MULTIOBJECTIVE OPTIMISATION

Thus far, the corner stones of multiobjective optimisation have been addressed with a degree of abstraction. With the context and motivation for the central ideas established, it is now appropriate to move from the abstract to the concrete via a more formal discussion.

### 2.2.1 DEFINITION OF A MULTIOBJECTIVE SOLUTION AND SOLUTION-SPACE

A *solution* (*decision vector*, *compromise* or *proposal*) to a multiobjective problem is taken to be a list of *values* (*parameters* or *decision variables*) that, when applied to an objective in the problem, give rise to a single score representing performance on that objective. In general, a solution  $\mathbf{x}$  has  $m$  decision variables:

$$\mathbf{x} = (x_1, x_2, \dots, x_m) \quad (1)$$

Though each decision variable may take on any alphabet, this work will focus on constrained real-valued domains:

$$x_i \in \mathbb{R} : ((Min_i \leq x_i \leq Max_i) \wedge (-\infty \ll Min_i < Max_i \ll \infty)) \forall i \in \{1, 2, \dots, m\} \quad (2)$$

where *Min* and *Max* settings define the range of values for decision variables.

The resultant *decision* (*solution* or *search*) *space*  $Y$  represents the  $m$  dimensional area of search available for a given multiobjective problem and is the set of all possible solution vectors:

$$Y = \{(x_1, x_2, \dots, x_m) \mid \forall x_{i=1..m} \in \mathbb{R} : \text{Min}_i \leq x_i \leq \text{Max}_i\} \quad (3)$$

### 2.2.2 DEFINITION OF A MULTIOBJECTIVE PROBLEM

A *multiobjective problem* is any problem that consists of two or more optimisation objectives that are subject to some form of constraint. In particular, all multiobjective problems can be given as:

$$\text{Minimise } f_{1..\beta}(\mathbf{x}) \text{ subject to } (\forall x_i \in \mathbf{x}, x_i \in C_i) \quad (4)$$

where  $\beta$  is the number of objectives being optimised;  $\mathbf{x}$  is a multi-variable solution; and  $C_i$  is the constraint set of allowable values for the  $i^{\text{th}}$  solution variable. Note that all optimisation objectives are arbitrarily categorised as minimisation functions — this results in no loss of generality, as all maximisation problems can be presented as minimisation functions via simple negation [105]:

$$\text{Minimise } (f_i(\mathbf{x})) = -\text{Maximise } (-f_i(\mathbf{x})) \quad (5)$$

### 2.2.3 DEFINITION OF OBJECTIVE-SPACE

The *fitness (objective) space*  $F$  of a problem is the mapping of all possible solutions in  $Y$  to the  $\beta$  dimensional set of results formed by applying each solution to each of the  $\beta$  objectives:

$$F = \{(f_1(\mathbf{y}), f_2(\mathbf{y}), \dots, f_\beta(\mathbf{y})) \mid \forall \mathbf{y} \in Y\} \quad (6)$$

It is important to note here that there is not necessarily a strict correspondence between trends seen in the solution-space and those seen in the fitness space — indeed, it is possible for entirely continuous search spaces to give rise to discontinuous objective-spaces, while solutions which are closely mapped in fitness space, may be well separated in the solution-space. Such a property has profound ramifications when it comes to searching for optimal solutions, as shall be seen later (Chapter 4).

### 2.2.4 DEFINITION OF A CONFLICTING MULTIOBJECTIVE PROBLEM

For the purposes of this work, the primary focus is on the subset of multiobjective problems that feature *conflict* — that is to say, those problems where there exists at

least some portion of fitness space where progress towards the optimal of one objective leads to movement away from the optimal of another objective:

$$\exists \mathbf{a}, \mathbf{b} \in Y; i, j \in \{1, 2, \dots, \beta\} : \left( (f_i(\mathbf{a}) < f_i(\mathbf{b})) \wedge (f_j(\mathbf{a}) > f_j(\mathbf{b})) \right) \quad (7)$$

Those problems that fail to meet these criteria can be more readily solved using single objective optimisation via simple reduction of objectives (since all solutions will tend to have approximately mutual utility across objectives) or through the merging of objectives.

### 2.2.5 DEFINITION OF PARETO DOMINANCE

Recalling that a multiobjective problem can always be defined in terms of a minimisation function, solution  $\mathbf{a}$  is said to strongly (strictly) dominate solution  $\mathbf{b}$  if and only if:

$$f_i(\mathbf{a}) < f_i(\mathbf{b}) \forall i \in \{1, 2, \dots, \beta\} \quad (8)$$

This condition can be relaxed to allow for weak dominance, whereby solution  $\mathbf{a}$  is said to weakly dominate solution  $\mathbf{b}$  if and only if:

$$(f_i(\mathbf{a}) \leq f_i(\mathbf{b}) \forall i \in \{1, 2, \dots, \beta\}) \wedge (\exists j \in \{1, 2, \dots, \beta\} : f_j(\mathbf{a}) < f_j(\mathbf{b})) \quad (9)$$

If solution  $\mathbf{a}$  strongly dominates solution  $\mathbf{b}$ , then it is written as:

$$\mathbf{a} \prec \mathbf{b} \quad (10)$$

If the relation between solutions is weak dominance then the correspondence is expressed as:

$$\mathbf{a} \preceq \mathbf{b} \quad (11)$$

It is worth noting that, in the strictest sense, Pareto relationships do not exist between solutions, they exist between the objective-space projections of those solutions. For brevity, when this work speaks of dominance, incomparability or equality between solutions, unless explicitly stated otherwise, it is their position in objective-space that is being addressed.

### 2.2.6 DEFINITION OF EQUALITY

Any two solutions  $\mathbf{a}$  and  $\mathbf{b}$  are *equal* (in objective-space) if and only if:

$$f_i(\mathbf{a}) = f_i(\mathbf{b}) \forall i \in \{1, 2, \dots, \beta\} \quad (12)$$

For brevity, such equality between  $\mathbf{a}$  and  $\mathbf{b}$  is written as:

$$\mathbf{a} = \mathbf{b} \quad (13)$$

### 2.2.7 DEFINITION OF INCOMPARABILITY

Any two solutions  $\mathbf{a}$  and  $\mathbf{b}$  are considered *incomparable* if and only if:

$$(\exists i \in \{1, 2, \dots, \beta\} : f_i(\mathbf{a}) < f_i(\mathbf{b})) \wedge (\exists j \in \{1, 2, \dots, \beta\} : f_j(\mathbf{a}) > f_j(\mathbf{b})) \quad (14)$$

Or, alternatively, any solution  $\mathbf{a}$  that is not weakly dominant or weakly dominated with respect to solution  $\mathbf{b}$ , must be incomparable with  $\mathbf{b}$  if they are not equal. Since the relation is transitive,  $\mathbf{b}$  must also be incomparable with  $\mathbf{a}$ .

In general, if solution  $\mathbf{a}$  is incomparable with solution  $\mathbf{b}$ , then it is written as:

$$\mathbf{a} \sim \mathbf{b} \quad (15)$$

### 2.2.8 DEFINITION OF PARETO FRONTS AND PARETO OPTIMALITY

A *Pareto estimate front* ( $P_{estimate}$ ) is composed of those members of a solution set  $S$  that are non-dominated with respect to all constituents of  $S$ :

$$P_{estimate} \subseteq S \subseteq Y : \forall \mathbf{p} \in P_{estimate}, ((\mathbf{p} \preceq \mathbf{s}) \vee (\mathbf{p} \sim \mathbf{s}) \vee (\mathbf{p} = \mathbf{s})) \forall \mathbf{s} \in S \quad (16)$$

If  $S$  enumerates the complete decision-space  $Y$ , then the estimate is the *true* (complete) *Pareto optimal set* and is defined as  $P_{front}$ :

$$(P_{estimate} = P_{front}) \text{ iff } (S = Y) \quad (17)$$

If  $S$  represents all solutions produced thus far, then  $P_{estimate}$  represents the collection of non-dominated proposals discovered throughout the optimisation process and is referred to as the *local optimal set* ( $P_{local}$ ).

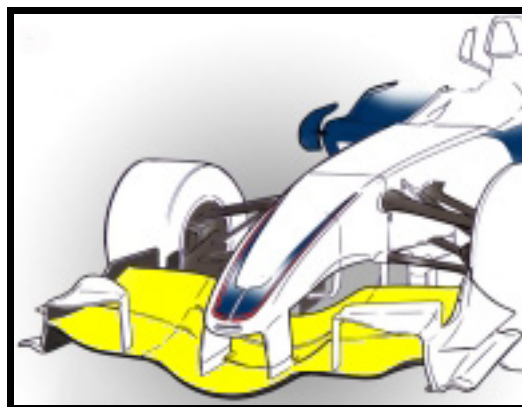
## **2.3 AN ILLUSTRATIVE EXAMPLE**

While the particulars of multiobjective optimisation are detailed through the theoretical definitions provided in the previous section, the utility of these ideas are best demonstrated through an illustrative example. As such, this section will consider a complex, though approximated, real-world design problem that multiobjective optimisation is ideally suited to.

### **2.3.1 FORMULA ONE FRONT WING DESIGN**

The design of a Formula One (F1) car is an inherently difficult proposition (as indicated by research and development costs that were remarkably estimated at over \$250,000,000 for the 2006 season [106]). This complexity is largely contingent on the interconnectedness of the various modules that must be constructed for any car — particularly with respect to aerodynamic performance, which can be fundamentally affected even with relatively minor changes to body shape. As such, the design of a Formula One front wing (see Figure 4) — which is charged with controlling airflow around and under the car — is of paramount importance to the success of the vehicle.

In general, the front wing is used to channel air in such a way as to generate negative lift and thus provide traction for the car as it races around the circuit. Such is the extent of the aerodynamic grip generated by contemporary open-wheel vehicles that they can race inverted on the ceiling of a tunnel and can pull manhole covers from



**Figure 4 — An Illustration of the Front Quarter of the Sauber BMW F107**

The highlighted yellow construct (attached to the nose cone of the vehicle) is the front wing. The sketch is produced and provided by Craig Scarborough, Technical Writer, Autosport.com [107].



the road<sup>3</sup>. The front wing — together with the rear wing — is responsible for producing much of this impressive downforce, without which high-speed cornering would be practically impossible.

However, there are seldom free lunches provided in engineering design and aerodynamic grip comes with a price befitting its utility. In general, greater downforce infers higher aerodynamic drag, which can severely inhibit the overall speed of a car — particularly on straights, where high maximum speeds are of pivotal importance.

Thus, the design of an effective front wing for a Formula One car is almost always a compromise between the cornering ability provided by impressive grip and the raw speed enabled by a low-drag configuration. More problematic still is the fact that there exists no single compromise that is ideally suited to all tracks — Monaco may be laden with tight corners and demand high downforce at all costs, but the same cannot be said of the gentle curves of Montreal. As such, multiobjective optimisation seems ideally suited. A comprehensive set of diverse compromise solutions can be provided for the team that can then be filtered according to the unique priorities of each race.

### **2.3.2 DEFINING THE MULTIOBJECTIVE PROBLEM AND SOLUTION STRUCTURE**

For simplicity, the design of a Formula One wing can be specified as a conflicting bi-objective task where the aim is to minimise the loss of aerodynamic grip and the amount of aerodynamic friction (drag). In reality, the performance of each objective would be measured either through the application of complex aerodynamic formulas and models or the practical testing of proposed solutions in a wind-tunnel. For now though, it is enough to assume that there exists some way of mapping a solution into results that represent performance on each of the objectives.

While the construction of a front wing is actually the product of a vast number of subtle variables, this approximation will limit the design to three key performance factors — namely, the position, angle of incidence and width of the front wing. These parameters are constrained by the technical regulations of the sport (see

---

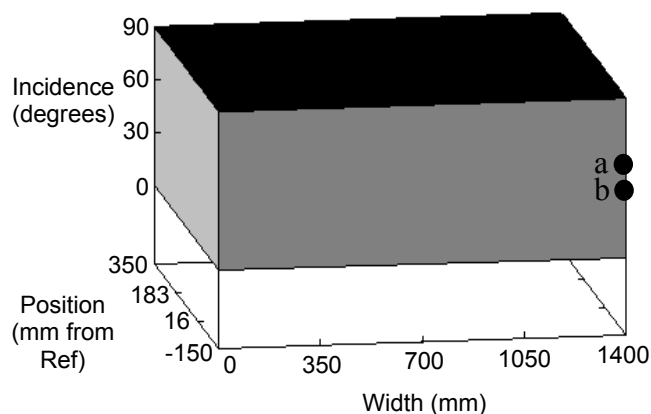
<sup>3</sup> Somewhat humorously, race stewards must weld down the man-hole covers on street circuits prior to proceedings to guard against just such a predicament.

[108]). With respect to the 2006 season, the width of the wing may be no more than 1400mm in diameter (regulation 3.4.1), while the position of the wing (regulation 3.7.1) may be no lower than 150mm below the reference plane (the agreed centre plane of the car) and no higher than 350mm above it. For practicality, the angle of incidence must lie between zero and ninety degrees (relative to the reference plane).

### 2.3.3 EXAMINING SOLUTION AND FITNESS SPACES

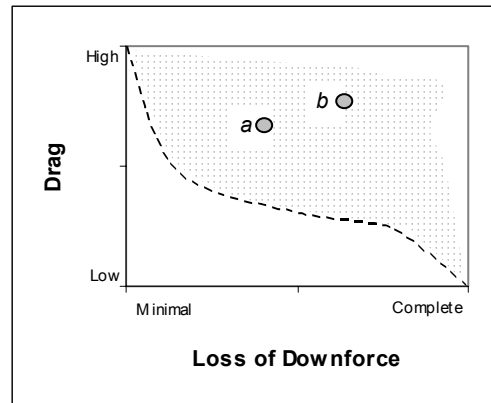
With the extent of each variable established, it is possible to define the search space from which the set of compromise solutions will ultimately come. Recalling that real-valued solutions give rise to a continuous decision-space and that the dimensionality of that space corresponds to the number of variables being optimised, Figure 5 illustrates every possible solution to the front-wing problem.

With the solution-space fully specified, it is now theoretically possible to define the two-dimensional objective-space by mapping every possible solution to each objective (see Figure 6). Note that, in practice, such a brute-force examination is typically infeasible due to the costs associated with each function evaluation and the size of the solution set (which is infinite if any variable is real-valued). Indeed, given the extreme expense of wind-tunnel testing, the slow production of results via aerodynamic models and the real-valued nature of the decision variables, a complete realisation of the front-wing objective-space is practically impossible. Thus, the aim



**Figure 5 — Decision-space for a Simplified F1 Wing**

The shaded region represents the continuous space that houses every feasible configuration of an F1 wing. The boundaries of the space are formed by the minimum and maximum values for each variable. Points *a* and *b* correspond to points *a* and *b* in the objective-space (see Figure 6).



**Figure 6 — Hypothetical Fitness Space for a Simplified Formula 1 Wing**

The collection of points represents the location of all possible results for the bi-objective problem, found by mapping every feasible solution to the functions for drag and downforce. The dashed line represents the continuous Pareto optimal front. Points *a* and *b* correspond to solutions *a* and *b* in Figure 5.

of multiobjective optimisation (both in general and with respect to this example) is not to fully represent the objective-space, but to move as rapidly as possible through the space and towards a set of the best compromise solutions — be it through the application of heuristics or artificial intelligence principles.

### 2.3.4 MOVING TOWARDS THE PARETO OPTIMAL FRONT

While the machinations of multiobjective search will be discussed at length in later sections (see Chapter 3 and Chapters 9–12), it is sufficient to assume that most of the power for progressing through the objective-space rapidly is grounded in the principles of Pareto optimality. Indeed, the most obvious way to form an impressive estimate of a Pareto front is to bias the search around well-spread non-dominated solutions in the belief that they will continue to lead to ever better compromises. This basic tenet still lies at the heart of almost all contemporary multiobjective search algorithms.

In the case of the wing design example, using the principles of Pareto optimality will discourage search around areas of poor downforce and high drag configurations (since these are easily dominated) and push exploration towards settings that result in lower drag and better downforce (since these are more likely to dominate other proposals). However, there is a danger in over-simplifying things here. The notion that picking continually better solutions will inevitably result in a well-distributed and accurate estimate of the Pareto optimal front is predicated on a certain

predictability in the objective- and search-spaces. Unfortunately, there is no guarantee that proximity in objective-space has any correlation with proximity in search-space (or *vice-versa*). As such, locating a highly dominant solution may be beneficial, but it may also represent an evolutionary dead-end: a single isolated solution that shares no commonality with any other optimal solution.

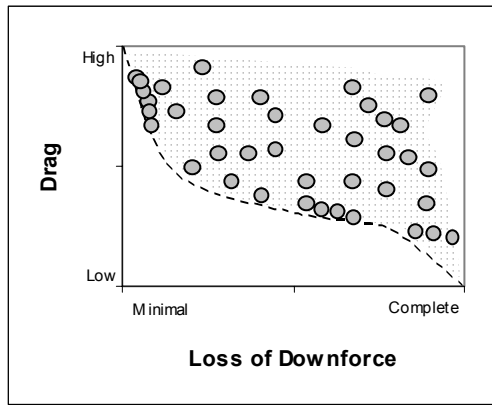
Such a poor accord between fitness- and solution-spaces certainly exists in front wing design and is best illustrated with an example pit stop. Consider a driver that has a well performing front wing, but believes that a subtle modification will result in the extra traction required to instigate a passing manoeuvre under braking. During the stop, the pit crew are instructed to make a minor adjustment to the wing's angle of incidence (a change from  $a$  to  $b$  in Figure 5), believing that such a small variation will potentially make some positive difference, but at worst should cause little damage to overall performance. On exit the car vibrates wildly, loses traction and spins off the track. The pit crew, in this case, failed to recognise that a small change in solution-space positioning need not necessarily incur a small change in objective-performance (note the difference in optimality between  $a$  and  $b$  in the objective-space represented by Figure 6).

If a multiobjective problem is overwhelmingly composed of these types of discontinuities between decision- and objective-spaces, then it becomes practically impossible to successfully apply any intelligent search algorithm. Picking one solution over another is arbitrary, because neither is more likely to aid in the location of the optimal front. Thus, in such circumstances, a random exploration is likely to perform as well as a carefully refined dominance-based approach.

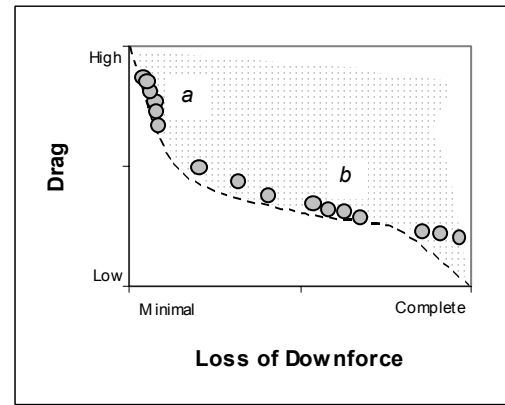
Fortunately for multiobjective researchers then, most real-world problems are not such thorny propositions. While there may be elements of the decision-spaces that behave in an apparently erratic manner, it is very rarely the case for the entirety of a space. As such, a multiobjective optimiser must be able to capitalise on the power of dominance-guided search in regions of space where such exploration is beneficial, but must also be robust enough to move beyond problematic areas where an intelligent search may otherwise degenerate.

### 2.3.5 TERMINATION AND PRESENTATION

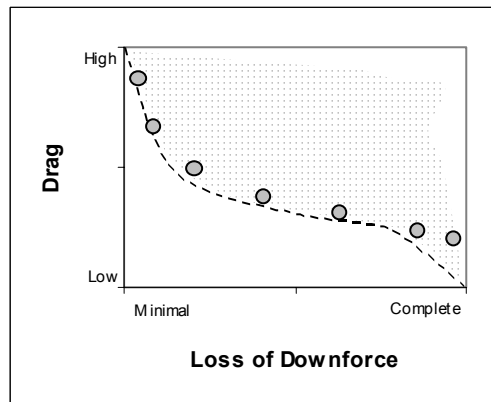
Given that the optimisation of the front wing is a real-valued problem and is one which carries expensive time and cost overheads, it is unreasonable to anticipate the output of the complete Pareto optimal front. It is therefore necessary to agree upon some set of stopping criteria that resolves the trade-off between solution quality and the expense of proposal evaluation. As will be explored later (Section 6.2.1.4 and Section 7.1), the definition of such termination marks is non-trivial, but for now it is enough to assume that the assigned criteria resulted in the search terminating after it discovered the population of solutions described in Figure 7a. Assuming that the designer is interested only in the best trade-offs discovered (which seems likely), the



(a) The Complete Collection of Discovered Solutions



(b) The Non-dominated Portion of the Complete Collection ( $P_{local}$ )



(c) The Reduced Presentational Set Derived from the Non-dominated Portion of the Complete Collection

**Figure 7 — End-of-Run Sets**

The complete population of solutions in (a) is reduced to the non-dominated set in (b). The largely unbiased presentational set in (c) is distilled from (b).

solution set serves a more practical purpose if it is reduced to contain only the non-dominated members ( $P_{local}$  — as illustrated in Figure 7b).

While having access to the complete non-dominated set at the completion of a run may be beneficial, in practice a reduced presentational set is generally preferred. In the case of front wing design this is particularly true — the expense and complexity of transforming models into full-scale mock-ups for more comprehensive on-car testing precludes the use of a large number of solutions (even if each option *is* Pareto optimal). The key then is to distil the terminated frontal set into an evenly distributed collection that expresses the range of available optimal trade-offs between downforce and aerodynamic friction. Again, the formation of appropriate presentation sets is not as straightforward as it may appear, and will be discussed at length in coming sections (see Section 7.2), but it is enough for now to assume that Figure 7c represents a suitable distillation of the complete non-dominated set exemplified in Figure 7a. This minimised collection is sufficiently rich to provide scope for improving performance under varying track conditions without favouring particular regions of the objective-space biased by the search (such as  $a$  and  $b$  in Figure 7b).

### **2.3.6 CONCLUDING REMARKS ON THE ILLUSTRATIVE EXAMPLE**

The application of multiobjective optimisation to the area of wing design in Formula One cars is an interesting hypothetical scenario. While the presented example is both simplified and largely contrived, it is representative of a host of real-world studies that illustrate the value of multiobjective optimisation in practical engineering tasks (see [12, 13, 34, 55, 59, 65, 74, 109-116] for a small sampling) and draws into focus some of the more central issues in multiobjective research.

For this particular study, the aim is to develop a concise set of promising trade-offs that provide the design engineer with not only a range of options tailored to a host of race scenarios, but an insight into the relationship between the decision variables and the nature of the objective-space. To achieve this goal, the use of intelligent Pareto-guided search procedures that capitalise on the implied accord between decision- and objective-spaces is appropriate. Such a procedure is likely to be considerably more efficient than enumeration or random search of the solution-space, while the use of presentational algorithms ensures that regions of the non-dominated objective-space

are clearly delineated. Given the monetary and time costs associated with wing design, the potential improvements offered by multiobjective optimisation with respect to efficacy and efficiency appear significant (particularly if appropriate stopping criteria can be formed).

## **2.4 MULTIOBJECTIVE AND SINGLE-OBJECTIVE OPTIMISATION**

Thus far, comparison between multiobjective search and its single objective forbearer has been relatively superficial, with the assertion that complex problems are better addressed from a multiobjective viewpoint since the resulting solutions will provide a rich understanding of the problem at hand. But is such a diverse solution set actually beneficial for the majority of problems? Can the more efficient and focussed single objective procedure give rise to solutions that are of greater utility to the designer?

### **2.4.1 THE ARGUMENT FOR SINGLE-OBJECTIVE OPTIMISATION**

Before the popularisation of multiobjective optimisation, problems with multiple goals were typically simplified through reduction (scalarisation) [4] — applying domain knowledge and preference information to combine objectives into a single aim (through objective-weighting, distance functions, min-max techniques, constraint methods or goal programming, for instance — see [4, 5, 117, 118] for summaries). To a surprising extent, this procedure remains popular [10], even when confronting potentially complex problems (see, for instance, [119-123]). While it could be argued that the proliferation of single-objective techniques in these areas can be attributed to the relative youth of the multiobjective field, it is more likely that scalarisation carries some benefit into these situations. In particular, searching for an optimal single-objective solution is an inherently less complex proposition than locating a well-distributed approximation of the Pareto optimal front and leads to a considerable increase in search-time efficiency. Moreover, many designers have very specific expectations of a system and are more concerned with achieving solutions that match their particular needs than examining a range of interesting alternatives. In this case, the additional overhead of a multiobjective system seems superfluous.

## 2.4.2 REJECTING THE ARGUMENT FOR SINGLE-OBJECTIVE OPTIMISATION

While the potential benefits inherent in scalarisation are alluring, it is ultimately too restrictive in comparison to the power and flexibility of multiobjective search.

### 2.4.2.1 *A PRIORI REQUIREMENTS*

While in principle the notion of reducing objectives according to pre-defined domain knowledge sounds straight-forward, the reality is that *a priori* requirements impose severe restrictions on the type of problems applicable for single objective reduction. In order to merge objectives, the domain expert must be both acutely aware of the preferences afforded to each objective (which is often impossible [124] and at least non-trivial [125]), how such preferences are to be modelled and how each objective functions (which is typically “a very expensive process” [5]). Moreover, since single-objective search will generally only produce a single optimal solution, minor errors in preference articulation or modelling can lead to results that fail to correctly map to the needs of the user. Worse still, since the single optimal solution lacks the context that is provided by a rich Pareto front estimation, such variations are difficult to identify and may lead to the acceptance of sub-optimal solutions.

While the cost of human error in *a priori* objective filtering is high, there exist more subtle ramifications in relying on hard preferences prior to a search. By biasing the process according to pre-conceived ideas about the nature of useful solutions and the properties of the decision-space, the search process is fundamentally constricted. Although the resultant solution may be optimal according to the expectations of the user, there is no potential for branching out beyond those expectations. Consider a design where great priority is placed on production cost and very low priority is placed on environmental impact — in this case, potentially fantastic ideas that result in minimal environmental damage at only a slight increase in monetary cost may be rejected due to sub-optimality. Since multiobjective optimisation illustrates a host of wide ranging solutions, it offers the scope to question previously held expectations and capture unexpected, but potentially useful, outcomes.

The need for *a priori* information also severely inhibits the use of single-objective scalarisation in exploratory design, where the aim is to develop an understanding of the problem at hand. Since express knowledge of the domain is typically minimal in



these circumstances, multiobjective optimisation, which requires very little *a priori* problem analysis (aside from that which is required to formulate the objectives), is ultimately better suited.

#### **2.4.2.2 *A POSTERIORI* ISSUES**

While scalarisation is inherently contingent upon sufficient *a priori* information, multiobjective search is similarly bound by the need for *a posteriori* solution filtering. Given that the extraction of a final solution from the set of compromises is essentially based on preference information (the end-user selects the solution they most prefer from the non-dominated set), it seems reasonable to assume that many of the criticisms levelled against preference articulation in single-objective search can be equally applied to the multiobjective case.

The key difference is context. The articulation of preferences in multiobjective optimisation is no longer based on the relatively abstract notions of objectives, but rather on a finite list of actual solutions. Consider purchasing a new home — in the *a priori* case it may be necessary to rank the relative importance of size, outlook, interior design and number of bedrooms; in the *a posteriori* case the buyer can make a selection based on visiting each property. It is also important to note that any error made in the *a posteriori* articulation of preferences simply requires rearticulation — in the *a priori* case, this can require both a complete re-structuring of the problem and a new search.

#### **2.4.2.3 *UNSTABLE USER NEEDS***

As illustrated in the Formula One wing design example (see Section 2.3), the utility of a multiobjective optimiser is particularly useful in an environment where the needs of a designer are dynamic — ever changing according to the situation at hand. Unlike a scalarised single objective search, a diverse set of compromises allows for the preferences of a designer to change without the need for a new search of the solution-space or a re-specification of the problem. Such disparity is particularly significant in time-sensitive systems where additional optimisation runs may be infeasible or undesired.

#### **2.4.2.4 DESCRIPTIVENESS**

Single-objective search is ultimately a mechanism that focuses on the development of a singular optimal idea — it does little to elucidate the relationships that exist between objectives, offer depth in a solution set or illustrate the context of a proposal. While such focus allows for rapid movement through the search-space, it lacks the descriptive value of multiobjective search, which is particularly powerful in detailing the interaction between objectives. Since most solutions produced by an optimiser will be adjusted, modified or experimented with in some way after the completion of a run, the additional information provided by multiobjective optimisation can guide this development process and potentially save both time and expense.

## **2.5 CONCLUSIONS**

This chapter has explored the central concepts and motivations behind contemporary multiobjective optimisation theory and application. In particular, it described and detailed the importance of the Pareto relations and how these may be harnessed to partially order the objective-space. When used in conjunction with incomparability, Pareto dominance provide the means for defining both locally optimal trade-offs, the ideal target set and a mechanism for building intelligent search algorithms. Most importantly of all, the Pareto relations offer a way of defining solution preferability without an explicit need for *a priori* information and the resultant non-dominated sets provide a range of contextualised, descriptive and flexible compromise proposals.

# Chapter



## How to Perform Multiobjective Optimisation

*“Have a bias towards action  
— let’s see something  
happen now. You can break  
that big plan into small steps  
and take the first step right  
away.”*

*Indira Gandhi — Politician*

### 3 HOW TO PERFORM MULTIOBJECTIVE OPTIMISATION

Preceding sections have been deliberately vague in the analysis of how multiobjective optimisation functions — it has been enough to assert that search techniques, harnessing the power of Pareto dominance, exist and that they provide mechanisms for moving through the objective-space towards apparently optimal regions. With the foundations and motivations of multiobjective theory now well established through overview, formalisation and illustrative examples, a more detailed, though still introductory, exploration of multiobjective optimisation is appropriate.

#### 3.1.1 DEFINING A BASIC INTELLIGENT ITERATIVE SEARCH ALGORITHM

For the purposes of this work, a basic search algorithm is charged with the progressive exploration of high-utility regions of the decision-space. Consider a population of solutions  $P$ : a basic search algorithm assesses the *utility* of each constituent ( $p$ ) of that population and *selects*  $b$  ( $1 \leq b \leq |P|$ ) members (the bag<sup>4</sup>  $S$ ) around which the search will be centred. Progression through the space is achieved by *varying* constituents of  $S$  (via the `vary()` operator – see Section 3.1.4) to produce  $c$  ( $c \geq 1$ ) new members. The resulting bag is added to  $e$  ( $0 \leq e \leq |P|$ ) solutions from  $P$  to form the *population* for the next iteration of the search. As outlined in Algorithm 1, a complete basic search is simply a repetition of this procedure, with provisions made for the generation of the initial bag and termination<sup>5</sup>.

**Algorithm 1 — A Basic Search Algorithm**

---

1: $P := \text{generateInitialPopulation}()$	Specify the starting point of the search.
2: <code>while(terminateSearch() <math>\neq</math> true)</code>	Continue searching until termination conditions met
3: $\forall p \in P, p^{\text{utility}} := \text{utility}(p)$	Calculate and record the utility of each solution.
4: $S := \text{select}(P, b)$	Select $b$ members of $P$ for variation.
5: $P' := \text{vary}(S, c)$	The population for the next iteration is the $c$ solutions
7: $\text{Archive} := \text{truncate}(P, e)$	created from the selection set and the members of
8: $P := P' \cup \text{Archive}$	the <i>Archive</i> (which features at most $e$ proposals).

---

<sup>4</sup> Note that  $S$  is a bag and not a set. The `select()` operator may extract the same solution from  $P$  multiple times and there is therefore no guarantee that the constituents of  $S$  are unique.

<sup>5</sup> See the work of Laumanns *et al.* [126] for an alternative, though equally valid, generic search model.

It is important to note that the  $b$ ,  $c$  and  $e$  parameters need not necessarily be fixed constants, they can vary over the course of the run (perhaps according to the changing properties of the population  $P$ ). For the most part though, this flexibility is not capitalised upon by contemporary optimisation algorithms.

### 3.1.2 ANALYSING THE BASIC SEARCH ALGORITHM

It is important to note that the analysis of the simple algorithm described in Section 3.1.1 is subject to a number of reasonable assumptions. Specifically, it is assumed that:

- The `select()` operator biases high-utility solutions and it is *always* possible to select  $b$  members from  $P$ ;
- The `vary()` operator produces new solutions that are derived from, though typically unique to, the members of  $S$ ; and
- The `truncate()` operator disposes of the  $(|P|-e)$  lowest utility solutions in  $P$ .

If these assumptions hold true, variation in the  $b$ ,  $c$  and  $e$  parameters dictates much of the behaviour of the algorithm. The  $b$  setting reflects the trade-off between exploration and exploitation — as  $\frac{b}{|P|}$  tends towards zero, more selective pressure is applied and only the most promising solutions (according to the selection operator) are pursued; as  $\frac{b}{|P|}$  tends towards one, the search becomes more random (due to a decrease in selection pressure). The  $e$  parameter indicates the level of elitism employed, where  $\frac{e}{|P|}$  is the fraction of the population carried over from previous iterations. When  $\frac{e}{|P|}=1$ , no solutions are ever lost and the selection operation operates on an unbounded population set; when  $\frac{e}{|P|}=0$ , selection considers only those solutions produced during the previous iteration. The  $c$  parameter suggests

how aggressively the search is focussed around the  $b$  selected members (the higher the value of  $\frac{c}{b}$ , the more thorough the search around  $S$  becomes).

Complementing the exploitation/exploration trade-off suggested by the  $b$  parameter setting is the selection operation. Though, the procedure should be charged with biasing high-utility solutions, it may incorporate noise to encourage the increased diversity offered by the inclusion of weaker proposals. While a host of differing approaches exist, at their core, most contemporary search algorithms make use of truncation, tournament or roulette-wheel selection methods (see [127] for summaries of popular methods). Truncation is the simplest of the available approaches and simply requires the selection of the  $b$  best proposals. In the tournament methodology, participating solutions are randomly selected and the winner is the entrant featuring highest utility, with the bag  $S$  produced by performing this procedure  $b$  times. In roulette-wheel, selection is probabilistic, with the likelihood of a solution being selected proportional to its normalised utility score. The inclusion of restrictions in these basic operations can give rise to the types of selection procedures encountered in Tabu Search [128, 129] (filter prospective members with a tabu list) and Simulated Annealing [130, 131] (select weaker solutions according to a cooling scheme).

### **3.1.3 CONSTRUCTING EXAMPLE GENERIC SEARCH ALGORITHMS**

With the basic algorithm and parameter definitions in place, it is possible to construct a number of common search strategies by simply adjusting the key parameters of the generic procedure (see Table 1 for a summary). Note that the strengths and weakness of these approaches in a multiobjective context will be examined later (see Chapter 6, Chapter 8 and Chapters 9–12, in particular) — this section is principally devoted to their core definitions.

#### **3.1.3.1 RANDOM SEARCH**

The simplest basic search technique, random search, disregards the utility of solutions and moves randomly throughout the decision-space. Clearly, any effective intelligent search should outperform such a rudimentary system, though it may be used as a base-line technique (see, for instance, Zitzler, Deb and Thiele’s study [92]) and is included here for the sake of completeness.

Table 1 — Various Basic Search Procedures using Algorithm 1

<b>Single Point Random Walk</b>	generate initial population()			Produce exactly one solution.		
	select()			Arbitrary		
	truncate()			Remove all members from $P$ .		
	$b$	$c$	$e$	1	1	0
<b>Population Based Random Search</b>	generate initial population()			Arbitrary.		
	select()			Return all members of $P$ .		
	truncate()			Remove all members from $P$ .		
	$b$	$c$	$e$	$v > 1$ (a user-defined constant)	$v$	0
<b>Greedy Hill Climbing</b>	generate initial population()			Produce exactly one solution.		
	select()			Return highest-utility solution from $P$ .		
	truncate()			Remove lowest-utility solution from $P$ .		
	$b$	$c$	$e$	1	1	1
<b>Non-Elitist Population Based Search</b>	generate initial population()			Produce $n$ solutions (where $n$ is a user specified number; $n \geq 1$ ).		
	select()			Biases retrieval of high-utility solutions from $P$ .		
	truncate()			Remove all members from $P$ .		
	$b$	$c$	$e$	$v > 1$ (a user-defined constant)	$n$	0
<b>Bounded Elitist Population Based Search</b>	generate initial population()			Produce $n$ solutions (where $n$ is a user specified number; $n \geq s$ ).		
	select()			Biases retrieval of high-utility solutions from $P$ .		
	truncate()			Biases removal of low-utility solutions from $P$ .		
	$b$	$c$	$e$	$v > 1$ (a user-defined constant)	$q - s$ ( $q$ is a user-defined constant; $q \geq s$ )	$s \geq 1$ (a user-defined constant)
<b>Unbounded Elitist Population Based Search</b>	generate initial population()			Produce $n$ solutions (where $n$ is a user-specified number; $n \geq 1$ ).		
	select()			Biases retrieval of high-utility solutions from $P$ .		
	truncate()			No solution is removed from $P$ .		
	$b$	$c$	$e$	$x$ (user-defined constant or proportion of $P$ )	$y$ (user-defined constant or proportion of $P$ )	$ P $

When the `generateInitialPopulation()` operator produces a single random solution and parameter values of  $b = 1$ ,  $c = 1$  and  $e = 0$  are employed, Algorithm 1 instigates a random walk about the solution-space. A population-based random search can be achieved by setting  $e = 0$ ,  $b$  and  $c$  to some user defined constant and using a `select()` operator that returns all members of  $P^6$ .

<sup>6</sup> Note that since selection does not bias high-utility solutions in either the random walk or population-based approaches, the analysis provided in Section 3.1.2 does not hold for either search procedure.

### 3.1.3.2 GREEDY HILL-CLIMBING SEARCH

Amongst the simplest of the intelligent search strategies, greedy hill-climbing algorithms actively pursue the single most promising proposal and operate on only two solutions — typically referred to as the *parent* and *child*. At any time  $t$ , the selection bag  $S$  is composed only of the best solution found thus far (the parent), with replacement only occurring when a variant of this solution (the child) is of superior utility<sup>7</sup>.

The random walk technique offered in the preceding section is transformed into a hill-climbing system by setting  $e = 1$  (thus allowing comparison between parent and child) and using a selection mechanism that *always* chooses the highest utility member of  $P$ .

### 3.1.3.3 NON-ELITIST POPULATION-BASED SEARCH

Where the hill-climbing search was tasked with the optimisation of a single solution, a population-based search endeavours to improve multiple proposals simultaneously. In a non-elitist form of population-based searches, solutions remain in the set  $P$  for exactly one iteration — irrespective of how impressive they are with respect to utility. As such, there can be no guarantee that the current population contains the best set of proposals discovered thus far or even the single best solution found.

A non-elitist population-based search can be formed by using a selection mechanism that biases high-utility solutions and setting  $e = 0$ ,  $c$  to any user-defined population size and  $b$  to any user-defined constant greater than zero. It is worth noting that  $b$  can be any positive integer since selection can extract the same solution multiple times.

### 3.1.3.4 ELITIST POPULATION-BASED SEARCH

If a less temporal store of valuable solutions is required, the population-based search can be amended to include historical data via the incorporation of elitism. A common technique is to include a truncated form of the complete history of the search that is bounded in size by some user-defined parameter  $s$  ( $1 \leq s < |P|$ ). The

---

<sup>7</sup> A common variation is to allow replacement when the solutions are of equivalent worth, with heuristics often used to determine which solution better aids the search.



truncated elite search requires  $e = s$ ,  $b > 0$ ,  $c > s$  and a `generateInitialPopulation()` operator that produces more than  $s$  solutions.

By bounding the number of solutions that may be maintained across iterations however, potentially useful proposals may be lost. An unbounded approach guarantees that this cannot occur and can be formed by ensuring no truncation ever occurs ( $e = |P|$ ). In this case, settings for  $b$  and  $c$  are flexible (not necessarily fixed) and dependent on the user.

### 3.1.4 SOLUTION VARIATION

The basic search techniques all assume the existence of some variation operator that is responsible for the transformation of solutions. Though the mechanisms employed vary (particularly with the type of alphabet used and the form of problem encountered) most techniques are built on the same general principle — that is, to offer variants of selected solutions that represent similar, though distinct, portions of the decision-space.

Amongst the most popular of the variation mechanisms in contemporary research, and the type that will be used for all optimisers explored in this thesis, are the genetic operators. Derived from biological inspirations, their use with the basic search procedure outlined in Section 3.1.1 gives rise to what is commonly known as the *genetic algorithm* [132, 133] (a specialisation of the more general *evolutionary algorithm*).

#### 3.1.4.1 GENETIC OPERATORS AND GENETIC ALGORITHMS

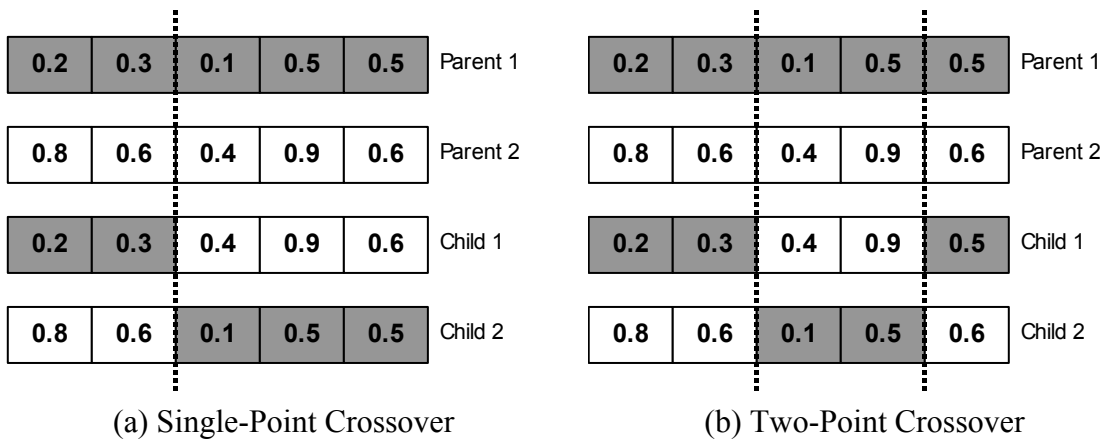
The genetic algorithm is little more than an analogy used to clarify the generic search technique when variation is performed via genetic operators. By recasting the core components of the basic search procedure into a biological framework, the process takes on the characteristics of a simplified Darwinian system. If solutions are *chromosomes* made up of *genes* and the utility of a chromosome is its *fitness* against a given problem, progressive improvement based on *reproduction* (variation) according to fitness implies an *evolution* of the chromosome; as in nature, the fit survive to propagate their genetic traits.

The genetic operators (acting under the banner of reproduction) vary an incoming set of solutions via rudimentary approximations of *crossover* and/or *mutation*.

Crossover mimics sexual reproduction by splicing together two solutions (chromosomes), while mutation perturbs selected decision variables (gene values or *alleles*). The specifics of how these procedures are achieved vary dramatically (see [134, 135] for a review), though single-point and two-point crossover are both popular approaches (see Figure 8 for illustrations), while mutation is typically achieved by shifting the original gene value according to some random Gaussian. Users must also specify the rates at which crossover and mutation are applied for any given variation — with common settings fluctuating across the field<sup>8</sup>.

### 3.1.5 DEFINING SOLUTION UTILITY

Defining solution utility (or fitness, in the parlance of evolutionary algorithms) for single-objective optimisation is trivial — the better a solution performs on the given problem, the better its utility. If a multiobjective problem is scalarised, the performance of a solution is similarly straight-forward — the operation proceeds precisely as it would in the context of a single-objective domain. However, scalarisation is an inherently flawed notion (as outlined in Section 2.4.2 and discussed, to a lesser extent, in Section 4.1) and the majority of contemporary evolutionary researchers prefer the performance of solutions to remain in its original



**Figure 8 — Sexual Crossover**

The dashed line represents the crossover point/s. Shaded genes represent those portions of a child inherited from the first parent; unshaded regions indicate genes provided by the second parent.

<sup>8</sup> Dumitrescu *et al.* [134] note that traditionally the probability of gene-mutation lies between 0.001 and 0.01 (with contemporary studies using a lower bound of  $1/m$ ), while crossover rates generally fall between 0.2 and 0.95. In contemporary genetic algorithm-based multiobjective optimisation research, it is far more common for crossover to occur between 60% and 100% of the time (see, for instance, [82, 91, 92, 136-138]).

vector form, with Pareto dominance guiding the search [95]<sup>9</sup>. As implied in Section 2.1 however, Pareto domination alone imparts only a partial order on solutions and is thus insufficient for a complete fitness function — it is ambiguous as to where incomparable solutions should be ranked. Given the explicit goal of most multiobjective optimisers is to develop an accurate, well spread and evenly distributed front [139], it appears that a reasonable resolution of the issue can be found if the dominance performance of a solution can be effectively merged with its capacity to improve the extent and distribution of the set in which it resides. Assuming such a combination infers a complete, unambiguous, order amongst any set of solutions, not only has an applicable fitness function been defined, but one which also drives the search according to the core needs of multiobjective optimisation.

It is perhaps unsurprising then that the vast majority of contemporary multiobjective evolutionary algorithms feature utility functions that, at their core, are predicated on these principles (as illustrated in Section 3.1.6). Though the specifics are a key point of differentiation between algorithms (again, see Section 3.1.6), the same general methodologies typically apply — a measure of solution dominance is augmented by an estimation of crowding that infers how valuable it is to the distribution and extent of the front. Under this procedure, non-dominated uncrowded solutions are the elite of the current population and it is these types of solution that the search is actively pursuing.

### **3.1.6 EXISTING EVOLUTIONARY MULTIOBJECTIVE OPTIMISERS**

Though it is beyond the scope of this work to offer a review of all existing evolutionary multiobjective optimisers (see [125] for an excellent, though increasingly dated, summary) and later sections will explore a host of leading techniques in finer detail (see Chapters 9–12), it is important to examine how existing approaches combine the properties of a basic search technique with unique definitions of solution utility to form a cohesive whole. As such, Table 2 and Table 3 offer a constructive summary of some of the most popular Pareto-based multiobjective evolutionary algorithms in the field.

---

<sup>9</sup> A study of all multiobjective evolutionary algorithms up until 2001 [125] offers further support to such a claim – with approximately twice as many papers on Pareto-based techniques than on scalarisation methods.

**Table 2 — Traditional Pareto-Based Approaches to Evolutionary Multiobjective Optimisation**

$r$  is the user-defined crossover probability. For convenience, this table always assumes that the desired size of  $P$  is directly related to the size of the initial population. This is typically the case, but need not necessarily be true.

<b>MOGA (1993) [140]</b>	generate initial population()	Produce $n$ solutions (where $n$ is a user specified number; $n \geq 1$ ).		
	utility()	Number of solutions dominated and fitness sharing.		
	select()	Tournament.		
	truncate()	Remove all members from $P$ .		
	$b$	$c$	$e$	$c * (1+r)$ $n$ 0
<b>NPGA (1993) [141, 142]</b>	generate initial population()	Produce $n$ solutions (where $n$ is a user specified number; $n \geq 1$ ).		
	utility()	Examine Pareto dominance in a subset of $P$ (size defined by end-user); if precisely one solution is non-dominated, it wins, otherwise perform fitness sharing.		
	select()	Tournament.		
	truncate()	Remove all members from $P$ .		
	$b$	$c$	$e$	$c * (1+r)$ $n$ 0
<b>NSGA (1994) [2]</b>	generate initial population()	Produce $n$ solutions (where $n$ is a user specified number; $n \geq 1$ ).		
	utility()	The population is split into distinct ordered fronts. In any front of rank $l$ , all members must be incomparable and non-dominated by any constituent of a front greater than $l$ . Performance is based on the frontal ranking of a solution and fitness sharing.		
	select()	Roulette wheel.		
	truncate()	Remove all members from $P$ .		
	$b$	$c$	$e$	$c * (1+r)$ $n$ 0

It is interesting to note that there is a striking paradigm-shift between the traditional and contemporary techniques. The pioneers of the field concentrated principally on non-elitist procedures that capitalised on rudimentary fitness sharing techniques — where the utility of a solution is modified according to the number of solutions that share a particular objective- or decision-space niche of pre-defined size. In contrast, the more contemporary techniques introduce complex density estimation procedures for utility augmentation and illustrate an increased reliance on elitism (that is, the maintenance of apparently useful solutions across generations).

**Table 3 — Contemporary Approaches to Pareto-Based Evolutionary Multiobjective Evolutionary Algorithms**

$r$  is the user-defined crossover probability;  $a$  is a user-defined archive size;  $a'$  is a user-defined archive size or the cardinality of the non-dominated set in  $P$  (whichever is smaller);  $x$  is a user-defined value for the number of solutions produced through variation. For convenience, this table always assumes that the desired size of  $P$  is directly related to the size of the initial population. This is typically the case, but need not necessarily be true.

<b>SPEA (1998)</b> [80, 137]	gen. initial population()	Produce $n$ solutions (where $n$ is a user specified number; $n \geq 1$ ).		
	utility()	Fitness of archival members is proportional to the number of proposals that they dominate in $P$ . Utility of remaining members is related to the sum of the fitness scores of the non-dominated proposals that dominate them.		
	select()	Binary tournament.		
	truncate()	Retain only non-dominated proposals from $P$ . If necessary, truncate according to clustering.		
	$b$   $c$   $e$	$c * (1+r)$	$n - a'$	$a'$
<b>PAES (1999)</b> [143]  More details provided in Section 9.2.1.1.	gen. initial population()	Produce exactly one solution.		
	utility()	Divides the objective-space into a series of cells. The variant (child) is selected only if it dominates the selected solution or is otherwise non-dominated by an external archive of good proposals <i>and</i> resides in a less crowded objective-space cell.		
	select()	Single competition.		
	truncate()	Remove the loser of the competition from $P$ .		
	$b$   $c$   $e$	1	1	1
<b>NSGA-II (2000)</b> [144]  More details provided in Section 9.2.3.1.	gen. initial population()	Produce $n$ solutions (where $n$ is a user specified number; $n \geq 1$ ).		
	utility()	As per NSGA, though with cuboid nearest-neighbour estimates replacing fitness sharing.		
	select()	Binary tournament with members of <i>Archive</i> .		
	truncate()	Worst according to utility.		
	$b$   $c$   $e$	$c * (1+r)$	$n/2$	$n/2$
<b>PESA (2000)</b> [83]  More details provided in Section 9.2.2.1.	gen. initial population()	Produce $n$ solutions (where $n$ is a user specified number; $n \geq 1$ ).		
	utility()	Number of solutions sharing objective-space cell with solution.		
	select()	Binary tournament of randomly extracted archival members.		
	truncate()	Retain non-dominated proposals, reduce according to cell-based crowding.		
	$b$   $c$   $e$	$c * (1+r)$	$n - a'$	$a'$
<b>SPEA2 2001</b> [81, 145]  More details provided in Section 9.2.4.1.	gen. initial population()	Produce $n$ solutions (where $n$ is a user specified number; $n \geq 1$ ).		
	utility()	Strength of all solutions dominating the member (where strength is the total number of proposals in $P$ that a solution dominates).		
	select()	Binary tournament of archival members.		
	truncate()	Retain non-dominated proposals. If resultant <i>Archive</i> is too large ( $>a$ ) reduce according to $\kappa^{\text{th}}$ nearest-neighbour crowding; if <i>Archive</i> is too small ( $<a$ ), increase by including highest utility dominated proposals.		
	$b$   $c$   $e$	$c * (1+r)$	$n - a$	$a$
<b>IBEA (2004)</b> [91]  More details provided in Section 10.2.	gen. initial population()	Produce $n$ solutions (where $n$ is a user specified number; $n \geq 1$ ).		
	utility()	Based on Pareto-compliant performance indicators.		
	select()	Binary tournament of archival members.		
	truncate()	Remove lowest utility solutions.		
	$b$   $c$   $e$	$c * (1+r)$	$n-a$	$a$

## **3.2 CONCLUSIONS**

This section has offered an insight into how the principles of Pareto optimality may be effectively merged into a standard search procedure to produce a range of distinct evolutionary multiobjective optimisers. At their core, all of the examined multiobjective techniques approximate the utility of a solution according to some combination of Pareto dominance and crowding (even if this is done implicitly, as in IBEA), with contemporary optimisers sharply differentiating themselves from the first-generation of techniques through the use of truncated elite archives that reduce the loss of valuable solutions. Subsequent sections (see Section 6.2.1, Section 8.1.1 and Chapters 9–12, in particular) will explore in finer detail this difference, the motivations that inspired such a dramatic sea-change in the field and some of the key algorithms that belong to the second generation of techniques. For now though, a more pressing concern is discovering precisely what it is about multiobjective problems that has presented such a challenge for algorithm designers and has inspired such a rich tapestry of research in the field.

# Chapter

# 4

## Real-Valued Multiobjective Problem Characteristics

*"We are continually faced with a series of great opportunities brilliantly disguised as insoluble problems."*

*John W. Gardner — Author*

*"A problem well stated is a problem half solved."*

*Charles Kettering —  
Engineer and Inventor*

*"It is a mistake to think you can solve any major problems just with potatoes."*

*Douglas Adams — Author*

## **4 REAL-VALUED MULTIOBJECTIVE PROBLEM CHARACTERISTICS**

As evidenced in the Formula One wing design example (see Section 2.3), multiobjective problems can contain seemingly subtle issues that fundamentally inhibit the search process. While adapting improved designs for the front wing was complicated by the disparity between solution- and objective-space proximity (Section 2.3.4), this illustrates just one of a host of complex problem features that appear in real-world multiobjective tasks. Thus, the overall performance of a multiobjective optimiser is contingent upon its robustness in light of these characteristics.

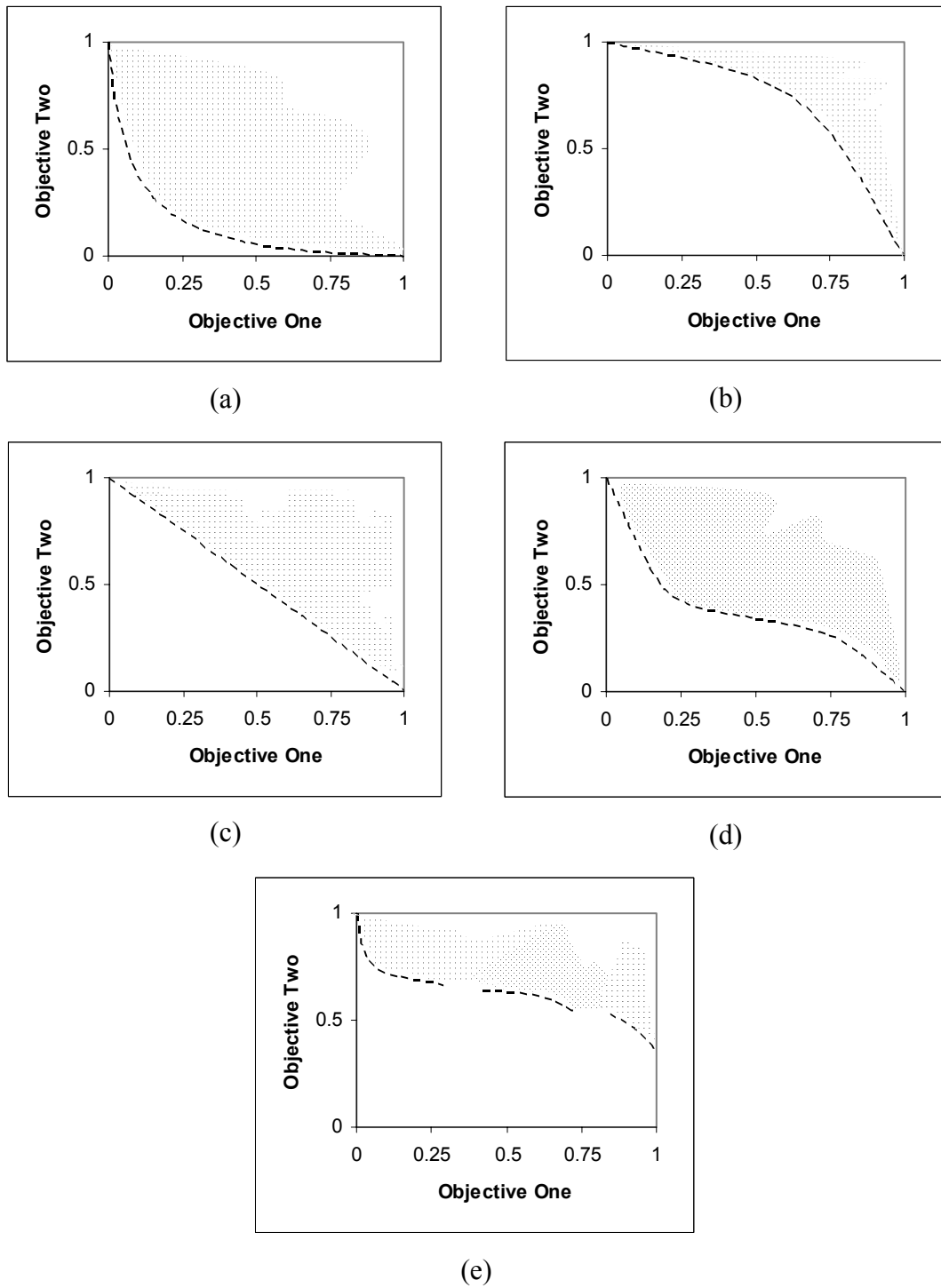
### **4.1 SHAPE OF THE PARETO OPTIMAL FRONT**

Multiobjective problems give rise to Pareto optimal fronts (or, more generally, trade-off surfaces) that feature convex, concave or linear regions — with no guarantee that all regions will be of the same shape or size (see Figure 9). The separation between each region represents discontinuities in fitness space (see Figure 9), where fronts consisting of minimal region size and high discontinuity lack any discernible shape and are better expressed as a collection of independent points.

While the difficulties caused by the various surface shapes will be addressed in more detail later (see Chapters 9–12, in particular), concave regions are generally recognised as the most problematic. For the wide range of search techniques that are reliant on implicit or explicit linear function approximation (such as the Vector Evaluated Genetic Algorithm [146] and the weighted-sum-based techniques reviewed in [5]), suitable approximations of concave optimal regions simply cannot be found [5, 147]. Instead, these searches strongly bias solutions of maximal reward in each objective — failing to adequately address intermediate or mutually beneficial solutions.

Although the community generally recognises objective-space convexity as a significantly less difficult problem feature than concavity (of all problem features examined in Zitzler, Deb and Thiele’s study [92], they note that convexity appears to





**Figure 9 — Example Pareto Optimal Fronts**  
 (a) Convex. (b) Concave. (c) Linear. (d) Mixed. (e) Discontinuous mixed.

“cause the least amount of difficulty”), certain forms of Pareto dominance-guided search will infer a bias around mutually beneficial solutions [148]. The effect of such biasing is that algorithms may face difficulties in locating the extreme solutions

that define the boundaries of the Pareto optimal front. Such boundaries are significant in placing compromises in context and, consequently, their absence will inevitably cost the end-user some understanding of the nature of the objectives being addressed.

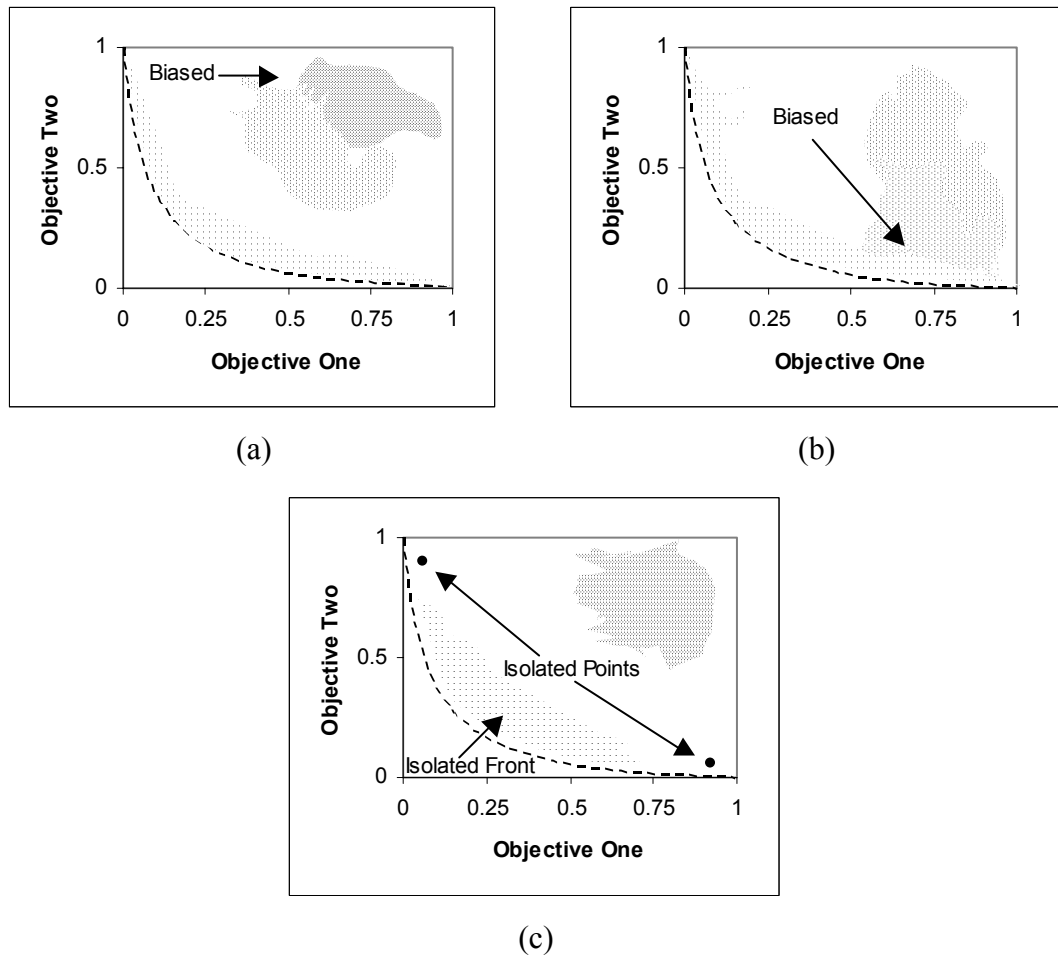
Finally, discontinuous surfaces can lead search algorithms to inappropriately bias isolated sub-regions [148], particularly if a given portion of space is more densely populated than other areas. The danger associated with such partially represented Pareto optimal fronts is that they can provide a misleading, or otherwise incomplete, representation of the relationships that exist between objectives and the extent of the fitness space. Moreover, an incomplete front fails to provide the rich palette of compromise solutions expected from a multiobjective optimisation run.

Given that concave, convex and discontinuous surfaces can each adversely affect the quality of the final output produced by a search, the analysis of multiobjective optimisers under the presence of differing frontal shapes has been a key point of investigation in many studies (see [92, 149-151], for instance). Still, the shape of the optimal trade-off surface will typically affect the search only near the completion of the process, where the optimiser is tasked with distributing the solutions along the discovered optimal front. Progression towards that front is more strongly influenced by the characteristics of the objective- and decision-spaces as a whole.

## **4.2 THE NATURE OF THE OBJECTIVE-SPACE**

### **4.2.1 BIAS**

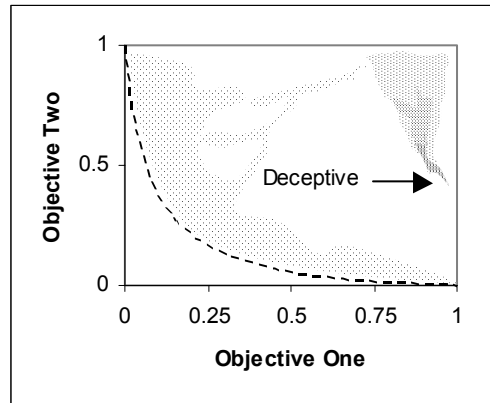
The uniformity of the objective-space for any real-world multiobjective problem is generally not well known in advance, but it is likely that at least some regions will be more densely populated than others. Since search processes tend to bias areas of high occupation [148], it follows that the performance of a multiobjective optimiser may be influenced by the placement of solutions in fitness space. In particular, if solutions are densely populated in a region of objective-space that is far removed from the Pareto optimal front (see Figure 10a), convergence onto the front is likely to be slow. Similarly, a high density of solutions focused around a specific segment of the Pareto optimal front (see Figure 10b) may result in overtly biased approximations or slower development of a more evenly distributed set.



**Figure 10 — Example Biased Fitness Spaces**

(a) Biased away from Pareto optimal region. (b) Biased around a small region of the Pareto optimal front. (c) Highlighted points are isolated from much of the fitness space. Darker shading represents a higher density of points.

Given the propensity for most search algorithms to favour areas of high solution density, it is not particularly surprising to note that isolated points (as illustrated in Figure 10c) are often particularly difficult to locate (assuming a reasonable accord between solution- and objective-space proximity). Since such solutions will provide little gradient information in fitness-space to encourage a search, isolated optimal regions represent amongst the most difficult of the multiobjective problem characteristics.



**Figure 11 — An Example Deceptive Fitness Space**

Objective-space is biased towards a locally impressive, but ultimately poor, region. Darker shading represents a higher density of points.

#### 4.2.2 DECEPTION

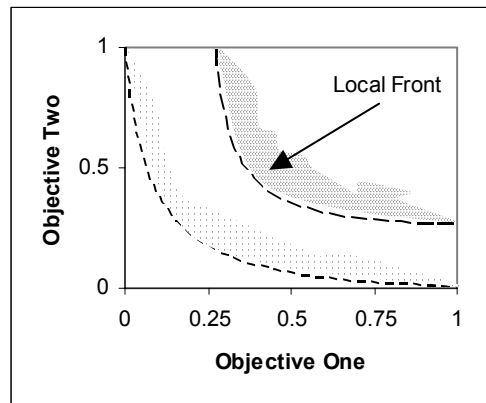
When an objective-space is biased in such a way as to mislead a search towards locally optimal, but globally sub-optimal, solutions, it is classed as deceptive. Typically, a deceptive problem requires both a high density of solutions near the sub-optimal region, but also some form of positive utility-gradient that a search can erroneously follow (see Figure 11, for instance). It is possible, given a suitably strong set of attractors that are well removed from other, more beneficial, portions of objective-space, that a search can become stranded in locally optimal space — much like a crayfish lured into a craypot. To escape the region, a search must be capable of retracing its steps (in spite of negative reward) or otherwise jumping to a different section of search space (typically through some form of stochastic variation).

#### 4.2.3 MULTIPLE FRONTS

Any region in objective-space that consists of well-distributed locally<sup>10</sup> non-dominated solutions can be classed as a sub-optimal front (see Figure 12 for an illustration). In most multiobjective works, the existence of such fronts is referred to as multi-modality (as suggested by Deb [148]), though Veldhuizen [152] and Coello

---

<sup>10</sup> In general, the concept of local space is ill-defined, and used to indicate an area *near* to the current region examined. Even the definition provided by Deb [148] fails to provide a fixed term for the extent of space covered.



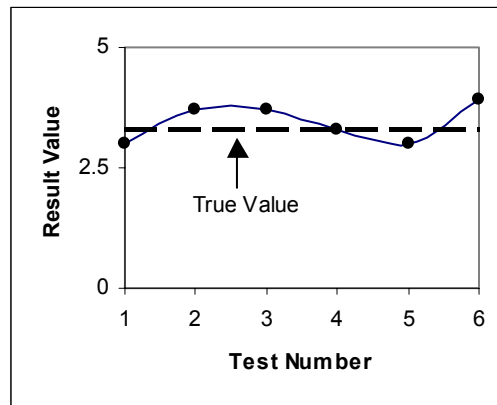
**Figure 12 — An Example Multi-Frontal Fitness Space**

The included fitness space features two isolated fronts — a sub-optimal, local front and the globally Pareto optimal front.

*et al.* [51] correctly note that the name is potentially misleading given its alternative use in single-objective optimisation (they prefer the term, multi-frontal). Irrespective of name, moving through sub-optimal fronts is difficult for most algorithms, particularly if the region is largely removed from better solutions. As with singly isolated solutions, isolated fronts provide little Pareto dominance-based gradient information upon which to move a search forward, and many optimisers will stall — endeavouring instead to distribute solutions *along* the front.

#### 4.2.4 NOISE

For real-world problems, particularly those that require some form of practical testing or imprecise modelling, there is no guarantee that a particular solution will consistently give rise to an identical point in objective-space [10, 153] (see Figure 13 for an example noisy solution) — it may be subject to inaccurate measurement, flaws in construction or variable test conditions. In the pathological case, such poor fidelity can lead to the rejection of good compromises in exchange for bad solutions — but even in systems of relatively low noise, potentially useful search ideas may be rejected. If multiobjective optimisation is to find successful application in real-world problems, search algorithms should be able to adequately circumvent the effects of such noise.



**Figure 13 — An Example Noisy Solution**  
The utility of the solution changes depending on when it is observed.

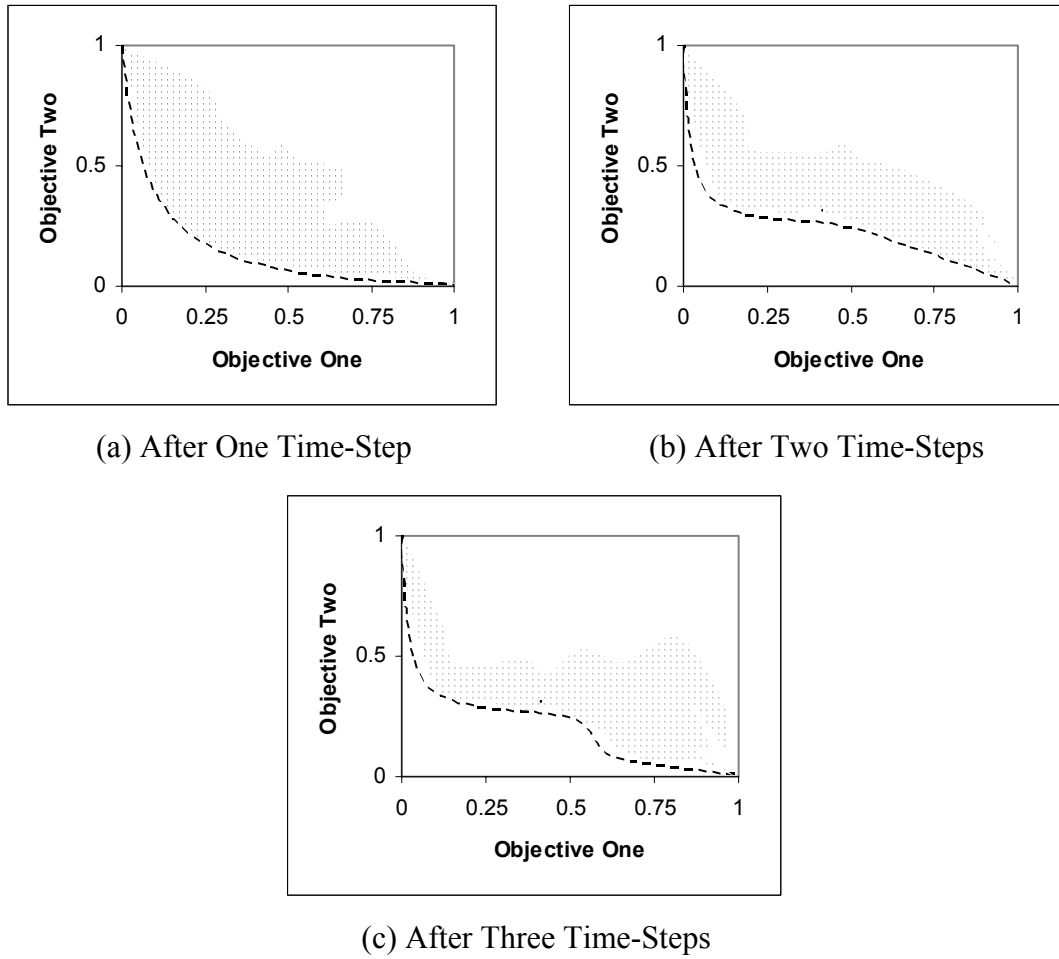
#### 4.2.5 COLLATERAL NOISE

A more subtle problem exists in the shape of collateral noise [148], where the utility of a solution is poor due to a small subset of the decision variables. If the remaining elements of the solution are beneficial to the search process, the poor performance is, at least partially, misleading, and may result in the loss of useful ideas. It is akin to building a magnificent house and destroying it because of a single unappealing brick.

Since, the potential for such disruptive noise is generally not well known in advance, it is difficult to address without a thorough understanding of the relationships that exist between parameters in the solution-space or a robust search algorithm that can perform well in spite of its presence.

#### 4.2.6 DYNAMIC OBJECTIVE-SPACES

While the shifting fitness landscapes of a noisy system are a by-product of inaccurate performance measurements, dynamic objective-space is a reflection of the problem itself. Specifically, a dynamic problem is one whose goals are not fixed — they will vary with time due to some property of the system being modelled (see Figure 14). For instance, a public power distribution system will have differing requirements depending on the demands of the population at any given time. Thus, where a search algorithm addressing a noisy function is fundamentally charged with finding the true



**Figure 14 — An Example Dynamic Fitness Space**

(a) The initial state of the illustrative space. (b) and (c) The fitness space after subsequent time steps — note that the Pareto optimal front, and the space in general, vary over time in this example.

optimal points that are concealed by noise, the goal in a dynamic problem domain is instead to locate and subsequently track the moving optima [154]. Consequently, dynamic problems and noisy functions represent conflicting requirements in an optimiser — a noisy function needs robustness and rigidity in the face of subtle change, while adaptability is paramount in dynamic problem sets.

#### 4.2.7 DIMENSIONALITY AND EXTENT

Perhaps unsurprisingly, the extent and dimensionality of the objective-space play a large part in determining how efficiently and effectively a Pareto optimal front approximation can be formed. In particular, with a larger number of objectives, it becomes considerably more difficult to establish Pareto dominance [155] — a greater

majority of solutions will be incomparable, and thus any Pareto guided search must develop a way of identifying which of these solutions is more valuable. Such are the difficulties associated with high-dimensional problems that Bellman and Dreyfus ([156] citing [157]) grimly describe these effects as a consequence of the “dimensionality curse”.

Generally less significant is the extent of the objective-space under consideration. In particular, growth in continuous spaces will tend only to slow the overall search process (since, fairly obviously, there is a greater area to explore). Still, if there are disparities between the extent of differing objectives, there does exist the potential for biasing — searches may focus greater exploratory pressure on the larger dimension, for instance.

More problematic are undefined spatial boundaries — where a lack of *a priori* domain knowledge prohibits a precise definition of the extent of particular dimensions. Since the range of potential values cannot be provided in advance, those algorithms reliant on objective-normalisation will be severely inhibited. In such cases, initial exploratory runs or localised normalisations (based on a subset of the solutions found thus far) are likely to be necessary.

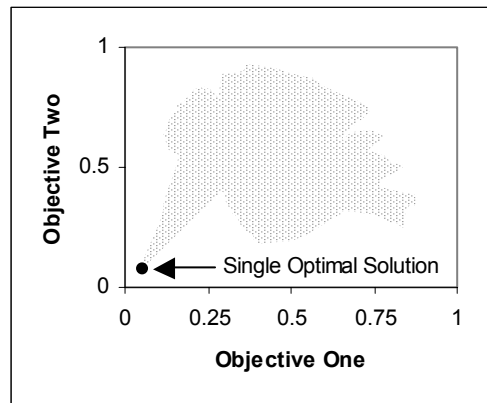
#### **4.2.8 DEGENERATIVE FRONTS**

There is no guarantee that the Pareto optimal front of a given problem will be expressed in every dimension of the objective-space. For instance, the optimal region of a three-dimensional objective-space may be fully expressed as a two-dimensional concave trade-off surface, or a bi-objective problem may degenerate towards a single optimal solution (as in Figure 15). While practical applications of multiobjective optimisation illustrate that such degenerative fronts are atypical<sup>11</sup> and though the characteristic has only recently drawn any attention from the field (see the impressive work of Huband *et al.* [158]), it rests as an interesting test of the robustness of a search under changing conditions.

---

<sup>11</sup> The author of this thesis is unaware of any contemporary practical multiobjective study that describes the presence of a completely degenerative front.





**Figure 15 — An Example Degenerative Problem**

The multiobjective problem degenerates towards a single ideal solution, rather than a set of trade-offs.

## 4.3 THE NATURE OF THE SEARCH SPACE

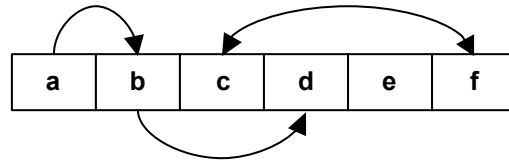
### 4.3.1 DIMENSIONALITY AND EXTENT

While the dimensionality of the search space is generally less pivotal than the dimensionality of the objective-space, it will still affect the performance of a multiobjective optimiser. In particular, the greater the number of decision variables, the more likely it is that collateral noise will influence the search process. Moreover, higher dimensional spaces will generally take longer to search — if only because of the exponential growth of the decision-space under increasing dimensions.

As in the objective-space (see Section 4.2.7), the extent of each decision-space dimension will affect search efficiency and potentially increase search space bias. In general, the potential for ill-defined spatial boundaries is less likely, though still possible, for decision-spaces, since this information is typically known *a priori*.

### 4.3.2 SEPARABILITY

Many real-world problems feature solutions whose utility is contingent on decision variable dependencies [158] (see Figure 16). In these circumstances, decision variables should not be considered independently, as the relationships that exist between them are significant. Consider the construction of an ideal dinner party — inviting the ten most charismatic people you know means little if each of those people hate each other. The interaction between variables dictates the success of the solution as a whole.



**Figure 16 — An Example Non-Separable Solution**

*a* is dependent on *b* and is therefore implicitly dependent on the value of *d*; *b* is dependent on *d*; *c* and *f* are dependent upon each other; and *e* is completely independent of all other variables.

In general, such non-separable problems are more difficult than tasks featuring variable independence [158] since they are subject to potentially high levels of collateral noise. Alter one decision variable — swap one person from a seat — and the fitness of all related variables will inevitably change. As such, optimisers that face high levels of non-separability require strong building-block linkages, where the search must somehow incorporate dependencies into the variation process.

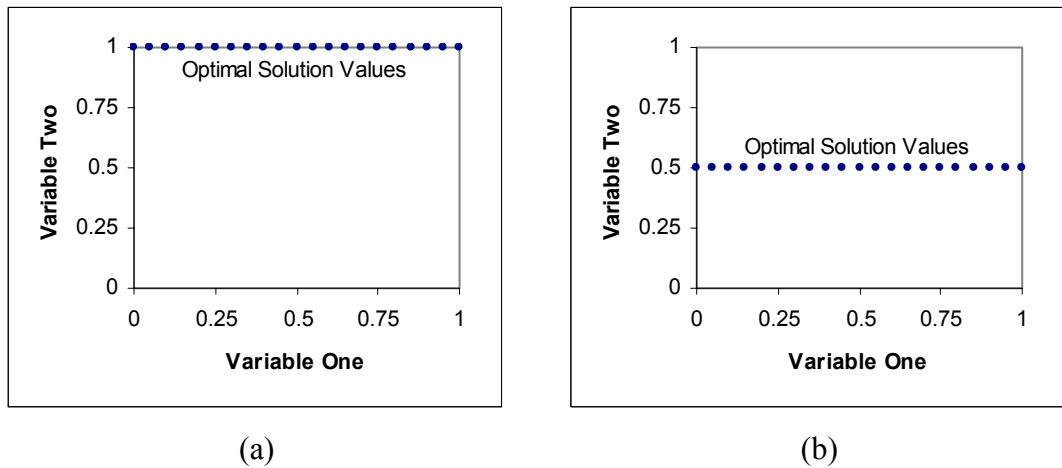
### 4.3.3 REDUNDANCIES

For problems where domain knowledge is low, the inclusion of high redundancy variables is always a possibility — that is, decision parameters that have little-to-no affect on the overall efficacy of the solution. By including such variables, the search space takes on greater dimensionality than required and will subsequently incur a reduction in search efficiency.

### 4.3.4 OPTIMA LOCALITY

It is not particularly surprising to note that the position of optima in solution-space will affect the performance of multiobjective optimisers. In particular, extreme or medial decision variables (see Figure 17) will be preferred by particular forms of search according to the type of solution variation used (as will be discussed in Section 5.1.1.1).

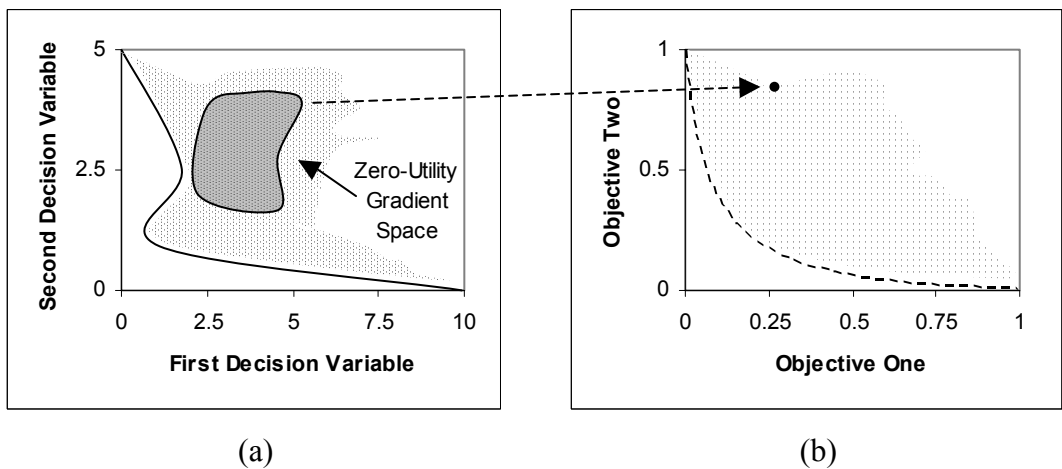
While it is unlikely that purely extreme or medial optima will exist in real-world problems, thus limiting the effect of any search bias, it can occur in test problems (as in the *ZDT* suite proposed by Zitzler *et al.* [92]) and is therefore important to identify.



**Figure 17 — Example Extreme and Medial Optimal Values in Solution-space**  
 (a) The optimal values for variable two are at an extreme of 1. (b) The optimal values for variable two are medial, centred on 0.5.

#### 4.3.5 ZERO UTILITY-GRADIENTS

If a large region of decision-space maps to a single point in objective-space, the search algorithm is provided with no utility-gradient information upon which to determine a useful direction for investigation. Consider a solution that is surrounded in decision-space by variations that are all considered equally good compromises (as illustrated in Figure 18) — in this case, there is no obvious way of deciphering



**Figure 18 — An Example Zero-Utility Gradient Space**  
 The highlighted portion of solution-space (a) is mapped to a single point in fitness space (b).

which, if any, of the surrounding options provide the best chance of moving towards an optimal point. It is like being lost in a forest composed only of identical trees — there are no landmarks for orientation. It is possible for algorithms to become inexorably stranded in such featureless regions, particularly if neighbouring areas are of lower utility. Even stochastic approaches with high variation in solution values will tend to oscillate around the centre of sufficiently large zero-gradient regions. Indeed, Deb [148] claims that if most of the search space lacks gradient information, then there exists “no optimisation algorithm [that] will perform better than an exhaustive search method”.

#### **4.3.6 GRANULARITY**

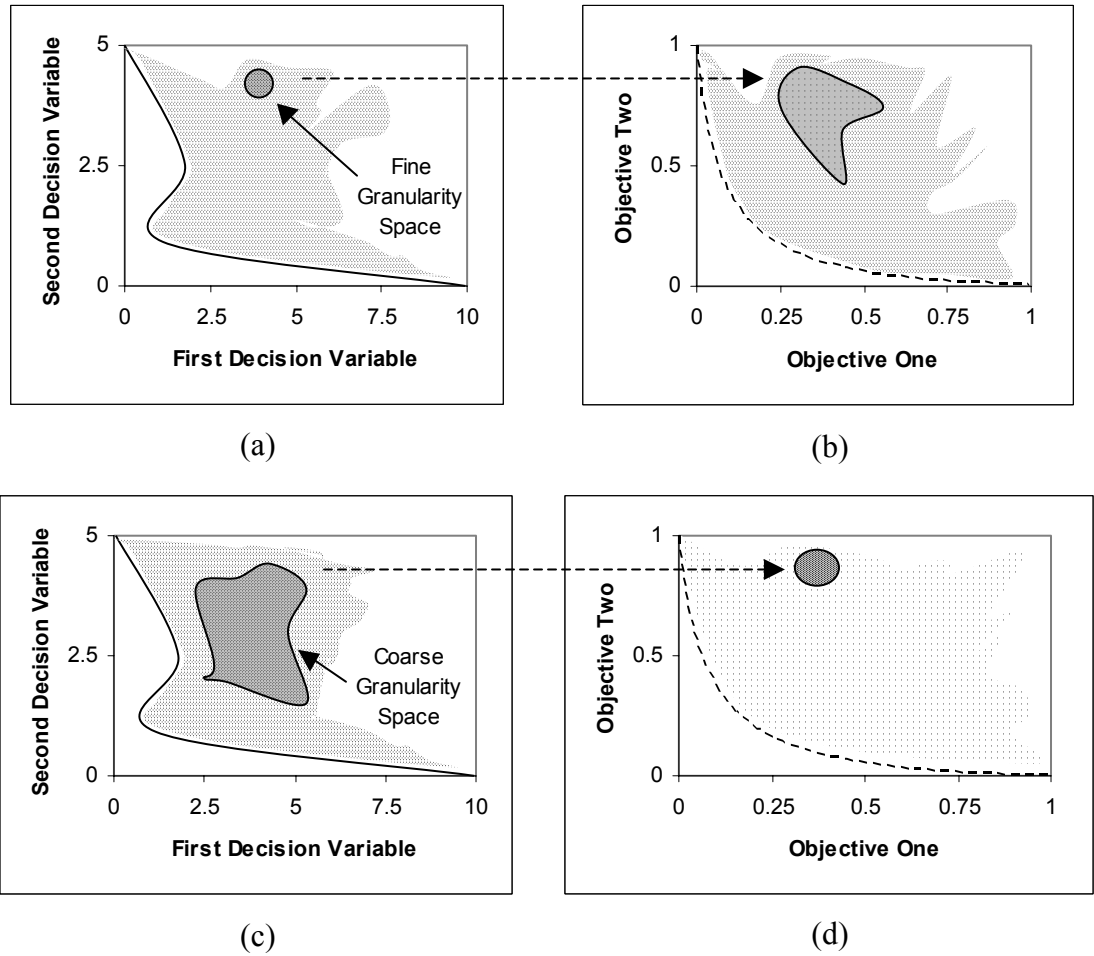
If a small region of decision-space maps to a wide variety of points in objective-space (as exemplified in Figure 19a and Figure 19b), utility-gradient information may be available to a search, but very fine exploration granularity is required to establish the best available choices. Since minor solution variations may lead to large movements through objective-space, it is possible for a search to over-step useful regions, potentially discouraging exploration of high-reward areas.

In contrast, if the granularity of a search space is coarse — that is, if large portions of the decision-space map to a small region of the objective-space (with the extreme leading to a zero utility-gradient surface) — then small step sizes will invariably slow the process with little in the way of reward (see Figure 19c and Figure 19d).

Thus, a successful search algorithm must balance the efficiency afforded by large step sizes and the efficacy of more precise movement. In general, this accord is tailored to the problem at hand, often through the application of parameter tuning (where step sizes are adjusted according to previous performance) or domain knowledge (where exploration granularity is dictated by presumptions about the nature of the problem).

### **4.4 CONSTRAINTS**

In many real-world design and engineering problems, there are regions of the search- or objective-space that are infeasible to realise in practice [159] — be it because of the complexities associated with construction, budgetary limitations or some more

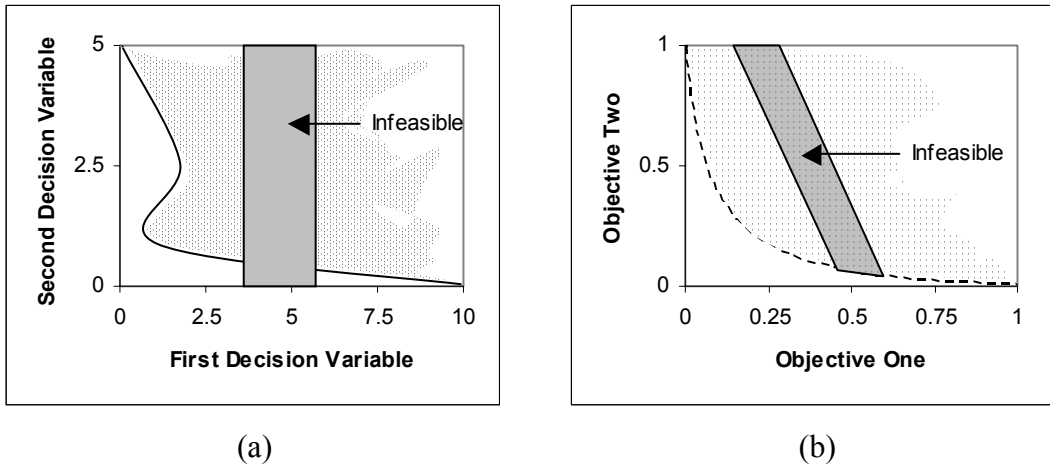


**Figure 19 — Example Search-Space Granularities**

A small region of the search space in (a) is mapped to a very large region of fitness space in (b), thus requiring a small step size to successfully navigate this region. In contrast, a large area of the decision-space in (c) is mapped to a relatively small segment in (d) — consequently, it is best served by large step sizes. Darker shading represents a higher density of points.

subtle reason that a domain expert may be aware of. In these cases, the hard constraints must be included in the formulation of the multiobjective problem, lest valuable search time be wasted locating solutions that are of no practical value. The inclusion of such side-constraints<sup>12</sup> can complicate the decision or objective-spaces — with the introduction of infeasible regions potentially introducing new discontinuities (see, for instance, Figure 20). As such, side-constraints can introduce similar isolation and frontality issues to those seen in non-uniform objective-spaces

<sup>12</sup> In multiobjective literature, the term *side-constraints* is generally used to reflect a constraint that affects more than just the boundaries of a search space.



**Figure 20 — Example Constrained Spaces**

Example (a) illustrates a constrained solution-space, while (b) demonstrates the effects of the constraints in objective-space. Note that, in this example, the feasible space in both illustrations is now discontinuous and a portion of the Pareto optimal front is completely obscured by infeasibilities.

(see Section 4.2). Furthermore, the search process must develop a successful strategy for handling infeasibility. While it may seem obvious that a search can simply reject progress into infeasible spaces, prohibiting such exploration may severely hamper progress towards optimal feasible regions (again, see Figure 20).

It is important to note that for some problems, portions of the space are better described as undesirable rather than infeasible. In these cases, soft constraints may be applied to indicate a bias towards preferable parts of the decision or objective-spaces. While both hard and soft constraints are conceptually similar, they “must be uniquely processed” [51], since the exploration of infeasible space is more rigorously opposed than in undesirable regions.

## 4.5 SUMMARY

As explored in this section, a multiobjective task is subject to a host of complex problem characteristics that affect not only the appearance of the final Pareto optimal front, but also the structures and relationships in both search- and objective-spaces. While the No Free Lunch (NFL) theorems [160, 161] state that no amount of testing on these problem features will elucidate a clearly dominant general multiobjective search algorithm (since one cannot exist) — successful test functions will indicate the applicability of algorithms to particular types of domain. As such, a

comprehensive suite of test problems that examine a rich set of features will do much to suggest the performance of algorithms on a wide ranging collection of domains, while illustrating the particular characteristics of a given search strategy.

# Chapter

# 5

## A New Multiobjective Test Suite

*“Valor is a gift. Those having it never know for sure whether they have it till the test comes. And those having it in one test never know for sure if they will have it when the next test comes”*

*Napolean Bonaparte —  
General and Emperor*

*“Knowledge must come through action; you can have no test, which is not fanciful; save by trial”*

*Sophocles — Greek  
Playwright*

*“Each problem I solved became a rule, which served afterwards to solve other problems”*

*Rene Descartes — Scientist  
and Philosopher*

*“Avoid problems and you will never be the one that overcame them”*

*Rene Descartes — Scientist  
and Philosopher*

*“When a problem comes along, you must whip it”*

*Devo — Musical Group*



## **5 A NEW MULTIOBJECTIVE TEST SUITE**

Since the pioneering work of Veldhuizen and Lamont [162], the number of test suites available to the multiobjective community has grown markedly. Moving away from the ad-hoc and limited test functions that are indicative of a fledgling field, contemporary multiobjective optimisation researchers have developed a host of problems that stretch the capabilities of search algorithms across a range of features [92, 148, 154, 158, 162-166]. Still, despite the impressive array of options available for the comparison of multiobjective optimisation approaches, the claim that any single suite, *in toto*, will address all of the key characteristics of a real-valued multiobjective problem, is largely untrue.

Indeed, there exists no single general-purpose test suite that investigates all of the key characteristics discussed in Chapter 4. Though existing works are by no means unsatisfactory, they are typically limited in scope and each fails to explore important domain properties. Moreover, the multi-faceted nature of most contemporary multiobjective test problems makes elucidating the cause of algorithmic improvement or decline difficult — by mixing domain characteristics, test functions may achieve greater kinship with practical systems but they can also muddy the waters of investigative studies. As such, this thesis endeavours to offer a comprehensive suite of problems that each examine a core, largely isolated, multiobjective characteristic. The resulting collection — featuring thorough problem descriptions — should act as a valuable repository for researchers, particularly those who are interested in examining the effects of each domain property on optimiser behaviour.

### **5.1 EXPLORING EXISTING TEST SUITES**

Since the new composite test suite includes existing and modified functions adapted from the literature, an exploration of the works from which they are derived is valuable. Such an analysis should draw into focus the strengths and weaknesses of the originating functions and provide an insight into the diversity of approaches that exist within the field.

### 5.1.1 ZDT FUNCTIONS

The test suite provided by Zitzler, Deb and Thiele [92] (the *ZDT* collection) builds on the impressively simple structures developed by Deb [148], to produce a range of functions that each focus on a specific problem characteristic. By examining a reasonably isolated feature in each task, the test suite provides insight into the behaviour of algorithms on particular classes of problem and suggests the overall value of that algorithm to the representative domain. While limited in scope (the five real-valued functions are clearly insufficient to explore all multiobjective problem characteristics), the test suite has seen frequent application in the community and has been used as the basis for performance testing in a range of studies (see, for instance, [44, 149-151, 167, 168]).

#### 5.1.1.1 ANALYSING THE ZDT SUITE

The real-valued *ZDT* test functions are static unconstrained bi-objective problems defined by Equation (18):

$$\begin{aligned} &\text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}) \\ &f_1(\mathbf{x}) = f_1(x_1) \\ &f_2(\mathbf{x}) = g(x_2, \dots, x_m) \times h(f_1(x_1), g(x_2, \dots, x_m)) \\ &\text{where } \mathbf{x} = (x_1, x_2, \dots, x_m) \end{aligned} \tag{18}$$

where  $f_1$  and  $f_2$  are the conflicting objectives of the problem;  $g$  is a single-objective function that controls the complexity of the objective-space (particularly with respect to modality and uniformity); and  $h$  is a function that dictates the shape and continuity of the Pareto optimal front (with  $f_1$  affecting the density of points along that front) [148]. A Pareto optimal point is formed by using *any* legal value for  $x_1$  and setting all remaining decision variables to zero.

It is this last point that has drawn criticism from the field. It has been argued that there is a disproportionate amount of selective pressure on the first objective and that the search algorithm is principally tasked with minimising  $f_2$  [97]. Moreover, by making the final  $m-1$  decision variables optimal at the value of zero, there exists an intrinsic bias towards the extreme portions of the decision-space. The consequence is that algorithms with variation procedures which tend towards boundary solutions will be granted an unfair advantage that is unlikely to prevail in typical domains. Non-pathological examples exist where performance results were fundamentally

affected by such a property — including a work by this author where the promise of a new algorithm was overstated due to a subtle bias in the variation procedure [169]<sup>13</sup>.

Still, the impact of this problem can be effectively nullified if the practitioner is careful to ensure that all examined algorithms feature variation techniques that are fixed and of known type. In this case, the comparison will at least occur on even footing. This thesis provides further precautions by modifying most of the selected *ZDT* functions such that the optimal values for many of the decision variables are displaced (see Section 5.3).

The simple nature of the Pareto optimal set in solution-space also leads to an interesting side-effect rarely noted in the literature. The requirement for identical values across a large number of dimensions means that a search will face considerable levels of collateral noise, where a single incorrect decision variable can affect the perceived utility of an otherwise useful solution. Consider the two solutions proposed in Figure 21 as applied to the *ZDT1* function. The first approach features nine of the ten decision variable values required to form a single point on the Pareto optimal front, while solution *b* includes only one (which will invariably occur for *any* generated solution for a *ZDT* problem). Moreover, while the second proposal lacks any decision variable settings that will feature in the *entire* Pareto optimal front, solution *a* features eight. Still, despite the high utility of the vast majority of variables in solution *a*, the collateral noise caused by the poor tenth variable means that it is dominated by the generally weaker second solution. The presence of such noise in the *ZDT* function suite affects not only the ability of an optimiser to locate a single Pareto optimal point, but also to develop the entire Pareto optimal set.

0.5	0	0	0	0	0	0	0	0	0.5	a
0.5	0.04	0.06	0.05	0.05	0.05	0.05	0.04	0.05	0.05	b

**Figure 21 — Illustrating Collateral Noise in *ZDT1***

*a* is inferior to *b*, despite featuring nine of the ten parameter values required for a Pareto optimal solution (*b* includes only one parameter common to a Pareto optimal solution).

<sup>13</sup> A full exploration of this issue will be the topic of a future study and is beyond the scope of this thesis.

### 5.1.2 CTP FUNCTIONS

The Constraint Test Problem (*CTP*) test suite [164] describes a flexible collection of constrained problems. Unlike other constraint-based suites (such as the generalised adaptation of Tanaka *et al.*'s [170] constrained problem — see [171] for details), the *CTP* collection allows for problems of high search-space dimensionality and facilitates complexities both near-to *and* beyond the Pareto optimal region — with potential for partially or fully obscured fronts and infeasibility holes (discontinuities caused by infeasible spaces). The flexibility and power of the suite should enable thorough testing of multiobjective search algorithms in the presence of various constraint types.

#### 5.1.2.1 ANALYSING THE CTP SUITE

While the overall nature of each constraint problem is ultimately distinct, there exist some commonalities in problem construction. All tests feature low-dimensional objective-spaces with an at least partially infeasible convex Pareto optimal front, obscured by the existence of constraints. The search spaces similarly include discontinuities emergent from the inclusion of constraints.

The *CTP* functions used in this composite suite are of the generic form:

$$\begin{aligned}
 &\text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}) \\
 &f_1(\mathbf{x}) = f_1(x_1) = x_1 \\
 &f_2(\mathbf{x}) = g(x_2, \dots, x_m) \times h(f_1(x_1), g(x_2, \dots, x_m)) \\
 &g(x_2, \dots, x_m) = 1 + \sum_{i=2}^m x_i^2 \\
 &h(f_1, g) = 1 - \sqrt{f_1 / g} \\
 &\text{where } \mathbf{x} = (x_1, x_2, \dots, x_m)
 \end{aligned} \tag{19}$$

Importantly, a solution  $\mathbf{x}$  is considered feasible only if Equation (20) holds true for selected real-valued parameters,  $a, b, c, d, e$  and  $\theta$ :

$$\begin{aligned}
 &(\cos(\theta) \times (f_2(\mathbf{x}) - e) - \sin(\theta) \times f_1(\mathbf{x})) \geq \\
 &\left( a \times \left| \sin(b\pi \times (\sin(\theta) \times (f_2(\mathbf{x}) - e) + \cos(\theta) \times f_1(\mathbf{x}))^c \right|^d \right)
 \end{aligned} \tag{20}$$

where  $a$  dictates the distance from the true Pareto optimal front that constraint-induced discontinuities begin to occur;  $b$  controls the number of disconnected infeasible regions;  $c$  specifies the distribution of discrete Pareto optimal regions in objective-space ( $c=1$  infers no bias;  $c>1$  favours objective one;  $c<1$  favours objective two);  $d$  defines the width of feasible regions;  $e$  shifts the location of the constraints up or down in objective-space; and  $\theta$  influences the slope of the Pareto optimal front.

The  $g$  and  $h$  functions are identical in motivation to those provided in the *ZDT* suite — namely, they control the shape and complexity of the fitness space. While both functions — and thus the true nature of the *CTP* problems — remain unspecified in the original work by Deb *et al.* [164], graphical illustrations infer convex spaces that are similar, if not identical, to those suggested here in Equation (19).

As should be reasonably obvious from the sheer number of parameters required in the definition of a *CTP* function, the flexibility of the approach is impressive. Such malleability allows for the definition of functions that test the power and adaptability of optimisers in a wide-array of constrained domains. In particular, the suite contains functions where the exploration of infeasible regions is of minimal value — where the vast majority of infeasible space represents a fools-errand, seldom leading to anything of value to the user. In contrast, other problems benefit from the aggressive exploration of infeasible areas, since optimal regions lie isolated beyond these spaces. How a search algorithm balances these conflicting demands is ultimately at the heart of constraint-based research and performance on these problems reflects not only the suitability of algorithms in general, but also any search bias that may exist — be it an inclination towards fearless intrusion into infeasible space or a more conservative approach.

### 5.1.3 FDA FUNCTIONS

An extension of the static problems illustrated in the works of Zitzler *et al.* [92] (see Section 5.1.1) and Deb [148], the *FDA* test suite (produced by Farina, Deb and Amato [41, 172]) provides a range of functions that exemplify the utility of multiobjective search algorithms in dynamic environments. Given the number of ways that a problem may vary over time (a subset of which is discussed by Jin and

Sendhoff [154]), the *FDA* suite is inevitably incomplete, but none-the-less illustrates a range of interesting dynamic properties that test the ability of a search algorithm in shifting objective- and decision-spaces. Thus, search algorithms that perform well on the functions merit further examination in dynamic domains.

### 5.1.3.1 ANALYSING THE FDA SUITE

The form of an *FDA* problem is expressed in a similar manner to the *ZDT* problems, tailored to facilitate the time-dependencies necessary in a dynamic problem. Specifically, Equation (24) illustrates the generic construction of an *FDA* problem, with  $t_c$  specifying the current time counter;  $t_t$  representing the total number of timer-increments required for an increase in time-step  $t$ ; and  $n_t$  indicating the number of unique states (steps) available to a problem.

$$\begin{aligned}
 &\text{Minimise } f_1(\mathbf{x}, t), f_2(\mathbf{x}, t) \\
 &f_2(\mathbf{x}, t) = g(\mathbf{x}, t) \times h(\mathbf{x}, f_1(\mathbf{x}, t), g(\mathbf{x}, t), t) \\
 &\text{where } \mathbf{x} = (x_1, x_2, \dots, x_m), \\
 &t = (1/n_t) \times \lfloor t_c / t_t \rfloor, t_t = 5, n_t = 10
 \end{aligned} \tag{21}$$

Though Farina *et al.* [41] readily admit that their suite fails to examine less common dynamic features (including changes in the number of objectives) the real-valued functions are impressive in their diversity and offer an important window into the performance of algorithms in variable domains. In particular, the suite includes functions with fixed and moving optimal regions in both decision- and objective-spaces, while providing mechanisms for variation in solution density along the optimal front. Furthermore, while not explicitly explored by Farina *et al.* [41], empirical analyses (see Section 5.3.6) illustrate that the spaces in which the optimal regions reside are also subject to change over time.

### 5.1.4 WFG FUNCTIONS

At the time of writing, the most contemporary, and amongst the most interesting, of the available test suites is the Walking Fish Group (*WFG*) collection produced by Huband *et al.* [158]. Covering the most diverse set of unconstrained problem characteristics of any available test bed, the suite brings non-separability and degenerative fronts, in particular, to the fore.

#### **5.1.4.1 ANALYSING THE WFG SUITE**

The production of a *WFG* problem is achieved by combining pre-defined shape and transformation functions housed in a flexible and powerful toolkit. Use of the toolkit facilitates the construction of static unconstrained test functions with linear, concave, convex, disconnected or mixed frontal shapes, and objective-spaces with flat-regions, deception, bias, multi-frontality and degeneration. When coupled with the capacity for non-separability, the toolkit is sufficient for the production of an impressive array of challenging functions — a subset of which is the *WFG* benchmark suite and the illustrative *I* functions.

While successful in modelling a rich set of problem characteristics, the provided functions could be criticised for a failure to focus on isolated domain features. As such, the proposed problems are useful approximations of real-world systems — where interaction between distinct domain characteristics is likely — but are generally less helpful in elucidating the behaviour of a given search strategy on a particular class of problem.

### **5.2 DESCRIBING TEST FUNCTIONS**

Given that performance on test problems forms the corner-stone of search algorithm analysis and comparison, it is surprising that the problems themselves are rarely discussed in any great detail outside of the work in which they were originally presented. Moreover, literature investigating the nature of multiobjective problems is traditionally focussed upon the properties of the objective-space, with solution-space characteristics seldom considered beyond the Pareto set (see, for instance, [92, 172]).

While the analysis of objective-spaces alone can provide insight, the mapping from solution-space into objective-space is equally significant. An objective-space may appear to be multi-frontal, biased and constrained, but that says nothing of the complexity of the problem if an obvious and simple search gradient exists in solution-space. Indeed, there can be no guarantee that, in general, any property viewed in objective-space need necessarily reflect an equivalent quality in the decision-space; thus an analysis of one space alone is likely insufficient. Since the goal of all test problems is, at the very least, to reflect the likely behaviour of algorithms in similar real-world environments, painting such an incomplete picture is

dangerous indeed. Improvement or decline in optimiser performance may be due to an objective-space property, it may be the consequence of a hidden search-space characteristic or it may be some interaction between hidden and known problem attributes.

Without a full understanding of the problem at hand, it is difficult to speak with certainty about which domains are suited to the tested optimiser or the behavioural trends and adaptability of the search algorithm. As such, this work provides an investigation both into the objective-space and its Pareto-optimal characteristics, and the mapping that exists from solution-space to the objective-space.

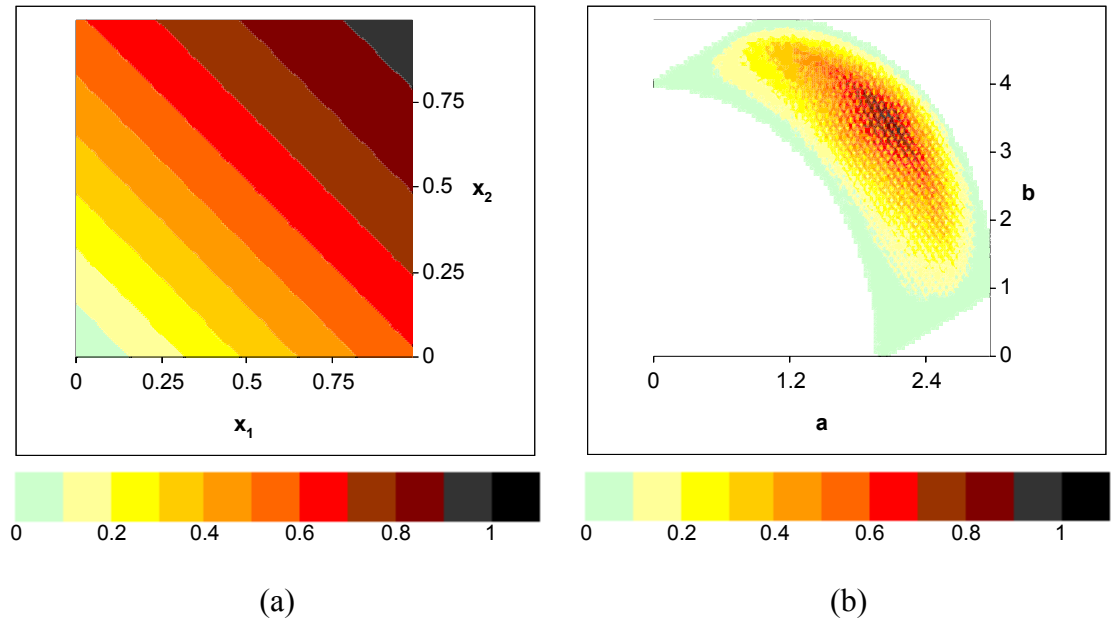
To ensure a suitably thorough exploration, functions should be examined through visualisations, textual descriptions and mathematical formulae. Note that when considered independently, each of the chosen descriptive approaches is largely insufficient: visualisations can be misleading and are subject to interpretation; textual overviews are frequently imprecise; and mathematical formulae are often inaccessible. However, when viewed *in toto*, the devices can offer a rich, multi-faceted, summation.

### **5.2.1 VISUALISATIONS**

This work offers a thorough set of visualisations designed to explore the properties of both the objective-space and solution-space. In both cases, there is a heavy emphasis on the use of spectral frequency graphs — a technique that is being used in this context for the first time.

The two-dimensional spectral frequency graphs employed here are simple grids, where the value of each cell is encoded as a particular colour. If the colours are mapped such that they follow a gradient from low values to high, the resultant grid offers a clear and concise summary of the values contained within it. As an example, consider Figure 22a: it is clear that the minimum occurs when  $x_1 = x_2 \approx 0$ , that the maximum occurs when  $x_1 = x_2 \approx 1$  and that there exists an obvious gradient that moves from the upper-right to the lower-left. If  $x_1$  and  $x_2$  are unique decision variables and each cell represents the average objective-performance for the range of solution-values represented by that cell, then the spectral frequency graph indicates





**Figure 22 — Example Spectral Frequency Graphs**

the utility-gradient that exists between the two variables for the selected objective. Importantly, the use of such graphs can emphasise the existence of multi-frontality, deception, bias and discontinuities.

The spectral graphs may also be used to explore the properties of the objective-space. If  $a$  and  $b$  in Figure 22b are distinct objectives and each cell contains the frequency with which that region is produced from an even sampling of the solution-space, then (assuming a suitably fine sampling) the graph illustrates the shape of the feasible objective-space and the influence of bias, isolation, discontinuities and deception. Thus, Figure 22b illustrates a concave objective-space with a bias away from the concave Pareto optimal front.

For the purposes of this work, each solution-space grid contains 40,000 evenly distributed cells, with values obtained by examining the objective-performance of at least 1,000,000 unique, evenly distributed, decision-variable pairs. The objective-space grids contain 20,000 cells, with frequencies obtained by sampling at least 8,000,000 unique points in solution-space. Since it is often infeasible to sample all solution-variables with suitable accuracy, the objective-space frequency grids typically examine only a subset of all possible variable combinations. As such, the objective-space graphs are indicative, rather than definitive, though care has been

taken to ensure that they display features consistent with a more complete investigation. For all spectral frequency graphs in this chapter, cell values are normalised and mapped to the colour-key provided in Figure 22b.

### **5.2.2 TEXTUAL DESCRIPTIONS**

Perhaps the most common, and accessible, of all available descriptive tools, textual overviews are useful in both emphasising key points and identifying smaller properties that may not be initially apparent in mathematical or visual characterisations. As such, the textual descriptions offer a useful reference, but generally lack the depth required for a full understanding of the problem at hand. In particular, an overview may point to bias in the problem — but the true extent of such bias and its position in space is only adequately addressed in a visual or mathematical framework.

### **5.2.3 MATHEMATICAL FORMULAE**

The inclusion of full mathematical formulae is important for two reasons: it provides a complete, though potentially complex, view of the problem that neither textual descriptions nor visualisations can ever hope to offer; and it facilitates the use of the suite by other members of the community, which is particularly important given the modifications made to existing problems in this work.

## **5.3 THE NEW ALTERNATIVE PROBLEM SUITE**

Having defined how the suite will be analysed, it is necessary to now identify and explore each function in the newly developed *AP* (Alternative Problem) suite. In the interests of clarity, this section principally offers mathematical equations and succinct textual descriptions that outline key problem characteristics. For a complete analysis, the reader is strongly encouraged to examine the rich set of additional resources provided in Appendix 0 (including spectral graphs for each problem and further mathematical definitions).

### **5.3.1 FRONTAL SHAPE PROBLEMS — AP-1, AP-2 AND AP-3**

By offering shape as the principle point of differentiation in a grouping of simple and consistent problems, any variations in algorithmic performance can be directly attributed to the concavity or convexity of the leading front. *AP-1*, *AP-2* and *AP-3* provides such a crisply defined collection (derived largely from *ZDT1*, *ZDT2* and

*ZDT3* [92]). As illustrated in Figure A1, Figure A2 and Figure A3 (see Appendix A.1), each decision-space features obvious and non-deceptive search gradients that encourage the exploration of optimal regions, while the density of points in objective-space suggest a bias away from the ideal front (with the intensity of the bias tending to increase as the number of decision variables grow).

To ensure that optimal values do not simply exist at extreme points (as they do in *ZDT1*, *ZDT2* and *ZDT3*) the *s\_linear* function suggested by Huband *et al.* [158] is integrated into the original *ZDT* problems to shift the location of the optimal solution values for objective-two (see Appendix A.23.1 for definitions and descriptions of *s\_linear*). Specifically (as described in Appendix A.1, Appendix A.2 and Appendix A.3), a non-extreme (and non-medial) value of 0.35 for  $x_{2..m}$  will map the solution onto the Pareto optimal front, while variation in  $x_1$  dictates position on that front.

### 5.3.1.1 *AP-1 — CONVEX PARETO OPTIMAL FRONT*

Based on *ZDT1* from Zitzler *et al.* [92], *AP-1* (Equation (22)) features a convex Pareto optimal front embedded in a convex objective-space (see Figure A1). Given the fact that convexity is generally recognised as the least challenging of the frontal shapes (see Section 4.1), and since it lacks other domineering problem features, *AP-1* is considered the least difficult of all the provided test functions. Regardless, the problem will examine the ability of search algorithms, particularly those based on Pareto optimality, to find a well-distributed and accurate front that adequately includes extrema points in objective-space.

$$\begin{aligned}
 &\text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}) \\
 &f_1(\mathbf{x}) = x_1 \\
 &f_2(\mathbf{x}) = g(\mathbf{x}) \times h(\mathbf{x}) \\
 &\text{where:} \\
 &\mathbf{x} = (x_1, x_2, \dots, x_m) : x_{1..m} \in [0, 1], \quad m = 30 \\
 &g(\mathbf{x}) = 1 + \frac{9}{m-1} \times \sum_{i=2}^m \text{linear\_s}(x_i, 0.35) \\
 &h(\mathbf{x}) = 1 - \sqrt{f_1(\mathbf{x}) / g(\mathbf{x})}
 \end{aligned} \tag{22}$$

### 5.3.1.2 *AP-2 — CONCAVE PARETO OPTIMAL FRONT*

As per *ZDT2*, *AP-2* (Equation (23)) includes a purely concave Pareto optimal front situated in a concave objective-space (see Figure A2). While the problem itself is

considered simple in light of advances made in contemporary multiobjective research, it exposes weaknesses in linear function approximators and acts as a counter-point to the convex case — where the difficulty now comes in finding mutually beneficial, not extreme, compromises.

$$\begin{aligned}
 &\text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}) \\
 &f_1(\mathbf{x}) = x_1 \\
 &f_2(\mathbf{x}) = g(\mathbf{x}) \times h(\mathbf{x}) \\
 &\text{where:} \\
 &\mathbf{x} = (x_1, x_2, \dots, x_m) : x_{1..m} \in [0, 1], \quad m = 30 \\
 &g(\mathbf{x}) = 1 + \frac{9}{m-1} \times \sum_{i=2}^m \text{linear\_s}(x_i, 0.35) \\
 &h(\mathbf{x}) = 1 - (f_1(\mathbf{x}) / g(\mathbf{x}))^2
 \end{aligned} \tag{23}$$

### 5.3.1.3 AP-3 — DISCONNECTED CONVEX OPTIMAL FRONT

While the resultant Pareto optimal front for AP-3 (Equation (24), derived from ZDT3) is disconnected and convex, it is embedded within a mixed-shaped objective-space (see Figure A3). It is important to note that the discontinuity does not occur due to holes in the objective-space (as is possible in constraint based problems — see Section 5.3.5), nor is it a consequence of a discontinuous decision-space. Instead, it is the existence of poor-utility concave regions between each distinct convex Pareto-optimal segment (as evidenced in the decision and objective-spaces illustrated in Figure A3). Consequently, an effective search must avoid spending too much energy investigating sub-optimal concave areas, while also maintaining a solution set that is diverse enough to avoid convergence onto a subset of the total available optimal regions.

$$\begin{aligned}
 &\text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}) \\
 &f_1(\mathbf{x}) = x_1 \\
 &f_2(\mathbf{x}) = g(\mathbf{x}) \times h(\mathbf{x}) \\
 &\text{where:} \\
 &\mathbf{x} = (x_1, x_2, \dots, x_m) : x_{1..m} \in [0, 1], \quad m = 30 \\
 &g(\mathbf{x}) = 1 + \frac{9}{m-1} \times \sum_{i=2}^m \text{linear\_s}(x_i, 0.35) \\
 &h(\mathbf{x}) = 1 - \sqrt{f_1(\mathbf{x}) / g(\mathbf{x})} - (f_1(\mathbf{x}) / g(\mathbf{x})) \times \sin(10\pi f_1(\mathbf{x}))
 \end{aligned} \tag{24}$$

### 5.3.2 A MULTI-MODAL PROBLEM — AP-4

The *AP-4* function (Equation (25), derived from *ZDT4*) contains multiple convex pseudo-optimal fronts in a convex objective-space (see Figure A4). The multifrontality of the problem is most evident in the mapping of solution-space into the second objective (see Figure A4b), with the search gradient interrupted by many dense regions of low utility. While Zitzler *et al.* [92] claim that such interruptions produce a remarkable  $21^9$  local fronts, that figure is open to debate and subject to the definition of locality used. Indeed, given that many of the fronts are indistinguishable in objective-space [92], it seems that a more logical definition of multifrontality — that is, one which considers only spatially distinct and reasonably isolated fronts — would lead to a significantly smaller number. Still, irrespective of the degree of frontality exhibited by *AP-4*, it has proven to be amongst the most difficult of the *ZDT* functions (see [92, 139, 173-175]) and adequately tests the performance of algorithms in the presence of many (if not billions of) false fronts.

As in the frontal shape problems (Section 4.3), the original *ZDT4* is augmented by the *s\_linear* function (see Section A.23.1) to ensure a reduction in the number of extreme optimal solutions. Again, the Pareto optimal front is formed by setting  $x_{2..m}$  to 0.35 and varying  $x_1$ .

$$\begin{aligned}
 &\text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}) \\
 &f_1(\mathbf{x}) = x_1 \\
 &f_2(\mathbf{x}) = g(\mathbf{x}) \times h(\mathbf{x}) \\
 &\text{where:} \\
 &\mathbf{x} = (x_1, x_2, \dots, x_m) : x_{1..m} \in [0, 1], x_{2..m} \in [-5, 5], m = 10 \\
 &g(\mathbf{x}) = 10m - 9 + \sum_{i=2}^m \left( (\text{linear\_s}(x_i, 0.35))^2 - 10 \cos(4\pi \times \text{linear\_s}(x_i, 0.35)) \right) \\
 &h(\mathbf{x}) = 1 - \sqrt{f_1(\mathbf{x}) / g(\mathbf{x})}
 \end{aligned} \tag{25}$$

### 5.3.3 BIASED PROBLEM — AP-5

Derived from *ZDT6*, the *AP-5* function (Equation (26)) features a highly biased non-convex objective-space. The function is an interesting one and exhibits features that are largely divergent from the remainder of the *ZDT* suite — specifically, the complexity of the problem is principally attributable to difficulties in optimising the first objective and tuning the first decision variable. Indeed, the objective-space is heavily biased away from minimal values for the first objective, while the decision-

space features a largely isolated and narrow optimal objective one region with little gradient information to inform the search (see Figure A5). The objective-space implies that optimisation of the second objective may be equally difficult, given the density of points away from the optimal front. However, an examination of the decision-space mappings (refer to Figure A5), which feature a crisply defined and simple search gradient, suggest that this is at most a secondary issue. Thus, a successful optimiser must be able to address the biases in the first objective and capitalise on the obvious search gradient in the second objective to produce a well-distributed and accurate set of solutions.

As with the preceding portions of the *AP* suite, the *ZDT6* problem is adapted to include an *s\_linear* function (see Appendix A.23.1) designed to shift extreme values. Since much of the complexity for *AP-5* is derived from optimising objective-one, unlike previous problems, only the optimal location for  $x_1$  is relocated to 0.35, all other variables remain optimal at zero (in part to ensure a particularly simple search-gradient for objective-two and thus promote greater disparity between the first and second objectives). Still, it is important that researchers note that a large portion of the search-space is optimal at extreme values and care should therefore be taken to ensure that any algorithms under analysis on this problem share matching variation procedures.

$$\begin{aligned}
 &\text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}) \\
 &f_1(\mathbf{x}) = 1 - e^{-4x_1} \times \sin^6(6\pi \times \text{linear\_s}(x_1, 0.35)) \\
 &f_2(\mathbf{x}) = g(\mathbf{x}) \times h(\mathbf{x}) \\
 &\text{where:} \\
 &\mathbf{x} = (x_1, x_2, \dots, x_m) : x_{1..m} \in [0, 1], m = 10 \\
 &\mathbf{x}' = (x'_1, x'_2, \dots, x'_m) : x'_1 = \text{linear\_s}(x_1, 0.35), x'_{i=2..m} = x_i \\
 &g(\mathbf{x}) = 1 + 9 \times \left( \sum_{i=2}^m \frac{x_i}{m-1} \right)^{0.25} \\
 &h(\mathbf{x}) = 1 - (f_1(\mathbf{x}) / g(\mathbf{x}))^2
 \end{aligned} \tag{26}$$

### 5.3.4 NOISY PROBLEMS — AP-6 AND AP-7

Bui, Abbass and Essam [44], Buche *et al.* [10] and Babbar *et al.* [43] (amongst others) integrate noise into simple pre-existing functions, where the problem characteristics are, to a large extent, already known. Inspired by these works, *AP-6*

and *AP-7* are explicitly simple functions built around the well-described *AP-2* problem. Though it is likely that analysis on these problems alone is insufficient to fully elucidate behaviour in a noisy domain (false outliers and multiplicative noise, for instance, is not examined), they should draw out behavioural characteristics of optimisers in uncertain environments and suggest avenues for future investigation. Moreover, the simple techniques described for *AP-6* and *AP-7* can be applied to any of the remaining *AP* test functions to examine the effects of noise under the presence of more complex problem characteristics, while noise intensity and location can be varied by simple manipulation of the *gaussian* and *random* functions.

#### 5.3.4.1 *AP-6 — A GAUSSIAN-BASED NOISY FUNCTION*

*AP-6* (Equation (27)) exhibits a noisy concave Pareto optimal front embedded in a noisy concave objective-space (see Figure A6 and Figure A7). The true performance of a solution on each objective is obscured by a Gaussian perturbation, with each objective-score shifted by a randomly chosen factor taken from a Gaussian distribution with a mean of zero and standard deviation of 0.5. Since the noise is derived from a narrow Gaussian function, the true utility of each solution is likely to lie relatively near to the apparent value, making *AP-6* the least challenging of the two noisy problems proposed in this test suite.

$$\begin{aligned}
 &\text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}) \\
 &f_1(\mathbf{x}) = x_1 \times \text{gaussian}(0, 0.5) + x_1 \\
 &f_2(\mathbf{x}) = i(\mathbf{x}) \times \text{gaussian}(0, 0.5) + i(\mathbf{x}) \\
 &\text{where:} \\
 &\mathbf{x} = (x_1, x_2, \dots, x_m) : x_{1..m} \in [0, 1], m = 30 \\
 &g(\mathbf{x}) = 1 + \frac{9}{m-1} \times \sum_{i=2}^m \text{linear\_s}(x_i, 0.35) \\
 &h(\mathbf{x}) = 1 - (f_1(\mathbf{x}) / g(\mathbf{x}))^2 \\
 &i(\mathbf{x}) = g(\mathbf{x}) \times h(f_1(\mathbf{x}), g(\mathbf{x}))
 \end{aligned} \tag{27}$$

#### 5.3.4.2 *AP-7 — A NON-GAUSSIAN-BASED NOISY FUNCTION*

*AP-7* (Equation (28)) is as per *AP-6*, though rather than defining noise through a Gaussian function, the factor by which the objectives are shifted is instead derived from an evenly distributed set of random values, with a minimum of  $-1$  and maximum of  $1$  (see Figure A8 and Figure A9). Since this form of variation offers a

less powerful bias around the true utility of a given solution, the apparent performance of a proposal in *AP-7* is considerably less reliable than in *AP-6*.

$$\begin{aligned}
 &\text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}) \\
 &f_1(\mathbf{x}) = x_1 \times \text{random}(-1, 1) + x_1 \\
 &f_2(\mathbf{x}) = i(\mathbf{x}) \times \text{random}(-1, 1) + i(\mathbf{x}) \\
 &\text{where:} \\
 &\mathbf{x} = (x_1, x_2, \dots, x_m) : x_{1..m} \in [0, 1], m = 30 \\
 &g(\mathbf{x}) = 1 + \frac{9}{m-1} \times \sum_{i=2}^m \text{linear\_s}(x_i, 0.35) \\
 &h(\mathbf{x}) = 1 - (f_1(\mathbf{x}) / g(\mathbf{x}))^2 \\
 &i(\mathbf{x}) = g(\mathbf{x}) \times h(f_1(\mathbf{x}), g(\mathbf{x}))
 \end{aligned} \tag{28}$$

### 5.3.5 PROBLEMS WITH SIDE-CONSTRAINTS — AP-8, AP-9 AND AP-10

In order to properly test the general performance of a multiobjective optimiser in a constrained domain, it is necessary to gauge system response against a range of differing constraint-based properties. In particular, the position, number and orientation of infeasible regions will each affect the behaviour and desirable properties of an optimiser and a suitable collection of tests must therefore vary each of these characteristics. The succinct set of *AP-8*, *AP-9* and *AP-10* functions (derived from the *CTP* suite) seek to emphasise such variation with a view to elucidating the performance of algorithms in general constrained domains<sup>14</sup>.

#### 5.3.5.1 AP-8

Drawing on the *CTP2* problem, *AP-8* (Equation (19) with settings as per Equation (29)) represents a convex objective-space with a large infeasible region that fully obscures the true convex Pareto optimal front (see Figure A10). The resultant feasible optimal front is linear and, due to peaks of infeasibility, discontinuous in objective-space. An analysis of the decision-space mappings (again, refer to Figure A10) illustrates that the majority of the search area is infeasible, with discontinuities making the pursuit of search gradients difficult — particularly with respect to objective-two, where infeasibilities obscure much of the optimal region. Such complexity is somewhat offset by the objective-space bias towards the feasible

---

<sup>14</sup> It is worth noting here that *AP-8*, *AP-9* and *AP-10* may represent the use of hard or soft constraints — it is the analysis of performance that will change, rather than the formation of each function.



optimal front and the fact that the constraints ultimately affect only regions surrounding that front.

$$a = 0.2, b = 10, c = 1, d = 6, e = 1, \theta = -0.2\pi \quad (29)$$

### 5.3.5.2 AP-9

While the true convex Pareto optimal front is fully obscured by infeasibilities in both *AP-8* and *AP-9* (Equation (19) with settings as per Equation (30)), the latter is differentiated by a continuous linear feasible optimal front that is isolated by a series of infeasible regions (see Figure A11). Each of these areas form discontinuities in both the decision- and objective-spaces, particularly impacting the search gradient that exists for objective-two (as exemplified in Figure A11b). The effect is akin to that seen in multi-frontal problems, with potential for search algorithms to become stranded in the narrow bands of sub-optimal feasible space. Again, the complexity of the problem is somewhat offset by an increased objective-space bias around the optimal feasible front.

It is interesting to note that *AP-8* and *AP-9* offer a sharp contrast and likely favour differing types of search strategy. For instance, to penetrate the discontinuities in the search gradient, an algorithm that features aggressive exploration of infeasible space is likely to perform well in *AP-9*. However, such disregard for constraints is unlikely to yield positive returns in *AP-8*, since the vast majority of infeasible space leads to an evolutionary dead-end. Thus, performance on these two problems should indicate the types of search bias (implicit or otherwise) that exist in constraint-based optimisers.

$$a = 40, b = 0.5, c = 1, d = 2, e = -2, \theta = 0.1\pi \quad (30)$$

### 5.3.5.3 AP-10

*AP-10* (Equation (19) with settings defined in Equation (31)) is the only one of the constrained problems that features a *partially* obscured true convex Pareto optimal front (see Figure A12). Moreover, the function features frequent narrow bands of infeasibility throughout the *entirety* of the objective- and decision-spaces, causing particular difficulties in the optimisation of the first objective (see Figure A12(a)) and inducing a large number of false fronts. Clearly this represents the most complex of the provided constraint-based problems and should indicate the

performance of algorithms in domains where the constraints have a great impact on the entirety of the objective-space.

$$a = 40, b = 20, c = 1, d = 6, e = 0, \theta = -0.05\pi \quad (31)$$

### 5.3.6 DYNAMIC PROBLEMS — AP-11, AP-12 AND AP-13

The range of potential characteristics that may be exhibited by a dynamic problem are many and it is not the goal of *AP-11*, *AP-12* and *AP-13* to offer a complete representation of those characteristics. Instead, these problems are selected to draw out the core properties of dynamic domains such that the fundamental strengths and weaknesses of multiobjective optimisers in shifting landscapes may be elucidated. Specifically, the functions offer an illustration of performance in domains with moving Pareto optimal fronts, variable objective-spaces and changing solution-spaces. Moreover, the rich analysis of each problem provided herein offers a considerably more thorough insight into the nature of dynamic functions than presently exists in the literature, which should further aid researchers in qualifying algorithmic performance.

#### 5.3.6.1 AP-11

Though the Pareto optimal front for *AP-11* (Equation (32), derived from *FDAI*) is a convex surface fixed in objective-space, the optimal portion of solution-space shifts over the course of a run. Indeed, the optimal  $x_{2..m}$  values vary sinusoidally with time according to Figure A14, while the search gradient for the second objective transforms dramatically (see Figure A13). Moreover, while the problem features a statically defined convex Pareto optimal front in objective-space, it is embedded in a varying convex space — with spatial extent oscillating between the minimum at  $t=0$  and the maximum at  $t=1$  (refer to Figure A13 and Figure A14). Also note that the position of the objective-space bias, which is always removed from the optimal front, oscillates at the same frequency— further augmenting the complexity of the problem over time.

$$\begin{aligned}
& \text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}, t) \\
& f_1(\mathbf{x}) = x_1 \\
& f_2(\mathbf{x}, t) = g(\mathbf{x}, t) \times h(\mathbf{x}, t) \\
& \text{where} \\
& \mathbf{x} = (x_1, x_2, \dots, x_m) : x_1 \in [0, 1], x_{2..m} \in [-1, 1], m = 20 \\
& g(\mathbf{x}, t) = 1 + \sum_{i=2}^m (x_i - F(t))^2 \\
& h(\mathbf{x}, t) = 1 - \sqrt{f_1(\mathbf{x}) / g(\mathbf{x}, t)} \\
& F(t) = \sin(0.5\pi t) \\
& t = 0.1 \times \lfloor t_c / 5 \rfloor
\end{aligned} \tag{32}$$

### 5.3.6.2 AP-12

A newly defined function (Equation (33), inspired partially by *FDA2*), *AP-12* is designed to offer a *varying* Pareto optimal front in objective-space, with a *static* optimal solution set (see Figure A15 and Figure A16). Specifically, the projection of the optimal front changes sinusoidally according to Figure A16e, moving through linear, concave and convex surface shapes. In contrast, the decision-space mappings (refer to see Figure A15) remain largely unaffected by the variations in objective-space (though there are subtle shifts in bias for objective-two). The problem will test the capacity of an optimiser to pursue relatively stable search-gradients under a shifting objective-space landscape.

It is worth noting that another member of the *FDA* suite could be used directly in place of this function, though it is not advised. All of the proposed real-valued *FDA* problems (and indeed, those problems also proposed by Jin and Sendhoff [154]) feature time-dependent optimal solution sets in decision-space — to properly test the performance of optimisers under distinct forms of dynamism, it is necessary to also examine dynamic problems that lack this feature, lest the comparison favour optimisers with a bias towards shifting decision-spaces.

$$\begin{aligned}
 &\text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}, t) \\
 &f_1(\mathbf{x}) = x_1 \\
 &f_2(\mathbf{x}, t) = g(\mathbf{x}) \times h(\mathbf{x}, t) \\
 &\text{where} \\
 &\mathbf{x} = (x_1, x_2, \dots, x_m) : x_1 \in [0, 1], x_{2..m} \in [-1, 1], m = 20 \\
 &g(\mathbf{x}) = 1 + \frac{9}{m-1} \left( \sum_{i=2}^m s\_linear(x_i, 0.35) \right) \\
 &h(\mathbf{x}, t) = 1 - (f_1(\mathbf{x}) / g(\mathbf{x}))^{H(t)} \\
 &H(t) = 2^{\sin(0.75\pi t)} \\
 &t = 0.1 \times \lfloor t_c / 5 \rfloor
 \end{aligned} \tag{33}$$

### 5.3.6.3 AP-13

The most difficult of the dynamic *AP* problems, *AP-13* — a newly designed function inspired by *FDA3* — features variation in both the solution- and objective-spaces (see Equation (34)). Specifically, the Pareto optimal front varies sinusoidally through linear, concave and convex shapes in objective-space as per *AP-12*, while the space itself has a shifting bias defined by the *G* function (Figure A17, Figure A18 and Figure A19). Though the ideal solution set is fixed for objective-two, the optimal values for the first objective move according to the cosine *F* function (as per Figure A19c), with constantly shifting search gradients. Thus, the problem as a whole features dynamism both near-to and away-from the Pareto optimal front in objective-space and in the properties of the decision-space — challenging an optimiser throughout the entirety of the search process.

It is worth noting that the *FDA3* function could have been used directly in this circumstance, though it lacks variability in the shape of the optimal front and features particularly aggressive levels of bias. While the function is interesting, such extreme bias is likely to test the performance of an optimiser in prejudiced domains as much as it is likely to provide insight into the behaviour of an algorithm under shifting search and objective landscapes. Additionally, the newly formed *AP-13* function allows for the investigation of algorithm performance when problem characteristics are varying at differing rates (see Figure A19c).

Minimise  $f_1(\mathbf{x}, t), f_2(\mathbf{x}, t)$

$$\begin{aligned}
 f_1(\mathbf{x}, t) &= \left( \frac{x_1 - F(t)}{[F(t) - x_1] + F(t)} \right)^{G(t)} \\
 f_2(\mathbf{x}, t) &= g(\mathbf{x}) \times h(\mathbf{x}, t) \\
 \text{where:} \\
 \mathbf{x} &= (x_1, x_2, \dots, x_m) : x_1 \in [0, 1], x_{2..m} \in [-1, 1], m = 20 \\
 g(\mathbf{x}) &= 1 + \frac{9}{m-1} \left( \sum_{i=2}^m s\_linear(x_i, 0.35) \right) \\
 h(\mathbf{x}, t) &= 1 - (f_1(\mathbf{x}, t) / g(\mathbf{x}, t))^{(H(t))} \\
 F(t) &= 0.5 + \frac{\cos(0.5\pi t)}{4} \\
 G(t) &= 1 + \frac{\cos(\pi t)}{1.5} \\
 H(t) &= 2^{\sin(0.75\pi t)}
 \end{aligned} \tag{34}$$

#### 5.3.6.4 NON-SEPARABLE PROBLEMS — AP-14 AND AP-15

A non-separable problem infers the existence of dependencies between the decision variables of proposed solutions. The effect of non-separability on algorithmic performance is likely tied to the degree to which such dependencies occur<sup>15</sup> — increasing the number of links from a single variable to other variables makes optimisation of that parameter increasingly difficult, while expanding the total number of links that exist across the solution as a whole is likely to disrupt the tuning of the entire proposal. With this in mind, AP-14 and AP-15 are offered to explore performance under simple, low-dependency, non-separable problems and complex, highly interconnected, domains.

#### 5.3.6.5 AP-14

Derived from the first, and easiest, of the non-separable *WFG* [158] functions (*WFG2*), AP-14 (Equation (35)) features simple bi-directional (mutual) dependencies between pairs of neighbouring decision variables in the last  $l$  parameter positions (see Figure A21 for illustrations and descriptions, and Appendix A.23.3 for an insight into the behavioural characteristics of the pivotal *interdepend* function). In order for an optimiser to perform efficiently, any search process should maintain these dependencies when varying or generating new solutions, lest the effects of

<sup>15</sup> Though this is yet to be explored thoroughly in the literature and is an interesting avenue of work, particularly given the claim that many real-world problems are non-separable.

collateral noise slow the process. However, since the number and complexity of the dependencies will generally be low (for all but very high values of  $l$ ), even a search process with limited relationship maintenance should still be able to converge to the mixed-shape Pareto optimal front with only minor efficiency losses.

The objective-space includes biased regions that are removed from the Pareto optimal front (see Figure A20), though the lack of a heavily affected search gradient (again, see Figure A20) suggests that the impact of such uneven density is likely to be minimal<sup>16</sup>.

Minimise  $f_1(\mathbf{x}), f_2(\mathbf{x})$

$$f_1(\mathbf{x}) = x_2'' + 2 \sin\left(\frac{\pi x_1''}{2}\right)$$

$$f_2(\mathbf{x}) = x_2'' + 4 \cos\left(\frac{\pi x_1''}{2}\right)$$

where:

$$\mathbf{x} = (x_1, x_2, \dots, x_k, x_{k+1}, \dots, x_m) : m = 60, \quad k = 10, \quad l = 50, \quad x_{i=1..m} \in [0, 2i]$$

$$\mathbf{x}' = (x'_{1..k+l/2}) :$$

$$x'_{i=1..k} = x_i / 2i,$$

$$x'_{i=k+1..k+l/2} = \text{interdepend} \left\{ \begin{array}{l} \text{s\_linear} \left( \frac{x_{k+2(i-k)-1}}{2k+4(i-k)-2}, 0.35 \right), \\ \text{s\_linear} \left( \frac{x_{k+2(i-k)}}{2k+4(i-k)}, 0.35 \right) \end{array} \right\}$$

$$\mathbf{x}'' = (x_1'', x_2'') : x_1'' = \sum_{i=1}^k x_i / k, \quad x_2'' = \sum_{i=k+1}^m x'_i / l$$

(35)

### 5.3.6.6 AP-15

Unlike AP-14, where dependencies were strictly localised and existed only directly between pairs of variables in a subset of the entire solution, AP-15 (Equation (36), inspired by WFG6) features linkages throughout the entire proposal and numerous types of non-separability via the new *nonsep\_special* function (see Appendix A.23.5). In particular, mutual, non-mutual, direct and indirect dependencies are present, with the first  $k$  decision variables affected by other members of  $x_{1..k}$ , and the final  $l$  parameters dependant on constituents of  $x_{1..m-k}$  (see Figure A22 for illustrations

<sup>16</sup> Even with minimal impact, it may be beneficial to reduce the objective-space bias to further narrow the focus onto the non-separable domain. A similar goal may also be sought for AP-15. This is left as future work.

and more exacting descriptions). Collateral noise is extremely high in this case: where a poor decision variable in *AP-14* can disrupt the apparent utility of at most one other variable, a similarly poor choice may mask the efficacy of a large segment of the entire solution in *AP-15*.

In terms of objective-space features, *AP-15* is a slightly biased concave space with a continuous Pareto optimal front (see Figure A22). As with *AP-14*, the search-space is simple when excluding the impact of non-separability, with uni-modality and a lack of deception or bias<sup>17</sup>. As such, the principle characteristic affecting performance is the scope and intricacy of the dependencies between variables.

Minimise  $f_1(\mathbf{x}), f_2(\mathbf{x})$

$$f_1(\mathbf{x}) = x_2'' + 2 \sin\left(\frac{\pi x_1''}{2}\right)$$

$$f_2(\mathbf{x}) = x_2'' + 4 \cos\left(\frac{\pi x_1''}{2}\right)$$

where:

$$\mathbf{x} = (x_1, x_2, \dots, x_k, x_{k+1}, \dots, x_m) : m = 60, k = 10, l = 50, x_{i=1..m} \in [0, 2i] \quad (36)$$

$$\mathbf{x}' = (x'_{1..m}) : x'_{i=1..k} = x_i / 2i, x'_{i=k+1..m} = \text{s\_linear}\left(\frac{x_i}{2i}, 0.35\right)$$

$$\mathbf{x}'' = (x''_1, x''_2) :$$

$$x''_1 = \text{nonsep\_special}(\{x'_1, x'_2, \dots, x'_k\}, \{x'_1, x'_2, \dots, x'_k\}, 3)$$

$$x''_2 = \text{nonsep\_special}(\{x'_{k+1}, x'_{k+2}, \dots, x'_m\}, \{x'_1, x'_2, \dots, x'_{(m-k)}\}, 3)$$

### 5.3.7 A PROBLEM WITH ZERO-UTILITY GRADIENTS — AP-16

*AP-16* (Equation (37)) elucidates the performance of optimisation algorithms in domains with portions of space that provide very little search information. Though inspired by *WFG1* (and its use of the *b\_flat* function described in Appendix A.23.2), *AP-16* features a greater focus on flat fitness-spaces due to the exclusion of explicit biasing in the first  $k$  parameters and an increase in the presence of zero-utility gradients. Specifically, values in the final  $l$  decision variables are now recast according to Figure A24a such that, for a great majority of the space (over 70% of all values are mapped to one of two objective-space points), perturbation leads to no tangible change in solution quality.

<sup>17</sup> Note that the complex inter-relationships between variables make a concise analysis of the solution-space mappings via spectral performance graphs (as used throughout this section) infeasible.

The problem also examines the balance of an algorithm when addressing a problem with mixed search granularity requirements — clearly, a more coarse search better enables escape from the zero-utility portions of the search-space, but the small optimal region (see Figure A23d, in particular) may cause difficulties for end-of-run convergence under such a configuration. In contrast, a more fine-grained search better enables the location of optimal-space in the final  $l$  decision parameters, but can lead to considerably more laborious movement through the flat regions of space. Thus, an analysis of algorithmic performance during early and late portions of a run may draw-out the search-granularity characteristics of differing optimisation approaches.

As should be expected with most-any zero-utility gradient problem, the objective-space is biased<sup>18</sup> — in this case, away from the most promising regions in continuous bands that echo the mixed-shape Pareto optimal front. This work notes that any optimiser that is endeavouring to efficiently address a search-space with zero-utility gradients must also be resistant to such bias in the objective-space.

Minimise  $f_1(\mathbf{x}), f_2(\mathbf{x})$

$$f_1(\mathbf{x}) = x_2'' + 2 \left( 1 - \cos \left( \frac{\pi x_1''}{2} \right) \right)$$

$$f_2(\mathbf{x}) = x_2'' + 4 \left( 1 - x_1'' - \frac{\cos(10\pi x_1'' + \pi/2)}{10\pi} \right)$$

where:

$$\mathbf{x} = (x_1, x_2, \dots, x_k, x_{k+1}, \dots, x_m) : m = 60, \quad k = 10, \quad l = 50, \quad x_{i=1..m} \in [0, 2i] \quad (37)$$

$$\mathbf{x}' = (x'_{1..m}) :$$

$$x'_{i=1..k} = x_i / 2i,$$

$$x'_{i=k+1..m} = \text{b\_flat} \left( \left( \text{b\_flat}(\text{s\_linear}(x_i / 2i, 0.35), 0.2, 0.05, 0.45) \right), 0.7, 0.5, 0.95 \right)$$

$$\mathbf{x}'' = (x_1'', x_2'') : x_1'' = \sum_{i=1}^k x_i / k, \quad x_2'' = \sum_{i=k+1}^m x_i / l$$

### 5.3.8 A DECEPTIVE PROBLEM — AP-17

The *WFG5* function — represented here as *AP-17* (Equation (38)) — is an extremely deceptive problem (due to its use of the *s\_decept* transformation described in Appendix A.23.4), with search-spaces offering gradients that tend to encourage the

<sup>18</sup> Note that the reverse is not true: biased objective-spaces need not come from problems that feature zero-utility gradients.



exploration of areas well-removed from the optimal point (see Figure A25). Indeed, objective-two is biased away from optimal space in all bar the final  $l$  decision variables, while objective-one is deceptive in every parameter. Moreover, the deception is not just prevalent, but strong — the optimal point (0.35 for normalised variables) is fully obscured by very low-utility regions in Figure A25a, Figure A25b and Figure A25c (due to the coarse quantisation) and is suggested by only an incredibly narrow localised gradient (as seen in Figure A26a). Should an algorithm pursue the more obvious gradient — it is likely to form globally inferior (though locally impressive) fronts using  $x_{k+1..m} = 0$  or  $x_{k+1..m} = 1$ .

$$\begin{aligned}
 &\text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}) \\
 &f_1(\mathbf{x}) = x_2'' + 2 \sin\left(\frac{\pi x_1''}{2}\right) \\
 &f_2(\mathbf{x}) = x_2'' + 4 \cos\left(\frac{\pi x_1''}{2}\right)
 \end{aligned} \tag{38}$$

where:

$$\mathbf{x} = (x_1, x_2, \dots, x_k, x_{k+1}, \dots, x_m) : m = 60, \quad k = 10, \quad l = 50, \quad x_{i=1..m} \in [0, 2i]$$

$$\mathbf{x}' = (x'_{1..m}) : x'_{i=1..m} = \text{s\_decept}(x_i / 2i, 0.35, 0.001, 0.05)$$

$$\mathbf{x}'' = (x''_1, x''_2) : x''_1 = \sum_{i=1}^k x_i / k, \quad x''_2 = \sum_{i=k+1}^m x_i / l$$

### 5.3.9 PROBLEMS WITH NUMEROUS OBJECTIVES — AP-18 AND AP-19

To examine the effects of the dimensionality curse (Section 4.2.7) on multiobjective optimisers it is necessary to examine performance degradation under an increasing number of objectives. With this in mind, *AP-18* and *AP-19* offer simple functions, where the key point of differentiation is in the dimensionality of the objective-space — specifically, *AP-18* is a three-objective problem, while *AP-19* features five dimensions. As such, any loss in performance from *AP-18* to *AP-19* is directly attributable to the dimensionality of the objective-space and should emphasise an algorithm's resistance to the dimensionality curse.

Note that since both functions are simply specialisations of Huband *et al.*'s *I1* problem [158], the interested reader may examine any number of dimensions — the choice of three and five objectives is largely arbitrary and used here primarily for illustrative purposes.

### 5.3.9.1 AP-18

As described in Figure A27 and Figure A28, AP-18 (see Equation (39)) features a simple three-dimensional concave Pareto optimal front embedded in a slightly biased three-dimensional objective-space. The solution-space is relatively straight-forward — lacking bias, dependencies and multi-frontality, while offering simple search-gradients.

$$\begin{aligned}
 &\text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}), f_3(\mathbf{x}) \\
 &f_1(\mathbf{x}) = x_3'' + \sin\left(\frac{\pi x_1''}{2}\right) \times \sin\left(\frac{\pi x_2''}{2}\right) \\
 &f_2(\mathbf{x}) = x_3'' + \sin\left(\frac{\pi x_1''}{2}\right) \times \cos\left(\frac{\pi x_3''}{2}\right) \\
 &f_3(\mathbf{x}) = x_3'' + \cos\left(\frac{\pi x_1''}{2}\right)
 \end{aligned} \tag{39}$$

where:

$$\mathbf{x} = (x_1, x_2, \dots, x_k, x_{k+1}, \dots, x_m) : m = 60, k = 10, l = 50, x_{i=1..m} \in [0,1]$$

$$\mathbf{x}' = (x'_1, x'_2, x'_3) :$$

$$x'_1 = \sum_{i=1}^{k/2} 2x_i / k, x'_2 = \sum_{i=k/2+1}^k 2x_i / k$$

$$x'_3 = \sum_{i=k+1}^m \text{s\_linear}((x_i / 2i), 0.35) / l$$

### 5.3.9.2 AP-19

Though it is difficult to visualise a problem consisting of five objectives, the decision- and objective-spaces (Figure A29, Figure A30, Figure A31 and Figure A32) illustrate that AP-19 (Equation (40)) features the same general properties as in AP-18 (particularly with respect to the type of search gradients encountered and the form of objective-space biasing found). It is again worth emphasising that, as in AP-18, the function features decision variables that are separable, without bias and uni-modal.

$$\begin{aligned}
& \text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}), f_3(\mathbf{x}), f_4(\mathbf{x}), f_5(\mathbf{x}) \\
& f_1(\mathbf{x}) = x'_5 + \sin\left(\frac{\pi x'_1}{2}\right) \times \sin\left(\frac{\pi x'_2}{2}\right) \times \sin\left(\frac{\pi x'_3}{2}\right) \times \sin\left(\frac{\pi x'_4}{2}\right) \\
& f_2(\mathbf{x}) = x'_5 + \sin\left(\frac{\pi x'_1}{2}\right) \times \sin\left(\frac{\pi x'_2}{2}\right) \times \sin\left(\frac{\pi x'_3}{2}\right) \times \cos\left(\frac{\pi x'_5}{2}\right) \\
& f_3(\mathbf{x}) = x'_5 + \sin\left(\frac{\pi x'_1}{2}\right) \times \sin\left(\frac{\pi x'_2}{2}\right) \times \cos\left(\frac{\pi x'_4}{2}\right) \\
& f_4(\mathbf{x}) = x'_5 + \sin\left(\frac{\pi x'_1}{2}\right) \times \cos\left(\frac{\pi x'_3}{2}\right) \\
& f_5(\mathbf{x}) = x'_5 + \cos\left(\frac{\pi x'_1}{2}\right)
\end{aligned} \tag{40}$$

where:

$$\mathbf{x} = (x_1, x_2, \dots, x_k, x_{k+1}, \dots, x_m) : m = 60, \quad k = 10, \quad l = 50, \quad x_{i=1..m} \in [0, 1]$$

$$\mathbf{x}' = (x'_1, x'_2, \dots, x'_5) :$$

$$\begin{aligned}
x'_1 &= \sum_{i=1}^{k/4} 4x_i / k, \quad x'_2 = \sum_{i=k/4+1}^{k/2} 4x_i / k, \\
x'_3 &= \sum_{i=k/2+1}^{3k/4} 4x_i / k, \quad x'_4 = \sum_{i=3k/4+1}^k 4x_i / k, \\
x'_5 &= \sum_{i=k+1}^m \text{s\_linear}((x_i / 2i), 0.35) / l
\end{aligned}$$

### 5.3.10 A DEGENERATIVE PROBLEM — AP-20

As illustrated in Figure A33 and Figure A34, the *AP-20* problem (Equation (41), an adaptation of *II* from the WFG suite) represents a degenerative two-dimensional objective-space, with the extent of the leading front diminishing until arrival at a single point. The consequence is that the import of the first  $k$  decision variables degrades the nearer the solution is to optimal space and so, in-turn, does the value of maintaining a diverse front. Thus, the optimiser must be able to adapt to shifting needs in diversification and an increasing bias in the utility of the final  $l$  parameters.

It is worth noting that the original degenerative function provided by Huband *et al.* (*WFG3* in [158]) is impressive, but is restricted to at least three-dimensions and is non-separable. Such inclusions make for a more complex problem, but it has the potential to confuse the issue at hand — that is, the effect of degrading fronts on optimiser performance. Poor results for an optimiser on *WFG3* may be attributable to the increase in dimensionality, the dependencies between decision variables or

degeneration — by muddying the waters with additional problem features, clarity in analysis is ultimately lost<sup>19</sup>. Thus, the *AP-20* function offers a more distilled focus.

$$\begin{aligned}
 &\text{Minimise } f_1(\mathbf{x}), f_2(\mathbf{x}) \\
 &f_1(\mathbf{x}) = x_2'' + \sin\left(\frac{\pi x_1''}{2}\right) \\
 &f_2(\mathbf{x}) = x_2'' + \cos\left(\frac{\pi x_1''}{2}\right) \\
 &\text{where:} \\
 &\mathbf{x} = (x_1, x_2, \dots, x_k, x_{k+1}, \dots, x_m) \\
 &\mathbf{x}' = (x'_{1..m}) : x'_1 = \sum_{i=1}^k x_i / k, \quad x'_2 = \sum_{i=k+1}^m \text{s\_linear}(x_i, 0.35) / l \\
 &\mathbf{x}'' = (x''_1, x''_2, x''_3) : x''_1 = 0.5 + x'_2 \times (x'_1 - 0.5), \quad x''_2 = x'_2 \\
 &m = 60, \quad k = 10, \quad l = 50, \quad x_{i=1..m} \in [0, 1]
 \end{aligned} \tag{41}$$

### 5.3.11A MULTI-FACETED PROBLEM — AP-21

Thus far, the *AP* suite has endeavoured to distil the core characteristics of multiobjective problems into a number of distinct, narrow-focus, functions. Such isolation of domain features allows the preceding functions to offer a unique insight into the effect of each characteristic on optimiser performance. The interlacing of distinct domain characteristics into a single function cannot offer such clarity of analysis, but it can indicate performance under particularly complex search- and objective-spaces, and suggest how specific properties interact. *AP-21* (Equation (42), an adapted form of *WFG9* that replaces the original non-separable reduction with the more complex *nonsep\_special* function) represents such a problem. Featuring multi-frontality (see Appendix A.23.6 and Figure A34), deception (see Appendix A.23.4 and Figure A33), dependencies between *all* decision variables (see Appendix A.23.7, Appendix A.23.5 and Figure A33) and a particularly biased objective-space<sup>20</sup> (see Figure A36), the problem is extremely challenging.

Though both complex and interesting, *AP-21* offers only a very small window into understanding the interaction of properties in multi-faceted domains. The problem,

<sup>19</sup> This is not intended as a criticism of the *WFG* suite, since it appears that its primary goal is to offer challenging problems akin to those that may be encountered in the real-world. This is a valid approach that simply runs counter to the central goals of this suite.

<sup>20</sup> The solution-space to objective-space mappings familiar to previous descriptions are excluded here as the complex inter-relationships between variables make it impossible to provide a succinct set.

of course, is that there are so many unique multiobjective characteristics, and so many ways in which those characteristics may be combined, that defining a suitably rich set of functions that offers a more complete insight is prohibitively complex (and the resulting suite is likely to be excessive in size). Since this thesis maintains that the most important step in understanding optimiser performance lies in focussed analysis on delineated domains, investigation on problems like *AP-21* is valuable, but ultimately of a secondary concern. Those users wishing to explore particular characteristic combinations (as may be appropriate when an algorithm is designed for a specific sub-set of domains) are encouraged to use and expand upon the *WFG* toolkit provided by Huband *et al.* [158] (an alternative to *AP-21* using this toolkit is provided in Appendix A.23) and to make the resultant problems available to the greater research community.

Minimise  $f_1(\mathbf{x}), f_2(\mathbf{x})$

$$f_1(\mathbf{x}) = x_2''' + 2 \sin\left(\frac{\pi x_1'''}{2}\right)$$

$$f_2(\mathbf{x}) = x_2''' + 4 \cos\left(\frac{\pi x_1'''}{2}\right)$$

where:

$$\mathbf{x} = (x_1, x_2, \dots, x_k, x_{k+1}, \dots, x_m) : m = 60, k = 10, l = 50, x_{i=1..m} \in [0, 2i]$$

$$\mathbf{x}' = (x'_1, x'_2, \dots, x'_m) :$$

$$x'_{i=1..m-1} = \text{b\_param}\left(x'_i / 2i, \left(\sum_{j=i+1}^{m-1} \frac{(x_j' / 2j)}{(m-i-1)}\right), \frac{0.98}{49.98}, 0.02, 50\right), \quad (42)$$

$$x'_m = x_m / 2m$$

$$\mathbf{x}'' = (x''_1, x''_2, \dots, x''_m) :$$

$$x''_{i=1..k} = \text{s\_decept}(x'_i, 0.35, 0.001, 0.05),$$

$$x''_{i=k+1..m} = \text{s\_multi}(x'_i, 30, 95, 0.35)$$

$$\mathbf{x}''' = (x'''_1, x'''_2) :$$

$$x'''_1 = \text{nonsep\_special}((x''_1, x''_2, \dots, x''_k), (x''_1, x''_2, \dots, x''_k))$$

$$x'''_2 = \text{nonsep\_special}((x''_{k+1}, x''_{k+2}, \dots, x''_m), (x''_1, x''_2, \dots, x''_{(m-k)}))$$

## 5.4 CONCLUSIONS

Existing real-valued multiobjective optimisation test suites have offered either complex multi-faceted problems that fail to provide a clear insight into performance on specific domain features or small collections of functions that lack scope or sufficiently detailed descriptions. The new *AP* test suite is an amalgamation of both suitable existing problems and new functions that are each designed to capture specific core multiobjective problem characteristics in a largely isolated manner. By offering a rich set of distilled, narrow-focus, problems (see Table 4 for a summary), the *AP* suite is intended as a resource for researchers looking to explore algorithmic performance — elucidating characteristics that most strongly affect the behaviour of an optimiser.

**Table 4 — Summarising the New *AP* Test Suite**

Problem	Key Domain Features Explored	
<i>AP-1</i>	Shape of Pareto optimal front in objective-space	Continuous convex.
<i>AP-2</i>		Continuous concave.
<i>AP-3</i>		Discontinuous convex.
<i>AP-4</i>	Multi-frontality	
<i>AP-5</i>	Bias	
<i>AP-6</i>	Noise	Gaussian noise.
<i>AP-7</i>		Evenly-distributed noise.
<i>AP-8</i>	Constraints	True Pareto optimal front: fully obscured. Feasible optimal front: discontinuous due to infeasibility holes. Objective-space beyond front: continuous.
<i>AP-9</i>		True Pareto optimal front: fully obscured. Feasible optimal front: continuous. Objective-space beyond front: features infeasibility holes.
<i>AP-10</i>		True Pareto optimal front: partially obscured. Feasible optimal front: discontinuous due to holes. Objective-space beyond front: features numerous fine-grained holes.
<i>AP-11</i>	Dynamism	Objective-space: fixed Pareto optimal front. Decision-space: variable Pareto optimal set and search-gradient.
<i>AP-12</i>		Objective-space: variable Pareto optimal front. Decision-space: fixed Pareto optimal set.
<i>AP-13</i>		Objective-space: variable Pareto optimal front and variable bias. Decision-space: variable Pareto optimal set.
<i>AP-14</i>	Non-Separability	Direct pair-wise mutually dependent relationships in part of the solution.
<i>AP-15</i>		Non-separability throughout the entire solution, including direct, indirect, mutual and non-mutual dependencies.
<i>AP-16</i>	Zero-Utility Gradient.	
<i>AP-17</i>	Deception.	
<i>AP-18</i>	Higher Dimensionality	Three-dimensional objective-space.
<i>AP-19</i>		Five-dimensional objective-space.
<i>AP-20</i>	Degeneration.	
<i>AP-21</i>	Multi-faceted problem.	

Though it is both extensive and well-described, the *AP* suite is not intended to be a static resource — it should grow and change as the community’s knowledge of multiobjective optimisation in general, and problem characteristics in particular, improve. As such, researchers are encouraged to develop further narrow-focus problems, with a view to emphasising new or differing characteristics to those already presented in the *AP* collection. Moreover, as this is the first step towards creating a comprehensive suite of characteristic-specific problems, it is likely that further refinements may be made that improve both the balance and function of the proposed problems — thus, analysis and critiquing of the suite is an important area of future work.

With a thorough understanding now of the mechanics of multiobjective optimisation and the domain features that are of significant interest (explored both from general and problem-centric views), the remainder of this work will narrow its focus to bi-objective specialisation. In particular, the unique properties that set bi-objective optimisation apart from generic multiobjective search will be explored, with the intention of harnessing such properties for impressive performance gains.

# Chapter

# 6

## Bi-Objective Optimisation & the Mak\_Tree

*“Nothing is so useless as a  
general maxim.”*

*Lord Macaulay — Poet*

*“To generalise is to be an  
idiot.”*

*William Blake — Poet*

*“If I have a thousand ideas  
and only one turns out to be  
good, I am satisfied..”*

*Alfred Nobel — Chemist,  
Engineer and Inventor of  
Dynamite*



## **6 BI-OBJECTIVE OPTIMISATION AND THE MAK TREE**

It should be obvious that single-objective problems represent a specialisation of the multiobjective class — that is, the notions of dominance, and Pareto relationships in general, take on a number of special characteristics under the constraints of a single objective. The differentiation is such that single objective optimisation and multiobjective optimisation exist as two distinct, well delineated, fields — each with their own unique, though often related, techniques, heuristics and methodologies. The implication is that while generic multiobjective techniques may be applicable to single objective optimisation, the host of specialist approaches available are typically better suited and better performing<sup>21</sup>.

Given the status granted to single-objective optimisation as a special subset of the generic multiobjective domain, and the benefits that such differentiation imparts, it is surprising that little research has been conducted into elucidating other special cases. The most striking omission is bi-objective domains, which have long been a centrepiece of multiobjective research, both in theoretical testing and practical applications. Indeed, Coello, Veldhuizen and Lamont's [51] definitive work on multiobjective optimisation up until the completion of 2001 cites that “the overwhelming majority” of published studies are focussed on the bi-objective domain and suggests that “most real-world [multiobjective problems] are effectively solved using only two or three” objectives. Since 2001, the trend has continued, with [12, 13, 20, 21, 30, 52-74] illustrating just a fraction of the contemporary studies that examine *exclusively* bi-objective domains.

Moreover recent studies [75-78] suggest that there is a lack of correlation between the performance of Pareto-based algorithms in domains featuring two or three objectives and those with higher dimensionality — the implication being that the requirements of an optimiser differ depending on the number of objectives under consideration. Cast under the shadow of generality, it is easy to view such findings negatively — it suggests that the pursuit of a single unifying algorithm that will function well for any number of objectives is a false one and that the hard focus on bi-objective domains was not the way to make strides towards such an ideal anyway.

---

<sup>21</sup> Though recent studies have shown that increasing the objective-space dimensionality of purely single-objective problems may yield benefits for particular problem classes – see the works of [176-180], for instance.

But that is to take a pessimists point of view; taken from a different perspective, the results are exciting — they suggest that there may exist unique characteristics in the lower-dimensional domains that can be exploited for performance gain. As such, rather than seeking out an ideal generalist algorithm, perhaps it is better to form more robust and powerful specialist techniques. Given the prevalence of bi-objective optimisation in real-world practical applications, the investigation and development of specialist strategies for this sub-class of multiobjective optimisation seems to be an ideal place to start.

While a number of unique bi-objective properties are held as common knowledge in the field (as will be discussed in Section 6.1.2), very few techniques capitalise on this knowledge and a number of more subtle, though no-less-powerful, properties have not been identified at all. As such, the following section offers a complete set of the unique characteristics of bi-objective domains and then builds upon these to create a number of powerful specialist techniques.

## **6.1 PROPERTIES OF BI-OBJECTIVE PROBLEMS**

For the purposes of this work, a bi-objective task is considered to be any conflicting multiobjective problem (as described in Section 2.2) where  $\beta=2$ . This simple constraint alone is enough to generate a host of interesting, and unique, properties that differentiate bi-objective optimisation from the more generic multiobjective case. Note that all properties assume, without loss of generality, that both objectives are to be minimised.

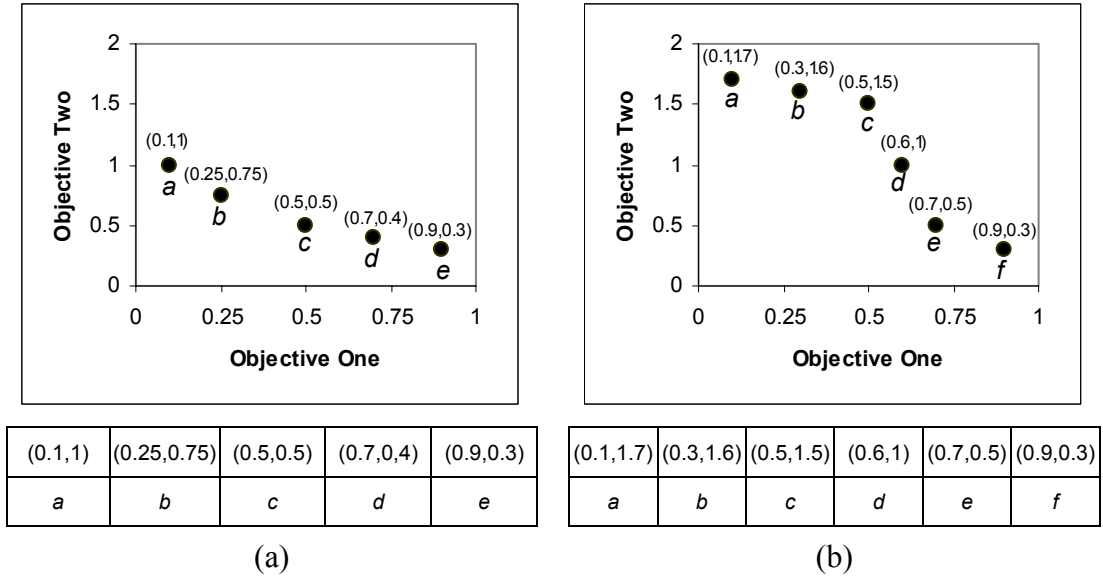
### **6.1.1 DEFINING AN ORDERED NON-DOMINATED BI-OBJECTIVE LIST**

Before examining each specific property, it is useful to define an ordered non-dominated bi-objective list. As will be demonstrated in the following sections, this simple construct is the key to developing many of the unique characteristics inherent in bi-objective optimisation.

With no loss in generality, a list  $R$  of solutions is ordered in the context of this work if and only if:

$$\forall \mathbf{a}, \mathbf{b} \in R, f_1(\mathbf{a}) \leq f_1(\mathbf{b}) \text{ iff } (\text{index}(\mathbf{a}) \leq \text{index}(\mathbf{b})) \quad (43)$$

where  $\text{index}(\mathbf{x})$  returns the position of solution  $\mathbf{x}$  in the list.



**Figure 23 - Example Ordered Non-dominated Lists**

Both lists (a) and (b) are arbitrarily ordered by solution performance on the first objective.

That is, the order of the list is entirely governed by performance on the first objective, with increasing indices indicating decreasing performance. An ordered non-dominated bi-objective list  $L$  is achieved by ensuring  $R$  is strictly non-dominated and includes only solutions to bi-objective problems (see Figure 23). For convenience, properties may assume that a *single* dominating or dominated solution may be temporarily inserted into  $L$  to enable sensible discussions about its list predecessors and successors, so long as  $L$  is composed only of mutually non-dominating proposals prior to such an insertion.

### 6.1.2 PROPERTY ONE: ORDERING

*If solutions in a non-dominated bi-objective list  $L$  are ordered according to their ascending value (decreasing performance) on the first objective, then it is always the case that the same order represents descending value (increasing performance) on the second objective<sup>22</sup>.*

If the in-order successor of solution  $a$  in the ordered non-dominated bi-objective list is denoted as  $\bar{a}$  and the in-order predecessor is  $\bar{a}$  then the ordering property states that:

<sup>22</sup> Without loss of generality, this type of ordering is assumed for all ordered bi-objective lists discussed herein.

$$\left( (f_1(\mathbf{a}) < f_1(\vec{\mathbf{a}})) \wedge (f_2(\mathbf{a}) > f_2(\vec{\mathbf{a}})) \right) \vee (\mathbf{a} = \vec{\mathbf{a}}) \forall \mathbf{a}, \vec{\mathbf{a}} \in L \quad (44)$$

This can be verified by considering any two solutions in a non-dominated bi-objective list: if solution  $\mathbf{a}$  outperforms solution  $\mathbf{b}$  on objective one, it follows that for  $\mathbf{b}$  to be non-dominated it must outperform  $\mathbf{a}$  on objective two:

$$(f_1(\mathbf{a}) < f_1(\mathbf{b})) \wedge (\mathbf{a} \sim \mathbf{b}) \Rightarrow (f_2(\mathbf{b}) < f_2(\mathbf{a})) \quad (45)$$

Interestingly, this property means that the extent of any bi-objective non-dominated front can be found by simply retrieving the front and end of the ordered list (consider the examples illustrated in Figure 23).

Such ordering also leads to the somewhat less intuitive conclusion that the nearest neighbour of a solution in objective-space will always be the successor or predecessor of that solution in the ordered list. To verify this notion, recall that the ordering property ensures that for any ordered non-dominated list, the following must hold:

$$\left( \begin{array}{c} \left( \dots < f_1(\vec{\vec{\mathbf{a}}}) < f_1(\vec{\mathbf{a}}) < f_1(\mathbf{a}) < f_1(\vec{\mathbf{a}}) < f_1(\vec{\vec{\mathbf{a}}}) < \dots \right) \\ \wedge \\ \left( \dots > f_2(\vec{\vec{\mathbf{a}}}) > f_2(\vec{\mathbf{a}}) > f_2(\mathbf{a}) > f_2(\vec{\mathbf{a}}) > f_2(\vec{\vec{\mathbf{a}}}) > \dots \right) \end{array} \right) \quad (46)$$

and it should be obvious that  $\mathbf{a}$  is numerically closest to  $\vec{\mathbf{a}}$  or  $\vec{\vec{\mathbf{a}}}$  for both objective one and objective two. Thus, for any sensible objective-space distance metric  $\delta$ , it is always the case that:

$$\left( \begin{array}{c} \left( \delta(\mathbf{a}, \vec{\mathbf{a}}) < \delta(\mathbf{a}, \vec{\vec{\mathbf{a}}}) < \delta(\mathbf{a}, \vec{\vec{\vec{\mathbf{a}}}}) < \dots \right) \\ \wedge \\ \left( \delta(\mathbf{a}, \vec{\mathbf{a}}) < \delta(\mathbf{a}, \vec{\vec{\mathbf{a}}}) < \delta(\mathbf{a}, \vec{\vec{\vec{\mathbf{a}}}}) < \dots \right) \end{array} \right) \quad (47)$$

and the list-predecessor or list-successor must contain the nearest objective-space neighbour for any given solution. By simple extension, it must also hold that the  $\kappa$  nearest-neighbours of any solution must come from the  $\kappa$  list-successors and the  $\kappa$  list-predecessors of that solution. Similarly, the furthest objective-space neighbour of a given solution must be located at the front or end of the list, and the  $\kappa$  furthest

neighbours must come from the  $\kappa$  list-predecessors of the end and the  $\kappa$  list-successors of the front.

Thus by simply ordering the non-dominated front on an arbitrarily chosen objective, crowding and extent-based information about members of the non-dominated front becomes instantly more accessible in the special two-dimensional case. Consider a simple data structure that contains the constituent solutions of the prevailing non-dominated front. In the generic multiobjective case, where the ordering property does not hold, locating a single nearest neighbour of any solution will take at least  $O(P_{estimate})$  time — since all other stored solutions must be checked against. Similarly, to determine the extent of the front an  $O(P_{estimate})$  search is required. Assuming an appropriately ordered list, locating the nearest neighbour of a solution in objective-space and finding the extent of a non-dominated front are constant time operations when the special properties of the bi-objective subset are capitalised upon — a marked improvement over the generic approach.

As further evidence of the advantages gained via leverage of this property, consider an algorithm that requires nearest neighbour scores for every stored solution — a reasonable technique for approximating the distribution of the current front. Sorting the population will require, at worst,  $O(P_{estimate} \log P_{estimate})$  operations given an appropriately chosen data structure (such as a self-balancing binary search tree). Using the special ordering property, subsequent calculation of all nearest neighbours will require  $O(P_{estimate})$  further comparisons, thus incurring a total complexity (dominated by the sorting procedure) of  $O(P_{estimate} \log P_{estimate})$ . In contrast, applying techniques appropriate to the generic multiobjective case requires a search across every stored solution — costing a far more expensive  $O((P_{estimate})^2)$ . As will be seen later (see Section 6.3) such time complexity advantages are particularly prevalent when the properties of the non-dominated bi-objective list are integrated into an appropriately powerful data structure.

### 6.1.3 PROPERTY TWO: OBJECTIVE-RESULT VARIANCE

*Any two solutions sharing an objective score in a non-dominated list will also share an identical point in objective-space. No horizontal or vertical line through a two-dimensional, non-dominated, objective-space will ever intersect more than one point.*

That is, no two solutions may share exactly one objective score in  $L$  — they may be completely equal, with respect to their position in objective-space, or they may be completely incomparable. The notion is easily verified by considering the converse case, where a solution ( $\mathbf{a}$ ) shares exactly one objective score with another solution ( $\mathbf{b}$ ):

$$\begin{aligned} ((f_i(\mathbf{a}) = f_i(\mathbf{b})) \wedge (f_j(\mathbf{a}) \neq f_j(\mathbf{b}))) &\Rightarrow (f_j(\mathbf{a}) < f_j(\mathbf{b})) \vee (f_j(\mathbf{a}) > f_j(\mathbf{b})) \\ &\Rightarrow (\mathbf{a} \succeq \mathbf{b}) \vee (\mathbf{a} \preceq \mathbf{b}) \end{aligned} \quad (48)$$

with  $i$  and  $j$  representing distinct objectives. Since Equation (48) implies that one of the solutions must dominate the other, and such an arrangement is impossible in a non-dominated list, the case must be disqualified.

While conceptually this property seems decidedly uninteresting, it takes on great significance when considering the storage of solutions in tree-like structures. For example, consider the simple, yet powerful, binary search tree: to maintain order, a binary search tree requires completely unique addresses (*identifiers* or *labels*) across all nodes. Assuming that each node in the tree houses solutions that represent a single point in objective-space, the performance on a given objective provides unique labels for all nodes. In the generic multiobjective case this property does not hold (since two solutions may share an objective score and still remain incomparable — for instance (3,3,1) and (3,1,3)), thus rendering simple binary search trees unsuitable. Therefore, the objective-result variance property permits use of a simple, accessible and efficient data structure that is otherwise unavailable in the generic multiobjective case. The importance of this result will be clearly illustrated in Section 6.3.

#### 6.1.4 PROPERTY THREE: DOMINATED SETS

*Consider a non-dominated list of solutions ordered on the performance of an arbitrarily chosen objective: if solutions at index  $i$  and  $j$  (where  $i \leq j$ ) are both dominated by some incoming proposal, then all solutions with an index between  $i$  and  $j$  (inclusive) are also dominated (referred to here as a dominated set). If  $i$  is the lowest dominated index and  $j$  is the highest, then the dominated set represents every dominated solution in the list and is the complete dominated set.*

That is, for an incoming dominating solution  $\mathbf{a}$  the following holds:

$$(\mathbf{a} \preceq L_i) \wedge (\mathbf{a} \preceq L_j) \Rightarrow (\mathbf{a} \preceq L_\varphi) \forall \varphi \in [i : j] \quad (49)$$

where both  $i$  and  $j$  are indexes into the ordered non-dominated list  $L$  ( $i \leq j$ ).

The complete dominated set is bounded by terminals at indices  $i$  and  $j$  such that:

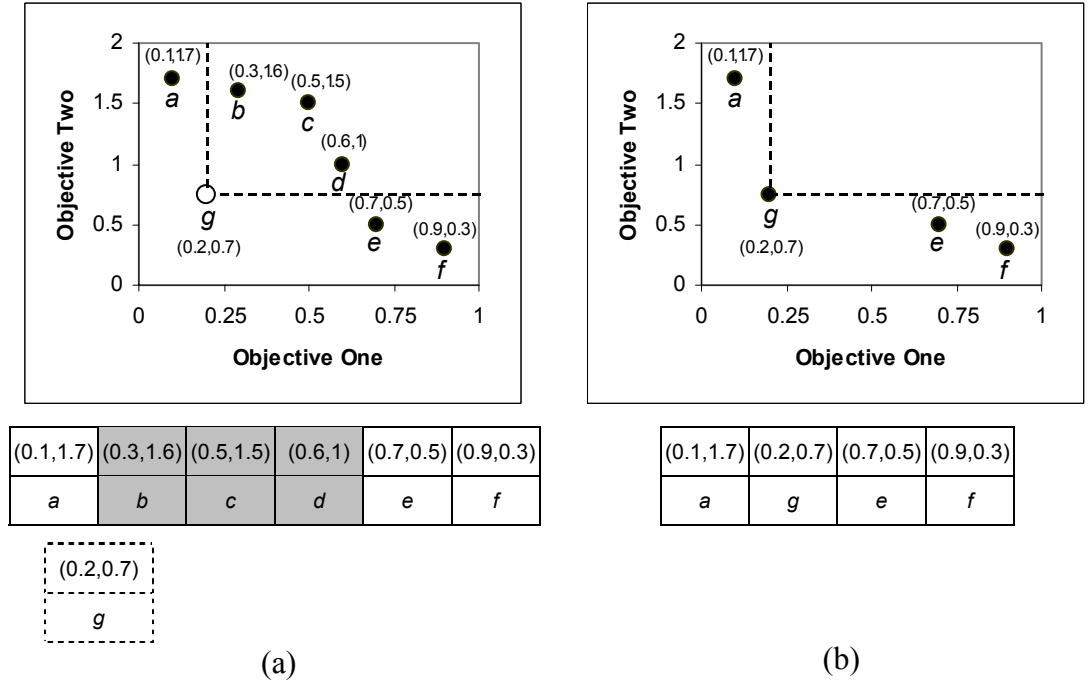
$$(a \preceq L_i) \wedge (a \preceq L_j) \wedge (a \sim L_{i-1}) \wedge (a \sim L_{j+1}) \quad (50)$$

As an example, consider Figure 24. If it is known that the incoming solution  $g$  dominates  $b$  and  $d$ , then it is always the case that  $g$  will also dominate any solution lying between  $b$  and  $d$  in the ordered list (in this case,  $c$ ). Since  $b$  and  $d$  also represent the lowest and highest indexed solutions that are dominated by  $g$ , the resulting dominated set is complete and solutions  $b$  and  $d$  are referred to as *dominated set terminals* (or the  $i\_terminal$  and  $j\_terminal$  respectively).

To verify this property, first let  $d$  be a solution that dominates list members at indexes  $i$  and  $j$  respectively such that:

$$(d \preceq L_i) \wedge (d \preceq L_j) \quad (51)$$

and consider a simplified form of the property where the ordered list is guaranteed to



**Figure 24 - Illustrating Dominated Sets**

(a) The initial non-dominated ordered list. The shaded region is dominated by the incoming solution  $g$ . Solution  $b$  and solution  $d$  are the terminals of the complete dominated set. (b) The resulting non-dominated ordered list.

contain solutions producing unique objective scores (that is, no two objective results may be the same). By Property One (see Section 6.1.2), it is always the case that:

$$(f_1(\mathbf{d}) \leq f_1(L_i) < f_1(L_j)) \wedge (f_2(\mathbf{d}) \leq f_2(L_j) < f_2(L_i)) \quad (52)$$

and it will also be true that all solutions with index  $w > i$  will be worse on objective one than  $\mathbf{d}$ :

$$(f_1(L_w) > f_1(L_i) \geq f_1(\mathbf{d})) \forall w \in \{i+1, \dots, |L|\} \quad (53)$$

while solutions with index  $w < j$  must be worse on objective two:

$$(f_2(L_w) > f_2(L_j) \geq f_2(\mathbf{d})) \forall w \in \{1, \dots, j-1\} \quad (54)$$

Thus, solutions with indexes in the range  $(i, j)$  must also be dominated by  $\mathbf{d}$ :

$$((f_1(L_w) > f_1(L_i) \geq f_1(\mathbf{d})) \wedge (f_2(L_w) > f_2(L_j) \geq f_2(\mathbf{d}))) \forall w \in \{i+1, \dots, j-1\} \quad (55)$$

However, this proof assumes that no two solutions will ever share objective scores. An extension to the more general case, where such duplication is permitted, is rudimentary. Since the non-dominated list is ordered, all solutions producing duplicate objective scores will occupy a continuous block of list-nodes (see Figure 25). Each block will have a single unique value and, as a consequence, the proofs provided in Equations (51)–(55) are analogous. Given that this proof guarantees that all solutions between the first member of the  $i^{\text{th}}$  block and last member of the  $j^{\text{th}}$  block will be dominated, the dominated set property must hold, even under the storage of duplicate objective-result values.

Block One						Block Two					
(0,1)	(0.2,0.9)	(0.4,0.7)	(0.4,0.7)	(0.4,0.7)	(0.4,0.7)	(0.5,0.6)	(0.5,0.6)	(0.5,0.6)	(0.8,0.3)	(0.9,0.2)	(1,0)
a	b	c	d	e	f	g	h	i	j	k	l

**Figure 25 — An Example of Dominated Sets With Duplicates**

The shaded region represents the complete dominated set when a solution with objective scores of (0.3,0.25) is to be inserted into the non-dominated list. Each of the identified blocks contains solutions that generate duplicate objective scores. Obviously, if any member of a block is dominated, then all members of that block must also be dominated.



The benefits inherent in the dominated set property should be immediately apparent. If it is possible to efficiently locate the  $i$  and  $j$  terminal nodes for the complete dominated set, a potentially large number of redundant dominance comparisons can be avoided. An excellent example of this, and the performance advantages induced by the application of such properties, is detailed in Section 6.3.

### 6.1.5 PROPERTY FOUR: NON-DOMINANCE

*Consider a non-dominated list ordered by performance on some arbitrarily chosen objective: any incoming solution  $\mathbf{a}$  will be non-dominated with respect to the entire ordered list if it is non-dominated by its list-predecessor. If  $\mathbf{a}$  has no predecessor, then the solution is the head of the ordered list and represents an extreme point in objective-space — it too will be non-dominated with respect to the members of the list.*

$$(\bar{\mathbf{a}} \not\leq \mathbf{a}) \vee (\bar{\mathbf{a}} = \text{null}) \Rightarrow (\mathbf{b} \not\leq \mathbf{a}) \forall \mathbf{b} \in L \quad (56)$$

Verification of the property is straight-forward. By the ordering property, it is always the case that  $\mathbf{a}$  cannot be dominated by any list-successor since  $\mathbf{a}$  will always outperform a successor on the first objective (assuming the ordering specified in Section 6.1.2 holds) or otherwise share an identical position in objective-space. If  $\bar{\mathbf{a}}$  does not dominate  $\mathbf{a}$  then, by Property One, the following is true:

$$(f_2(\mathbf{a}) < f_2(\bar{\mathbf{a}}) \leq f_2(\bar{\bar{\mathbf{a}}}) \leq \dots) \vee (\mathbf{a} = \mathbf{b}) \quad (57)$$

and  $\mathbf{a}$  cannot be dominated by any preceding element in the list. Thus,  $\mathbf{a}$  must be non-dominated with respect to the entire list.

Again, the implications of this property are most important in efficiency concerns. Assuming that it is possible to rapidly identify the list-predecessor of any incoming proposal, the non-dominance of a given solution (and thus membership of the prevailing front) can be determined with only a single dominance comparison. In contrast, the naïve list stores most commonly used in generic multiobjective front storage (see Section 6.2.1) will require a comparison with every stored solution to arrive at the same conclusion.

### 6.1.6 PROPERTY FIVE: DOMINANCE

If an incoming solution  $\mathbf{a}$  dominates any member of an ordered non-dominated list of solutions, then it must dominate its first (non-equivalent) successor  $\tilde{\mathbf{a}}$  in that list.

$$\text{if } \mathbf{b} \in L : \mathbf{a} \preceq \mathbf{b} \text{ then } \mathbf{a} \preceq \tilde{\mathbf{a}} \quad (58)$$

This is a simple extension of Property Four that features a suitably simple proof. Consider some solution  $\mathbf{b}$  that is dominated by  $\mathbf{a}$ . It is obvious that the following holds from first principles:

$$((f_1(\mathbf{a}) \leq f_1(\mathbf{b})) \wedge (f_2(\mathbf{a}) < f_2(\mathbf{b}))) \vee ((f_1(\mathbf{a}) < f_1(\mathbf{b})) \wedge (f_2(\mathbf{a}) \leq f_2(\mathbf{b}))) \quad (59)$$

and, by the ordering property, it is also true that:

$$(f_1(\mathbf{a}) < f_1(\tilde{\mathbf{a}})) \wedge (f_2(\mathbf{b}) \leq f_2(\tilde{\mathbf{a}})) \quad (60)$$

Consequently,  $\tilde{\mathbf{a}}$  must be dominated by the incoming solution.

With this property in place, it is possible to determine whether a solution is strictly non-dominating by examining only the in-order successor of that solution. Moreover, the dominance property indicates the location of the  $i$ -terminal node of the dominated set — it will always be the immediate unique successor of the incoming dominating solution.

### 6.1.7 SUMMARISING THE KEY BI-OBJECTIVE PROPERTIES

Table 5 outlines the key properties that distinguish the special bi-objective subset from the generic multiobjective domain. As evidenced in the table (and in preceding sections), the advantages that bi-objective specialisation imparts are numerous and have a profound impact on the efficiency of storing and updating non-dominated sets.

## 6.2 ASSESSING LIMITATIONS OF CONTEMPORARY GENERIC MULTI-OBJECTIVE OPTIMISATION

Before developing data structures and algorithms that can capitalise on the unique properties of bi-objective sets, it is useful to first assess some of the limitations inherent in contemporary generic multiobjective optimisation. If bi-objective

Table 5 — Special Properties of Ordered Non-dominated Bi-Objective Lists

	Name	Property Characteristics	Advantages
<b>Property One</b>	Ordering	Increasing objective-one scores lead to decreasing objective-two scores (and vice-versa).	Front-extent calculated from front and end of list. Nearest objective-space neighbour is always the in-order predecessor or successor of a solution.
<b>Property Two</b>	Variance	No two solutions may share precisely one objective result.	Solutions may be completely ordered by performance on a single objective — critical in data storage.
<b>Property Three</b>	Dominated Sets	If solutions at index $i$ and $j$ are dominated by an incoming proposal, all solutions with an index between $i$ and $j$ are also dominated (and form the dominated set).	If $i$ and $j$ are known, no further comparisons are required to determine all dominated points in the list.
<b>Property Four</b>	Non-Dominance	If an incoming solution is not dominated by its in-order predecessor, it will not be dominated by any solution in the list.	To verify that a solution is non-dominated, only the predecessor must be compared against.
<b>Property Five</b>	Dominance	If an incoming solution dominates any member of the list, it must dominate at least its list-successor.	To verify that a solution is non-dominating, only the successor must be compared against. The first unique successor of the dominating solution is the $i$ -terminal node.

specialisations can free algorithms of such limitations then the argument that bi-objective domains represent a usefully differentiable subset of the generic multiobjective case can only be strengthened.

### 6.2.1 ELITE ARCHIVING

Contemporary multiobjective optimisation research holds that the use of an elite archive of impressive solutions can fundamentally increase the performance of a given algorithm [92-94]. It is for this reason that the second generation of evolutionary optimisation techniques — that is, those that have followed the pioneering works of Schaffer [146], Srinivas and Deb [2], Fonesca and Flemming [181], Horn, Nafpliotis and Goldberg [142], and others — are typically reliant on an active store of apparently good solutions. Most notable amongst this burgeoning array of methodologies are the Pareto Archived Evolution Strategy (PAES [79]), the Strength Pareto Evolutionary Algorithm (SPEA [80]) and its sequel (SPEA2 [81]), the elitist Non-dominated Sorting Genetic Algorithm (NSGA-II [82]) and the Pareto Envelope Selection Algorithm (PESA [83]): all of which are reliant on archive-based

elitism to drive solutions towards ever-better approximations of the Pareto optimal front.

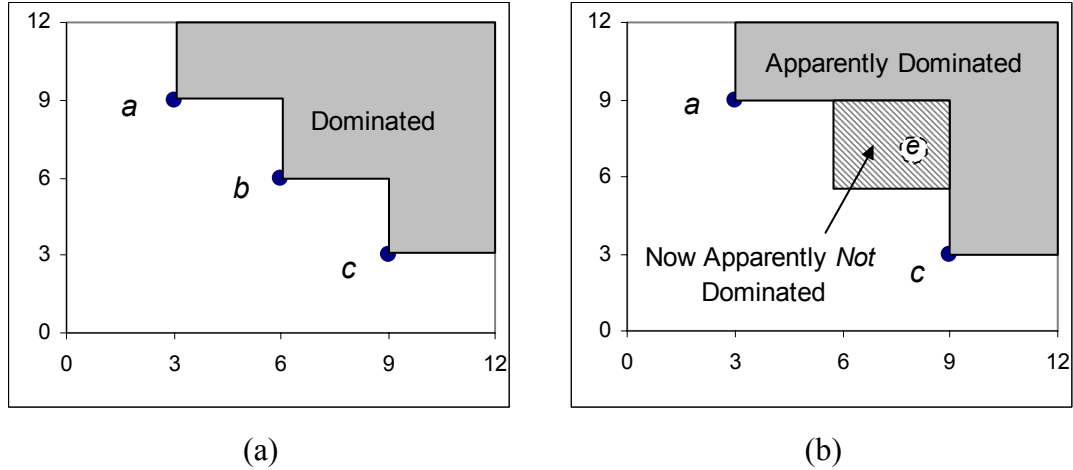
Since these approaches implicitly (and often explicitly, see [79]) use a naïve list representation for members of the archive [95], the cost of searching and maintaining large or unbounded stores is prohibitive and must therefore be avoided. Indeed, the worst case complexity while using a list data structure is  $O(\beta n)$  per archival insertion, where  $\beta$  is the number of objectives and  $n$  is the size of the archive. Given such inefficiency, efforts to reduce the size of  $n$  are hardly surprising.

Unfortunately, the artificial truncation of elite stores can lead to a variety of problems that result in performance degradation. Chief amongst these issues is the potential for fronts to oscillate or recede due to the removal of members from the archive. As will be seen in the following sections, such frontal degradation can lead to inaccurate crowding estimation and inhibit the use of dynamic stopping conditions. Additionally, truncation operations may limit the applicability of contemporary multiobjective optimisation techniques in particularly complex objective-spaces, where the loss of an expensive region due to set limitation may severely inhibit the efficiency and efficacy of the search.

#### **6.2.1.1 *FRONTAL DEGRADATION***

Given that a local optimal front will be composed of solutions that are strictly non-dominated by any other previously generated proposal, it is reasonable to expect all members of an elite archive to meet the same requirements (even if they form only a subset of the true front). However, the truncation operation means that the quality of the archive may degrade over time — invalidating the optimality requirement by allowing weak solutions into the archive.

The potential for decreasing archive quality is best exemplified via a diagram. Consider the extremely simple three-member elite archive illustrated in Figure 26a: if solution  $b$  is removed due to a truncation operation, the map of dominated space becomes inaccurate — the highlighted region in Figure 26b should be included, but is not. The effect is that a poor solution which was dominated by  $b$ , but is otherwise incomparable with the remaining members of the archive (such as the incoming  $e$  solution), will be incorrectly accepted as an elite member.



**Figure 26 — Frontal Degradation in Objective-space**

$x$ -axis is objective one;  $y$ -axis is objective two. The loss of solution  $b$  from the archive means that the highlighted region is incorrectly labelled as non-dominated in the resultant truncated archive. As a result, the weak  $e$  solution would be accepted as an elite member in (b).

The effect of degrading fronts can be extremely detrimental to the performance of any given algorithm. In the extreme case, frequent truncation of archives may lead to a retreating front — where the archive becomes a progressively worse approximation of the Pareto optimal set. The more likely case however is frontal oscillation — where good solutions are truncated from the archive and then rediscovered as part of the algorithm's search procedure. Obviously, the time spent rediscovering good ideas is better spent investigating less populated areas of the front and is costly to both the efficiency and efficacy of the search process.

It is important to note that the potential for frontal degradation is more than a theoretical concern that exists only in pathological cases. Indeed, Fieldsend *et al.* [97] demonstrate the capacity for oscillation in a simple ES(1+1) multiobjective optimiser (with an elite archive size of twenty) on the original *ZDT1* function, while Laumanns *et al.* [98] briefly address the tendency for NSGA-II [175] and SPEA [137] to degrade on a basic knapsack problem. While these empirical investigations into frontal degradation are significant, they are limited in scope and, in the case of Laumanns *et al.* [98], largely superficial. It could be argued that performance on a singular test function says little about the behaviour of truncated archives in general; that the archive of Fieldsend *et al.* [97] is unrealistically small and based on an

overly basic algorithm; and that the conclusions of Laumanns *et al.* [98] are based on assumptions and conjecture.

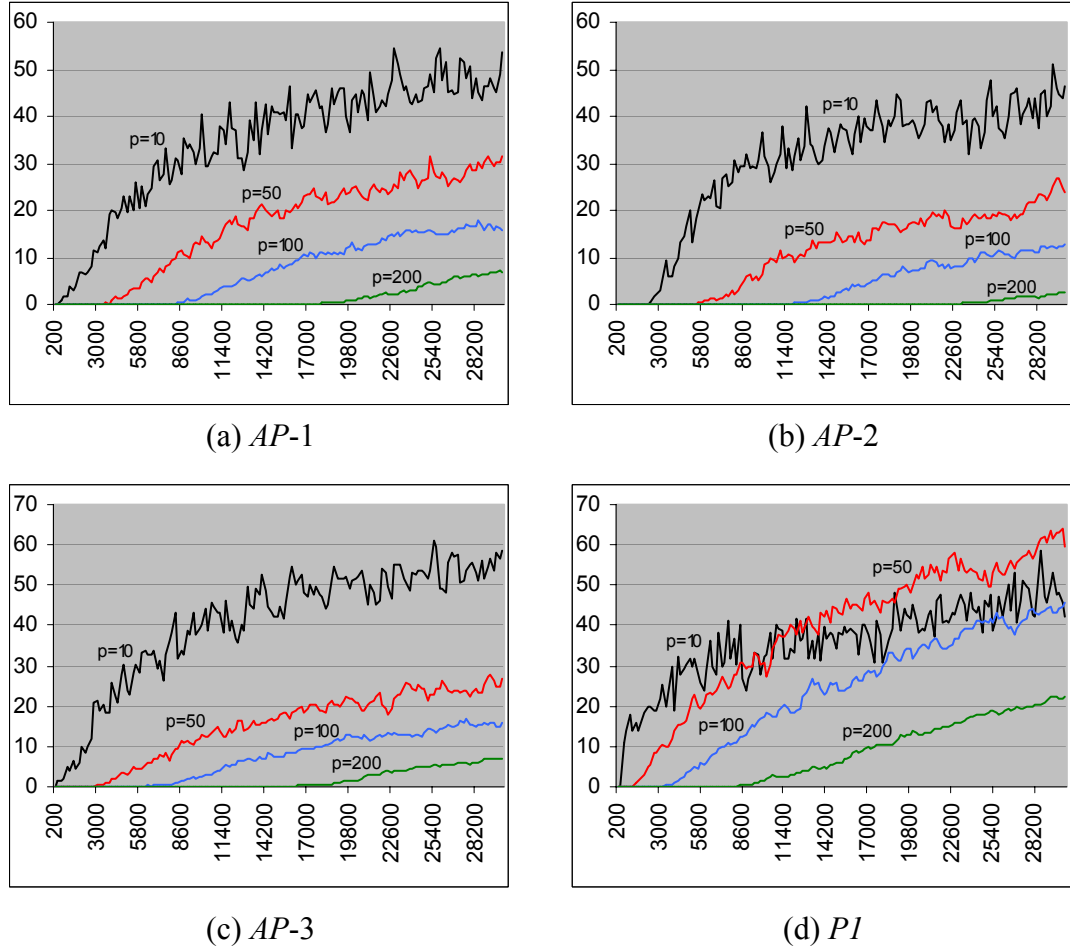
This section presents new results extracted from runs using the contemporary and powerful NSGA-II algorithm with differing archival thresholds across a diverse range of problems (including *AP-1*, *AP-2*, *AP-3* and the multi-faceted *P1* problem defined in Appendix A.22). The specific settings for the NSGA-II algorithm are provided in Appendix B.1.1 — and are consistent with those commonly seen in the literature. Results indicate the average percentage of dominated solutions that have been incorrectly labelled as non-dominated (and are therefore erroneously included as part of the leading front in the parent population) across twenty distinct runs.

As evidenced in Table 6 and Figure 27, the proportion of archival members that have been incorrectly labelled is large — even at relatively high population thresholds (where a smaller number of archival members are truncated per iteration — as illustrated in Figure 28). These findings are particularly noteworthy given the degradation visible in runs with thresholds of 50 and 100, since these are common levels used throughout the literature in performance analyses of NSGA-II (see, for instance, [21, 74, 81, 82, 92, 182-185]). Consider the performance of the 50 member NSGA-II system: looking at the final 20,000 evaluations of the *AP-1* test, and

**Table 6 — Incorrectly Labelled Non-dominated Solutions**

Results indicate the average percentage of dominated (weak) solutions that are incorrectly labelled as non-dominated in the truncated elite set. Averages are derived from twenty distinct NSGA-II runs per problem per population level.

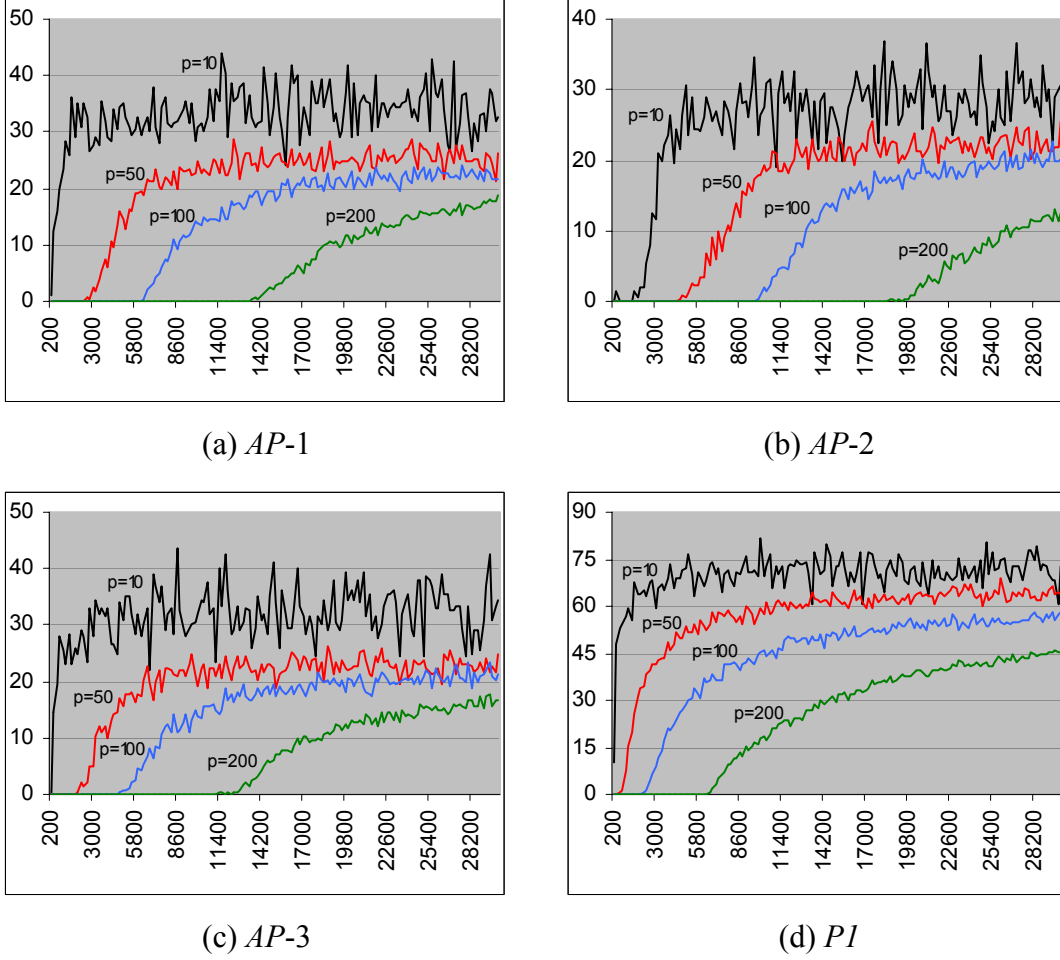
	Evaluations	<i>AP-1</i>	<i>AP-2</i>	<i>AP-3</i>	<i>P1</i>
<b>Population Size = 10</b>	0k — 10k	18.84	15.04	23.38	25.54
	10k — 20k	38.66	35.45	46.07	37.45
	20k — 30k	46.37	40.55	52.76	44.84
<b>Population Size = 50</b>	0k — 10k	4.15	1.43	4.35	17.04
	10k — 20k	19.48	13.42	17.12	42.90
	20k — 30k	26.85	19.75	23.19	55.68
<b>Population Size = 100</b>	0k — 10k	0.16	0.00	0.43	5.76
	10k — 20k	7.46	3.03	7.86	25.56
	20k — 30k	15.02	10.10	13.95	39.80
<b>Population Size = 200</b>	0k — 10k	0.00	0.00	0.00	0.19
	10k — 20k	0.12	0.00	0.31	7.08
	20k — 30k	4.05	0.91	4.75	17.87



**Figure 27 — The Percentage of Weak Solutions in NSGA-II Archives that are Incorrectly Labelled as Non-dominated**

For all graphs, the *x-axis* represents the number of evaluations that the NSGA-II algorithm has performed and the *y-axis* specifies the percentage of weak solutions in the supposedly non-dominated leading front stored in the NSGA-II archive. *p* represents the archival size threshold used.

assuming an approximately even distribution of points along the apparently non-dominated front, approximately 20% of all solutions produced are derived from some weak, dominated, proposal. Thus, given the prevalence of inferior solutions in the elite sets studied, there exists a very real danger that the efficiency of the search process will be fundamentally affected, particularly with respect to end-of-run convergence — where poor elite set fidelity is typically at its highest. If the central notion of multiobjective evolutionary optimisation techniques is to build on ever better approximations of the Pareto optimal front then it stands to reason that the type of frontal degradation seen in these tests renders truncated elite approaches at a severe disadvantage.



**Figure 28 — Number of Solutions Truncated from the Elite ( $\mathcal{F}$ ) Portion of the NSGA-II Archive**

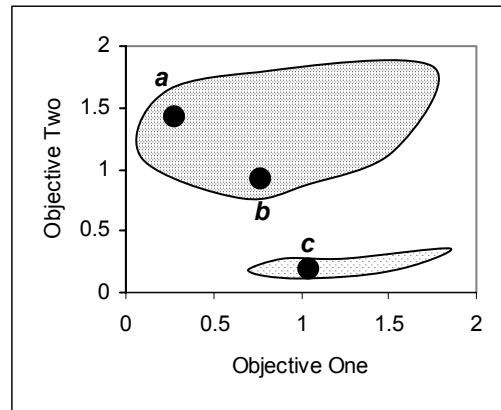
For all graphs, the  $x$ -axis represents the number of evaluations that the NSGA-II algorithm has performed and the  $y$ -axis specifies the number of solutions truncated from the supposedly elite leading front (given as a percentage of the corresponding archive size threshold  $p$ ).

### 6.2.1.2 LOSS OF EXPENSIVE REGIONS

Under the presence of discontinuous fronts, isolated regions, constraints, deception, and bias, certain solutions can reasonably be thought of as being more valuable to the search process than others. Consider Figure 29: the loss of the highly valuable solution  $c$ , which resides isolated in a disconnected and sparsely populated portion of objective-space, would severely inhibit the capacity of the algorithm to search the region in which it resides. Indeed, the effect is even more pronounced than oscillation, as simply rediscovering the lost solution may be extremely difficult.

The principle here is an important one — the more complex the objective-space becomes, the more important the size of the archive. Specifically, archival size acts





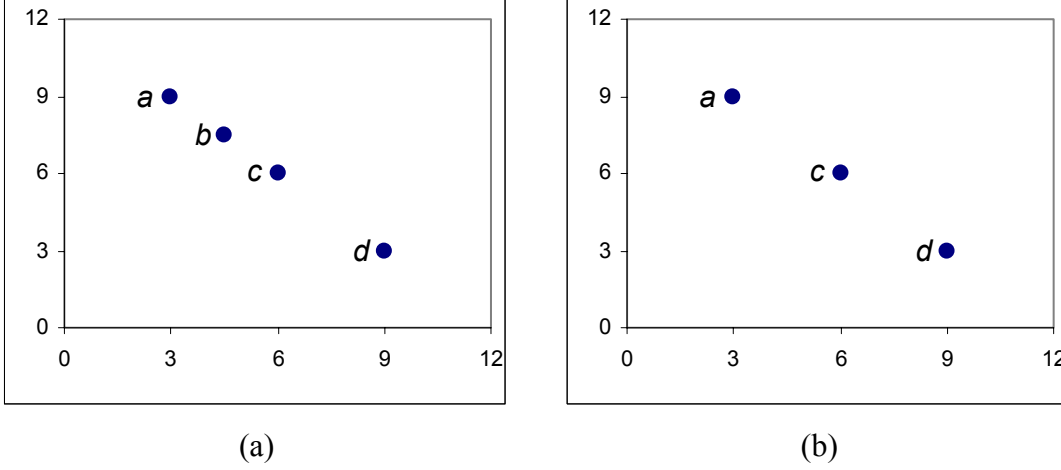
**Figure 29 - Losing a Valuable Region**

Darker shading represents a higher density of points in objective-space. If solution *c* is truncated from the archive, rediscovering the expensive (isolated and sparsely populated) region of space in which it resides may be difficult.

as an artificially enforced threshold on the number of distinct regions that can be explored simultaneously by a search algorithm. Should the number of distinct regions in objective-space surpass the size of the archive, the algorithm will either be confined to a subset of the objective-space or, more likely, suffer significant performance degradation as the archive consistently cycles between available zones — repetitively discarding solutions that may have been invaluable to a balanced search.

### 6.2.1.3 INACCURATE CROWDING ESTIMATION

Given that one of the explicit aims in developing a good approximation of the Pareto optimal front is achieving an evenly distributed set of solutions in objective-space, crowding estimation is very important in directing the search towards areas of low result-density. Unfortunately, truncation of archives can form misleading density approximations, as crowding measures will be based on an incomplete set of the true local optimal front. In Figure 30a it is obvious that solutions *a*, *b* and *c* occupy the most densely explored region of objective-space, but the truncation of point *b* from the archive leads to a uniform spacing of points that de-emphasises the isolation of *d* (as illustrated in Figure 30b). The danger that exists here is that solutions may be added and subsequently truncated from the crowded region — worsening the true distribution of points with little effect on the apparent archive-based crowding of the same area.



**Figure 30 — Misleading Objective-Space Crowding Information**

*x*-axis is objective one; *y*-axis is objective two. (a) The unbounded archive is concentrated about solution *b*. (b) The truncation of solution *b* results in a misleading even distribution of points in objective-space.

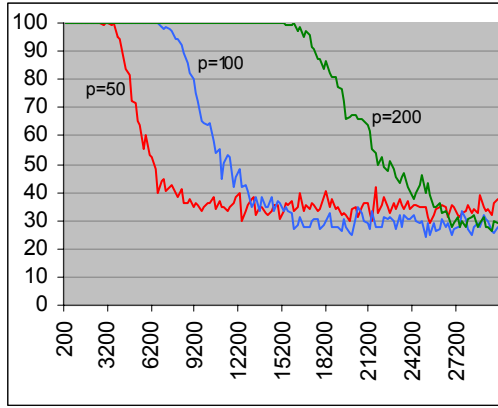
Empirical evidence of the propensity for localised crowding approaches to incorrectly estimate the true distribution of solutions in objective-space can be illustrated by examining leading ( $\mathcal{F}_1$ ) fronts produced during standard NSGA-II runs (see Appendix B.1.1 for parameter settings). With each front it is possible to assess which solutions are deemed to be in the least-crowded portions of space with respect to both the locally stored (truncated) solutions and the complete non-dominated set. If there is a poor correlation between the locally and globally derived estimations (assuming that both approaches use an otherwise identical crowding metric), then algorithms capitalising on a truncated solution set may not be driving the search adequately towards areas of low frontal exploration. To assess whether a poor correlation exists, the ten least-crowded solutions are selected from a given front according to the simple cuboid method described by Deb<sup>23</sup> [175] — with potential nearest-neighbours limited to the truncated set in one case and expanded to include any member of the complete non-dominated set in the other. The average observed correlation between both approaches (given as a percentage) across 20 distinct runs is illustrated in Table 7 and Figure 31.

<sup>23</sup> The approach suggested by Deb is modified slightly here such that solutions sharing a point in objective-space are treated as a single node in the non-dominated list. They therefore share an identical, non-zero, cuboid score.

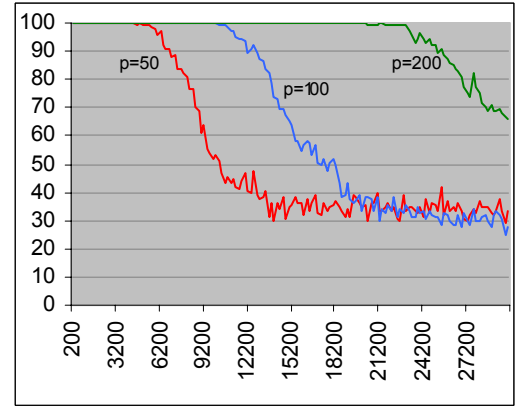
**Table 7 — Crowding Inaccuracies in Truncated Elite Sets**

Results indicate the average shared membership percentage between least-crowded sets produced with local and unbounded non-dominated lists. Averages are derived from twenty distinct NSGA-II runs per problem per population level, with least-crowded sets containing the ten least-crowded solutions of the non-dominated members in the supposedly elite leading front (referred to as  $\mathcal{F}_1$  in NSGA-II).

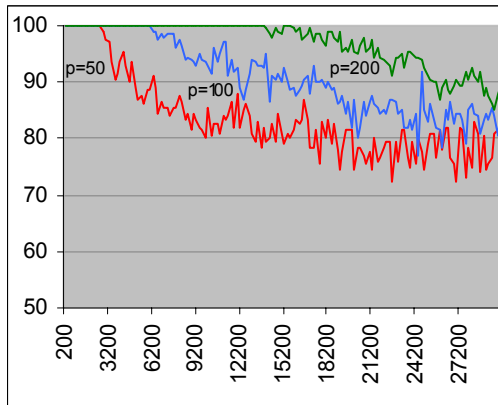
	Evaluations	AP-1	AP-2	AP-3	P1
Population Size = 50	0k — 10k	69.52	90.00	91.17	48.68
	10k — 20k	34.87	37.89	81.58	34.61
	20k — 30k	34.33	34.07	78.03	36.63
Population Size = 100	0k — 10k	95.01	100.00	98.44	69.77
	10k — 20k	36.89	68.27	90.47	31.42
	20k — 30k	31.63	33.09	81.04	31.94
Population Size = 200	0k — 10k	100.00	100.00	100.00	98.25
	10k — 20k	93.72	100.00	98.91	63.55
	20k — 30k	41.49	87.42	92.00	44.24



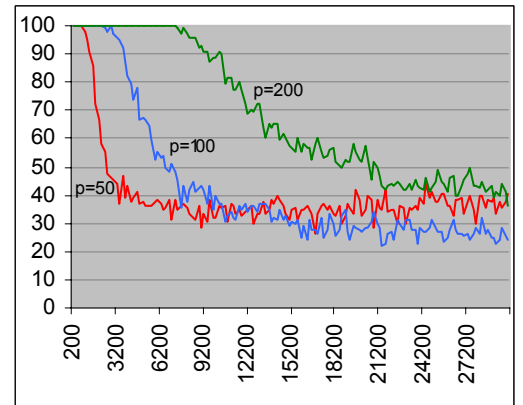
(a) AP-1



(b) AP-2



(c) AP-3



(d) P1

**Figure 31 — Crowding Inaccuracies in Truncated Elite Sets**

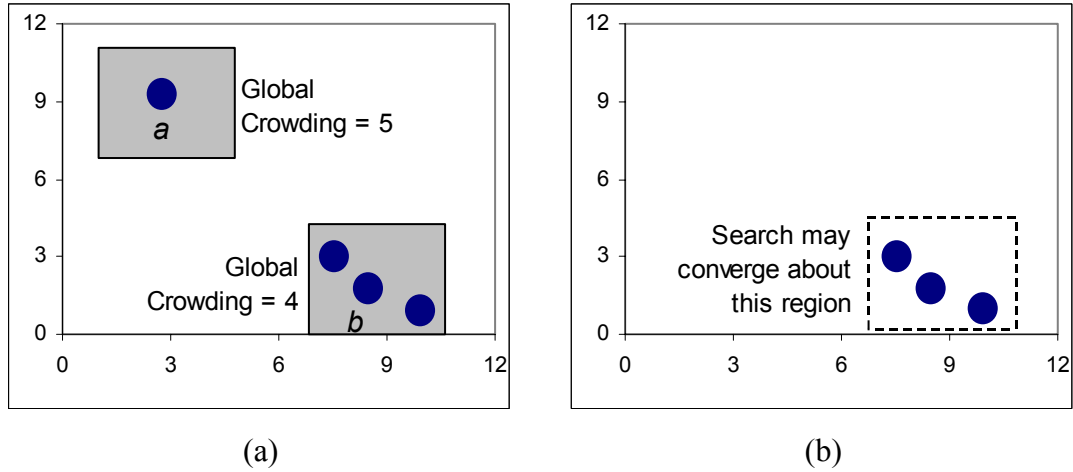
For all graphs, the *x-axis* represents the number of evaluations that the NSGA-II algorithm has performed, while the *y-axis* indicates the percentage of solutions that are shared between two sets: the ten least-crowded members of the  $\mathcal{F}_1$  front derived purely from local archival members, and the equivalent set produced when crowding calculations include the complete non-dominated front.

The results here are clear. Even for relatively high truncation thresholds, there is poor equivalency between the observed and actual distributions of the frontal solutions in objective-space, particularly as the optimisation process moves from a focus on frontal progression (earlier evaluations) towards frontal exploration (later evaluations). Better correlations are seen in the *AP-3* problem, but this is primarily due to the disjoint nature of the optimal front — the cuboid function will strongly bias the extreme members of any disconnected region and therefore, quite correctly, include these in the uncrowded set. It is the correct analysis of frontal distribution in the interior of continuous regions that is most problematic for the cuboid approach to crowding in truncated sets — the resolution of the continuous area is simply too coarse to correctly estimate those areas that have been poorly explored over the length of a run.

Thus, the claim is that the incomplete nature of truncated elite archives may lead to inaccurate frontal density information since the crowding measures are only ever operating on an incomplete picture of the objective-space. This notion is particularly significant given the regularity with which elite archives are truncated according to localised crowding information — most notably, in contemporary optimisation approaches such as SPEA2, PAES and NSGA-II.

It seems reasonable then, to apply crowding measures that consider all non-dominated solutions encountered throughout the run, rather than just those stored locally in the archive. However, such an approach is also open to problems. Consider Figure 32: if truncation is to be applied according to some global crowding measure, the archive will lose solution *a* and become overly concentrated around solution *b*. Such degeneration into locally crowded arrangements not only diminishes the likelihood of finding a richly distributed front efficiently (since the search will become focused around a single area of objective-space), but it also permits increased frontal oscillation

Crowding estimation in a truncated environment is therefore a complex problem. If the non-dominated set is too small to provide an accurate picture of the explored objective-space, it is also likely too small to capitalise on globally derived objective-



**Figure 32 — Dangers of Global Crowding Measures in a Truncated Environment**  
 $x$ -axis is objective one;  $y$ -axis is objective two. (a) With respect to *all* non-dominated solutions,  $a$  represents a (globally) crowded region. With respect to the truncated front, it is both isolated and uncrowded. (b) Truncation of  $a$  due to global crowding results in a poorly distributed truncated archive.

space information. Moreover, even if these issues are diluted via increasing truncation thresholds, there still exists a range of serious limitations in contemporary crowding metrics that may not be so easily dispelled (see Section 8.1).

#### 6.2.1.4 STOPPING CONDITIONS

Although the development of robust stopping criteria has long been held as an important area of future development in multiobjective optimisation, surprisingly little work has been focussed around the notion [97] (Dorn and Ranjithan [21], Hansen and Jaszkiewicz [186], and Jaszkiewicz [187] are amongst the only to offer even a preliminary investigation). Consider the most influential evolutionary algorithms that have been developed since Coello’s appeal for stopping conditions in 1999 [5] — of the papers introducing NSGA-II [175], SPEA2 [81], IBEA (the Indicator Based Evolutionary Algorithm) [91] and PESA [136], not one proposes the use of an alternative termination condition beyond an arbitrary evaluation threshold.

The failure to produce satisfactory end-of-run conditions may not rest as a particularly large problem for proof-of-concept and theoretical testing work — where hard evaluation thresholds are necessary to produce consistent conditions — but it is a substantial problem for real-world practitioners. If an arbitrary evaluation-threshold is used in practical studies, there is a real danger that the front may not have converged onto — or even near — a good approximation of the Pareto optimal

front. Conversely, if the front converges prematurely, then a large number of potentially costly evaluations will have been ill-used. Moreover, alternative methods, such as repetitive front interpretation (where a user must examine output, and the leading front specifically, at selected intervals), require intensive input from expert decision makers and are therefore expensive and permit the integration of human performance expectations into the system.

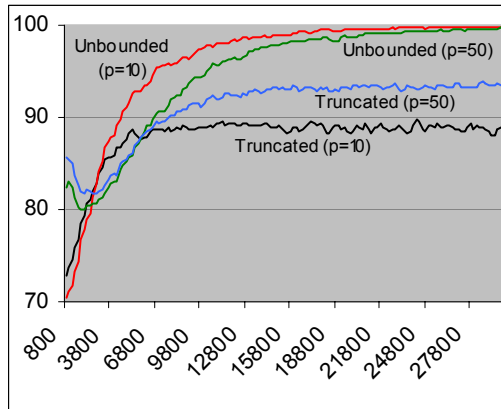
It seems obvious then that the construction of better termination conditions should be at the forefront of the continuing expansion of the multiobjective optimisation field. However, progress in this area is not as straight-forward as may be expected. The likely reason for the use of basic termination techniques in both traditional and contemporary works is that determining the rate of progression and expansion of the prevailing optimal front is fundamentally difficult in a truncated environment. An evolutionary algorithm may appear to be generating new members of some elite archive, but since a truncated elite set is subject to frontal degradation, the acceptance of new members says little about the true progression of a front. Similarly, recall the inaccuracies of crowd estimation in a truncated environment (see Section 6.2.1.3) — given such potentially poor approximations, gauging when a suitable end-of-run distribution has been achieved is also difficult. Thus, it seems probable that any stopping criterion formed around a truncated set is unlikely to yield suitable termination points.

As an illustration of the problems at hand, consider a convergence indicator that simply records the average percentage of non-beneficial solutions that are produced over a given time. If the indicator approaches a score of 100%, the vast majority of new solutions produced are superfluous — they offer no practical improvement to the corresponding non-dominated front. To determine whether a solution is beneficial, the objective-space is simply divided into identically sized squares (with a length of 0.001); any solution that resides in an unoccupied non-dominated square is deemed to be useful to the front — it is either moving it forward or otherwise investigating apparently unexplored regions. Table 8 and Figure 33 illustrate the average convergence indicator scores (the percentage of beneficial solutions

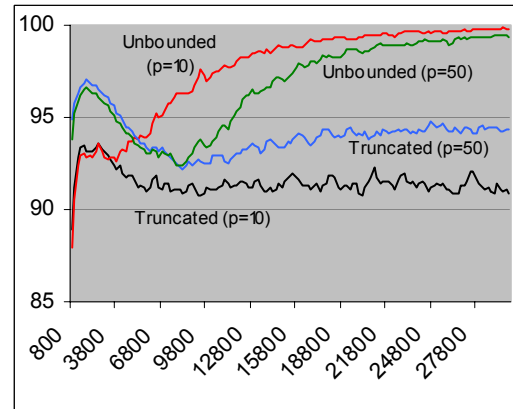
**Table 8 — Stopping Condition Performance in Truncated Sets**

Results indicate the average front performance indicator scores for truncated and unbounded archives. Averages are derived from twenty distinct NSGA-II runs per problem per population level.

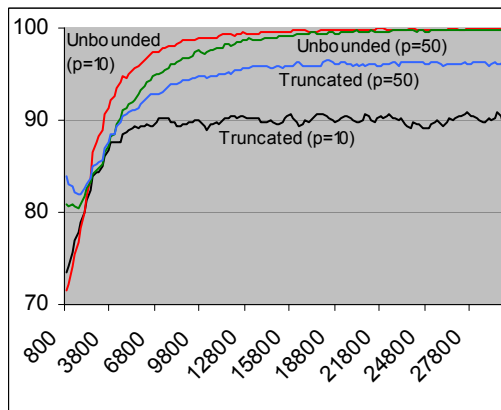
		Truncated			Unbounded		
		0k – 10k	10k – 20k	20k – 30k	0k – 10k	10k – 20k	20k – 30k
<b>AP-1</b>	<b>P=10</b>	80.40	88.98	88.83	83.87	98.83	99.72
	<b>P=50</b>	81.57	92.83	93.38	81.69	97.52	99.52
	<b>P=100</b>	83.15	94.67	96.47	82.10	96.17	99.10
<b>AP-2</b>	<b>P=10</b>	86.26	91.35	91.35	88.66	98.65	99.64
	<b>P=50</b>	88.58	93.58	94.30	88.48	96.89	99.37
	<b>P=100</b>	90.92	94.69	96.58	90.54	95.31	98.62
<b>AP-3</b>	<b>P=10</b>	81.38	90.05	89.99	86.52	99.51	99.89
	<b>P=50</b>	84.46	95.71	96.14	85.21	98.88	99.82
	<b>P=100</b>	85.93	97.57	98.28	85.44	98.49	99.61
<b>P1</b>	<b>P=10</b>	70.63	76.60	76.98	71.68	90.75	96.88
	<b>P=50</b>	70.61	76.74	76.55	72.18	91.64	96.63
	<b>P=100</b>	73.41	81.35	81.90	72.29	90.61	95.89



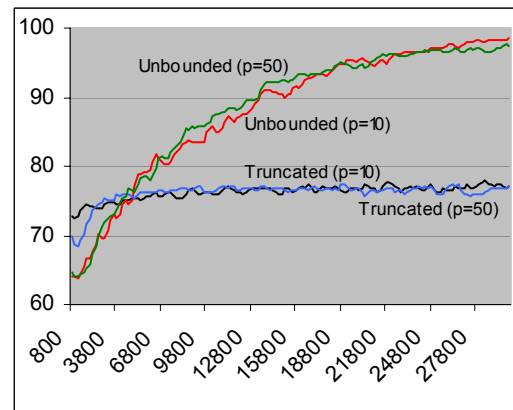
(a) AP-1



(b) AP-2



(c) AP-3



(d) P1

**Figure 33 — Stopping Condition Performance in Truncated Sets**

For all graphs, the  $x$ -axis represents the number of evaluations that the NSGA-II algorithm has performed, while the  $y$ -axis is the average front convergence indicator score.  $p$  is the truncated population size from which the results were obtained (note that unbounded archives obviously have no lower-size threshold; the population simply indicates the runs from which the indicators are derived).

generated over the preceding 500 evaluations) produced across twenty distinct NSGA-II runs (see Appendix B.1.1) per problem.

It is clear that the unbounded archives achieve consistently higher indicator scores than those generated from the limited sets (though the margin is dictated by the resolution of the truncated archive<sup>24</sup>). As expected, the frontal degradation inherent in truncated approaches means that solutions residing in dominated or pre-explored regions of objective-space are mistakenly accepted as beneficial, leading to inaccurate convergence estimates. Consider also the behavioural characteristics that may be extracted from the convergence graphs seen in Figure 33d — results for the truncated system indicate that the leading front is improving at a near-constant rate of approximately 23%. The reality, though, is that front progression is decelerating rapidly and, by the end of the run, is achieving a less than a 4% improvement rate; the fidelity between observed and actual convergence is poor. Such disparity is more than a theoretical concern — in practice there is little reason why a decision maker would voluntarily terminate a run that is apparently improving at both a high and constant rate, leading to an inappropriately excessive number of redundant evaluations.

Given these results, the development of termination conditions based on unconstrained elite archives is an important avenue of work. As will be explored in Chapter 7, elaboration and expansion of the convergence indicator outlined here represents one such piece of work — providing both concise and accurate frontal-behaviour information that leads to flexible and powerful termination conditions.

#### **6.2.1.5 SUMMARISING DEFICIENCIES IN GENERIC ELITE ARCHIVING**

The cost of maintaining unbounded archives in simple list structures has resulted in the majority of contemporary multiobjective optimisation algorithms (such as NSGA-II [82], PAES [143], PESA [136], SPEA2 [81] and IBEA [91]) using an approximation of an elite set that is reliant on frequent, and potentially aggressive,

---

<sup>24</sup> Higher population levels will tend to achieve more accurate convergence scores because there is a smaller fraction of squares that will be lost due to truncation. Still, unless the population size is greater than (or equal to) the number of squares required to properly encapsulate the optimal front, frontal degradation is still likely and the accuracy of the convergence indicator cannot be assured. Given that the shape and extent of the true Pareto front is rarely known *a priori*, setting archive thresholds that will generate consistently accurate convergence scores without incurring a significant run-time cost (as would occur with larger thresholds) is difficult, if not impossible.



truncations. While techniques using such a methodology have demonstrated impressive power, there exist a host of potentially serious drawbacks that affect both the performance and applicability of the optimisation process. In particular, truncation may induce frontal degradation (where weak solutions are mistakenly accepted into the supposedly elite set); the loss of potentially beneficial regions (when an archive is simply too small to express all of the important areas in objective-space); and poor crowding estimation (since any approximations are generally derived from an incomplete picture of the true front). Due to such issues, termination conditions are also difficult to produce, since little concrete information can be formed as to the true behaviour of the prevailing front (apparent front expansion may be caused by oscillation and apparent front progression may be a consequence of preceding frontal recession, for example).

Given the problems which stem from the simple list representation of the archive and the mechanisms required to curb the complexity burden it carries, it is unusual that more work has not been directed at improving or finding better suited data structures or algorithms. Indeed, only Fieldsend *et al.* [97], Mostaghim *et al.* [96] and Jensen [95] have offered suitable alternatives via augmented tree structures. As will be seen in Section 6.2.2, while these approaches each yield efficiency gains when applied to the generic case, there may be scope for further refinement and improvement in the special two-dimensional case.

### 6.2.2 EXISTING UNBOUNDED ARCHIVING TECHNIQUES

Unbounded archives only become feasible for use in practical multiobjective optimisation systems if they gain an obvious performance advantage over the existing  $O(\beta n)$  time costs<sup>25</sup> associated with naïve linear list storage techniques, while incurring only a marginal increase in the storage costs seen in truncated approaches. With this in mind Fieldsend *et al.* [97], Mostaghim *et al.* [96] and Jensen [95], each offer data structures and updating procedures that are suitably efficient to be viable alternatives to the prevailing truncation-based methodologies. This section will explore the steps made by these researchers towards a powerful new standard in multiobjective optimisation thought, and those problems that have thus far curtailed community enthusiasm for unbounded archiving.

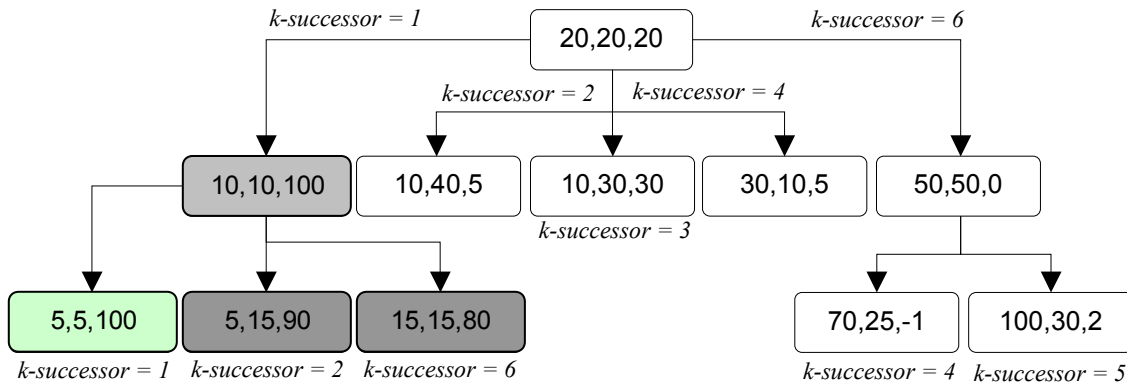
---

<sup>25</sup> Recall that  $\beta$  is the number of objectives and  $n$  is the number of members stored in a population (the archive, in this case).

### 6.2.2.1 MOSTAGHIM QUAD-TREES

The first series of algorithms described specifically for unbounded archiving are the three variants proposed by Mostaghim, Teich and Tyagi [96] (namely, *Quad-Tree1*, *Quad-Tree2* and *Quad-Tree3*). All three approaches result in the storage of non-dominated solutions in quad-trees<sup>26</sup> (as illustrated in Figure 34) and differ primarily in the way deletion occurs. Specifically, Quad-Tree1 must re-insert all descendants of a deleted node; Quad-Tree2 instead checks descendants of a deleted node for domination and flags these for subsequent extraction; while Quad-Tree3 moves the smallest  $k$ -successor into the position of the dominated node and re-inserts all members of the remaining  $k$ -successor sub-trees at this point. As an example, if the shaded (10,10,100) node is dominated in the quad-tree illustrated in Figure 34, then the Quad-Tree1 algorithm requires the re-insertion of all shaded descendent nodes; Quad-Tree2 must check (and potentially remove) the same set of nodes; and Quad-Tree3 moves the green descendent up one level and re-inserts the darkly shaded descendants at this point.

Regardless of approach however, the proposed algorithms are beset with the same general class of problems — that is, potentially expensive tree maintenance under the



**Figure 34 — An Example Mostaghim Quad-Tree for Unbounded Elite Archiving**

$k$ -successorship is given by  $\sum_{i=1}^{\beta} k_i \times 2^{\beta-i-1}$ , where  $\beta$  is the number of objectives and  $k_i$  is zero if the child node's  $i^{\text{th}}$  objective score is less than the  $i^{\text{th}}$  objective score of the parent, or one otherwise.

<sup>26</sup> For those unfamiliar with the quad-tree data structure, it may be useful to know that each node contains at most four branches (hence *quad*) *only* when labels are two-dimensional, as in the bi-objective case. When  $\beta$ -dimensional labels are required (when addressing  $\beta$  objectives), each node can have a maximum of  $2^{\beta}$  branches (making the quad-tree name more than a little misleading here). To further confuse matters, the Mostaghim quad-trees *never* require the first or last branches, so the maximum number of children from each node is in fact  $2^{\beta}-2$ .

insertion of dominating nodes and an excessive number of (largely unnecessary) dominance checks. Moreover, empirical evidence displayed in their own work [96] suggests that the approach is more costly than a simple linear list for a large number of evaluations. Indeed, on the relatively simple *ZDT1* function developed by Zitzler, Deb and Thiele [188], the variants are slower on average than an unbounded list even after 400,000 evaluations. Thus, the proposed Mostaghim quad-tree algorithms are ultimately of limited practical worth (and will therefore not be discussed further in this work), but provided the impetus for similarly motivated ideas to be explored.

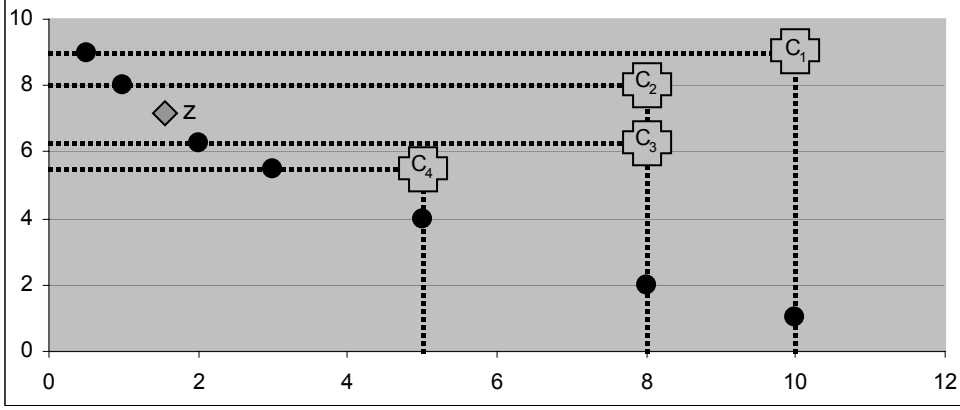
### 6.2.2.2 DOMINATED TREES

Improving on the performance of the Quad-Tree algorithms, Fieldsend, Everson and Singh [97] introduce both a new selection mechanism for unbounded archives — namely, Partitioned Quasi-Random Selection (PQRS) — and efficient data structures for the storage and maintenance of archival solution sets — specifically, Dominated and Non-dominated Trees.

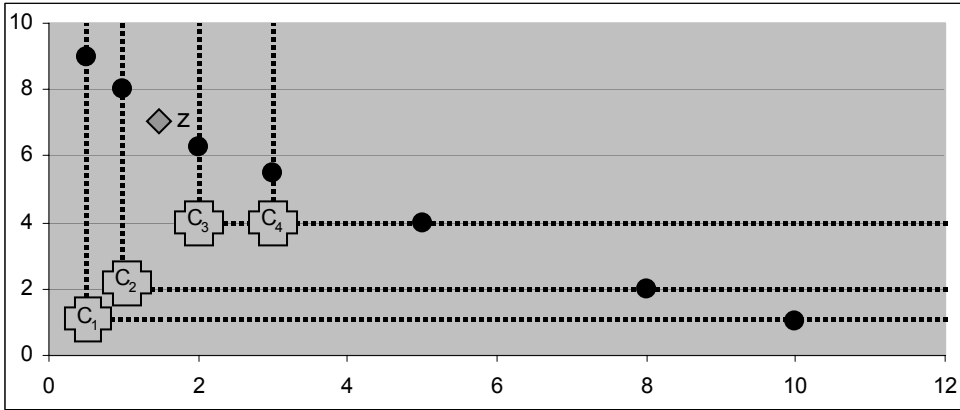
With respect to selection, a forest of  $\beta$  self-balancing trees is proposed (referred to as PQRS trees), with each objective-based tree partitioned into  $(e-1)$  identically sized bins for solution storage (where  $e$  is the number of solutions to be selected). With the data structure in place, an approximation of a uniform sampling of any dimension is achieved by simply selecting a single solution from each bin in the corresponding tree (unless a bin is empty, in which case a nearest neighbour is selected). Note that to preserve the extent of the front, Fieldsend *et al.* [97] also suggest the inclusion of extreme solutions in each selection set.

In contrast, maintenance of the archive occurs in two data structures — the Dominated Tree and the supporting Non-dominated Tree<sup>27</sup>. As illustrated in Figure 35, both approaches are similarly composed of approximately  $\lceil n/\beta \rceil$  unique composite points: combinations of the objective-results from multiple independent solutions (constituents). Importantly, no composite point is ever incomparable with another, and so the weakly-dominates relation can impose a complete order on any set of

<sup>27</sup> Perhaps confusingly, Fieldsend *et al.* [97] also use *Dominated Tree* to denote the PQRS, Dominated and Non-dominated Trees *in toto* — and results indicating the performance of Dominated Trees refer to this grouping, as opposed to the singular data structure. This work follows suit, since no alternative group name is provided.



(a) Dominated Tree



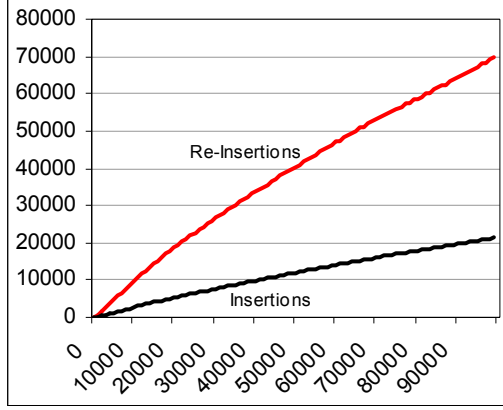
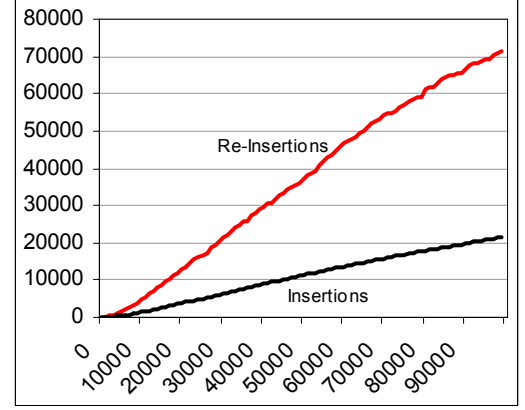
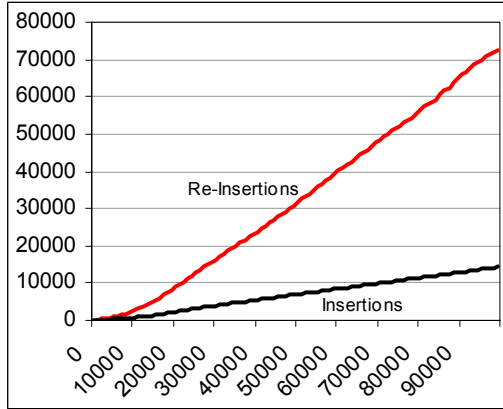
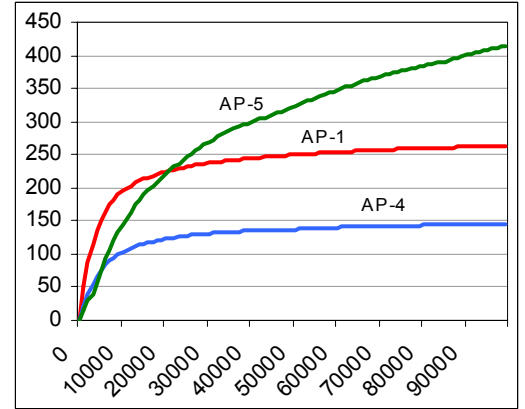
(b) Non-dominated Tree

**Figure 35 — Example Dominated and Non-dominated Trees**

Dashed lines illustrate the constituents of each composite point;  $z$  is a query point. Note that the illustrated trees are sub-optimal as some constituents are used in multiple composite points; only periodic cleaning of the tree can alleviate the inefficiencies caused by this phenomena.

points — thus facilitating the use of ordered data structures, such as balanced binary search trees.

Empirically, Fieldsend *et al.* [97] demonstrate the efficiency, and efficacy, of using the proposed mechanisms, with particularly impressive timing results when compared to the performance of an unbounded linear list. However, the approach is not without its limitations. In particular, the selected Dominated Tree structure may sporadically require complete re-building in order to maintain a suitable approximation of the optimal number of composite points ( $\lceil n/\beta \rceil$ ). As illustrated in Figure 36, which displays averages based on twenty distinct NSGA-II runs per

(a) *AP-1*(b) *AP-4*(c) *AP-5*(d) *P1***Figure 36 — The Cost of Rebuilding Dominated Trees**

For all graphs, the  $x$ -axis represents the the number of solutions (taken from NSGA-II runs) presented to the archive. (a), (b) and (c) display the average total number of solutions successfully added to the Dominated Tree and the average total number of solutions that must be re-inserted when naïve tree rebuilding is required (with cleaning occurring when the number of composites exceeds 60% of the archive size). (d) shows the average total number of Dominated Tree rebuilds required on each tested problem.

problem (see Appendix B.1.1), the potential for such rebuilding is neither infrequent nor low in impact. Recalling that a naïve approach to tree rebuilding will come at a cost of  $O(n \log n)$  for a tree with  $n$  members (even if a PQRS Tree is referenced), the need for complete reconstruction is expensive indeed, particularly given the unbounded nature of  $n$ . Furthermore, to determine non-dominance of a solution with respect to the archive — and additional to the normal binary search — the constituents of composite points that are incomparable with the applicant solution must each be checked for dominance in-turn. Referring to the example in Figure 35, this means that incoming solution  $z$  must be compared against constituents of



original proposal, particularly on problems of low dimensionality, lies as an important area of future work. In particular, in the special bi-objective case, a one-dimensional range tree may be all that is required if the properties outlined in Section 6.1 are capitalised upon, thus negating the need for fractional cascading and secondary level sub-trees.

## 6.3 INTRODUCING THE **MAK\_TREE**

All unbounded archiving algorithms proposed thus far are, quite reasonably, designed for generic  $\beta$ -objective non-dominated sets. However, as introduced in Section 6.1, the bi-objective case carries a number of unique properties that may be manipulated to form more efficient specialised data structures and storage algorithms. The *Mak\_Tree*<sup>28</sup> algorithm represents such an approach, delivering a highly efficient, simple, flexible and low-cost technique that is specifically tailored to the needs of bi-objective optimisation.

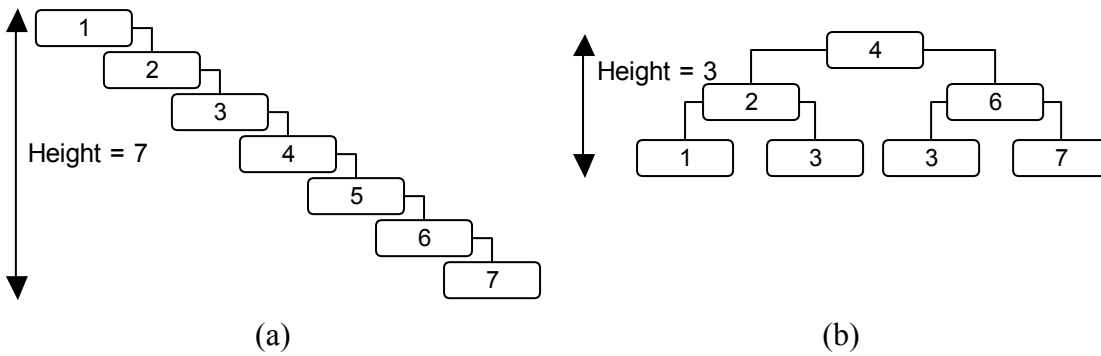
### 6.3.1 THE **MAK\_TREE** DATA STRUCTURE

The *Mak\_Tree* is a generic label for any binary tree structure that is dynamic, self-balancing and ordered (arbitrarily) by performance on the first objective of a bi-objective problem. A node in any *Mak\_Tree* represents a collection of solutions with identical objective scores (to enable Property Two, as described in Section 6.1.3) while every member of the tree is strictly non-dominated. As such, the tree itself is not particularly interesting and can take on a wide variety of forms, from AVL structures [190, 191] to the Red-Black trees [192, 193] used herein. It is the *Mak\_Tree* algorithms, which build upon the unique properties of bi-objective sets and binary search trees, that are interesting and will form the focus of this work.

Still, to place the behaviour of the *Mak\_Tree* in context — and to bridge the gulf between theoretical discussions and practical implementations — it is beneficial to offer at least a brief exploration of the Red-Black tree structure that forms the foundation upon which the *Mak\_Tree* algorithms build. Again, it is worthwhile noting that the Red-Black tree represents but one of the wide range of self-balancing binary tree structures that are suitable for *Mak\_Tree* application; its choice is largely arbitrary, though its simplicity and popularity (see Appendix F.1 for example source

---

<sup>28</sup> The *Mak\_Tree* is so named for an important person in the author's life.



**Figure 38 — Unbalanced and Balanced Binary Search Trees**

(a) A traditional (unbalanced) binary search tree formed under the insertion of ever-ascending numbers.  
 (b) A balanced equivalent of the tree found in (a).

code in a variety of programming languages) are valuable factors in facilitating straight-forward Mak\_Tree implementations.

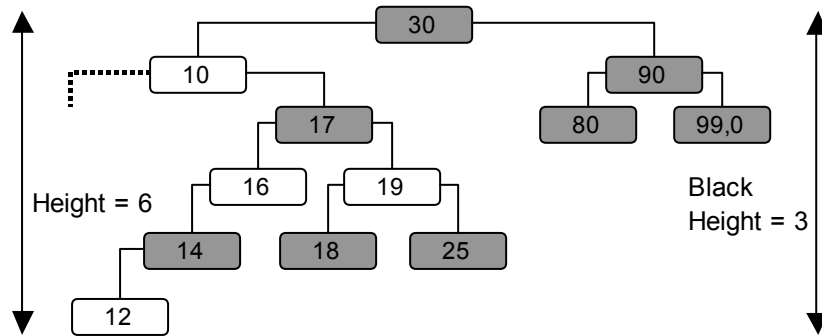
### 6.3.2 RED-BLACK TREES

The Red-Black tree — borne out of seminal works by Bayer [193] and Guibas and Sedgwick [192] — is representative of a host of innovative data structures developed during the 1960's and 1970's that attempted to rectify inefficiencies inherent in standard binary search trees ([194, 195] offer succinct reviews and analyses). Specifically, traditional binary structures are prone to becoming unbalanced, where the height of a tree may vary tremendously from the optimal ( $\log_2 n$ ). The consequence is that the performance of standard binary search trees is subject to the order of updates (see Figure 38 for a classic, though pathological, example) — and the  $O(\log n)$  search and insert bounds become unreliable in practice. As such, the Red-Black tree uses simple localised (colour-based) information to maintain a pseudo-balanced structure that adheres to  $O(\log n)$  bounds.

The power of the Red-Black tree comes from a number of simple properties that must always hold true. Specifically, a Red-Black tree is composed of nodes that are coloured red or black, such that:

- The root is always black;
- No two red nodes may be directly connected to each other;
- Each newly inserted node is initially coloured red; and
- The tree (and any given sub-tree) is perfectly balanced when considering only the black nodes.





**Figure 39 — An Extremely Unbalanced Red-Black Tree**

Black nodes feature heavy shading, while red nodes are unshaded. Note that the tree is balanced when considering only black nodes, but unbalanced when considering all nodes. Adding any element beneath the 12-node would result in the tree being re-balanced via a rotation and re-colouring.

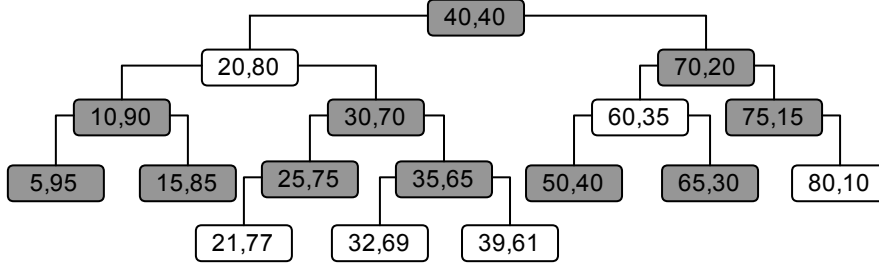
Since the tree is balanced under the consideration of black nodes and no two red nodes may be connected, it follows that the most a tree can be unbalanced is by a factor of two (see Figure 39 for an example) and the maximum height is  $\lfloor \log_2 2n \rfloor$ . Thus, while the Red-Black tree can only claim an approximation of balance, the difference is negligible in practice and  $O(\log n)$  bounds apply.

Updating a Red-Black tree proceeds precisely as it would in a standard binary search tree, though with the added burden of ensuring that the aforementioned Red-Black properties hold true. While conceptually complex (and beyond the scope of this overview), property maintenance requires at most one rotation and  $O(\log n)$  node re-colourings, thus ensuring that the total time costs related to insertion and deletion do not expand beyond  $O(\log n)$ .

By tightening the worst-case bounds — and by doing so with no practical cost beyond a slight increase in implementation complexity — the Red-Black tree offers the type of consistent performance that is beyond conventional binary search trees, but is necessary for use in time-critical real-world systems.

### 6.3.2.1 A RED-BLACK MAK\_TREE

When the properties of the Red-Black tree are combined with the simple objective-based ordering requirements of the Mak\_Tree, it gives rise to the type of structure exemplified in Figure 40. Note that the structure requires only minimal changes to



**Figure 40 — An Example Red-Black Mak\_Tree**

Black nodes feature heavy shading, while red nodes are unshaded. Note that the tree is ordered by performance on the first objective.

the basic Red-Black tree, and should therefore be accessible and simple to implement (especially given the host of pre-existing Red-Black tree implementations that are readily available, see Appendix F.1).

### 6.3.3 UPDATING THE MAK\_TREE

It is obvious that for a Mak\_Tree to remain non-dominated, it must only accept non-dominated solutions into the tree and prune any solution that becomes dominated due to an insertion. The basic algorithm to achieve such behaviour is outlined in Algorithm 2, where solution  $a$  is inserted into archive  $A$ . To further illustrate the behaviour of the update operations introduced in the algorithm, it is beneficial to consider two general concepts: verifying non-dominance and locating dominated nodes.

#### 6.3.3.1 VERIFYING NON-DOMINANCE

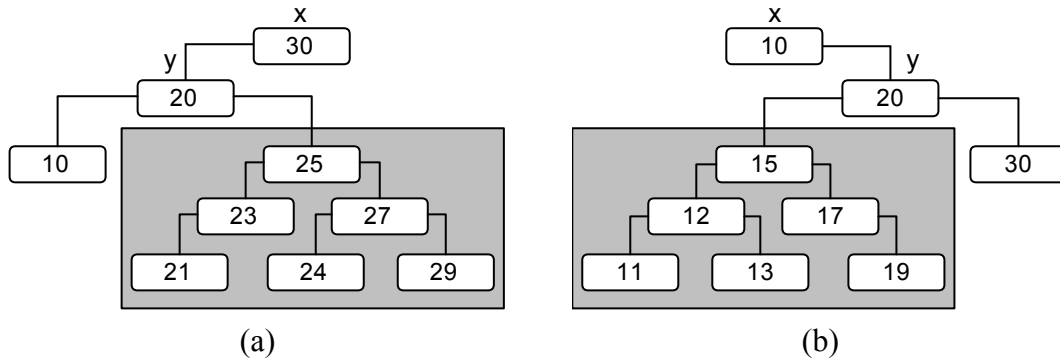
In order for a proposal to be inserted into the archive, it must not be dominated by any other stored solution. Considering the procedure adopted in Algorithm 2, it is not necessarily intuitive how this is achieved. However, recall that a unique property of ordered bi-objective non-dominated fronts is that if a solution is non-dominated with respect to its predecessor, then it is also non-dominated with respect to the remainder of the front (Property Four, Section 6.1.5). Since insertion mirrors simple binary-tree insertion, and this means that the proposal will always be compared with its predecessor (unless it is dominated before this point or no such predecessor exists), any successfully added solution is guaranteed to be non-dominated.

**Algorithm 2 — Insertion Into The Mak\_Tree****Inputs:**

<b><math>a</math></b>	The solution to be inserted into the archive (tree) $A$ .
1: if ( $A^{root} = \text{null}$ )	If the tree is empty (root is null) add a new
2: $A^{root} := \text{createNode}(a)$	node containing the solution to the tree.
3: else	
4: $rejected := \text{false}; del\_nodes := \{\}$	
5: $del\_subs := \{\}; b := \text{null}; node := A^{root}$	Otherwise, start the search at the root of $A$ .
6: while ( $(node \neq \text{leaf})$ and ( $rejected \neq \text{true}$ ))	
7: if ( $node \prec a$ )	If the current node dominates $a$
8: $rejected = \text{true}$	then the algorithm ends.
9: else if ( $a \prec node$ )	If the solution dominates the
10: handle_dominance( $a, node, B,$ $del\_nodes, del\_subs$ )	current node then call the
12: $node := \text{left\_or\_insert}(node, a)$	handle_dominance helper (Algorithm 3).
13: else if ( $a = node$ )	Move left, or insert a new node with $a$ if at a leaf.
14: $node := node \cup \{a\}$	If the solution and node share
15: $rejected := \text{true}$	objective scores, add $a$ to $node$
16: else if ( $f_1(a) < node^{obj1\_label}$ )	and end the algorithm.
17: $node := \text{left\_or\_insert}(node, a)$	If the objective-one score of $a$
18: else	is less than that of $node$ , move
19: $node := \text{right\_or\_insert}(node, a)$	left or insert $a$ if $node$ is a leaf.
20: if ( $(B \neq \text{null})$ and ( $rejected = \text{false}$ ))	Otherwise, move/insert to the right of $node$ .
21: delete_all_sub_trees( $del\_subs$ )	If the solution was dominating
22: delete_all_nodes( $del\_nodes$ )	then remove all dominated
	nodes and sub-trees.

**Algorithm 3 — Handle\_Dominance Helper****Inputs:**

<b><math>a</math></b>	The inserted solution.
<b><math>node</math></b>	The dominated node being examined.
<b><math>b</math></b>	The first found dominated node.
<b><math>del\_nodes</math></b>	The set of individual nodes that are dominated by $S$ .
<b><math>del\_subs</math></b>	The set of sub-trees that are dominated by $S$ .
1: $del\_nodes := del\_nodes + node$	
2: if ( $b = \text{null}$ )	If $b$ has not yet been found,
3: $b := node$	then label this $node$ as $B$ and find
4: $current := b^{rightChild}$	all dominated nodes to the right of it.
5: while( $current \neq \text{null}$ )	
6: if ( $f_2(a) \leq current^{obj2\_label}$ )	If the solution dominates the
7: $del\_nodes := del\_nodes \cup \{current\}$	current node, both the node and
8: if ( $\text{left}(current) \neq \text{null}$ )	its left sub-tree (if it exists)
9: $del\_subs := del\_subs \cup \{current^{leftChild}\}$	should be deleted.
10: $current := current^{rightChild}$	Move right.
11: else	The solution can only dominate
12: $current := current^{leftChild}$	nodes to the left, so move left.
13: else if ( $node^{rightChild} \neq \text{null}$ )	If $B$ has been found, all nodes
14: $del\_subs := del\_subs \cup \{node^{rightChild}\}$	to the right of $node$ must be dominated.

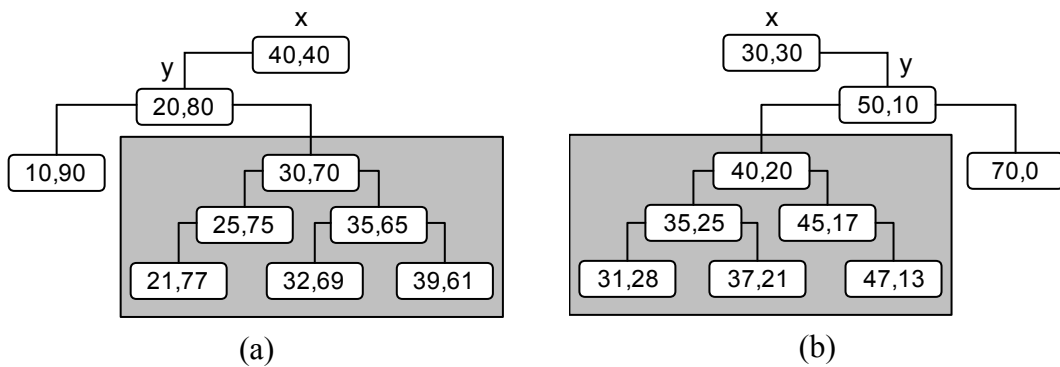


**Figure 41 — Illustrating the Range Properties of Binary Search Trees**

Any node in the sub-tree represented by the shaded region will always feature values between  $x$  and  $y$ . Specifically, for (a) the shaded region will always contain nodes with values between 20 ( $y$ ) and 30 ( $x$ ), while the shaded region in (b) will always contain values between 10 ( $x$ ) and 20 ( $y$ ).

### 6.3.3.2 LOCATING DOMINATED NODES

Central to the algorithm for locating dominated nodes is the concept of dominated sets introduced in Property Three (Section 6.1.4) — that is, if a solution dominates nodes at index  $i$  and index  $j$ , it must also dominate the set of nodes with indices between  $i$  and  $j$ . Since this essentially represents a range-query, it is useful to build on the range-related properties inherent in binary search trees. Specifically, it is always the case that if a node  $y$  is the left-child of  $x$ , then all right-descendents of  $y$  will have in-order indices between the indexes of  $x$  and  $y$  (see Figure 41a). This property also holds if  $y$  is a right-child of  $x$  and all left-descendents of  $y$  are considered (as demonstrated in Figure 41b). The effect is that once an initial dominated node has been identified, any dominated node to its right is guaranteed to have a dominated left sub-tree, while any dominated node to its left must have a dominated right sub-tree (as illustrated in Figure 42).

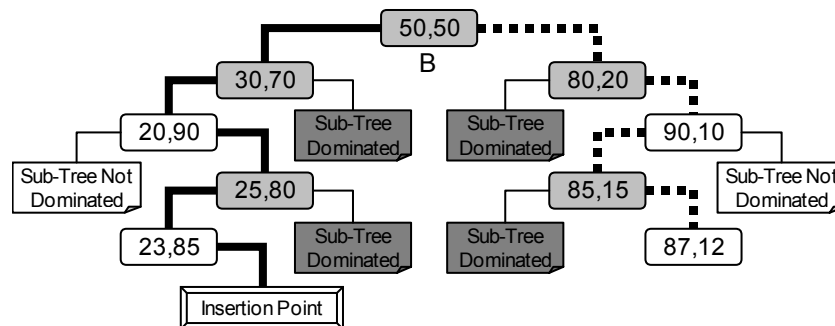


**Figure 42 — Illustrating the Range Properties of Mak\_Trees**

If both  $x$  and  $y$  are dominated, any node in the shaded regions *must* also be dominated.

With this in mind, locating dominated nodes becomes relatively straight-forward and particularly efficient in the Mak\_Tree algorithm. After the discovery of the first dominated node  $b$  at index  $h$ , the location of all dominated nodes with indices between  $h+1$  and  $j$  (the right-most member of the complete dominated set) requires at most  $O(\log n)$  comparisons. Specifically, the search proceeds as follows (starting at the right-child of  $b$ ): if the current node is dominated label it and the left sub-tree as dominated and search the right sub-tree; otherwise, everything to the right of the current node is non-dominated, so move left. The discovery of all nodes with indices between  $i$  (the left-most member of the complete dominated set) and  $h-1$  requires little modification to the standard insertion procedure — on dominance, the search progresses left as usual, with the node and right sub-tree marked as being dominated. Figure 43 provides an example of the procedure in action (with a further example offered in Appendix F.3.1).

Once discovered, the dominated nodes must be removed from the tree (lines 20–22 in Algorithm 2). It is important to note that dominated sub-trees (the heavily shaded boxes in Figure 43) and the individual dominated nodes (the lightly shaded cells in Figure 43) are handled independently during this deletion procedure. As evidenced in Section 6.4.1.2.2, the deletion of large sub-trees, in particular, can afford the Mak\_Tree algorithm an impressive performance gain.



**Figure 43 — Locating Dominated Nodes in an Example Mak\_Tree**

Assume that a solution with results (24,14) is being inserted into the tree and that  $B$  is the first dominated node to be identified. The dashed bold line represents the  $O(\log n)$  path taken to identify all dominated nodes to the right of  $B$ . The solid bold line represents the  $O(\log n)$  path taken to both identify all dominated nodes to the left of  $B$  and insert the solution at the correct location in the tree. For clarity, individual dominated nodes are lightly shaded, while dominated sub-trees feature heavy shading.

## 6.4 PERFORMANCE OF THE MAK\_TREE

### 6.4.1 COMPLEXITY ANALYSIS

When considering the performance of a given data structure and algorithm, both time and space complexity are significant. While emphasis is typically placed on efficiency, high space complexity can induce tighter limits on the feasible capacity of a given structure. Since the very nature of unbounded archives is to store particularly large solution sets, it is important to avoid such limits.

#### 6.4.1.1 SPACE COMPLEXITY

The optimal space complexity for any unbounded archive is  $O(n)$ , as all solutions in the archival set must be accessible. Since the Mak\_Tree contains at most  $n$  nodes and the simple nature of the tree requires no repetition of nodes or solutions, the space complexity of the Mak\_Tree is optimal and equal to  $O(n)$ .

The Dominated Tree structures achieve similarly optimal space complexity, though they require cleaning at appropriate thresholds to ensure that such optimality holds. Without cleaning, constituents may contribute to multiple composites and the spatial cost will increase accordingly.

Finally, the Dynamic Range Tree suggested by Jensen requires  $O(n \log n)$  space (see [196]) due to its two-level tree structure and thus represents the most expensive storage option presented in the literature thus far.

#### 6.4.1.2 RUN-TIME COMPLEXITY

With respect to performance complexity, it is useful to consider the insertion of two distinct types of solution: strictly non-dominating — those that are dominated by or equal to some component of the archive, or otherwise completely incomparable — and dominating.

##### 6.4.1.2.1 Insertion of Non-dominating Solutions

As alluded to in Section 6.3.3.1, the Mak\_Tree algorithm is particularly efficient when inserting non-dominating solutions, as the algorithm requires at most  $O(\log n)$  dominance comparisons during the simple binary navigation. Since insertion in Red-Black Trees will only ever require at most  $O(\log n)$  node re-colourings and one rotation, the worst-case time cost for the insertion of any non-dominating solution in

the Mak\_Tree is  $O(\log n)$ . Such performance is optimal for any structure based on binary search trees.

The Dominated Tree structures achieve similar performance, though the need for linear checking of composite constituents may be costly. If the solution under consideration is dominated, then performance is optimal and requires only  $O(\log n)$  dominance comparisons. However, the algorithm is sub-optimal under the insertion of strictly incomparable solutions, with the burden of checking  $c$  constituents for dominance resulting in a search cost of  $O(\log n + c)$ . While not discussed explicitly in the source paper, it also seems likely that verification of solution equality would require the checking of constituents belonging to the composite point sharing an axis with the solution — thus leading to sub-optimal performance if the inserted solution already has an equivalent stored. Additionally, note that these time complexities only hold under suitable maintenance of the corresponding Dominated Tree and assume that the need for cleaning is infrequent (which cannot be guaranteed in general).

A query in the two-dimensional fractional cascading Dynamic Range Tree will cost  $O(\log n \log(\log n) + \alpha)$ , where  $\alpha$  is the set of solutions satisfying the range query (see [196] for succinct summaries of Dynamic Range Tree behaviours and costs). For any insertion, the algorithm will require at most two orthogonal range queries — the first identifies those solutions that dominate the proposal, while the second highlights archival members dominated by the proposal. For a non-dominating solution, the second query will always return the empty set, while the first query need only return the first dominating node (since any  $\alpha > 0$  is enough to disqualify the incoming solution from inclusion). Thus, the total query cost for identifying a non-dominating node is  $O(\log n \log(\log n))$ . Since the update cost of the structure is also  $O(\log n \log(\log n))$ , the total time cost for the insertion of a non-dominating solution into the Dynamic Range Tree is  $O(\log n \log(\log n))$ .

#### 6.4.1.2.2 Insertion of Dominating Solutions

Handling dominating solutions is an inherently more expensive proposition as it requires both the identification of dominated solutions and their subsequent removal. As discussed in Section 0, all  $\eta$  dominated nodes can be efficiently discovered with  $O(\log n)$  dominance comparisons using the Mak\_Tree. If the naïve approach is taken

and each of the  $\eta$  dominated nodes are deleted in-turn, the final cost of identifying and removing dominated nodes with the Mak\_Tree is  $O(\eta \log n)$ . However, given that the query may return dominated sub-trees, it is useful to capitalise on sub-tree deletion in Red-Black Trees. Specifically, for any sub-tree requiring deletion, the cost of removing the sub-tree is  $O(\log \tau \log n)$  rather than  $O(\tau \log n)$ , where  $\tau$  is the size of the sub-tree. For small  $\eta$  the difference is not particularly impressive, but as  $\eta$  increases, so must the size or number of sub-trees identified for deletion. The effect is particularly evident in analysing the worst-case insertion bound, where the number of dominated nodes tends towards the size of the tree<sup>29</sup> ( $\eta \approx n$ ). In this case there will be approximately  $(2 \log n)$  sub-trees and  $(2 \log n)$  separate individual nodes that require deletion. The individual nodes will cost a total of  $O((\log n)^2)$  to remove (though the amortised cost of this operation can be reduced if individual deletion occurs after sub-tree deletion is completed). Both left and right of the root, deleted sub-tree sizes follow the pattern  $(2^2-1, 2^3-1, 2^4-1, \dots, 2^{(\log n)-1})$  and so the cost of deleting all sub-trees is:

$$\begin{aligned}
 & \left( 2 \times (\log(2^2 - 1) + \log(2^3 - 1) + \log(2^4 - 1) + \dots + \log(2^{(\log n) - 1} - 1)) \right) \times \log n \quad (61) \\
 & = (2 \times (1 + 2 + 3 + \dots + (\log n) - 2)) \times \log n \\
 & \Rightarrow O((\log n)^2 \times \log n) \\
 & = O((\log n)^3)
 \end{aligned}$$

Thus, the total worst-case bound of identifying and removing dominated nodes from a Mak\_Tree is  $O((\log n)^3)$ .

The Dynamic Range Tree can identify all  $\eta$  dominated solutions using an orthogonal query that costs  $O(\log n \log (\log n) + \eta)$ . Once discovered, the dominated solutions must then be removed from the tree at a cost of  $O(\eta \log n \log (\log n))$ . Thus, when worst-case insertion occurs, the Dynamic Range Tree carries a cost of  $O(n \log n \log (\log n))$ .

To locate all  $\eta$  nodes for deletion, the Dominated Tree must perform two binary searches and subsequently check every constituent of dominated composite points. Thus, the cost of locating all dominated nodes in the archive will be at least

---

<sup>29</sup> Though it is actually trivial to handle complete domination: if the left-most and right-most nodes in the tree are dominated, the inserted node simply becomes the root of a new Mak\_Tree.



$O(\log n + \eta)$  and, more generally,  $O(\log n + \eta + \delta)$ , where  $\delta$  is the number of constituents belonging to dominated composite points. To minimise this cost, Fieldsend *et al.* [97] propose the use of the alternative (though conceptually very similar<sup>30</sup>) Non-dominated Tree structure which ensures that any dominated composite point will feature only dominated constituents. While offering some performance gain, constituents belonging to incomparable composite points will still need to be verified. Thus, the cost of locating dominated nodes using the Non-dominated Tree variant will be  $O(\log n + \mu)$ , where  $\mu$  is the number of incomparable composite constituents that must be examined.

Upon the deletion of any solution, the corresponding Dominated Tree structure must update all composite points for which the solution was a constituent. If the solution represented is the only constituent of a composite point, the composite point will be removed from the archive in  $O(\log n)$  time (assuming use of a self-balancing tree). If the solution was used in the most dominating composite point then, in the two-dimensional case, the remaining constituent of the composite represents the new coordinates of the dominating point and the update can be completed in constant time. Otherwise, the affected composite point is updated through the re-use of a constituent of the succeeding (dominated) composite — requiring an  $O(\log n)$  successor search. Thus, the cost of deleting a single solution from any Dominated Tree is  $O(v \log n)$ , where  $v$  is the number of composite points that the solution is a member of. Deletion of  $\eta$  solutions will cost  $O((u+\eta) \log n)$ , where  $u$  is the total number of extra composite points to which the  $\eta$  dominated solutions belong.

Note that some practical improvements can be made over this performance if it is known when all constituents of a composite point are dominated (as is the case for certain composite points in Non-dominated Trees). Given this knowledge, the completely dominated composite point can be deleted from the tree without the need for intermediary coordinate re-labelling. However, given that the constituents of the dominated composites may also form part of non-dominated composites elsewhere in the tree, the overriding complexity costs remain much the same unless frequent cleaning occurs. Better improvements are seen when applying binary sub-tree deletion to remove sets of completely dominated nodes, though the need to check

---

<sup>30</sup> So similar, in fact, that the performance of the Non-dominated Tree echoes that of the Dominated Tree for all but the special deletion case.

and update constituents in non-dominated composite points curb the advantages afforded to such an approach when  $\eta$  is small. Still, such techniques may hold merit and are certainly worth further consideration in future work.

#### 6.4.1.2.3 Amortised Cost of Inserting Dominating Solutions into Mak\_Trees

The cost of deletion in a Mak\_Tree may appear prohibitive, but the worst-case scenario is misleading in practice. This discrepancy between theory and practical reality is best illustrated via amortised time analysis (for those unfamiliar, Cormen, Leiserson and Rivest [197] provide a straight-forward introduction into amortised complexity and its value to real-world practitioners). For any number of insertions  $\chi$ , the maximum number of deletions that can occur is also  $\chi$  (fairly obviously, the Mak\_Tree cannot remove more than  $\chi$  solutions or the archive would have fewer than zero members). In this case, the amortised complexity of maintaining the Mak\_Tree on an insertion-by-insertion basis is:

$$O\left(\frac{\chi(\log n) + (\chi \log n)}{\chi}\right) = O(\log n) \quad (62)$$

In other words, when the maintenance process is considered *in toto* (across multiple insertions), the performance overhead generated by deletion is completely eroded.

#### 6.4.1.2.4 Summary of Run-Time Complexity

As evidenced in Table 9, the Mak\_Tree provides superior time complexities to those previously discussed in the literature. The Dynamic Range Tree is generally more expensive due to its two level structure — though the optimisation of the approach or the application of one-dimensional range trees rests as an interesting area of future work. The Dominated Tree structures proposed by Fieldsend *et al.* [97] provide similar outcomes to the Mak\_Tree, but only under the assumption that constituents contribute to a very small number of points and that composite cleaning is infrequent. Since composite cleaning requires the successive deletion of all constituents from all composite points they contribute to (excluding the least dominating node) or the complete rebuilding of the tree, the cost of such a procedure is prohibitive in all but sparse usage (which, again, cannot be guaranteed in general).

**Table 9 — Big-Oh Space and Performance Complexity for Examined Data Structures**

† Assumes constituents do not contribute to a large number of composites. ‡ Can be reduced to  $O(\log n)$  for insertion of dominated solutions. ♦ Assumes infrequent composite cleaning. ♣ Can be improved as  $\eta$  increases. ♥ Amortised time removes the  $\eta$  factor.

Approach	Spatial Complexity	Inserting Non-dominating Solutions	Searching for Dominated Solutions	Deleting Dominated Solutions
<b>Mak_Tree</b>	$O(n)$	$O(\log n)$	$O(\log n)$	$O(\eta \log n)^{\clubsuit\heartsuit}$
<b>Dominated Tree</b>	$O(n)^\dagger$	$O(\log n + c)^{\ddagger\diamond}$	$O(\log n + \mu)^\diamond$	$O((u+\eta) \log n)^{\clubsuit\heartsuit}$
<b>Dynamic Range Tree</b>	$O(n \log n)$	$O(\log n \log(\log n))$	$O(\log n \log(\log n) + \eta)$	$O(\eta \log n \log(\log n))^\heartsuit$

## 6.4.2 EMPIRICAL PERFORMANCE

While theoretical analysis of algorithms provides an important grounding for the understanding of performance — particularly with respect to worst-case bounds — empirical examinations can elucidate behaviour under more realistic conditions. As such, this section investigates the performance of the Mak\_Tree across a diverse set of problems against both contemporary and common techniques.

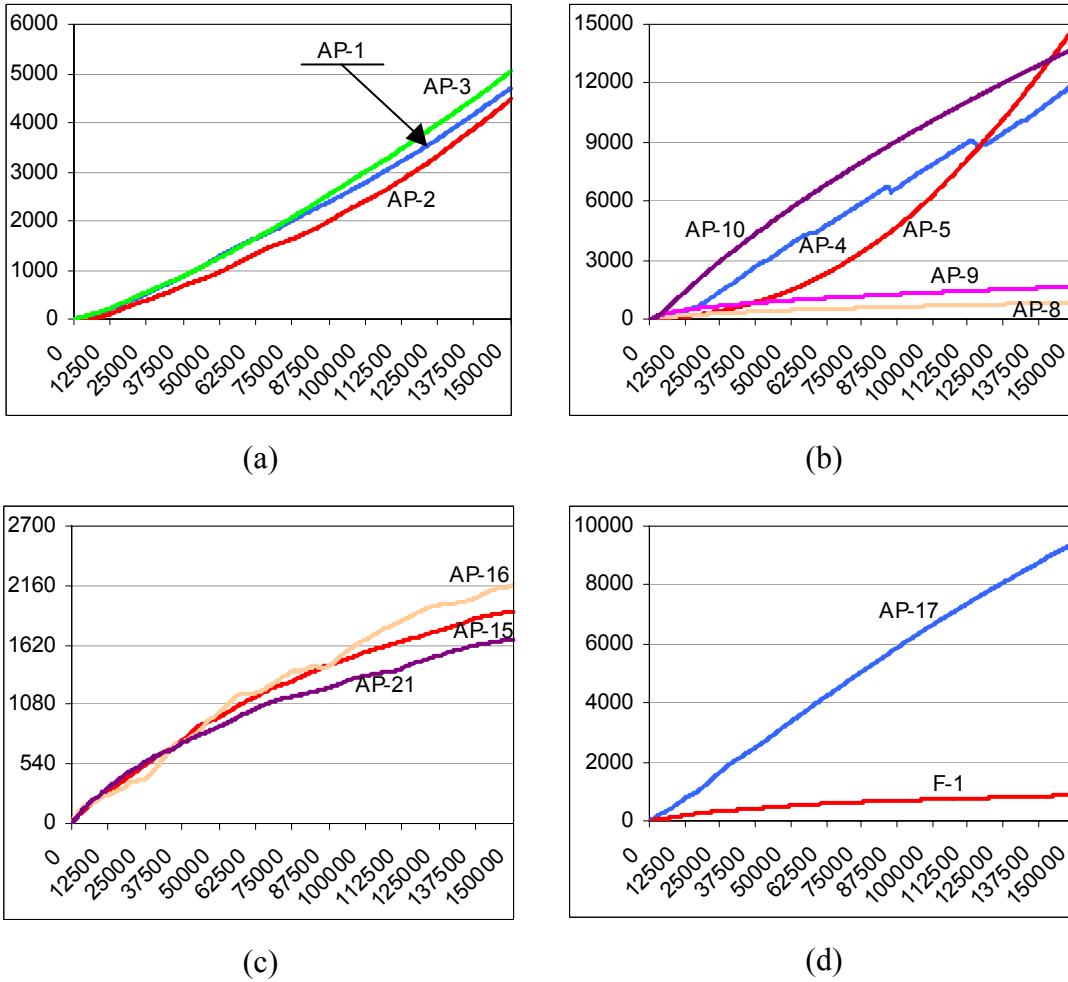
### 6.4.2.1 THE TEST PROBLEMS

As described in Section 5.3, the *AP* test suite provides an excellent base for elucidating the performance of optimisation algorithms under the presence of a diverse set of interesting and potentially challenging problem features. Many of these features are also important for understanding the general behaviour of unbounded elite stores — the shape of fronts, multi-frontal regions and the presence of constraints may all affect the performance of a given data structure or algorithm (some algorithms may better handle convex non-dominated fronts, for example). Of more explicit importance to the performance of unbounded archiving techniques however are the size and growth characteristics of the non-dominated set.

The effect of archival size should be obvious: the larger the unbounded set, the greater the potential number of solutions that an incoming proposal must be validated against. Any inefficiencies in elite archiving will be more clearly elucidated in large unbounded sets, while smaller archives may highlight thresholds for which the overhead of sophisticated data structures outweighs the apparent benefits (if such a threshold exists). Similarly, the performance of archiving strategies against differing

growth characteristics — such as variations in the acceptance/rejection rate trends and the number of archived solutions that become dominated over time — provides insight into the efficiency of update procedures and dominance verifications.

It is useful then that, when optimised by a standard NSGA-II algorithm (with settings specified in Appendix B.1.1), the static noise-free two-dimensional *AP* functions produce sets that have diverse membership numbers and varied growth characteristics. These factors are illustrated and discussed in Figure 44, which displays the average number of unique<sup>31</sup> members in the prevailing non-dominated front over twenty distinct NSGA-II runs; Table 10, which analyses some points of



**Figure 44 — Elite Unbounded Archive Sizes**

For all graphs, the *x-axis* represents the number of evaluations the NSGA-II algorithm has performed and the *y-axis* reflects the average size of the NSGA-II-produced non-dominated front.

<sup>31</sup> Recall that any two non-dominated solutions sharing result vectors are considered equivalent in this work, regardless of the variables that compose each solution.

Table 10 — Unbounded Elite Set Characteristics

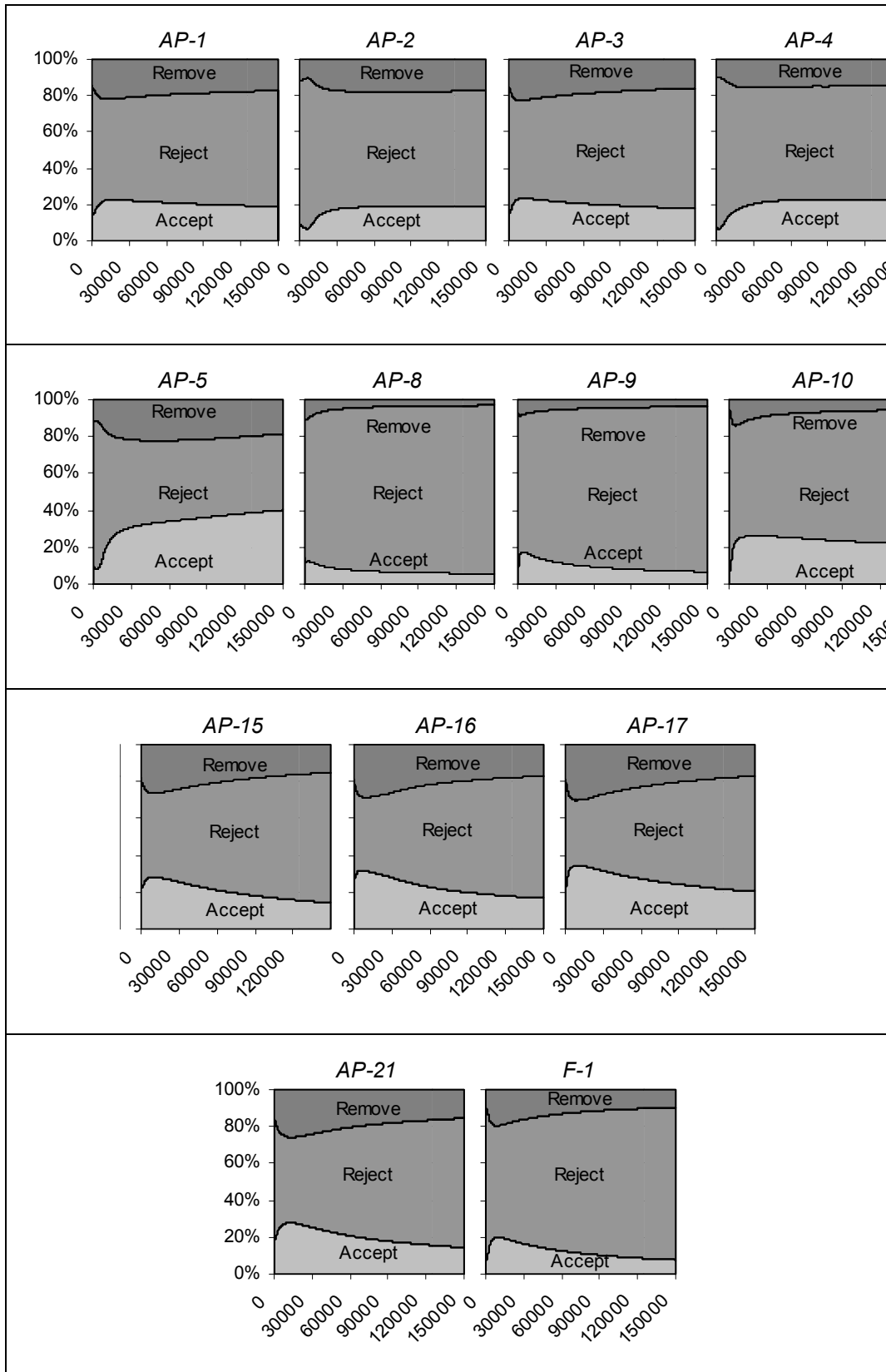
Problem	Final Size	Comments
<i>AP-1</i>	4705	Discovery of the optimal front is rapid. Subsequent evaluations focus on diversification of the front, leading to low removal:acceptance ratios (few late-generation solutions are highly dominating).
<i>AP-2</i>	4472	As above.
<i>AP-3</i>	5047	As above.
<i>AP-4</i>	11973	Discovery of improved fronts leads to step-like growth rates (the problem is multi-frontal).
<i>AP-5</i>	14829	Initially small set sizes grow rapidly until completion of run. Figure 45 shows that the ratio of acceptance:removal is increasing over time.
<i>AP-8</i>	844	High rejection levels due to constraint violations.
<i>AP-9</i>	1651	High rejection levels due to constraint violations.
<i>AP-10</i>	13788	Considerably fewer constraint-related rejections than in <i>AP-8</i> and <i>AP-9</i> .
<i>AP-15</i>	1930	Movement to optimal space, and early diversification in that space, is rapid, but subsequent improvement is both difficult and slow.
<i>AP-16</i>	2166	As above.
<i>AP-17</i>	9429	As above.
<i>AP-21</i>	1671	As above.
<i>F-1</i>	863	Low final set size is principally due to large rejection rates (generating better proposals is difficult) and relatively high removal rates (when a better proposal is generated, it tends to be highly dominating).

note; and Figure 45, which illustrates the ratios of solution acceptance, rejection and removal (archival deletion due to dominance) across the same test set.

Note that the *AP* functions alone do not thoroughly test consistently small non-dominated fronts except in two of the constraint-based problems (*AP-8* and *AP-9*); to ensure that this case is suitably examined, the *F-1* test function proposed in Fieldsend *et al* [97] is included as an additional problem. The *F-1* function describes a large objective-space with a large continuous, and convex, Pareto optimal front. The size of the elite set remains low in these experiments as forward progression through the substantial objective-space is favoured over diversification when optimising with the NSGA-II algorithm (as discussed in Table 10).

#### 6.4.2.2 THE IMPLEMENTATION OF ALTERNATIVE ELITE STORAGE TECHNIQUES

To place the performance of the Mak\_Tree algorithm in context, it is compared to the conventional unbounded linear list, a Dominated Tree and truncated (bounded) linear lists on the proposed test functions. The specifics of each implementation are worth further discussion as they will invariably affect overall performance.



**Figure 45 — Acceptance, Rejection and Removal Ratios**

Each graph reflects the ratios of acceptance, rejection and removal across 150,000 evaluations.

#### 6.4.2.2.1 Unbounded Linear Lists

The naïve list is conceptually very simple: an incoming solution is compared to each member of the non-dominated list — if the incoming solution is dominated, comparisons cease and the solution is rejected; if the solution dominates a pre-existing member, that member is removed; if it is equivalent to an existing member, it is added to that node. For the empirical analysis detailed in this section, member-nodes are stored in a singly linked list arbitrarily ordered by performance on the first objective (such ordering carries no additional operational overhead).

#### 6.4.2.2.2 Dominated Trees

The Dominated Tree structures are implemented as per the guidelines in Fieldsend *et al.* [97]. Note that existing implementations (such as that which is referenced in the original work [97]) use linked list structures for the storage of all composite points and PQRS cells. This is an expensive variation from the directions provided in the paper since any search within a linked list is bounded by  $O(n)$ , as opposed to  $O(\log n)$  for a balanced binary tree structure. The effect is that a linked-list approach to the Dominated Tree carries a worst-case complexity that is *more* expensive than the naïve list (though the practical reality is that the Dominated Tree approach, even in this malformed guise, would still likely be preferable due to the beneficial ordering of composites). It is also worth noting that many of the results reported in Fieldsend *et al.* [97] imply a similar misuse of linked lists, as the deviation between timing results for the naïve approach and the Dominated Tree technique are generally small (particularly relative to the differences seen later in this work — see Section 6.4.2.3). To avoid such problems, the Dominated Tree structures are all built upon Red-Black Trees in this work — a consequence of which is that the results reported herein may be the first to accurately describe the performance of Dominated Trees with appropriate underlying data structures.

With respect to cleaning, Non-dominated and Dominated Trees are re-built from PQRS cells whenever the number of composites exceeds 60% of the size of the archive (as-per the approach used in implementations provided by Fieldsend [198]). Assuming a PQRS Tree of size  $n$ , a Dominated or Non-dominated Tree is cleaned in  $O(n \log n)$  time using this method.

#### 6.4.2.2.3 Truncated Archives

The truncated archive builds upon the naïve list, such that an upper threshold on membership numbers is maintained. Once a threshold breach occurs, a density estimate for each solution is calculated and members deemed to be in the most densely populated region of objective-space are removed. For this work, two types of truncated archival maintenance are examined: strict and batch. In the case of strict maintenance, solutions are incrementally inserted into the archival set until the number of members exceeds the threshold and truncation occurs. In batch processing, a collection of solutions is inserted into the archive and truncation may only occur at the completion of this procedure. The two techniques emulate the behaviour typical of single-member and population-based optimisation techniques respectively and should therefore act as a good guide to practical truncated archive performance.

With both truncation approaches, the density estimate ( $\delta$ ) of any solution ( $\mathbf{a}$ ) is:

$$\delta = \frac{\text{euclidean\_dist}(\mathbf{a}, \bar{\mathbf{a}}) + \text{euclidean\_dist}(\mathbf{a}, \vec{\mathbf{a}})}{2} \quad (63)$$

where  $\bar{\mathbf{a}}$  is the in-order predecessor of  $\mathbf{a}$  and  $\vec{\mathbf{a}}$  is the in-order successor. If no predecessor or no successor exists, the Euclidean distance is infinite (to bias the inclusion of extreme members).

Using this simple method, density estimation requires only  $O(n)$  comparisons<sup>32</sup> and the maintenance process as a whole will require only  $O(n + \text{breach})$  operations (where *breach* is the size of the threshold breach). Such performance is optimal for any list-based truncated archive designed for operation in generic multiobjective optimisation.

In this work, truncation levels of 50 and 100 are used, with batch processing approaches using equivalent batch sizes. Again, these settings are consistent with those commonly seen throughout the literature (as demonstrated in [21, 74, 81, 82, 92, 182-185]) and should provide an accurate representation of typical bounded archive performance.

---

<sup>32</sup> Note that if batch processing results in  $s$  solutions surpassing the threshold, a single pass through the list is still all that is required — the  $s$  most crowded solutions are derived entirely from estimates formed on the first pass.



### 6.4.2.3 *EMPIRICAL RESULTS AND DISCUSSION*

All reported results (see pages 139–141) illustrate the average cumulative processing times of each archiving approach across twenty solution sets per problem. Each solution set is produced by a distinct NSGA-II run (again, for parameter settings, see Appendix B.1.1), with timing results excluding the costs incurred during this optimisation process. It is worth noting that any solution that has violated a constraint is immediately denied inclusion in any elite archival store — this is a simplification that may not be appropriate for all practical uses, where a secondary data structure may need to be employed for the storage of invalid proposals. Such extension lies beyond the scope of this work, but rests as an interesting avenue of future work.

#### 6.4.2.3.1 *The Mak\_Tree and Unbounded Archiving Techniques*

As evidenced in Figure 46, Figure 47, Figure 48 and Table 11, the Mak\_Tree outperforms the naïve list approach on every examined problem. The need to traverse typically large portions of the set on insertion (or indeed the entire set, if the incoming proposal is non-dominated) results in exponential cumulative time costs — even when addressing sets with high rejection rates, where a smaller number of incoming non-dominated solutions should be expected. In sharp contrast, the Mak\_Tree maintains a near-linear growth-rate on every tested function. Moreover, Table 11 illustrates that the Mak\_Tree is at least competitive with, and generally faster than, the list even after a small number of insertions<sup>33</sup>. This result runs counter to the rather intuitive theory that the maintenance and use of sophisticated data structures is inappropriate at small population levels and for brief runs [96, 97] where “the time cost of maintaining the data structures [outweigh] the search cost reduction” [97]. While such a statement is largely shown to hold true for the relatively complex Dominated Tree technique (Table 11), the simplicity of the Mak\_Tree data structure and the efficiency of the Mak\_Tree algorithms diminish the performance trade-off that once existed between brief and protracted runs.

When compared with the contemporary Dominated Tree approach, the Mak\_Tree is faster on every tested problem. Like the unbounded list, the Dominated Tree tends to

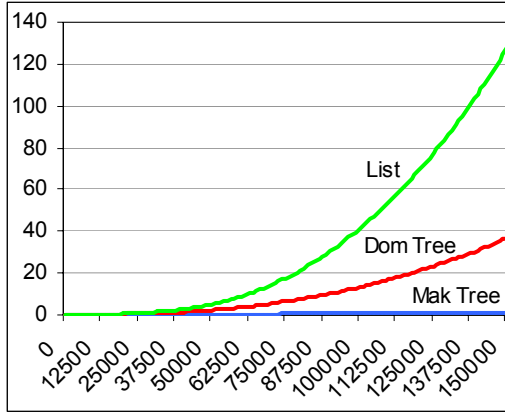
---

<sup>33</sup> The Mak\_Tree is slower early in the run only on *AP-2* and this is principally because of extremely small non-dominated set sizes during the initial phase of optimisation (the average set size over the first 5,000 evaluations is 6; over the first 10,000 evaluations the average size is 15).

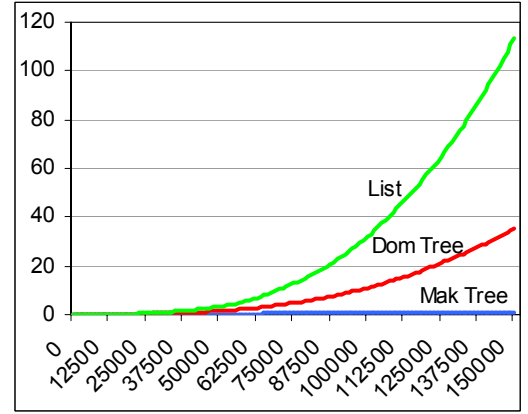
have exponential growth in time costs — in this case principally due to the inefficiency inherent in constituent verifications and, more significantly, cleaning. It is for this reason that the performance of Dominated Trees vary so noticeably with the size of the set being stored — as  $n$  grows, the  $O(n \log n)$  cleaning operation carries a sharply increasing burden<sup>34</sup>. By avoiding the need for re-building, the Mak\_Tree avoids wild fluctuations in performance and is consistent across inputs. To illustrate the point, consider the total cumulative time costs of the Mak\_Tree and the Dominated Tree on problems *AP-5* (final set size: 14829) and *F-1* (final set size: 862): the Mak\_Tree is 1.73 times slower on the larger set than on the small; the Dominated Tree is a remarkable 101.18 times slower. The difference between good performance and bad performance is *huge*. Given that the size of a non-dominated set will rarely, if ever, be known in advance, such variations are unacceptable in practice.

---

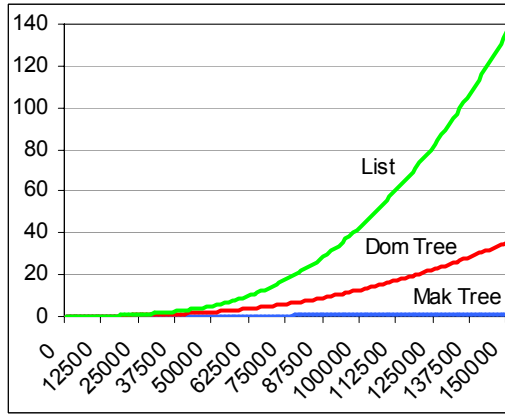
<sup>34</sup> It is tempting to think that abandoning cleaning altogether may be preferable in this case. Unfortunately, the consequence of such an action is that the costs incurred by increased composite searching and updating exceed those of cleaning by a large margin. See Appendix C.1 for a collection of results that illustrate the point.



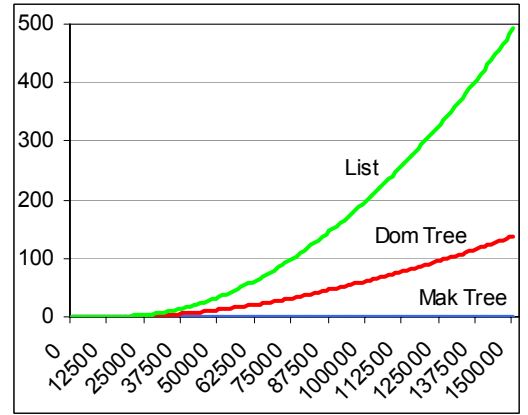
(a) AP-1



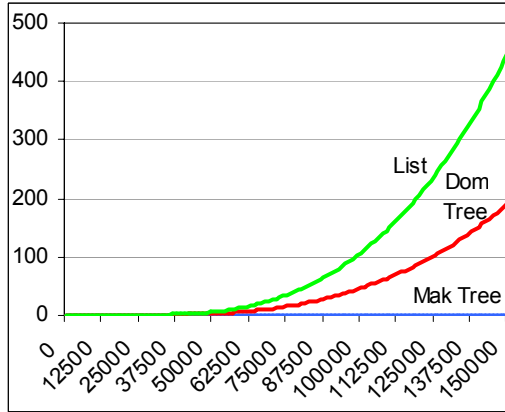
(b) AP-2



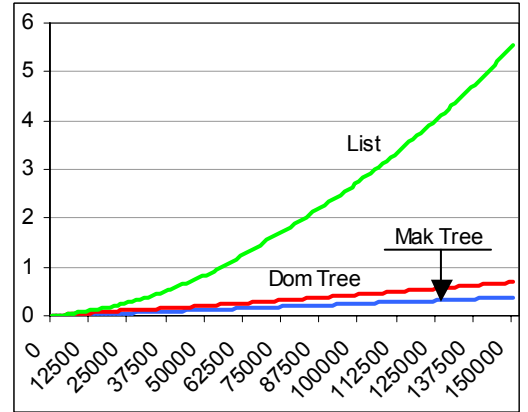
(c) AP-3



(d) AP-4

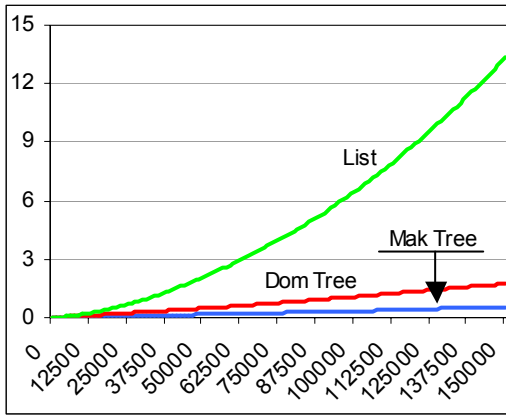


(e) AP-5

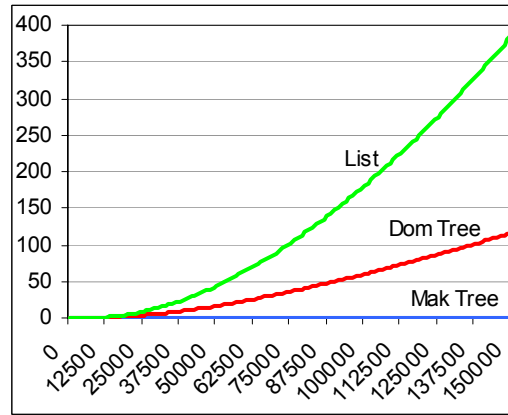


(f) AP-8

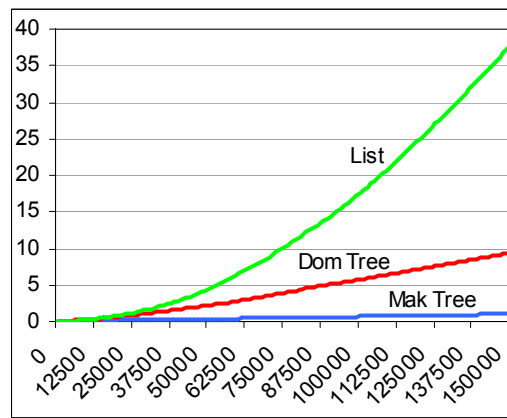
**Figure 46 — Average Cumulative Time Costs of Differing Unbounded Archiving Techniques**  
 For all graphs, the *x-axis* represents the number of solutions presented to the archive and the *y-axis* is the average cumulative processing time in seconds



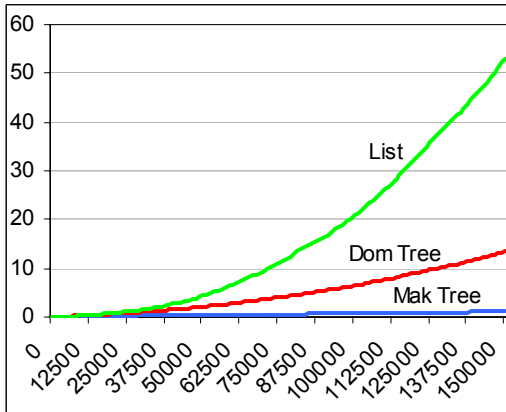
(a) *AP-9*



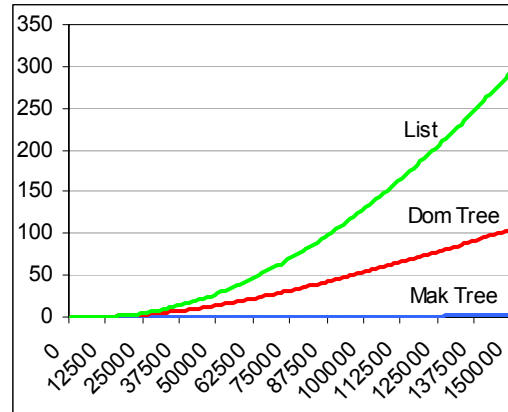
(b) *AP-10*



(d) *AP-15*

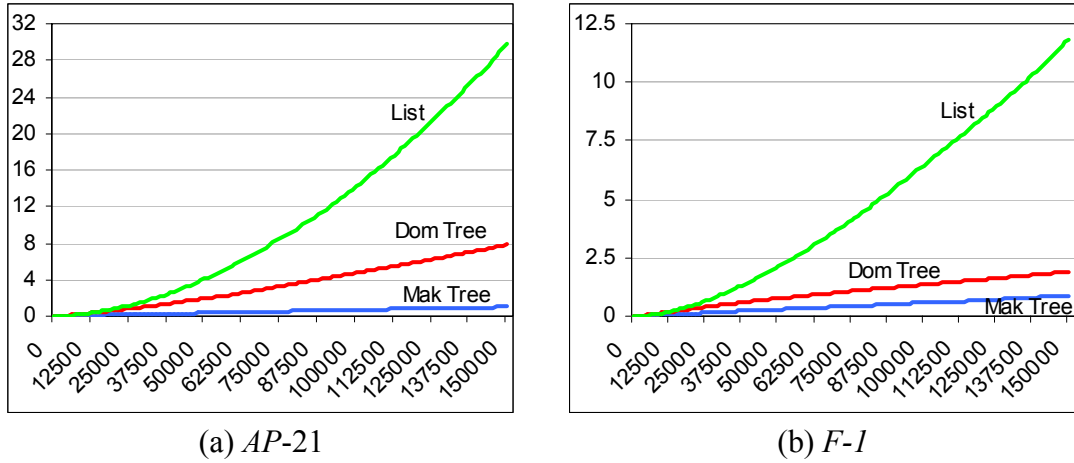


(e) *AP-16*



(f) *AP-17*

**Figure 47 — Average Cumulative Time Costs of Differing Unbounded Archiving Techniques**  
For all graphs, the *x-axis* represents the number of solutions presented to the archive and the *y-axis* is the average cumulative processing time in seconds



**Figure 48 — Average Cumulative Time Costs of Differing Unbounded Archiving Techniques**  
 For all graphs, the *x-axis* represents the number of solutions presented to the archive and the *y-axis* is the average cumulative processing time in seconds

**Table 11 — Average Total Cumulative Times for Elite Archiving Techniques**

<b>Problem</b> (Set Sizes After 10k/50k/150k Evals.)	<b>10,000 Evaluations</b>			<b>50,000 Evaluations</b>			<b>150,000 Evaluations</b>		
	<b>Mak</b>	<b>Dom</b>	<b>List</b>	<b>Mak</b>	<b>Dom</b>	<b>List</b>	<b>Mak</b>	<b>Dom</b>	<b>List</b>
<b>AP-1</b> (150/1283/4705)	0.06	0.19	0.10	0.30	4.78	4.44	1.17	36.66	127.21
<b>AP-2</b> (55/958/4472)	0.06	0.06	0.03	0.30	3.66	3.15	1.18	35.66	113.75
<b>AP-3</b> (173/1253/5047)	0.04	0.21	0.14	0.28	5.24	5.25	1.13	35.66	136.74
<b>AP-4</b> (316/3831/11973)	0.06	0.17	0.16	0.33	30.76	36.72	1.20	138.67	491.98
<b>AP-5</b> (97/1481/14829)	0.06	0.13	0.06	0.33	9.99	7.62	1.49	191.98	449.08
<b>AP-8</b> (161/459/844)	0.03	0.06	0.08	0.13	0.37	0.95	0.37	0.69	5.54
<b>AP-9</b> (406/957/1651)	0.04	0.16	0.16	0.18	1.30	2.43	0.52	1.75	13.31
<b>AP-10</b> (1075/5652/13788)	0.05	0.94	0.59	0.38	45.52	56.99	1.01	115.89	385.13
<b>AP-15</b> (242/967/1930)	0.06	0.36	0.24	0.40	6.27	6.22	1.02	9.41	37.87
<b>AP-16</b> (245/1008/2166)	0.08	0.40	0.26	0.43	6.09	6.76	1.15	13.37	53.01
<b>AP-17</b> (587/3376/9429)	0.07	0.89	0.45	0.54	49.42	45.13	1.42	105.42	295.19
<b>AP-21</b> (275/881/9429)	0.06	0.38	0.24	0.45	6.01	6.17	1.03	7.81	29.77
<b>F-1</b> (146/518/1671)	0.06	0.16	0.12	0.30	4.78	4.44	0.86	1.90	11.84

### *The Mak\_Tree and Truncated Archives*

Perhaps the most surprising, and certainly the most significant, result however is the relative performance of the Mak\_Tree against the truncated archives (see Figure 49, Figure 50, Figure 51, Table 12). On every test problem the Mak\_Tree is not just competitive with the size-limited archiving techniques, but *markedly* better. The reasoning is less obtuse than may be imagined.

Firstly, consider the number of operations required when an incoming solution is non-dominating. In the Mak\_Tree, the cost is  $O(\log n)$ , while in a truncated archive the cost is  $O(n')$  (where  $n'$  is the size of the truncated archive). For these tests, where  $n'$  is 50, there is no practical<sup>35</sup>  $n$  such that  $O(\log n) > O(n')$  and the Mak\_Tree is clearly superior.

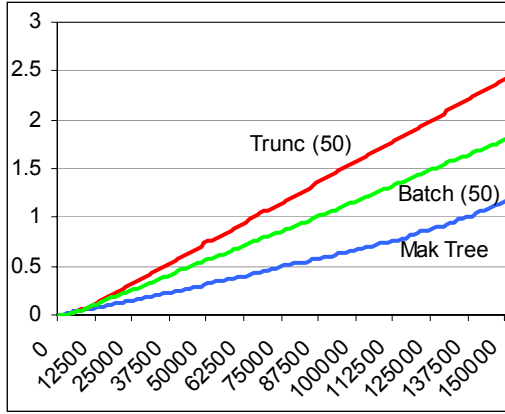
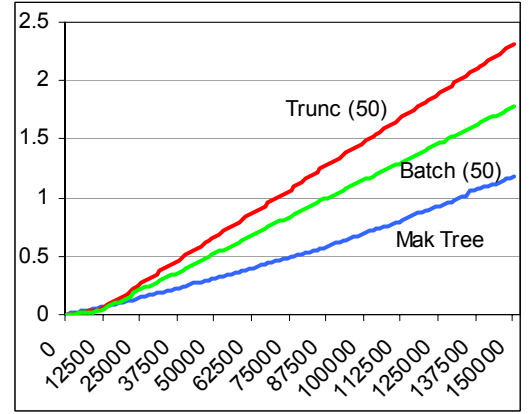
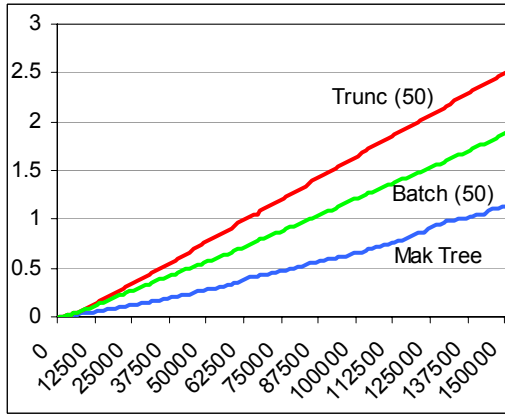
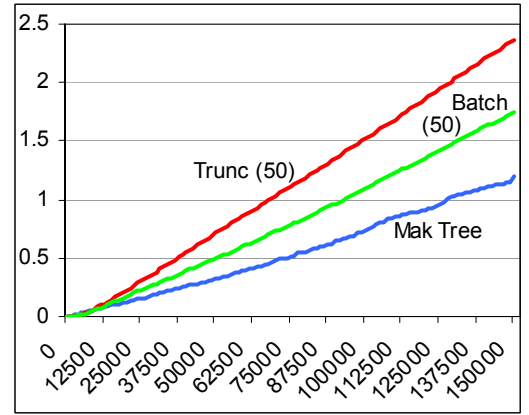
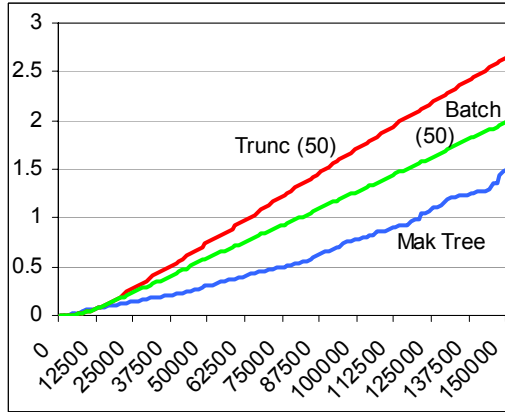
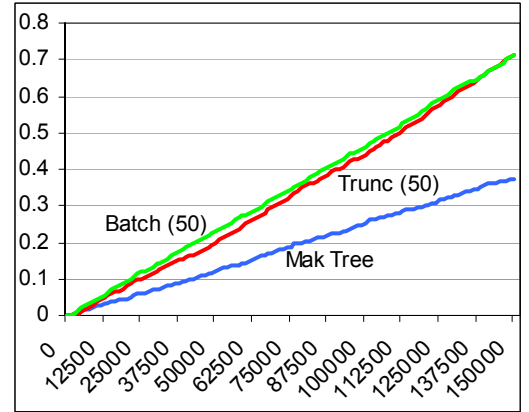
The more complicated case is when a dominating solution enters the archive. The Mak\_Tree will take  $O(\eta \log n)$  operations (though sub-tree deletion makes for large practical improvements over this when  $\eta$  is high), while the bounded archive remains at  $O(n')$ . So long as  $\eta < n' / (\log n)$  the Mak\_Tree will be at least approximately equivalent to the truncated list. For  $n' = 50$  and  $n = 500$ , for instance, there must be more than five deletions on every insertion for the Mak\_Tree to begin falling behind — this is *extremely* unlikely.

That is to say that the results indicate that a bounded archiving approach, complete with all of the disadvantages that such an approach entails (see Section 6.2), can be replaced by a truly unbounded archiving technique with practically no performance degradation whatsoever<sup>36</sup>. Beyond the difficulties in implementing the Mak\_Tree (which should be few, given its simplicity) and the increased storage requirements that unbounded archiving inevitably requires, there are few reasons to support the continued use of truncated elite sets in bi-objective optimisation.

---

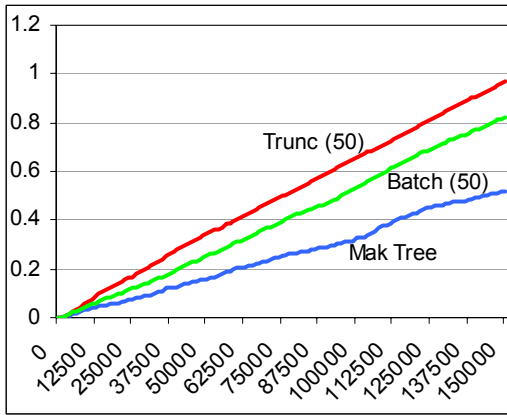
<sup>35</sup> Even an unbounded archive is unlikely to exceed more than 100,000,000.

<sup>36</sup> The truncated approach is marginally better suited to very small elite sets (particularly those with sizes that are consistently lower than a truncation threshold of fifty). The existence of such sets in practice however, particularly for sustained runs, is unlikely.

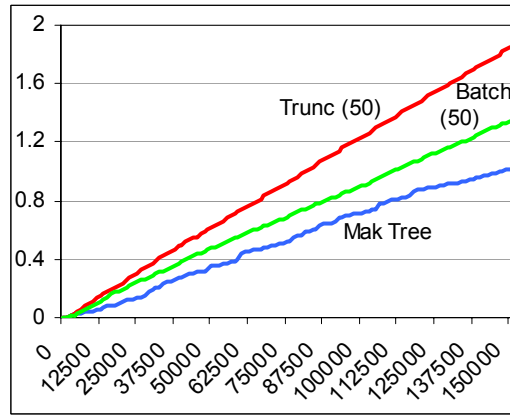

 (a) *AP-1*

 (b) *AP-2*

 (c) *AP-3*

 (d) *AP-4*

 (e) *AP-5*

 (f) *AP-8*

**Figure 49 — Average Cumulative Time Costs of Unbounded and Bounded Archiving Techniques**

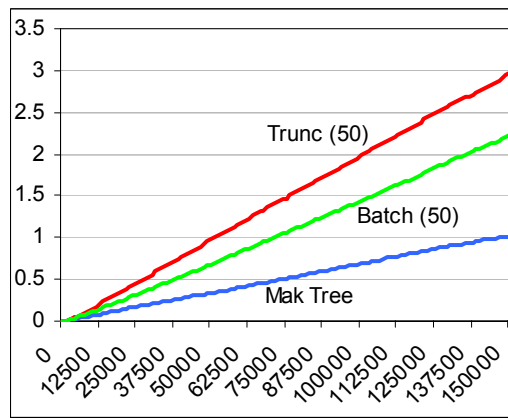
For all graphs, the *x-axis* represents the number of solutions presented to the archive and the *y-axis* is the average cumulative processing time in seconds



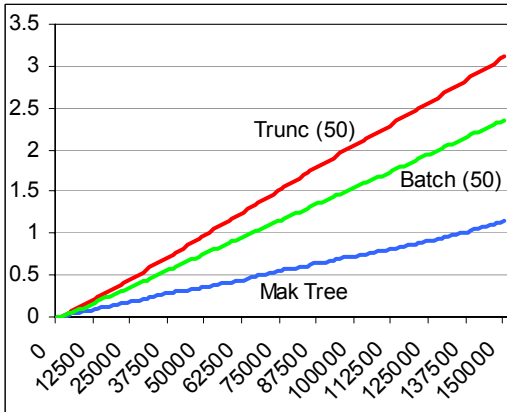
(a) *AP-9*



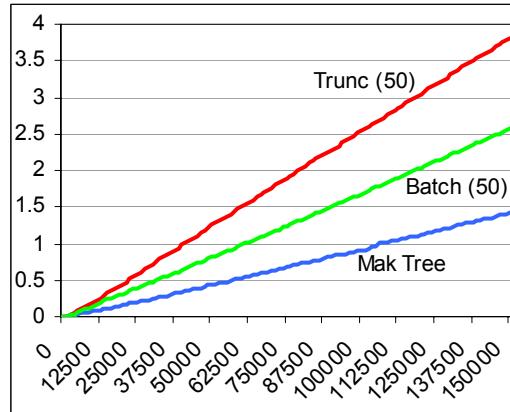
(b) *AP-10*



(d) *AP-15*



(e) *AP-16*

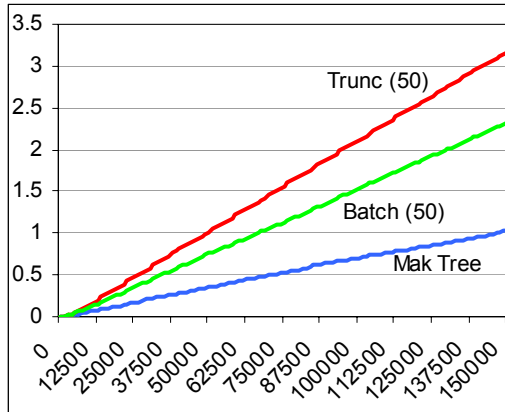
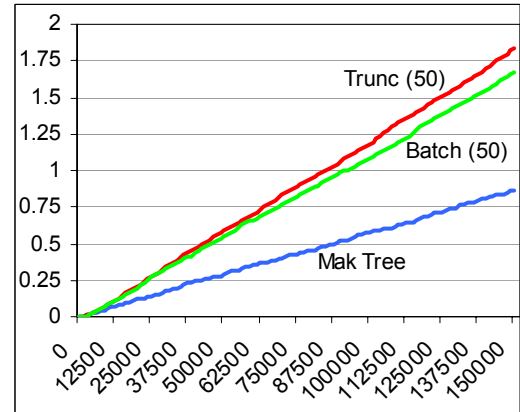


(f) *AP-17*

**Figure 50 — Average Cumulative Time Costs of Unbounded and Bounded Archiving Techniques**

For all graphs, the *x-axis* represents the number of solutions presented to the archive and the *y-axis* is the average cumulative processing time in seconds




 (a) *AP-21*

 (b) *F-1*

**Figure 51 –Average Cumulative Time Costs of Differing Unbounded Archiving Techniques**  
For all graphs, the *x-axis* represents the number of solutions presented to the archive and the *y-axis* is the average cumulative processing time in seconds

**Table 12 — Average Total Cumulative Times for Elite Archiving Techniques**

	10,000 Evaluations					50,000 Evaluations				
	Mak	Batch (50)	Trunc (50)	Batch (100)	Trunc (100)	Mak	Batch (50)	Trunc (50)	Batch (100)	Trunc (100)
<b>AP-1</b>	0.06	0.08	0.09	0.09	0.10	0.30	0.54	0.71	0.90	1.19
<b>AP-2</b>	0.06	0.03	0.04	0.03	0.04	0.30	0.51	0.65	0.80	1.01
<b>AP-3</b>	0.04	0.09	0.10	0.14	0.13	0.28	0.57	0.78	0.99	1.22
<b>AP-4</b>	0.06	0.06	0.08	0.08	0.10	0.33	0.51	0.74	0.93	1.19
<b>AP-5</b>	0.06	0.05	0.05	0.06	0.05	0.33	0.63	0.80	1.06	1.38
<b>AP-8</b>	0.03	0.04	0.04	0.06	0.07	0.13	0.25	0.22	0.42	0.45
<b>AP-9</b>	0.04	0.05	0.06	0.08	0.09	0.18	0.28	0.37	0.52	0.57
<b>AP-10</b>	0.05	0.08	0.11	0.12	0.18	0.38	0.53	0.69	0.83	1.22
<b>AP-15</b>	0.06	0.11	0.15	0.18	0.18	0.40	0.82	1.16	1.37	1.72
<b>AP-16</b>	0.08	0.13	0.16	0.19	0.23	0.43	0.92	1.21	1.47	1.85
<b>AP-17</b>	0.07	0.14	0.19	0.24	0.31	0.54	1.01	1.54	1.82	2.57
<b>AP-21</b>	0.06	0.12	0.15	0.18	0.20	0.45	0.94	1.29	1.55	1.77
<b>F-1</b>	0.06	0.08	0.08	0.11	0.12	0.30	0.54	0.71	0.90	1.19

**Table 13 — Average Total Cumulative Times for Elite Archiving Techniques After 150,000 Evaluations**

	Mak	Batch (50)	Trunc (50)	Batch (100)	Trunc (100)		Mak	Batch (50)	Trunc (50)	Batch (100)	Trunc (100)
<b>AP-1</b>	1.17	1.81	2.43	3.13	4.30	<b>AP-10</b>	1.01	1.34	1.85	2.19	3.16
<b>AP-2</b>	1.18	1.78	2.31	2.96	3.88	<b>AP-15</b>	1.02	2.22	2.97	3.63	4.62
<b>AP-3</b>	1.13	1.89	2.51	3.14	4.13	<b>AP-16</b>	1.15	2.36	3.11	3.91	4.88
<b>AP-4</b>	1.20	1.75	2.37	3.16	4.31	<b>AP-17</b>	1.42	2.57	3.82	4.60	6.61
<b>AP-5</b>	1.49	1.99	2.65	3.48	5.00	<b>AP-21</b>	1.03	2.33	3.17	3.83	4.74
<b>AP-8</b>	0.37	0.71	0.71	1.18	1.29	<b>F-1</b>	0.86	1.68	1.83	2.78	2.83
<b>AP-9</b>	0.52	0.82	0.97	1.40	1.66						

## **6.5 CONCLUSIONS AND DISCUSSIONS**

Capitalising on the unique properties of bi-objective optimisation that were investigated in Section 6.1, the `Mak_Tree` represents a simple and efficient approach to elite unbounded archiving. A rich set of theoretical and practical results illustrate the superiority of the `Mak_Tree` over pre-existing approaches to unbounded archiving when applied to bi-objective domains, while also highlighting improvement over simple list-based truncation. Indeed, the end-of-run performance of the completely unbounded `Mak_Tree` was better than every examined approach on all thirteen test functions. Thus, given the disadvantages inherent in traditional truncation approaches — such as receding or oscillating fronts, the loss of expensive solutions, poor crowding estimates and stopping condition complications — the `Mak_Tree` offers an exciting unbounded alternative that is largely free from the performance overhead and complexity burden that typically accompanies sophisticated archiving techniques.

It is also worth noting that even those (typically first-generation) optimisers that do not explicitly integrate an elite archival set into the evolutionary process can benefit from the performance advantages offered by the `Mak_Tree` as an offline accumulator of the best solutions found thus far. As noted by Coello, Veldhuizen and Lamont [199] (supporting a similar statement by Horn [4]) “any practical MOEA (Multi-Objective Evolutionary Algorithm) implementation must include a secondary population composed of all Pareto optimal solutions found...” since the stochastic nature of the optimisation process may result in the loss of impressive proposals. Given this requirement, the efficiency increases seen in the application of the `Mak_Tree` are of benefit to most any existing evolutionary multiobjective optimiser operating in a bi-objective domain.

# Chapter



## Harnessing the Mak\_Tree

*“If you want a happy ending, that depends, of course, on where you stop your story.”*

*Orson Welles — Film Maker*

*“You are terminated.”*

*Arnold Schwarzenegger —  
Politician and Actor  
(Terminator 3)*

## **7 HARNESSING THE MAK TREE**

With the foundations of the Mak\_Tree now in place, it is interesting to explore how a complete archival set can be used effectively. While later sections (Chapters 8–12) will investigate how unbounded archiving can be used to shape the evolutionary process, the basic Mak\_Tree described in Section 6.3, which does not address efficient density estimation or crowding-based selections, is best used as an off-line<sup>37</sup> storage system. In this capacity, the Mak\_Tree can be used in the development of powerful real-time stopping criteria and as a base from which accurate end-of-run presentational sets are formed.

### **7.1 STOPPING CONDITIONS**

Though Section 6.2.1.4 decried the lack of sophisticated stopping conditions for contemporary multiobjective optimisers, their sparsity is not without some justification. Indeed, deriving an appropriate set of terminating criteria is an innately difficult proposition.

Firstly, the multiobjective optimisation task itself is a multiobjective problem, with goals of producing a suitably distributed, accurate and well-spread front. As such, any terminating condition must be able to effectively summarise performance on these distinct tasks — a typically difficult exercise given that little is generally known about the Pareto optimal front that the optimiser is attempting to estimate. How can accuracy be assessed, for instance, if it is not known where or what the target is?

Secondly, in practice, successful termination of a run is less about indicating the formation of a near-perfect non-dominated set, and more about providing a *good* approximation in the shortest time possible. Many decision makers would accept a front with one less optimal point if it could be produced in half of the time. It is therefore important for a termination condition to gauge the rate of frontal improvement — but even then, how is such a rate used to effectively terminate a run? Moreover, how are these two conflicting tasks — the quest for optimality and the need for efficiency — resolved?

---

<sup>37</sup> For the purposes of this work, an off-line archive is one which is excluded from active participation in the evolutionary process — it is used principally for the storage of valuable solutions.

### 7.1.1 TERMINATION USING UNBOUNDED ARCHIVES

As discussed earlier (Section 6.2), truncated archives provide a poor basis for termination analysis due to their potential for frontal degradation [97] and inaccuracies in density estimates. By eradicating such concerns, unbounded elite archives offer a more complete picture of the prevailing optimal front and, in-turn, the behaviour of the optimiser as a whole.

It is disappointing then that no practical work has been completed that examines the performance of unbounded archives in providing useful termination scenarios to end-users. Indeed, the only widely-available research that even discusses such potential is in the work of Fieldsend *et al.* on Dominated Trees [97], where a number of rudimentary suggestions are made. In particular, they propose termination conditions based on the use of thresholds, such that:

- a front is no longer improving if a given number of generations have passed without the creation of a dominant solution;
- a front is appropriately distributed if the maximum nearest-neighbour distance is below some pre-defined threshold; and
- the extent of the front is frozen if it has not changed over some number of evaluations.

While these ideas correctly grasp the notions of front improvement, distribution and spread, they are potentially misleading and difficult to use in practice. Consider frontal improvement and extent expansion: there is nothing in either of these procedures that gauges the value or regularity of beneficial insertions. Thus, infrequent fine-grained improvements may substantially extend the run-time of the optimisation task with little practical benefit. Furthermore, the distribution measurement is poor, since any isolated solution in objective-space will skew results. Indeed, a single outlier will often be enough to ensure that the front is viewed as being ill-distributed through the complete duration of the optimisation run. Moreover, procedures for defining threshold levels are undefined, there is no clear outline as to how the distinct conditions can be used together to form a single coherent termination mechanism, and there is no empirical investigation as to the

overall value of the ideas expressed<sup>38</sup>. As such, there is significant scope for improvement and refinement beyond these initial suggestions — with the end goal of producing practically useful, simple and efficient terminating conditions that can truly capitalise on unbounded elite sets.

### **7.1.2 KEY PROPERTIES OF TERMINATION CONDITIONS THAT USE UNBOUNDED ELITE SETS**

Before exploring how Mak\_Trees can be used for effective run-termination, it is important to analyse precisely what should be expected of contemporary stopping criteria in a practical context. In particular, any termination system should be easy to use, efficient, general, correct and autonomous, with analysis derived from truly unbounded elite sets.

#### **7.1.2.1 SIMPLICITY**

At the forefront of stopping-criteria design should always be simplicity. It is pivotal that end-users need not know specific details regarding the nature of the objective-space in general and the particulars of the true Pareto optimal front in order to operate the produced system. Any stopping criterion that requires such knowledge precludes effective termination in ill-defined domains and limits use to experts only.

Moreover, the end-user should be able to succinctly express their goals for a given run. Large numbers of interacting user-defined parameters can make initial set-up difficult, may obscure meaning and can lead to potentially expensive parameter tuning. As such, the parameter set should be small enough to avoid these pitfalls, yet rich enough to capture decision-maker requirements. Simplicity is essential, but it should not come at the cost of expressive power.

#### **7.1.2.2 EFFICIENCY**

Given that optimiser termination will be derived from analyses of unbounded elite sets, efficiency is a primary concern. In particular, any operation that requires frequent traversals of the entire archive will severely inhibit the overall performance of the multiobjective optimiser and will limit the applicability of the termination system in practice.

---

<sup>38</sup> It is important to note that the Fieldsend *et al.* work [97] is not focused on termination and this is not intended as a criticism of the paper as a whole. Rather, it is used as an illustration of the lack of thorough optimiser termination research.

### 7.1.2.3 GENERALITY

The proposed termination conditions must function in a wide variety of domains and must not make assumptions about the nature of the objective-space or Pareto-optimal front. That is to say, stopping criteria must never rely on the concavity or convexity of a front, an unbiased distribution of points in objective-space, or the connectedness of regions — they should provide correct estimates irrespective of domain. Again, any loss of generality here will damage the practical applicability of any proposed system.

### 7.1.2.4 CORRECTNESS

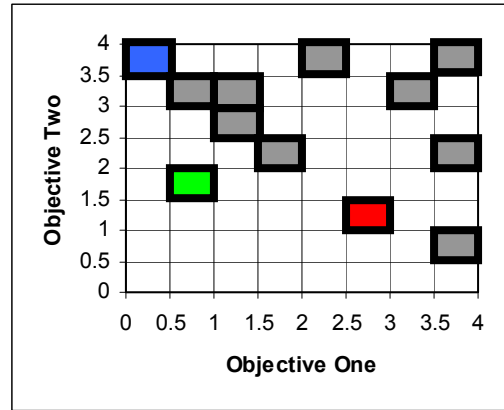
Fairly obviously, an underlying goal of all termination criteria should be correctness — the ability to appropriately terminate an optimiser in a way that satisfies the requirements of the end-user. Efficient and simple termination conditions mean little if the sets they produce are inappropriately distributed or inaccurate.

### 7.1.2.5 AUTONOMY

The termination criteria should require no (or very little) user interaction beyond the specification of initial parameters. This is a fundamentally important practical concern — increasing user interaction incurs greater monetary costs (since the optimisation run must be observed in some capacity) and limits applicability in long-run optimisations (which may require multiple days). It is also important to note that the user should never be required to interpret the front that is being produced — this is the central task of the termination system and should be fully abstracted from the user where possible.

## 7.1.3 A NEW UNBOUNDED APPROACH TO TERMINATION IN BI-OBJECTIVE DOMAINS — INTRODUCING THE MAK\_TERMINATOR

Given that the Mak\_Tree provides an impressively efficient, and potentially powerful, mechanism for maintaining an unbounded elite set of solutions, it is an appropriate foundation upon which to build a contemporary termination system that endeavours to satisfy the many properties outlined in Section 7.1.2. The new system, henceforth known as the *Mak\_Terminator*, capitalises on this foundation and introduces an additional data structure that enables the assessment of solution value relative to simply defined user requirements.



**Figure 52 — Solution Utility in Objective-Space Grid**

Gray cells are previously occupied; the incoming *green* cell results in frontal progression; the new *blue* cell increases frontal extent; the incorporation of the *red* cell improves distribution.

### 7.1.3.1 DEFINING SOLUTION UTILITY

Consider a rudimentary sub-division of objective-space according to some pre-defined resolution (as in PAES [143]). Assuming that all proposals admitted into such a grid are non-dominated with respect to all previously inserted solutions, a number of interesting properties may be observed. In particular, if the solution enters an unoccupied cell, it is either improving the distribution of the prevailing front, expanding the extent of the front, or otherwise moving the front forward (see Figure 52). Quite clearly, if the incoming solution does indeed reside in an unoccupied cell, then it is improving the value of the non-dominated front and the creation of that solution must be considered worthwhile. Conversely, any solution that resides in occupied space offers no improvement to the *cell-based projection* of that front<sup>39</sup>. The integration of these notions into simple, yet powerful, stopping criteria lies at the heart of the Mak\_Terminator.

### 7.1.3.2 MAINTAINING THE OBJECTIVE-SPACE GRID

The objective-space grid is a deliberately simple construct. It can be represented as a rudimentary two-dimensional array of cells, with each cell maintaining an occupancy flag (note that no reference to the solutions which reside in a cell is required — such information is superfluous to the operation of the Mak\_Terminator). By capitalising on the efficient Mak\_Tree for non-dominance determination, updating the grid is also straightforward. Any solution that is accepted into the Mak\_Tree is passed onto

<sup>39</sup> The italicised distinction here is key. Solutions entering occupied space may be useful to the underlying non-dominated front, but of no value to the coarse representation of that front in the grid. The value of this difference will be discussed throughout this section.



the grid, with the flag in the corresponding cell set accordingly. Determining the cell location of a solution ( $\mathbf{a}$ ) in an objective-space with a fixed number of regions per dimension is trivial, but is defined in Equation (64) for the sake of completeness:

$$\text{cell}(\mathbf{a}) = \left\lfloor x\_regions \times \frac{(f_1(\mathbf{a}) - \min\_x)}{(\max\_x - \min\_x)} \right\rfloor, \left\lfloor y\_regions \times \frac{(f_2(\mathbf{a}) - \min\_y)}{(\max\_y - \min\_y)} \right\rfloor \quad (64)$$

If the boundaries of the objective-space grid can be statically specified in advance, the  $O(1)$  cost of inserting each solution into the grid does not affect the running performance of the standard Mak\_Tree. In the more likely case that the end-user is unaware of correct boundary values, an adaptive mechanism can be used that re-sizes the grid when the extent of the prevailing non-dominated front changes (as with the PAES hyper-grid [143]). The need to re-insert every member of the Mak\_Tree into the objective-space grid under such boundary movements carries an  $O(n)$  overhead (assuming that links between successive nodes in the Mak\_Tree are maintained) that appears prohibitive. In practice however, it is unlikely that incoming proposals will regularly affect the extent of the prevailing front and so long as rebuilding occurs less frequently than once in every  $O(\log n)$  attempted insertions, the amortised time cost of the Mak\_Tree remains unchanged. However, if users are particularly concerned by the potential costs that adaptation incurs, the boundaries of the space can be made to rest some distance outside the true extent of the front, further reducing the likelihood of regular reconstructions.

An alternative procedure, that implicitly specifies the resolution of the grid, is to define the objective-based precision required by the end-user. In this case, since the maximum number of cells is unknown and the boundaries are undefined, a better supporting structure for the storage of cells is a simple balanced binary tree, ordered (arbitrarily) by the  $x$  (objective one) coordinate of the cell<sup>40</sup>. Note that calculation of coordinates in this case is also trivial — for given precision settings  $\rho\_x$  and  $\rho\_y$ , a solution  $\mathbf{a}$  belongs to the cell with coordinates defined by:

$$\text{cell}(\mathbf{a}) = \left\lfloor \frac{f_1(\mathbf{a})}{\rho\_x} \right\rfloor, \left\lfloor \frac{f_2(\mathbf{a})}{\rho\_y} \right\rfloor \quad (65)$$

<sup>40</sup> When distinct cells share an  $x$  value, the order is instead defined by the  $y$  coordinate.

The run-time performance of such a structure is impressive. Since the binary tree need only support insertions, the worst-case overhead is  $O(\log n)$  — leaving the performance of the Mak\_Tree algorithm unaffected. Moreover, unlike the array-based approach, the behaviour of the system is not sensitive to frontal expansion and therefore offers more stable performance. For these reasons, this technique is recommended over the array methodology in general, and will form the basis of subsequent analyses.

### 7.1.3.3 *MAK\_TERMINATOR PARAMETERS*

The end-user is required to select two parameters for use by the Mak\_Terminator: the maximum average *time* between the production of useful solutions and the required precision of the optimiser. The key to these parameters is that they require little detailed knowledge about the domain or objective-space and that they have real practical meanings. A user will know how long they are willing to wait for meaningful changes in the front — it matters not whether that time equates to forty-five evaluations or ten thousand, and nor should it. Similarly, setting the precision level of the terminator should be intuitive — the value is simply defined as the point at which improvements no longer bring practical benefits. As an illustration, if the loss of a dollar in a production cost objective is largely insignificant, but an improvement of \$10 is worth mention, then the precision level could reasonably be set at the \$1 mark. The point is, a non-technical end-user, with little knowledge of the peculiarities of multiobjective optimisation and the finer workings of the problem at hand, should be able to assign Mak\_Terminator parameters without moving far beyond their comfort zone.

### 7.1.3.4 *THE MAK\_TERMINATOR IN PRACTICE*

The performance  $r$  of a given optimiser at time  $t$  is related to the number of valuable solutions  $\nu$  (those which were non-dominated in the Mak\_Tree and subsequently inserted into unoccupied cells) produced during the preceding  $\varepsilon$  evaluations<sup>41</sup> such that:

$$r = \frac{\nu}{\varepsilon} \tag{66}$$

---

<sup>41</sup> This interval could be included as a separate parameter, but is best derived from the timing threshold defined by the user. A sensible setting would be to have  $\varepsilon \geq \Gamma/\chi$ , with a preliminary recommendation of  $\varepsilon = 4\Gamma/\chi$  proposed to reduce the impact of noise in  $\nu$ .

The termination of the run is based on this simple  $r$  value and occurs when:

$$\frac{\chi}{r} > \Gamma \quad (67)$$

where  $\chi$  is the average time-cost per evaluation and  $\Gamma$  is the user-defined maximum time between production of valuable solutions. Note that the user does not need to know the evaluation time of each operation in advance — this can be observed and updated throughout the run with little overhead<sup>42</sup>. Again, this reduces the need for intimate domain knowledge and allows the termination system to match user requirements, irrespective of the particular configuration of the underlying algorithm or machine.

### 7.1.3.5 EMPIRICAL RESULTS

To observe the Mak\_Terminator in a practical setting, the data sets produced by NSGA-II on the thirteen problems examined in Section 6.4.2 were assessed by the termination system under various precision settings, with Figure 53, Figure 54 and Figure 55 illustrating the average  $r$  scores from twenty distinct runs, each with  $\varepsilon = 500$  (see pages 158–160). To explore the type of sets suggested by the Mak\_Termination system, Figure 57, Figure 58 and Figure 59 (pages 161–163) display the median leading fronts for each problem at the median termination points specified in Figure 56. For this analysis, termination occurs when  $r$  is zero — indicating no practically significant change over the preceding  $\varepsilon$  evaluations.

The most obvious initial conclusion is that decreased precision leads to more rapid terminations (see Figure 53, Figure 54 and Figure 55). The reasoning should be equally obvious — since the leading front consists of fewer objective-space cells, fine-grained improvements in extent, distribution and progression are less well identified; most changes are occurring *within* previously occupied cells. So long as the precision level is appropriately specified, this is a desirable property, leading to termination when the changes are minimal with respect to their practical importance.

It is also worth noting that the progressive  $r$ -value graphs (Figure 53, Figure 54 and Figure 55) offer insight into the behavioural characteristics of the optimiser, with the NSGA-II system displaying poor initial performance in *AP-2*, rapid convergence in

<sup>42</sup> Though safeguards should exist to ensure that  $\Gamma > \chi$ .

*AP-8*, *AP-9* and, to a lesser extent, *AP-10*, and the discovery of false fronts (leading to temporal convergence) in *AP-4*. The use of `Mak_Terminator` outputs to track real-time performance of a system is therefore an interesting avenue of future work, particularly given the low complexity overhead afforded by such an approach.

It is difficult to gain a definitive insight into the nature of termination sets at differing precision levels (Figure 57, Figure 58 and Figure 59), as they are obviously subject to the peculiarities of the underlying optimiser, though a number of general comments can be made. In particular, the results indicate that precision level specification defines where an acceptable compromise between set quality and termination speed can be found (as one would reasonably expect). Consider the relatively coarse  $\rho = 0.1$  setting: in all but *AP-10*, the `Mak_Terminator` ends the run at least twice as fast as the  $\rho = 0.01$  system, but offers less impressive fronts. The fact that the inferior sets are still good approximations of the Pareto optimal front (with respect to both proximity and shape) in all but *AP-2* is a testament to the robustness and power of the proposed termination system.

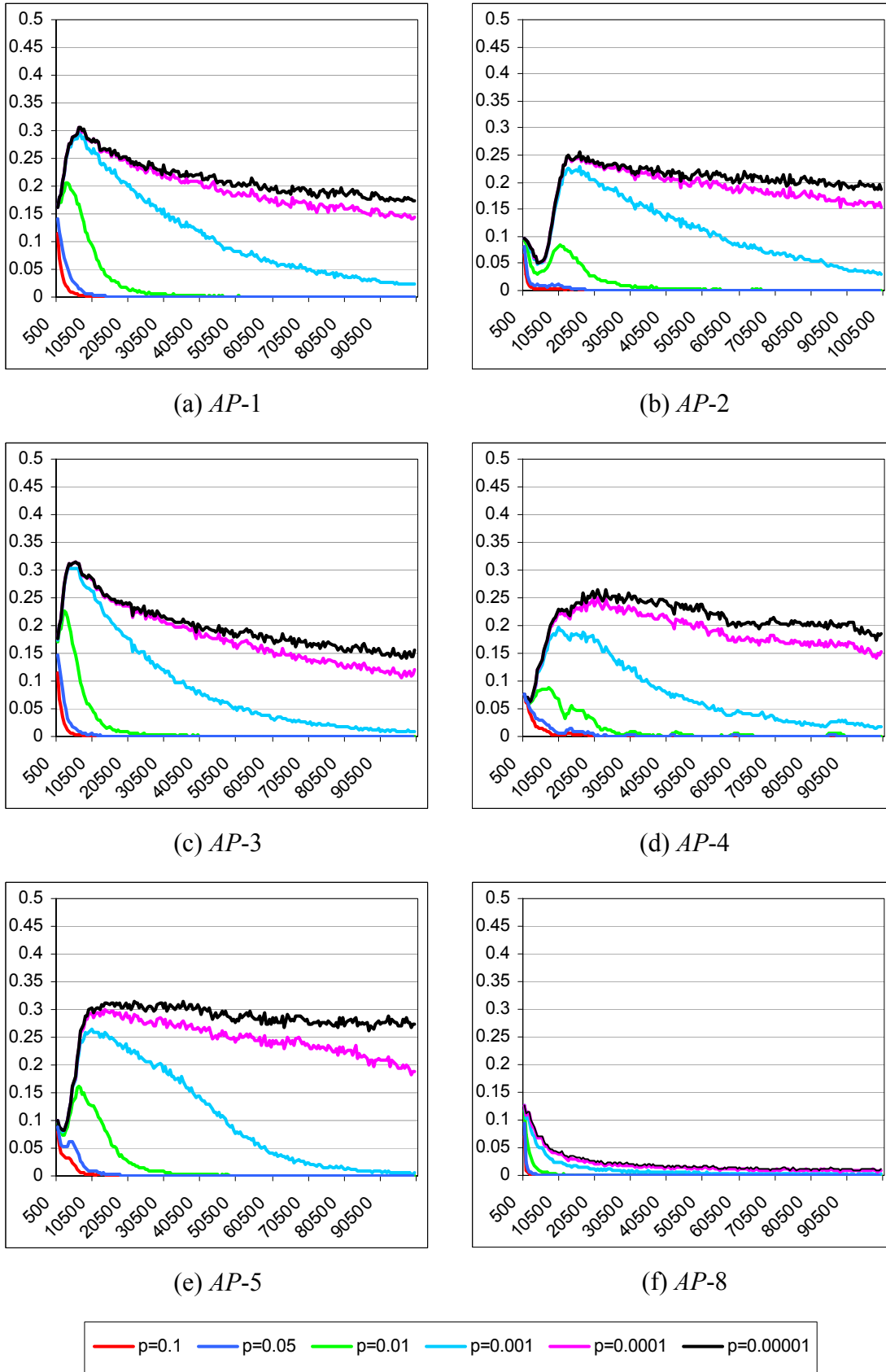
It is worth noting that the poor performance of the less precise systems on *AP-2* is related to how NSGA-II optimises the concave objective-space. As illustrated in Figure 60, NSGA-II can become fixated on an extreme of the concave front, with small levels of frontal progression strongly favoured over diversification. As the leading front nears optimal space, the progressions reduce in both frequency and magnitude (as illustrated by the declining early performance in Figure 53b), leading to system termination when the grid resolution is low. Upon converging near the extreme optimal point however, NSGA-II then quickly and effectively distributes the search across the entirety of the front, resulting in the considerably more impressive  $\rho = 0.01$  set seen in Figure 57b. The point here is an important one — any termination system is forming conclusions about the likely future performance of an optimiser based on previous evidence, but can offer no guarantees as to the accuracy of those conclusions<sup>43</sup>. While it may be true that the best predictor of future behaviour is past behaviour<sup>44</sup>, the difference between frontal stagnation and end-of-run convergence is ultimately impossible to differentiate. Thus, those looking to capitalise on intelligent stopping criteria should be aware that the results are subject

---

<sup>43</sup> Assuming that the Pareto optimal set is not known in advance.

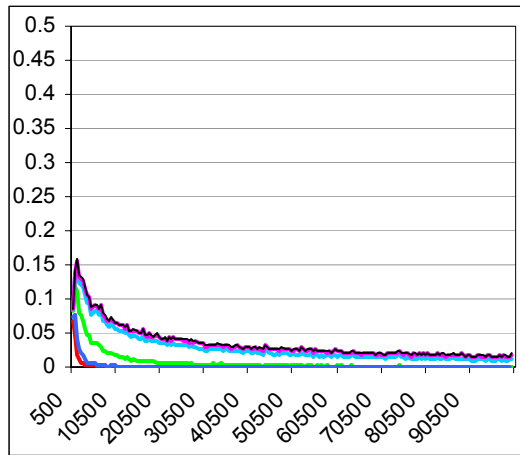
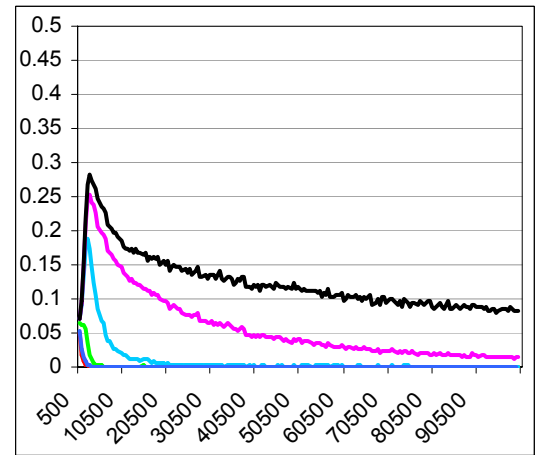
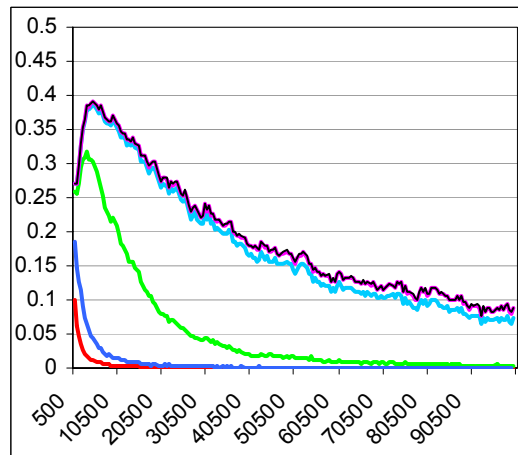
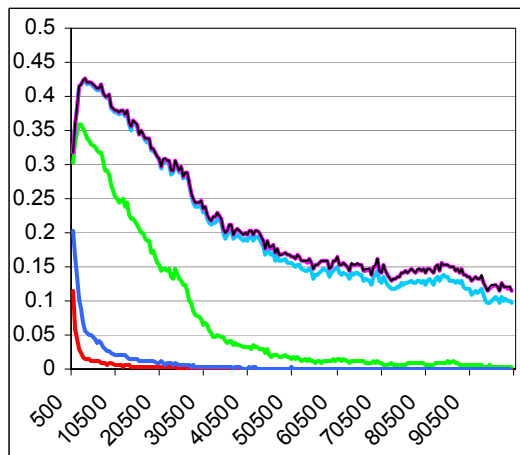
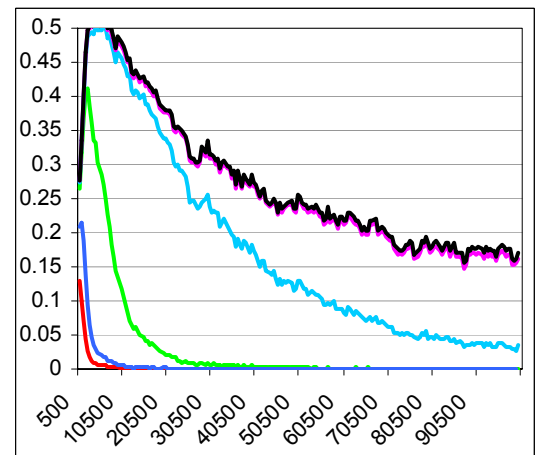
<sup>44</sup> A statement made famous by Doctor Phillip McGraw.

not only to the user-defined parameters, but also to the consistency of optimiser behaviour for the problem at hand.

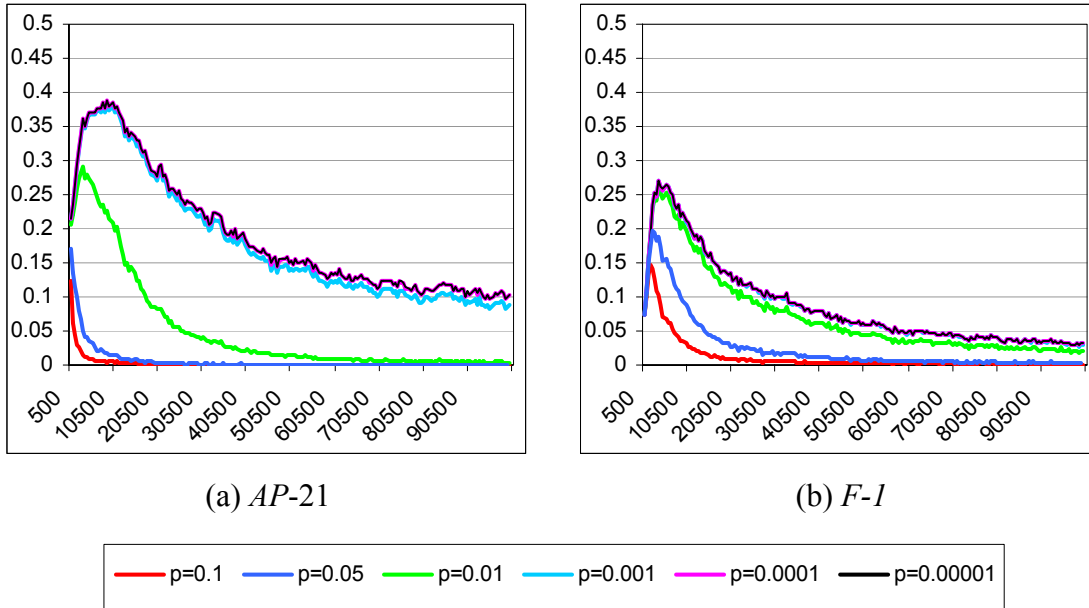


**Figure 53 — Average  $r$ -Scores for Various Precision Levels**

For all graphs, the  $x$ -axis represents the number of solutions presented to the archive and the  $y$ -axis is the average  $r$ -score over the preceding 500 evaluations.

(a) *AP-9*(b) *AP-10*(d) *AP-15*(e) *AP-16*(f) *AP-17***Figure 54 — Average  $r$ -Scores for Various Precision Levels**

For all graphs, the  $x$ -axis represents the number of solutions presented to the archive and the  $y$ -axis is the average  $r$ -score over the preceding 500 evaluations.

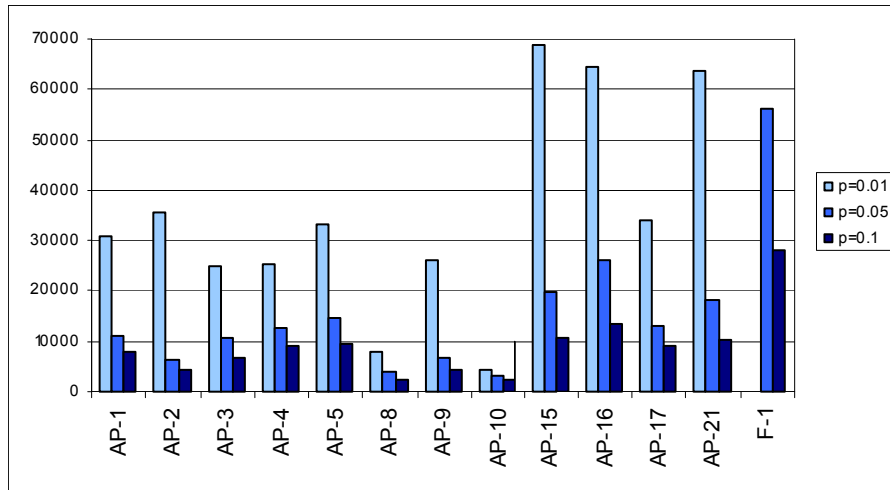


(a) AP-21

(b) F-1

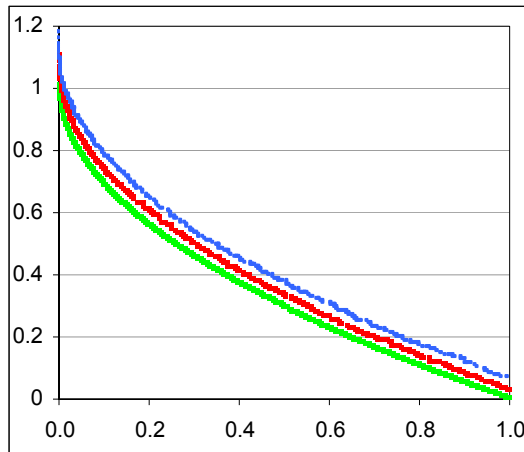
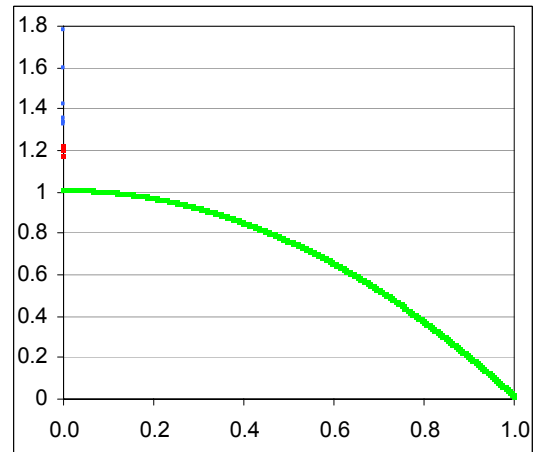
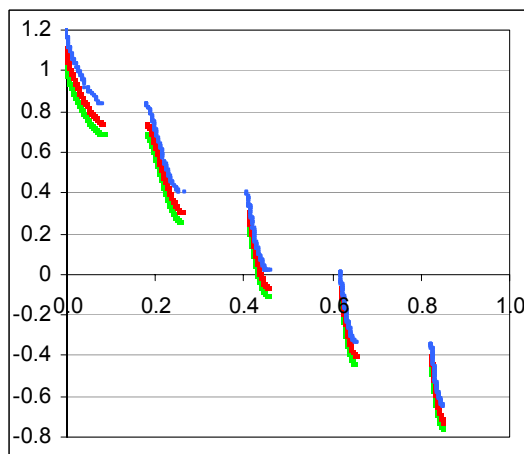
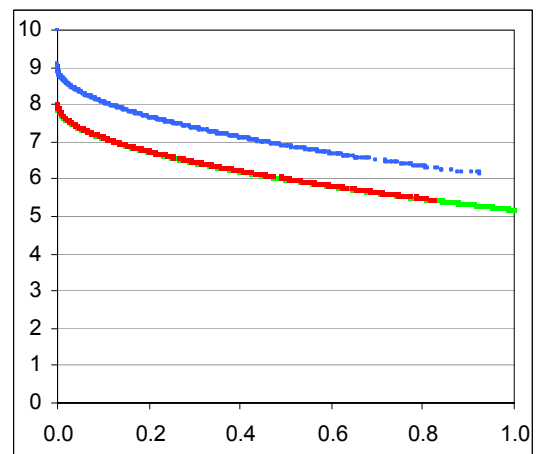
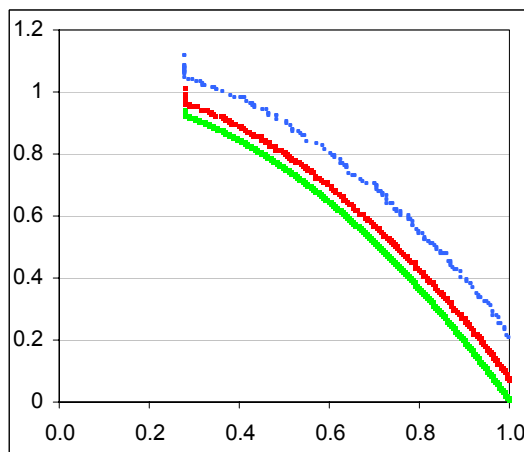
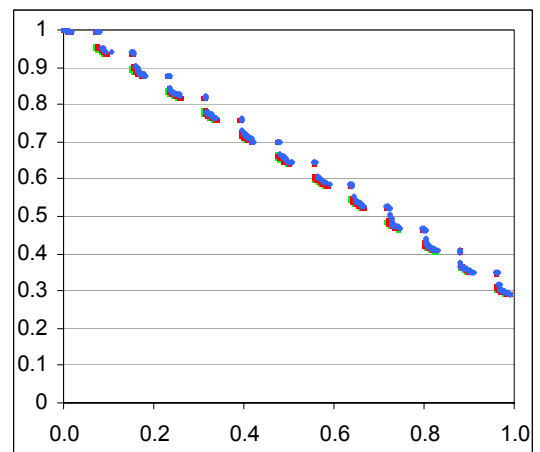
**Figure 55 — Average  $r$ -Scores for Various Precision Levels**

For all graphs, the  $x$ -axis represents the number of solutions presented to the archive and the  $y$ -axis is the average  $r$ -score over the preceding 500 evaluations.

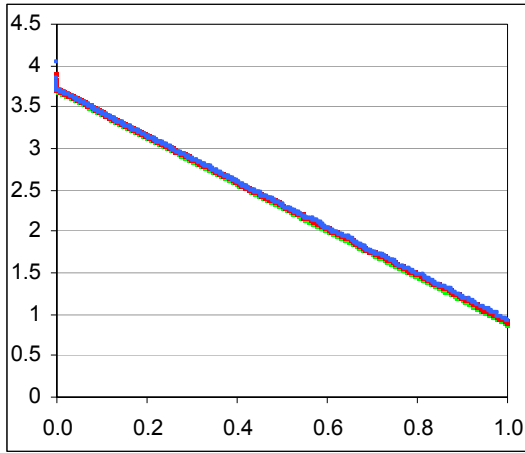
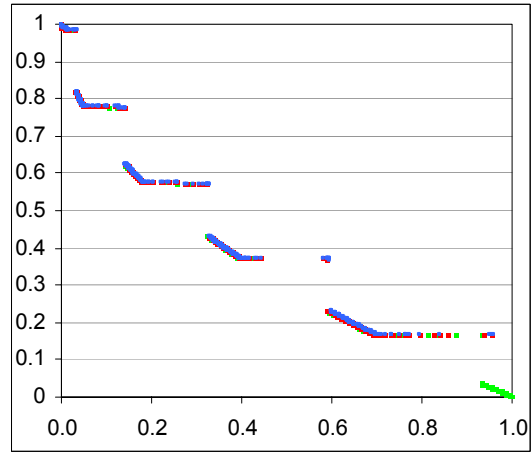
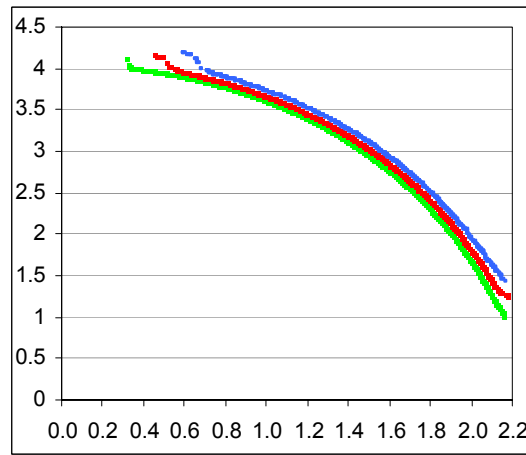
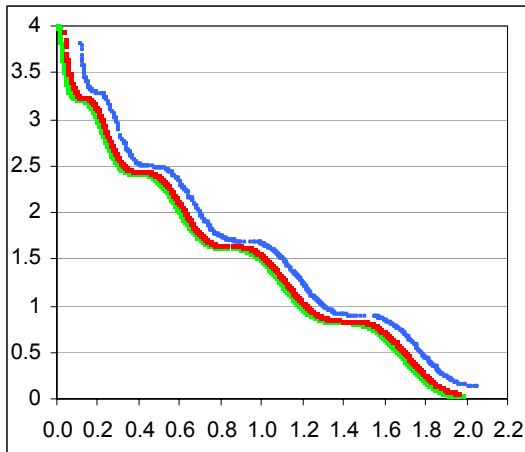
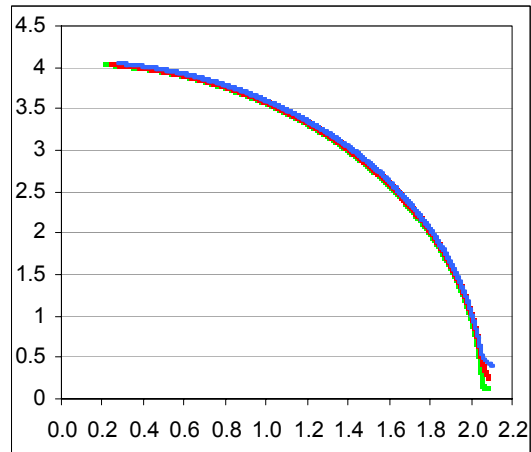

**Figure 56 — Median Number of Evaluations before Convergence Termination**

$y$ -axis represents the median number of evaluations;  $x$ -axis is the problem being optimised;  $p$  is the precision of the Mak\_Terminator. F-1 excludes  $p = 0.01$  as it fails to achieve full convergence.

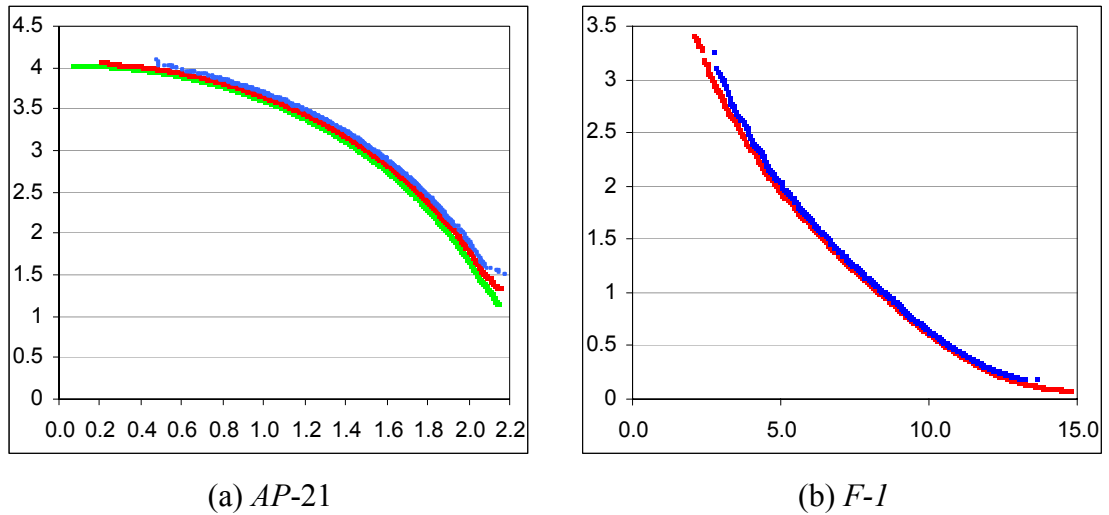


(a) *AP-1*(b) *AP-2*(c) *AP-3*(d) *AP-4*(e) *AP-5*(f) *AP-8***Figure 57 — Median Fronts as at Mak\_Termination**

For all graphs, the *x-axis* represents objective one; the *y-axis* is objective two; *green* points are frontal members at termination with  $\rho = 0.01$ ; *red* are terminated members with  $\rho = 0.05$ ; and *blue* represents the median terminated front with  $\rho = 0.01$ .

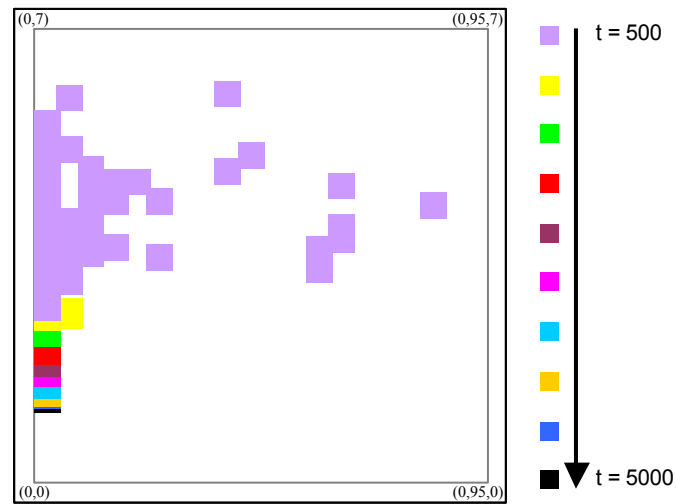

 (a) *AP-9*

 (b) *AP-10*

 (d) *AP-15*

 (e) *AP-16*

 (f) *AP-17*
**Figure 58 — Median Fronts as at Mak Termination**

For all graphs, the *x-axis* represents objective one; the *y-axis* is objective two; *green* points are frontal members at termination with  $\rho = 0.01$ ; *red* are terminated members with  $\rho = 0.05$ ; and *blue* represents the median terminated front with  $\rho = 0.01$ .



**Figure 59 — Median Fronts as at Mak Termination**

For all graphs, the *x-axis* represents objective one; the *y-axis* is objective two; *green* points are frontal members at termination with  $\rho = 0.01$ ; *red* are terminated members with  $\rho = 0.05$ ; and *blue* represents the median terminated front with  $\rho = 0.01$ .



**Figure 60 — Example Frontal Progression on *AP-2* with  $\rho = 0.05$**

Colour change represents a new set of  $\epsilon$  observations. *x-axis* is objective one; *y-axis* is objective two.

Note that progression becomes increasingly fixated on one extreme region and is relatively fine-grained. Such narrow local progression may cause premature termination.

### 7.1.3.6 LIMITATIONS AND EXTENSIONS

Though the performance of the Mak\_Terminator is promising, it is subject to a number of limitations. In particular, by using a single precision setting, there is an explicit assumption that frontal progression, distribution and extent are all equally valued in the optimisation process. However, in practice, it is possible that an end-user may wish to independently assign the precision levels for each of these improvement types, leading to, for instance, finer analysis of frontal progression than distribution. Though some modification to the base approach is required, the Mak\_Termination system is flexible enough to address such an extension. Indeed, structurally all that is required is the inclusion of multiple objective-space grids — one for each uniquely defined precision setting. Insertion into each grid is as per Section 7.1.3.2 (using the appropriate precision level to determine cell coordinates), with the utility of a non-dominated solution dictated by its membership in one of the grids. Specifically, if the proposal is *dominant* with respect to some member of the primary Mak\_Tree, it is considered valuable only if it occupies an empty cell in the *dominance-based* grid; if the solution expands the *extent* of the Mak\_Tree, it is considered useful only if it belongs to an unoccupied cell in the *extent-based* grid; otherwise, the incoming member is of high utility only if it resides in a new region according to the *distribution-based* grid. Since, at worst, this approach requires the inclusion of two additional binary supporting structures, the big-oh and amortised run-time complexities defined in Section 7.1.3.2 remain unchanged and the extended Mak\_Terminator system remains an efficient option for real-time end-of-run determination.

In cases where very little is known about the range and type of outputs expected from optimisation of the multiobjective problem at hand, definition of precision levels may be difficult. In these instances, the array-based grid can be used (Section 7.1.3.2), with the number of regions replacing the more problem-specific precision parameter. The cost of rebuilding the adaptive grid under this variation is an interesting subject of future work, though it is unlikely to preclude application in most real-time practical settings. As such, a more pressing concern is how intuitive the region parameter is. While the user may be aware of how many distinct areas they are interested in examining, it is not necessarily clear whether a grid sub-divided according to this will effectively capture frontal progression. Consider a decision-

maker who is interested in examining five distinct regions — the resulting resolution of the grid is so coarse that premature termination is likely. A better technique is therefore to use the basic region parameter for the definition of distribution and extent-based grids (as per the preceding paragraph) — where the meaning is consistent with the intention of the user — and a more fine-grained grid for frontal progression. Further development of this concept and the construction of heuristics that guide the settings of the dominance-based grid are therefore interesting avenues of future exploration.

Finally, the  $r$ -value is subject to noise. The smaller the number of evaluations used to examine the performance of the optimiser, the higher the risk of anomalies leading to premature termination. However, increasing the length at which  $r$ -value averages are observed can unduly increase the total run-time of the optimiser. As such, an appropriate area of future study is to examine the trade-offs that exist between differing  $\varepsilon$  settings with a view to establishing suitable heuristics for its formation. For the moment, this thesis maintains that a setting of  $\varepsilon \geq \Gamma/\chi$  is appropriate and tentatively suggests  $\varepsilon = 4\Gamma/\chi$  to reduce the influence of noise.

## 7.2 THE PRESENTATION OF AN UNBOUNDED MAK\_TREE

While maintaining a complete and accurate elite archive throughout the optimisation process is inherently valuable (for reasons elaborated in Section 6.2), presentation of such a potentially large set to the decision maker at the completion of a run may be inappropriate [200, 201], not least because it may be overwhelming and difficult to process. As such, transforming archives into an appropriately distributed set of some defined size ( $\gamma$ ) is of key practical importance.

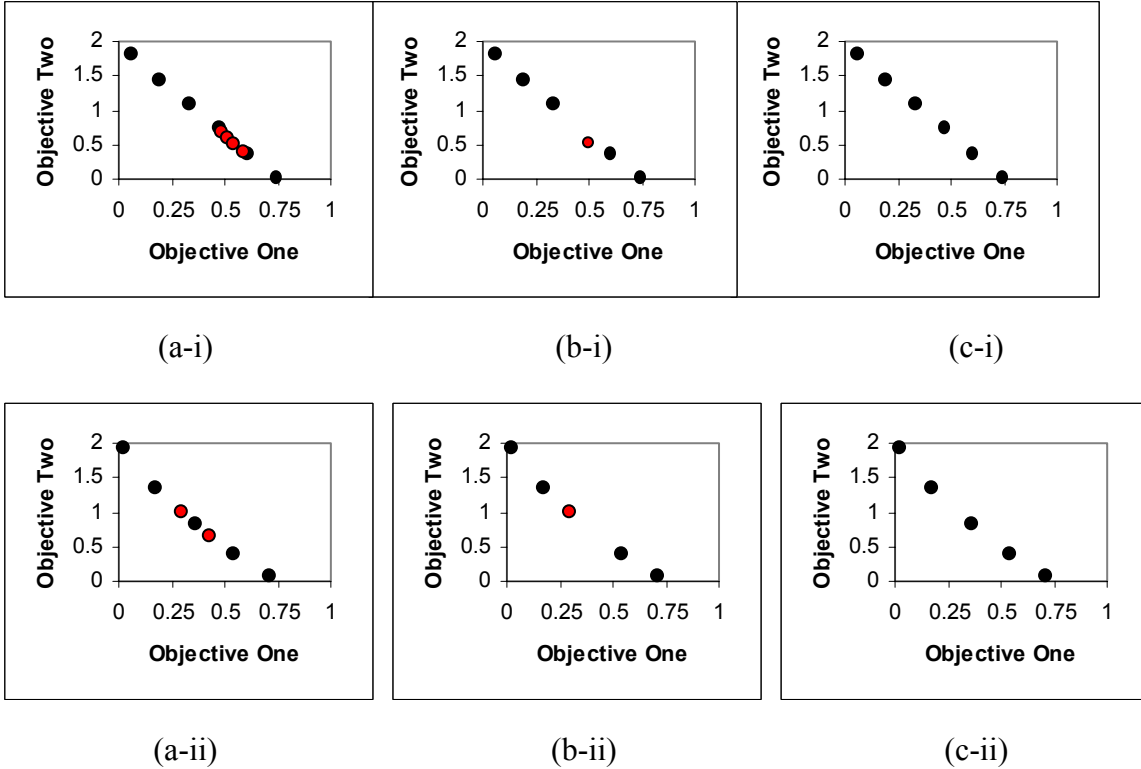
Obviously, standard truncation-based systems are freed of this burden to some extent<sup>45</sup>, though there can be no guarantee that the final truncated set represents an evenly distributed approximation of the leading front<sup>46</sup>. The existence of frontal degradation in truncated sets (see Section 6.2.1.1) rejects the assertion that

---

<sup>45</sup> Deb and Goyal [200] note that even truncated archives are often too large for practical purposes and may require further summarisation.

<sup>46</sup> An argument can be made that ill-distributed fronts that emphasise clusters or isolation may inform decision makers on the fragility or scarcity of particular solutions. The validity of such a claim is debatable, since it is difficult to confirm that frontal distribution characteristics are a product solely of the problem being addressed — variations in distribution may also reflect biases in the optimiser, for instance.

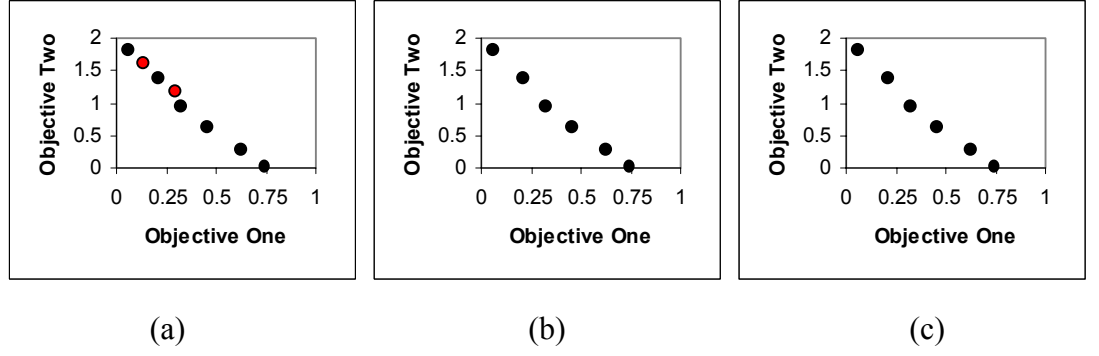
presentational members are non-dominated with respect to all generated solutions, while the incremental nature of archival updating makes establishing an evenly distributed set difficult. The latter point is a subtle one — typically the goal of truncated archives in multiobjective optimisers is *not* to maintain an evenly distributed set, it is to maintain a collection of *uncrowded* solutions according to an (often inaccurate) approximation of the space explored thus far (to encourage the pursuit of poorly explored portions of objective-space). Consider Figure 61: if the archiving procedure was charged with establishing an even distribution<sup>47</sup> — as is important to the quality of the presentation set — the resultant archive should be (c); instead it is (b). While it is true that the goals of seeking uncrowded solutions and an evenly distributed set may often overlap (as illustrated in Figure 62), the potential for



**Figure 61 — Archival Truncation Leading to Poorly Distributed Presentation Sets**

(i) illustrates truncation according to hierarchical clustering (as in Zitzler [201]). (ii) shows truncation according to the cuboid method of Deb *et al.* [82]. (a) is the initial set that requires the truncation; the red solutions should be removed if the presentation set is to be evenly distributed. (b) is the resultant set after truncation — note the presence of the red solution in each case, indicating a sub-optimal distribution. (c) is the ideally distributed truncated set.

<sup>47</sup> This work assumes that an even distribution in objective-space is required. Generating evenly distributed sets in multi-dimensional decision spaces is a non-trivial task that is beyond the scope of this study.



**Figure 62 — Archival Truncation Leading to Appropriately Distributed Presentation Sets**  
 (a) represents the initial set that requires truncation; the red solutions should be removed if the presentation set is to be evenly distributed. (b) is the resultant truncated archive using cuboid crowding estimates, which achieves the same distribution as the optimal set presented in (c).

divergence indicates that the performance of truncated archives for decision-maker presentation is, at the very least, sub-optimal.

Given the apparent weaknesses of the truncated approach (which will be explored empirically in Section 7.2.1), developing a mechanism for the reduction of *unbounded* Mak\_Trees into a more manageable presentational form seems to be a promising avenue of exploration. While Deb and Goyal [200] offer a clustering-based system (derived from the average-linkage method used in [201]) that is applicable, the process is expensive (carrying a computational burden of  $O(n^2)$  [95]) and is subject to the same performance issues illustrated in Figure 61i. A better method, which produces a suitably distributed collection of solutions with low computational overhead, is to derive the average nearest-neighbour distance ( $\theta$ ) across all members and space the selection of solutions by a distance of  $\iota$ :

$$\iota = \frac{\theta n}{\gamma} \quad (68)$$

If presentation only occurs at the completion of a run, the  $O(n)$  cost of such an operation will not affect the theoretical run-time performance of the Mak\_Tree and can be achieved by simply traversing the non-dominated list twice: once to determine the value of  $\theta$ , and once to select the members for presentation (see Algorithm 4 for details). As illustrated in Table 14 however, this simplistic algorithm is prone to producing sets whose cardinality can vary quite dramatically from the  $\gamma$  value specified by the end-user.

**Algorithm 4 — Forming A Basic Presentation Set**
**Inputs:**

$\iota$	The optimal length between successive nodes.
$root$	The root of the non-dominated Mak_Tree
1: $Presentation := \emptyset$	The presentation set is initially empty.
2: $current := \text{leftmost}(root)$	Navigate from the start of the list.
3: while ( $current \neq \text{null}$ )	While there are more nodes in this list,
4: $Presentation := Presentation \cup \{current\}$	include $current$ node in the display set
5: $next := \overrightarrow{current}$	and find the next node that is further
6:   while ( $(\delta(current, next) < \iota) \wedge (next \neq \text{null})$ )	than $\iota$ away.
7: $next := \overrightarrow{next}$	
8: $current := next$	
9: $Presentation_{ Presentation } := \text{rightmost}(root)$	Ensure the last node is an extreme member.

**Table 14 — Presentation Set Sizes Using the Basic Technique**

Results reflect averages produced from twenty distinct NSGA-II runs (see Appendix B.1.2 for settings) of 10,000 evaluations each. Values in brackets represent standard deviation.

	$\gamma = 10$	$\gamma = 20$	$\gamma = 50$	$\gamma = 100$
<b>AP-1</b>	14.40 (0.50)	27.15 (0.88)	58.90 (2.00)	99.35 (3.98)
<b>AP-2</b>	14.90 (0.85)	26.60 (1.70)	54.20 (5.27)	82.50 (9.61)
<b>AP-3</b>	18.10 (1.41)	32.50 (1.70)	66.90 (3.74)	105.10 (5.69)
<b>AP-4</b>	15.95 (2.87)	28.55 (6.96)	58.85 (21.69)	97.85 (44.95)
<b>AP-5</b>	14.00 (0.79)	25.80 (1.54)	54.25 (3.04)	84.20 (8.19)
<b>AP-8</b>	21.20 (2.02)	30.80 (1.64)	57.10 (2.38)	89.25 (3.88)
<b>AP-9</b>	19.90 (1.17)	36.80 (2.04)	75.55 (2.54)	120.30 (3.81)
<b>AP-10</b>	26.30 (2.08)	39.15 (2.52)	73.90 (5.45)	120.05 (11.24)
<b>AP-15</b>	14.65 (0.59)	27.50 (1.05)	60.35 (2.58)	103.50 (4.08)
<b>AP-16</b>	15.00 (0.56)	28.50 (1.05)	62.00 (2.05)	102.25 (3.67)
<b>AP-17</b>	15.45 (0.60)	29.60 (0.94)	67.55 (2.06)	121.90 (4.55)
<b>AP-21</b>	14.85 (0.67)	27.85 (1.27)	61.40 (2.85)	105.25 (5.79)
<b>F-1</b>	16.35 (0.67)	30.05 (1.10)	60.90 (1.89)	94.60 (5.05)

If the fidelity of the  $\gamma$  parameter is of utmost importance, a more complex algorithm that is capable of refining the presentation set is necessary. As described in Algorithm 5 (henceforth referred to as the *Mak\_Presentation* technique), the most intuitive way to achieve such a task is to adjust the value of  $\iota$  according to the cardinality of the presentation set — lengthening the spacing if the collection is too large and shortening it when the set is too small. Provided the number of modifications is limited to approximately  $\log n$ , this improved methodology is



## Algorithm 5 — The Mak\_Presentation Algorithm

**Inputs:**

$\theta$	The average objective-space distance between solutions.
$Presentation$	The current presentation set.
$\gamma$	The required size of the presentation set.
$root$	The root of the non-dominated Mak_Tree
$max$	Maximum difference between required and actual presentation set size.
$maxAdjusts$	The maximum number of times the set may be adjusted.
1: $ProducedSets := \emptyset$	Establish storage for all sets.
2: $adjusts := 0$	
3: $form\_presentation\_set(root, \frac{\theta n}{\gamma})$	Form the initial presentation set and record it for future reference.
4: $ProducedSets := ProducedSets \cup \{Presentation\}$	Record difference between the
5: $diff := \gamma -  Presentation $	desired and actual size of the set,
6: if $((\gamma + diff) < 0)$	ensuring $diff$ is never less than $-\gamma$ .
7: $diff := 1 - \gamma$	So long as the size of the set
8: while $((  Presentation  - \gamma  > max) \wedge (adjusts < maxAdjusts))$	is inappropriate and the set may
9: $\iota := \theta \times \frac{n}{\gamma + diff}$	still be adjusted
10: $form\_presentation\_set(root, \iota)$	Update the length and form
11: $ProducedSets := ProducedSets \cup \{Presentation\}$	a new presentation set.
12: $taper := \frac{maxAdjusts - adjusts}{maxAdjusts}$	Record the set for reference later.
13: $diff := diff + (taper \times (\gamma -  Presentation ))$	Taper the scaling of the change
14: if $((\gamma + diff) < 0)$	according to number of $adjusts$ .
15: $diff := 1 - \gamma$	Record the tapered change in
16: $adjusts := adjusts + 1$	difference.
17: $Presentation := selectBest(ProducedSets)$	Make the set with cardinality
	nearest to $\gamma$ the final presentation
	set (ties favour larger sets).

considerably less expensive than the clustering approach, with an operational overhead of  $O(n \log n)$  that does not affect the overall amortised time cost of the Mak\_Tree<sup>48</sup>. As illustrated in Table 15, this basic extension is sufficient for the production of sets that are consistently very near to the specified  $\gamma$  value, with marked variations evident only at the higher setting levels in AP-4. Still, the potential for such variation is a concern and future work should explore more sophisticated approaches to adjusting the value of  $\iota$ , with annealing techniques offering a particularly interesting avenue.

<sup>48</sup> So long as presentation only occurs at the end of the run or at a small number of fixed intervals.

**Table 15 — Presentation Set Sizes Using the Mak\_Presentation Algorithm**

Results reflect averages produced from twenty distinct NSGA-II runs (settings defined in Appendix B.1.2) of 10,000 evaluations each.  $max = 1$ ;  $maxAdjusts = 10$ ; values in brackets represent standard deviation.

	$\gamma = 10$		$\gamma = 20$		$\gamma = 50$		$\gamma = 100$	
	Set Size	Adjustments	Set Size	Adjustments	Set Size	Adjustments	Set Size	Adjustments
<b>AP-1</b>	10.65 (0.49)	2.00 (0.00)	20.20 (0.70)	2.10 (0.31)	50.00 (0.79)	1.45 (0.60)	100.00 (0.79)	1.25 (1.07)
<b>AP-2</b>	10.30 (0.66)	2.15 (0.49)	19.80 (0.77)	1.70 (0.66)	49.90 (0.79)	1.45 (1.28)	96.00 (5.68)	7.75 (3.24)
<b>AP-3</b>	10.70 (0.66)	3.75 (0.55)	20.45 (0.83)	2.65 (0.67)	50.00 (0.73)	2.00 (0.65)	100.05 (0.76)	1.40 (1.10)
<b>AP-4</b>	9.70 (0.73)	2.70 (1.26)	19.70 (0.86)	3.00 (1.95)	46.75 (6.59)	4.95 (3.35)	85.45 (25.0)	5.25 (3.80)
<b>AP-5</b>	10.35 (0.75)	2.10 (0.45)	19.80 (0.70)	1.65 (0.49)	49.95 (0.76)	1.25 (0.72)	96.65 (3.65)	6.95 (3.38)
<b>AP-8</b>	9.95 (0.83)	6.25 (1.62)	19.95 (0.89)	5.10 (2.81)	50.20 (0.83)	1.50 (0.61)	98.65 (1.18)	5.80 (3.38)
<b>AP-9</b>	10.15 (0.93)	4.30 (0.57)	20.10 (0.97)	4.20 (0.70)	50.25 (0.72)	2.90 (1.25)	100.10 (0.64)	2.65 (1.23)
<b>AP-10</b>	10.15 (0.67)	7.05 (0.89)	20.15 (0.81)	5.40 (1.31)	50.10 (0.79)	2.70 (1.69)	100.00 (0.65)	2.50 (1.88)
<b>AP-15</b>	10.40 (0.75)	2.20 (0.41)	20.35 (0.67)	2.05 (0.22)	49.75 (0.85)	1.55 (0.51)	100.00 (0.92)	1.50 (1.15)
<b>AP-16</b>	9.60 (0.82)	2.70 (0.47)	20.25 (0.79)	2.30 (0.47)	49.85 (0.88)	1.45 (0.51)	99.80 (0.77)	1.25 (1.07)
<b>AP-17</b>	9.95 (0.76)	2.65 (0.49)	20.10 (0.72)	2.65 (0.49)	50.15 (0.75)	2.20 (0.52)	99.75 (0.79)	2.05 (1.43)
<b>AP-21</b>	10.25 (0.72)	2.25 (0.44)	20.05 (0.76)	2.10 (0.55)	50.00 (0.86)	1.65 (0.59)	99.70 (0.86)	2.10 (1.55)
<b>F-1</b>	9.85 (0.88)	2.70 (0.47)	20.25 (0.85)	2.35 (0.49)	50.10 (0.85)	1.75 (1.21)	99.00 (1.21)	3.65 (3.62)

### 7.2.1 EMPIRICAL ANALYSIS OF PRESENTATIONAL ALGORITHMS

With the low-cost Mak\_Presentation algorithm in place, it is worthwhile examining its performance relative to pre-existing methods — namely, the clustering technique operating on both bounded and unbounded sets, and the use of end-of-run NSGA-II archives. Since this work posits that the quality of a presentational set corresponds to how evenly distributed that set is, the performance of each system is indicated by the  $q$  metric:

$$\begin{aligned}
 \Delta_i &= \delta(P_i, P_{i+1}) \forall i \in \{1, 2, \dots, |P| - 1\} \\
 \sigma &= \sqrt{\frac{1}{|\Delta| - 1} \sum_{i=1}^{|\Delta|} (\Delta_i - \bar{\Delta})^2} \\
 q &= \sigma \div \bar{\Delta}
 \end{aligned} \tag{69}$$

where  $P$  is the set under examination;  $\Delta$  is the set of (Euclidean) distances between successive neighbouring solutions in objective-space;  $\sigma$  is the standard deviation of those distances; and  $q$  is the standard deviation divided by the mean (to prevent tightly packed presentation sets from becoming biased in the analysis).

To ensure a rich exploration of the characteristics of each algorithm,  $q$ -based box-plots and averages are provided, with results for each problem derived from twenty

runs of NSGA-II (using 10,000 evaluations and parameters defined in Appendix B.1.2). To provide statistical clarity, non-parametric two-tailed Kruskal-Wallis significance tests [202, 203] are performed (using tools provided as part of the PISA assessment suite [204, 205]) on the collected results for each problem, while randomly selected outputs are graphed to offer a visual representation (for all relevant tables and graphs see pages 173–177).

The results indicate the power of the Mak\_Presentation algorithm. The new technique achieves better-distributed presentation sets on every tested function with respect to both box-plots (Figure 64) and averages (Table 16), with statistically *significant* differences (at the 5% level) in all cases. The visualisations (Figure 65, Figure 66, Figure 67 and Figure 68) support these conclusions: the Mak\_Presentation technique produces well-distributed sets irrespective of the characteristics of the unbounded front, with obvious disparity between it and the reduced bounded archive in *AP-1*, *AP-5*, *AP-15* and *F-1*. Note that even non-contiguous fronts (particularly *AP-3* and *AP-10*) are effectively captured by the Mak\_Presentation mechanism, an important consequence of using successor-distances in the selection phase and nearest-neighbour measurements in the formation of  $\theta$ . By capitalising on nearest-neighbour metrics, the average distance between points is not skewed by large unoccupied regions of the objective-space<sup>49</sup>, while successor-distances encourage the selection of members in distinct isolated regions (see Figure 63 for an illustration). It is also important to note that the simple technique described in Algorithm 5 captures the extreme members of each non-dominated set, which is significant given that Knowles *et al.* [206] cite this as a desirable archival property<sup>50</sup>.

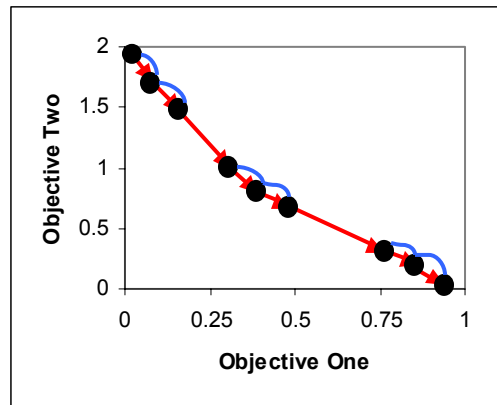
The results also offer empirical proof that bounded NSGA-II archives fail to maintain an evenly distributed collection of solutions suitable for immediate presentation, with  $q$ -based box-plots (Figure 64) and averages (Table 16) that are

<sup>49</sup> Only those isolated regions that are occupied by a single point in objective-space will induce misleading averages – and these will only become significant if the number of such isolated regions is high relative to the total number of occupants in the set.

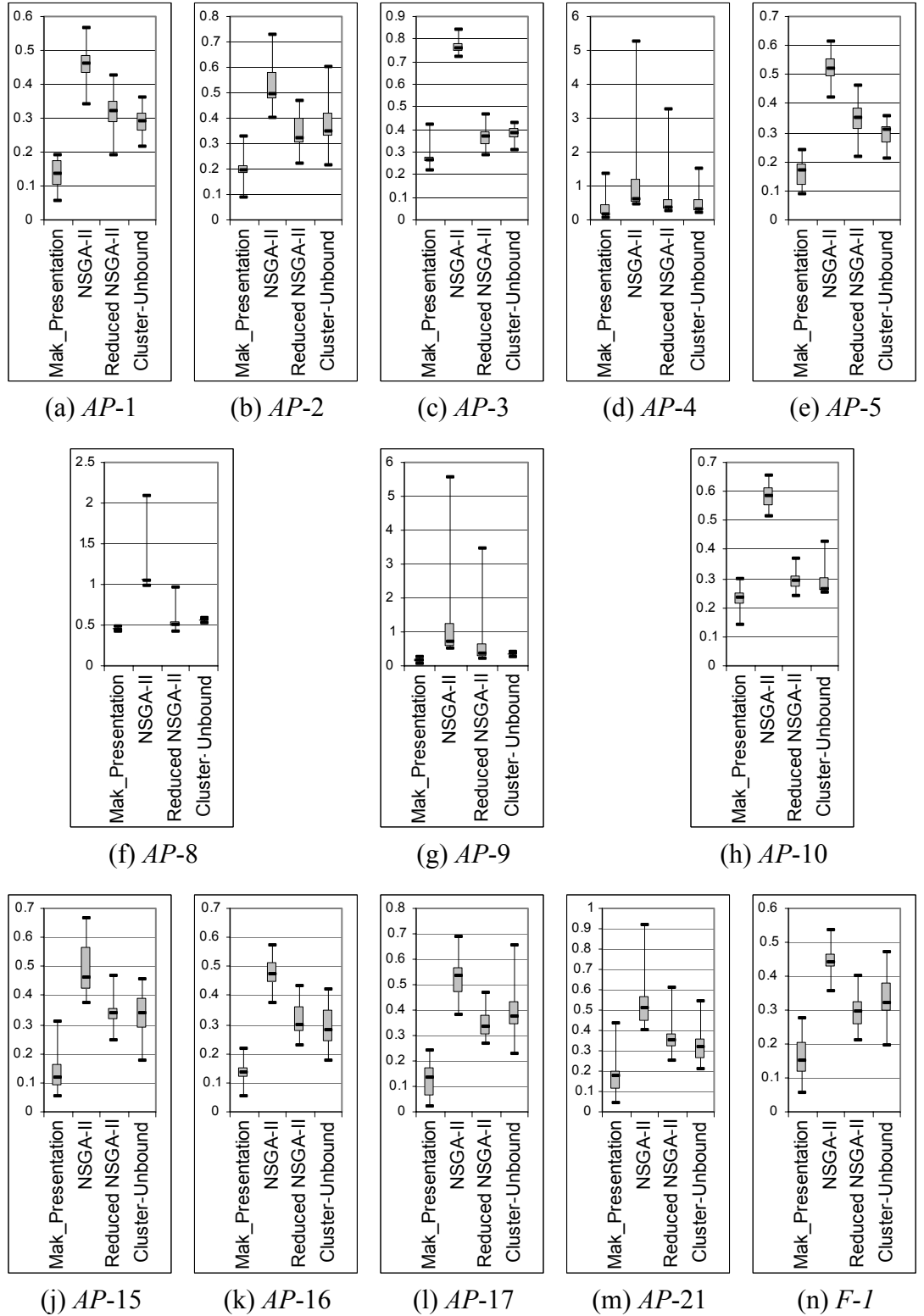
<sup>50</sup> It is worth commenting here that Deb and Goyal [200] suggest the importance of maintaining extreme solutions but offer no practical advice as to how this can be achieved with standard clustering. For this work, the clusters are ordered by their position in two-dimensional objective-space, with the first and last clusters offering extreme members as their contribution to the presentation set.

worse than every other examined approach. Moreover, the Kruskal-Wallis tests confirm that these differences are again statistically *significant* (at the 5% level) in *every* instance. Such inferiority is directly attributable to the difference between the maintenance of uncrowded regions and the development of an evenly distributed set. In practice, the pursuit of low-density areas comes at the expense of a well-spaced archive.

Of the remaining techniques, it is difficult to establish a clear preference between clustering of the complete unbounded set and of the final archive. Where performances are *significantly* different, the unbounded approach is preferable on *AP-1*, *AP-5* and *AP-9*, while the bounded technique is better on *AP-2*, *AP-8*, *AP-17* and *F-1*. It is reasonable to conclude that such results infer that the depth of the set under analysis is at most a secondary issue when employing clustering methodologies for set reduction (though using the truncated archive does permit the inclusion of weak solutions, as per Section 6.2.1). Whether this statement is a general maxim for the production of presentation sets, or limited only to clustering techniques, is an interesting question that is left as an avenue for future work.



**Figure 63 — Nearest Neighbours and Successor Distances in Determining Presentation Sets**  
*blue lines indicate nearest neighbours; red arrows show successors.*



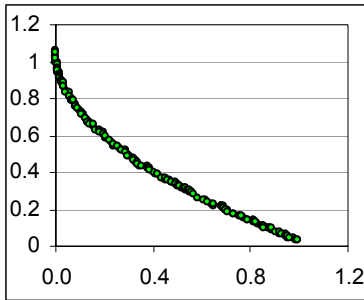
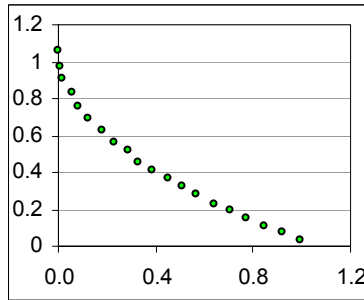
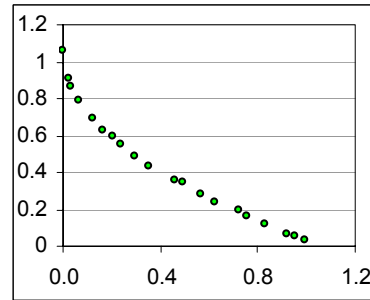
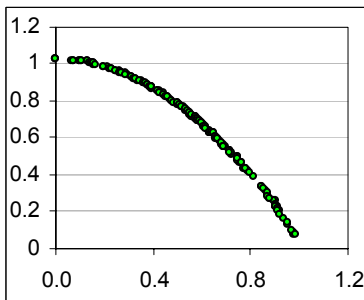
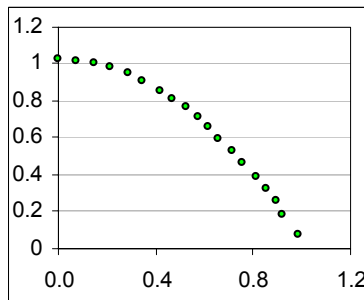
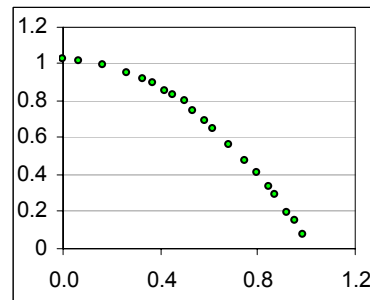
**Figure 64 —  $q$ -value Performance of Presentational Algorithms (Box Plots)**

$y$ -axis represents  $q$ -value scores;  $x$ -axis reflects the chosen algorithm. *NSGA-II* is the final unrefined 50 member archive for *NSGA-II*; *Reduced NSGA-II* trims this store to 20 members via clustering. *Mak\_Presentation* and *Cluster-Unbound* each operate on the unbounded *Mak\_Tree* and refine it to 20 members. Lower  $q$ -values are preferred.

**Table 16 — Average  $q$ -value Performance of Presentational Algorithms**

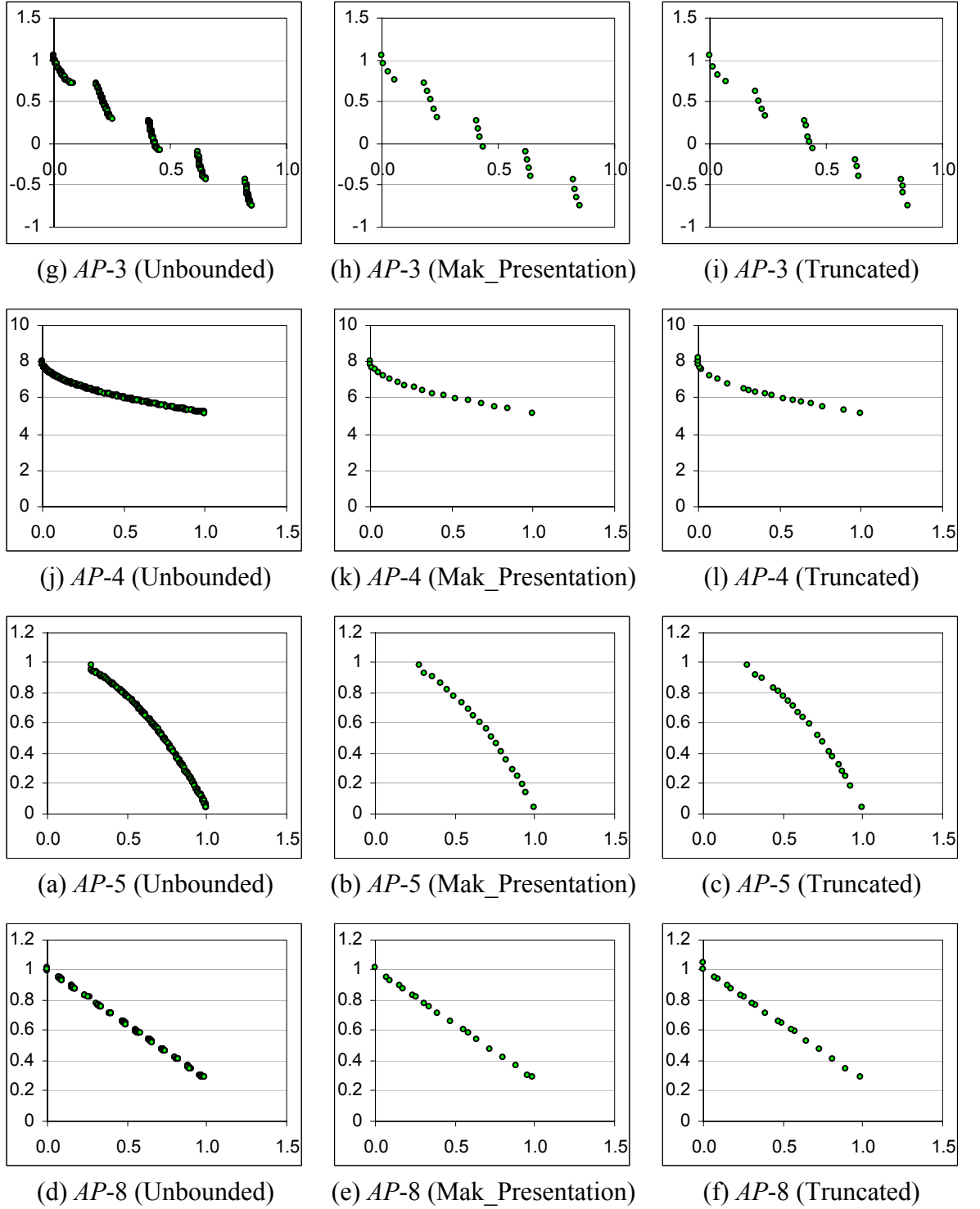
*NSGA-II Archive* is the final unrefined 50 member elite store for NSGA-II; *Cluster-Reduced NSGA-II* trims this store to 20 members via clustering. *Mak\_Presentation* and *Cluster-Reduced Unbounded* each operate on the unbounded Mak\_Tree and refine it to 20 members. Lower  $q$ -values are preferred.

	Mak Presentation of Unbounded Archive	NSGA-II Archive	Cluster- Reduced NSGA-II Archive	Cluster- Reduced Unbounded Archive
<b>AP-1</b>	0.13	0.46	0.32	0.29
<b>AP-2</b>	0.19	0.53	0.34	0.37
<b>AP-3</b>	0.28	0.76	0.37	0.38
<b>AP-4</b>	0.34	1.17	0.70	0.46
<b>AP-5</b>	0.16	0.52	0.34	0.30
<b>AP-8</b>	0.44	1.14	0.54	0.55
<b>AP-9</b>	0.14	1.36	0.77	0.33
<b>AP-10</b>	0.23	0.58	0.29	0.29
<b>AP-15</b>	0.13	0.49	0.34	0.33
<b>AP-16</b>	0.13	0.48	0.31	0.29
<b>AP-17</b>	0.12	0.52	0.34	0.39
<b>AP-21</b>	0.17	0.52	0.36	0.33
<b>F-1</b>	0.15	0.44	0.30	0.33

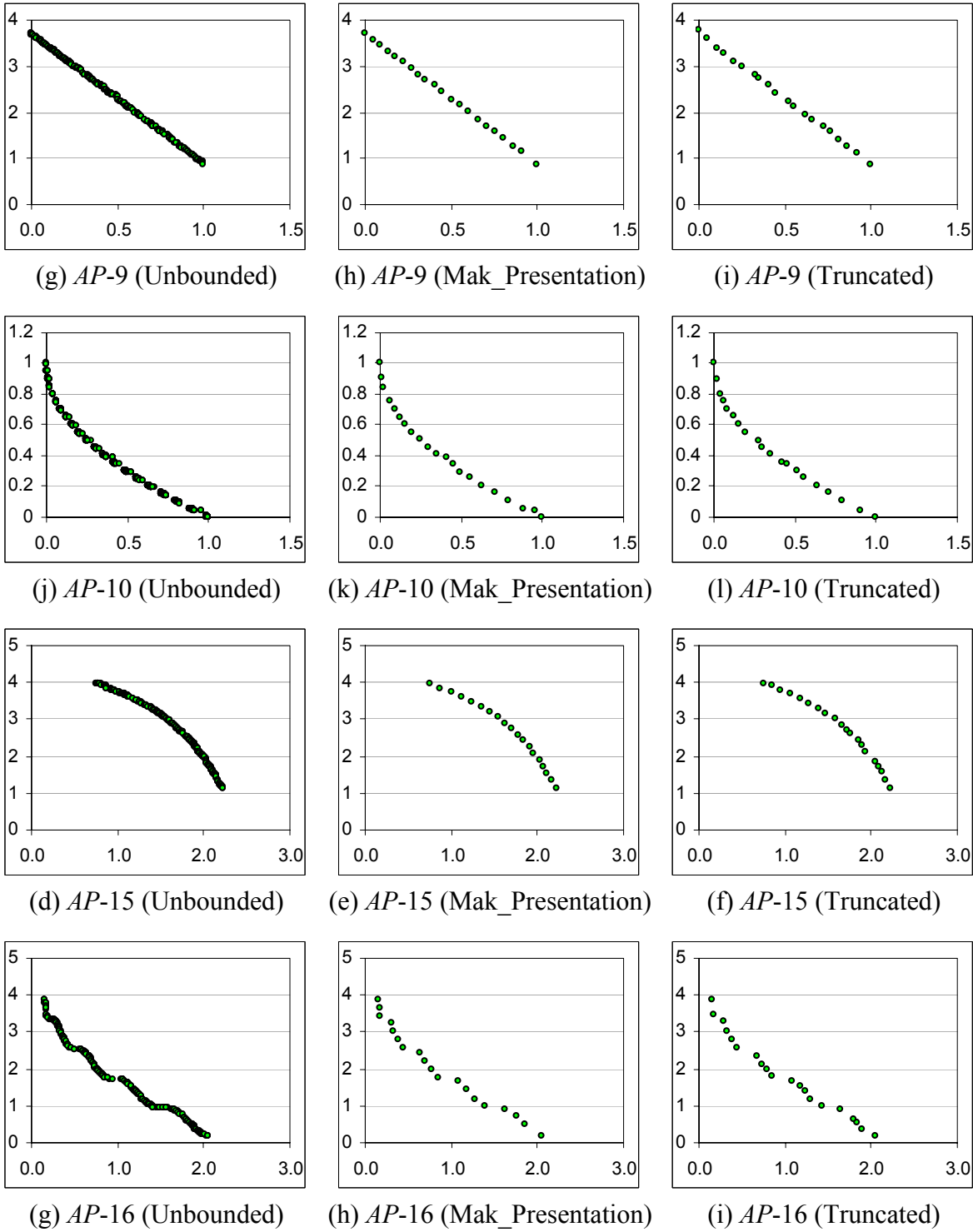

 (a) *AP-1* (Unbounded)

 (b) *AP-1* (Mak\_Presentation)

 (c) *AP-1* (Truncated)

 (d) *AP-2* (Unbounded)

 (e) *AP-2* (Mak\_Presentation)

 (f) *AP-2* (Truncated)

**Figure 65 — Example Presentation Sets ( $\gamma=20$ )**

$x$ -axis is objective-one;  $y$ -axis is objective-two. *Unbounded* is the complete representation of the randomly selected front; *Mak\_Presentation* uses the Mak\_Presentation algorithm to reduce this unbounded set; *Truncated* uses the clustering algorithm to reduce the final NSGA-II archive.

**Figure 66 — Example Presentation Sets ( $\gamma=20$ )**

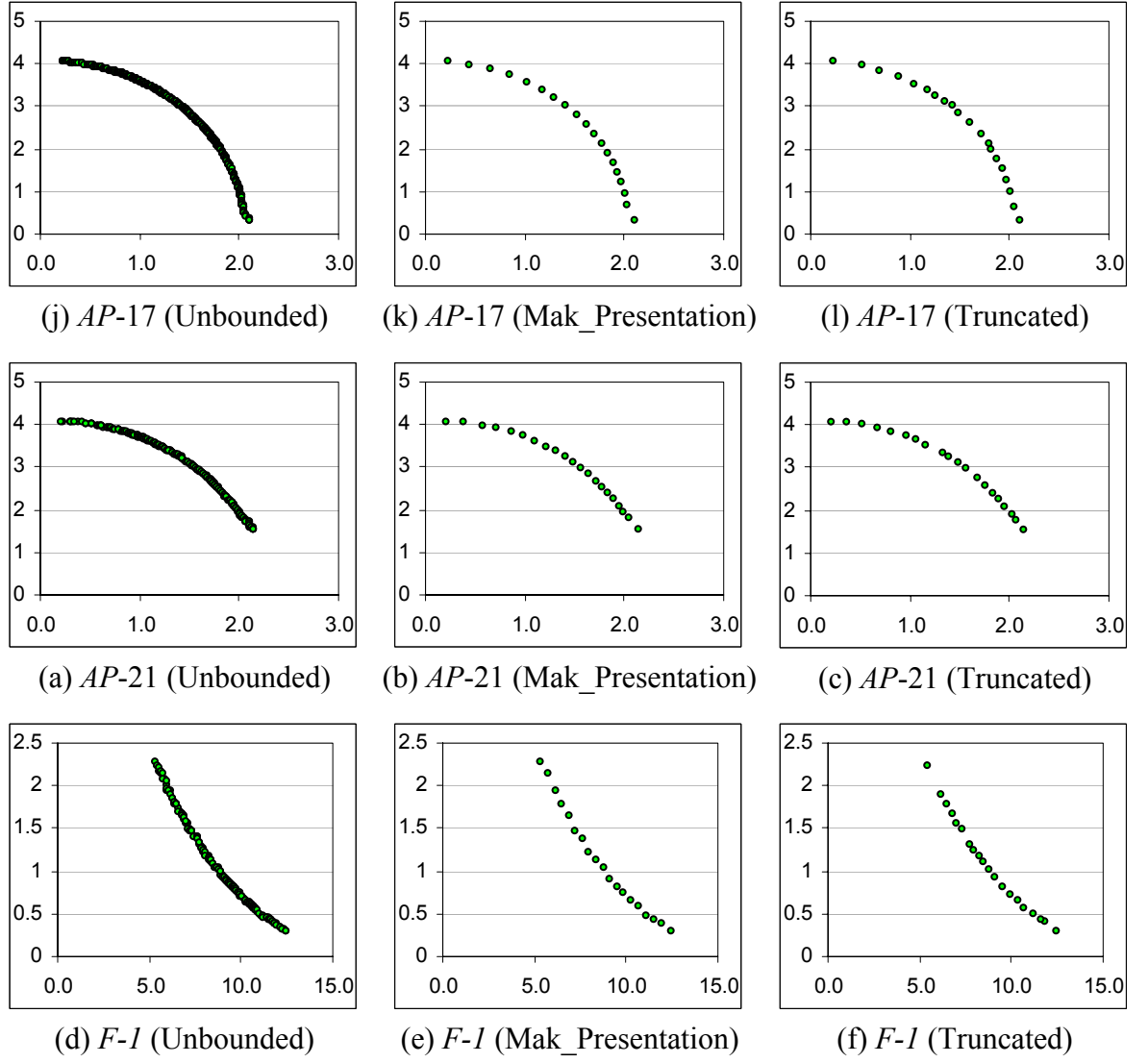
*x-axis* is objective-one; *y-axis* is objective-two. *Unbounded* is the complete representation of the randomly selected front; *Mak\_Presentation* uses the Mak\_Presentation algorithm to reduce this unbounded set; *Truncated* uses the clustering algorithm to reduce the final NSGA-II archive.



**Figure 67 — Example Presentation Sets ( $\gamma=20$ )**

$x$ -axis is objective-one;  $y$ -axis is objective-two. *Unbounded* is the complete representation of the randomly selected front; *Mak\_Presentation* uses the Mak\_Presentation algorithm to reduce this unbounded set; *Truncated* uses the clustering algorithm to reduce the final NSGA-II archive.



**Figure 68 — Example Presentation Sets ( $\gamma=20$ )**

$x$ -axis is objective-one;  $y$ -axis is objective-two. *Unbounded* is the complete representation of the randomly selected front; *Mak\_Presentation* uses the Mak\_Presentation algorithm to reduce this unbounded set; *Truncated* uses the clustering algorithm to reduce the final NSGA-II archive.

### 7.3 CONCLUDING REMARKS

Exploring the potential of the unbounded Mak\_Tree as an off-line supporting structure, this chapter has examined how an optimiser may be improved when an efficient elite store can be effectively manipulated. Particular benefit is seen in the development of stopping criteria, an ill-explored field whose advancement has been hamstrung by the degenerative nature of truncated archival sets. By harnessing the unbounded elite store, the proposed Mak\_Terminator offers a simple, intuitive and particularly efficient approach to termination that can, in real-time, identify stopping points in optimiser runs according to user-defined precision settings and timing thresholds. This work represents the first fully realised development of stopping criteria in an unbounded domain, and the promising empirical results suggest that the technique is one of merit and worthy of further investigation.

Post-termination, the unbounded nature of the Mak\_Tree means that, for practical use, the construction of a reduced presentational set is a necessity. However, existing clustering-based techniques scale poorly to large archival sets due to their  $O(n^2)$  complexity. In response, this section has proposed a novel Mak\_Presentation algorithm that is explicitly tasked with achieving evenly distributed sets at a relatively low cost. The empirical results emphasise that beyond the improvements in computational overhead, the system also produces consistently better distributed sets than pre-existing clustering techniques. Such promising findings suggest that the Mak\_Presentation algorithm offers an excellent means for reducing the potentially cumbersome unbounded Mak\_Tree into a valuable decision making tool for end-users. Additionally, results illustrate that the deficiencies of the truncated archive are also evident when it is used as an unrefined presentational set. The unevenly distributed NSGA-II archives, together with the potential for weak members, should underline that the use of truncated elite stores as a *de facto* presentational set is unacceptable.

# Chapter

# 8



## Extending the Mak\_Tree

*“All progress is precarious,  
and the solution of one  
problem brings us face to  
face with another problem.”*

*Martin Luther King Jr. —  
Politician and Human  
Rights Activist*

## 8 EXTENDING THE MAK TREE

Section 6.3 introduced a highly efficient approach to unbounded archiving in the special bi-objective case. While both empirical and theoretical results proved to be promising, the structure itself could be considered bare-boned — the Mak\_Tree is principally concerned with maintaining access to the elite solutions of the prevailing non-dominated front, and little more. In practice, supplementary information regarding the nature of the non-dominated front, such as crowding and extent-based statistics, are also of significant value, particularly in the selection phase of an evolutionary algorithm. As such, this chapter describes a range of extensions to the Mak\_Tree that are both efficient and of value to real-world optimisers.

### 8.1 MAINTAINING CROWDING INFORMATION

As illustrated in Section 6.2.1.3, simply having access to an unbounded elite set facilitates considerably more accurate frontal density analysis than is possible in the coarse approximations provided by truncated sets. The principal issue then is whether crowding information can be efficiently extracted from, or maintained in, an unbounded archive. Unfortunately, pre-existing generic approaches tend to be highly expensive and the naïve application of such techniques is therefore inappropriate.

#### 8.1.1 GENERIC CROWDING METRICS

As discussed in Section 6.2.1.3, multiobjective evolutionary algorithms principally employ crowding metrics to push the search towards insufficiently explored areas of the objective-space<sup>51</sup>. While the techniques vary considerably from algorithm to algorithm, all approaches are burdened with deficiencies that inhibit their application in an unbounded environment (and many are susceptible to poor performance even with sharply truncated sets).

##### 8.1.1.1 *FITNESS SHARING AND DISTANCE-BASED CLUSTERING*

Common to most first-generation evolutionary multiobjective optimisers (such as NSGA [2], NPGA [142], SPEA [137] and MOGA [181]) is the notion of fitness sharing and distance-based clustering (see [51, 207] for summaries). While the techniques used to achieve solution clustering are diverse, the general motivation

---

<sup>51</sup> Crowding metrics also play an important role in archive truncation, but that is obviously of little value to this section.

remains the same — to group like solutions<sup>52</sup> together, typically with the goal of penalising those that reside in densely populated regions (the act of penalising is often meekly referred to as *fitness sharing*, since a single fitness value is generally divided amongst the members of a cluster).

While clustering and fitness sharing are popular, and often successful, approaches to maintaining population diversity, they are not without significant flaws. Most proposed clustering approaches are reliant on a statically defined sharing parameter that dictates the size of clusters, the consequence of which is that performance is contingent on the correct selection of the parameter. Specifically, when clusters are too large, narrow unexplored regions may be de-emphasised; and when clusters are too small, even identifying densely populated regions may be difficult (this is known as the bias/variance dilemma in classifier research). While guides are available for identifying suitable cluster sizes (see [140], for example), they are typically reliant on at least some *a priori* knowledge about the domain and still require the user to have input regarding the desired resolution of the optimal set.

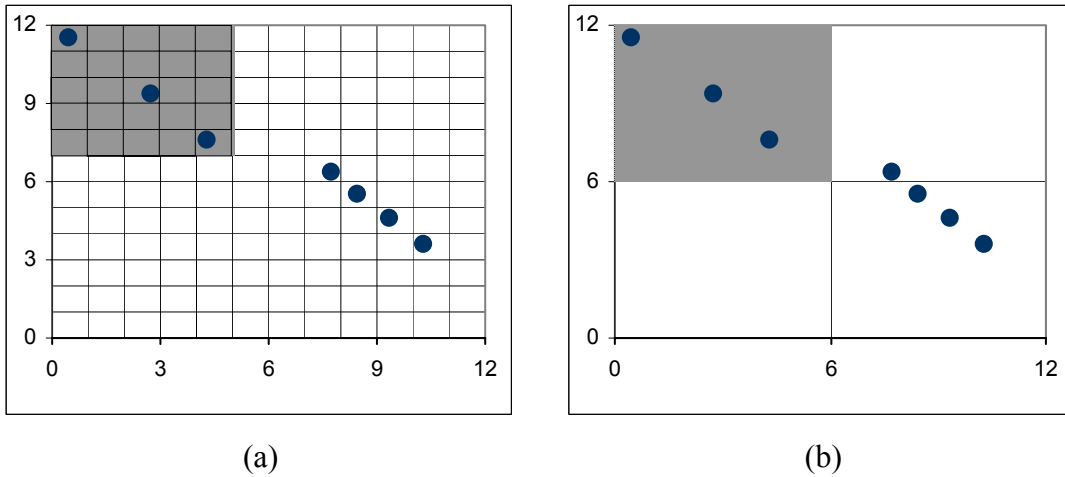
Perhaps to address these concerns, Zitzler and Thiele [137] use a dynamic, parameter-less, hierarchical clustering algorithm known as the average linkage method to maintain a diverse front. While freeing the user from the pressures of a sensitive parameter, the approach is extremely expensive (particularly when considering application in an unbounded archive), carrying a complexity burden of  $O(n^2)$  for the discovery of *only* the smallest possible cluster. If it is necessary to cluster  $c$  solutions, the cost is  $O(cn^2)$ , with a worst-case bound of  $O(n^3)$  when every solution must be clustered.

### 8.1.1.2 CELL-BASED CLUSTERING

An alternative to the traditional distance-based clustering algorithms familiar to the first generation of multiobjective optimisers is a cell-based approach (equivalent to a histogram density estimation — see [208, 209] for introductions), where the

---

<sup>52</sup> Groupings may occur in objective-space or solution-space, though the cost of comparing genotypic similarity can obviously become prohibitively expensive in high-dimensional solution spaces.



**Figure 69 — Cell-Based Crowding Inaccuracies**

The shaded region is clearly less densely populated in both (a) and (b). The high resolution of the grid in (a) means that all solutions have identical crowding. The low resolution in (b) means that the shaded region is considered the equally *most* densely populated area of the objective space. *x-axis* is objective one; *y-axis* is objective two.

objective-space is subdivided into a hyper-grid<sup>53</sup> and each solution is assigned a particular grid location (as exemplified in the Dynamic Multiobjective Evolutionary Algorithm [210], PAES [143], PESA [83] and PESA-II [84]). In this case, the crowding of a particular solution is reflected by the number of members sharing its grid location. Assuming the use of an appropriate data structure to store the grid, such as a quad-tree (as suggested by Jensen [95]), the crowding of a particular solution can be updated and retrieved in logarithmic time.

As with the distance-based clustering approaches however, there is parameter sensitivity, particularly with respect to the size of the cells. While many contemporary approaches are adaptive (adjusting cell-size as the nature of the prevailing front changes), the resolution of the grid must still be specified and the approach remains prone to poor approximations if the given resolution is inappropriate (see Figure 69 — another example of the bias/variance dilemma). Moreover, there are problems inherent in the nature of the archetypical adaptive structures suggested by Knowles and Corne [143]. Since the extent of the grid is defined by the extent of the prevailing front, any significant expansion of the front will require a complete rebuild of the cell structure — in the unbounded case, the

<sup>53</sup> Again, it is possible to form such a grid in solution-space, but it is considerably less common.

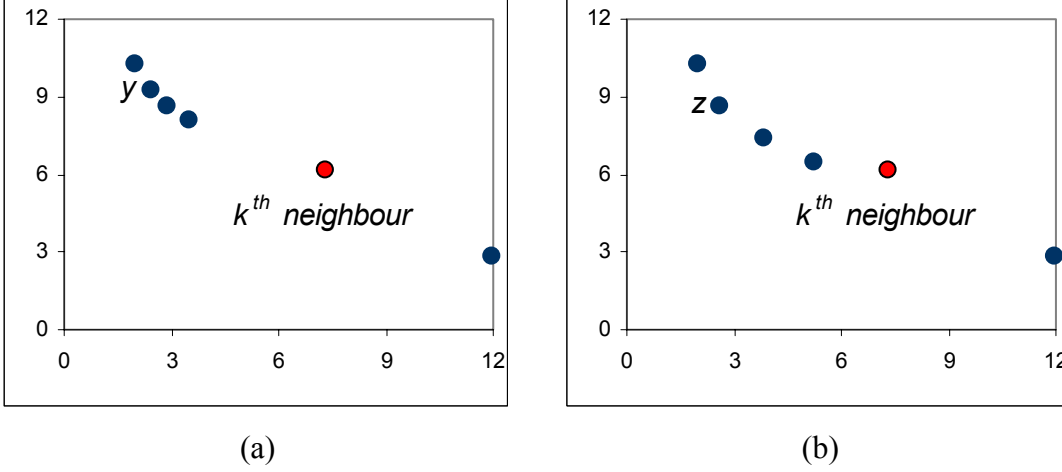
$O(n \log n)$  cost that such reconstruction entails may be prohibitive if the extent of the front expands regularly.

### 8.1.1.3 $\kappa^{\text{th}}$ NEAREST-NEIGHBOUR

In describing the SPEA2 algorithm, Zitzler *et al.* [81] propose the use of  $\kappa^{\text{th}}$  nearest-neighbour measures to approximate the crowding of a solution in objective-space. While particularly popular in fields outside of multiobjective optimisation (most notably, in classifier studies — see, for instance, [211-214]), the basic approach becomes prohibitively expensive when used with unbounded archives. For any given solution, the distance to all other elite members must be calculated, and subsequently sorted, simply to deduce the  $\kappa^{\text{th}}$  neighbour. Since this procedure carries an average time cost of  $O(n \log n)$  for each member in the archive<sup>54</sup>, the cost of providing diversity statistics for the whole front is  $O(n^2 \log n)$  if it is calculated from scratch each time. An alternative way to approach  $\kappa^{\text{th}}$  nearest-neighbours in a persistent unbounded archive is to update all members after an insertion or a deletion. While deriving the neighbourhood of an incoming proposal will still carry a cost of  $O(n \log n)$ , updating the remaining members of the set will cost  $O(un^2)$ , assuming that each member maintains a sorted neighbourhood list and that  $u$  represents the number of elements that are added to or deleted from the list. With either approach however, the bounds are likely to be prohibitive for most practical unbounded archives, and the  $O(n^2)$  storage requirements implicit in the persistent technique may be problematic if storage space is a concern.

It is also worthwhile noting that the  $\kappa^{\text{th}}$  nearest-neighbour algorithm described by Zitzler *et al.* [81] — where the crowding of a solution is governed *only* by the distance to the  $\kappa^{\text{th}}$  neighbour — is prone to producing misleading estimates. Consider Figure 70: it is clear that member  $y$  in (a) is in a more crowded region of objective-space than member  $z$  in (b), yet both solutions will yield identical  $\kappa^{\text{th}}$  nearest-neighbour scores. By emphasising only the  $\kappa^{\text{th}}$  neighbour, the method proposed by Zitzler *et al.* [81] can miss more subtle information about the true nature

<sup>54</sup> It is worth noting that the implementation of the  $\kappa^{\text{th}}$  nearest-neighbour algorithm used in this work makes use of quicksort, since it is generally considered to be amongst the fastest general-purpose sorting algorithms. Because the (unlikely) worst-case for quicksort is  $O(n^2)$ , the worst-case bound for the  $\kappa^{\text{th}}$  nearest-neighbour algorithm, in this case, is  $O(n^3)$ .



**Figure 70 —  $\kappa^{th}$  Nearest-Neighbour Inaccuracies**

y and z share identical  $\kappa^{th}$  ( $\kappa = 4$ ) nearest-neighbour scores, despite the differences in the objective-spaces in which they reside. *x-axis* is objective one; *y-axis* is objective two.

of the front, particularly with respect to local objective-space density. A simple extension that reduces such problems is averaging the distances from the examined solution to *all* of its  $\kappa$  nearest-neighbours.

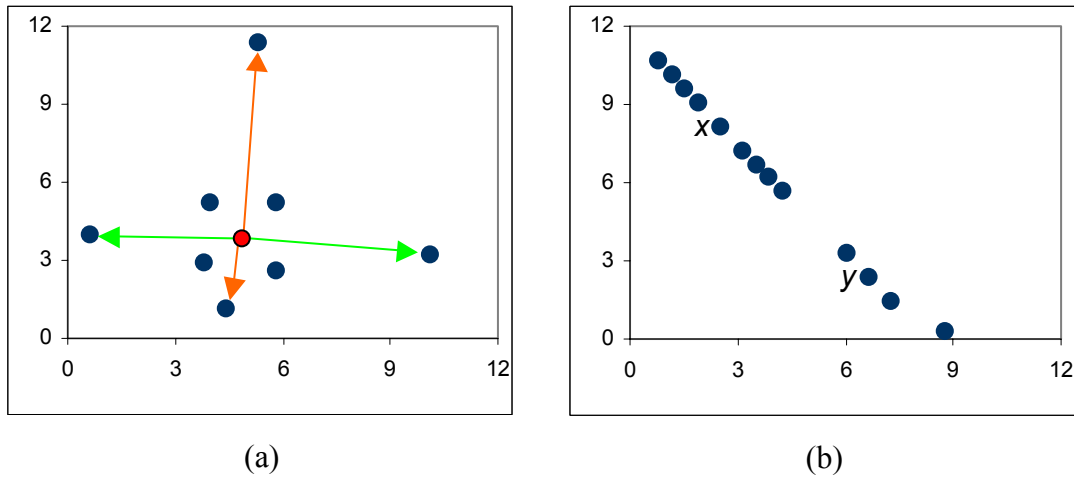
Given that there is no practical complexity burden associated with the proposed extension, this thesis posits that the averaging methodology should be favoured over the traditional approach suggested by Zitzler *et al.* [81], even in bounded archives. This is an interesting avenue of future work.

#### 8.1.1.4 CUBOID NEAREST-NEIGHBOUR CROWDING

To enhance the efficiency of crowd density estimation, Deb *et al.* [144] offer a coarse approximation derived from a simplification of existing nearest-neighbour approaches. For each objective, the solutions are ordered according to performance on that objective and dimensional-crowding is defined as the distance to preceding and succeeding elements in the ordered-list (with extreme points given an infinite score). The overall crowding of a solution (referred to here as cuboid crowding) is the average dimensional-crowding distance produced across all objectives.

The efficiency advantages of this approach are impressive. Assuming the use of a data structure that provides  $O(n \log n)$  sorting, cuboid crowding information about every member in a provided non-dominated set can be produced in  $O(\beta n \log n)$ .





**Figure 71 — Cuboid Crowding Inaccuracies**

(a) Green arrows indicate nearest cuboid neighbours in the  $x$ -dimension; orange arrows indicate nearest cuboid neighbours in the  $y$ -dimension. (b)  $x$  is in more densely occupied space than  $y$ , but the localised nature of the cuboid method means that both are given identical crowding scores.

The downside, of course, is that cuboid crowding can result in misleading density estimates. In general, the neighbours of a solution in a given dimension need not necessarily reside close to that solution in objective-space (as illustrated in Figure 71a), and so any deductions drawn from this incorrect picture are of debatable value. It is worth noting that such problems are not a concern for non-dominated sets in the two-dimensional case however, as Property One from Section 6.1.2 will hold and a correlation between proximity along a particular dimension and proximity in the complete objective-space exists<sup>55</sup>. Instead, the overriding issue in the two-dimensional case is that the consideration of only two neighbours may be insufficient to correctly gauge the nature of the objective-space surrounding a given solution. In particular, heavily explored regions may not be correctly identified due to localised pockets of low-density (as exemplified in Figure 71b), while emphasis of a sparse area may be negated by solutions that feature high localised crowding.

#### 8.1.1.5 CONCLUSIONS ABOUT EXISTING GENERIC CROWDING METRICS

While the range of crowding metrics available to contemporary multiobjective optimisation researchers is rich, they are all burdened with underlying deficiencies that inhibit their use in an unbounded environment. For instance, both distance and

<sup>55</sup> Results [77] indicate that the performance of NSGA-II degrades markedly as the number of objectives increases. The diminishing accuracy of the cuboid crowding metric on higher dimensions may be related to this finding.

cell-based clustering approaches are sensitive to parameter settings, even when they capitalise on dynamic tuning (though the effect is typically less pronounced in these cases). Alternatively, nearest-neighbour methodologies generally carry tremendous efficiency overheads, and efforts to reduce these costs produce density estimates of lower fidelity. Moreover, it is true that neither nearest-neighbour nor clustering techniques provide a definitive insight into the true nature of the objective-space. Even the improved version of the  $\kappa^{\text{th}}$  nearest-neighbour method described in Section 8.1.1.3, is susceptible to diminishing accuracy under poorly selected  $\kappa$  values and biasing, where disconnected regions and solutions residing nearer to the extremes of the front will be favoured over balanced compromises in an evenly distributed set.

Consequently, the following chapters and sections examine more efficient ways to produce crowding estimates in unbounded bi-objective elite sets (Section 8.1.2–8.1.4, Section 8.3.1 and Section 8.3.6) and how such approximations may be used in selection (Section 8.1.5 and Section 8.3.2); consider an amalgamation of both cell-based and neighbour-based methodologies (Section 8.3.3) that may aid in the production of more consistent, and generally more accurate, estimates of objective-space density; and explore the empirical impact of differing crowding mechanisms in both bounded and unbounded optimisers (Chapters 9–12).

### **8.1.2 SINGLE-NEAREST-NEIGHBOUR CROWDING IN MAK\_TREES**

Recalling that Property One (Section 6.1.2) of bi-objective non-dominated sets means that the nearest objective-space neighbour of any solution in an ordered list will always be the predecessor or successor of that solution, the single-nearest-neighbour of a member can be calculated with the same cost as locating its predecessor and successor in the list. Assuming the use of a Mak\_Tree that maintains direct links between nodes and their successors and predecessors (which can be done with no change to the  $O(\log n)$  insertion costs), deriving the nearest-neighbour scores for all members will carry an additional  $O(n)$  cost. A naïve approach would require the complete recalculation of the neighbourhood for every update to maintain member scores, though this is clearly sub-optimal, with  $O(un)$  complexity for  $u$  updates. As described in Algorithm 6, a better approach is to update only three members on insertion (the incoming node, its new successor and its

**Algorithm 6 — Updating Nearest-Neighbour Scores After Insertion** $\delta$  is a distance metric;  $dist$  refers to the current nearest-neighbour distance of a node.**Inputs:**

- $x$             The inserted node.  
 $\vec{x}$             The successor of the inserted node.  
 $\overleftarrow{x}$            The predecessor of the inserted node.

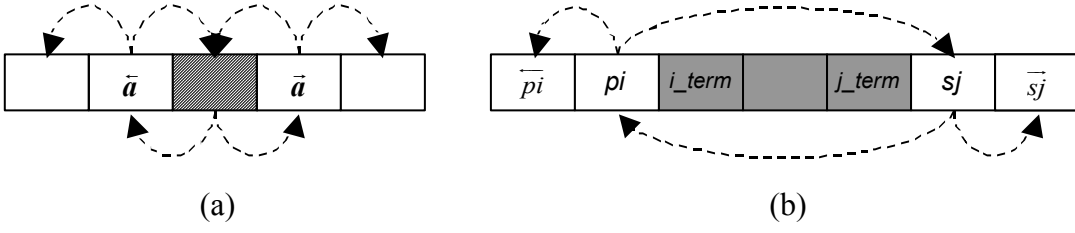
- 1: if  $\left(\delta(x, \vec{x}) < \delta(x, \overleftarrow{x})\right)$             If the distance from the inserted node to the successor is less
- 2:     $x^{dist} := \delta(x, \vec{x})$             than to the predecessor, set the successor as the nearest-neighbour.
- 3: else            Otherwise, set the predecessor as the nearest-neighbour.
- 4:     $x^{dist} := \delta(x, \overleftarrow{x})$
- 5: if  $\left(\vec{x}^{dist} > \delta(x, \vec{x})\right)$             If the incoming node is the nearest-neighbour of the
- 6:     $\vec{x}^{dist} := \delta(x, \vec{x})$             successor, update the successor accordingly.
- 7: if  $\left(\overleftarrow{x}^{dist} > \delta(x, \overleftarrow{x})\right)$             If the incoming node is the nearest-neighbour of the
- 8:     $\overleftarrow{x}^{dist} := \delta(x, \overleftarrow{x})$             predecessor, update the predecessor accordingly.

**Algorithm 7 — Updating Nearest-Neighbour Scores After Deletion** $\delta$  is a distance metric;  $dist$  refers to the current nearest-neighbour distance of a node.**Inputs:**

- $pi$             The predecessor of the first node in the dominated set.  
 $sj$             The successor of the last node in the dominated set.  
 $\overleftarrow{pi}$            The predecessor of  $pi$ .  
 $\vec{sj}$             The successor of  $sj$ .

- 1: if  $\left(\delta(pi, sj) < \delta(pi, \overleftarrow{pi})\right)$
- 2:     $pi^{dist} := \delta(pi, sj)$             The  $sj$  node is the nearest-neighbour of the  $pi$  node.
- 3: else
- 4:     $pi^{dist} := \delta(pi, \overleftarrow{pi})$             The predecessor of  $pi$  is the nearest-neighbour of the  $pi$  node.
- 5: if  $\left(\delta(sj, pi) < \delta(sj, \vec{sj})\right)$
- 6:     $sj^{dist} := \delta(sj, pi)$             The  $pi$  node is the nearest-neighbour of the  $sj$  node.
- 7: else
- 8:     $sj^{dist} := \delta(sj, \vec{sj})$             The successor of  $sj$  is the nearest-neighbour of the  $sj$  node.

new predecessor) and to capitalise on Property Three (Dominated Sets) for deletions (see Algorithm 7), so that only the predecessor of the  $i$ -terminal node and the successor of the  $j$ -terminal node need change (see Figure 72 for illustrations). With this improved methodology, keeping nearest-neighbour scores for every member of



**Figure 72 — Updating Nearest Neighbours After Insertion and Deletion**

(a) The highlighted region is the dominated set. (b) The highlighted region is the inserted node. For (a) and (b) the dashed arrows represent potential objective-space nearest-neighbours for members.

the unbounded list carries an  $O(1)$  cost per update<sup>56</sup>, irrespective of the number of members that are removed from the archive due to domination. Thus, the overall performance of the Mak\_Tree suffers no practical performance degradation when maintaining a complete and accurate set of nearest-neighbour scores.

### 8.1.3 CUBOID CROWDING IN MAK\_TREES

Maintaining cuboid crowding estimates in a Mak\_Tree is a simple modification of the methodology described for updating nearest-neighbours in Section 8.1.2. As described in Algorithm 8, the cuboid score of an incoming node is *always* derived from its predecessor ( $\vec{a}$ ) and successor ( $\vec{a}$ ), while  $\vec{a}$  and  $\vec{a}$  estimates are *always* updated according to the incoming node. Deletion (Algorithm 9) requires only the predecessor of the  $i$ -terminal and the successor of the  $j$ -terminal to be updated. As

**Algorithm 8 — Updating Cuboid Scores After Insertion**

$\delta$  is a distance metric;  $pred\_dist$  refers to the nearest preceeding neighbour distance for a given node;  $succ\_dist$  is the nearest succeeding neighbour distance.

**Inputs:**

$x$                 The inserted node.  
 $\vec{x}$                 The successor of the inserted node.  
 $\vec{x}$                 The predecessor of the inserted node.

- 1:  $x^{succ\_dist} := \delta(x, \vec{x})$                 Record distance to successor of incoming node.
- 2:  $x^{pred\_dist} := \delta(x, \vec{x})$                 Record distance to predecessor of incoming node.
- 3:  $\vec{x}^{pred\_dist} := \delta(x, \vec{x})$                 Update the successor's predecessor distance.
- 4:  $\vec{x}^{succ\_dist} := \delta(x, \vec{x})$                 Update the predecessor's successor distance.

<sup>56</sup> For all complexity analyses of nearest-neighbour techniques, it is assumed that the distance calculations carry no practical overhead. This is a reasonable assumption given that the objective-space will be strictly two-dimensional.

**Algorithm 9 — Updating Cuboid Scores After Deletion**

$\delta$  is a distance metric;  $pred\_dist$  and  $succ\_dist$  refer to the current nearest preceding and succeeding neighbour distances for the given node.

**Inputs:**

- |                       |   |
|-----------------------|---|
| $pi$                  | The predecessor of the first node in the dominated set (the $pi$ node). |
| $sj$                  | The successor of the last node in the dominated set (the $sj$ node).    |
| $\overleftarrow{pi}$  | The predecessor of $pi$ .   |
| $\overrightarrow{sj}$ | The successor of $sj$ .   |
- 1:  $pi^{succ\_dist} := \delta(pi, sj)$     The successor of  $pi$  is now  $sj$ . Update  $pi$  successor score accordingly.
  - 2:  $sj^{pred\_dist} := \delta(sj, pi)$     The predecessor of  $sj$  is now  $pi$ . Update  $sj$  predecessor score accordingly.

per Section 8.1.2, the maintenance of cuboid crowding values after insertion or deletion is an  $O(1)$  operation in the Mak\_Tree and therefore carries no practical overhead.

**8.1.4  $\kappa$  NEAREST-NEIGHBOUR CROWDING IN MAK\_TREES**

As suggested in Section 8.1.1.4, the analysis of a small number of neighbours may promote inaccurate density estimates and the use of a more rich set of members is typically preferable. As such, it is useful to generalise the techniques described in Section 8.1.2 into  $\kappa$  nearest-neighbour measures. Since the  $\kappa$  nearest-neighbours of a member will always reside in the  $\kappa$  predecessors and successors of that member (Property One), the cost of calculating all  $\kappa$  nearest-neighbour scores for all nodes is, fairly obviously,  $O(\kappa n)$  — which becomes prohibitively expensive if completed on every insertion, carrying an  $O(u\kappa n)$  cost. As with single and cuboid nearest-neighbour methods however, there exist optimisations that can considerably improve performance when crowds are estimated on a per-update basis.

**8.1.4.1.1 Updating  $\kappa$  Nearest-Neighbour Crowds After Insertion**

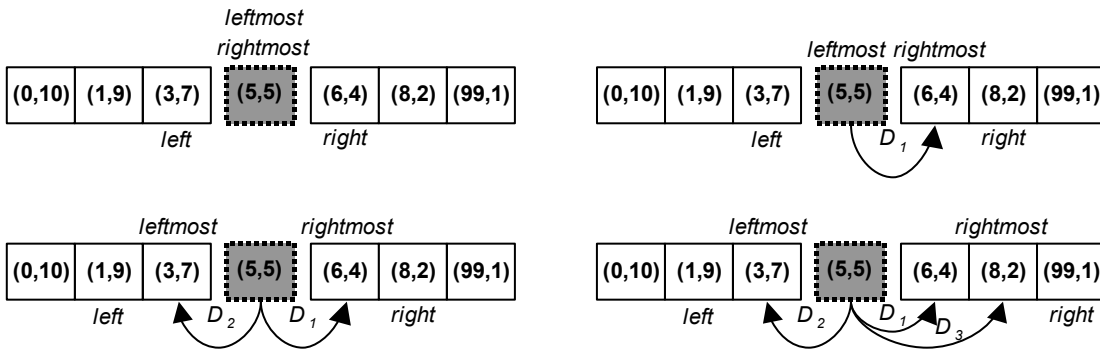
With respect to insertion it is necessary to derive the  $\kappa$  nearest-neighbours score of the incoming solution and update those members whose nearest neighbours have changed due to the insertion.

#### 8.1.4.1.2 Deriving the $\kappa$ Nearest-Neighbour Score of The New Member

The nearest-neighbour score of the incoming solution requires the investigation of precisely  $\kappa+1$  members<sup>57</sup>, starting with the in-order predecessor and successor of the solution (see Algorithm 10 and Figure 73). As the algorithm progresses, the search simply expands in the direction of the most recently identified neighbour –

**Algorithm 10 —  $\kappa$  Nearest-Neighbours Insertion: Updating the New Node**

<b>Inputs:</b>	
$x$	The inserted node.
1: let $left := \bar{x}; right := \bar{x}; D := \emptyset$	$D$ is the collection of nearest-neighbour distances.
2: let $x^{leftmostNeighbour} := \text{null}; x^{rightmostNeighbour} := \text{null}$	Leftmost and rightmost neighbours define the $\kappa$ neighbourhood of $x$ (initially undefined).
3: for ( $i = 1$ to $\kappa$ )	(Note: if $n < \kappa$ , let $\kappa = n$ )
4: if $((right \neq \text{null}) \vee (left \neq \text{null}))$	If there are archived nodes to examine,
5: if $(\delta(x, left) < \delta(x, right))$	if $left$ node is nearer to $x$ than the $right$
6: $D_i := \delta(x, left)$	then note the distance to the $left$ node,
7: $x^{leftmostNeighbour} := left$	set $left$ as the leftmost neighbour of $x$
8: $left := \overleftarrow{left}$	and shift the $left$ node.
9: else	If the $right$ node is nearer to $x$ than $left$
10: $D_i := \delta(x, right)$	then note the distance to the $right$ node,
11: $x^{rightmostNeighbour} := right$	set $right$ as the new rightmost neighbour
12: $right := \overrightarrow{right}$	of $x$ and shift the $right$ node.
13: calculateKNNScore( $x, D$ )	Assign $x$ a crowding score.
14: updateNeighbourScores( $x$ )	Update predecessors and successors of $x$ .



**Figure 73 — Updating the Inserted Node for  $\kappa$  Nearest-Neighbours Crowding**

Illustrates an example execution of Algorithm 10 for  $\kappa=3$ . The shaded regions represent the inserted node and arrows depict nearest neighbours discovered thus far. Note that the leftmost and rightmost boundaries of the neighbourhood are recorded for future reference.

<sup>57</sup> In general, the complexity analysis of  $\kappa$  nearest-neighbour crowding assumes the existence of at least  $\kappa$  members. If  $n < \kappa$ , then  $n$  should replace  $\kappa$  in all big-oh complexity estimates.

Algorithm 11 —  $\kappa$  Nearest-Neighbours Insertion: Updating Neighbours**Inputs:** $x$ 

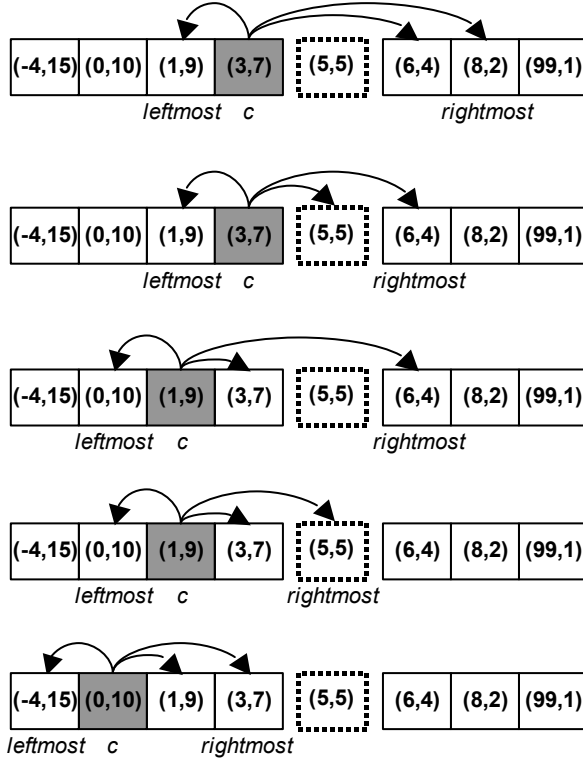
The inserted node.

1: let $c := \bar{x}$	Start updating at predecessor of $x$ .
2: while $\left( \left( \delta(c, c^{\text{rightmostNeighbour}}) \geq \delta(c, x) \right) \vee \left( \delta(c, c^{\text{leftmostNeighbour}}) \geq \delta(c, x) \right) \right)$	While the inserted solution is closer than either of the $c$ node's extreme neighbours
3: if $\left( \delta(c, c^{\text{rightmostNeighbour}}) < \delta(c, c^{\text{leftmostNeighbour}}) \right)$	Remove the furthest of the
4:   updateKNNScore( $c, x, c^{\text{leftmostNeighbour}}$ )	extreme neighbours from $c$ 's
5: $c^{\text{leftmostNeighbour}} := \text{successor}(c^{\text{leftmostNeighbour}})$	neighbourhood and update the
6: else	crowding estimate accordingly.
7:   updateKNNScore( $c, x, c^{\text{rightmostNeighbour}}$ )	
8: $c^{\text{rightmostNeighbour}} := \text{predecessor}(c^{\text{rightmostNeighbour}})$	Move to the next node to see if
9: $c := \bar{c}$	it also requires updating.
10: let $c := \bar{x}$	Now check the successors of the
11: while $\left( \left( \delta(c, c^{\text{rightmostNeighbour}}) \geq \delta(c, x) \right) \vee \left( \delta(c, c^{\text{leftmostNeighbour}}) \geq \delta(c, x) \right) \right)$	inserted solution, using a
12: if $\left( \delta(c, c^{\text{leftmostNeighbour}}) < \delta(c, c^{\text{rightmostNeighbour}}) \right)$	symmetrical method of lines 2-9.
13:   updateKNNScore( $c, x, c^{\text{rightmostNeighbour}}$ )	
14: $c^{\text{rightmostNeighbour}} := \text{predecessor}(c^{\text{rightmostNeighbour}})$	
15: else	
16:   updateKNNScore( $c, x, c^{\text{leftmostNeighbour}}$ )	
17: $c^{\text{leftmostNeighbour}} := \text{successor}(c^{\text{leftmostNeighbour}})$	
18: $c := \bar{c}$	

incrementally moving left for every selected neighbour that precedes the new member and right for neighbours that succeed. Given the use of suitably inexpensive distance metrics and an efficient `calculateKNNScore` function, this simple technique, as used in one-dimensional  $\kappa$  nearest-neighbour estimates, carries a cost of  $O(\kappa)$ .

#### 8.1.4.1.3 Updating The $\kappa$ Nearest-Neighbour Scores of Remaining Members

When a new proposal is inserted into the archive, Property One implies that at most  $\kappa$  preceding and  $\kappa$  succeeding members may have affected neighbourhoods and require updating. Specifically, the algorithm (as described in Algorithm 11 and illustrated in



Updating (3,7). Check whether either boundary node is further away than the inserted node. If so, remove the furthest boundary solution from the neighbourhood of  $c$  and update.

The rightmost boundary node is updated by shifting it one place to the left. The inserted solution is included in the neighbourhood.

Updating (1,9). Check whether either boundary node of  $c$  is further away than the inserted node. If so, remove the furthest boundary solution and update.

The rightmost boundary node is updated by shifting it one place to the left. The inserted solution is included in the neighbourhood.

Updating (1,9). Check whether either boundary node of  $c$  is further away than the inserted node. Neither are, so stop.

**Figure 74 — Updating the Neighbourhood for  $\kappa$  Nearest-Neighbours Crowding After Insertion**

Dashed region is the inserted node; arrows indicate current neighbours; shaded region is the node being updated. For brevity, the figure illustrates left updates only.

Figure 74) proceeds by adjusting the neighbourhoods of each of the  $\kappa$  preceding members of the newly inserted node  $x$  (lines 3-8 in Algorithm 11), until  $x$  is too far away to warrant inclusion in a predecessor's neighbourhood (the condition at line 2 in Algorithm 11 becomes false). A symmetrical procedure is used to correctly update the members lying to the right of the new proposal in the ordered list (see lines 10-17 in Algorithm 11).

With an efficient algorithm in-place, and again assuming suitably inexpensive distance and `updateKNNScore` procedures, the worst-case time costs of updating the neighbourhoods of all preceding and succeeding members is only  $O(\kappa)$ .

#### 8.1.4.2 UPDATING $\kappa$ NEAREST-NEIGHBOUR CROWDS AFTER DELETION

Maintaining nearest-neighbour scores after deletion is a somewhat more complicated affair than after insertion, as it is necessary to consider the impact of losing multiple neighbours. Still, the special properties of bi-objective sets can be capitalised upon to produce efficient results. In particular, recall that the deleted set will occupy a



contiguous region in the ordered list (Property Three) and that the  $\kappa$  nearest-neighbours of any solution must come from the  $\kappa$  successors and  $\kappa$  predecessors of that solution (Property One). As such, at most  $\kappa$  predecessors of the first dominated member ( $i$ -terminal node) and  $\kappa$  successors of the last dominated node ( $j$ -terminal node) will require neighbourhood updating (see Algorithm 12 and Figure 75). In the worst case, when all members of the deleted set are original members of the neighbourhood, the first predecessor of the dominated set will require  $\kappa$  comparisons to generate an updated crowding score since a replacement neighbour for each deleted member is required. The second predecessor will require at most  $\kappa-1$  comparisons; the third,  $\kappa-2$ ; and so on. Thus, the worst-case complexity is  $O(\kappa+\kappa-1+\kappa-2+\dots+1) = O(\kappa^2)$ . The procedure for updating successors of the deleted set is symmetrical and the overall worst-case complexity must therefore be  $O(\kappa^2)$ . Furthermore, if the number of nodes residing in the dominated set is less than  $\kappa$ , the worst-case complexity reduces to  $O(\kappa d)$ , where  $d$  is the number of members that are to be removed from the list.

### 8.1.4.3 THE COMPLEXITY OF MAINTAINING $\kappa$ NEAREST-NEIGHBOURS

The worst-case time costs for maintaining a complete and accurate set of nearest-neighbour estimates after the insertion of a non-dominating new proposal is  $O(\kappa)$ . When a dominating solution is to be included, the need to update multiple solutions with multiple new neighbours leads to a worst-case bound of  $O(\kappa^2)$  when  $\kappa < d$  and  $O(\kappa d)$  otherwise. Given that a generic approach, akin to that of the  $\kappa^{\text{th}}$  nearest-neighbour algorithm employed by Zitzler *et al.* [81], will carry a burden of  $O(n^2)$  [95]

---

#### Algorithm 12 — $\kappa$ Nearest-Neighbours Deletion

---

**Inputs:**

$i\_term$	The left-most terminal of the dominated (delete) set.
$j\_term$	The right-most terminal of the dominated (delete) set.

- |   |                             |
|---|-----------------------------|
| 1: let $\overleftarrow{left} := \overleftarrow{i\_term}$ ; $\overrightarrow{right} := \overrightarrow{j\_term}$                 | Update all nodes before     |
| 2: while $((\overleftarrow{left} \neq \text{null}) \wedge (\text{deleteFromRight}(\overleftarrow{left}, i\_term, j\_term)))$    | dominated set with          |
| 3: $\overleftarrow{left} := \overleftarrow{\overleftarrow{left}}$   | affected neighbourhoods.    |
| 4: while $((\overrightarrow{right} \neq \text{null}) \wedge (\text{deleteFromLeft}(\overrightarrow{right}, i\_term, j\_term)))$ | Update nodes after set with |
| 5: $\overrightarrow{right} := \overrightarrow{\overrightarrow{right}}$  | affected neighbourhoods.    |
-

**Algorithm 13 —  $\kappa$  Nearest-Neighbours DeleteFromRight (Helper Algorithm)**

DeleteFromLeft is a symmetrical form of this algorithm.

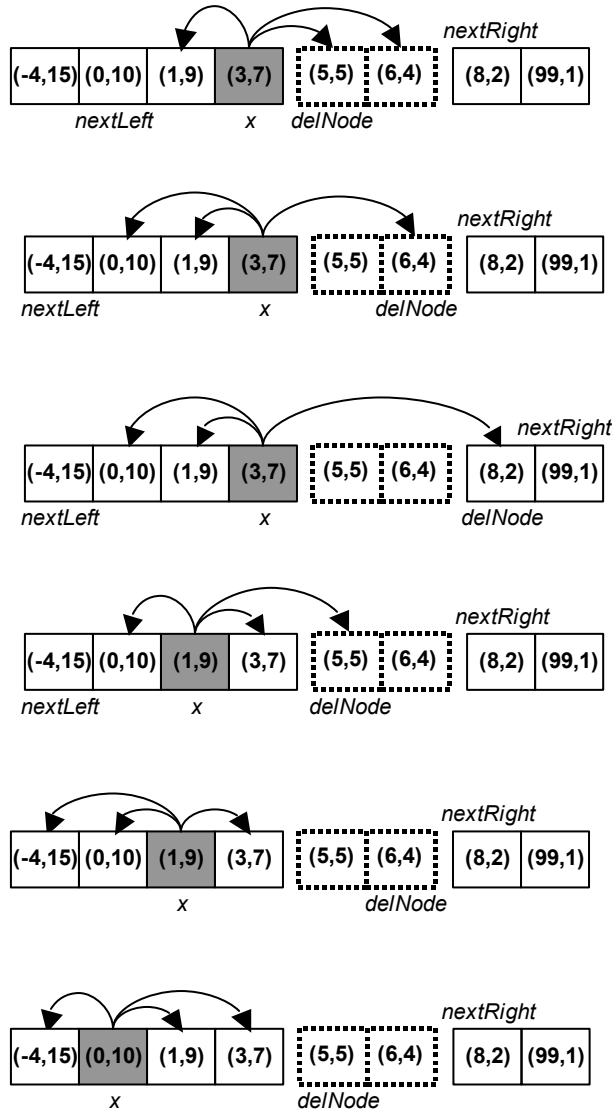
**Inputs:**

$x$                                       The node being tested for neighbourhood updating.  
 $i\_term$                                 The left-most terminal of the dominated (delete) set.  
 $j\_term$                                 The right-most terminal of the dominated (delete) set.

```

let  $delNode := i\_term$ ;  $end := successor(x^{rightmostNeighbour})$ ;
1:   $nextLeft := predecessor(x^{leftmostNeighbour})$ 
2:  if  $(\delta(x, x^{rightmostNeighbour}) < \delta(x, i\_term))$ 
3:    return false
4:  else
5:    if  $(\delta(x, x^{rightmostNeighbour}) \leq \delta(x, j\_term))$ 
6:       $nextRight := j\_term$ 
7:    else
8:       $nextRight := successor(x^{rightmostNeighbour})$ 
9:    while  $((delNode \neq j\_term) \wedge (delNode \neq end))$ 
10:   if  $((nextLeft = null) \wedge (nextRight = null))$ 
11:     updateKNNRemove( $x, delNode$ )
12:     if  $(delNode = x^{rightmostNeighbour})$ 
13:        $x^{rightmostNeighbour} := i\_term$ 
14:   else if  $(\delta(x, nextLeft) < \delta(x, nextRight))$ 
15:     updateKNNScore( $x, nextLeft, delNode$ )
16:      $x^{leftmostNeighbour} := nextLeft$ 
17:      $nextLeft = \overleftarrow{nextLeft}$ 
18:     if  $(delNode = x^{rightmostNeighbour})$ 
19:        $x^{rightmostNeighbour} := i\_term$ 
20:   else
21:     updateKNNScore( $x, nextRight, delNode$ )
22:      $x^{rightmostNeighbour} := nextRight$ 
23:      $nextRight = \overrightarrow{nextRight}$ 
24:     if  $(nextRight = i\_term)$ 
25:        $nextRight := j\_term$ 
26:    $delNode = \overrightarrow{delNode}$ 
27: return true
    
```

If this node is unaffected by the deletion, return false.  
 Ensure that the next rightmost candidate is not a deleted node.  
 While  $delNode$  is in a deleted part of  $x$ 's neighbourhood  
 If there are no more candidates, simply remove  $delNode$  from  $x$ 's neighbourhood.  
 Otherwise, update the neighbourhood of  $x$  with the next nearest candidate node ( $nextLeft$  or  $nextRight$ ) and supply a new candidate for the next iteration (by updating  $nextLeft$  or  $nextRight$ ).  
 Examine the next deleted node.  
 Return: neighbourhood updated.



**Figure 75 — Updating Neighbourhoods for  $\kappa$  Nearest-Neighbours Crowding after Deletion**

Dashed region is the deleted (dominated) set; arrows indicate current neighbours; shaded region is the node being updated. For brevity, the figure illustrates left updates only.

Updating the neighbourhood of  $(3,7)$ . Check if  $delNode$  is to be deleted and that it is in the neighbourhood of  $x$ .

Select the next nearest (non-dominated) neighbour for inclusion in the neighbourhood of  $x$  and remove the deleted neighbour. Move  $delNode$  to the right and confirm that it is in the deleted set and a member of the  $x$  neighbourhood.

Select the next nearest (non-dominated) neighbour for inclusion in the neighbourhood of  $x$  and remove the deleted neighbour. After moving the  $delNode$  to the right it lies outside the range of the deleted set — it must be reset.

Reset  $delNode$  to  $(5,5)$  and move  $x$  to the left. Now updating the neighbourhood of  $(1,9)$ . Check if  $delNode$  is to be deleted and that it is in the neighbourhood of  $x$ .

Select the next nearest (non-dominated) neighbour for inclusion in the neighbourhood of  $x$  and remove the deleted neighbour. After moving the  $delNode$  to the right it lies outside the range of the  $x$  neighbourhood — it must be reset.

Reset  $delNode$  to  $(5,5)$  and move  $x$  to the left.  $delNode$  is not a member of the  $x$  neighbourhood, so stop.

and that typically  $\kappa \ll n$ , the difference in performance is massive<sup>58</sup> — particularly in unbounded archives, where potentially large values of  $n$  will essentially preclude the use of generic techniques in practice. The reason for the disparity is the correlation between proximity in the archival list and proximity in objective-space. While Property One ensures that neighbours in the specialised bi-objective non-dominated list are neighbours in objective-space, there is no way to efficiently

<sup>58</sup> As an illustration, Zitzler *et al.* [81] recommend a value of  $k = \sqrt{n}$ .

construct a list that features a similar correlation in the generic case and it is therefore difficult to avoid expensive searches of the entire population.

It is also worth noting that the list members described in Algorithm 10, Algorithm 11 and Algorithm 12 are identical to standard `Mak_Tree` nodes, but for the need to store a neighbourhood score and references to their leftmost and rightmost neighbours. Consequently, this extended `Mak_Tree` carries no increase in big-oh storage complexity over the basic structure described in Chapter 6.

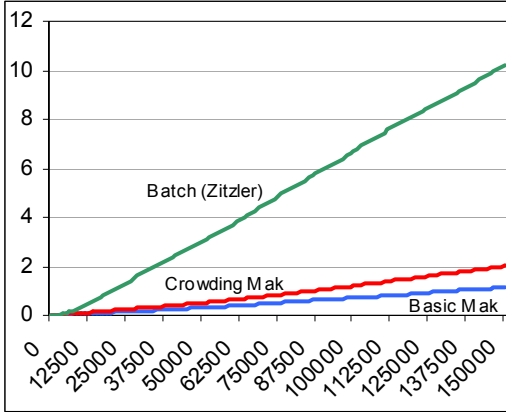
#### 8.1.4.4 EMPIRICAL RESULTS

While the theoretical performance edge generated by bi-objective specialisation of density estimation is impressive, it is not until practical performance analysis that the ramifications of such gains can be properly observed. As such, the extended `Mak_Tree` is compared with a range of contemporary generic approaches on the same diverse data sets described in Section 6.4.2 (produced by NSGA-II runs with settings defined in Appendix B.1.1). Specifically, the simple two-neighbour cuboid function suggested by Deb *et al.* [82] and the  $\kappa^{\text{th}}$  nearest-neighbour algorithm explored by Zitzler *et al.* [81] are both used for comparative purposes — with truncation thresholds of 50 employed for both batch and incremental processing, as per Section 6.4.2.2.3. Since the goal is to maintain a complete and accurate set of nearest-neighbour scores for the non-dominated archive, each generic technique updates the estimates after a successful insertion (when incremental procedures are used) or otherwise after a complete batch has been processed. The extended `Mak_Tree` itself uses the averaged  $\kappa$  neighbours approach proposed in Section 8.1.1.3 and is updated after every successful insertion such that all members of the unbounded non-dominated list maintain correct density estimates. All results are derived from the average time-costs of processing twenty data sets per-problem.

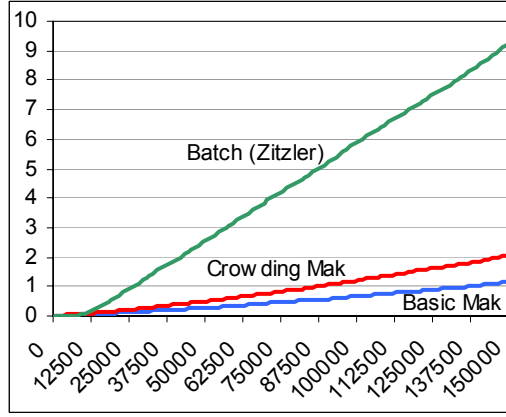
The results illustrated in Figure 76, Figure 77, Figure 78 and Table 17 (see pages 198–201) are remarkable. The unbounded `Mak_Tree` successfully stores a complete set of density estimates at a cost that is markedly lower than with traditional  $\kappa^{\text{th}}$  nearest-neighbour approaches in a heavily truncated environment — even when both the size of the unbounded list is high and numerous neighbours are considered (as in *AP-10* and *AP-5*). Perhaps even more impressively, the extended `Mak_Tree` is competitive with, and frequently better than (when  $\kappa = 5$ ), amongst the simplest of all

available distance-based crowding estimators — the two-neighbour cuboid function operating on a tightly bound list.

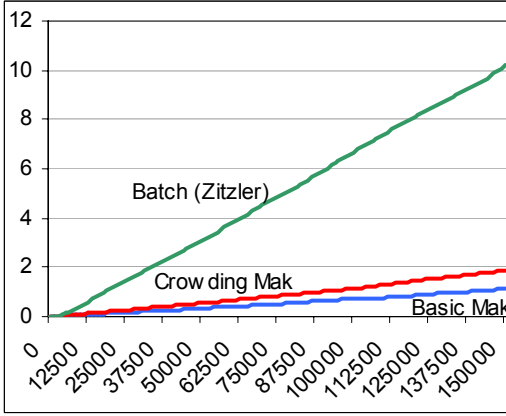
The implications are profound. Given that generic density estimators operating on truncated elite sets are prone to both considerable inaccuracy (see Section 6.2) and are inherently costly, the development of a procedure which simultaneously addresses both of these problems is pivotal to improving the performance of optimisation algorithms, both with respect to efficiency and efficacy. Since crowding estimates consider the complete non-dominated set in the extended Mak\_Tree and results indicate that the procedure is less expensive than contemporary truncated approaches, the extended Mak\_Tree may rest as the key to improved algorithmic performance in bi-objective domains.



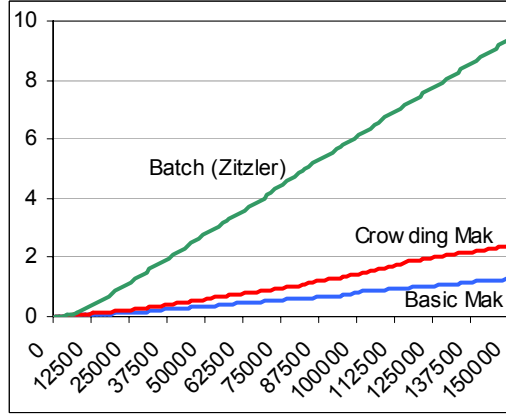
(a) AP-1



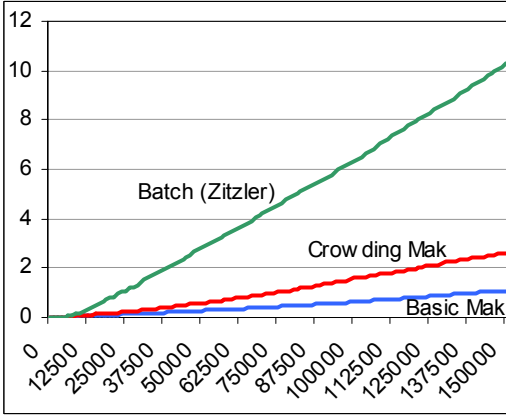
(b) AP-2



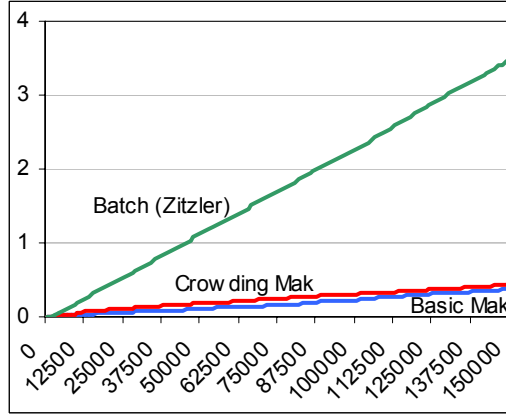
(c) AP-3



(d) AP-4



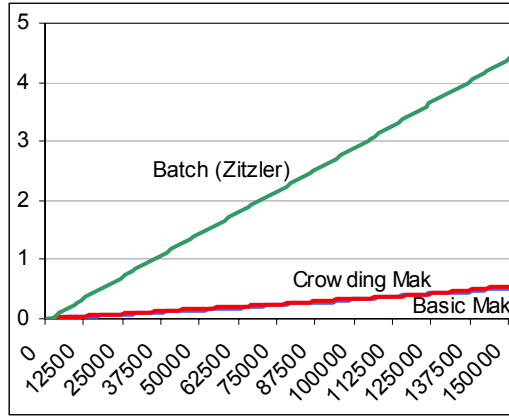
(e) AP-5



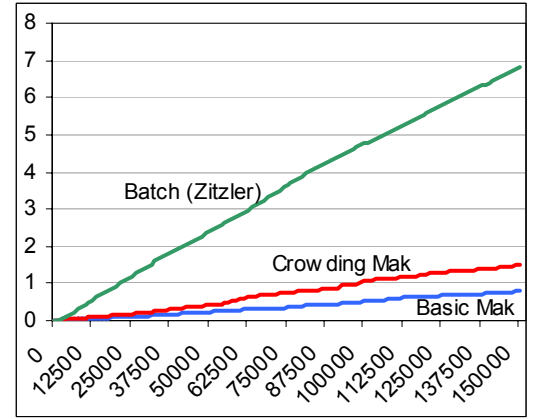
(f) AP-8

**Figure 76 — Average Cumulative Time Costs of Unbounded and Bounded Archiving Techniques With Complete Density Estimates ( $\kappa = 20$ )**

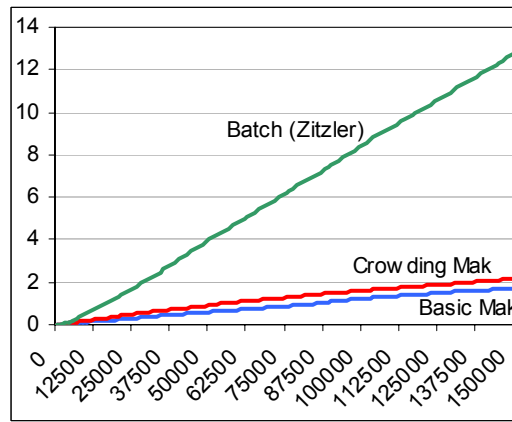
$x$ -axis represents the number of solutions presented to the archive;  $y$ -axis is the average cumulative processing time in seconds. The Zitzler results refer to  $\kappa^{\text{th}}$  neighbour performance in a batch-processed truncated archive of 50. The Basic Mak results feature no crowding measures and are intended as a base-line.



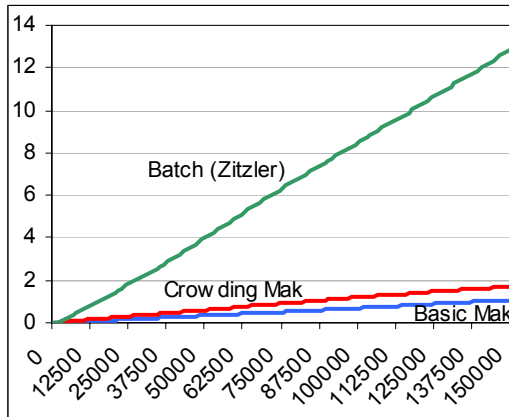
(a) AP-9



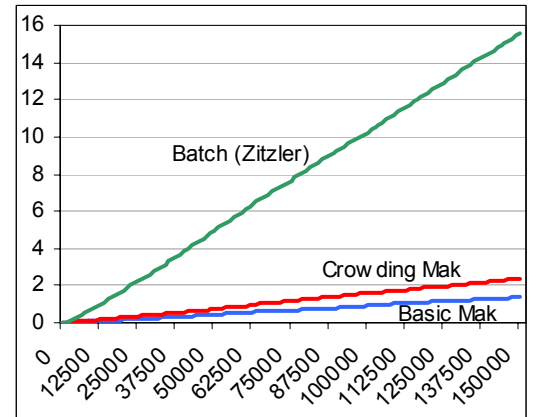
(b) AP-10



(d) AP-15



(e) AP-16



(f) AP-17

**Figure 77 — Average Cumulative Time Costs of Unbounded and Bounded Archiving Techniques With Complete Density Estimates ( $\kappa = 20$ )**

$x$ -axis represents the number of solutions presented to the archive;  $y$ -axis is the average cumulative processing time in seconds. The Zitzler results refer to  $\kappa^{\text{th}}$  neighbour performance in a batch-processed truncated archive of 50. The Basic Mak results feature no crowding measures and are intended as a base-line.



**Figure 78 — Average Cumulative Time Costs of Unbounded and Bounded Archiving Techniques With Complete Density Estimates ( $\kappa = 20$ )**

*x-axis* represents the number of solutions presented to the archive; *y-axis* is the average cumulative processing time in seconds. The Zitzler results refer to  $\kappa^{\text{th}}$  neighbour performance in a batch-processed truncated archive of 50. The Basic Mak results feature no crowding measures and are intended as a base-line.

**Table 17 — Time-Based Performance of Archives with Complete Density Estimates**

*B* indicates batch processing; *I* is incremental processing; *Zitz* is the  $\kappa^{\text{th}}$  neighbour approach as used by Zitzler *et al.* [81]; *Deb* is the cuboid approach used by Deb *et al.* [82]; time presented in seconds.

	Mak K=0	Mak K=5	Zit (I) K=5	Zit (B) K=5	Mak K=20	Zitz (I) K=20	Zitz (B) K=20	Deb (I)	Deb (B)	
<b>AP-1</b>	0.06	0.08	3.40	0.28	0.09	3.45	0.34	0.14	0.14	10,000 Evaluations
<b>AP-2</b>	0.05	0.06	0.47	0.07	0.07	0.47	0.08	0.05	0.05	
<b>AP-3</b>	0.07	0.07	4.14	0.37	0.11	4.30	0.44	0.14	0.16	
<b>AP-4</b>	0.05	0.07	1.82	0.21	0.07	1.75	0.27	0.09	0.08	
<b>AP-5</b>	0.04	0.06	1.53	0.17	0.07	1.42	0.19	0.08	0.07	
<b>AP-8</b>	0.03	0.03	1.44	0.16	0.06	1.67	0.18	0.06	0.06	
<b>AP-9</b>	0.02	0.05	2.21	0.23	0.04	2.63	0.28	0.07	0.06	
<b>AP-10</b>	0.04	0.05	4.71	0.37	0.07	4.55	0.43	0.13	0.12	
<b>AP-15</b>	0.06	0.08	6.66	0.51	0.13	6.56	0.56	0.19	0.16	
<b>AP-16</b>	0.05	0.10	7.67	0.60	0.14	7.64	0.66	0.21	0.18	
<b>AP-17</b>	0.07	0.12	8.26	0.63	0.14	7.77	0.75	0.26	0.22	
<b>AP-21</b>	0.05	0.09	6.77	0.56	0.12	6.60	0.60	0.19	0.15	
<b>F-1</b>	0.05	0.06	3.07	0.30	0.08	3.73	0.34	0.13	0.12	50,000 Evaluations
<b>AP-1</b>	0.32	0.46	34.07	2.52	0.51	38.69	3.03	0.97	0.85	
<b>AP-2</b>	0.28	0.45	25.26	2.09	0.50	30.15	2.57	0.78	0.67	
<b>AP-3</b>	0.34	0.41	35.11	2.80	0.55	40.50	3.10	0.93	1.09	
<b>AP-4</b>	0.34	0.49	24.08	2.38	0.59	24.78	2.82	0.80	0.75	
<b>AP-5</b>	0.28	0.44	27.99	2.56	0.58	25.29	2.82	0.97	0.86	
<b>AP-8</b>	0.11	0.16	9.24	0.93	0.18	11.48	1.12	0.29	0.31	
<b>AP-9</b>	0.15	0.21	11.74	1.18	0.17	17.01	1.46	0.38	0.38	
<b>AP-10</b>	0.24	0.36	28.63	2.19	0.42	29.11	2.40	0.75	0.68	
<b>AP-15</b>	0.34	0.48	45.14	3.36	0.60	43.82	4.00	1.16	0.96	
<b>AP-16</b>	0.35	0.46	47.19	3.70	0.60	45.24	4.05	1.21	1.08	
<b>AP-17</b>	0.44	0.62	52.54	4.07	0.75	47.23	4.97	1.44	1.18	
<b>AP-21</b>	0.34	0.44	47.03	3.57	0.56	44.87	4.01	1.23	1.05	
<b>F-1</b>	0.27	0.33	24.69	2.26	0.38	36.92	2.72	0.76	0.72	



**Table 18 — Time-Based Performance of Archives with Complete Density Estimates**

*B* indicates batch processing; *I* is incremental processing; *Zitz* is the  $\kappa^{\text{th}}$  neighbour approach as used by Zitzler *et al.* [81]; *Deb* is the cuboid approach used by Deb *et al.* [82]; time measured in seconds.

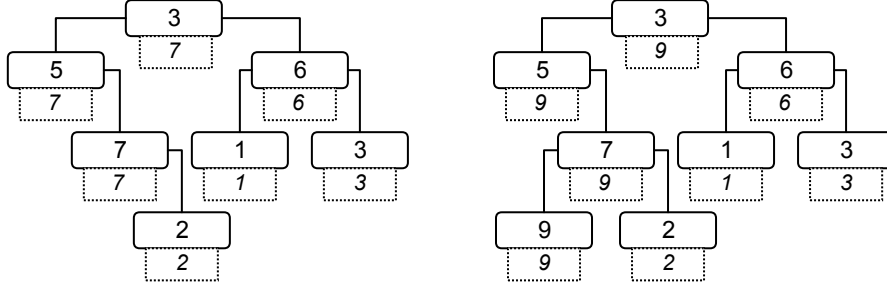
	<b>Mak K=0</b>	<b>Mak K=5</b>	<b>Zit (I) K=5</b>	<b>Zit (B) K=5</b>	<b>Mak K=20</b>	<b>Zitz (I) K=20</b>	<b>Zitz (B) K=20</b>	<b>Deb (I)</b>	<b>Deb (B)</b>	<b>150,000 Evaluations</b>
<b>AP-1</b>	1.15	1.76	113.36	8.61	2.01	132.35	10.20	2.98	2.64	
<b>AP-2</b>	1.14	1.59	93.02	7.61	2.04	115.99	9.27	2.66	2.36	
<b>AP-3</b>	1.12	1.53	114.47	8.62	1.87	134.20	10.18	3.05	2.97	
<b>AP-4</b>	1.29	1.89	83.83	8.41	2.40	85.84	9.41	2.68	2.49	
<b>AP-5</b>	1.09	1.88	98.88	9.16	2.60	88.14	10.27	3.29	2.85	
<b>AP-8</b>	0.37	0.51	30.89	2.71	0.45	40.21	3.48	0.91	0.98	
<b>AP-9</b>	0.51	0.66	36.39	3.67	0.55	55.75	4.41	1.22	1.13	
<b>AP-10</b>	0.81	1.08	83.56	6.32	1.50	88.25	6.80	2.30	1.95	
<b>AP-15</b>	1.01	1.38	142.60	10.92	1.71	140.49	12.76	3.58	3.18	
<b>AP-16</b>	1.08	1.56	148.21	11.86	1.72	144.00	12.91	3.77	3.41	
<b>AP-17</b>	1.38	2.09	163.41	13.61	2.39	145.65	15.55	4.40	3.84	
<b>AP-21</b>	0.98	1.35	147.85	11.49	1.65	142.49	13.13	3.80	3.35	
<b>F-1</b>	0.84	1.07	76.80	7.23	1.06	126.05	8.80	2.32	2.28	

### 8.1.5 LOCATING ISOLATED SOLUTIONS IN THE ARCHIVE

While maintaining a complete set of nearest-neighbour scores for the entire unbounded archive is a powerful utility, it does not guarantee an efficient mechanism for the identification and recovery of isolated (or crowded<sup>59</sup>) solutions residing in the store. Indeed, a naïve approach will require the exploration of the entire set simply to locate the single most isolated node — thus carrying a prohibitive  $O(n)$  cost per retrieved element. Given that many contemporary multiobjective optimisation algorithms must frequently identify the least (or most) crowded solutions in the archival population (see, for instance, NSGA-II [82, 175], SPEA [137] and SPEA2 [81, 145]), discovering a more efficient approach is therefore a valuable endeavour.

As proposed by Jensen [95], a suitable alternative to the naïve exploration-based approach, is to annotate the nodes of a (balanced) tree-based structure. Specifically, for any given node  $d$ , the annotation should indicate the largest neighbourhood score of all members belonging to the sub-tree rooted at  $d$  (as illustrated in Figure 79). With the annotations in place, locating the least-crowded node is a simple matter of following the annotations until arriving at the isolated solution — an  $O(\log n)$  search.

<sup>59</sup> The procedures described herein for locating isolated solutions in an elite archival set are analogous to those required for locating crowded members.



**Figure 79 — Example Tree Annotations**

Dashed cells represent crowding annotations; larger cells represent crowding scores.

However, generic annotations have a problem. While searching is efficient, maintaining correct annotations can be costly. Whenever the neighbourhood scores of a member change, a new solution is inserted or an element is deleted,  $O(\log n)$  (at worst) annotations must be updated, as the change flows up through the tree. Thus, the insertion of a dominating solution that leads to the deletion of  $d$  members and results in  $s$  score changes will result in  $O(s \log n + d \log n)$  annotation updates. By specialising annotations for use in Mak\_Trees, the unique properties of non-dominated bi-objective sets and binary trees can be combined to form a considerably more efficient procedure.

### 8.1.5.1 ANNOTATING THE MAK\_TREE

#### 8.1.5.1.1 Annotating The Mak\_Tree Without Deletion

When examining the techniques required to efficiently produce accurate annotations in Mak\_Trees, it is beneficial to first consider incoming solutions that are non-dominating, since deletion complicates proceedings. As illustrated in Section 8.1.4.1.3, assuming that the incoming solution is non-dominating, unique<sup>60</sup> and accepted into the unbounded archive, at most  $2\kappa+1$  nodes (the incoming node and the  $\kappa$  successors and  $\kappa$  predecessors of that node) will have updated crowding scores. Intuitively, the easiest way to update the scores of the altered nodes is to simply push each change up the tree, as in the generic case, yielding a worst-case cost of  $O(\kappa \log n)$ . However, since the updated nodes will *always* lie in a contiguous block (by virtue of Property One), the range-related characteristics of a binary tree (see

<sup>60</sup> Unique in the sense that no other equivalent solution is already stored in the set.

## Algorithm 14 — Updating Crowding Annotations After Insertion

**Inputs:**

$l$	The leftmost updated node.
$r$	The rightmost updated node.
1: let $current := l$	Start at the leftmost updated node.
2: while ( $current \neq null$ )	While there are more nodes to consider.
3: $rightChild := current^{rightChild}$	
4: if $\left( \begin{array}{l} (rightChild \neq null) \wedge \\ (rightChild^{obj2\_label} \leq l^{obj2\_label}) \end{array} \right)$	If right child has been updated then make sure the sub-tree is correctly annotated.
5: $annotateSubTree(rightChild, l, r)$	
6: $updateAnnotation(current)$	Update the annotation of the current node.
7: $leaveBreadCrumb(current)$	Note that the node has been visited.
8: $current := current^{parent}$	Move up the tree.
9: let $current := r$	Start at the rightmost updated node.
10: while ( $(current \neq null) \wedge (noBreadCrumb(current))$ )	While there are more unvisited nodes.
11: $leftChild := current^{leftChild}$	
12: if $\left( \begin{array}{l} (leftChild \neq null) \wedge \\ (leftChild^{obj1\_label} \geq l^{obj1\_label}) \end{array} \right)$	If left child has been updated then make sure the sub-tree is correctly annotated.
13: $annotateSubTree(leftChild, l, r)$	
14: $current^{annotation} := \text{best} \left( \begin{array}{l} knnScore(current^{knnScore}), \\ annotation(left), \\ annotation(right) \end{array} \right)$	Update the annotation of the current node — any value flagged with ignore is disregarded.
15: $current := current^{parent}$	Move up the tree.

Section 0) can be capitalised upon to improve performance. As illustrated in Algorithm 14 and Figure 80, the resulting annotation procedure is relatively straightforward: start at the leftmost updated node (at worst, the  $\kappa^{\text{th}}$  predecessor of the inserted node) and progress upwards through the tree (along the *primary path*) as in the generic case; if the right child of the current node has been updated, recursively adjust the annotations for the corresponding sub-tree (in the *secondary path*), and then continue as normal. Once this initial parse is complete, a symmetrical form of the procedure is repeated from the rightmost updated node, resulting in a correctly annotated tree.

Since the recursive function requires at most  $O(\kappa)$  annotation updates and both primary paths have length  $O(\log n)$ , the worst-case performance bound of the procedure is  $O(\kappa + \log n)$  annotation updates per insertion of non-dominating nodes; if there are only  $s$  score changes ( $s < \kappa$ ), the performance is improved to  $O(s + \log n)$ .

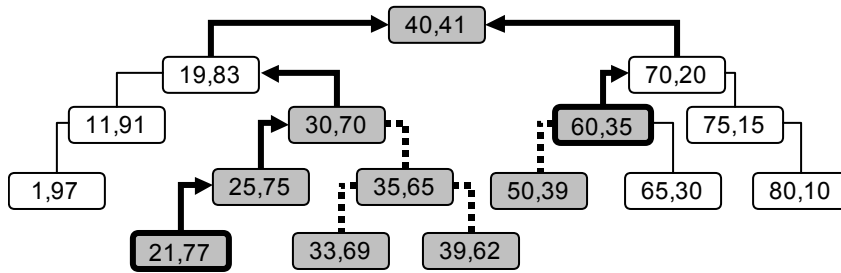
**Algorithm 15 — Annotate Sub-Tree (Recursive Helper Function)**
**Inputs:**

$root$                       The root of the sub-tree  
 $l$                               The leftmost updated node.  
 $r$                               The rightmost updated node.

```

1: let  $current := root$ 
2: let  $left := root^{leftChild}$ 
3: let  $right := root^{rightChild}$ 
4: if (noBreadCrumb( $current$ ))
5:   if ( $right \neq null$ )
6:     if  $\left( \begin{array}{l} (right^{obj1\_label} \leq r^{obj1\_label}) \wedge \\ (right^{obj1\_label} \geq l^{obj1\_label}) \end{array} \right)$ 
7:        $rightVal := annotateSubTree(right, l, r)$ 
8:     else
9:        $rightVal := right^{annotation}$ 
10:   else
11:      $rightVal := ignore$ 
12:   if ( $left \neq null$ )
13:     if  $\left( \begin{array}{l} (left^{obj1\_label} \leq r^{obj1\_label}) \wedge \\ (left^{obj1\_label} \geq l^{obj1\_label}) \end{array} \right)$ 
14:        $leftVal := annotateSubTree(left, l, r)$ 
15:     else
16:        $leftVal := left^{annotation}$ 
17:   else
18:      $leftVal := ignore$ 
19:    $current^{annotation} := \text{best} \left( \begin{array}{l} current^{knnScore} \\ leftVal, rightVal \end{array} \right)$ 
20: return  $current^{annotation}$ 
    
```

If the annotation of this node has not already been updated.  
 If right has been updated then its annotation must be updated first. Otherwise record the annotation of the right for reference later.  
 If right is null then it is excluded from annotation calculations.  
 If left has been updated then its annotation must be updated first. Otherwise record the annotation of the left for reference later.  
 If left is null then it is excluded from annotation calculations.  
 Set and return the current annotation.


**Figure 80 — An Example Annotation Update After Insertion**

The bold lines represent the primary path; the dashed lines indicate the secondary path. Shaded cells have changed crowding scores due to the insertion (note that they are in a contiguous range, as they must be). Shaded cells with dark borders are the leftmost and rightmost adjusted cells. All annotation changes are pushed up to the root at (40,41).

\_\_\_\_\_

The leftmost deleted node (the  $i$ -terminal node)

(c)  $\left( \frac{1}{\sqrt{\pi}} e^{-x^2} \right)$

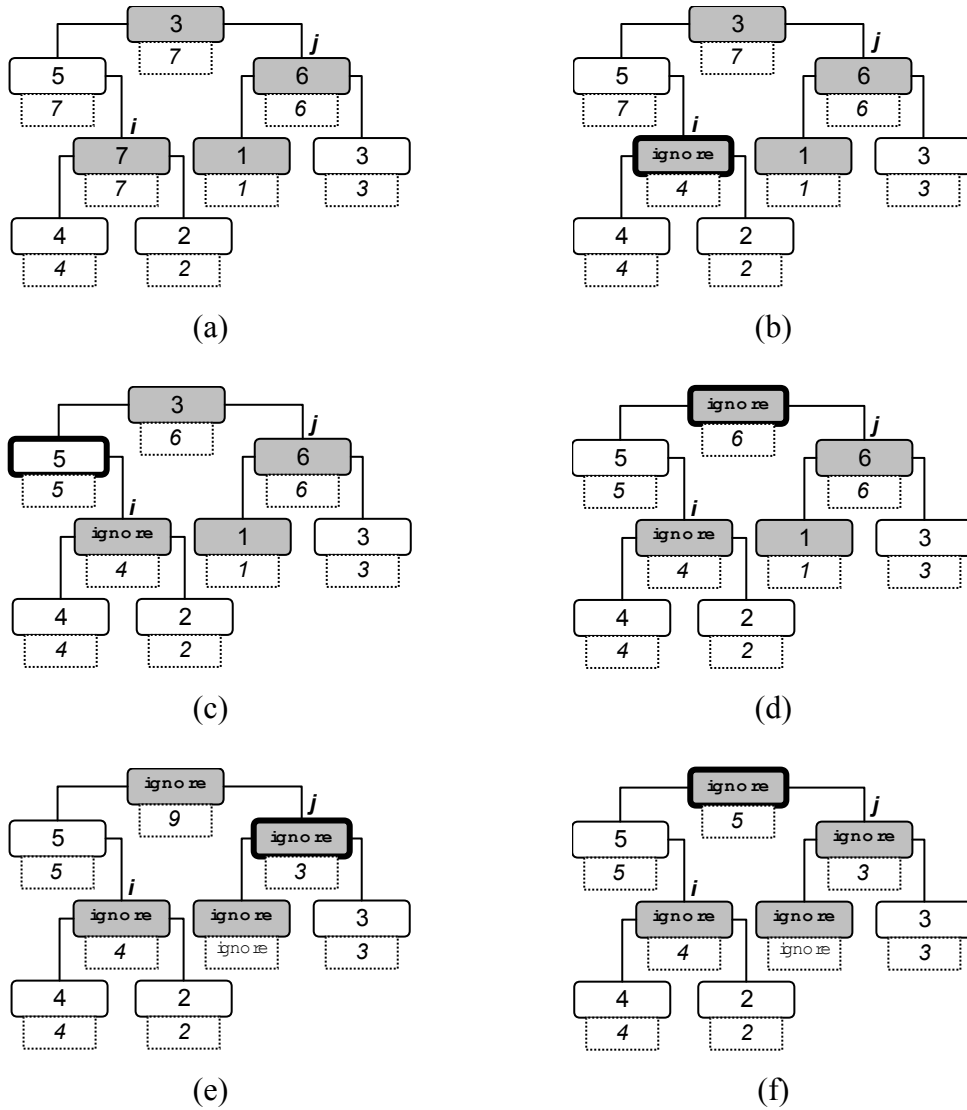
0

187. ( ) ( )

100% 50% 0%

204. *Current* : *Current* 1

have been made. Assuming that the standard Mak\_Tree insertion procedure (Section 0) tags both the roots of dominated sub-trees (which will be removed via sub-tree deletion) and distinct dominated nodes, maintaining correct annotations for the ancestors of deleted nodes becomes straight-forward. As illustrated in Algorithm 16 (and exemplified in Figure 81), the technique requires changes to be pushed up *only* from the first and last dominated node in the list, with special provisions made for those nodes labelled as dominated to ensure the integrity of the annotations (both in this operation and in subsequent deletion-induced rotations). Thus, the incorporation



**Figure 81 — Example Annotation Updates With Deletion**

Dashed cells represent crowding annotations; solid cells indicate crowding scores; shaded cells are to be deleted (they are dominated); cells with emphasised borders are being updated by the algorithm. Note that the procedure starts at the *i*-terminal node (7) and progresses to the root. Once arriving at the root, a symmetrical procedure takes place from the *j*-terminal node (6).

of deletion into the procedure outlined in Section 8.1.5.1.1 carries only an additional  $O(\log n)$  cost, irrespective of the size of the deleted set.

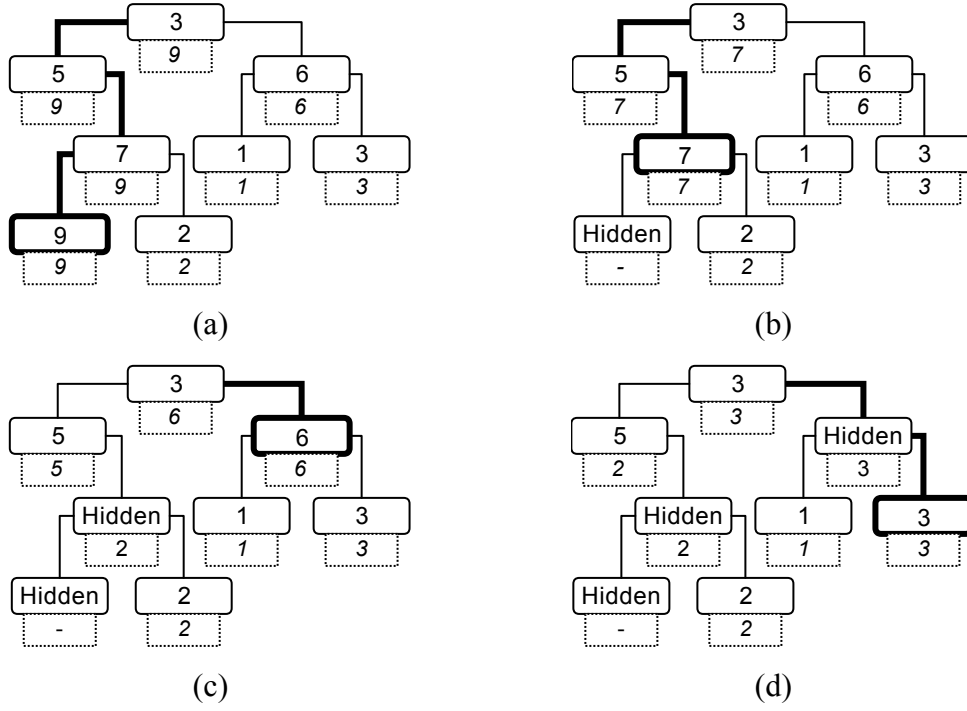
#### 8.1.5.1.3 The Overall Complexity Burden of Mak\_Tree Annotations

A Mak\_Tree featuring a continuously updated set of accurate neighbourhood-based annotations carries at most an additional complexity burden of  $O(\kappa + \log n)$  updates per insertion — even if the incoming solution is highly dominant. Relative to the  $O(\kappa \log n + d \log n)$  costs associated with generic approaches, the improvements offered by such specialisation into the bi-objective domain are noteworthy. As in the maintenance of  $\kappa$  nearest-neighbour scores (Section 8.1.4.3), it is the correlation between proximity in objective-space and proximity in the ordered list that facilitates the efficiency gains. Where generic approaches are obliged to push annotations up from each updated node in-turn, the range-properties of the binary Mak\_Tree tree facilitate the maintenance of large numbers of updated nodes *en masse*.

#### 8.1.5.2 LOCATING A SET OF ISOLATED NODES

While it is useful to know the location of the single least-crowded member of an unbounded elite list, it is perhaps more beneficial in practice to locate a set of isolated nodes. This is particularly true in the case of evolutionary selection, where most systems require a diverse collection of breeding agents to ensure thorough exploration of the front and to minimise the effect of biasing, deception and discontinuities in the objective-space. Fortunately, extending the annotation techniques described in Section 8.1.5.1 to facilitate the selection of the  $\varphi$  most isolated nodes is both simple and efficient.

While it should be obvious that the annotations can be adjusted to represent not just the lowest crowd score of a node's progeny, but the  $\varphi$  lowest scores, this approach expands the spatial complexity of Mak\_Trees to  $O(\varphi n)$  — since each node must maintain a list of at least  $\varphi$  scores — and the additional operational complexity of annotating the tree rises to  $O(\varphi \kappa + \varphi \log n)$ . A better approach then, is to temporarily “hide” each node after it has been selected from the Mak\_Tree (see Figure 82). In this case, changes are pushed up the tree from the selected node as if it were deleted, ensuring that the root annotation subsequently refers to the next least crowded node. Once the  $\varphi$  nodes have been selected, the tree is restored to its initial configuration



**Figure 82 — An Example of Selecting Multiple Uncrowded Nodes from the Mak\_Tree**

Bold lines indicate the selection paths taken to locate the least crowded available node. Dashed cells indicate crowd annotations; solid cells contain crowd scores.

— either by replacing the newly modified tree with the original annotated structure, or by simply “revealing” the original scores of the hidden nodes and pushing up the correct values accordingly. In either case, the spatial complexity reduces to at most  $O(\varphi + n) \equiv O(n)$  (assuming that a reference to the list of selected agents is required<sup>61</sup>), while selection costs only  $O(\varphi \log n)$ , since each time a node is hidden or revealed there are at most  $O(\log n)$  annotation updates required.

While the proposed technique provides good performance when the selection of isolated nodes is infrequent relative to the number of tree updates, minor practical improvements can be made if a constant reference to the collection of the  $\varphi$  uncrowded solutions is required. In this case, the set of selected nodes is stored as a simple binary-tree ordered by crowd-score, with each node remaining hidden in the primary Mak\_Tree until a successful insertion occurs. Upon insertion, any dominated members of the selection set are simply removed, while updated members of the set are extracted and revealed in the Mak\_Tree. Each updated node is then

<sup>61</sup> It is debatable as to whether this cost is attributable to the Mak\_Tree or the multiobjective optimisation algorithm, since the algorithm will almost certainly require an equivalent list.



tested against the worst-scoring member of the selection-set, being added if it is preferable — always ensuring that the set size is never greater than  $\varphi$ . Once all updated nodes are tested, the remainder of the set is filled using the simple hiding mechanic described earlier. While the worst-case time-cost bound is inferior, with a cost of  $O(\kappa + v \log \varphi + h \log n)$  (where  $h$  is the number of nodes which must be hidden or revealed and  $v$  is the number of deletions or insertions into the subsidiary binary tree), the typical performance is likely to be better since it is probable that  $c < \varphi$ ,  $v < \kappa$  and  $\log \varphi$  will generally be very small. Still, the improvements are relatively minor and for the remainder of this work, it is assumed that the simpler approach is adopted.

## 8.2 THE COMPLEXITY OF THE FULLY-FEATURED EXTENDED MAK\_TREE

In section 6.4.1.2, the performance of the basic Mak\_Tree was described as being  $O(\log n)$  when strictly non-dominating solutions are inserted into the archive and  $O(\eta \log n)$  when  $\eta$  members of the archive are dominated by the incoming proposal. How then does the overall performance of a fully-featured extended Mak\_Tree — complete with  $\kappa$  nearest-neighbour estimates, annotations and the periodic selection of uncrowded members — compare with this extremely efficient base case? When taken *in toto*, are the features of the extended Mak\_Tree prohibitively expensive for use in an unbounded setting?

Firstly consider the insertion of a non-dominating solution. The fully-featured extended Mak\_Tree must maintain a non-dominated list (as per the basic Mak\_Tree), update the neighbourhood scores of affected solutions, maintain neighbourhood-based annotations and facilitate periodic selection. Assuming selection occurs at most every  $O(\varphi)$  attempted insertions (as is typical of most contemporary multiobjective evolutionary algorithms — see, for instance, NSGA-II [82], SPEA2 [81] and PESA [136]) the total time cost is:

$$O\left((\log n) + (\kappa) + (\kappa + \log n) + \left(\frac{\varphi \log n}{\varphi}\right)\right) = O(3 \log n + 2\kappa) = O(\log n + \kappa) \quad (70)$$

Therefore the cost of maintaining and using a rich set of useful crowding estimates carries only an  $O(\kappa)$  cost beyond that which is required to preserve the complete

unbounded archival set when non-dominating solutions are inserted. Given that  $\kappa$  will typically be a relatively small constant, this overhead is insignificant given the impressive potential that the extended Mak\_Tree affords.

As with the basic Mak\_Tree, insertion of a dominating solution is more complex and carries a greater burden. Specifically, when  $\kappa > \eta$  the fully-featured extended Mak\_Tree has a worst-case time complexity of:

$$O\left((\eta \log n) + (\kappa \eta) + (\kappa + \log n) + \left(\frac{\varphi \log n}{\varphi}\right)\right) = O(\eta(\log n + \kappa)) \quad (71)$$

When  $\kappa \leq \eta$ , the cost becomes:

$$O\left((\eta \log n) + (\kappa^2) + (\kappa + \log n) + \left(\frac{\varphi \log n}{\varphi}\right)\right) = O(\eta \log n + \kappa^2) \quad (72)$$

At first glance, these complexities look somewhat overwhelming — particularly in the presence of highly dominating solutions and large values of  $\kappa$ . It is reassuring then that amortised time analysis illustrates a far more efficient practical reality.

## 8.2.1 AMORTISED COST OF THE FULLY FEATURED MAK\_TREE

Since the performance of the deletion operation varies with the size of  $\kappa$  relative to  $\eta$ , the worst-case amortised time must consider various configurations of  $\kappa$  and  $\eta$ .

### 8.2.1.1 AMORTISED COST WHEN $\kappa > \eta > 0$

If  $\kappa > \eta$  across a set of  $\chi$  insertion and deletion operations, the worst-case amortised time is:

$$O\left(\frac{\chi(\log n + \kappa) + \chi(\log n + \kappa)}{\chi}\right) = O(\log n + \kappa) \quad (73)$$

Similarly, if  $\chi$  solutions are inserted, the entire set is subsequently dominated and  $\kappa > \eta$ , then the cost is:

$$O\left(\frac{\chi(\log n + \kappa) + (\chi \log n + \kappa)}{\chi}\right) = O\left(2 \log n + \kappa + \frac{\kappa}{\chi}\right) \quad (74)$$

and since  $0 < \chi < \kappa$ , the worst-case amortised cost is:

$$O\left(2\log n + \kappa + \frac{\kappa}{\chi}\right) = O(2\log n + \kappa + 1) = O(\log n + \kappa) \quad (75)$$

Thus, the final worst-case amortised cost of deletion in the extended Mak\_Tree when  $\kappa > \eta$  is  $O(\log n + \kappa)$ .

### 8.2.1.2 AMORTISED COST WHEN $0 < \kappa \leq \eta$

If multiple deletes are performed across  $\chi$  operations and  $\kappa \leq \eta$  for each delete (worst-case is when  $\kappa = \eta$ ), the amortised time is:

$$O\left(\frac{\chi(\log n + \kappa) + (\eta \log n + \eta^2)}{\chi}\right) = O\left(\frac{\chi(\log n + \kappa) + \eta(\log n + \kappa)}{\chi}\right) \quad (76)$$

Since  $\chi$  cannot be less than  $\eta$  (otherwise, deletion will be operating on an empty set) and any  $\chi > \eta$  will improve performance (deletion is more costly than insertion), the worst case is when  $\chi = \eta$  and is therefore equivalent to Equation (76) above:

$$O\left(\frac{\eta(\log n + \kappa) + \eta(\log n + \kappa)}{\eta}\right) = O(\log n + \kappa) \quad (77)$$

If  $\chi$  solutions are inserted and the entire set is subsequently dominated and  $\chi = \kappa \leq \eta$ , then the worst-case amortised cost is:

$$O\left(\frac{\chi(\log n + k) + (\chi \log n + k^2)}{\chi}\right) = O(\log n + \frac{k^2}{\chi}) \Rightarrow O(\log n + k) \quad (78)$$

Therefore, the final worst-case amortised performance of deletion in the fully featured Mak\_Tree when  $\kappa \leq \eta$  is  $O(\log n + \kappa)$ .

### 8.2.1.3 AMORTISED COST WHEN $\eta = 0$

When there are no nodes deleted, the worst-case amortised cost for insertion into the fully featured Mak\_Tree remains at  $O(\log n + \kappa)$ . Analysis is trivial, but provided for completeness. Over  $\chi$  non-dominating insertions, the amortised cost is:

$$O\left(\frac{\chi(\log n + \kappa)}{\chi}\right) = O(\log n + \kappa) \quad (79)$$

#### 8.2.1.4 **OVERALL AMORTISED COST OF USING THE FULLY FEATURED *MAK\_TREE***

As described in Sections 8.2.1.1, 8.2.1.2 and 8.2.1.3, maintaining a truly unbounded non-dominated set of solutions, complete with rich diversity statistics, annotations and periodic selections, has an amortised cost of only  $O(\log n + \kappa)$  in the extended *Mak\_Tree*. Given the power inherent in such extra functionality — particularly with respect to the integration of unbounded archives into existing and new evolutionary algorithms — such high levels of efficiency are impressive. Moreover, the amortised time costs indicate performance consistency that is not available in any pre-existing unbounded elite archival system — dominance frequency and scope do not affect the theoretical performance of the fully-featured extended *Mak\_Tree* beyond the way in which such properties affect the size of the unbounded set. Consequently, both optimisation algorithms that focus on diversification ahead of frontal progression and more elitist options will be equally well served by the extended *Mak\_Tree*. Such general applicability is of pivotal importance if unbounded bi-objective archives are to find wide-ranging application in the field. Still, theoretical results are one thing and practical results quite another, it is therefore important to thoroughly analyse the fully-featured extended *Mak\_Tree* in a practical environment. If the theoretical results are supported by impressive practical performance, the value of the extended *Mak\_Tree* will be confirmed.

#### 8.2.2 **EMPIRICAL RESULTS**

In Section 8.1.4.4 the extended *Mak\_Tree* proved to be not only comparable with existing generic approaches to density estimation in truncated environments, but frequently *preferable*. It is therefore worth examining whether the costs associated with annotation and frequent selections skew the practical performance advantages of the *Mak\_Tree*. As such, the behaviour of the extended unbounded *Mak\_Tree* is examined under differing rates of selection and again compared to contemporary generic approaches that feature heavily truncated archives ( $n = 50$ ). Specifically, results (see pages 215–219 for relevant tables and graphs) describe performance across the now familiar set of problems described in Section 6.4.2.1, with selection of  $\phi$  agents occurring after every  $\phi$  attempted insertions (as would be encountered in most evolutionary algorithms). For the generic approaches, selection requires the sorting of  $n$  (50) solutions per  $\phi$  insertions (irrespective of the size of  $\phi$ ) and the use

of a list-truncation procedure when  $\varphi < n$  (to return the appropriate selection set), but remain otherwise unaltered from the techniques described in Section 8.1.4.4. When only annotations are being examined, both the batch-processed  $\kappa^{\text{th}}$  neighbour and cuboid approaches remain unchanged from Section 8.1.4.4. As per previous empirical examinations (see Section 6.4.2.3, for instance) all results reflect the average cumulative time-costs of each approach on twenty data sets per problem (produced by NSGA-II with settings defined in Appendix B.1.1).

Table 19, Figure 83, Figure 84 and Figure 85 clearly illustrate that when only annotations are required, the extended Mak\_Tree is superior to the truncated  $\kappa^{\text{th}}$  nearest-neighbour technique on every tested function. Given that the Mak\_Tree is unbounded and maintains annotations for every node in the tree, these results are particularly impressive. Even when compared to the two-neighbour cuboid method, the annotated Mak\_Tree achieves consistently better end-of-run results when  $\kappa = 5$ , and is frequently more efficient, and generally competitive, when  $\kappa$  is as high as 20.

Perhaps more importantly, the incorporation of frequent solution selections — an important element of most contemporary evolutionary algorithms — does not severely inhibit the performance of the extended Mak\_Tree. Indeed, Figure 83, Figure 84 and Figure 85 and Table 20, Table 21 and Table 22 illustrate that the proposed hiding mechanic results in a fully-featured Mak\_Tree that is preferable to the incremental cuboid approach on all tested selection frequencies when  $\kappa = 5$  and is faster than both batch  $\kappa^{\text{th}}$  nearest-neighbour and batch cuboid approaches when  $\varphi$  is fifty<sup>62</sup>. It is also worth noting that the extended Mak\_Tree is comparable, and typically preferable, to the cuboid approach even early in the run, as illustrated in Table 20 (again discounting the notion that sophisticated unbounded data structures are only relevant if population levels are particularly high).

Figure 83, Figure 84 and Figure 85 also illustrate an interesting trend with respect to the extension of the basic Mak\_Tree. While each layer of added complexity degrades the overall performance of the Mak\_Tree, the resultant time-costs remain approximately linear with respect to the number of evaluations, despite the diverse problem set, the range of population sizes and the varying archival growth rates.

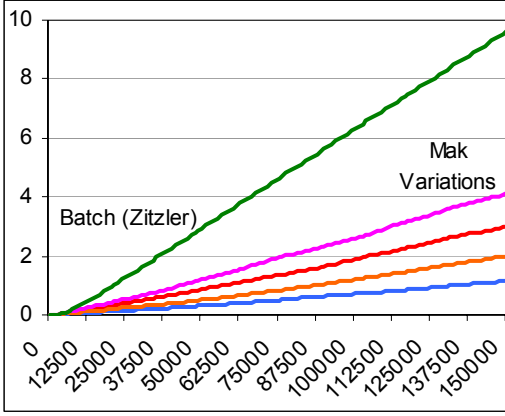
<sup>62</sup> Note that for these batch truncated systems, when  $\varphi = m$  the selection procedure is optimal, as additional distance calculations and solution sorts are not required if truncation takes place.

This is a noteworthy result. If the extensions carried an exponentially increasing burden, there would be a danger that longer runs or higher population levels could lead to prohibitive time costs in real-world settings. In contrast, the lack of such a burden and the consistency of the results imply that the performance of the fully-featured extended Mak\_Tree will remain efficient irrespective of problem or archival characteristics — a promising feature for systems seeking practical application.

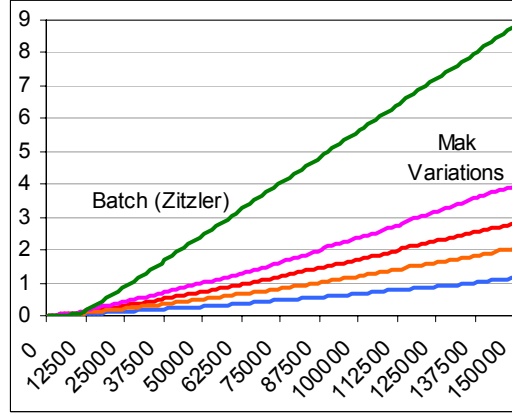
**Table 19 — Empirical Performance of the Annotated Mak\_Tree**

*Mak* refers to the fully-featured Mak\_Tree with  $\kappa$ -neighbours crowding, annotations and periodic selections; *Deb* refers to the cuboid-based truncated approach with either incremental or batch updating; *Zit* refers to the generic  $\kappa$  nearest-neighbour approach operating on a truncated set. Results indicate average time in seconds.

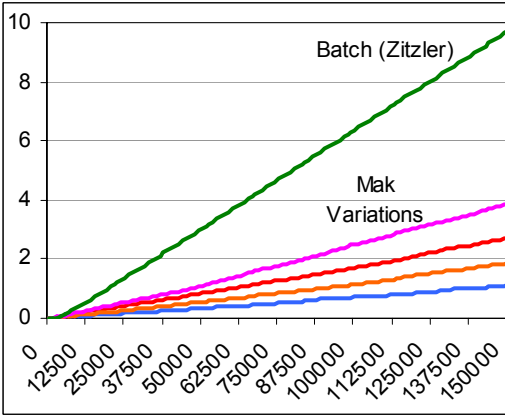
	Mak k=0	Mak_NA k=5	Mak_A k=5	Zit (B) k=5	Mak_NA k=20	Mak_A k=20	Zit (B) k=20	Deb (B)	
<b>AP-1</b>	0.06	0.08	0.10	0.28	0.09	0.15	0.34	0.14	<b>10,000 Evaluations</b>
<b>AP-2</b>	0.05	0.06	0.07	0.07	0.07	0.08	0.08	0.05	
<b>AP-3</b>	0.07	0.07	0.10	0.37	0.11	0.17	0.44	0.14	
<b>AP-4</b>	0.05	0.07	0.07	0.21	0.07	0.12	0.27	0.09	
<b>AP-5</b>	0.04	0.06	0.10	0.17	0.07	0.12	0.19	0.08	
<b>AP-8</b>	0.03	0.03	0.04	0.16	0.06	0.06	0.18	0.06	
<b>AP-9</b>	0.01	0.05	0.04	0.23	0.04	0.08	0.28	0.07	
<b>AP-10</b>	0.04	0.05	0.08	0.37	0.07	0.15	0.43	0.13	
<b>AP-15</b>	0.06	0.08	0.12	0.51	0.13	0.20	0.56	0.19	
<b>AP-16</b>	0.05	0.10	0.14	0.60	0.14	0.25	0.66	0.21	
<b>AP-17</b>	0.07	0.12	0.15	0.63	0.14	0.27	0.75	0.26	
<b>AP-21</b>	0.05	0.09	0.14	0.56	0.12	0.21	0.60	0.19	
<b>F-1</b>	0.05	0.06	0.09	0.30	0.08	0.13	0.34	0.13	
	Mak k=0	Mak_NA k=5	Mak_A k=5	Zit (B) k=5	Mak_NA k=20	Mak_A k=20	Zit (B) k=20	Deb (B)	
<b>AP-1</b>	0.32	0.46	0.58	2.52	0.51	0.87	3.03	0.97	<b>50,000 Evaluations</b>
<b>AP-2</b>	0.28	0.45	0.47	2.09	0.50	0.73	2.57	0.78	
<b>AP-3</b>	0.34	0.41	0.56	2.80	0.55	0.84	3.10	0.93	
<b>AP-4</b>	0.34	0.49	0.52	2.38	0.59	0.93	2.82	0.80	
<b>AP-5</b>	0.28	0.44	0.63	2.56	0.58	0.97	2.82	0.97	
<b>AP-8</b>	0.11	0.16	0.15	0.93	0.18	0.22	1.12	0.29	
<b>AP-9</b>	0.18	0.21	0.24	1.18	0.17	0.29	1.46	0.38	
<b>AP-10</b>	0.24	0.36	0.47	2.19	0.42	0.72	2.40	0.75	
<b>AP-15</b>	0.34	0.48	0.60	3.36	0.60	0.91	4.00	1.16	
<b>AP-16</b>	0.35	0.46	0.64	3.70	0.60	0.98	4.05	1.21	
<b>AP-17</b>	0.44	0.62	0.77	4.07	0.75	1.32	4.97	1.44	
<b>AP-21</b>	0.34	0.44	0.60	3.57	0.56	0.92	4.01	1.23	
<b>F-1</b>	0.27	0.33	0.42	2.26	0.38	0.57	2.72	0.76	
	Mak k=0	Mak_NA k=5	Mak_A k=5	Zit (B) k=5	Mak_NA k=20	Mak_A k=20	Zit (B) k=20	Deb (B)	
<b>AP-1</b>	1.15	1.76	1.99	8.61	2.01	3.00	10.20	2.98	<b>150,000 Evaluations</b>
<b>AP-2</b>	1.14	1.59	1.90	7.61	2.04	2.80	9.27	2.66	
<b>AP-3</b>	1.12	1.53	1.84	8.62	1.87	2.69	10.18	3.05	
<b>AP-4</b>	1.29	1.89	2.03	8.41	2.40	3.25	9.41	2.68	
<b>AP-5</b>	1.09	1.88	2.46	9.16	2.60	4.09	10.27	3.29	
<b>AP-8</b>	0.37	0.51	0.45	2.71	0.45	0.58	3.48	0.91	
<b>AP-9</b>	0.32	0.66	0.74	3.67	0.55	0.87	4.41	1.22	
<b>AP-10</b>	0.81	1.08	1.34	6.32	1.50	1.91	6.80	2.30	
<b>AP-15</b>	1.01	1.38	1.62	10.92	1.71	2.17	12.76	3.58	
<b>AP-16</b>	1.08	1.56	1.80	11.86	1.72	2.49	12.91	3.77	
<b>AP-17</b>	1.38	2.09	2.36	13.61	2.39	3.63	15.55	4.40	
<b>AP-21</b>	0.98	1.35	1.56	11.49	1.65	2.15	13.13	3.80	
<b>F-1</b>	0.84	1.07	1.12	7.23	1.06	1.39	8.80	2.32	



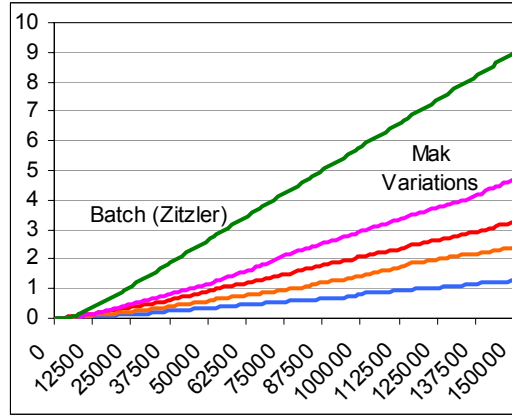
(a) AP-1



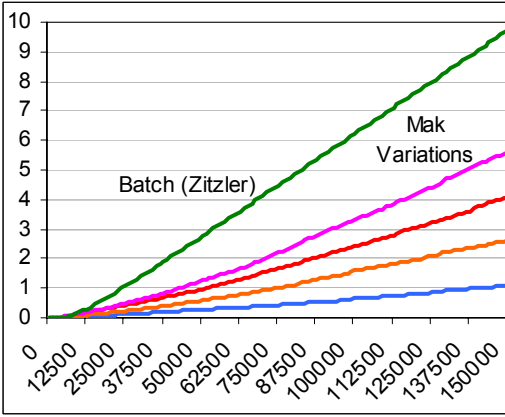
(b) AP-2



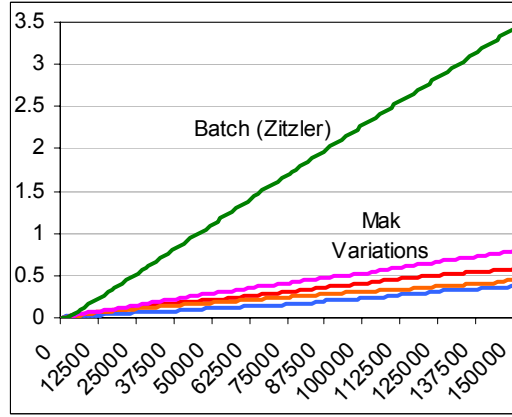
(c) AP-3



(d) AP-4



(e) AP-5

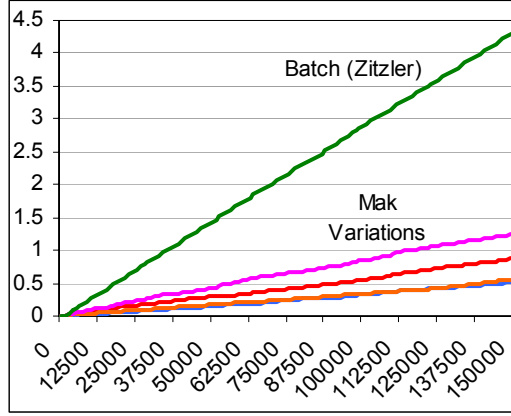
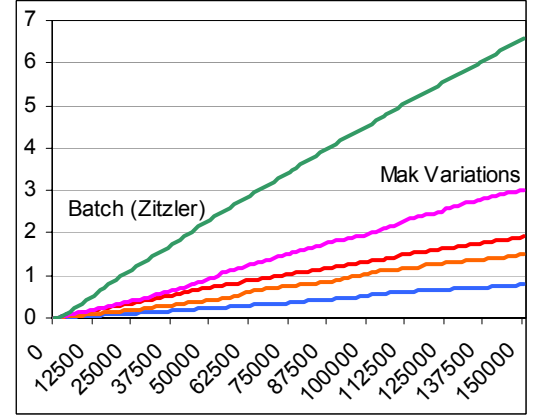
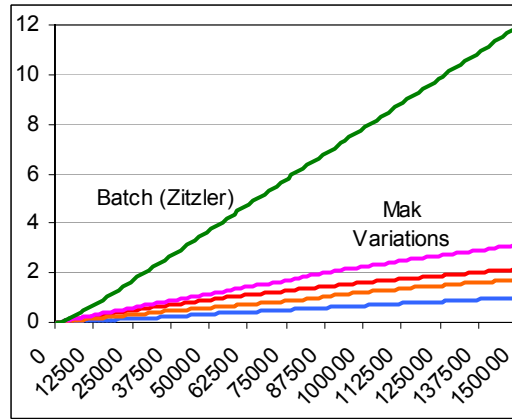
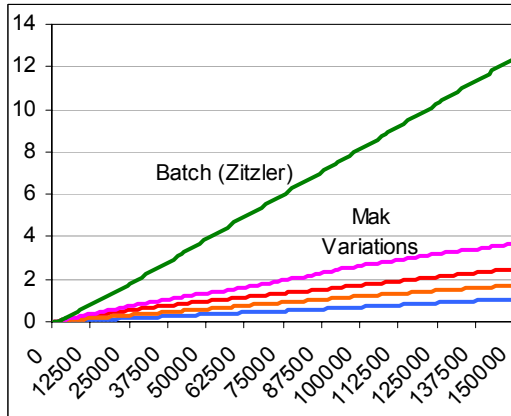
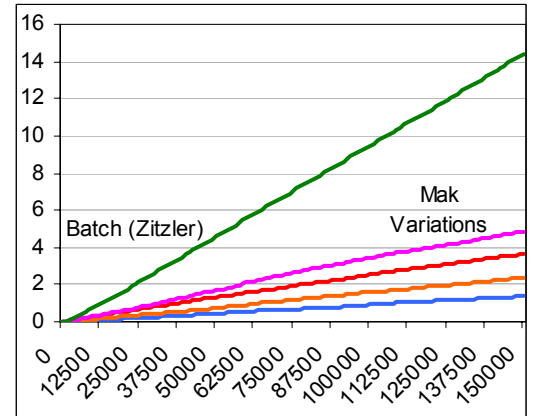


(f) AP-8

**Figure 83 — Average Cumulative Time Costs of Unbounded and Bounded Archiving Techniques With Complete Density Estimates, Annotations and Selection ( $\kappa = 20, \varphi = 50$ )**

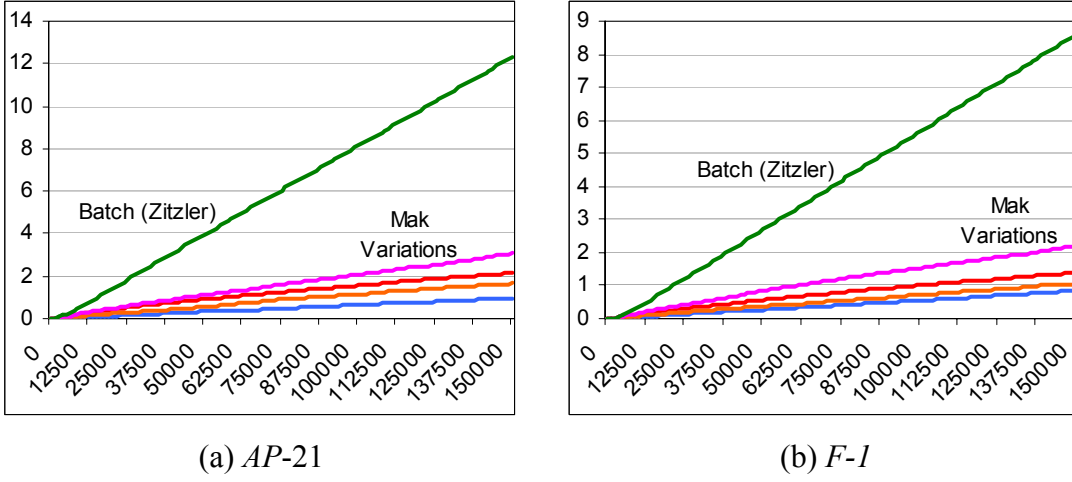
For all graphs, the  $x$ -axis represents the number of solutions presented to the archive and the  $y$ -axis is the average cumulative processing time in seconds. The Zitzler results refer to performance in a batch-processed truncated archive of 50. The Mak Variations are, in increasing order of time-cost, the Basic Mak\_Tree, the Crowded Mak\_Tree, the Annotated Crowded Mak\_Tree, and the Annotated Crowded Mak\_Tree with Selection.



(a) *AP-9*(b) *AP-10*(d) *AP-15*(e) *AP-16*(f) *AP-17*

**Figure 84 — Average Cumulative Time Costs of Unbounded and Bounded Archiving Techniques With Complete Density Estimates, Annotations and Selection ( $\kappa = 20$ ,  $\varphi = 50$ )**

For all graphs, the *x-axis* represents the number of solutions presented to the archive and the *y-axis* is the average cumulative processing time in seconds. The Zitzler results refer to performance in a batch-processed truncated archive of 50. The Mak Variations are, in increasing order of time-cost, the Basic Mak\_Tree, the Crowded Mak\_Tree, the Annotated Crowded Mak\_Tree, and the Annotated Crowded Mak\_Tree with Selection.



**Figure 85 — Average Cumulative Time Costs of Unbounded and Bounded Archiving Techniques With Complete Density Estimates, Annotations and Selection ( $\kappa = 20$ ,  $\varphi = 50$ )**

For all graphs, the  $x$ -axis represents the number of solutions presented to the archive and the  $y$ -axis is the average cumulative processing time in seconds. The Zitzler results refer to performance in a batch-processed truncated archive of 50. The Mak Variations are, in increasing order of time-cost, the Basic Mak\_Tree, the Crowded Mak\_Tree, the Annotated Crowded Mak\_Tree, and the Annotated Crowded Mak\_Tree with Selection.

**Table 20 — Empirical Performance of the Fully-Featured Mak\_Tree and the Truncated Cuboid After 10,000 NSGA-II Evaluations**

*Mak* refers to the extended Mak\_Tree with  $\kappa$ -neighbours crowding, annotations and periodic selections; *Deb* refers to the cuboid truncated approach with either incremental or batch updating. Results indicate average time in seconds.

	$\varphi = 10$			$\varphi = 20$			$\varphi = 30$			$\varphi = 40$			$\varphi = 50$			
	Mak k=5	Mak k=20	Deb Inc	Mak k=5	Mak k=20	Deb Inc	Mak k=5	Mak k=20	Deb Inc	Mak k=5	Mak k=20	Deb Inc	Mak k=5	Mak k=20	Deb Inc	Deb Bat
<b>AP-1</b>	0.16	0.20	0.26	0.17	0.24	0.15	0.16	0.29	0.18	0.13	0.32	0.19	0.14	0.23	0.18	0.24
<b>AP-2</b>	0.08	0.10	0.06	0.11	0.10	0.07	0.09	0.11	0.06	0.06	0.10	0.05	0.08	0.11	0.09	0.08
<b>AP-3</b>	0.13	0.20	0.24	0.14	0.25	0.21	0.15	0.30	0.20	0.14	0.28	0.18	0.13	0.30	0.22	0.27
<b>AP-4</b>	0.09	0.13	0.13	0.12	0.17	0.13	0.12	0.17	0.11	0.13	0.19	0.13	0.10	0.14	0.15	0.14
<b>AP-5</b>	0.11	0.14	0.15	0.12	0.21	0.11	0.14	0.21	0.11	0.11	0.19	0.12	0.13	0.23	0.12	0.13
<b>AP-8</b>	0.05	0.07	0.10	0.06	0.09	0.09	0.04	0.13	0.08	0.07	0.14	0.07	0.06	0.09	0.10	0.09
<b>AP-9</b>	0.07	0.09	0.13	0.08	0.12	0.10	0.07	0.10	0.10	0.08	0.15	0.10	0.07	0.12	0.13	0.11
<b>AP-10</b>	0.13	0.19	0.19	0.11	0.19	0.17	0.12	0.24	0.18	0.12	0.23	0.20	0.12	0.21	0.22	0.19
<b>AP-15</b>	0.20	0.25	0.32	0.20	0.33	0.28	0.17	0.35	0.23	0.17	0.31	0.23	0.17	0.38	0.31	0.31
<b>AP-16</b>	0.18	0.26	0.34	0.19	0.36	0.30	0.20	0.35	0.27	0.18	0.34	0.26	0.19	0.39	0.34	0.31
<b>AP-17</b>	0.21	0.33	0.31	0.20	0.35	0.32	0.23	0.45	0.29	0.23	0.42	0.34	0.21	0.38	0.41	0.31
<b>AP-21</b>	0.21	0.24	0.31	0.19	0.33	0.29	0.16	0.30	0.25	0.17	0.43	0.24	0.18	0.29	0.31	0.25
<b>F-1</b>	0.14	0.18	0.23	0.15	0.21	0.19	0.13	0.22	0.18	0.12	0.24	0.16	0.13	0.23	0.21	0.19

**Table 21 — Empirical Performance of the Fully-Featured Mak\_Tree and the Truncated Cuboid After 50,000 NSGA-II Evaluations**

*Mak* refers to the extended Mak\_Tree with  $\kappa$ -neighbours crowding, annotations and periodic selections;  
*Deb* refers to the cuboid truncated approach with either incremental or batch updating. Results indicate average time in seconds.

	$\varphi = 10$			$\varphi = 20$			$\varphi = 30$			$\varphi = 40$			$\varphi = 50$			
	Mak k=5	Mak k=20	Deb Inc	Mak k=5	Mak k=20	Deb Inc	Mak k=5	Mak k=20	Deb Inc	Mak k=5	Mak k=20	Deb Inc	Mak k=5	Mak k=20	Deb Inc	Deb Bat
<b>AP-1</b>	0.91	1.15	1.55	0.95	1.43	1.35	0.92	1.53	1.34	0.87	1.52	1.34	0.81	1.48	1.51	1.46
<b>AP-2</b>	0.68	0.94	1.30	0.77	1.17	1.11	0.75	1.33	1.08	0.66	1.45	1.11	0.67	1.26	1.29	1.17
<b>AP-3</b>	0.85	1.07	1.53	0.88	1.34	1.44	0.78	1.50	1.27	0.75	1.64	1.42	0.75	1.49	1.53	1.45
<b>AP-4</b>	0.91	1.12	1.41	0.92	1.33	1.20	0.79	1.52	1.12	0.81	1.69	1.27	0.82	1.59	1.33	1.30
<b>AP-5</b>	0.88	1.22	1.51	1.02	1.53	1.46	0.89	1.69	1.24	0.88	1.90	1.31	0.82	1.62	1.44	1.39
<b>AP-8</b>	0.24	0.25	0.50	0.26	0.34	0.43	0.26	0.41	0.39	0.24	0.44	0.40	0.22	0.36	0.44	0.52
<b>AP-9</b>	0.31	0.36	0.61	0.35	0.46	0.52	0.32	0.53	0.56	0.37	0.53	0.54	0.31	0.57	0.60	0.63
<b>AP-10</b>	0.75	0.92	1.22	0.75	1.06	1.03	0.72	1.18	0.98	0.63	1.41	1.09	0.70	1.29	1.22	1.11
<b>AP-15</b>	0.93	1.15	1.79	0.96	1.47	1.66	0.90	1.63	1.64	0.90	1.57	1.54	0.85	1.57	1.78	1.70
<b>AP-16</b>	0.97	1.18	1.96	0.95	1.50	1.77	0.92	1.73	1.72	0.83	1.71	1.65	0.90	1.65	1.85	1.61
<b>AP-17</b>	1.25	1.73	2.20	1.29	1.98	1.92	1.17	2.20	1.94	1.22	2.40	1.94	1.13	2.25	2.24	1.97
<b>AP-21</b>	0.97	1.15	1.94	0.92	1.50	1.74	0.84	1.61	1.74	0.82	1.69	1.76	0.82	1.39	1.96	1.74
<b>F-1</b>	0.81	0.79	1.33	0.75	1.01	1.13	0.67	1.05	1.01	0.62	1.12	1.11	0.60	1.00	1.34	1.18

**Table 22 — Empirical Performance of the Fully-Featured Mak\_Tree and the Truncated Cuboid After 150,000 NSGA-II Evaluations**

*Mak* refers to the extended Mak\_Tree with  $\kappa$ -neighbours crowding, annotations and periodic selections;  
*Deb* refers to the cuboid truncated approach with either incremental or batch updating. Results indicate average time in seconds.

	$\varphi = 10$			$\varphi = 20$			$\varphi = 30$			$\varphi = 40$			$\varphi = 50$			
	Mak k=5	Mak k=20	Deb Inc	Mak k=5	Mak k=20	Deb Inc	Mak k=5	Mak k=20	Deb Inc	Mak k=5	Mak k=20	Deb Inc	Mak k=5	Mak k=20	Deb Inc	Deb Bat
<b>AP-1</b>	3.03	3.81	4.93	3.27	4.73	4.40	3.12	5.28	4.22	2.93	5.51	4.35	2.91	5.18	4.78	4.45
<b>AP-2</b>	2.84	3.52	4.47	3.10	4.42	3.97	2.95	4.87	3.90	2.59	5.18	3.80	2.57	4.76	4.36	4.19
<b>AP-3</b>	2.98	3.63	4.91	3.19	4.45	4.48	2.87	5.00	4.30	2.79	5.09	4.36	2.79	4.94	4.85	4.47
<b>AP-4</b>	3.31	4.04	4.63	3.49	4.84	4.08	3.30	5.68	3.94	3.22	5.95	4.14	3.31	5.68	4.62	4.36
<b>AP-5</b>	3.74	5.21	5.12	4.07	6.48	4.79	3.80	7.12	4.52	3.41	7.40	4.68	3.43	7.39	5.19	4.71
<b>AP-8</b>	0.75	0.72	1.50	0.80	0.91	1.34	0.76	1.06	1.30	0.71	1.15	1.34	0.68	1.05	1.52	1.58
<b>AP-9</b>	0.97	1.07	1.91	1.11	1.34	1.76	0.99	1.43	1.67	0.98	1.48	1.70	0.99	1.63	1.87	1.91
<b>AP-10</b>	2.14	2.81	3.54	2.25	3.35	3.16	2.08	3.81	3.05	1.83	3.89	3.24	1.89	3.63	3.59	3.10
<b>AP-15</b>	2.69	2.99	5.56	2.69	3.77	5.15	2.49	4.23	4.96	2.47	4.22	4.97	2.36	4.09	5.74	5.48
<b>AP-16</b>	2.91	3.31	6.04	2.85	4.24	5.51	2.78	4.70	5.35	2.65	4.65	5.36	2.68	4.63	6.00	5.53
<b>AP-17</b>	3.64	4.73	6.75	3.91	5.64	6.09	3.46	6.18	5.85	3.41	6.53	6.10	3.27	6.25	6.92	6.07
<b>AP-21</b>	2.56	2.91	5.88	2.60	3.65	5.37	2.40	4.19	5.26	2.41	4.21	5.28	2.28	3.87	5.94	5.59
<b>F-1</b>	2.09	2.06	3.92	2.10	2.62	3.51	1.98	2.82	3.22	1.84	3.04	3.30	1.83	2.75	3.79	3.71

### 8.3 CELL-BASED CROWDING IN MAK\_TREES

As suggested in Section 8.1.1.2, tree data structures are well suited to generic cell-based crowding procedures, offering an efficient alternative to distance-based density estimates. The question is therefore whether specialisation into the bi-objective domain yields any noteworthy benefits beyond those already seen in the generic case. Does the use of the extended Mak\_Tree provide advantages that simply cannot be achieved when the broad brush-stroke of generality is applied?

#### 8.3.1 USING A MAK\_TREE FOR CELL-BASED DENSITY ESTIMATION

It should be obvious that a Mak\_Tree can be used to store cells with little modification to the original structure. Indeed, if the ordering of solutions is defined by cell-coordinates rather than by precise objective-scores, the tree will store a list of non-dominated objective-space cells. As such, the worst-case cost of using a cell-based Mak\_Tree is only  $O(\eta \log c)$  when  $\eta > 0$  deletions are required, or  $O(\log c)$  otherwise (where  $c$  is the number of stored cells, and assuming cell sizes remain fixed). As with the basic Mak\_Tree (Section 6.4.1.2), the overall complexity of insertions into the cell-based tree reduces to  $O(\log c)$  when amortised time analysis is used; again, this is a more faithful representation of practical performance.

Extension of the cell-based Mak\_Tree to incorporate a complete set of crowding scores is trivial — each cell need only maintain a count of the number of members stored. Since a successfully inserted member will only belong to one cell<sup>63</sup>, and because the crowding update procedure can be performed in constant time (it is little more than an increment procedure), the extension of the cell-based Mak\_Tree to allow crowding statistics comes at no cost to the overall complexity of the original technique.

Using generic approaches means that such impressive performance is impossible — the storage and updating of density estimates can be achieved efficiently, but the same cannot be said of maintaining an unbounded list of non-dominated cells (the inefficiencies associated with maintaining individual solutions apply at the cell-level here). As such, the Mak\_Tree provides an excellent foundation for cell-based elite

---

<sup>63</sup> Note that this precludes the use of overlapping cells.

archiving — but further efficacy and efficiency gains can be made when a modified form of the extended Mak\_Tree is considered.

### 8.3.2 ANNOTATING AND SELECTING CELLS

When selecting the least or most crowded cell is beneficial, the Mak\_Tree specialisation can be used to provide extremely efficient tree annotations. Via a simplified form of the procedures described in Section 8.1.5.1, complete annotations can be maintained at a cost of only  $O(\log c)$  per successful insertion. Specifically, the cell annotation algorithm proceeds as follows: if an insertion results in a deletion, use the procedure described in Section 8.1.5.1.2 (though there is no need to consider neighbourhood effects); otherwise simply push changes up from the updated cell.

With annotations in place, selection of cells based on crowding information can capitalise on the same hiding mechanic as described in Section 8.1.5.2 and will therefore carry an equivalent  $O(\varphi \log c)$  cost.

### 8.3.3 MERGING DENSITY PROCEDURES

A potentially interesting, and largely unexplored, idea is to merge cell-based and distance-based density estimates in elite archives. In principle, by capitalising on the localised cell-based approach *and* the wider-reaching neighbourhood technique, the impact of the bias/variance dilemma diminishes. For instance, if the cell-based Mak\_Tree is annotated with both cell-crowding scores and  $\kappa$  nearest-neighbour estimates (and assuming that the resolution and  $\kappa$  parameters are complementary<sup>64</sup>), the optimisation algorithm can balance localised and global density concerns by simply alternating the selection mechanism from one annotation to the other. The corresponding incorporation of nearest-neighbour estimates and suitable annotations in the extended cell-based Mak\_Tree will increase the overall performance cost to  $O(\log c + \kappa)$  under the insertion of non-dominating solutions. Under the insertion of dominating proposals, the cost increases to  $O(\eta(\log c + \kappa))$  when they dominate fewer than  $\kappa$  solutions and to  $O(\eta \log n + \kappa^2)$  otherwise. Again, the more practical amortised time analysis reduces these costs to  $O(\log c + \kappa)$  per insertion.

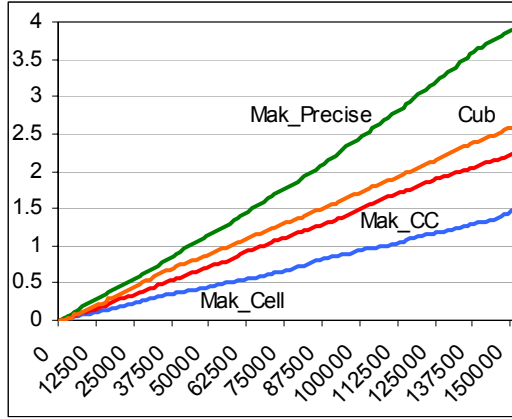
<sup>64</sup> Ideally, the grid resolution should be fine and the  $k$  value large, though the exact settings are largely subject to the problem at hand. Development of heuristics for determining trade-offs between the two parameters is an area of potential future work.

### 8.3.4 EMPIRICAL RESULTS AND DISCUSSION

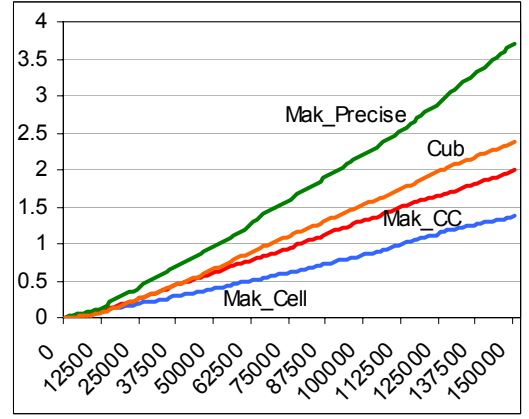
To examine the performance of the basic cell-based Mak\_Tree and the fully-featured alternative that provides facilities for merged density estimation, both techniques are compared with the truncated (non-cell-based) batch-processed two-neighbour cuboid approach (the most efficient of the examined truncated techniques) across a rich set of test problems (as per the methodologies described in Sections 6.4.2.3, 8.1.4.4 and 8.2.2). The tested Mak\_Trees feature non-overlapping square cells of side length  $b$ , with a range of  $b$  values investigated to elucidate the impact of cell sizing on performance.

As evidenced in Figure 86, Figure 87 and Figure 88 and in Table 23, Table 24 and Table 25 (pages 223–226), across a range of  $b$  values the fully-featured cell-based Mak\_Trees with merged density estimates achieve consistently faster run-times than even the simple truncated cuboid approach (excepting very early in the run, where the Mak\_Trees remains highly competitive, but not noticeably superior). This is an impressive result given that the fully-featured cell-based Mak\_Trees are unbounded in size, maintain density estimates based on a relatively large neighbourhood size of twenty, and provide tree-spanning annotations for both cell- and neighbourhood-derived statistics.

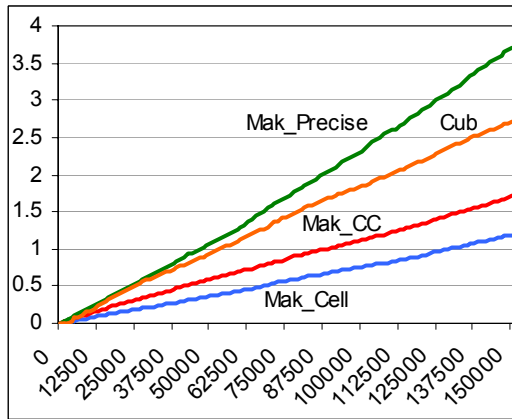
Clearly, the fundamental reason for the increased efficiency seen in the cell-based approach is a reduction in the number of nodes that must be stored in the Mak\_Tree. Consider Figure 89, Figure 90 and Figure 91 (pages 227–229) — even the relatively precise  $b = 0.0001$  setting produces trees that require markedly fewer nodes than the original Mak\_Tree. The question must therefore be: does the loss of precision adversely affect the efficacy of the Mak\_Tree?



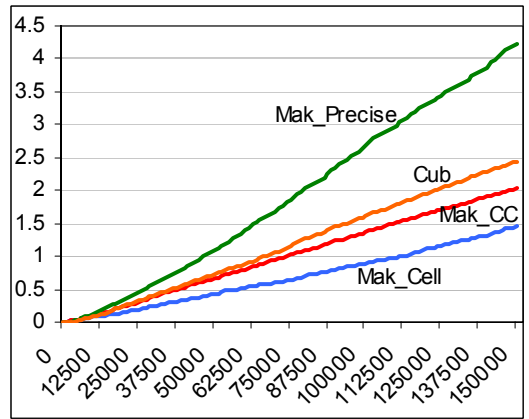
(a) AP-1



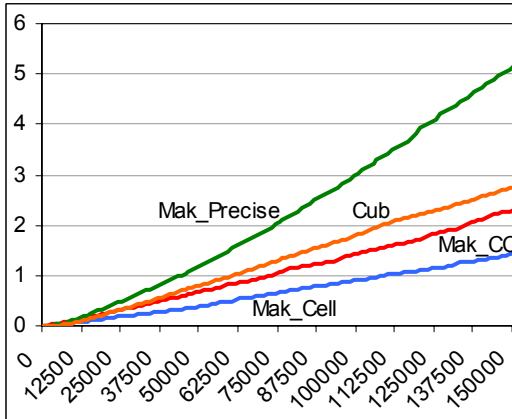
(b) AP-2



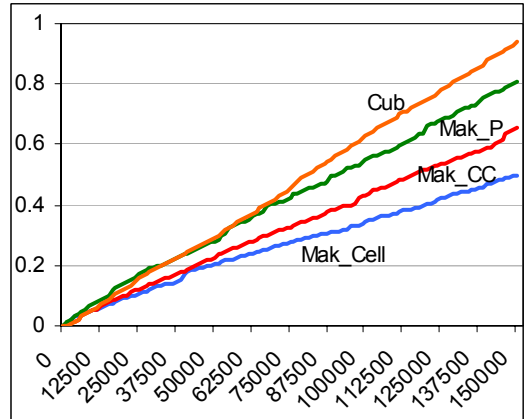
(c) AP-3



(d) AP-4



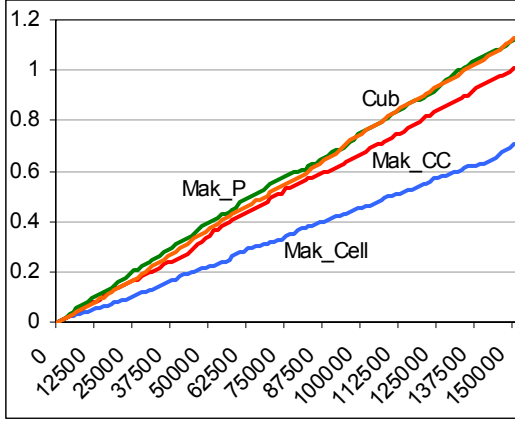
(e) AP-5



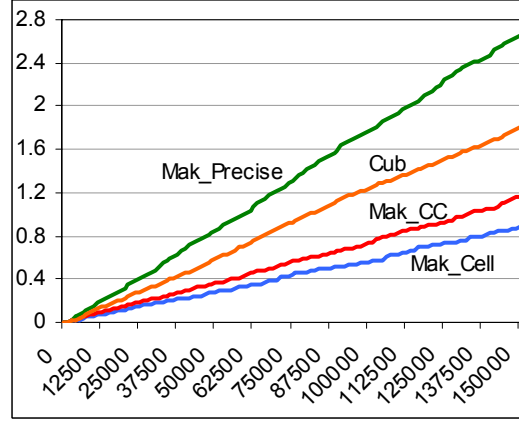
(f) AP-8

**Figure 86 — Average Cumulative Time Costs of Cell-Based Archiving Techniques and Precise Archiving Approaches ( $b = 0.001$ ,  $\kappa = 20$ ,  $\varphi = 50$ )**

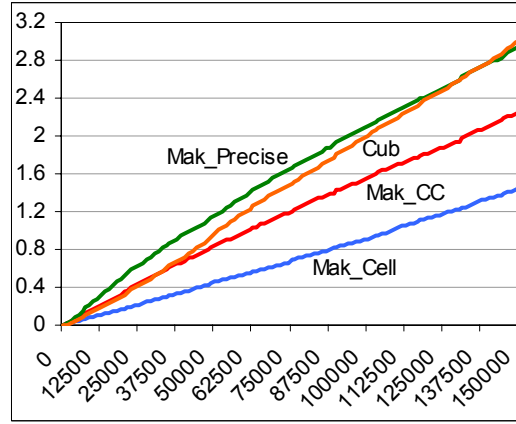
For all graphs, the *x-axis* represents the number of solutions presented to the archive and the *y-axis* is the average cumulative processing time in seconds. *Mak\_CC* refers to a complete cell-based Mak\_Tree with crowding,  $\kappa$  nearest-neighbour estimates, annotations and selection. *Mak\_Precise* is the equivalent data structure with individuals used in place of cells. *Mak\_Cell* is a base-line, with no extended features in the cell-based environment. The *Cub* results refer to performance in a batch-processed truncated archive of 50 using the two-neighbour cuboid method.



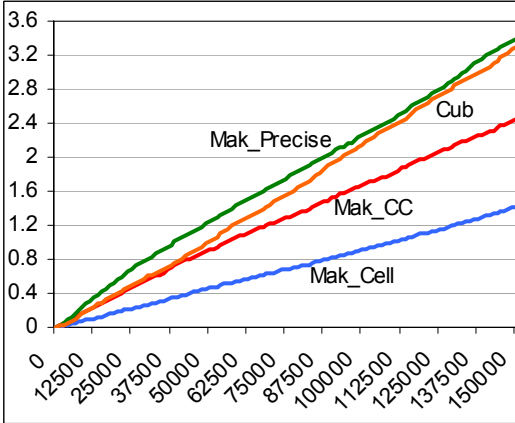
(a) AP-9



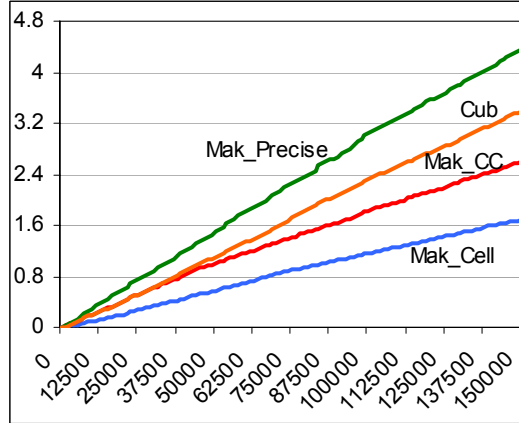
(b) AP-10



(d) AP-15



(e) AP-16

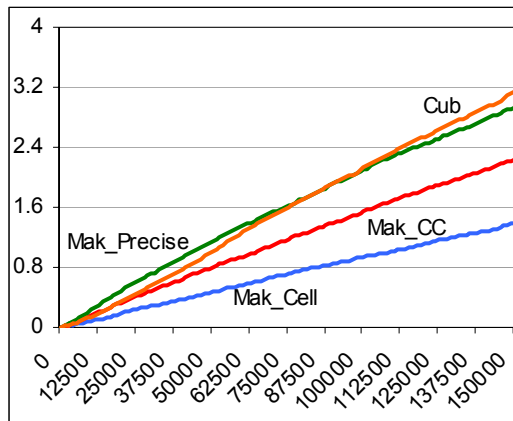
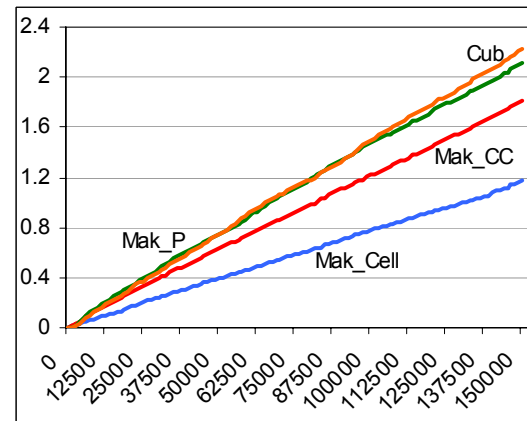


(f) AP-17

**Figure 87 — Average Cumulative Time Costs of Cell-Based Archiving Techniques and Precise Archiving Approaches ( $b = 0.001$ ,  $\kappa = 20$ ,  $\varphi = 50$ )**

For all graphs, the  $x$ -axis represents the number of solutions presented to the archive and the  $y$ -axis is the average cumulative processing time in seconds. *Mak\_CC* refers to a complete cell-based *Mak\_Tree* with crowding,  $\kappa$  nearest-neighbour estimates, annotations and selection. *Mak\_Precise* is the equivalent data structure with individuals used in place of cells. *Mak\_Cell* is a base-line, with no extended features in the cell-based environment. The *Cub* results refer to performance in a batch-processed truncated archive of 50 using the two-neighbour cuboid method.



(e) *AP-21*(f) *F-1*

**Figure 88 — Average Cumulative Time Costs of Cell-Based Archiving Techniques and Precise Archiving Approaches ( $b = 0.001$ ,  $\kappa = 20$ ,  $\varphi = 50$ )**

For all graphs, the  $x$ -axis represents the number of solutions presented to the archive and the  $y$ -axis is the average cumulative processing time in seconds. *Mak\_CC* refers to a complete cell-based Mak\_Tree with crowding,  $\kappa$  nearest-neighbour estimates, annotations and selection. *Mak\_Precise* is the equivalent data structure with individuals used in place of cells. *Mak\_Cell* is a base-line, with no extended features in the cell-based environment. The *Cub* results refer to performance in a batch-processed truncated archive of 50 using the two-neighbour cuboid method.

**Table 23 — Empirical Performance of Cell-Based Mak\_Trees and the Truncated Cuboid After 10,000 NSGA-II Evaluations**

When  $b$  is zero, non-cell-based systems are used. Results indicate the average time in seconds.

	Mak $b = 0$ $k = 20$ $\varphi = 50$	Mak $b = 0.001$ $k = 0$ $\varphi = 0$	Mak $b = 0.001$ $k = 20$ $\varphi = 50$	Mak $b = 0.005$ $k = 0$ $\varphi = 0$	Mak $b = 0.005$ $k = 20$ $\varphi = 50$	Mak $b = 0.01$ $k = 0$ $\varphi = 0$	Mak $b = 0.01$ $k = 20$ $\varphi = 50$	Cuboid $b = 0$ $k = 0$ $\varphi = 50$
<b>AP-1</b>	0.24	0.09	0.13	0.07	0.12	0.07	0.11	0.16
<b>AP-2</b>	0.10	0.06	0.06	0.08	0.08	0.07	0.15	0.04
<b>AP-3</b>	0.22	0.08	0.14	0.06	0.12	0.08	0.10	0.17
<b>AP-4</b>	0.13	0.08	0.09	0.07	0.10	0.07	0.10	0.08
<b>AP-5</b>	0.13	0.07	0.11	0.08	0.10	0.07	0.11	0.07
<b>AP-8</b>	0.07	0.05	0.05	0.04	0.05	0.03	0.07	0.05
<b>AP-9</b>	0.08	0.04	0.07	0.05	0.06	0.04	0.06	0.07
<b>AP-10</b>	0.15	0.06	0.08	0.05	0.06	0.04	0.05	0.11
<b>AP-15</b>	0.25	0.09	0.17	0.10	0.16	0.08	0.14	0.14
<b>AP-16</b>	0.27	0.08	0.19	0.10	0.17	0.09	0.16	0.19
<b>AP-17</b>	0.30	0.09	0.20	0.09	0.18	0.08	0.15	0.20
<b>AP-21</b>	0.23	0.08	0.16	0.09	0.16	0.08	0.14	0.15
<b>F-1</b>	0.17	0.08	0.14	0.07	0.12	0.09	0.14	0.15

**Table 24 — Empirical Performance of Cell-Based Mak\_Trees and the Truncated Cuboid After 50,000 NSGA-II Evaluations**

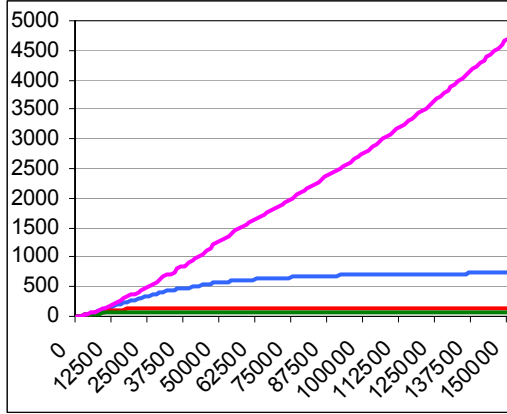
When  $b$  is zero, non-cell-based systems are used. Results indicate the average time in seconds.

	Mak	Mak	Mak	Mak	Mak	Mak	Mak	Cuboid
	$b = 0$ $k = 20$ $\varphi = 50$	$b = 0.001$ $k = 0$ $\varphi = 0$	$b = 0.001$ $k = 20$ $\varphi = 50$	$b = 0.005$ $k = 0$ $\varphi = 0$	$b = 0.005$ $k = 20$ $\varphi = 50$	$b = 0.01$ $k = 0$ $\varphi = 0$	$b = 0.01$ $k = 20$ $\varphi = 50$	$b = 0$ $k = 0$ $\varphi = 50$
<b>AP-1</b>	1.15	0.45	0.72	0.37	0.61	0.37	0.54	0.89
<b>AP-2</b>	0.98	0.40	0.61	0.38	0.51	0.37	0.57	0.67
<b>AP-3</b>	1.08	0.37	0.59	0.38	0.51	0.37	0.46	0.94
<b>AP-4</b>	1.10	0.43	0.66	0.42	0.58	0.41	0.47	0.73
<b>AP-5</b>	1.20	0.38	0.68	0.45	0.56	0.35	0.51	0.81
<b>AP-8</b>	0.28	0.20	0.23	0.17	0.20	0.14	0.27	0.29
<b>AP-9</b>	0.40	0.22	0.35	0.22	0.28	0.19	0.30	0.38
<b>AP-10</b>	0.86	0.28	0.36	0.29	0.38	0.30	0.35	0.59
<b>AP-15</b>	1.16	0.45	0.84	0.48	0.71	0.41	0.63	0.96
<b>AP-16</b>	1.24	0.46	0.91	0.49	0.76	0.44	0.67	1.01
<b>AP-17</b>	1.49	0.58	1.00	0.50	0.73	0.42	0.61	1.11
<b>AP-21</b>	1.15	0.47	0.80	0.44	0.71	0.37	0.59	1.01
<b>F-1</b>	0.75	0.40	0.64	0.42	0.61	0.47	0.60	0.74

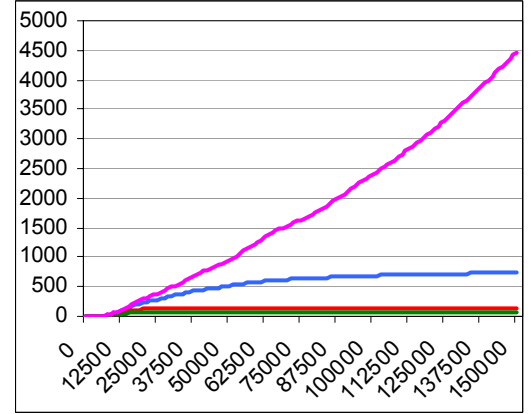
**Table 25 — Empirical Performance of Cell-Based Mak\_Trees and the Truncated Cuboid After 150,000 NSGA-II Evaluations**

When  $b$  is zero, non-cell-based systems are used. Results indicate the average time in seconds.

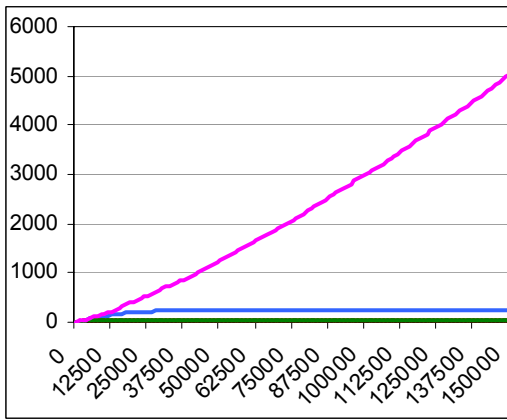
	Mak	Mak	Mak	Mak	Mak	Mak	Mak	Cuboid
	$b = 0$ $k = 20$ $\varphi = 50$	$b = 0.001$ $k = 0$ $\varphi = 0$	$b = 0.001$ $k = 20$ $\varphi = 50$	$b = 0.005$ $k = 0$ $\varphi = 0$	$b = 0.005$ $k = 20$ $\varphi = 50$	$b = 0.01$ $k = 0$ $\varphi = 0$	$b = 0.01$ $k = 20$ $\varphi = 50$	$b = 0$ $k = 0$ $\varphi = 50$
<b>AP-1</b>	3.90	1.49	2.22	1.44	1.77	1.31	1.71	2.58
<b>AP-2</b>	3.70	1.39	1.99	1.25	1.62	1.18	1.63	2.37
<b>AP-3</b>	3.70	1.19	1.71	1.17	1.53	1.11	1.30	2.72
<b>AP-4</b>	4.22	1.46	2.04	1.39	1.80	1.35	1.66	2.44
<b>AP-5</b>	5.15	1.43	2.29	1.48	1.66	1.35	1.78	2.76
<b>AP-8</b>	0.81	0.50	0.65	0.52	0.63	0.47	0.70	0.94
<b>AP-9</b>	1.12	0.71	1.01	0.69	0.86	0.66	0.85	1.13
<b>AP-10</b>	2.65	0.88	1.17	0.91	1.05	0.85	1.00	1.80
<b>AP-15</b>	2.95	1.45	2.25	1.41	1.95	1.32	1.68	3.01
<b>AP-16</b>	3.39	1.41	2.44	1.44	1.99	1.32	1.78	3.29
<b>AP-17</b>	4.36	1.69	2.60	1.52	2.05	1.32	1.84	3.40
<b>AP-21</b>	2.93	1.39	2.24	1.33	1.90	1.26	1.71	3.14
<b>F-1</b>	2.11	1.17	1.81	1.22	1.73	1.20	1.60	2.23



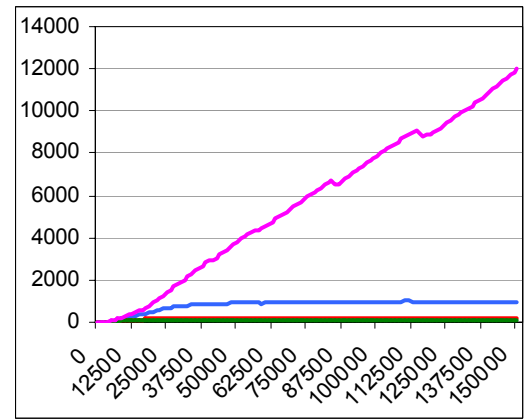
(a) AP-1



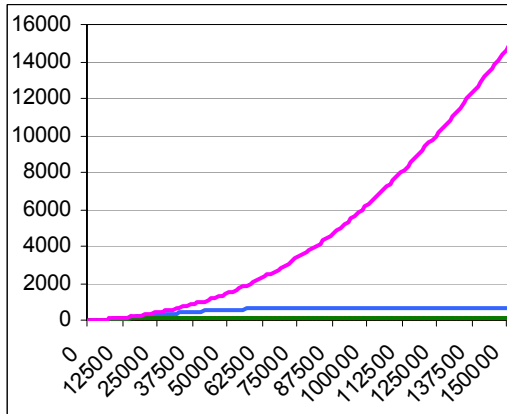
(b) AP-2



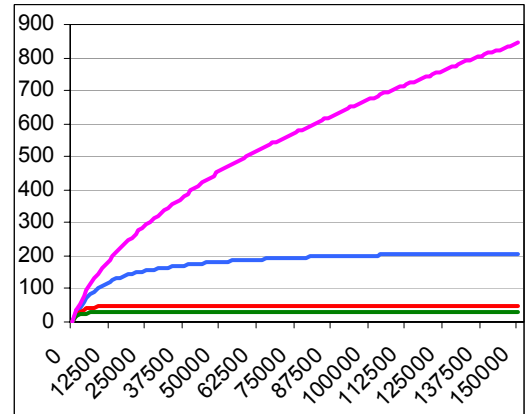
(c) AP-3



(d) AP-4



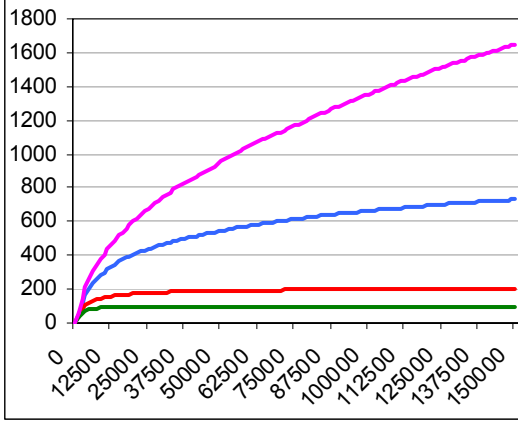
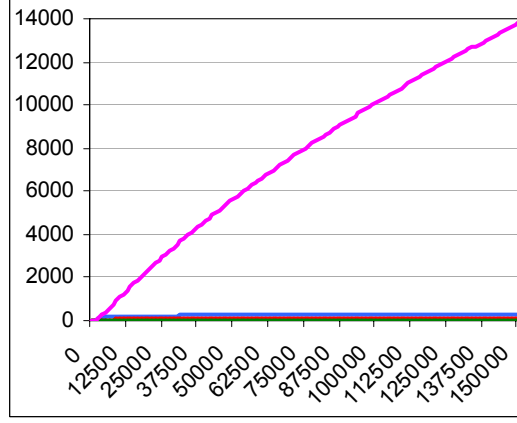
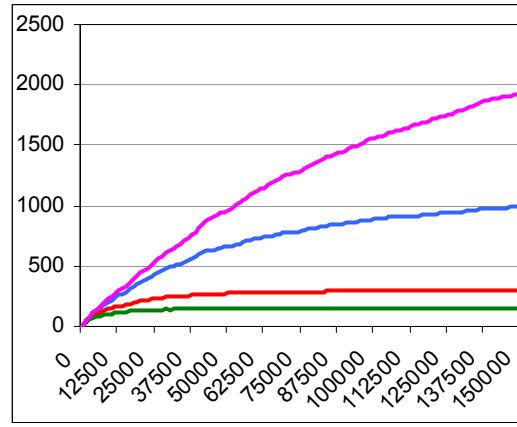
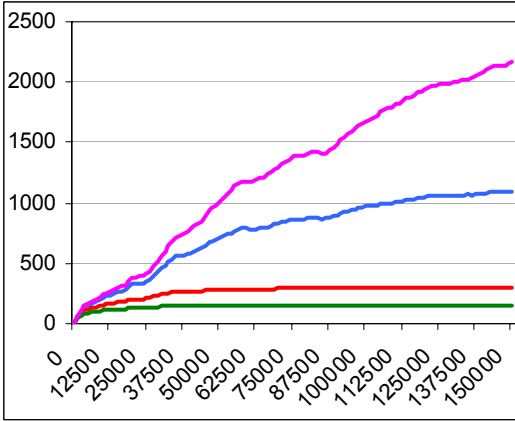
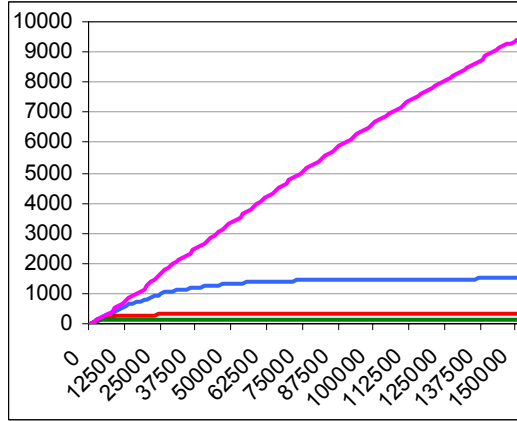
(e) AP-5



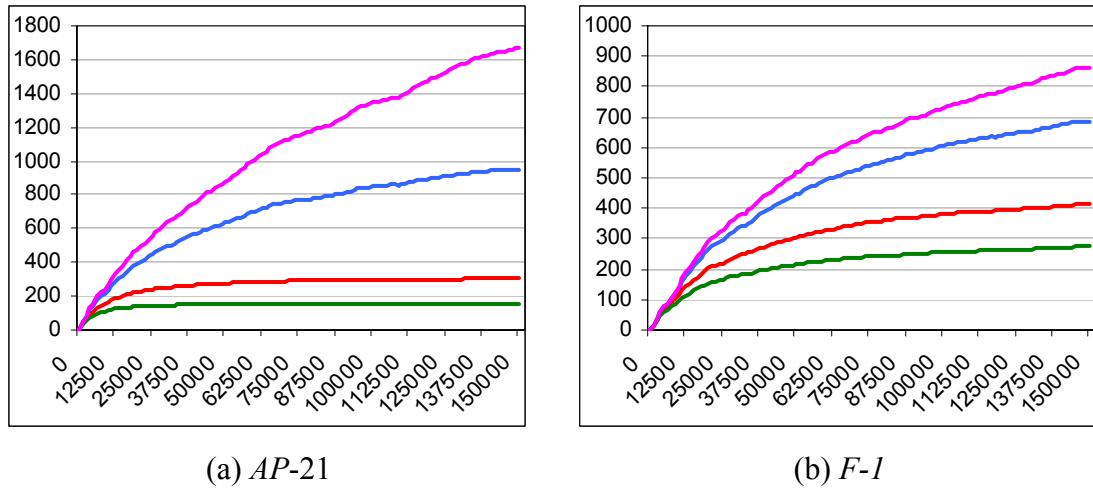
(f) AP-8

**Figure 89 — Average Number of Nodes Stored in Various Mak\_Tree**

For all graphs, the *x-axis* represents the number of solutions presented to the archive and the *y-axis* is the average number of nodes stored in a given Mak\_Tree. The highest node count is the Mak\_Tree, the second highest is the cell-based Mak\_Tree with  $b = 0.001$ , the third highest uses  $b = 0.005$ , and the lowest features  $b = 0.01$ .


 (a) *AP-9*

 (b) *AP-10*

 (d) *AP-15*

 (e) *AP-16*

 (f) *AP-17*
**Figure 90 — Average Number of Nodes Stored in Various Mak\_Trees**

For all graphs, the *x-axis* represents the number of solutions presented to the archive and the *y-axis* is the average number of nodes stored in a given Mak\_Tree. The highest node count is the Mak\_Tree, the second highest is the cell-based Mak\_Tree with  $b = 0.001$ , the third highest uses  $b = 0.005$ , and the lowest uses  $b = 0.01$ .

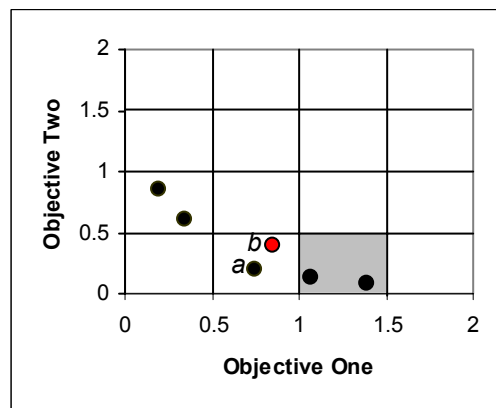


**Figure 91 — Average Number of Nodes Stored in Various Mak\_Trees**

For all graphs, the *x-axis* represents the number of solutions presented to the archive and the *y-axis* is the average number of nodes stored in a given Mak\_Tree. The highest final node count is the Mak\_Tree, the second highest is the cell-based Mak\_Tree with  $b = 0.001$ , the third highest uses  $b = 0.005$ , and the lowest uses  $b = 0.01$ .

### 8.3.5 LIMITATIONS IN THE CELL-BASED APPROACH

In practice, cell-based archiving is similar to, and subject to many of the same deficiencies as, the  $\varepsilon$ -dominance approach explored by Laumanns *et al.* [98]. In particular, the use of cell-based dominance in the construction and maintenance of an elite set is prone to both permitting the inclusion of weak solutions and excluding preferable proposals. Consider Figure 92: solution *a* dominates *b*, but since they both reside in the same non-dominated cell they will both be included as members of the elite set. Additionally, the shaded region represents a dominated cell whose

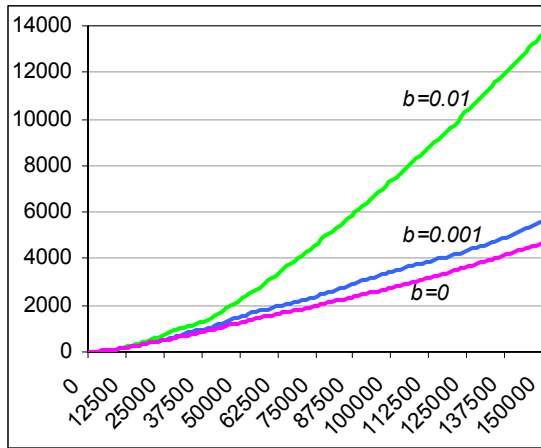


**Figure 92 — An Example of the Limitations of Cell-Based Mak\_Trees**

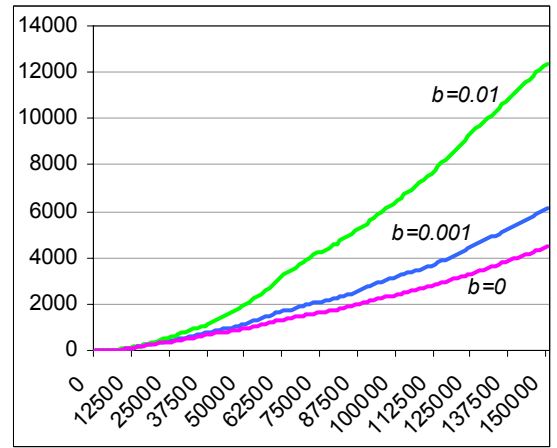
Points residing in non-shaded cells are members of the non-dominated front according to cell-based elite archiving. Points in shaded cells are not included in the front.

constituents must be rejected, despite the fact that each constituent is a non-dominated proposal.

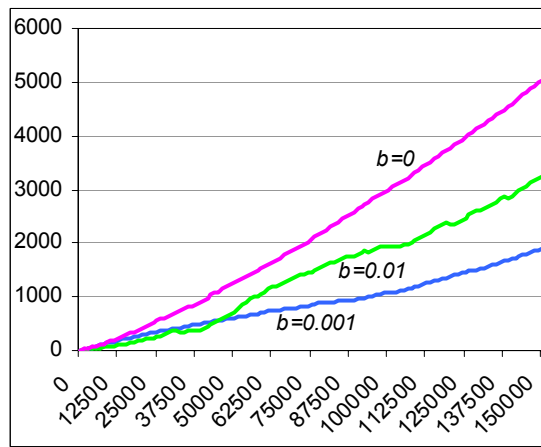
To illustrate the potential for poor performance in practice, the total number of stored solutions (this time including replicated solutions) in a precise unbounded Mak\_Tree was compared with the number of proposals stored in cell-based Mak\_Trees of differing granularity across a rich test suite (see Section 6.4.2 for an outline of how the proposals were generated by NSGA-II). As illustrated in Figure 93, Figure 94 and Figure 95 (pages 231–233), the cell-based archive has poor fidelity with the true elite unbounded set, with results indicating both the inclusion of weak solutions (see *AP-1*, *AP-2*, *AP-5*, *AP-8*, *AP-9* and *AP-10*, in particular) and the exclusion of strong proposals (as best illustrated in *AP-3*, *AP-4* and *AP-16*).



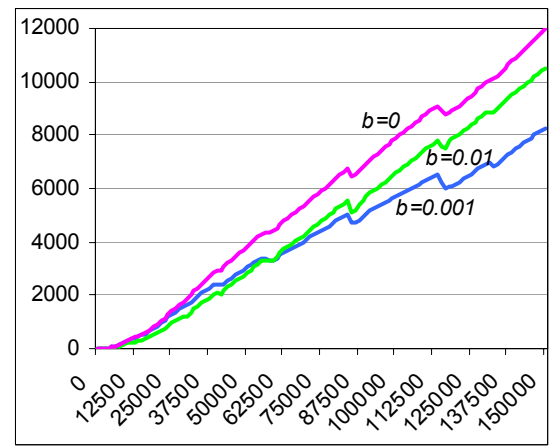
(a) AP-1



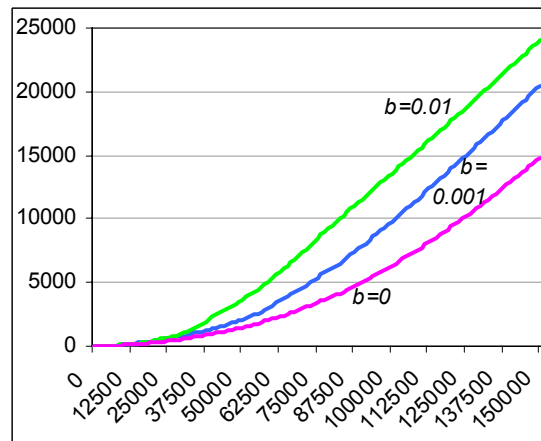
(b) AP-2



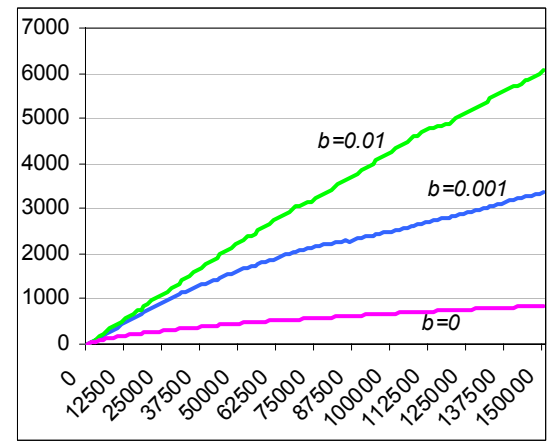
(c) AP-3



(d) AP-4

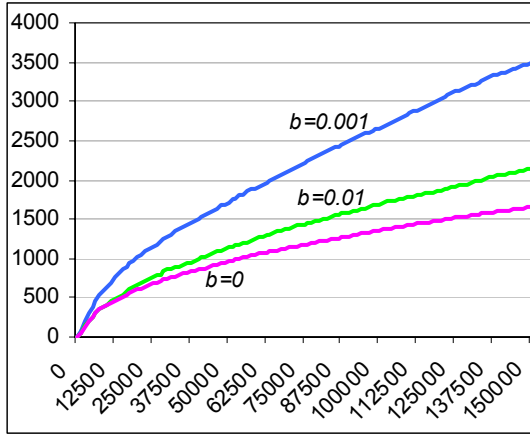


(e) AP-5

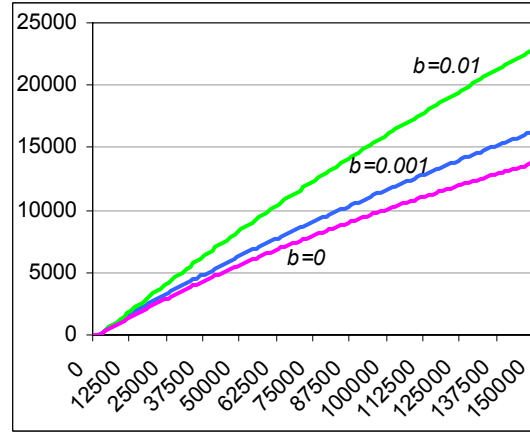


(f) AP-8

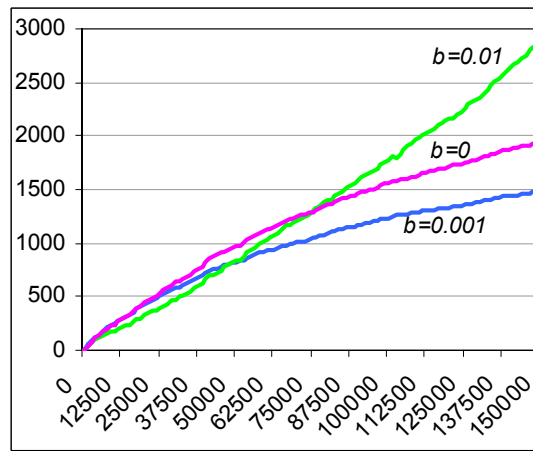
**Figure 93 — Average Number of Solutions Stored in Various Mak\_Trees**  
 $b$  is the size of the cell; when  $b=0$ , non-cell-based Mak\_Trees are used.



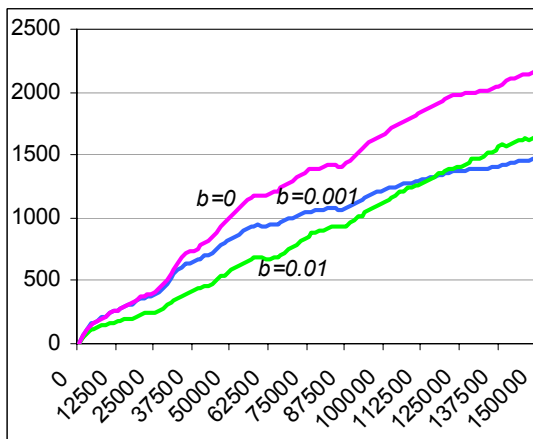
(a) AP-9



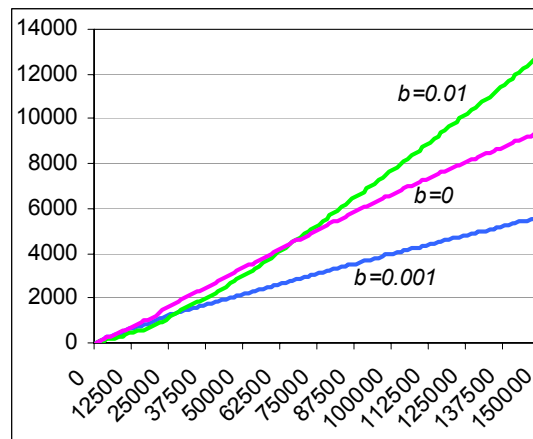
(b) AP-10



(d) AP-15



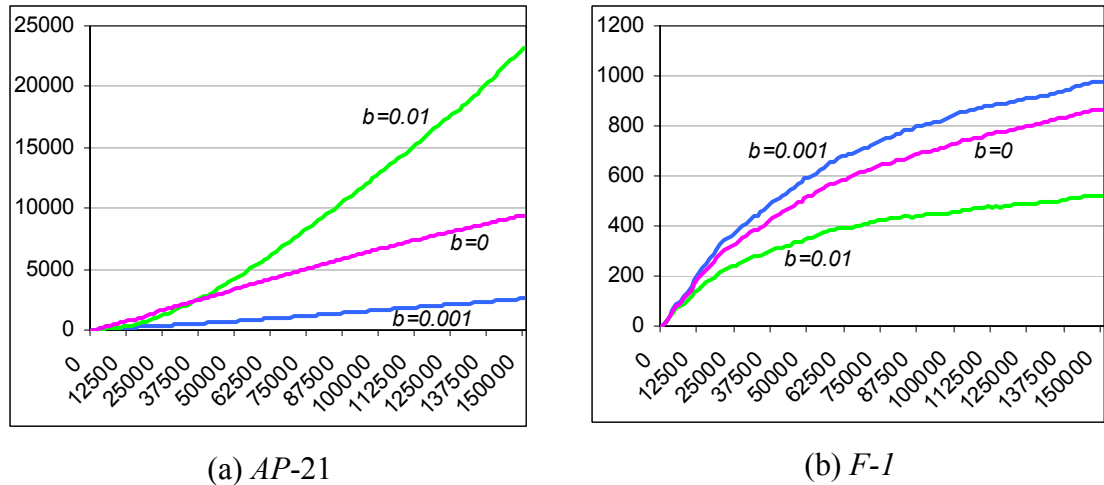
(e) AP-16



(f) AP-17

**Figure 94 — Average Number of Solutions Stored in Various Mak\_Trees**  
 $b$  is the size of the cell; when  $b=0$ , non-cell-based Mak\_Trees are used.





**Figure 95 — Average Number of Solutions Stored in Various Mak\_Trees**  
 $b$  is the size of the cell; when  $b=0$ , non-cell-based Mak\_Trees are used.

The consequence of such results is that an archive constructed with cells can offer no guarantee as to the quality of its members — as an elite store around which an entire optimisation algorithm can be built, it is therefore of debatable value. This work therefore proposes that the cell-based Mak\_Tree outlined in Section 8.3.1 should only be used when the efficiency of the data structure is of primary concern. When accuracy is of greater import, this thesis recommends the use of two tree data structures.

### 8.3.6 AN ACCURATE CELL-BASED APPROACH

Recall that a basic Mak\_Tree guarantees that its constituents are non-dominated with respect to every solution presented thus far. With this in mind, the cell-based Mak\_Tree should satisfy a similar constraint, lest it become prone to the degenerative behaviours familiar to truncated techniques. Unfortunately, the simple approach formed in Section 8.3.1 — efficient though it might be — cannot claim such efficacy. A simple approach that rectifies this deficiency is to capitalise on two data structures: one to ensure the non-dominance of members and another to maintain cell-based crowding information.

In this case, the basic Mak\_Tree is extended such that nodes contain individuals (as per Section 6.3) *and* a link to the objective-space cell that houses them. Cells are stored in a simple binary tree ordered by an arbitrarily chosen objective-space coordinate, with crowding annotations included if cell-based selection is necessary. Depending on intended usage, each cell can contain the housed solutions, a reference

to the solutions or simply a count. Deletion from the simple cell-tree occurs only when all solutions residing in a cell have been removed. Interestingly, if merged density approximations are required — with  $\kappa$  nearest-neighbours estimates between individuals and cell crowding to be employed — the only required structural change is the replacement of the basic Mak\_Tree with the extended technique.

Though it seems reasonable to assume that the use of two data structures would severely inhibit the overall performance of the unbounded archive, the computational burden is surprisingly low. As with the previous complexity studies reported in this thesis, it is best to decompose performance into a number of distinct cases — namely, insertion of dominated, non-dominating and dominating solutions.

Under the insertion of a dominated solution, the complexity is derived entirely from the underlying Mak\_Tree structure. Since a dominated solution will be identified during standard navigation of the Mak\_Tree, the cost of rejecting a dominated solution is only  $O(\log n)$ .

When inserting a non-dominating, yet non-dominated, individual, both the Mak\_Tree and the simple cell-tree must be explored, incurring a cost of  $O(\log n + \log c)$  (where  $c$  is the number of stored cells). Since the worst possible configuration of cells in this case is for each individual to belong to precisely one cell, the worst-case cost is  $O(\log n + \log n) = O(\log n)$ . Cell-based annotations, at worst, increase this complexity to  $O(\log n + 2\log n) = O(\log n)$ . If  $\kappa$  nearest-neighbours estimations and annotations are also required, the total complexity overhead of inserting non-dominated non-dominating solutions will rest at  $O(\log n + \kappa)$ .

The insertion of a dominating proposal will require the deletion of  $\eta$  members from the Mak\_Tree, the removal of  $\eta_{ci}$  individuals from cells and the extraction of up to  $\eta_{cc}$  cells themselves, leading to a computational overhead of  $O(\eta \log n + \eta_{ci} + \eta_{cc} \log c)$  when crowding estimations and annotations are not considered. The worst-case deletion is when each dominated individual results in the loss of a cell, leading to a run-time complexity of  $O(\eta \log n + \eta + \eta \log c)$ . Since,  $c \geq \eta$  and  $c \leq n$ , the worst-case arrangement of cells is again when each individual is stored in a single cell, leading to  $O(\eta \log n + \eta + \eta \log n) = O(\eta \log n)$ . When cell-based crowding and annotations are to be included, pushing annotation updates from each cell carries an

additional  $O(\eta_{cc} \log c)$  cost, that results in a worst-case of  $O(\eta \log n + \eta + 2\eta \log n) = O(\eta \log n)$ . If  $\kappa$  nearest-neighbours scores and annotations are required, the total worst-case complexity for the insertion of dominating proposals is  $O(\eta \log n + \kappa^2)$  when  $\kappa < \eta$  and  $O(\eta \log n + \eta\kappa)$  otherwise. Following the same methodology as described in Section 8.2, the amortised cost of this procedure reduces to  $O(\log n + \kappa)$ .

Thus, the addition of a data structure for the maintenance of cell-based density information does not affect the big-oh performance complexity of the fully-featured Mak\_Tree. The resulting hybrid methodology, which enables merged crowding estimation without the potential frontal degradation, is an interesting technique in addressing the bias/variance dilemma and rests as an avenue for future investigation.

## 8.4 CONCLUSIONS

While the basic Mak\_Tree illustrated impressive performance with respect to the maintenance of a truly unbounded bi-objective set, it lacked many of the features necessary for use in a practical contemporary multiobjective optimiser. In particular, there was no provision for efficient unbounded frontal density analysis and no mechanism for selection. The extended Mak\_Tree addresses these limitations by offering facilities for density estimation based on  $\kappa$  nearest-neighbours, cuboid and cell-based estimates, with a unique annotation system enabling periodic selection. Theoretical and empirical evidence suggests that the extended Mak\_Tree delivers this additional functionality in an extremely efficient way — carrying at most an additional  $O(\kappa)$  cost per insertion under amortised time analysis and frequently outperforming even heavily truncated ( $n = 50$ ) archiving techniques in empirical studies. By providing a system that matches the needs of modern algorithm designers, avoids the dangers of truncation and achieves impressively fast results, the extended Mak\_Tree should make unbounded bi-objective archiving a practical reality.

# Chapter

# 9

## Liberating Bounded Optimisers

*“Be not afraid of growing slowly, be afraid only of standing still.”*

*Chinese Proverb*

*“Is there anything in life so disenchanting as attainment?”*

*Robert Louis Stevenson —  
Writer and Poet*

## **9 LIBERATING BOUNDED OPTIMISERS**

Although preceding chapters have established the advantages of using the specialist Mak\_Tree for unbounded bi-objective archiving, there remain questions as to how the approach can be successfully integrated into evolutionary algorithms and whether such integration yields suitably impressive results to make the effort worthwhile. Thus, in this chapter, the Mak\_Tree is incorporated into a diverse range of popular and contemporary generalist multiobjective evolutionary algorithms with a view to establishing the practical advantages that unbounded archiving and bi-objective specialisation can yield. By using the Mak\_Tree in a variety of roles — from simple reference set to active population — the chapter can also be used as a primer for researchers looking to integrate the unbounded techniques into their own algorithms.

### **9.1 PERFORMANCE ANALYSIS**

Before examining the performance of Mak\_Tree powered evolutionary algorithms, it is necessary to first determine precisely how that performance is to be measured. Perhaps surprisingly, this is an inherently difficult proposition.

As suggested in Section 7.1, performance analysis in a multiobjective environment is complex because the multiobjective optimiser is itself endeavouring to satisfy multiple, potentially conflicting, goals [92] — specifically, the derived front should be accurate (near to the Pareto optimal front), diverse and well-spread. But how does an accurate, though poorly spread, front compare with a diverse, inaccurate result set? If two fronts are equally accurate, but one has wide reaching extent while the other is evenly distributed, which is preferable? Until recently, such questions remained largely unaddressed.

Early studies (such as [2, 88, 142, 182, 215]) generally featured simple graphical representations of frontal sets (often little more than a plotting of the members of the final population) coupled with rudimentary support in the form of metrics designed to elucidate the extent, cardinality, diversity and/or accuracy of the prevailing front. The problem with this approach to analysis is two-fold: interpretation of such basic visual output is potentially biased, bereft of statistical rigour and difficult to assess in

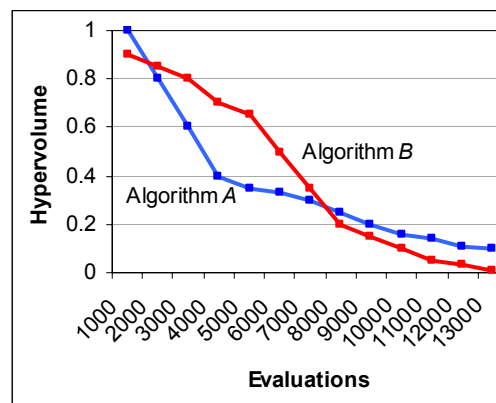
all but clear-cut cases of superiority<sup>65</sup>; and the basic numerical performance metrics are potentially misleading and prone to inconclusive results. In particular, unless all measures point to a single algorithm, it is difficult to indicate the preferable system in an unbiased manner and, more significantly, even if all tests suggest a preferable algorithm on a given function, there is no guarantee that such a result is correct. For instance, the output of two optimisers could be analysed with the cardinality-based ONVG (Overall Non-dominated Vector Generation) measure [216], the accuracy based generational distance metric [216] and the distribution indicator proposed by Deb [217], and have the weakly dominated front declared the unanimous winner in a comparative case study (Knowles *et al* [218] illustrate that such worrying variations are true even in non-pathological cases). Clearly, such an outcome is inherently flawed — the preferability of dominating fronts rests as a foundation of contemporary multiobjective optimisation thought. As recognised by Knowles *et al.* [218, 219], the key to successful performance metrics therefore lies not in highlighting the individual aspects of given fronts, as was fast becoming the *de facto* standard, but in producing comparative measures which are compatible with the Pareto dominance relation.

Since the overwhelming majority of diversity, cardinality and spread measures fail the Pareto compliance test, the implication is that combinatorial metrics, which seek to produce a single output based on the multi-faceted Pareto front, loom as the most likely candidates for successful performance analysis. Contemporary research supports this claim — hypervolume [137], epsilon [100] and the  $R_2$  and  $R_3$  [186] metrics are all Pareto compliant and are gaining increasing recognition in the literature (see, for instance, [91, 145, 218-221]). Still, the approaches are not without their limitations. In particular, combinatorial metrics typically feature some form of implicit bias — the effect of which is often difficult to gauge. However, given a suitable collection of combinatorial metrics used in league with other analysis techniques, the effects of such bias are likely to be minimised and the metrics rest as the current best available option. A more pressing concern then, is how best to use these performance indices.

---

<sup>65</sup> As will be explored later, attainment surfaces and frequency matrices illustrate that it is possible to have meaningful visualisations in a multiobjective context — but even these should play no more than a supplemental role in performance analysis.

While there can be little doubt that Knowles *et al.* [218, 219] (and Hansen and Jaszkiewicz [186] before them) were fundamental in moving the field towards the acceptance of better (Pareto-compliant) performance metrics, their impressive work, and the work it clearly inspired (such as [91] and [145]), inadvertently and indirectly perpetuates a standard in multiobjective optimisation that is rarely questioned (save for the interesting work by Deb and Jain [103] and Laumanns *et al.* [102]) — namely, that performance is best gauged at a single point in the run ([81, 86, 137, 142, 143, 200, 222, 223] are a small sample of the many works that display this trend). The truth, of course, is that this is patently incorrect — simple, non-pathological cases such as the one illustrated in Figure 96 show that results can be fundamentally altered depending on where the optimiser output is sampled (*A* is the better optimiser early in the run; *B* is preferable later — perhaps because it is superior at distributing solutions along the Pareto front). By sacrificing analysis at multiple points in the run, conventional studies are not only producing potentially misleading results, but they are missing an opportunity to extract possibly interesting observations about the development of a run, which may shed light on both the nature of the problem and the optimiser. As a consequence, this thesis capitalises on Pareto compliant metrics that illustrate the performance of each optimiser at periodic intervals, with attainment-based visualisation techniques used to supplement these results where appropriate.



**Figure 96 — An Example Run-Time Performance Analysis**

Note that the relative performance of Algorithm *A* and Algorithm *B* varies depending on which point in the run the results are compared.

### 9.1.1 SELECTED PERFORMANCE METRICS

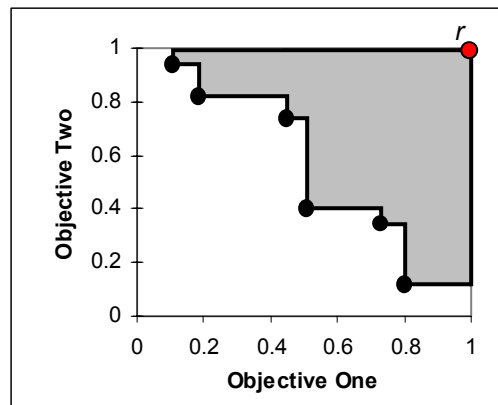
With the motivation for performance measure selection now in place, it is necessary to further explore the specific approaches chosen — namely, the hypervolume ( $I_H$ ) [137], epsilon ( $I_{\varepsilon_+}$ ) [100], attainment surface [224] and frequency matrix metrics<sup>66</sup>.

#### 9.1.1.1 THE HYPERVOLUME INDICATOR

Given a point  $r$  which is dominated by all members of a frontal set  $S$ , it is possible to calculate the hypervolume ( $I_H$ ) of space that is dominated by members of  $S$ , but not dominated by or incomparable with  $r$  (see Figure 97 for an illustration). With respect to performance, if  $S$  is the prevailing front of an optimiser, the larger the resultant hypervolume, the better the algorithm has performed. However, it would be convenient if the hypervolume was tending towards some fixed optimal value (say, zero), so that its proximity to “ideal” performance could be gauged. If an optimal reference set  $W$  is known, then this relative hypervolume measure ( $I_H$ ) can be given as:

$$I_H(S) = I_H(W) - I_H(S) \quad (80)$$

For the purposes of this work, the reference set  $W$  is produced from multiple long runs using a rich set of optimisers (including, but not limited to the optimisers and



**Figure 97 — An Example Hypervolume**

The shaded area, bounded by the prevailing front ( $S$ ) and the reference point  $r$ , represents the region from which the hypervolume is derived.

<sup>66</sup> In the interests of brevity and due to time constraints, the  $R_2$  and  $R_3$  metrics were not employed in this study. An important area of future work lies in the investigation of performance under such measures.



runs used for the analyses of this thesis) such that  $W$  contains no members that are dominated by any proposal in the set  $S$ . The reference point  $r$  is derived from the same set of runs and is the composite of the worst (highest) value for objective one and the worst value for objective two — in this case, there is no member of any set  $S$  that does not dominate  $r$ .

Researchers looking to use the same data for comparative purposes are invited to contact the author of this work via e-mail at Adam.Berry@utas.edu.au.

### 9.1.1.2 THE ADDITIVE EPSILON INDICATOR

The unary additive epsilon performance metric indicates the minimum offset value  $\varepsilon$  that must be applied to all points in reference set  $W$  to make  $W$  weakly dominated by frontal set  $S$ :

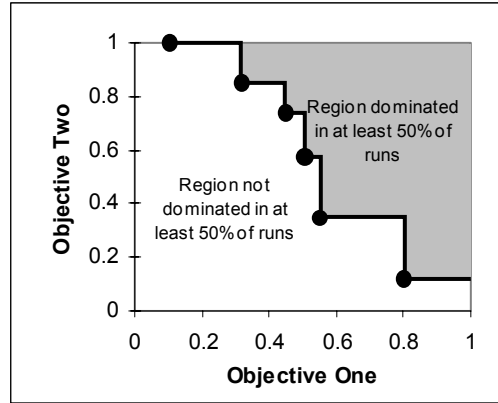
$$\begin{aligned} \min \{ \varepsilon \in \mathbb{R}, \exists s \in S : (s \preceq_{\varepsilon} w) \forall w \in W \} \\ \text{where } s \preceq_{\varepsilon} w = (s_a \leq (w_a + \varepsilon)) \forall a \in \{1, \dots, \beta\} \end{aligned} \quad (81)$$

As with the hypervolume measure, this thesis uses  $W$  as a goal set for a particular test problem (derived as in Section 9.1.1.2), and  $S$  as the output of a given optimiser. Thus, the explicit goal of an optimiser in this context is to minimise the value of  $\varepsilon$  over the course of a run.

### 9.1.1.3 50% ATTAINMENT SURFACES

Given a collection of output fronts  $C$  taken from  $|C|$  distinct optimiser runs at an identical evaluation point  $e$ , the 50% attainment surface represents the median attained optimal front over those  $|C|$  runs after  $e$  evaluations. That is, the 50% attainment surface illustrates the region of objective-space that is at least weakly dominated by the prevailing optimal front on at least 50% of the runs (see Figure 98).

As with most visual measures, attainment surfaces are at their most powerful when there is a clearly discernable difference between the produced fronts. In these instances, the pair-wise comparisons can elucidate differences in the median distribution, extent and accuracy of produced fronts, and act as an important supplement to the numerical findings illustrated by epsilon and hypervolume



**Figure 98 — An Example 50% Attainment Surface**

The dark line represents the attainment surface for this example — it describes the median optimal front produced by the optimiser over a number of distinct runs.

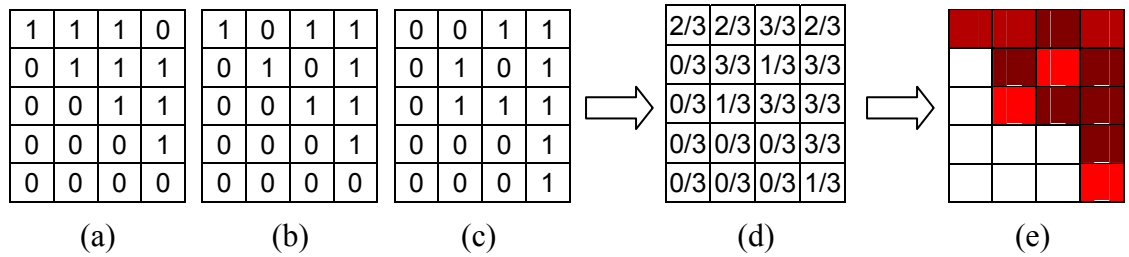
indicators. However, the 50% attainment surface is narrow in focus and provides little insight into the behaviour of an algorithm leading up to convergence on an optimal front. A visual approach that is better suited to this task is the frequency matrix.

#### 9.1.1.4 FREQUENCY MATRICES

Frequency matrices illustrate the regularity with which a given optimiser achieves particular regions of the objective-space across multiple runs. Obviously there are a number of ways to derive this information, but for the purposes of this thesis the results are reliant on the sub-division of the objective-space into a static grid  $G$  (the dimensions of which are defined in Appendix F.2). Each cell in  $G$  simply represents a frequency based on the number of runs in which the corresponding objective-space region was achieved:

$$\forall c \in G, \text{frequency}(c) = \frac{\text{runsOccupied}(c)}{\text{totalRuns}} \quad (82)$$

with colour coding of these values employed to produce a more accessible spectral graph (see Figure 99 for an example and Section 5.2.1 for a more detailed description).



**Figure 99 - An Example Frequency Matrix**

(a), (b) and (c) represent independent runs; (d) represents the frequencies of attaining each objective-space cell; (e) is a spectral representation of the frequency matrix in (d).

For pair-wise comparisons, simple matrix-subtraction is applied, such that the frequency matrix formed by the first optimiser is subtracted from that which is formed by the second. In this case, positive values reflect regions that were more frequently found by optimiser one, while negative values suggest areas that were more often identified by optimiser two. Again, colour coding can emphasise these differences in a graphical form (for this chapter, positive values are assigned a red palette, while negative values are assigned a blue palette).

The key to the power of frequency matrices is that they are capable of illustrating trends in algorithmic performance — not just along the optimal front, but throughout the course of a run. As such, the resultant graphs can give visual clues as to search bias, convergence points and inefficiencies. Moreover, the graphs provide insight into the nature of the problem at hand — indicating such important factors as deceptive regions (and their strength), false fronts and isolated portions of objective-space. Clearly, it is difficult to achieve such results in anything other than a visual analysis and this capacity alone renders frequency matrices an important component of any bi-objective comparative study.

### 9.1.2 METHODOLOGY

With the selected performance measures in place it is necessary to now explore how they are to be used in the context of this thesis, particularly with respect to the data being analysed and the way in which results are to be expressed.

### **9.1.2.1 DATA ACQUISITION**

A total of sixty independent runs per optimiser are performed for each test function, with data sampled at regular intervals of one thousand evaluations until termination occurs. The termination point itself is arbitrarily chosen, but is never less than seven thousand evaluations and should be sufficient to explore the behaviour of each optimiser (as will become obvious in subsequent sections). All optimisers use the mutation scheme described in Appendix B.2 and, where necessary, Simulated Binary Crossover (SBX [225]).

To enable statistical analysis, the first twenty results are used exclusively for hypervolume performance analysis, the second twenty are sourced for frequency matrices and the corresponding attainment surfaces, and the final set of results are used with epsilon metrics. Without such a division, Bonferroni corrections (at least) would be required when exploring multiple statistical inferences, which can severely weaken the validity of any rigorous study (for more details on the behaviour of multiple inferences under a single sample set, the interested reader is directed to Knowles *et al.* [218]).

The selected test functions are extracted from the *AP* test suite and offer a disparate set of problem features. Since not all of the basic evolutionary algorithms under consideration in this section are expressly designed to address the unique properties of noise, dynamic problem spaces and side-constraints, *AP*-1,2,3,4,5,15,16,17 and 21 are selected<sup>67</sup>. These nine functions alone are sufficient to test the performance of the evolutionary algorithms under the presence of concave, convex, mixed-shape and disconnected optimal fronts, deception, isolation, multi-frontality, zero-utility gradients and non-separability. The results should therefore be indicative of optimiser use across a far-reaching set of potential domains.

### **9.1.2.2 OUTPUT REPRESENTATION**

First order analysis of the hypervolume scores for a given set of evolutionary algorithms is provided on a per-function basis, with line graphs illustrating the average cumulative performance of each algorithm at 1000 evaluation intervals. The same methodology is applied in the production of averaged epsilon graphs. These

---

<sup>67</sup> Note that *AP*-14 would be a valid addition to this collection. A minor error in its original implementation is the primary reason for its exclusion here.

online performance analyses depict the progression of given optimisers on each problem and can elucidate both pre- and post-frontal-convergence performance.

First *and* second order analyses of hypervolume and epsilon scores are provided by simplified box-plots<sup>68</sup> at specific evaluation points (7,000 evaluations for *AP-5*; 10,000 evaluations for all other functions). Importantly, such a representation succinctly describes both the variability and quality of results with respect to the chosen performance indicator at a given point in time and is well matched with the non-parametric Kruskal-Wallis-based [202, 203] significance tests used in this and all subsequent chapters.

Note that the complete set of two-tailed pair-wise Kruskal-Wallis tests (produced via the assessor package provided in the PISA suite [204, 205]) for hypervolume and epsilon metrics are provided in Appendix E, with each section of results offering relevant portions of these tables to maximise clarity. Whenever a pair-wise statistical comparison returns a result (*p-value*) of less than 0.05, the difference in end-of-run performance is considered *significant* (at the 5% level).

Attainment graphs and frequency matrices provide pair-wise performance comparisons when required and are intended as a supplement to the epsilon and hypervolume measures. In general, visually interpreting multiple (>2) algorithms with either of these approaches is difficult and of limited value<sup>69</sup>, so both visual analysis techniques are used primarily in paired scenarios.

## 9.2 MAK\_TREE INTEGRATION

In principle, the application of unbounded bi-objective archiving in place of existing truncated variations should yield improved crowding estimation (Section 6.2.1.3), reduce the potential for frontal degradation (Section 6.2.1.1 and Section 6.2.1.2) and offer a platform for the accurate determination of solution dominance. However, the question remains as to whether such improvements carry a real practical effect and whether existing generalist algorithms can benefit from the specialisation process. As such, a diverse set of both contemporary and popular evolutionary algorithms are extended, tested and analysed — with a view to establishing the impact of Mak\_Tree

---

<sup>68</sup> The simplified box plots used in this study feature *true* minimum and maximum values, without filtering of outliers.

<sup>69</sup> Unless there are marked differences between all tested algorithms, which is typically unlikely.

integration and to offer prototypical methodologies for harnessing unbounded archives in a variety of distinct ways (including simple replacement, passive referencing and hybridisation). In particular, PAES [79], PESA [136], NSGA-II [144] and SPEA2 [81] are each compared with an equivalent unbounded Mak\_Tree variation to emphasise points of improvement and differentiation. If improvement can be seen across these varied algorithmic techniques, it can be taken as a strong indicator of the power that the Mak\_Tree can impart.

### **9.2.1 THE PARETO ARCHIVED EVOLUTIONARY ALGORITHM**

Since PAES [79] was expressly designed to be “the simplest possible non-trivial [multiobjective] evolutionary algorithm” available [143] and was amongst the first wave of techniques to weave a truncated archive into the evolutionary process, it represents an ideal starting point for Mak\_Tree integration and bi-objective specialisation. Since the effectiveness of the PAES algorithm is so heavily predicated on the successful use of its archived population (as will be seen in the following section), the benefits (and potential disadvantages) of Mak\_Tree replacement should be clearly identifiable.

#### **9.2.1.1 DESCRIBING THE PAES ALGORITHM**

At its core, PAES is little more than a basic hill-climbing algorithm (see Algorithm 17) — employing a simple (1+1) Evolutionary Strategy (ES). That is, the evolutionary algorithm simply picks the *better* of the parent and its child as the parent for the next generation, with basic asexual reproduction ensuring ongoing exploration. The complexity of the approach therefore comes from the way in which the “better” relation is defined. Clearly, a dominant agent is preferable over a dominated agent, but in the case of incomparability the pair-wise comparison is more difficult to resolve. For PAES, the key to addressing this problem is the use of a truncated archive and corresponding objective-space grid. Since the archive contains an approximation of the prevailing optimal front, locally incomparable solutions can be contrasted with frontal members to determine which is the preferable choice with respect to a larger population. As such, a candidate that is dominated by some member of the archive is always rejected, since it is not furthering the cause of the greater population. In the alternative case, where both candidates have membership

in the prevailing front, the solution that resides in the least-occupied objective-space cell is granted the right to breed.

---

**Algorithm 17 — The Basic PAES Algorithm**

---

19: <i>primary</i> := generateRandomSolution()	Start the search at a random location.
20: while (terminationConditionMet() ≠ true)	
21: <i>child</i> := asexualReproduction( <i>primary</i> )	Generate the child solution.
22:   if ( <i>child</i> < <i>primary</i> )	If the child dominates the current
23:     updateArchive( <i>child</i> )	solution, update the archive,
24:     updateGrid( <i>child</i> )	update the grid and
25: <i>primary</i> := <i>child</i>	set the child as the new primary solution.
26:   else if ( <i>primary</i> ~ <i>child</i> )	If the child is incomparable with the
27:     if ( $\nexists x \in \text{Archive} : x < \text{child}$ )	current solution and is not dominated by
28:       updateArchive( <i>child</i> )	the archival members, update the archive
29:       updateGrid( <i>child</i> )	and the grid.
30:     if ( <i>grid.crowd</i> ( <i>child</i> ) ≤ <i>grid.crowd</i> ( <i>primary</i> ))	Make the least crowded of the two
31: <i>primary</i> := <i>child</i>	solutions the new primary solution.

---

PAES archival truncation is also a relatively simple process. When a membership threshold level is exceeded, the maintenance procedure requires the removal of a solution residing the most densely occupied objective-space cell, thus ensuring a more evenly distributed set of solutions against which comparisons can be made. Clearly, the approach is subject to the same concerns as raised in Section 6.2.1.3 and Section 8.1.1.2 (namely, inaccuracies in crowding estimation and sensitivity to grid resolution). Subsequent results will verify whether such concerns are merited.

### 9.2.1.2 INTEGRATING THE MAK\_TREE INTO PAES

Integration of the specialist unbounded archive into PAES is trivial: the only requisite change is the replacement of the truncated archive with the basic Mak\_Tree. Note that it may be tempting to assume that some form of wholesale change is necessary with respect to the grid as well, but the structure can remain precisely as it is defined by Knowles and Corne [79] — its use and purpose are unaffected by the cardinality of the archival set.

### 9.2.1.3 EMPIRICAL ANALYSIS METHODOLOGY

The extended PAES algorithm (henceforth referred to as Mak\_PAES, for convenience) is compared with the basic PAES approach (with a truncated archive of fifty members) via the methodology described in Section 9.1.2. Both algorithms use

random initial solutions, asexual reproduction with a mutation rate of  $1/m$  and an adaptive grid comprised of 10,000 distinct cells. For aesthetic reasons, all related graphs and tables are provided at the completion of this subsection (see pages 250–260).

#### 9.2.1.4 EMPIRICAL ANALYSIS

Given the simplicity of the extension, the results are impressive. With respect to the progressive (averaged) hypervolume metric (Figure 100 and Figure 101), the unbounded PAES algorithm is preferable during the latter portions of the run on every tested function except *AP-4*. The end-of-run box-plots (Figure 102) support this claim, with the median hypervolume performance being *always* preferable to the basic PAES technique, while also featuring superior (lower-lying) inter-quartile ranges in all tested functions bar *AP-4*. With respect to statistical significance (Table 26), the end-of-run Mak\_PAES hypervolume results are never inferior, and they are preferable in the *AP-1*, *AP-2*, *AP-3* and *AP-5* functions in particular<sup>70</sup>.

Similarly encouraging results are reported in the epsilon-based analyses. In particular, the MAK\_PAES algorithm is never *significantly* inferior and is *significantly* better on the *AP-2* function when considering end-of-run performance (Figure 105 and Table 27). Additionally, the progressive (Figure 103 and Figure 104) and box-plot (Figure 105) results again indicate a general performance improvement during the later evaluations for the Mak\_PAES alternative, though *AP-15* show a bias towards the original PAES technique. Note that these results lead to an interesting outcome: Mak\_PAES is superior on *AP-15* under the hypervolume metric, but inferior when an epsilon indicator is applied; similarly, the basic PAES technique is both inferior (epsilon) and superior (hypervolume) when addressing the *AP-4* function. This disagreement between performance indicators illustrates the importance of multi-metric testing and suggests an inconclusive result on the corresponding functions<sup>71</sup>.

---

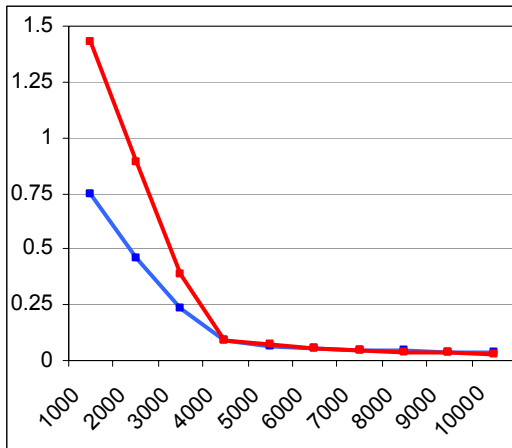
<sup>70</sup> For brevity, when statistical tests return a *significant* difference in the distribution of indicator scores between a pair of algorithms and the box-plots illustrate superior performance in one of the two systems, the better performing algorithm is referred to as *significantly* better than its opponent for the corresponding test function under the given indicator.

<sup>71</sup> If function performance was *significantly* better according to one metric and *significantly* worse according to another, the results *would* be conclusive — the produced fronts, in this case, are incomparable [218].

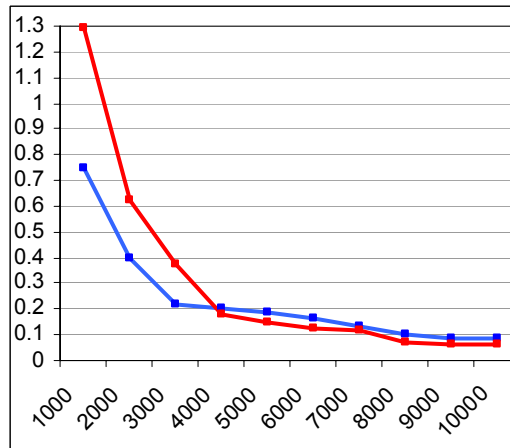


Considering the visual analyses, the Mak\_PAES attainment surfaces (Figure 108 and Figure 109) are clearly preferable with respect to both accuracy and distribution on *AP-4*, *AP-15*, *AP-16*, *AP-17* and *AP-21*. Regarding frequency matrices (Figure 106 and Figure 107), the unbounded PAES technique achieves impressive fronts on a more consistent basis for the *AP-2*, *AP-4*, *AP-5*, *AP-15*, *AP-16*, *AP-17* and *AP-21* test functions in particular, with statistically *significant* differences seen in *AP-4*, *AP-15* and *AP-17*. There are no test functions in which the PAES algorithm produces *significantly* better frequency performances.

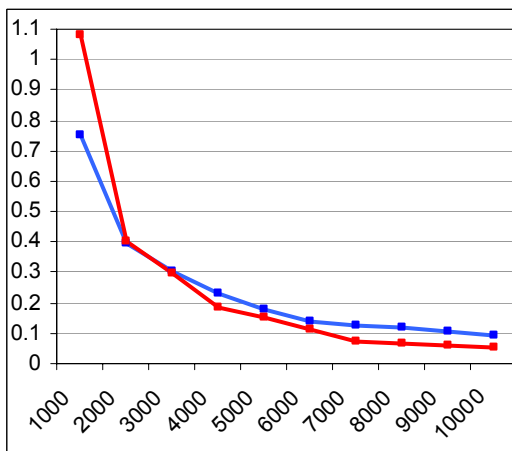
Thus, under at least one indicator, the end-of-run performance of the Mak\_PAES algorithm is *significantly* better than the basic PAES approach on *AP-1*, *AP-2*, *AP-3*, *AP-4*, *AP-5*, *AP-15* and *AP-17* — that is, on seven of the nine selected test functions. Across all statistical analyses, there are no occasions where a *significant* difference infers superior end-of-run performance for the original, truncated, technique. The reason for such a marked difference in performance is related to a lack of fidelity between the goals of the PAES algorithm and empirical reality. At its core, PAES is a hill-climber where the gradient is (hierarchically) defined firstly by a pair-wise dominance comparison, secondly by membership in the prevailing front and finally by a crowding measure. A failure to correctly express any of these factors and the intended gradient by which the search is governed is irrevocably changed. While the pair-wise dominance comparison is correct, the representation of the prevailing front and occupied-space is prone (and, in the case of this study at least, subject) to errors caused by the truncation process. Since the archive is open to degradation (as discussed in Section 6.2.1.1), non-elite solutions can become the primary (parent) solution in the evolutionary process. Similarly, inaccuracies in truncated crowding (see section 6.2.1.3) mean that there is no guarantee that the selected solution resides in a valuable region of objective-space. By using a complete set of elite solutions, the Mak\_PAES algorithm ensures that *only* members of the prevailing front can become the primary solution, while the unbounded grid ensures that the crowding scores offer an accurate reflection of the space occupied by the front. The results illustrate the primacy of such notions to the PAES algorithm and the potential for unbounded archiving to improve the results (with no marked increase in complexity) of this base-line algorithm.



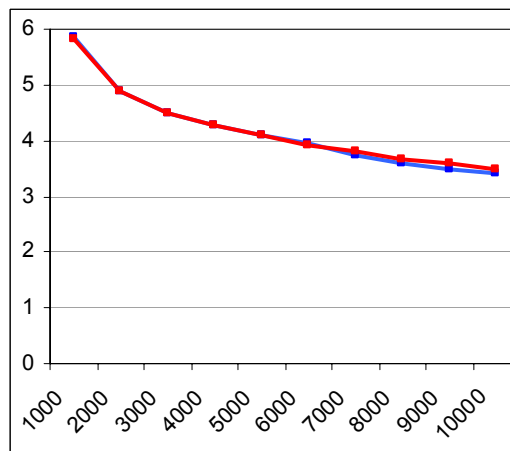
(a) AP-1



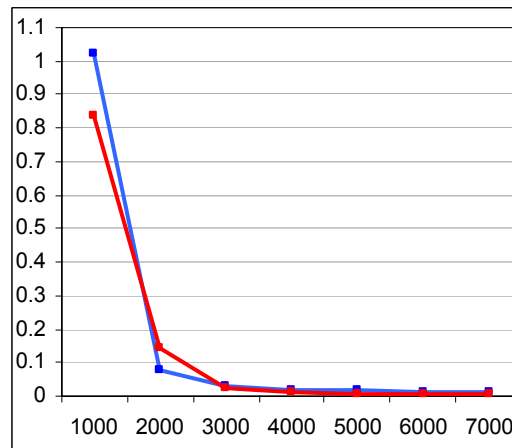
(b) AP-2



(c) AP-3



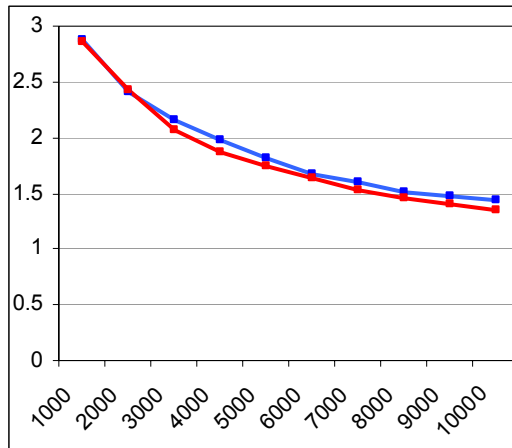
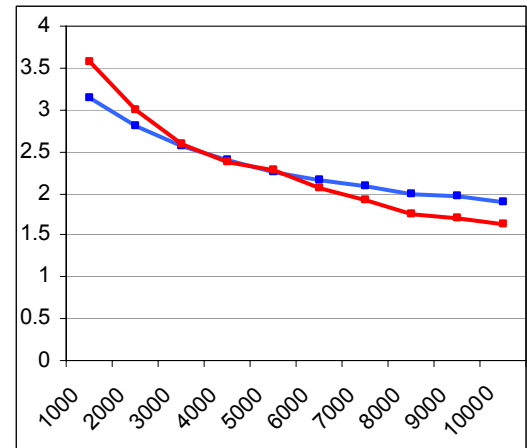
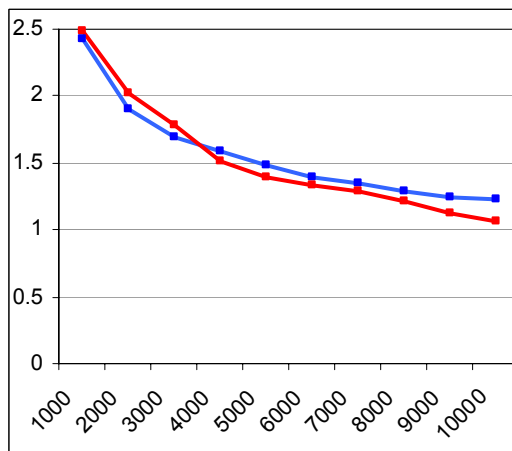
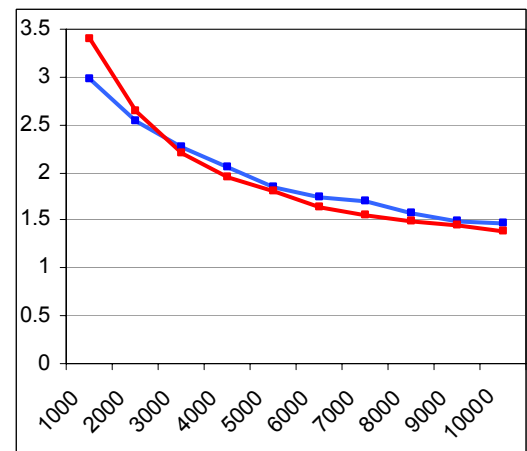
(d) AP-4



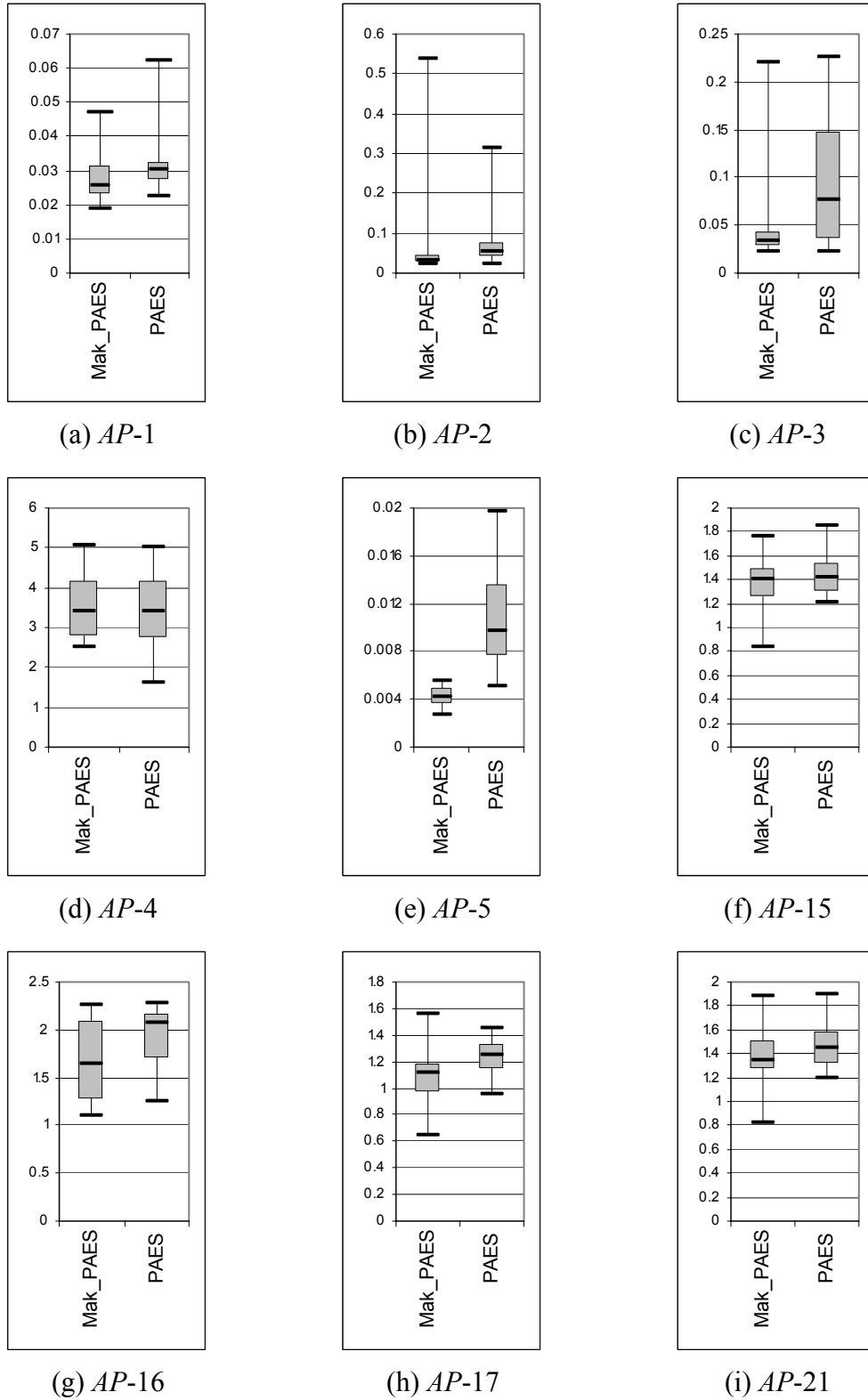
(e) AP-5

**Figure 100 — Progressive Hypervolume Averages**

PAES: Blue; Mak\_PAES: Red. *y-axis* is the average hypervolume performance; *x-axis* represents the number of evaluations executed by each optimiser.

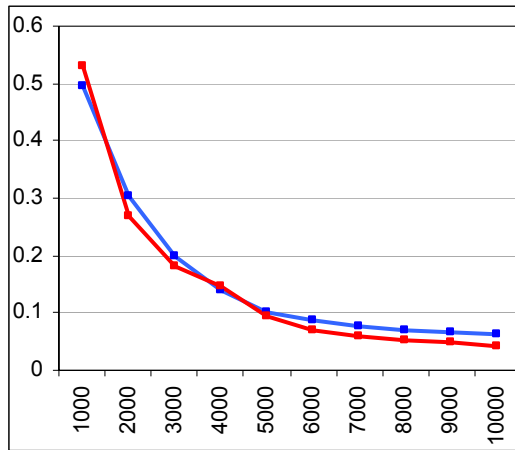
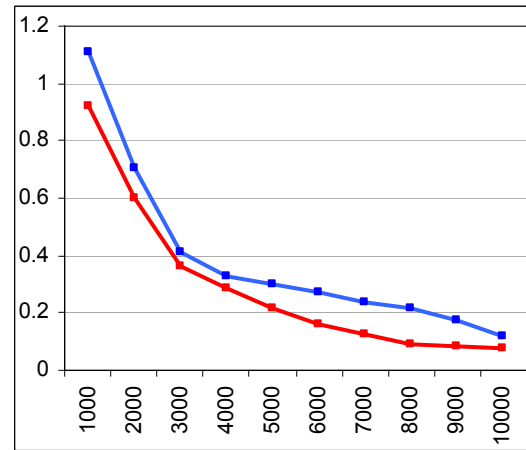
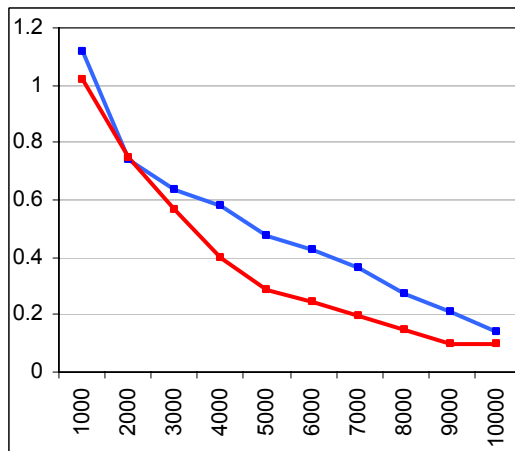
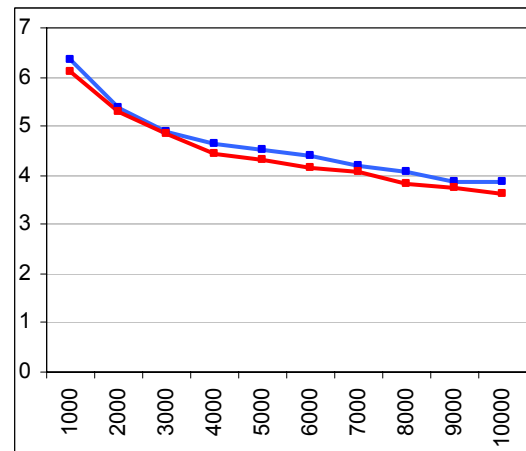
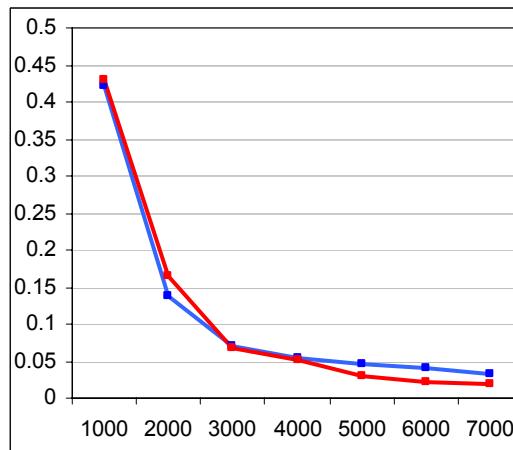
(a) *AP-15*(b) *AP-16*(c) *AP-17*(d) *AP-21***Figure 101 — Progressive Hypervolume Averages**

PAES: Blue; Mak\_PAES: Red. *y-axis* is the average hypervolume performance; *x-axis* represents the number of evaluations executed by each optimiser.

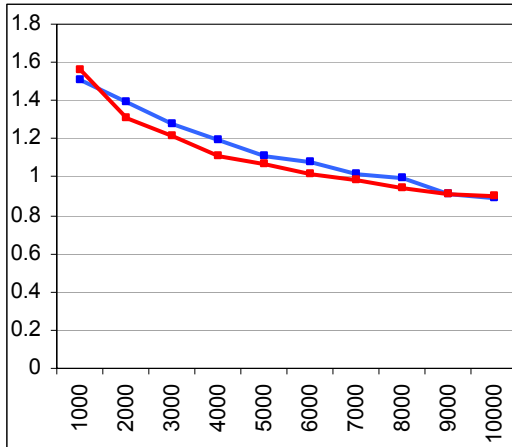


**Figure 102 — End-of-Run Hypervolume Box-Plots**

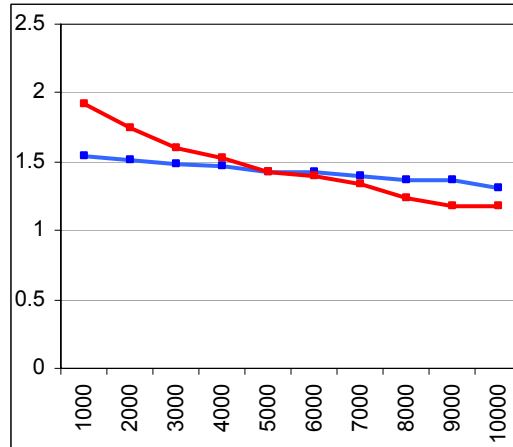
*y-axis* is the hypervolume performance at the end of the run (7,000 evaluations in AP-5, 10,000 evaluations in all remaining functions); *x-axis* indicates the selected optimiser.

(a) *AP-1*(b) *AP-2*(c) *AP-3*(d) *AP-4*(e) *AP-5*

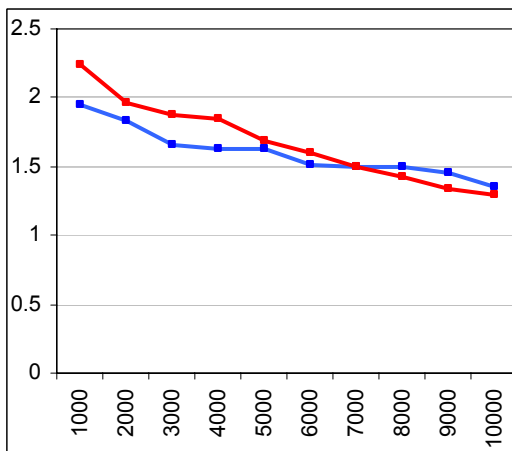
**Figure 103 — Progressive Epsilon Averages**  
 PAES: Blue; Mak\_PAES: Red. *y-axis* is the average epsilon performance; *x-axis* represents the number of evaluations executed by each optimiser.



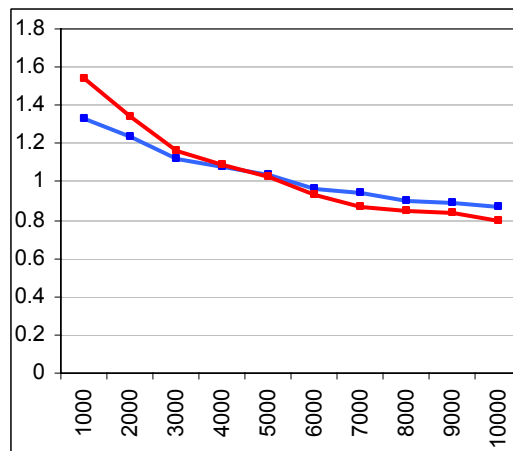
(a) AP-15



(b) AP-16



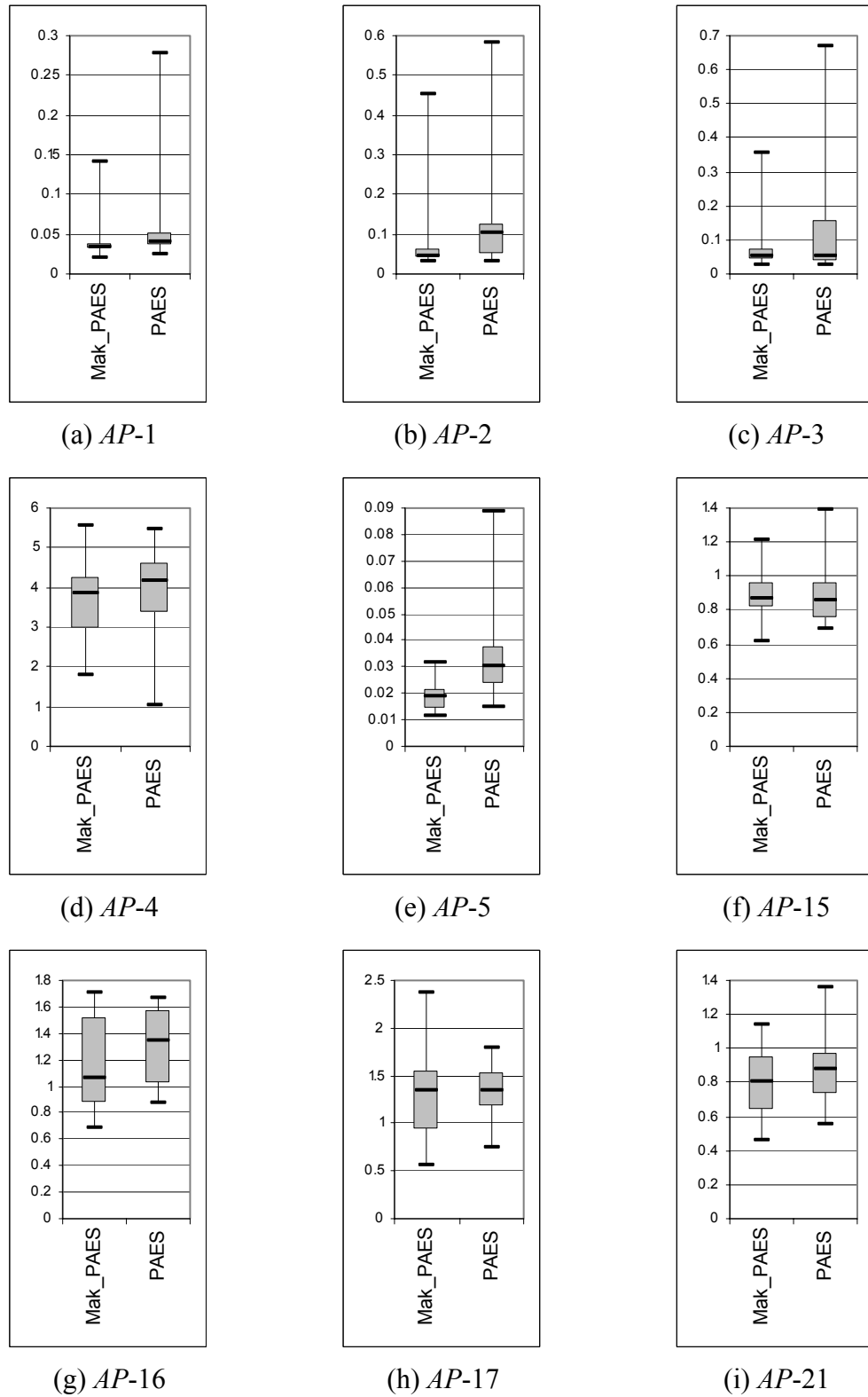
(c) AP-17



(d) AP-21

**Figure 104 — Progressive Epsilon Averages**

PAES: Blue; Mak\_PAES: Red. *y-axis* is the average epsilon performance; *x-axis* represents the number of evaluations executed by each optimiser.



**Figure 105 — End-of-Run Epsilon Box-Plots**

*y-axis* is the epsilon performance at the end of the run (7,000 evaluations in AP-5, 10,000 evaluations in all remaining functions); *x-axis* indicates the selected optimiser.

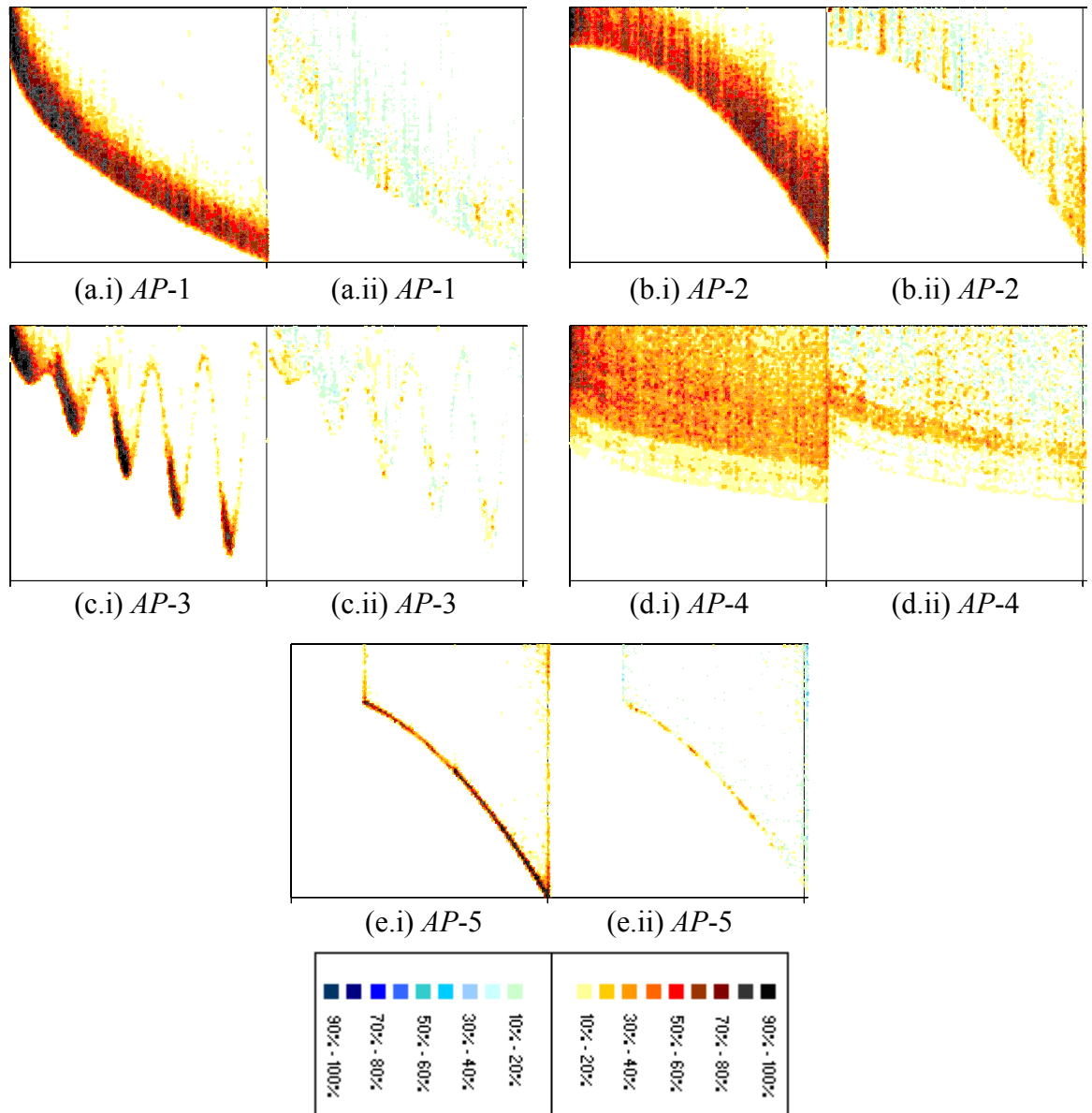
**Table 26 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Hypervolume Results**  
 Bold italics indicate significant differences at the 5% level.

	Hypervolume
<b><i>AP-1</i></b>	<b><i>4.283E-02</i></b>
<b><i>AP-2</i></b>	<b><i>1.591E-05</i></b>
<b><i>AP-3</i></b>	<b><i>9.485E-04</i></b>
<b><i>AP-4</i></b>	8.666E-01
<b><i>AP-5</i></b>	<b><i>2.156E-05</i></b>
<b><i>AP-15</i></b>	2.216E-01
<b><i>AP-16</i></b>	1.531E-01
<b><i>AP-17</i></b>	3.009E-01
<b><i>AP-21</i></b>	3.359E-01

**Table 27 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Epsilon Results**  
 Bold italics indicate significant differences at the 5% level.

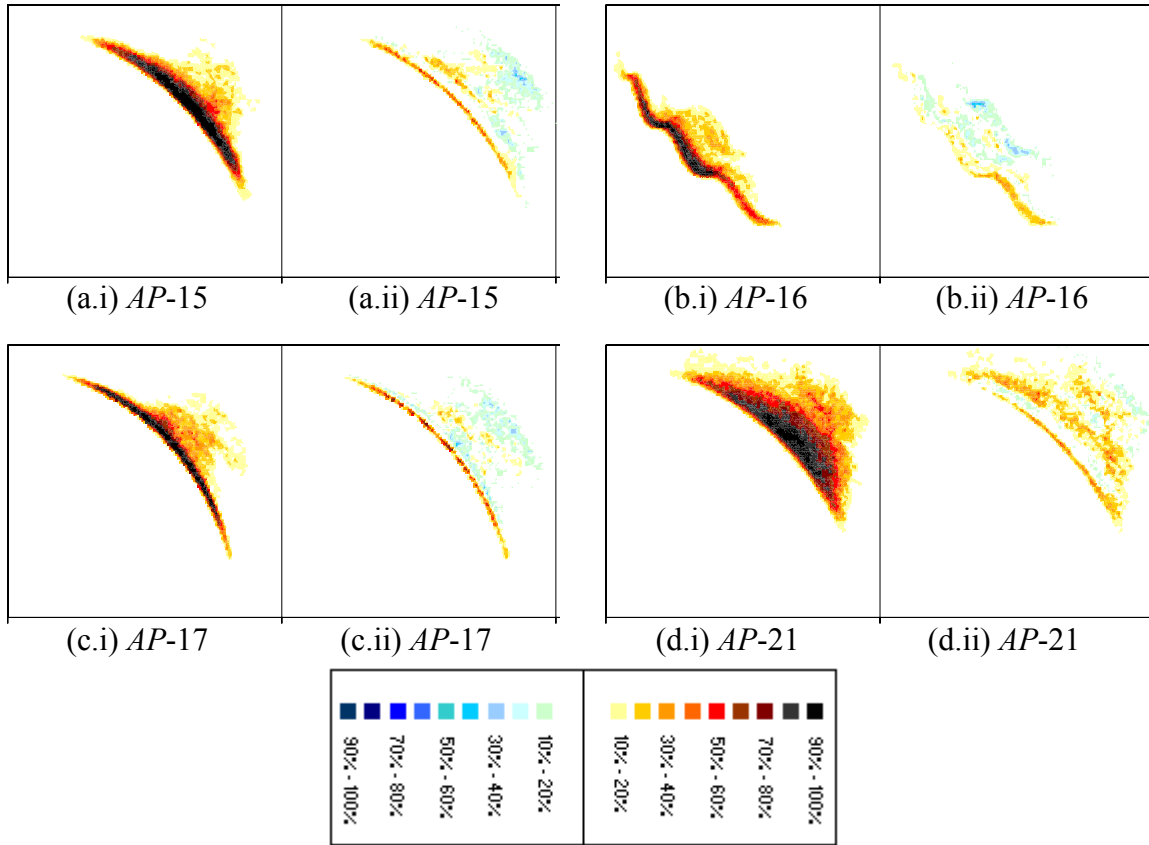
	Epsilon
<b><i>AP-1</i></b>	8.527E-02
<b><i>AP-2</i></b>	<b><i>2.122E-03</i></b>
<b><i>AP-3</i></b>	7.469E-01
<b><i>AP-4</i></b>	2.613E-01
<b><i>AP-5</i></b>	1.085E-01
<b><i>AP-15</i></b>	8.352E-01
<b><i>AP-16</i></b>	3.782E-01
<b><i>AP-17</i></b>	5.501E-01
<b><i>AP-21</i></b>	3.864E-01





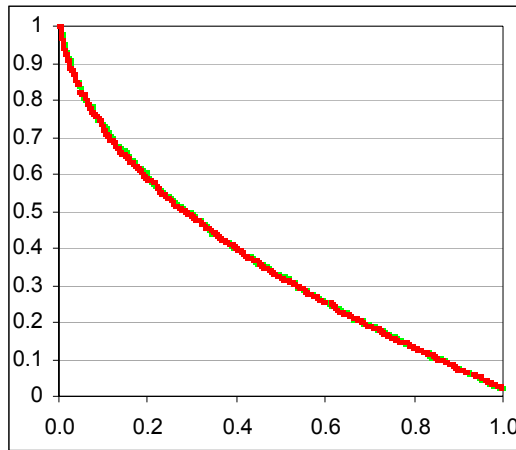
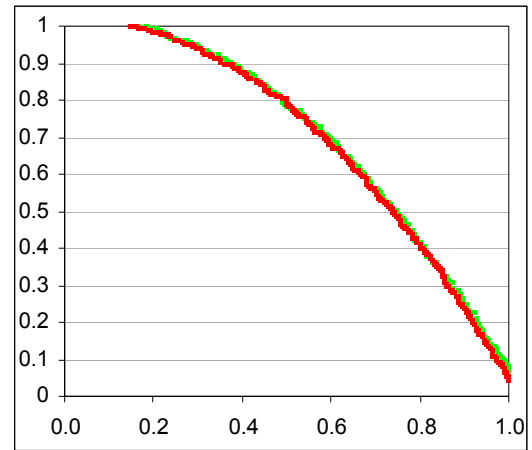
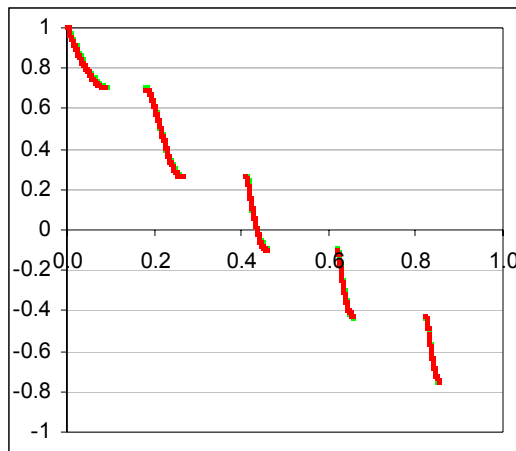
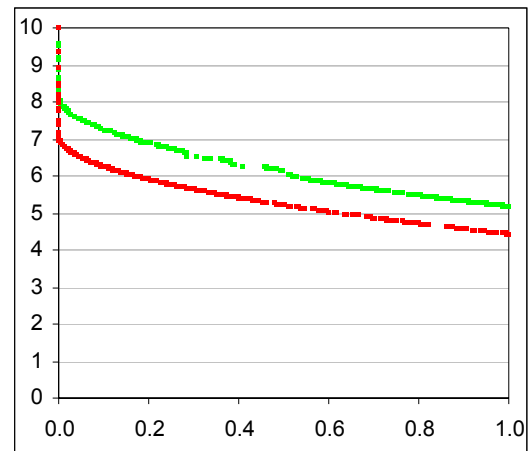
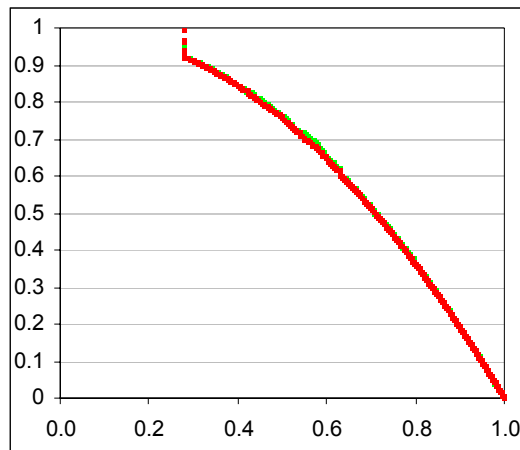
**Figure 106 — End-of-Run Frequency Matrices**

(i) Frequency matrix for Mak\_PAES. (ii) The red palette represents where Mak\_PAES more frequently attains a given region; the blue palette indicates where PAES attains an area more consistently.

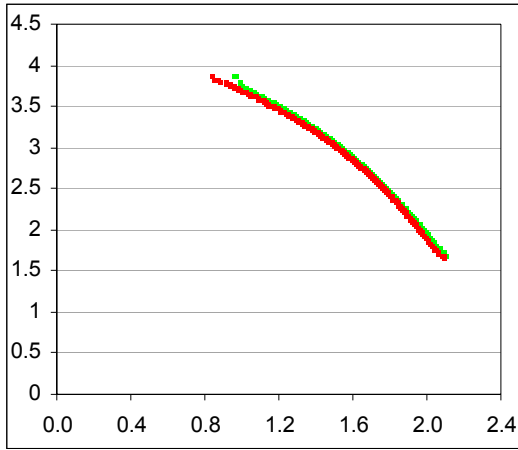


**Figure 107 — End-of-Run Frequency Matrices**

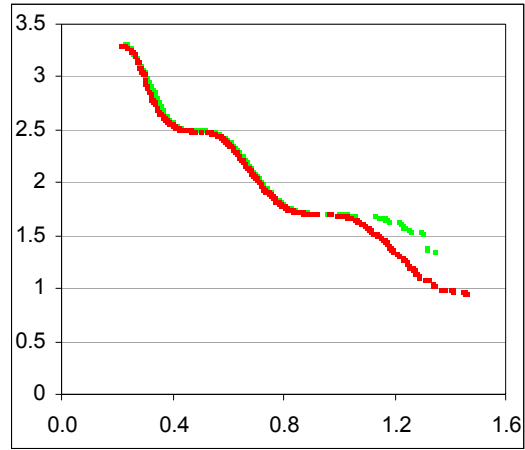
(i) Frequency matrix for Mak\_PAES. (ii) The red palette represents where Mak\_PAES more frequently attains a given region; the blue palette indicates where PAES attains an area more consistently.

(a) *AP-1*(b) *AP-2*(c) *AP-3*(d) *AP-4*(e) *AP-5*

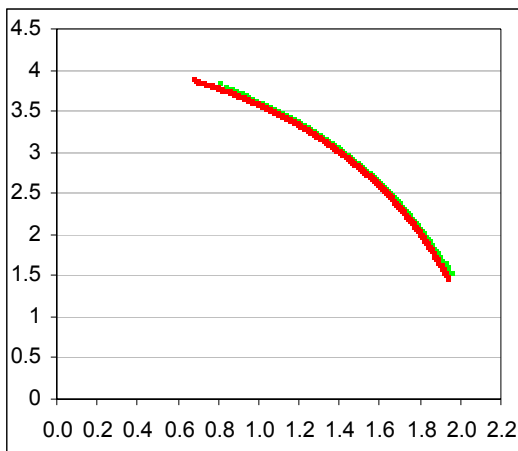
**Figure 108 — 50% Attainment Surfaces**  
 PAES: Green, Mak\_PAES: Red. *y-axis* is objective two; *x-axis* is objective one.



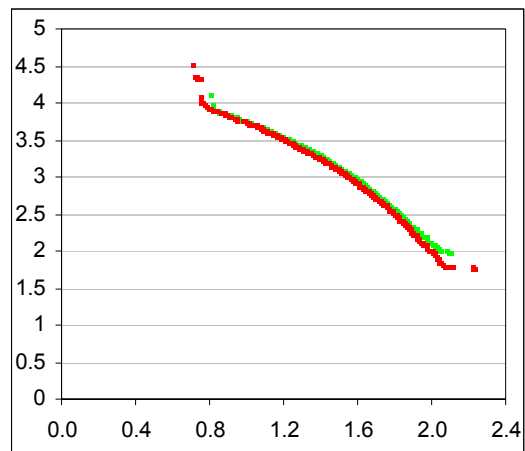
(a) *AP-15*



(b) *AP-16*



(c) *AP-17*



(d) *AP-21*

**Figure 109 — 50% Attainment Surfaces**

PAES: Green, Mak\_PAES: Red. *y-axis* is objective two; *x-axis* is objective one.

### 9.2.2 THE PARETO ENVELOPE SELECTION ALGORITHM

Built around the principles of the grid-based crowding methods explored in PAES (see Section 9.2.1), the Pareto Envelope Selection Algorithm (PESA) [136] moves away from the simple hill-climbing strategy employed by that approach and towards a population-based genetic algorithm. Importantly, where PAES used the archive purely as a reference set, the PESA algorithm explores the use of the (supposedly) elite store as a mating pool. As such, the integration of the Mak\_Tree into the PESA algorithm tests the suitability of specialist unbounded archiving when the prevailing front is an active part of the evolutionary process.

#### 9.2.2.1 DESCRIBING PESA

As with PAES, the central construct of the PESA technique (see Algorithm 18) is an adaptive grid that is used to approximate the cell-based crowding of members of the truncated archive. In particular, the selection of breeding agents is derived from a binary tournament of randomly selected archival members, where the winner of each match is derived from its cell-based crowding score. The resultant offspring of the selected members are tested for eligibility in the archival store and included when appropriate, with archival truncation performed as in PAES. The algorithm as a whole is little more than a repetition of this selection/update cycle.

**Algorithm 18 — The Pareto Envelope Selection Algorithm**

**Inputs:**

$b$	The maximum number of breeders required for each iteration.
1: $Archive := \emptyset; Children := \text{generateRandomPop}()$	Initialise the starting populations.
2: while (terminationConditionMet() $\neq$ true)	
3:   updateArchive( $Children$ )	Insert elite children into the archive and truncate if necessary.
4: $Pool := \emptyset; Parents := \emptyset$	Clear the pool and parent sets.
5:   while ( $ Pool  < 2b$ )	Fill the breeding pool with random members of the archive.
6: $Pool := Pool \cup \{\text{selectRandom}(Archive)\}$	
7:   while ( $ Parents  < b$ )	Host a binary tournament to select the parents for the next generation.
8: $first := \text{selectRandom}(Pool)$	
9: $second := \text{selectRandom}(Pool)$	
10:    if ( $grid.crowd(first) \leq grid.crowd(second)$ )	
11: $Parents := Parents \cup \{first\}$	
12:    else	
13: $Parents := Parents \cup \{second\}$	
14: $Children := \text{reproduce}(Parents)$	Breed the parents to produce the next generation.

### 9.2.2.2 INTEGRATING THE MAK\_TREE INTO PESA

The incorporation of an unbounded archive into the PESA technique is somewhat less straight-forward than in PAES. In particular, note that selection in PESA is predicated on using random extraction from the archival set and simple binary tournaments to apply evolutionary pressure on search diversification. This is suitable in a heavily truncated environment as random extraction is still likely to sample a considerable portion, if not the entirety, of the approximated front. In a purely unbounded archive, the random selection of frontal members will bias *more* densely occupied objective-spaces (since there will be more members in these areas) and the simplicity of a binary tournament offers insufficient pressure to rectify the imbalance. As such, when using an unbounded archive it is necessary to replace random frontal extraction with an approach that provides either an even distribution of the objective-space or that otherwise removes the bias afforded to crowded members.

An interesting approach is to replace random extraction with neighbourhood-based crowding selection. In this case, the  $b$  least-crowded members are passed into the binary tournament, with match results dictated by the grid-based scores. By using both neighbourhood and cell crowding estimates, this PESA variation also has the added benefit of observing objective-space density from two distinct, though equally valid, perspectives. Moreover, since the approach requires no truncation of the true

---

#### Algorithm 19 — Evenly Sub-Dividing the Objective-Space

---

**Inputs:**

$s$	The number of solutions to be extracted.
$obj$	The current objective against which sub-division is occurring.
$leftmost$	The solution with the smallest value on the specified objective.
$rightmost$	The solution with the largest value on the specified objective.
1: $range := leftmost^{obj} - rightmost^{obj}$	The range is the extent of the front.
2: $Set := \{leftmost, rightmost\}$	Always include extreme members in set.
3: $increment := \frac{range}{s}$	Evenly sub-divide the space according to the extent of the front.
4: $val := leftmost^{obj}$	
5: while $( Set  < s)$	
6: $val := val + increment$	Locate the solution with the objective score nearest
7: $Set := Set \cup \{findNearest(val, obj)\}$	to the current value.

---

non-dominated set, the selected members are guaranteed to be constituents of the prevailing front — a claim that is impossible to make in the standard PESA algorithm.

Alternatively, in an approach that shadows the PQRS technique suggested for use with Dominated Trees [97] (Section 6.2.2.2), the extraction procedure could endeavour to evenly sample the elite set. In this case, the procedure alternates between focussing on the first and second objective for each generation and follows the procedure outlined in Algorithm 19. For a selection set of size  $s$ , the cost of the new extraction procedure is  $O(s \log n)$ , as a simple binary search is required to identify each solution. The benefit of the approach is that it is more closely matched to the basic PESA selection algorithm — taking an approximately even distribution of the current front and filtering this with cell-based crowding — but it does so with the assurances of archival-quality that an unbounded store provides.

### **9.2.2.3 EMPIRICAL ANALYSIS METHODOLOGY**

The truncated PESA technique is compared with the crowding-based (Mak\_PESA\_C) and even-distribution (Mak\_PESA\_E) specialist unbounded algorithms via the methodology described in Section 9.1.2. All approaches feature random initial populations, archives with 10,000 distinct cells (as per Section 9.2.1.3), a crossover probability of 90% (with one child produced per interaction), a mutation rate of  $1/m$  and binary tournaments with one hundred matches (the winners of which are used to produce fifty children). With respect to Mak\_PESA\_C, to maximise efficiency, simple cuboid-based crowding measures (see Section 8.1.1.4) are employed via the extended Mak\_Tree<sup>72</sup>. The original PESA technique is bound by an archival threshold of fifty members. As in Section 9.2.1.4, all relevant result graphs and tables are presented at the completion of this sub-section (see pages 266–277).

### **9.2.2.4 EMPIRICAL ANALYSIS**

The results are impressive. Based on the end-of-run significance test results reported in Table 28 and Table 29 and the box-plots provided in Figure 112 and Figure 115,

---

<sup>72</sup> Since this is essentially a first-parse crowding estimation (the cell-based binary tournaments represent the second parse), the added accuracy of a  $\kappa$ -nearest neighbours approach is unlikely to yield sufficient gains to merit the corresponding performance trade-off.

the Mak\_PESA\_C algorithm *significantly* outperforms the original PESA technique on epsilon metrics and (independently) hypervolume metrics across *AP-1*, *AP-2*, *AP-5*, *AP-15*, *AP-16*, *AP-17* and *AP-21* (seven of the nine tested functions). Additionally, the progressive hypervolume (Figure 110 and Figure 111) and epsilon graphs (Figure 113 and Figure 114) show that the average performance of the Mak\_PESA\_C algorithm is preferable throughout the majority of the run on every tested function.

The Mak\_PESA\_E algorithm reports similarly impressive results against the numerical metrics — with *significant* performance improvements seen under the hypervolume indicator (Figure 112 and Table 28) on *AP-2*, *AP-3*, *AP-5*, *AP-15*, *AP-16*, *AP-17* and *AP-21* (seven of the nine examined functions), and *significant* epsilon gains (Figure 115 and Table 29) made in all bar the *AP-4* function. As with Mak\_PESA\_C, the Mak\_PESA\_E system achieves consistently better average performance across progressive indicators (Figure 113 and Figure 114) when compared to the basic PESA technique.

Looking at the relative performance of the two unbounded approaches, the crowding technique displays a *significant* end-of-run performance edge on *AP-1*, *AP-17* and *AP-21* under the hypervolume metric (Figure 112 and Table 28), and on *AP-17* under the epsilon metric (Figure 115 and Table 29). The evenly distributed approach, in contrast, only *significantly* outperforms the unbounded cuboid approach on *AP-3* (with respect to both hypervolume and epsilon measures). The relative inferiority of the crowding approach on the *AP-3* function is likely related to its use of cuboid density estimation — the disconnected nature of the *AP-3* front will lead to a heavy biasing of members that lie at the extremes of each isolated region (see Section 8.1.1.4). Clearly, such emphasis of extremal solutions comes at the expense of exploring truly uncrowded areas of objective-space and will affect the capacity of the Mak\_PESA\_C algorithm to effectively fill each disconnected region.

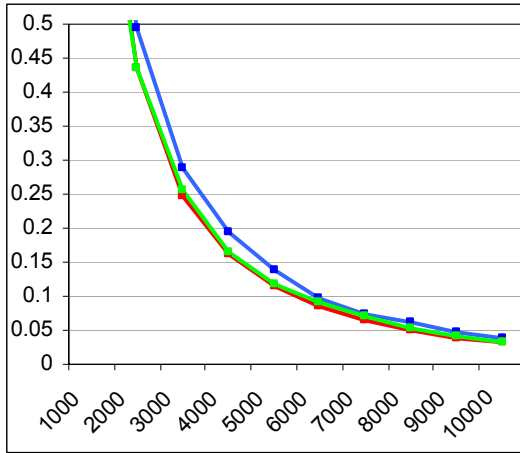
Given the statistical (end-of-run) superiority or equivalence of Mak\_PESA\_C against Mak\_PESA\_E on all functions other than *AP-3*, the pair-wise visual analysis of the unbounded and bounded techniques will focus on Mak\_PESA\_C and the basic PESA approach. Frequency matrices (Figure 116 and Figure 117) illustrate that the unbounded technique is more consistent in locating members of leading fronts and



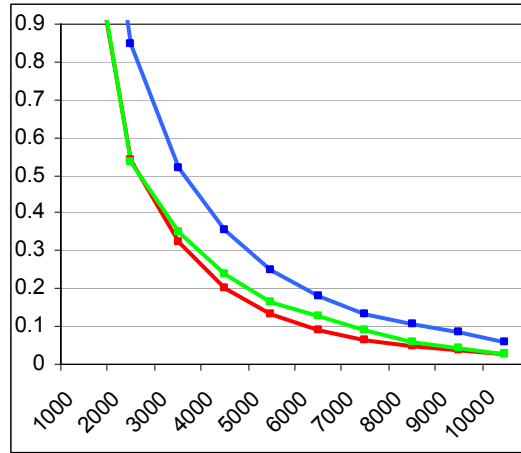
that the unbounded approach better distributes the search (as best illustrated in *AP-15*, *AP-16*, *AP-17* and *AP-21*, where the search is shown to be far more expansive along each leading front). The statistical significance tests support these claims and report mathematically *significant* differences in *AP-2*, *AP-3*, *AP-5*, *AP-15*, *AP-16* and *AP-17*. The attainment surfaces (Figure 118 and Figure 119) also demonstrate a clear advantage (particularly with respect to the spread of solutions) for the *Mak\_PESA\_C* algorithm on *AP-2*, *AP-4*, *AP-5*, *AP-15*, *AP-16*, *AP-17* and *AP-21*.

Thus, when compared with the original PESA technique across all metrics (hypervolume, epsilon and frequency), the unbounded *Mak\_PESA\_C* technique has *significantly* better end-of-run results on *AP-2*, *AP-5*, *AP-15*, *AP-16* and *AP-17* (five of the nine tested functions) and is *significantly* superior on at least one metric for all functions other than *AP-4*. The relative failure of the original PESA algorithm is again related to the necessary duality of the truncated archive — members must be well-spread along the front to reduce the likelihood of accepting weak solutions, but they must also reflect areas of over-crowding in order to correctly guide the search. Clearly, the two goals are at odds — by creating an evenly distributed archival set, it is inherently difficult to identify regions of space which require exploration, while an ill-distributed set which better approximates objective-space crowding is more vulnerable to the acceptance of weak solutions. The inability of the PESA algorithm to explore as much of the optimal objective-space as the unbounded approach in the later *AP* functions (*AP-15* onwards) illustrates this deficiency — the truncated archive is insufficient for the correct identification of valuable regions.

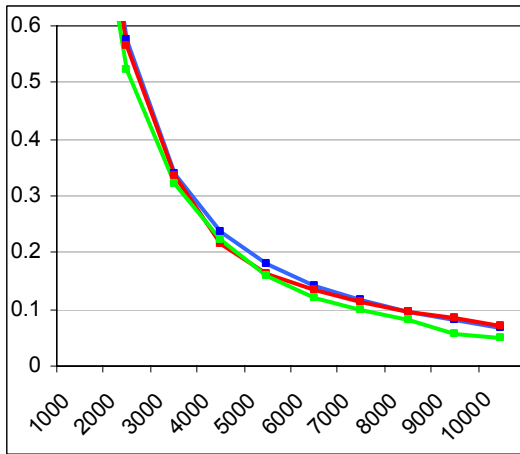
With respect to the two archival extraction procedures proposed, the differences are not particularly great, though they tend to favour the crowding technique when exploring *significant* differences (likely because this places further impetus on frontal exploration). The minimal difference between the procedures suggests that the majority of the search power comes from the binary tournament and so the extraction procedure is of only minor import (though this work maintains that extraction should never favour crowded-regions — the procedure should actively bias against this, or otherwise seek an even spread of solutions). The investigation of alternative extraction procedures and the effects on search performance should further elucidate the role of pre-processing in binary tournaments and rests as an interesting avenue of future work.



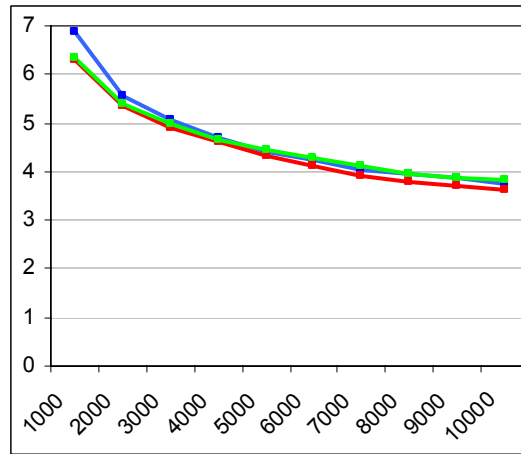
(a) AP-1



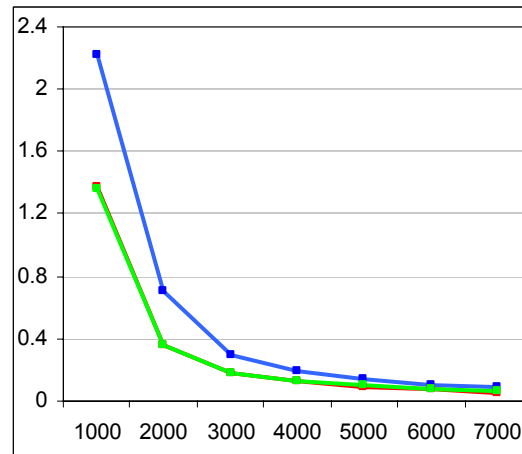
(b) AP-2



(c) AP-3



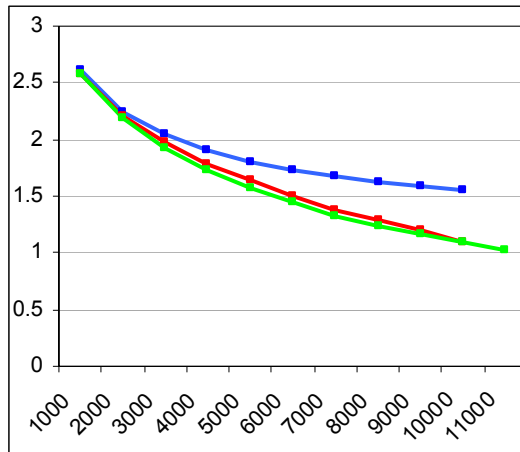
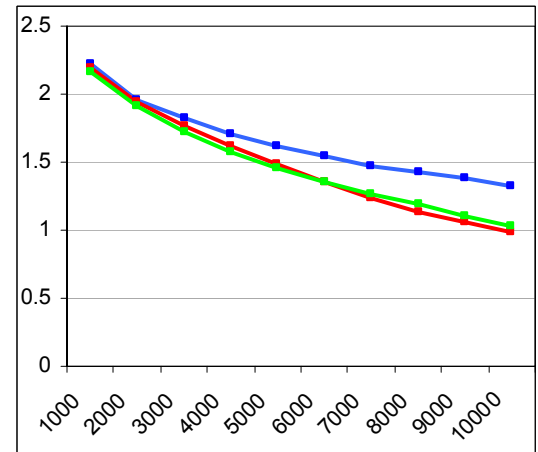
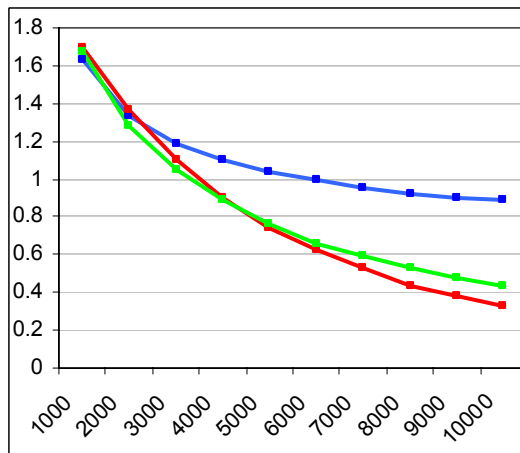
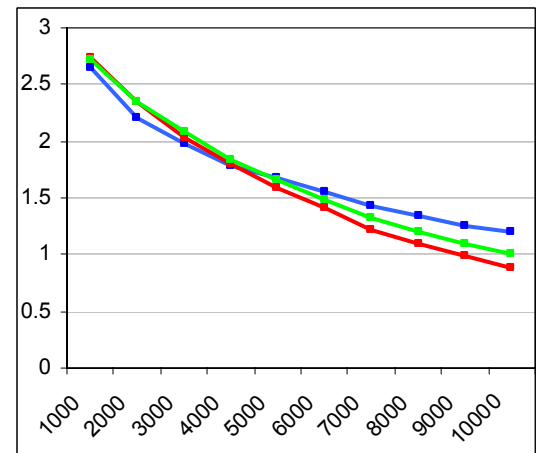
(d) AP-4



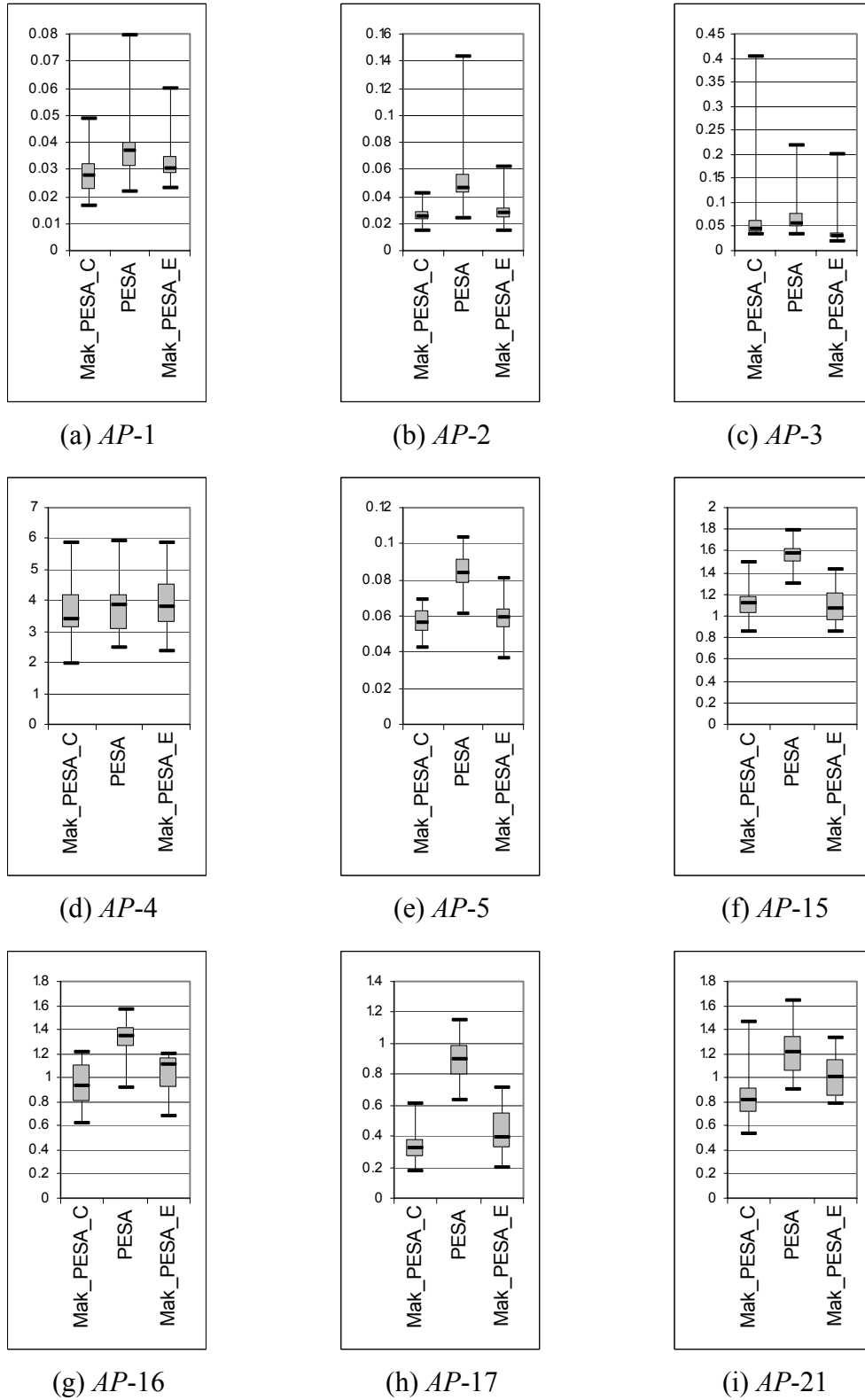
(e) AP-5

**Figure 110 — Progressive Hypervolume Averages**

PESA: Blue; Mak\_PESA\_C: Red; Mak\_PESA\_E: Green. *y-axis* is the average hypervolume performance; *x-axis* represents the number of evaluations executed by each optimiser.

(a) *AP-15*(b) *AP-16*(c) *AP-17*(d) *AP-21***Figure 111 — Progressive Hypervolume Averages**

PESA: Blue; Mak\_PESA\_C: Red; Mak\_PESA\_E: Green. *y-axis* is the average hypervolume performance; *x-axis* represents the number of evaluations executed by each optimiser.



**Figure 112 — End-of-Run Hypervolume Box-Plots**

*y-axis* is the hypervolume performance at the end of the run (7,000 evaluations in *AP-5*, 10,000 evaluations in all remaining functions); *x-axis* indicates the selected optimiser.

**Table 28 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Hypervolume Results**  
 Bold italics indicate significant differences at the 5% level.

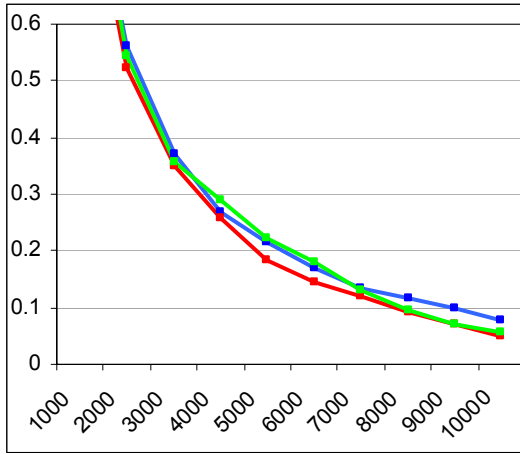
	PESA	Mak_ PESA_C	Mak_ PESA_E	(a) AP-1
PESA	-	<b><i>4.09E-05</i></b>	1.11E-01	
Mak_ PESA_C	<b><i>9.41E-11</i></b>	-	<b><i>1.07E-02</i></b>	
Mak_ PESA_E	<b><i>2.80E-08</i></b>	3.11E-01	-	
(b) AP-2				

	PESA	Mak_ PESA_C	Mak_ PESA_E	(c) AP-3
PESA	-	1.09E-01	<b><i>3.07E-07</i></b>	
Mak_ PESA_C	6.10E-01	-	<b><i>3.29E-04</i></b>	
Mak_ PESA_E	8.54E-01	4.88E-01	-	
(d) AP-4				

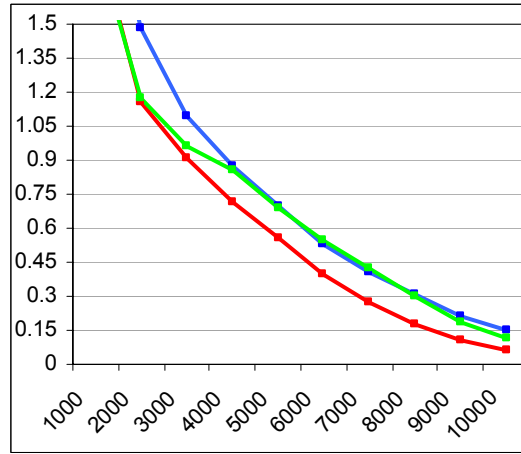
	PESA	Mak_ PESA_C	Mak_ PESA_E	(e) AP-5
PESA	-	<b><i>1.12E-16</i></b>	<b><i>1.20E-14</i></b>	
Mak_ PESA_C		-	4.97E-01	

	PESA	Mak_ PESA_C	Mak_ PESA_E	(f) AP-15
PESA	-	<b><i>2.30E-06</i></b>	<b><i>3.25E-08</i></b>	
Mak_ PESA_C	<b><i>1.03E-09</i></b>	-	3.90E-01	
Mak_ PESA_E	<b><i>9.97E-06</i></b>	7.02E-02	-	
(g) AP-16				

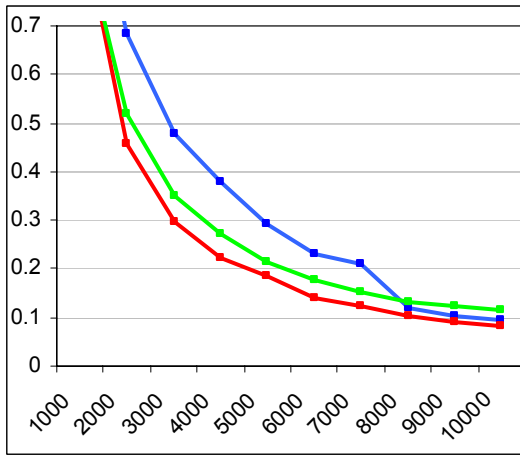
	PESA	Mak_ PESA_C	Mak_ PESA_E	(h) AP-17
PESA	-	<b><i>1.51E-11</i></b>	<b><i>1.24E-06</i></b>	
Mak_ PESA_C	<b><i>2.29E-06</i></b>	-	<b><i>3.87E-02</i></b>	
Mak_ PESA_E	<b><i>8.10E-03</i></b>	<b><i>3.18E-02</i></b>	-	
(i) AP-21				



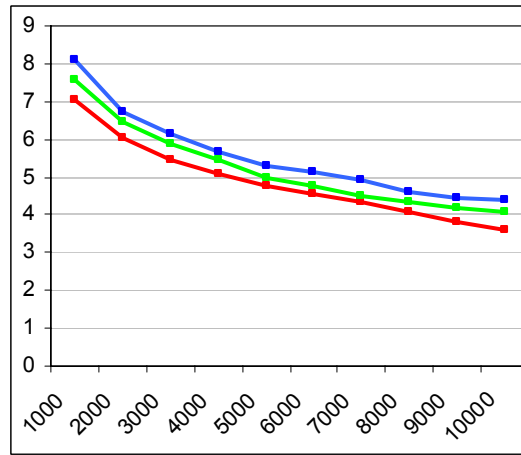
(a) AP-1



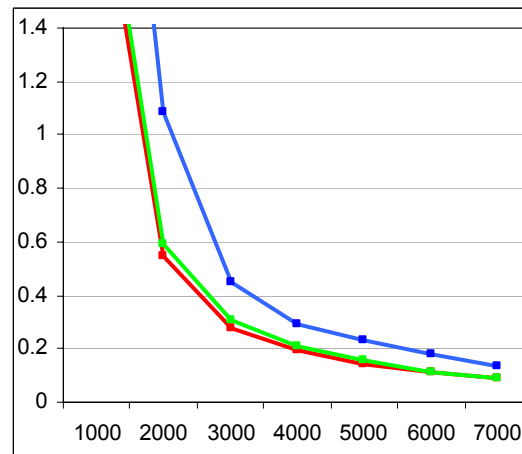
(b) AP-2



(c) AP-3



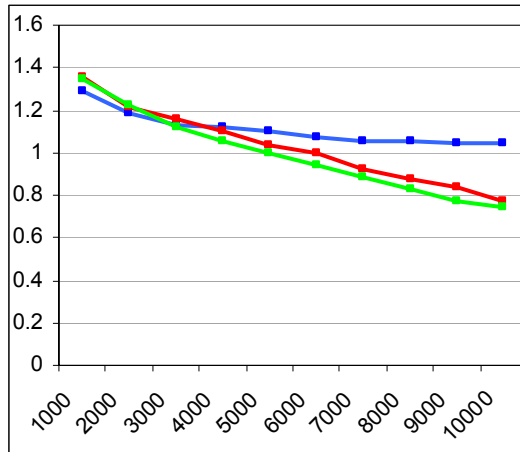
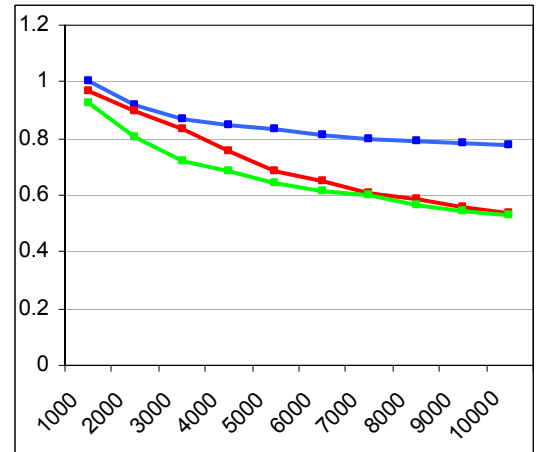
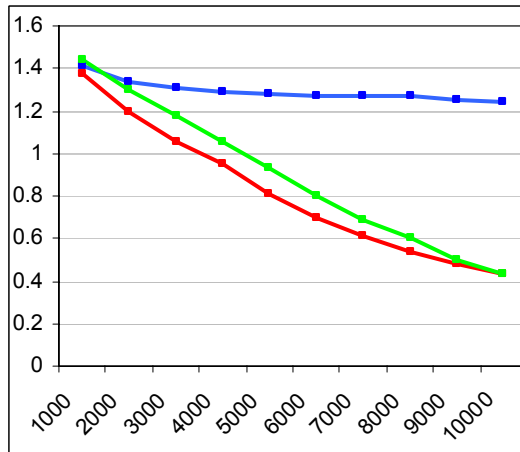
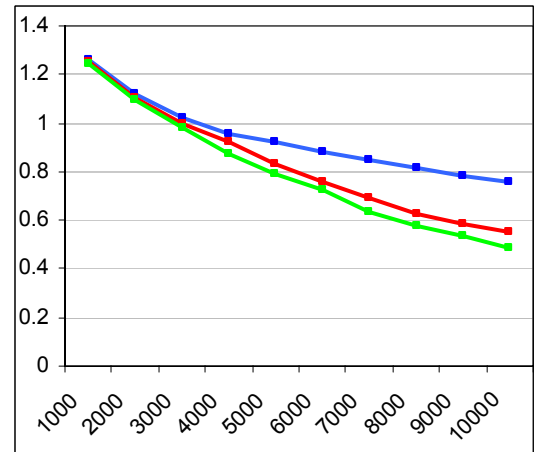
(d) AP-4



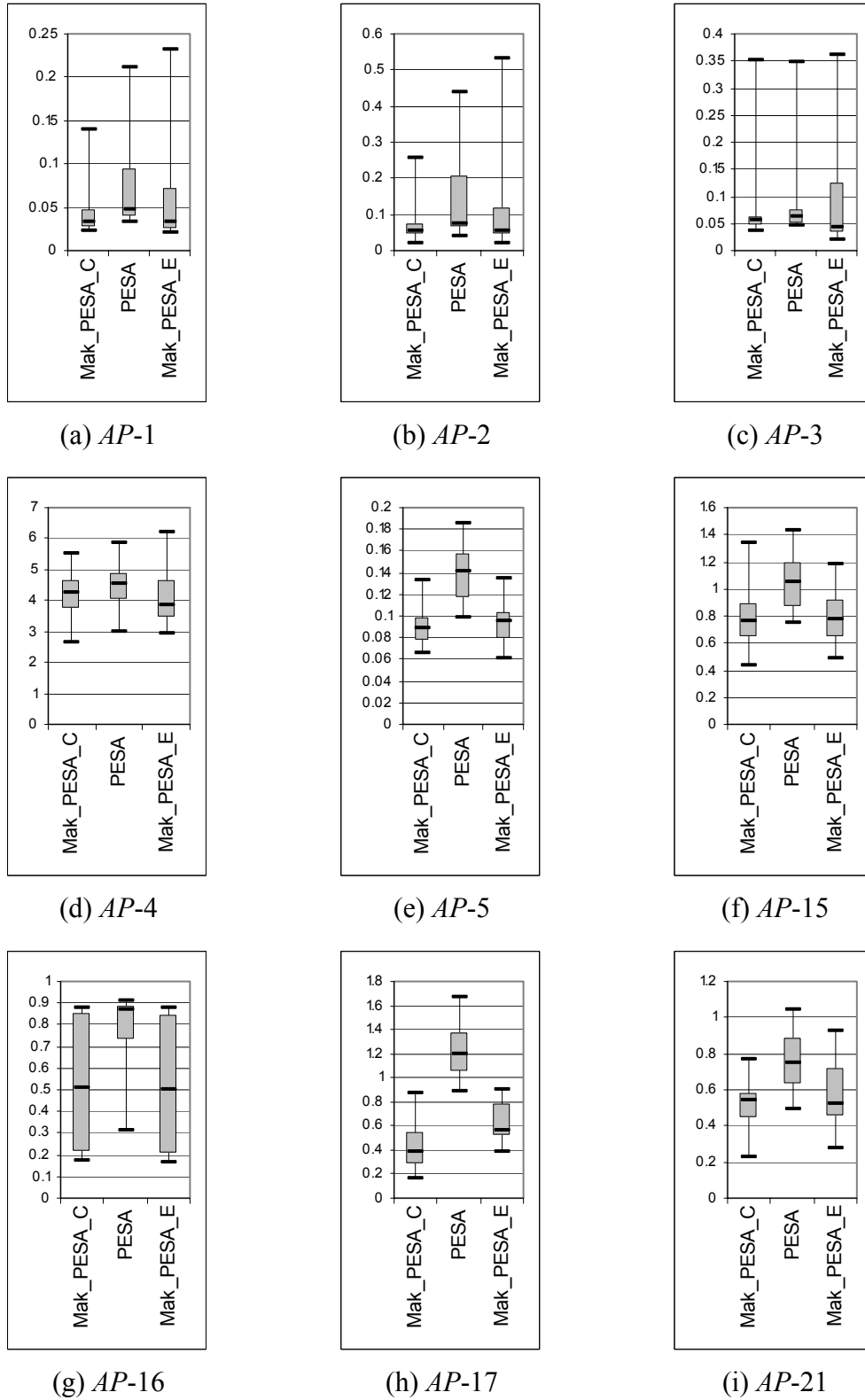
(e) AP-5

**Figure 113 — Progressive Epsilon Averages**

PESA: Blue; Mak\_PESA\_C: Red; Mak\_PESA\_E: Green. *y-axis* is the average hypervolume performance; *x-axis* represents the number of evaluations executed by each optimiser.

(a) *AP-15*(b) *AP-16*(c) *AP-17*(d) *AP-21***Figure 114 — Progressive Epsilon Averages**

PAES: Blue; Mak\_PAES: Red. *y-axis* is the average epsilon performance; *x-axis* represents the number of evaluations executed by each optimiser.



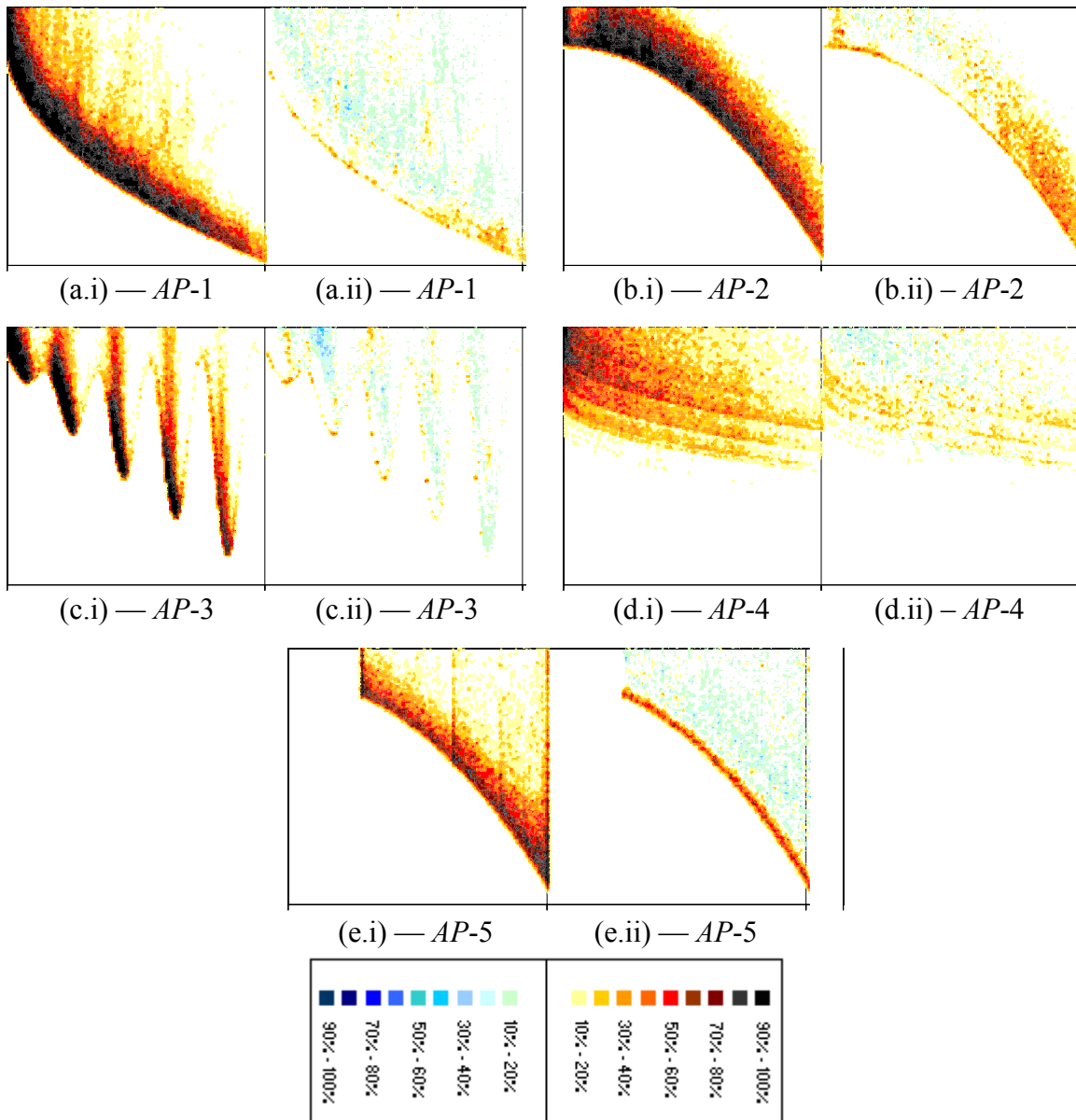
**Figure 115 — End-of-Run Epsilon Box-Plots**

*y-axis* is the epsilon performance at the end of the run (7,000 evaluations in *AP-5*, 10,000 evaluations in all remaining functions); *x-axis* indicates the selected optimiser.



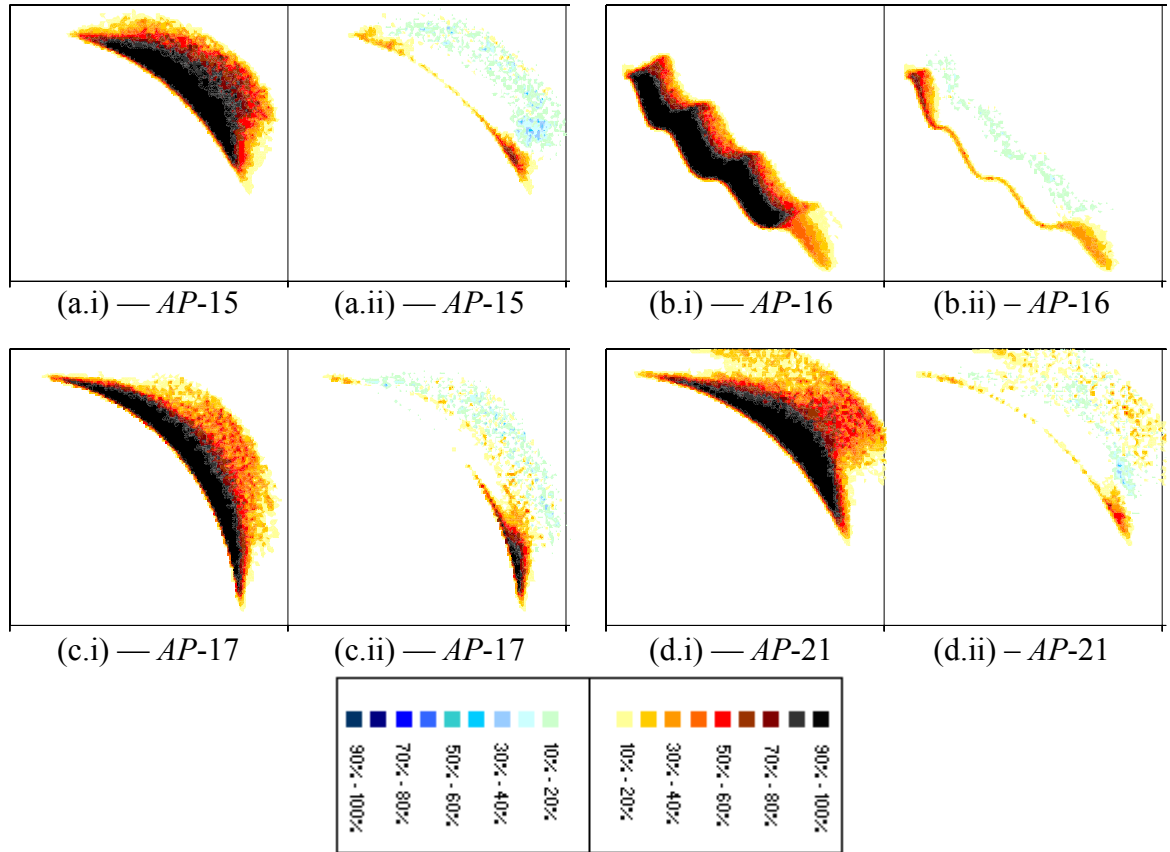
Table 29 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Epsilon Results  
 Bold italics indicate significant differences at the 5% level.

				(a) AP-1
	PESA	Mak_ PESA_C	Mak_ PESA_E	
PESA	-	<b><i>4.60E-04</i></b>	<b><i>6.43E-05</i></b>	
Mak_ PESA_C	<b><i>4.60E-04</i></b>	-	6.09E-01	
Mak_ PESA_E	<b><i>6.43E-05</i></b>	6.09E-01	-	
(b) AP-2				
				(c) AP-3
	PESA	Mak_ PESA_C	Mak_ PESA_E	
PESA	-	2.19E-01	<b><i>9.13E-04</i></b>	
Mak_ PESA_C	3.38E-01	-	<b><i>3.50E-02</i></b>	
Mak_ PESA_E	1.29E-01	5.73E-01	-	
(d) AP-4				
				(e) AP-5
	PESA	Mak_ PESA_C	Mak_ PESA_E	
PESA	-	<b><i>5.71E-15</i></b>	<b><i>5.72E-13</i></b>	
Mak_ PESA_C		-	4.85E-01	
				(f) AP-15
	PESA	Mak_ PESA_C	Mak_ PESA_E	
PESA	-	<b><i>1.70E-04</i></b>	<b><i>6.73E-04</i></b>	
Mak_ PESA_C	<b><i>3.21E-03</i></b>	-	7.10E-01	
Mak_ PESA_E	<b><i>1.21E-03</i></b>	7.66E-01	-	
(g) AP-16				
				(h) AP-17
	PESA	Mak_ PESA_C	Mak_ PESA_E	
PESA	-	<b><i>2.32E-14</i></b>	<b><i>1.10E-05</i></b>	
Mak_ PESA_C	<b><i>5.27E-05</i></b>	-	<b><i>4.18E-04</i></b>	
Mak_ PESA_E	<b><i>7.46E-04</i></b>	4.87E-01	-	
(i) AP-21				



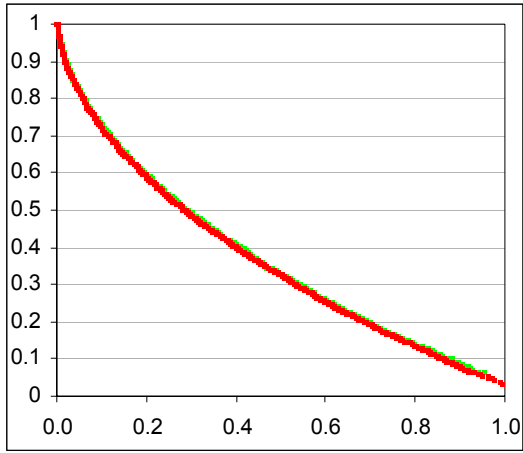
**Figure 116 — End-of-Run Frequency Matrices**

(i) Frequency matrix for Mak\_PESA\_C. (ii) The red palette represents where Mak\_PESA\_C more frequently attains a given region; the blue palette indicates where PESA attains an area more consistently.

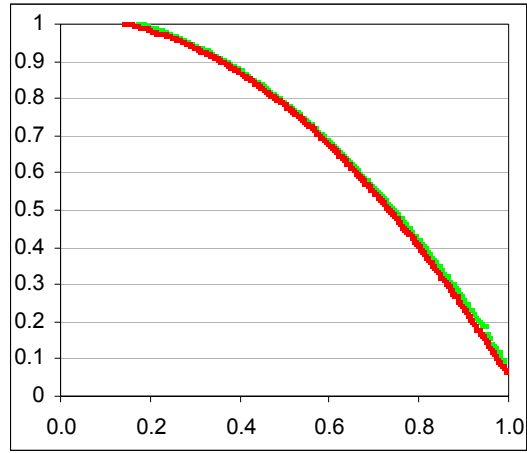


**Figure 117 — End-of-Run Frequency Matrices**

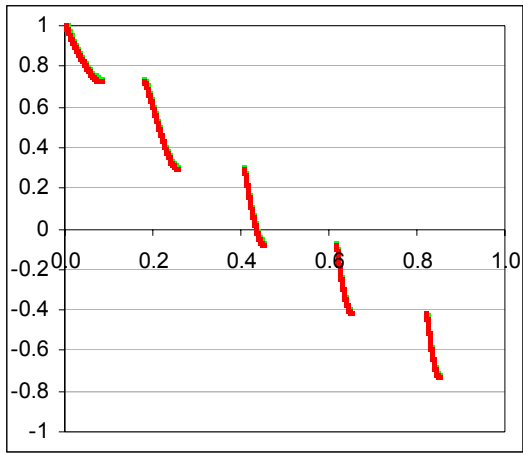
- (i) Frequency matrix for Mak\_PESA\_C. (ii) The red palette represents where Mak\_PESA\_C more frequently attains a given region; the blue palette indicates where PESA attains an area more consistently.



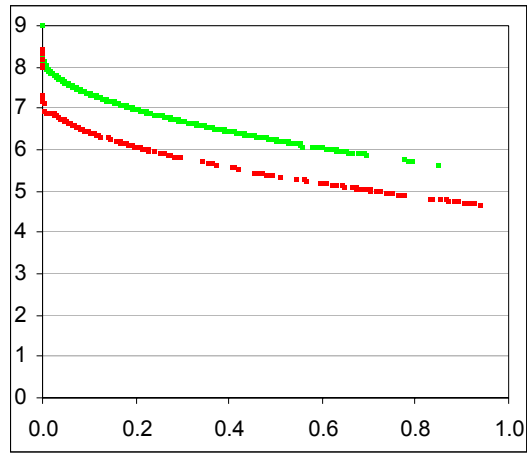
(a) *AP-1*



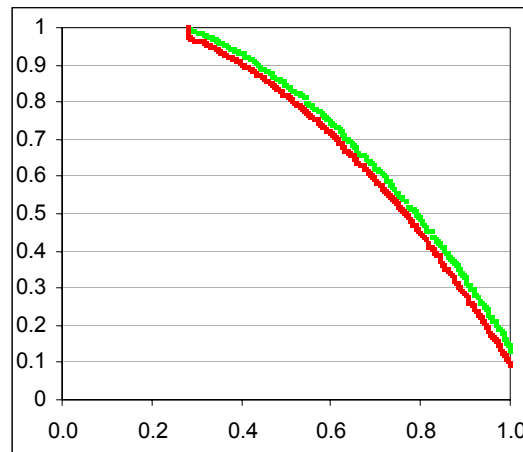
(b) *AP-2*



(c) *AP-3*



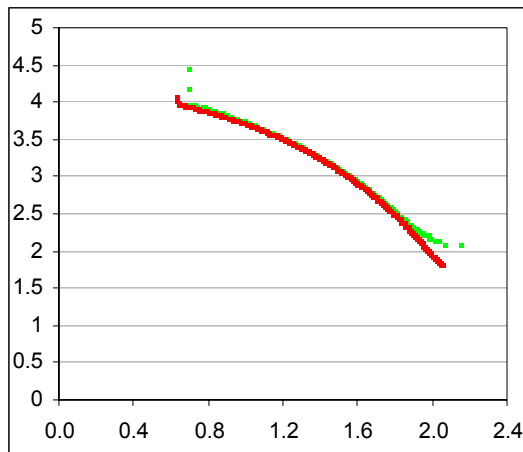
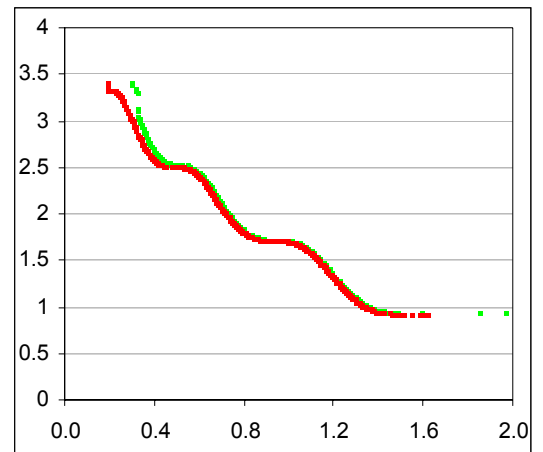
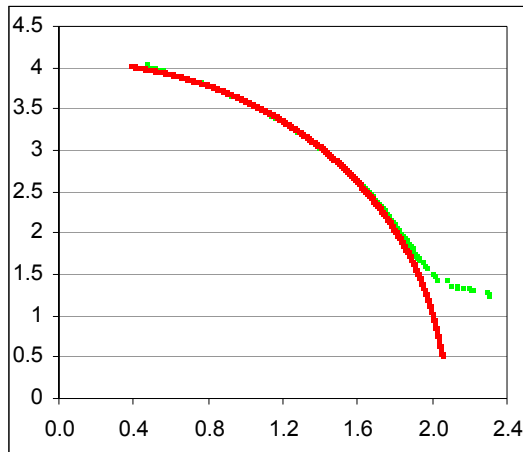
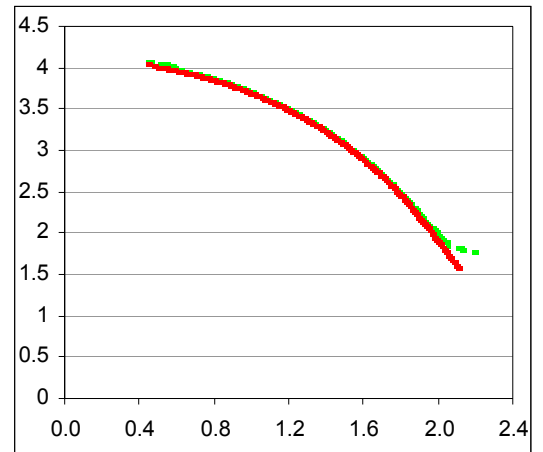
(d) *AP-4*



(e) *AP-5*

**Figure 118 — 50% Attainment Surfaces**

PESA: Green, Mak\_PESA\_C: Red.  $y$ -axis is objective two;  $x$ -axis is objective one.

(a) *AP-15*(b) *AP-16*(c) *AP-17*(d) *AP-21***Figure 119 — 50% Attainment Surfaces**

PESA: Green, Mak\_PESA\_C: Red. *y*-axis is objective two; *x*-axis is objective one.

### **9.2.3 THE ELITIST NON-DOMINATED SORTING GENETIC ALGORITHM (NSGA-II)**

Based simply on the number of applications (see, for instance, [113, 226-229]) and studies (including [43, 95, 151, 182, 230, 231]) devoted to the NSGA-II algorithm [82, 175], there can be little doubt that it stands as a corner-stone of contemporary multiobjective optimisation research. It solidified the importance of elitism (particularly in combination with Knowles and Corne [143] and Zitzler *et al.* [80, 81, 92]) and established a new, though simple, mechanism for parameter-less crowding estimation — namely, the cuboid function (see Section 8.1.1.4 for more details).

#### **9.2.3.1 DESCRIBING NSGA-II**

NSGA-II (see Algorithm 20) varies from its PESA contemporary in two key areas: it moves away from grid-based crowding estimation and towards a neighbourhood-based approach; and it features an archive that is not charged with storing exclusively non-dominated solutions. Indeed, the archival set contains as many fronts as are necessary to reach the threshold level, with excess members removed from the worst of the included fronts according to estimated crowding scores.

Still, despite the apparent theoretical dissimilarities, the core mechanics of the NSGA-II algorithm differ little to those seen in the likes of PESA. In both cases, the archival set is randomly sampled for entry into a binary tournament, with match outcomes dictated by some form of quality metric. In the case of NSGA-II, the quality indicator considers frontal membership (lines 20–23 in Algorithm 20) in addition to crowding (lines 24–27), though it could be argued that this change is typically important only early in the run. When the archival set is insufficiently large to store the entire prevailing front, the test for frontal membership is entirely meaningless: all archived solutions will reside in the *same* front and have the same frontal ranking. Thus, so long as the prevailing front contains more members than can be stored in the archive, the only practical differentiation between PESA and NSGA-II is the selected crowding measure — be it neighbourhood-based or cell-based.

#### **9.2.3.2 INTEGRATING THE MAK\_TREE INTO NSGA-II**

To further explore the capacity of the Mak\_Tree to improve the performance of evolutionary algorithms under differing roles, the unbounded archive is used here as a

purely referential set. That is to say, the unbounded store may be queried for information, but the archival extraction process operates entirely on a truncated NSGA-II archive. Thus, this section aims to explore whether access to more complete density estimations and dominance checks alone are enough to improve the performance of a powerful contemporary algorithm.

Specifically, the NSGA-II algorithm described in Section 9.2.3.1 is modified such that the leading (rank one) non-dominated front may consist only of members that reside in the unbounded archive (since any other solution is dominated with respect

#### Algorithm 20 — The NSGA-II Algorithm

##### Inputs:

$s$	The maximum archive size.
$b$	The number of solutions allowed in the breeding pool.
1: $Arch := \text{generateRandomPop}()$	Initialise the starting population.
2: $Parents := \emptyset$	
3: $\text{deriveFronts}(Arch)$	
4: while (terminationConditionMet() $\neq$ true)	
5:   while ( $ Parents  < b$ )	Perform binary tournament selection
6: $first := \text{selectRandom}(Arch)$	with members of the archive.
7: $second := \text{selectRandom}(Arch)$	
8:     if (front( $first$ ) < front( $second$ ))	The winner is the solution residing in a better
9: $Parents := Parents \cup \{first\}$	(leading) front
10:    else if (front( $first$ ) > front( $second$ ))	
11: $Parents := Parents \cup \{second\}$	
12:    else if (cubScore( $second$ ) $\leq$ cubScore( $first$ ))	or, in the case where both solutions reside in
13: $Parents := Parents \cup \{first\}$	the same front, the winner is the least
14:    else	crowded of the two solutions according
15: $Parents := Parents \cup \{second\}$	to their cuboid-derived neighbourhoods.
16: $Children := \text{reproduce}(Parents)$	Generate the child set from parents.
17: $Parents := \emptyset$	
18: $Arch := Arch \cup Children$	Add all children to the archive and derive
19: $Fronts := \text{deriveFronts}(Arch)$	the set of unique fronts in the archive.
20: $Arch := \emptyset$	Empty the archive.
21: $currentFront := 1$	
22: while ( $ Arch  < s$ )	Until the archive is filled,
23:   if ( $ Fronts_{currentFront} \cup Arch  \leq s$ )	
24: $Arch := Arch \cup \{a\} \forall a \in Fronts_{currentFront}$	add all members of the current front and
25: $currentFront := currentFront + 1$	progress to the next front,
26:   else	unless the archive will overflow, in which case,
27: $n := s -  Fronts_{currentFront} \cup Arch $	add only the $n$ least-crowded members of the
28: $Arch := Arch \cup \text{leastCrowded}(Fronts_{currentFront}, n)$	current front.

to the reference set) and that rank one members utilise unbounded cuboid crowding estimates when binary tournaments occur (using the efficient update process defined in Section 8.1.3). It is important to note that the unbounded archive is entirely passive; it is simply probed for information about the current NSGA-II population in an effort to better represent the current state of the prevailing front.

### **9.2.3.3 EMPIRICAL ANALYSIS METHODOLOGY**

The truncated NSGA-II algorithm featuring an unbounded reference set (Mak\_NSGA-II) is compared to the original NSGA-II algorithm according to the methodology outlined in Section 9.1.2 (provide relevant tables and graphs). Both systems are similarly defined. Namely, they feature: random initial populations, a crossover probability of 90% (with two children produced per interaction); a mutation rate of  $1/m$ ; and a truncated archive of fifty members from which fifty randomly selected pairs engage in matches to enter breeding pool. The *only* point of differentiation is the use of an unbounded extended Mak\_Tree as a reference set in the Mak\_NSGA-II algorithm. Again, for aesthetic reasons, all relevant graphs and tables are provided at the completion of this sub-section in pages 284–294.

### **9.2.3.4 EMPIRICAL ANALYSIS**

The performance of the Mak\_NSGA-II algorithm is strong despite using the unbounded archive acting as a purely passive reference set. In particular, the end-of-run hypervolume results (Figure 122) and statistical tests (Table 30) illustrate that the Mak\_NSGA-II technique *significantly* outperforms the basic NSGA-II approach on *AP-1*, *AP-3*, *AP-15*, *AP-16*, *AP-17* and *AP-21* (six of the nine tested functions), with the later (*AP-15*, *AP-16*, *AP-17* and *AP-21*) averaged progressive graphs (Figure 120 and Figure 121) displaying considerable improvement levels through the length of the run. The original technique is preferable in *AP-2*, *AP-4* and *AP-5* with respect to both progressive hypervolume performance (Figure 120 and Figure 121) and end-of-run box-plots (Figure 122), though the difference is never statistically *significant* (Table 30).

Performance measures under the epsilon metric are also positive. The Mak\_NSGA-II algorithm achieves *significantly* better end-of-run results (Figure 125 and Table



31) on *AP-3* and *AP-17*. In contrast, the original technique garners a *significant* end-of-run performance improvement only on *AP-5*. It is worth noting that the averages for NSGA-II appear noticeably better than the Mak\_NSGA-II alternative in *AP-16* (see Figure 124) — this is linked to the more erratic performance of Mak\_NSGA-II under the epsilon metric on this problem (see Figure 125), though is insufficient to generate a statistically *significant* difference.

The frequency matrices (Figure 126 and Figure 127) illustrate that the original algorithm is only markedly better on the *AP-4* function, and even then the result is statistically insignificant. Indeed, Mak\_NSGA-II is better able to find impressive fronts more consistently in *AP-1*, *AP-5*, *AP-15*, *AP-16*, *AP-17* and *AP-21* — with a *significant* difference in *AP-15*, *AP-16* and *AP-17*. Note that where NSGA-II is *significantly* inferior, it tends to place a greater (and apparently less successful) search emphasis around the extremes of the optimal front — this is likely due to a subtle deviation in the way cuboid crowding operates under an unbounded reference set. Where the original technique guarantees that the extremes of the locally stored optimal front are always granted maximal crowding scores to ensure victory in binary tournament matches, the same assertion can only be made under the Mak\_NSGA-II scheme if the extremes of the locally stored front are also the extremes of the unbounded, global, front. It is open to debate as to whether this approach is preferable in general as it may lead to shrinking local fronts (due to the potential loss of local extremes), though the impressive results here suggest that this is at most a secondary concern.

The median attainment surfaces (Figure 128 and Figure 129) support the assertions made about the frequency matrices: namely that NSGA-II offers a clear advantage only in *AP-4* and that *AP-15*, *AP-16*, *AP-17* and *AP-21* are better suited to the Mak\_NSGA-II technique. The remaining functions offer no clear visual indicator as to which algorithm is ideal.

Thus, the results as a whole suggest that the Mak\_NSGA-II algorithm is at least competitive with and generally better than the NSGA-II approach on the rich set of tested functions. Indeed, the Mak\_NSGA-II approach was *significantly* better according to at least one metric in six of the nine *AP* functions (*AP-1*, *AP-3*, *AP-15*, *AP-16*, *AP-17* and *AP-21*) and *significantly* better under two metrics on four

problems (*AP-3*, *AP-15*, *AP-16* and *AP-17*). The key to the improved performance of the Mak\_NSGA-II algorithm lies in weakening the duality of the truncated archive. The key drawback of the conventional truncated archive is that it is fundamentally torn between storing a solution set that will better guide the search towards poorly explored regions of objective-space and one that will offer a suitably distributed set to reduce the likelihood of frontal degradation. By using a reference set for non-dominance determination and crowding estimation, the truncated archival set can concentrate on storing solutions that are situated in valuable regions of space without the inaccurate crowding and dominance checks that plague a conventional truncated store. As discussed earlier, the only danger that exists under this new scheme is the potential for shrinking local fronts (if a number of solutions lie in a single region of useful objective-space, the local archive could well converge onto this region). This, however, opens up some interesting avenues of future work. For instance, maximal local front spread could be maintained by incorporating the two extreme Mak\_Tree solutions into the truncated archive (though this clearly breaches the passivity of the reference set), or local crowding scores could be used in the archival truncation phase to ensure an evenly distributed and well-spread local front (though this reduces the influence of global crowding measures in guiding the search).

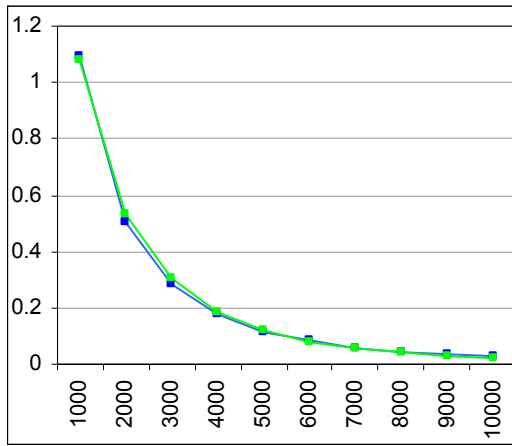
Across all tests, the only functions that caused continued problems for the unbounded reference approach were *AP-4* and *AP-5*, though *AP-5* illustrated the only *significant* difference (on the epsilon performance indicator). The *AP-4* result is particularly interesting — since, the function is multi-frontal and the frequency matrix (Figure 126) illustrates a tendency for the optimisers to at least partially converge on sub-optimal fronts, the reason for improved performance in NSGA-II may be related to noise. As the basic NSGA-II archive is subject to the acceptance of weak solutions (as for most truncated approaches), the prevailing local front may be a noisy one — consisting of both dominated and non-dominated solutions<sup>73</sup>. It is possible that these noisy solutions — which lie away from the false front — may provide the solution diversity necessary to escape continued premature convergence. Thus, surprisingly,

---

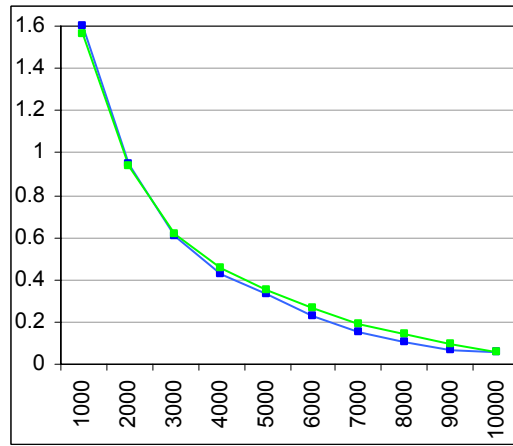
<sup>73</sup> Note that this does not refer to the noise caused by maintaining multiple fronts in the archival set (which is relevant only when the prevailing front is small in any case); it instead refers to the incorrect labelling of dominated solutions as non-dominated and the acceptance of weak members into the rank one local front.

the improved accuracy of the archival set in the Mak\_NSGA-II algorithm may be a factor in reducing performance against multi-frontal tasks.

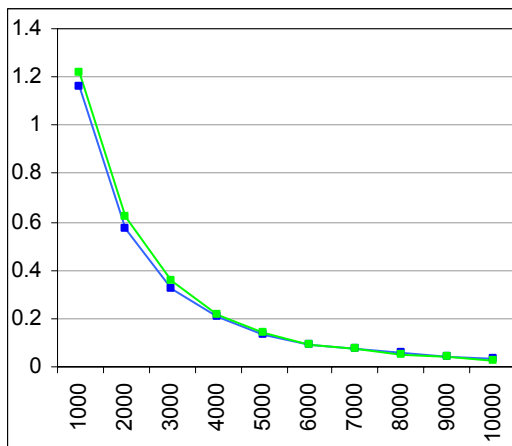
It is more difficult to extract why numerical indicators infer that performance is inferior on *AP-5* as visual metrics suggest that the Mak\_NSGA-II algorithm is at least competitive and, in the case of frequencies, marginally superior. From a theoretical standpoint, it is possible that the lack of noise in the quality of solutions stored in the leading front decreases the ability of the Mak\_NSGA-II optimiser to escape biased or deceptive regions of objective-space, but the empirical results do not particularly support this notion (the frequency graphs illustrate a consistently well-spread search). An alternative potential cause is a narrowing of the leading front around biased or isolated regions due to their temporary preferability with respect to global crowding measures (as predicted in Section 6.2.1.3), though again there is little empirical proof of such an occurrence.



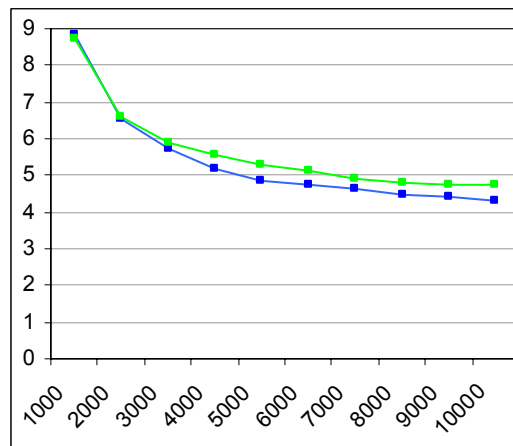
(a) AP-1



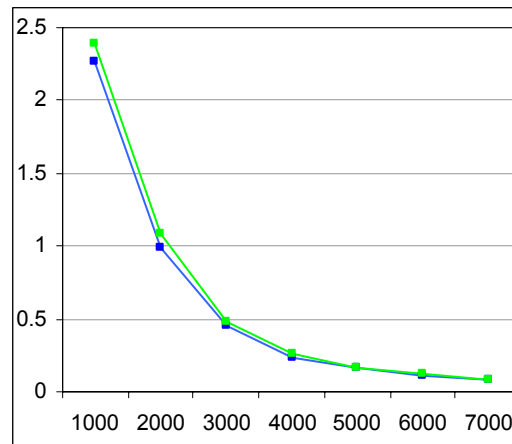
(b) AP-2



(c) AP-3



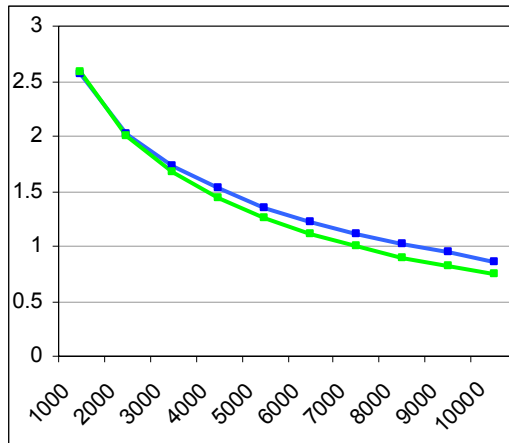
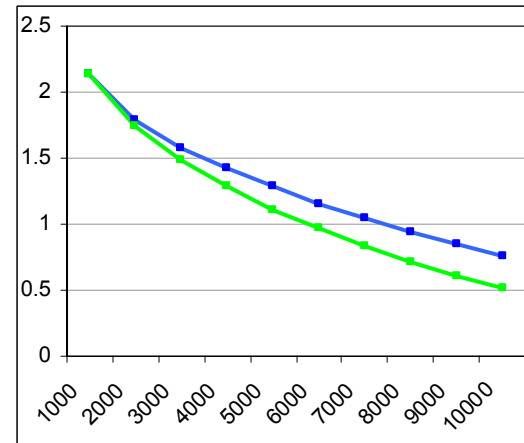
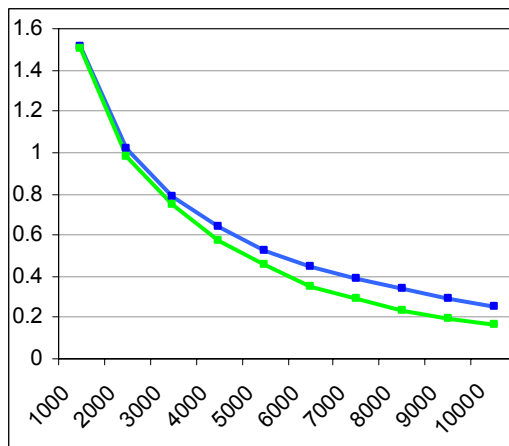
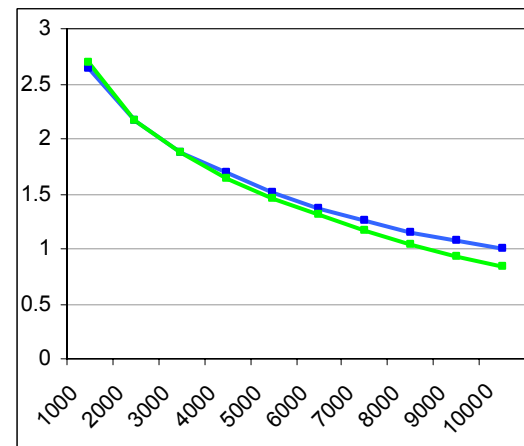
(d) AP-4



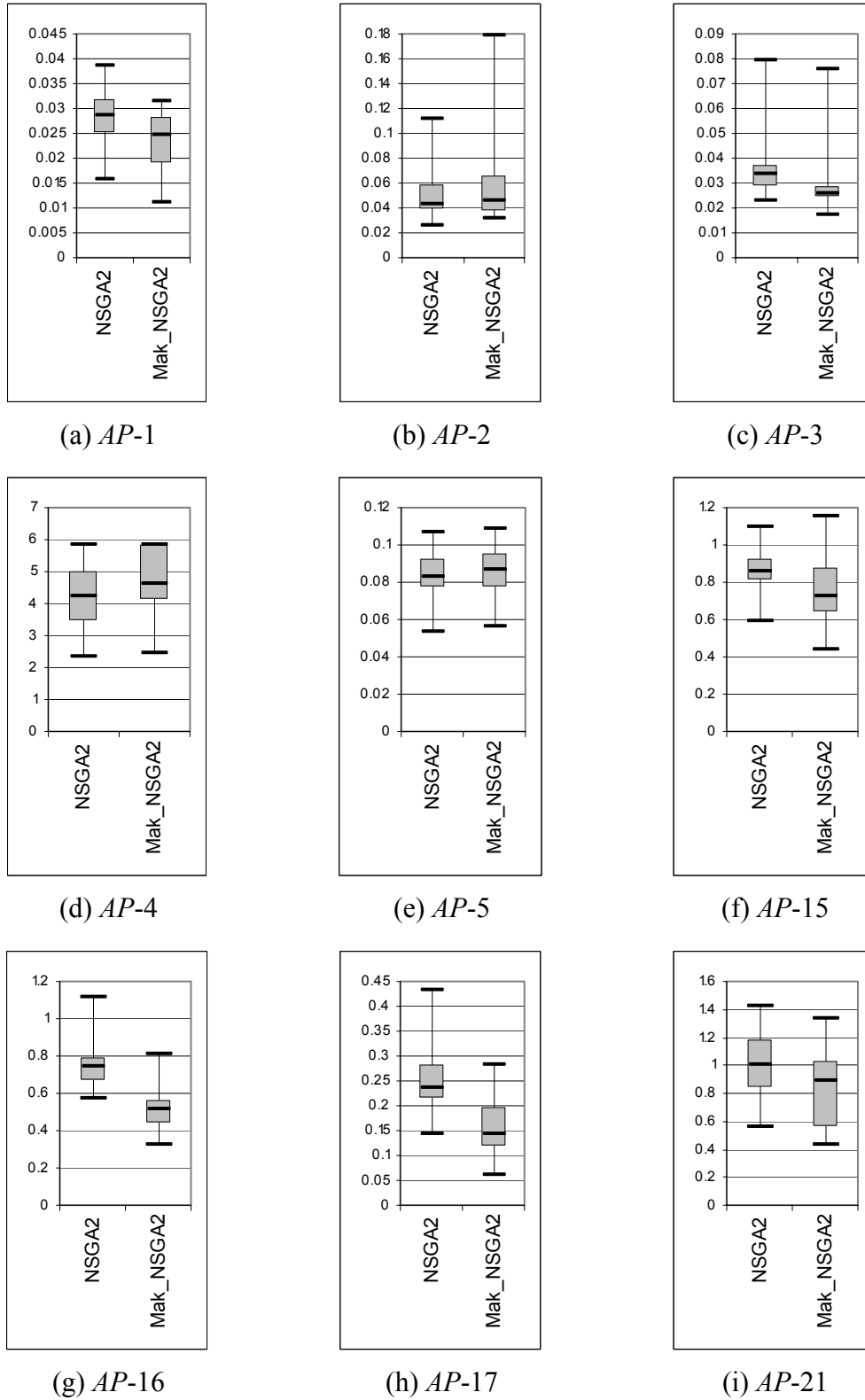
(e) AP-5

**Figure 120 — Progressive Hypervolume Averages**

NSGA-II: Blue; Mak\_NSGA-II: Green.  $y$ -axis is the average hypervolume performance;  $x$ -axis represents the number of evaluations executed by each optimiser.

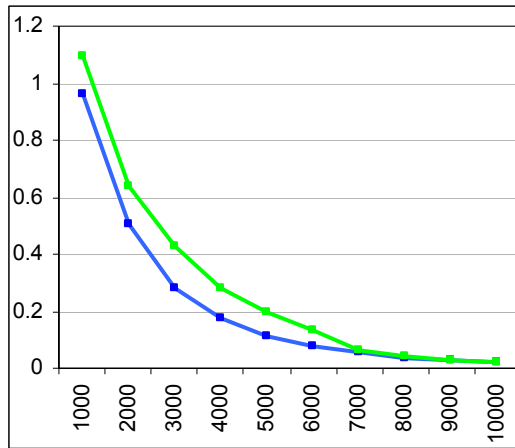
(a) *AP-15*(b) *AP-16*(c) *AP-17*(d) *AP-21***Figure 121 — Progressive Hypervolume Averages**

NSGA-II: Blue; Mak\_NSII: Green. *y-axis* is the average hypervolume performance; *x-axis* represents the number of evaluations executed by each optimiser.

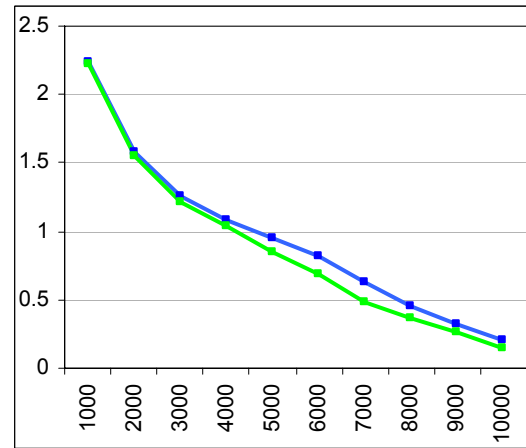


**Figure 122 — End-of-Run Hypervolume Box-Plots**

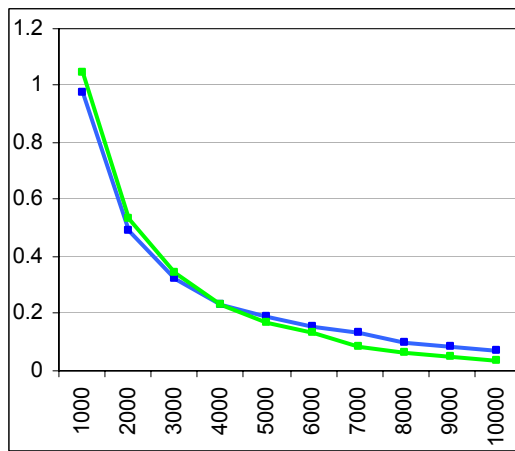
*y-axis* is the hypervolume performance at the end of the run (7,000 evaluations in *AP-5*, 10,000 evaluations in all remaining functions); *x-axis* indicates the selected optimiser.



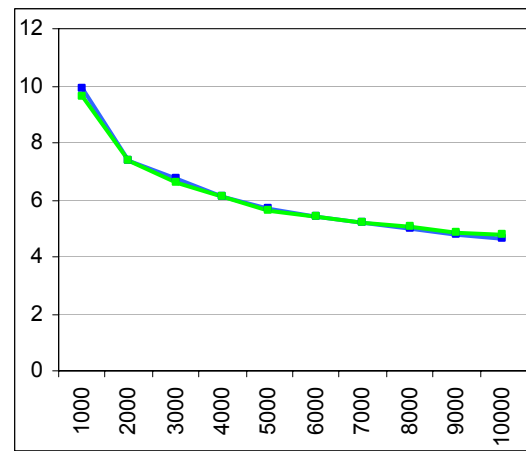
(a) AP-1



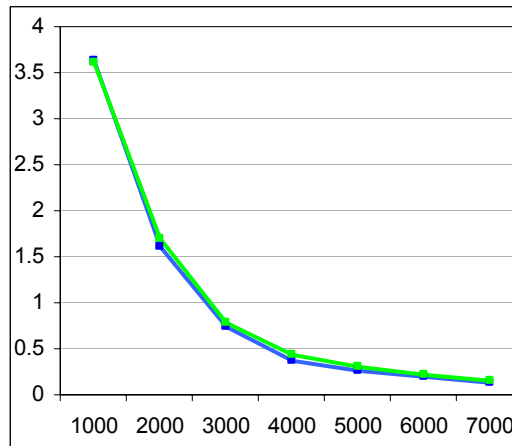
(b) AP-2



(c) AP-3

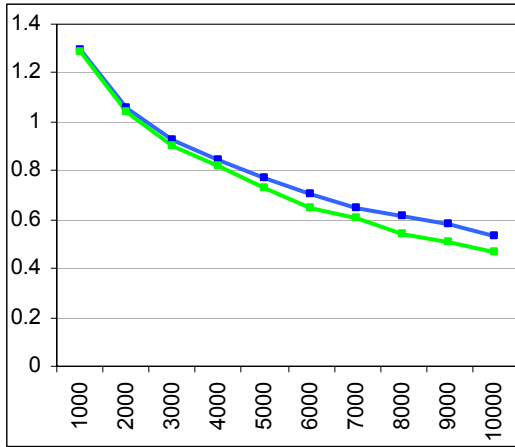


(d) AP-4

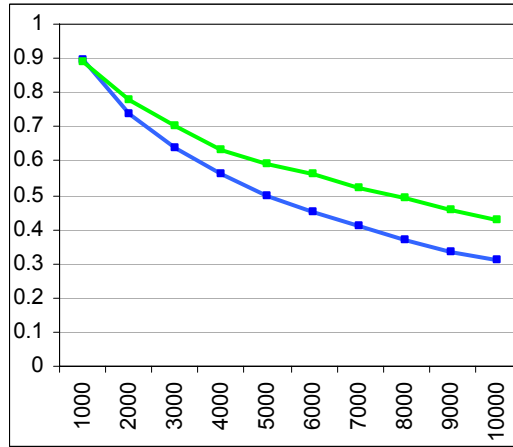


(e) AP-5

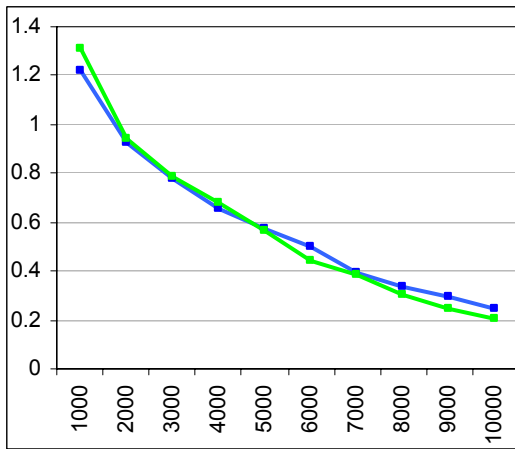
**Figure 123 — Progressive Epsilon Averages**  
 NSGA-II: Blue; Mak\_NSGA-II: Green. *y-axis* is the average hypervolume performance; *x-axis* represents the number of evaluations executed by each optimiser.



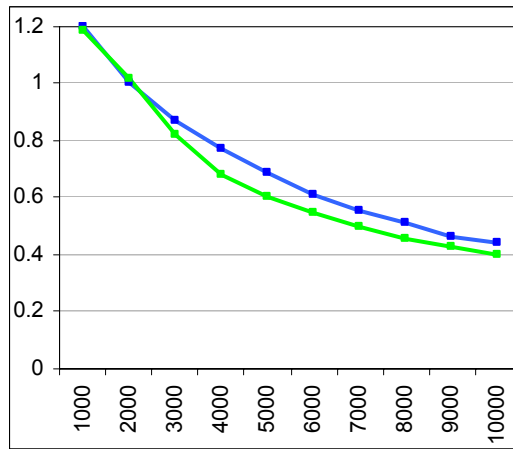
(a) AP-15



(b) AP-16



(c) AP-17

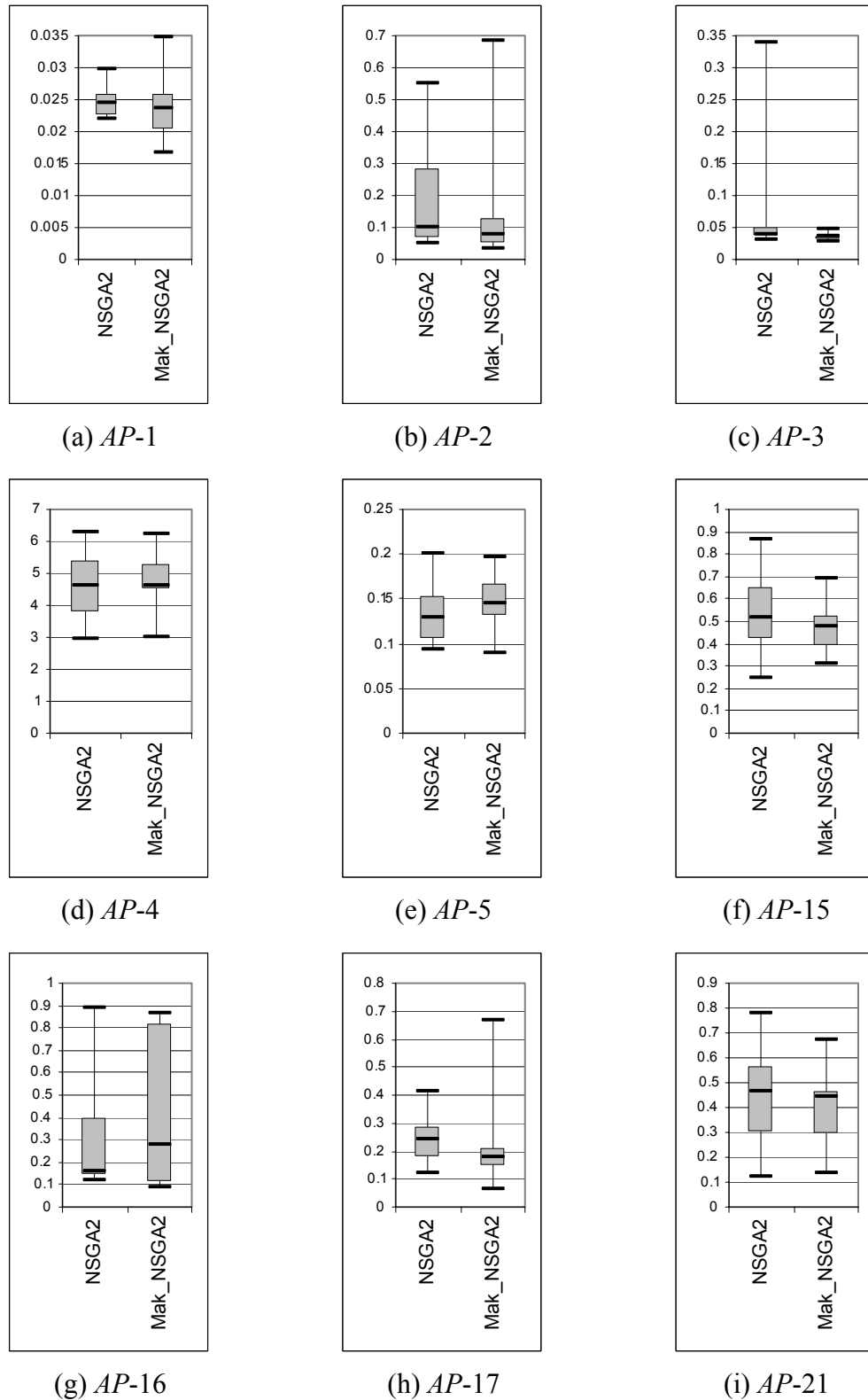


(d) AP-21

**Figure 124 — Progressive Epsilon Averages**

NSGA-II: Blue; Mak\_NSQA-II: Green. *y-axis* is the average epsilon performance; *x-axis* represents the number of evaluations executed by each optimiser.



**Figure 125 — End-of-Run Epsilon Box-Plots**

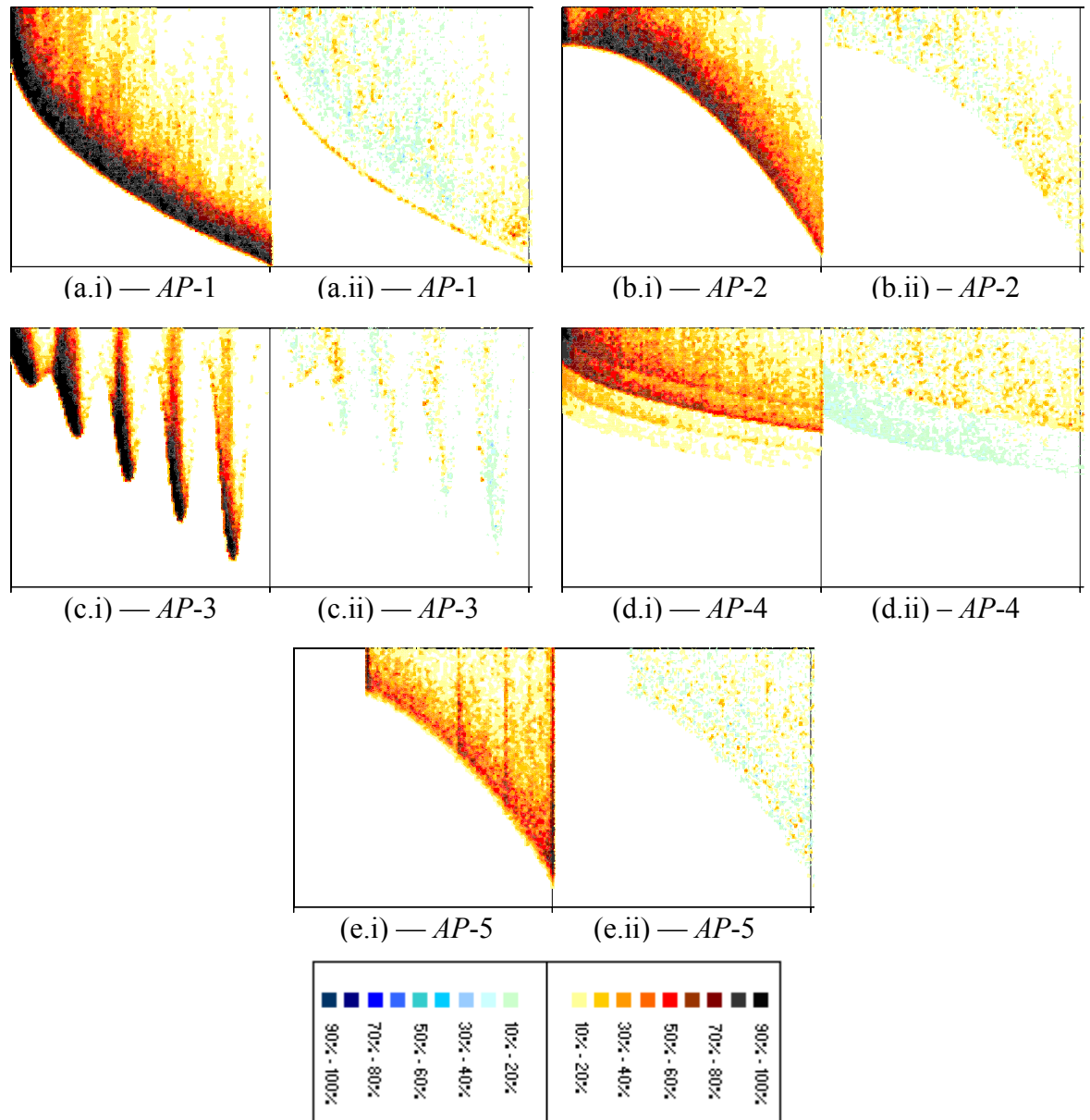
*y-axis* is the epsilon performance at the end of the run (7,000 evaluations in *AP-5*, 10,000 evaluations in all remaining functions); *x-axis* indicates the selected optimiser.

**Table 30 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Hypervolume Results**  
 Bold italics indicate significant differences at the 5% level.

	Hypervolume
<b><i>AP-1</i></b>	<b><i>3.340E-03</i></b>
<b><i>AP-2</i></b>	6.871E-01
<b><i>AP-3</i></b>	<b><i>2.076E-02</i></b>
<b><i>AP-4</i></b>	3.391E-01
<b><i>AP-5</i></b>	6.599E-01
<b><i>AP-15</i></b>	<b><i>1.377E-02</i></b>
<b><i>AP-16</i></b>	<b><i>2.110E-06</i></b>
<b><i>AP-17</i></b>	<b><i>5.887E-06</i></b>
<b><i>AP-21</i></b>	<b><i>4.062E-02</i></b>

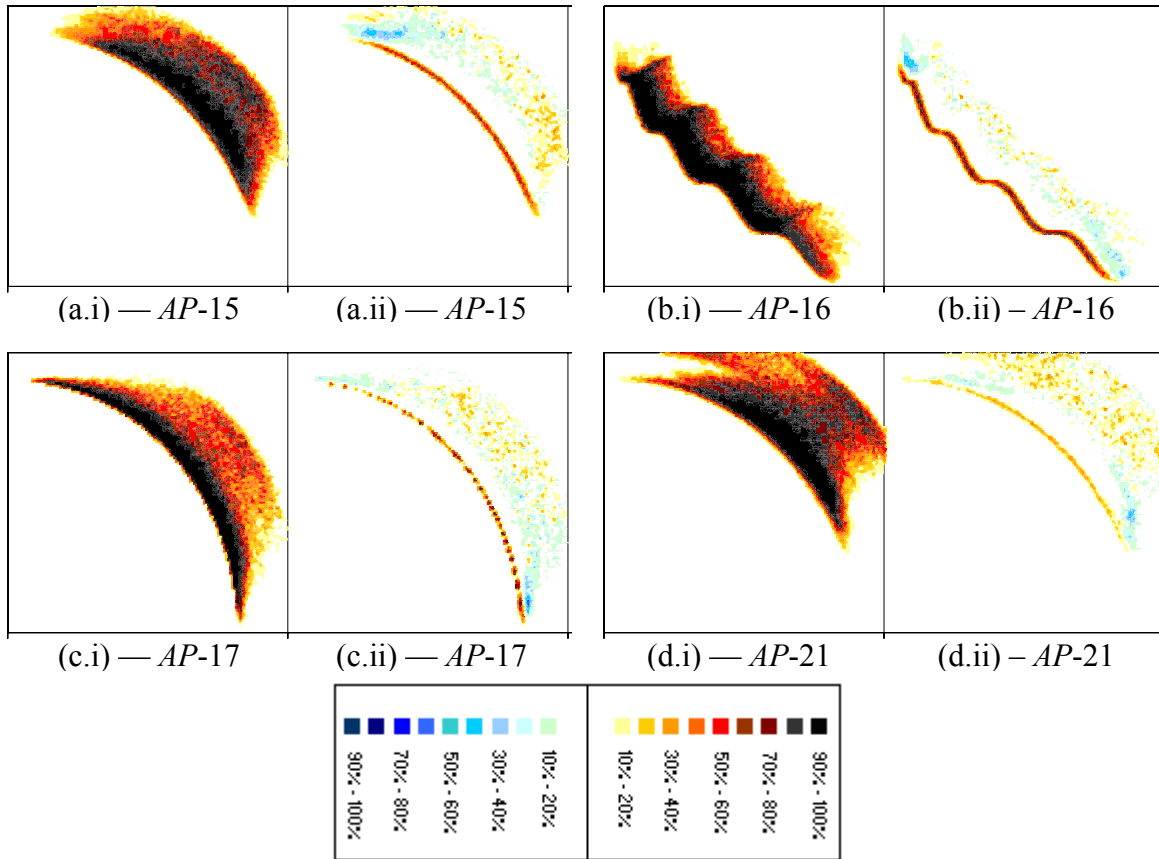
**Table 31 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Epsilon Results**  
 Bold italics indicate significant differences at the 5% level.

	Epsilon
<b><i>AP-1</i></b>	8.318E-01
<b><i>AP-2</i></b>	8.426E-02
<b><i>AP-3</i></b>	<b><i>1.364E-02</i></b>
<b><i>AP-4</i></b>	4.965E-01
<b><i>AP-5</i></b>	<b><i>1.261E-02</i></b>
<b><i>AP-15</i></b>	1.157E-01
<b><i>AP-16</i></b>	8.107E-01
<b><i>AP-17</i></b>	<b><i>4.230E-02</i></b>
<b><i>AP-21</i></b>	3.457E-01



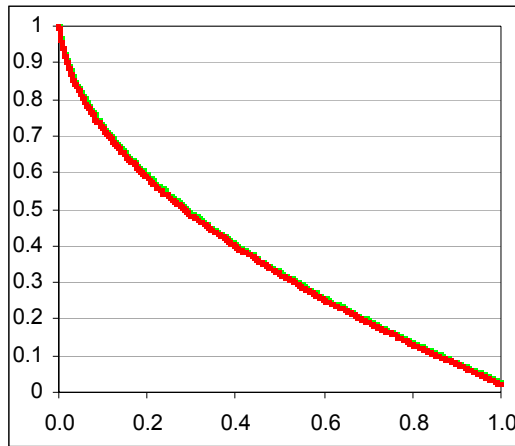
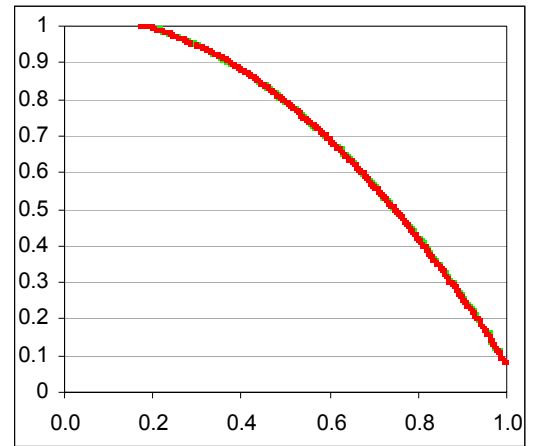
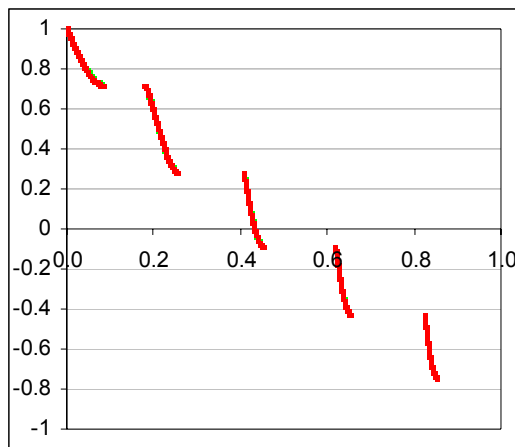
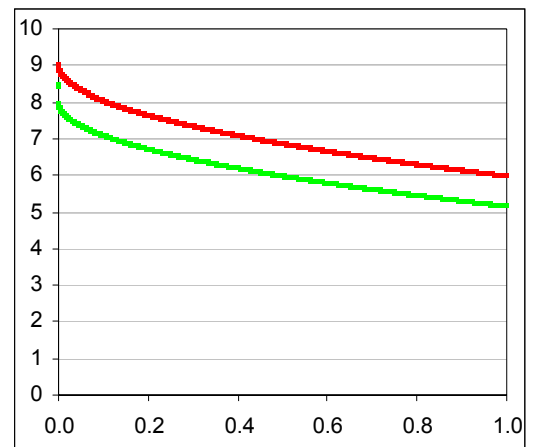
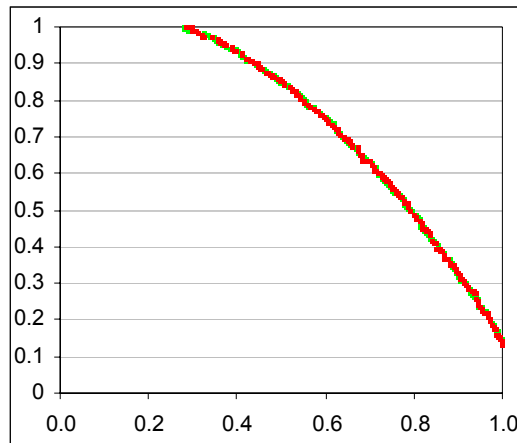
**Figure 126 — End-of-Run Frequency Matrices**

(i) Frequency matrix for Mak\_NSGA-II. (ii) The red palette represents where Mak\_NSGA-II more frequently attains a given region; the blue palette indicates where NSGA-II attains an area more consistently.

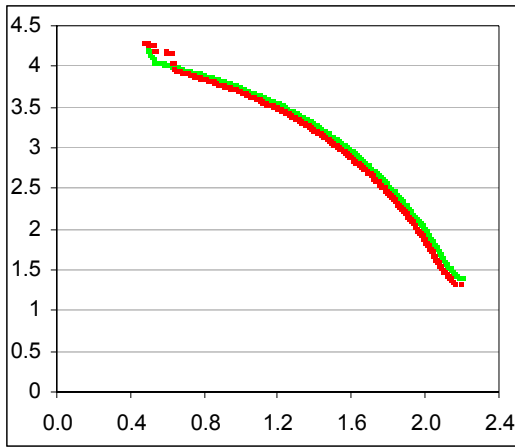


**Figure 127 — End-of-Run Frequency Matrices**

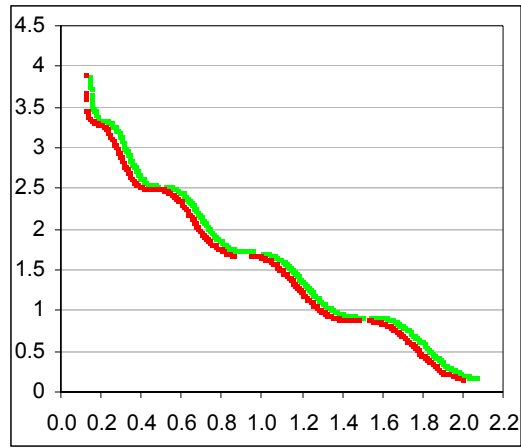
(i) Frequency matrix for Mak\_NSGA-II. (ii) The red palette represents where Mak\_NSGA-II more frequently attains a given region; the blue palette indicates where NSGA-II attains an area more consistently.

(a) *AP-1*(b) *AP-2*(c) *AP-3*(d) *AP-4*(e) *AP-5***Figure 128 — 50% Attainment Surfaces**

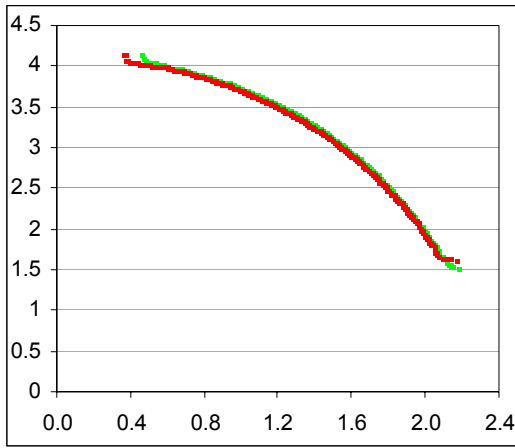
NSGA-II: Green, Mak\_NSGA-II: Red. *y*-axis is objective two; *x*-axis is objective one.



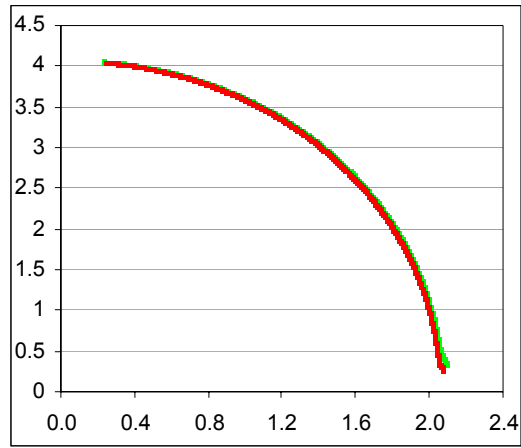
(a) *AP-15*



(b) *AP-16*



(c) *AP-17*



(d) *AP-21*

**Figure 129 — 50% Attainment Surfaces**

NSGA-II: Green, Mak\_NSGA-II: Red.  $y$ -axis is objective two;  $x$ -axis is objective one.

### 9.2.4 THE IMPROVED STRENGTH PARETO EVOLUTIONARY ALGORITHM (SPEA2)

The original SPEA algorithm [80, 137] must be considered one of the flag-bearers of elitism in multiobjective optimisation. As initial studies [92, 137, 232] emphasised the superiority of SPEA over existing, non-elitist, first-generation algorithms such as NSGA [2], VEGA (the Vector Evaluated Genetic Algorithm) [146] and NPGA [141, 142], the statement became clear: elitism was the way forward for multiobjective optimisation. Amongst those heeding this message were Deb and Corne, who ultimately went on to produce algorithms that surpassed the SPEA progenitor (specifically, with the NSGA-II and PESA techniques). In response, Zitzler refined the initial algorithm with subtle, though important, changes to fitness assignment, crowding estimation and archival truncation. The result was the impressive, though unsurprisingly titled, SPEA2 algorithm [81, 145].

#### 9.2.4.1 DESCRIBING SPEA2

The SPEA2 algorithm (see Algorithm 21) shares a great deal in common with the NSGA-II system explored in Section 9.2.3 — specifically, it features a constantly sized archive, binary tournament selection from the archive and fitness assignment that considers both crowding and dominance. It differs markedly though in the finer points of execution. In SPEA2, the fitness of a solution is not defined according to frontal membership, but is derived from the strength of solutions that dominate it, while ties in raw fitness scores are broken not by cuboid crowding estimates, but by  $\kappa^{\text{th}}$  nearest-neighbour approximations. Specifically, the final fitness of a solution  $\mathbf{a}$  is given as:

$$\mathbf{a}^{\text{fitness}} = r(\mathbf{a}) + \frac{1}{\delta_{\kappa}(\mathbf{a}) + 2} \quad (83)$$

Here,  $\delta_{\kappa}(\mathbf{a})$  is the  $\kappa^{\text{th}}$  nearest-neighbour score of  $\mathbf{a}$  and:

$$r(\mathbf{a}) = \sum \zeta(\mathbf{b}) \forall \mathbf{b} \in P : \mathbf{b} \prec \mathbf{a} \quad (84)$$

where  $\zeta(\mathbf{b})$  is the strength function: the number of solutions in the combined population  $P$  (see Algorithm 21) that are dominated by  $\mathbf{b}$ .

**Algorithm 21 — The SPEA2 Algorithm**

---

**Inputs:**

- $b$                       The maximum number of solutions allowed in the breeding pool.  
 $s$                       The maximum archive size.

```

1:  $Arch := \text{generateRandomPop}()$                       Initialise the starting population.
2:  $Parents := Arch$ 
3: while (terminationConditionMet()  $\neq$  true)
4:    $Children := \text{reproduce}(Parents)$                       Generate the child set from parents.
5:    $Parents := \emptyset$                       Add all children to the unprocessed archive and
6:    $Combined := Arch \cup Children$                       calculate fitness of members according to
7:   calculateFitness( $Combined$ )                      strength and crowding scores.
8:    $NonDom := \text{extractNonDominated}(Combined)$                       Remove non-dominated solutions from the
9:   while ( $|NonDom| > s$ )                      combined set and store them in  $nonDom$ .
10:    extractMostCrowded( $NonDom$ )                      Reduce the archival set to  $s$  if it is too large
11:    calculateCrowd( $NonDom$ )                      by removing the most crowded members.
12:    $Arch := NonDom$ 
13:   while ( $|Arch| < s$ )                      If the archive is too small, fill it with the
14:      $Arch := Arch \cup \{\text{extractFittest}(Combined)\}$                       best scoring dominated solutions.
15:   while ( $|Parents| < b$ )                      Perform binary tournament selection
16:     first := selectRandom( $Arch$ )                      with random members of the newly
17:     second := selectRandom( $Arch$ )                      updated archive.
18:     if (fitness(first) < fitness(second))                      The winner of each tournament match is
19:        $Parents := Parents \cup \{\text{first}\}$                       the solution with best fitness score.
20:     else
21:        $Parents := Parents \cup \{\text{second}\}$ 

```

---

It is worth noting that while the strength function is an effective way of emphasising valuable regions in the objective-space, it is rendered entirely ineffective so long as the number of non-dominated members in the population exceeds the maximum archival size. In this case, both the truncation procedure and binary tournament are governed entirely by the crowding operator (echoing the behaviour of NSGA-II in equivalent scenarios).

#### 9.2.4.2 INTEGRATING THE MAK\_TREE INTO SPEA2

As a final exploration of the ways in which the unbounded Mak\_Tree may be successfully integrated into existing algorithms, the SPEA2 system is extended via hybridisation. Specifically, so long as the Mak\_Tree contains fewer solutions than the truncated archive, SPEA2 is allowed to perform exactly as it does in standard operation (with no interference from the unbounded set whatsoever). When the Mak\_Tree breaches this artificial threshold, in lieu of the standard selection



methodology, globally uncrowded solutions are extracted from the unbounded set (as in Section 9.2.2.2) and passed to a crowding-based binary tournament<sup>74</sup>. Thus, the hybridised technique endeavours to capitalise on the strength-based fitness function when the leading front is small and the rich information provided by an unbounded set when it is large. Importantly, since the truncated archive will only be used when it is large enough to store the complete non-dominated front, it can be guaranteed that issues related to the dual nature of bounded sets (see Section 9.2.1.4, Section 9.2.2.4 and Section 9.2.3.4) are completely avoided. When used, the truncated archive will always contain (at least) the complete non-dominated front and, as such, dominance comparisons and crowding estimations are of high fidelity.

To investigate the impact of different crowding measures, a simple cuboid method is employed during the unbounded phase in Mak\_SPEA2, while a  $\kappa$  nearest-neighbours technique (using the averaging methodology described in Section 8.1.1.3) is similarly capitalised upon in Mak\_SPEA2\_KNN.

### **9.2.4.3 EMPIRICAL ANALYSIS METHODOLOGY**

The original SPEA algorithm is compared with the two alternative hybrid techniques according to the methodology described in Section 9.1.2. All three approaches use random starting populations, a crossover probability of 90% (with two children produced per breeding interaction), a mutation rate of  $1/m$  and binary tournaments that operate on fifty pairs of solutions. Both the original SPEA and the SPEA portion of the hybridised techniques feature truncated archives with exactly fifty solutions and use a  $\kappa^{\text{th}}$  nearest-neighbour technique ( $\kappa = 11$ ) for crowding estimation. During the unbounded portion of the hybrid algorithm, the fifty least crowded solutions are extracted from the extended Mak\_Tree (using the efficient methods described in Section 8.1), with randomly selected members of this set entering a crowd-based binary tournament. The Mak\_SPEA2 algorithm uses simple cuboid crowding as per Section 9.2.2.2, while the Mak\_SPEA2\_KNN system uses a  $\kappa$  nearest-neighbours technique ( $\kappa = 20$ )<sup>75</sup>. It is important to note that the unbounded portion of the archive is used only so long as there exists at least fifty distinct

---

<sup>74</sup> It is worth noting that both the extraction and binary tournaments use the same Mak\_Tree-derived global crowding estimates (unlike in the PESA extensions, where two differing global metrics are used).

<sup>75</sup> The larger value of  $\kappa$  (relative to the basic SPEA2 setting) is employed here due to the unbounded size of the Mak\_Tree-based archive.

members in the prevailing optimal front, with the hybrid algorithms reverting to the basic SPEA2 methodology whenever the leading front falls below this threshold level. As with preceding empirical investigations, all relevant result graphs and tables are provided at the completion of this sub-section (pages 302–314).

#### 9.2.4.4 EMPIRICAL ANALYSIS

The results are impressive and clearly illustrate the power of unbounded Mak\_Trees in hybrid environments. In particular, the end-of-run hypervolume results (Figure 134) and the corresponding significance tests (Table 32) demonstrate a *significant* performance advantage for Mak\_SPEA2 over the basic algorithm in *AP-1*, *AP-3*, *AP-5*, *AP-15*, *AP-16*, *AP-17* and *AP-21* (seven of the nine tested functions), while it is preferable (though not *significantly* so) in the remaining *AP* functions. The progressive average performance (Figure 132 and Figure 133) of the cuboid-based Mak\_SPEA2 approach is also strong, with it outperforming the original technique across the majority of the run in *AP-4*, *AP-15*, *AP-16*, *AP-17* and *AP-21*. In contrast, the SPEA2 technique delivers consistently preferable average progressive performance only in *AP-2*.

The epsilon-derived results are also positive for the Mak\_SPEA2 technique. Specifically, end-of-run performance (Figure 137) is *significantly* better (Table 33) for the cuboid-based hybridised approach in *AP-1*, *AP-3*, *AP-5*, *AP-16*, *AP-17* and *AP-21* (six of the nine tested functions) and is never *significantly* worse. Similarly, average performance (Figure 135 and Figure 136) of the Mak\_SPEA2 algorithm is markedly better across the majority of the run in *AP-1*, *AP-3*, *AP-4*, *AP-17* and *AP-21*.

Interestingly, the performance of the averaged  $\kappa$  nearest-neighbours-based SPEA2 hybrid is inferior to that of its cuboid-based relative. Examining end-of-run hypervolume performance (Figure 134), the Mak\_SPEA2 algorithm is preferable to Mak\_SPEA2\_KNN on every tested function and is *significantly* better (Table 32) on *AP-1*, *AP-3*, *AP-5*, *AP-16* and *AP-21* (five of the nine tested functions). Similarly, end-of-run epsilon indicators (Figure 137) and significance tests (Table 33) illustrate *significant* performance gains in the cuboid-based approach in *AP-1*, *AP-3*, *AP-5*, *AP-16* and *AP-21*. The probable cause for such inferiority is related to the size of the neighbourhood used in the KNN approach — at twenty, it is simply too large for

most of the problems encountered. Given that Zitzler *et al.* [81] recommend a neighbourhood size of approximately  $\sqrt{n}$  for nearest-neighbour-based estimations and that the cardinality of the archive is rarely greater than 400 for any substantial portion of the run (see Figure 130), the KNN crowding estimations are likely to be misleading. This is the bias/variance dilemma (see Section 8.1.1) in action: the neighbourhood represents too large a portion of the explored front and so de-emphasises the importance of locally uncrowded solutions. Moreover, by using an inappropriately large  $\kappa$  value, the diversity of the extracted set is diminished (since solutions residing in uncrowded regions will share similar crowding estimates). This can be explored empirically: consider Figure 131 — when  $\kappa=10$ , the extracted solutions cover a large portion of the objective-space, but when  $\kappa=20$  the objective-space diversity of the extraction set is severely affected. The exploration of hybrid performance under more appropriately chosen  $\kappa$  values therefore rests as an interesting avenue of future work (as does research into heuristics for setting  $\kappa$  in an unbounded environment and the application of adaptive  $k$  values in Mak\_Trees).

Still, despite the clear advantage afforded to the apparently superior cuboid crowding method, the MAK\_SPEA2\_KNN algorithm yields generally better results than those of the basic SPEA2 system. Indeed, end-of-run hypervolume results (Figure 134 and Table 32) indicate that the Mak\_SPEA2\_KNN approach is *significantly* better than the SPEA approach in *AP-15*, *AP-16* and *AP-17*, and is never *significantly* worse. With respect to epsilon results (Figure 137 and Table 33), the Mak\_SPEA2\_KNN technique is *significantly* better on *AP-16*, *AP-17* and *AP-21*, and is never *significantly* outperformed.

Although the KNN-based results are impressive, the cuboid technique remains clearly superior and, as a consequence, the visual analyses focus on the comparison of the Mak\_SPEA2 hybrid technique and the truncated SPEA2 approach. Frequency matrices (Figure 138 and Figure 139) illustrate that, in general, the Mak\_SPEA2 system more consistently locates valuable fronts than in the basic truncated approach (only *AP-4* suggests inconclusive results), with *significant* differences in *AP-1*, *AP-3*, *AP-15*, *AP-16*, *AP-17* and *AP-21* (six of the nine tested functions). The frequency matrices also elucidate some interesting run-time properties of the cuboid-based hybridised system. In particular, *AP-3* shows a slightly more frequent exploration of

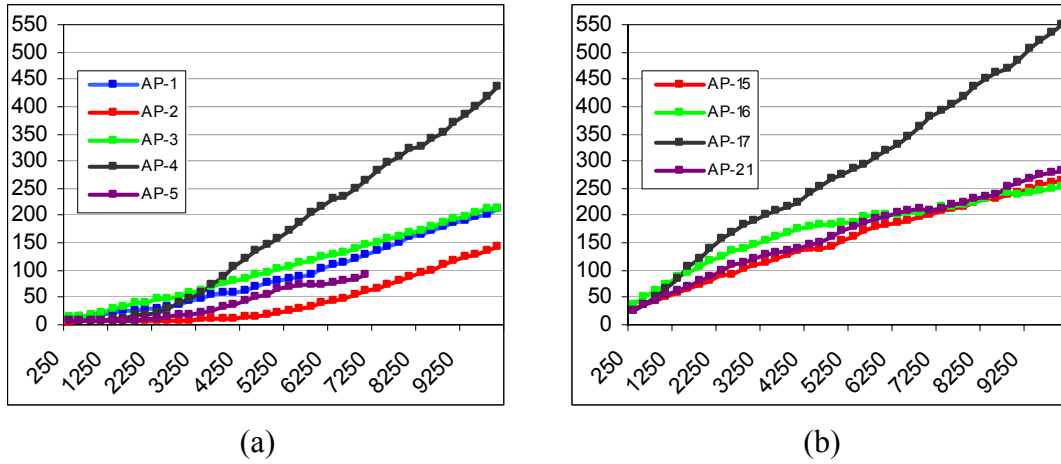
the extremities of each disconnected region (likely because the cuboid crowding estimate will favour these extremes — as discussed in 9.2.3.4), while *AP-21* indicates that the hybrid technique is more frequently drawn to the deceptive region in the upper-left of the objective-space, but that it is also capable of escaping it to produce impressive fronts (escape is aided by having access to the complete frontal set).

The median end-of-run attainment surfaces (Figure 140 and Figure 141) further advance the notion that the Mak\_SPEA2 technique is preferable to the basic SPEA2 approach. In particular, results indicate better performance under this metric for the hybridised approach in *AP-4*, *AP-5*, *AP-15*, *AP-16*, *AP-17* and *AP-21*. Moreover, the Mak\_SPEA2 technique is never clearly inferior to the median fronts produced by SPEA2.

Looking at the performance measures *in toto*, there can be little debating the superiority of the Mak\_SPEA2 algorithm over the contemporary, and powerful, SPEA2 technique. Indeed, the end-of-run performance of Mak\_SPEA2 is *significantly* better under at least one metric on every function other than *AP-2* and *AP-4*, and is *significantly* better under two or more metrics on seven of the nine tested functions: namely, *AP-1*, *AP-3*, *AP-5*, *AP-15*, *AP-16*, *AP-17* and *AP-21*. Across all numerical metrics, the end-performance of SPEA2 is never significantly better than the cuboid-based hybridised approach, while it is inferior to, or no better than, all attainment surfaces and frequency matrices produced by the new technique.

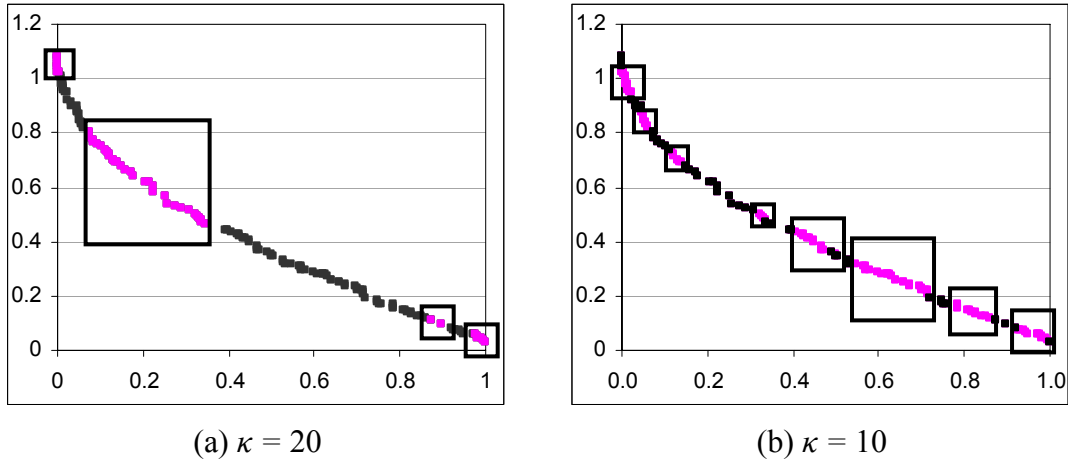
The reasons for such superiority are similar to those stated elsewhere (see Section 9.2.1.4, Section 9.2.2.4 and Section 9.2.3.4). In particular, once the true leading front is larger than the capacity of the archive, the SPEA2 algorithm is open to inaccurate crowding estimations and the acceptance of weak solutions into the supposedly elite store. The hybridised approach precludes such problems by granting access to the unbounded set when the leading front becomes too large for the truncated archive to function effectively. Moreover, by using a hybrid methodology, the Mak\_SPEA2 and Mak\_SPEA2\_KNN approaches capitalise on the strength-based fitness measures when the truncated archive *can* be used effectively, allowing for more than just the (potentially small) leading front to contribute to the search. The end result is a technique whose function better matches the intentions of the SPEA2 algorithm —

namely, the acceleration of the search via strength measures when the front is small and the expansion of the search via crowding when the front is large.



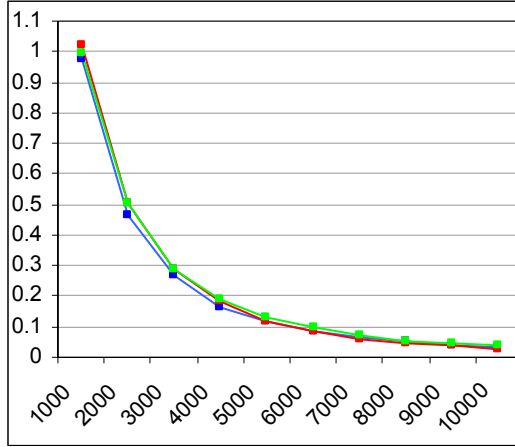
**Figure 130 - The Progressive Size of the Mak\_SPEA2\_KNN Unbounded Sets**

$y$ -axis is the average number of solutions in the unbounded elite archive;  $x$ -axis is the number of evaluations performed. Note that the elite archive populations are small relative to the size of  $\kappa$  used (specifically,  $\kappa = 20$ ).

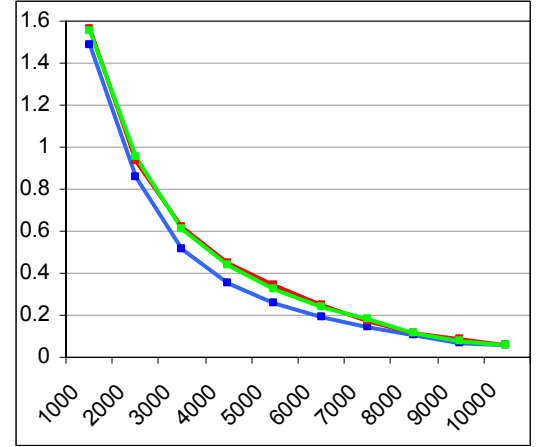


**Figure 131 — Example Selections with Averaged  $\kappa$  Nearest-Neighbours**

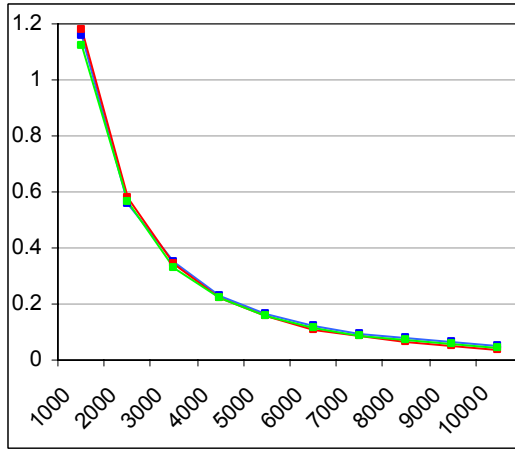
$y$ -axis is objective-two;  $x$ -axis is objective one. Pink represents fifty selected solutions from an archive containing 152 members; boxes used for emphasis. Note that (b) represents a more diverse selection of solutions than (a) due to its smaller  $\kappa$  value.



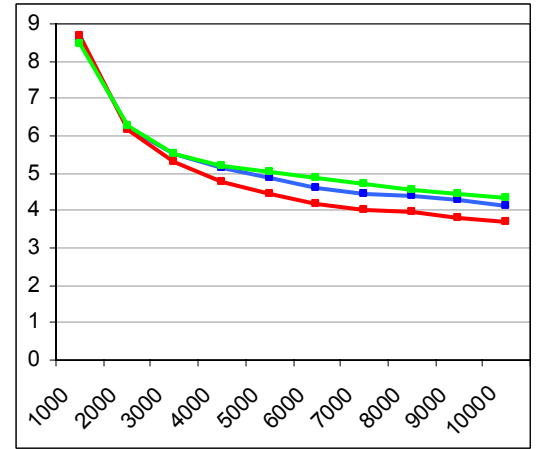
(a) AP-1



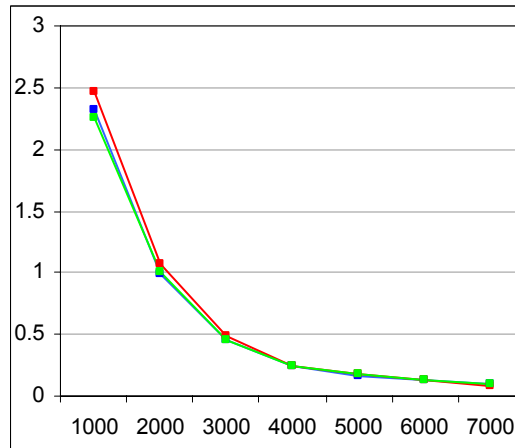
(b) AP-2



(c) AP-3



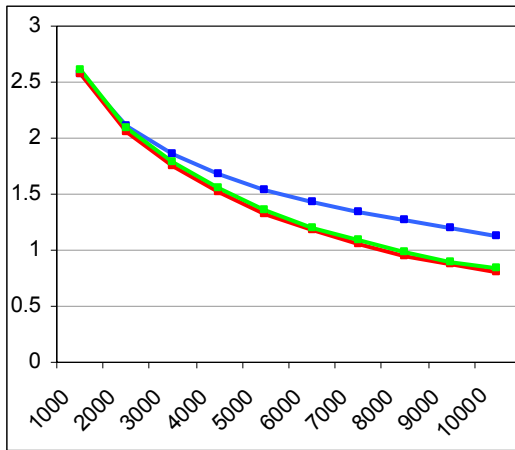
(d) AP-4



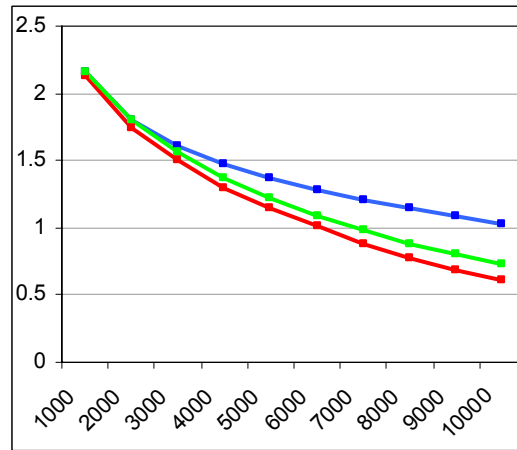
(e) AP-5

**Figure 132 — Progressive Hypervolume Averages**

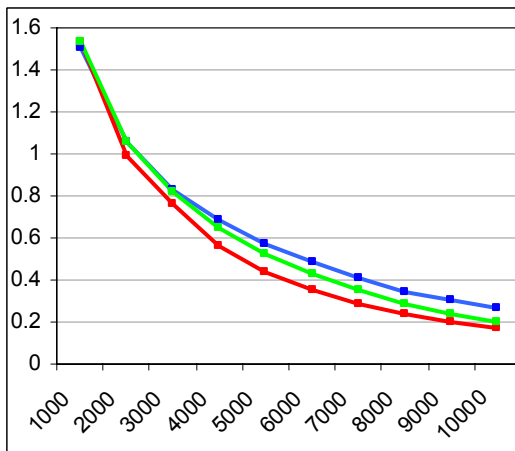
SPEA2: Blue; Mak\_SPEA2: Red; Mak\_SPEA2\_KNN: Green. *y-axis* is the average hypervolume performance; *x-axis* represents the number of evaluations executed by each optimiser.



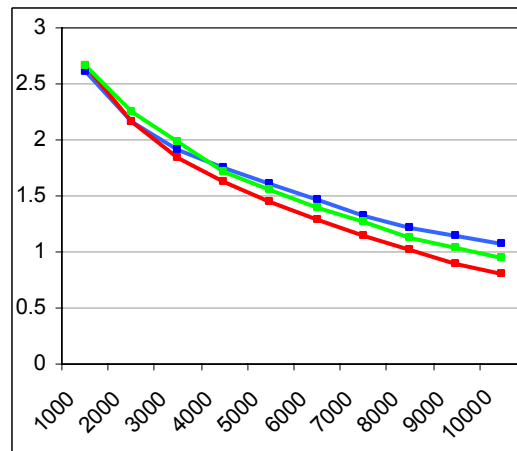
(a) AP-15



(b) AP-16



(c) AP-17

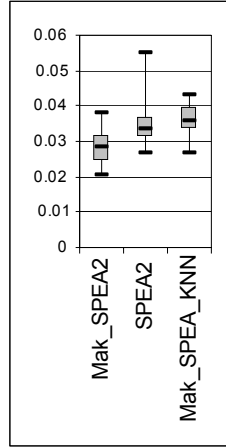
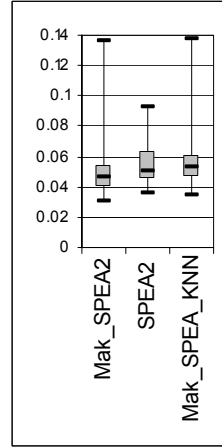
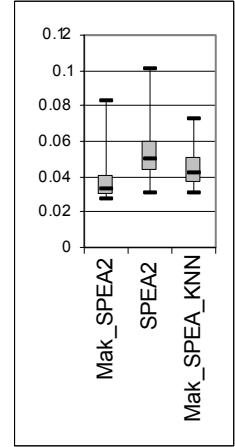
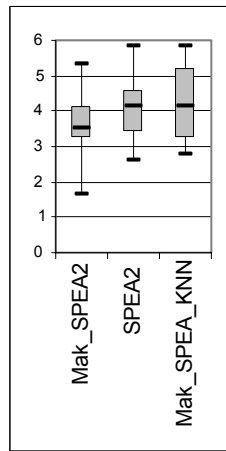
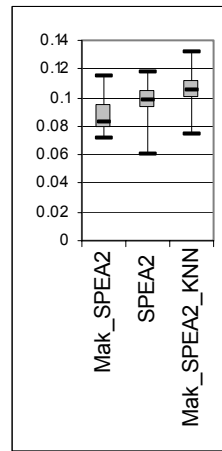
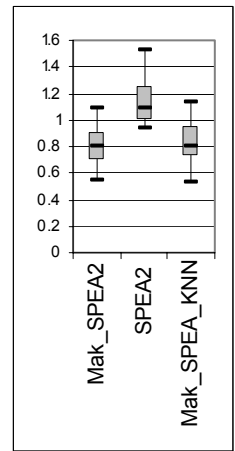
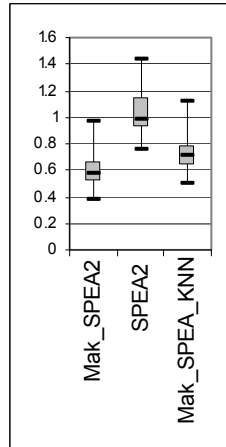
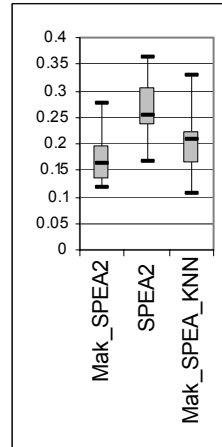
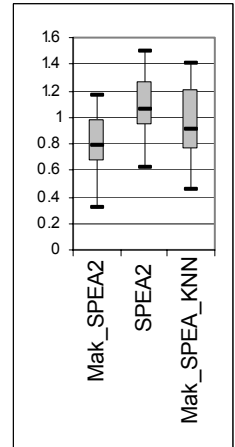


(d) AP-21

**Figure 133 — Progressive Hypervolume Averages**

SPEA2: Blue; Mak\_SPEA2: Red; Mak\_SPEA2\_KNN: Green. *y-axis* is the average hypervolume performance; *x-axis* represents the number of evaluations executed by each optimiser.

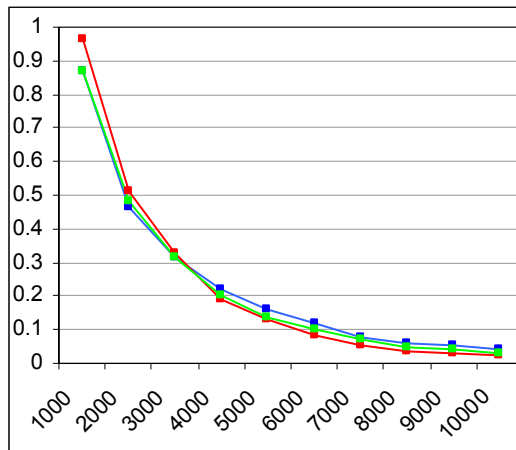


(a) *AP-1*(b) *AP-2*(c) *AP-3*(d) *AP-4*(e) *AP-5*(f) *AP-15*(g) *AP-16*(h) *AP-17*(i) *AP-21***Figure 134 — End-of-Run Hypervolume Box-Plots**

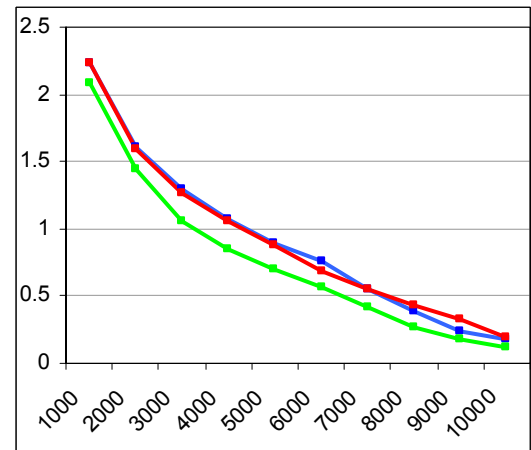
*y*-axis is the hypervolume performance at the end of the run (7,000 evaluations in *AP-5*, 10,000 evaluations in all remaining functions); *x*-axis indicates the selected optimiser.

**Table 32 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Hypervolume Results**  
 Bold italics indicate significant differences at the 5% level.

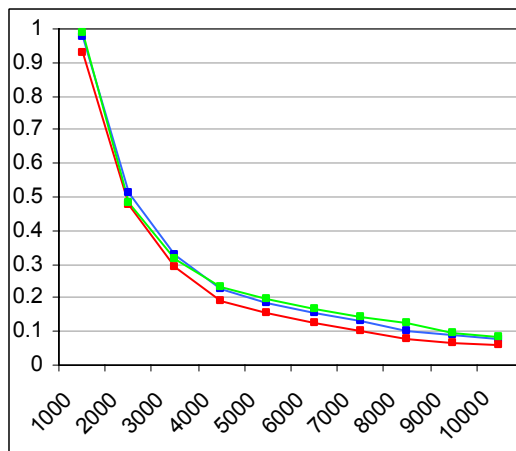
				(a) AP-1
	SPEA2	Mak_ SPEA2	Mak_ SPEA2_ KNN	
SPEA2	-	<b><i>1.37E-03</i></b>	4.67E-01	
Mak_ SPEA2	3.16E-01	-	<b><i>9.42E-05</i></b>	
Mak_ SPEA2_ KNN	9.51E-01	2.88E-01	-	
(b) AP-2				
				(c) AP-3
	SPEA2	Mak_ SPEA2	Mak_ SPEA2_ KNN	
SPEA2	-	<b><i>1.45E-03</i></b>	2.41E-01	
Mak_ SPEA2	1.16E-01	-	<b><i>4.21E-02</i></b>	
Mak_ SPEA2_ KNN	7.85E-01	6.54E-02	-	
(d) AP-4				
				(e) AP-5
	SPEA2	Mak_ SPEA2	Mak_ SPEA2_ KNN	
SPEA2	-	<b><i>6.09E-03</i></b>	7.66E-02	
Mak_ SPEA2		-	<b><i>8.29E-06</i></b>	
				(f) AP-15
	SPEA2	Mak_ SPEA2	Mak_ SPEA2_ KNN	
SPEA2	-	<b><i>1.31E-11</i></b>	<b><i>5.67E-10</i></b>	
Mak_ SPEA2	<b><i>5.58E-15</i></b>	-	5.26E-01	
Mak_ SPEA2_ KNN	<b><i>2.01E-08</i></b>	<b><i>1.36E-02</i></b>	-	
(g) AP-16				
				(h) AP-17
	SPEA2	Mak_ SPEA2	Mak_ SPEA2_ KNN	
SPEA2	-	<b><i>3.40E-07</i></b>	<b><i>1.01E-04</i></b>	
Mak_ SPEA2	<b><i>1.17E-04</i></b>	-	2.02E-01	
Mak_ SPEA2_ KNN	6.58E-02	<b><i>4.04E-02</i></b>	-	
(i) AP-21				



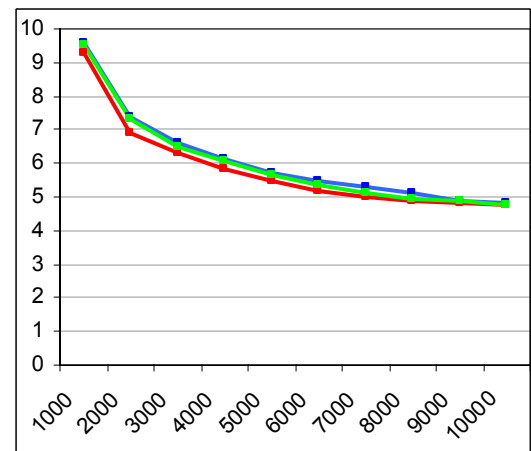
(a) AP-1



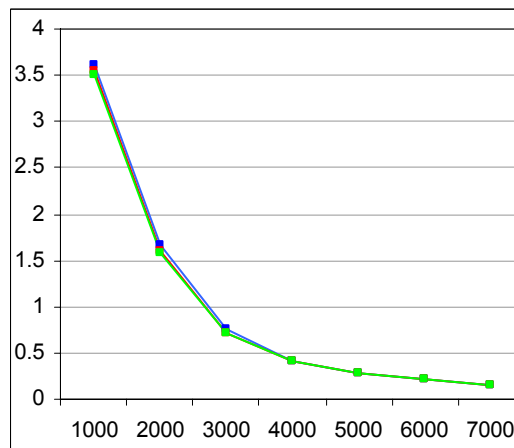
(b) AP-2



(c) AP-3



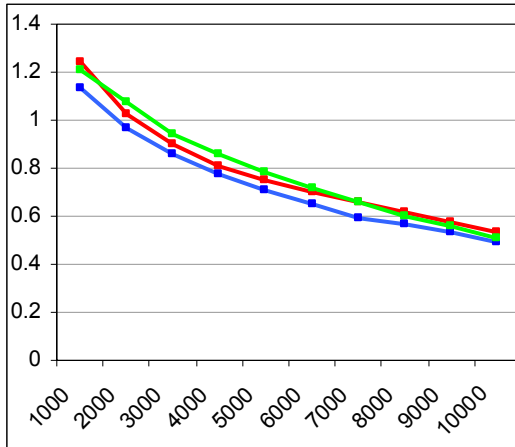
(d) AP-4



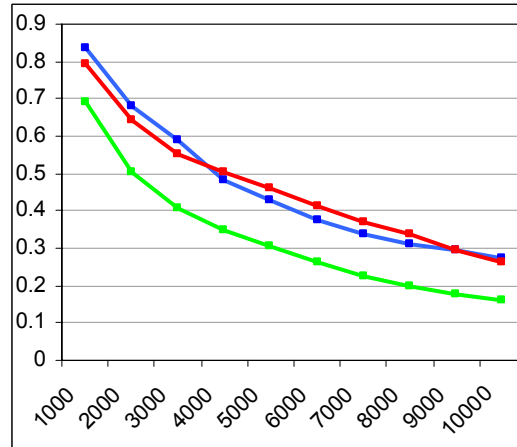
(e) AP-5

Figure 135 — Progressive Epsilon Averages

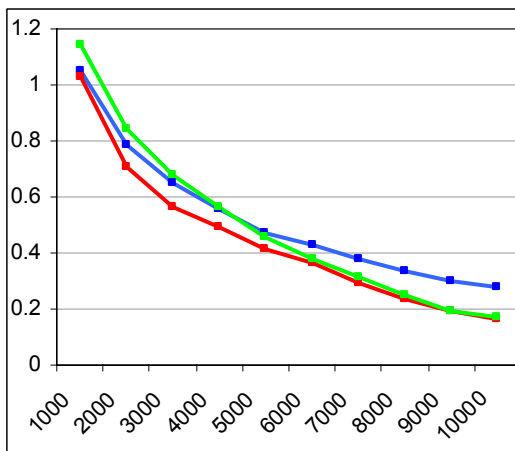
SPEA2: Blue; Mak\_SPEA2: Red; Mak\_SPEA2\_KNN: Green. *y-axis* is the average hypervolume performance; *x-axis* represents the number of evaluations executed by each optimiser.



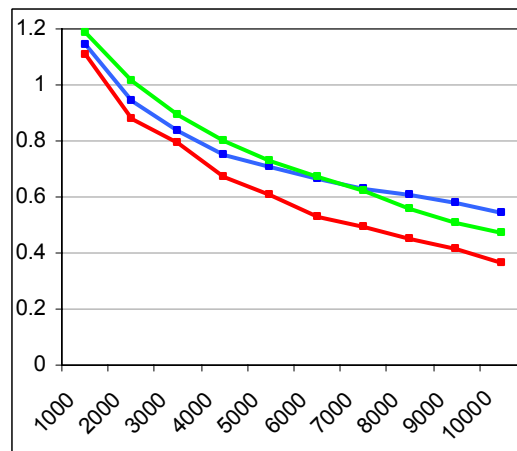
(a) AP-15



(b) AP-16



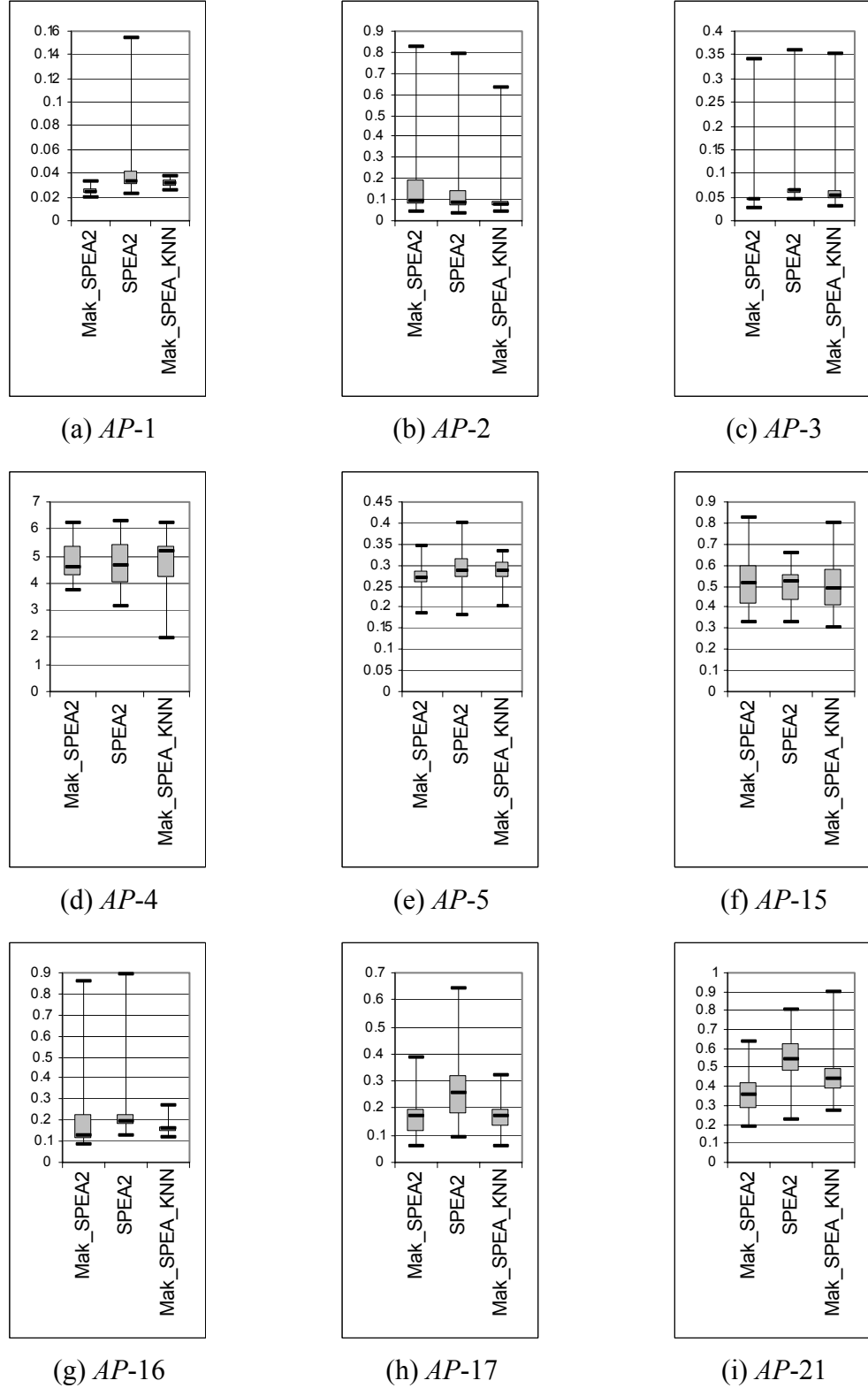
(c) AP-17



(d) AP-21

**Figure 136 — Progressive Epsilon Averages**

SPEA2: Blue; Mak\_SPEA2: Red; Mak\_SPEA2\_KNN: Green. *y-axis* is the average epsilon performance; *x-axis* represents the number of evaluations executed by each optimiser.

**Figure 137 — End-of-Run Epsilon Box-Plots**

*y-axis* is the epsilon performance at the end of the run (7,000 evaluations in *AP-5*, 10,000 evaluations in all remaining functions); *x-axis* indicates the selected optimiser.

**Table 33 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Epsilon Results**  
 Bold italics indicate significant differences at the 5% level.

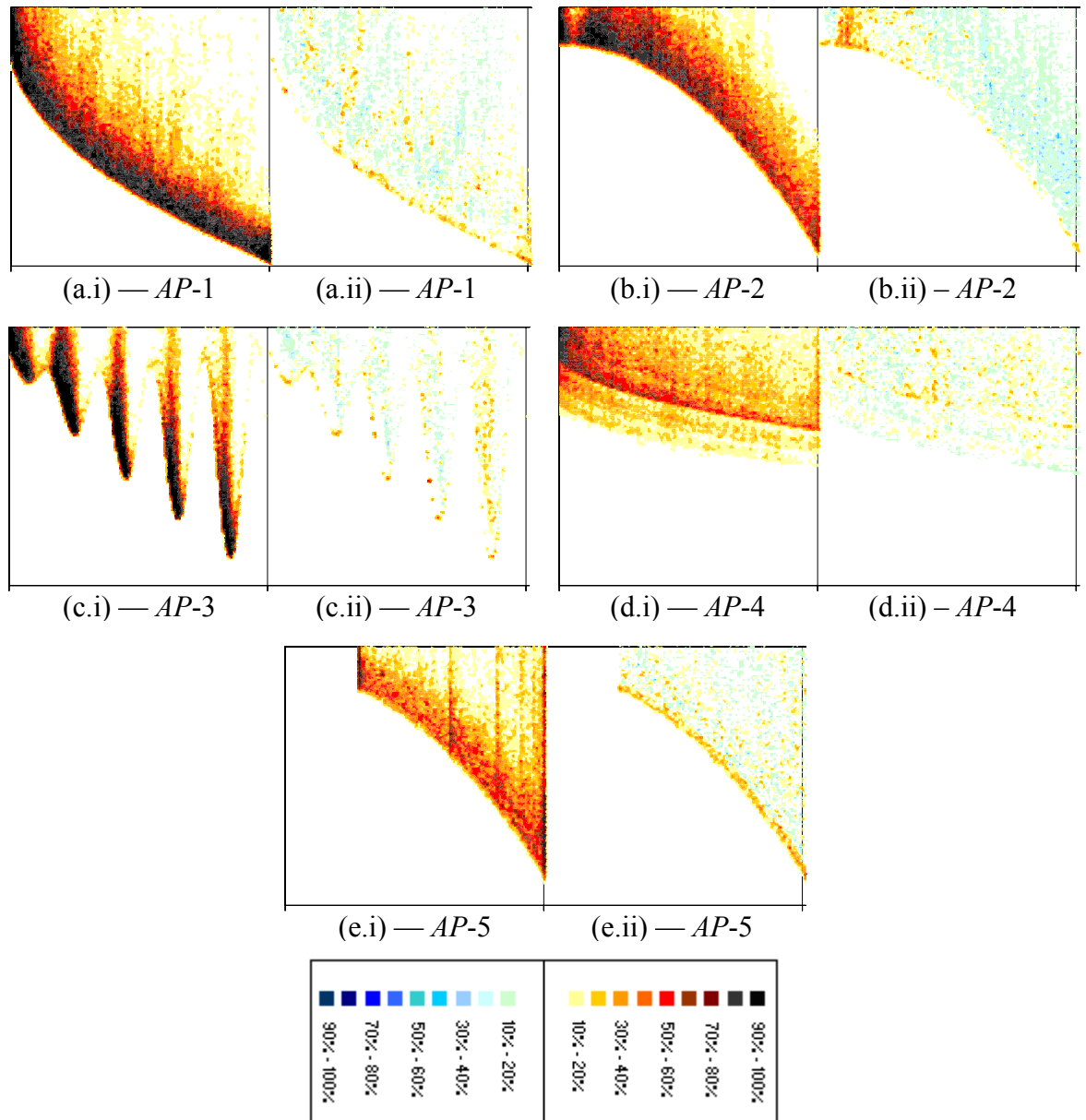
				(a) AP-1
	SPEA2	Mak_ SPEA2	Mak_ SPEA2_ KNN	
SPEA2	-	1.06E-06	1.60E-01	
Mak_ SPEA2	5.47E-01	-	4.05E-04	
Mak_ SPEA2_ KNN	5.84E-01	2.50E-01	-	(b) AP-2

				(c) AP-3
	SPEA2	Mak_ SPEA2	Mak_ SPEA2_ KNN	
SPEA2	-	4.01E-04	1.12E-01	
Mak_ SPEA2	7.26E-01	-	4.77E-02	
Mak_ SPEA2_ KNN	9.63E-01	6.91E-01	-	(d) AP-4

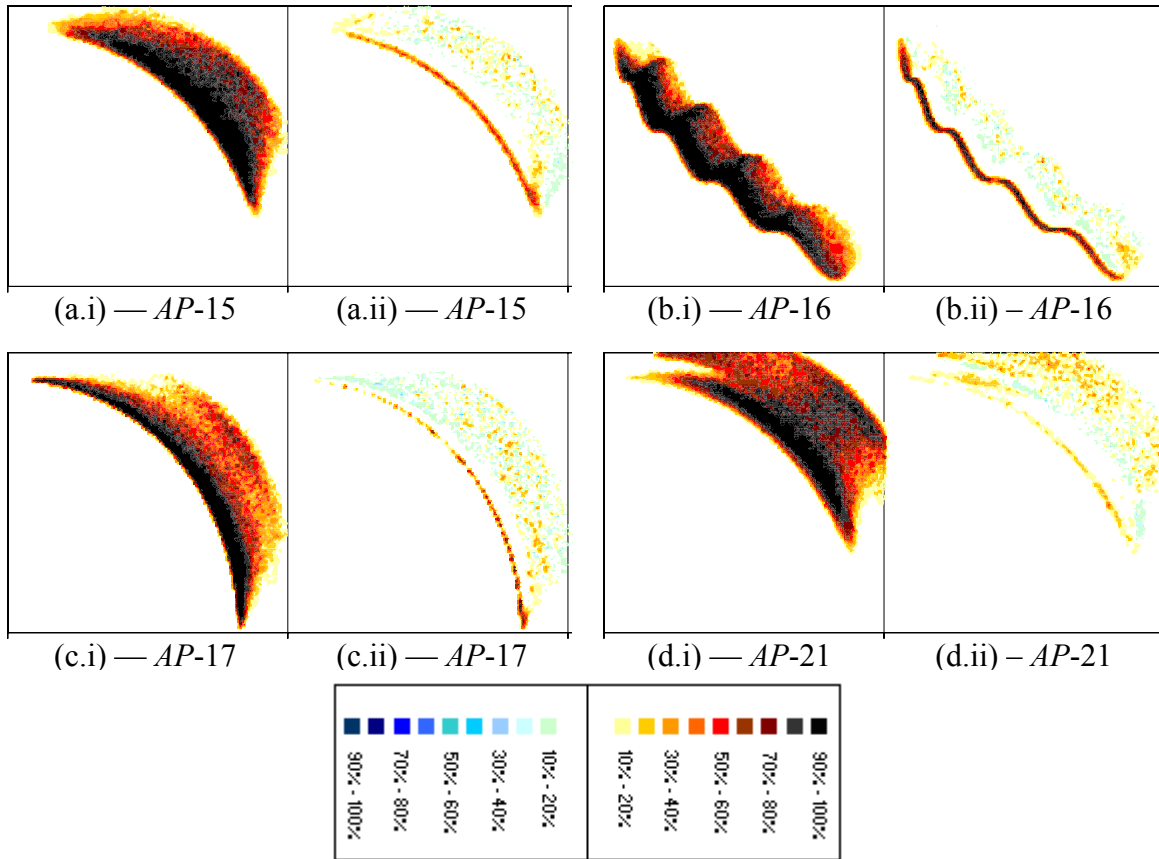
				(e) AP-5
	SPEA2	Mak_ SPEA2	Mak_ SPEA2_ KNN	
SPEA2	-	2.35E-02	7.76E-01	
Mak_ SPEA2		-	1.09E-02	

				(f) AP-15
	SPEA2	Mak_ SPEA2	Mak_ SPEA2_ KNN	
SPEA2	-	3.90E-01	8.48E-01	
Mak_ SPEA2	2.20E-02	-	5.04E-01	
Mak_ SPEA2_ KNN	7.51E-03	6.97E-01	-	(g) AP-16

				(h) AP-17
	SPEA2	Mak_ SPEA2	Mak_ SPEA2_ KNN	
SPEA2	-	2.18E-04	4.55E-04	
Mak_ SPEA2	5.32E-05	-	8.43E-01	
Mak_ SPEA2_ KNN	4.36E-02	3.88E-02	-	(i) AP-21

**Figure 138 — End-of-Run Frequency Matrices**

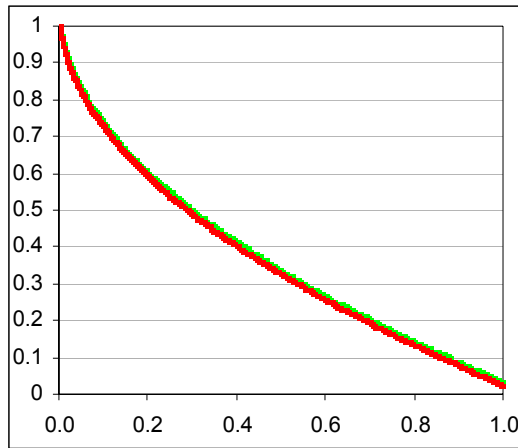
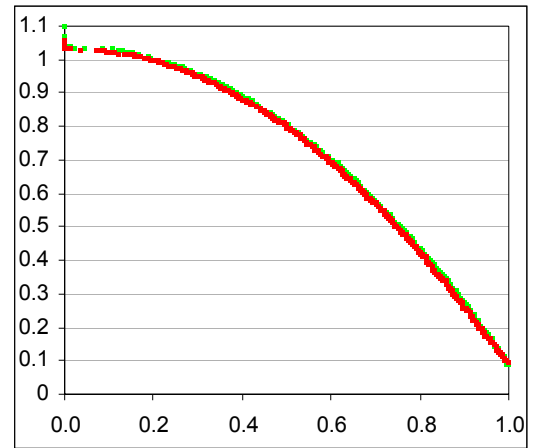
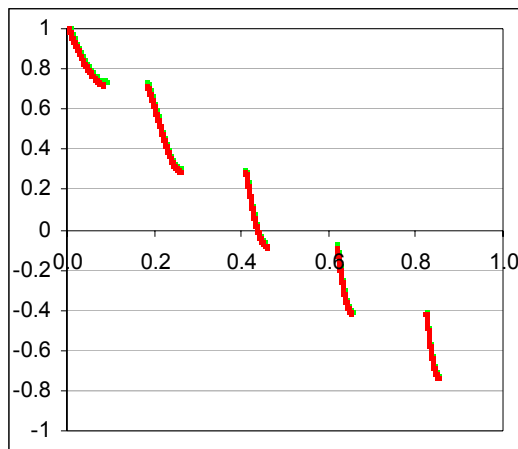
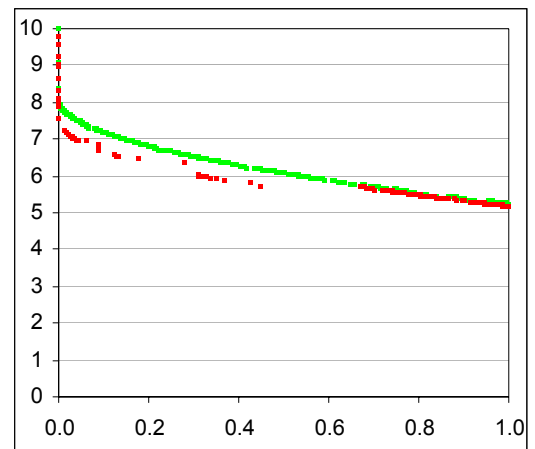
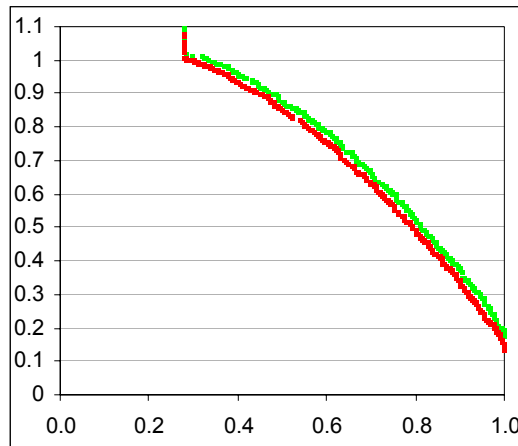
(i) Frequency matrix for Mak\_SPEA2. (ii) The red palette represents where Mak\_SPEA2 more frequently attains a given region; the blue palette indicates where SPEA2 attains an area more consistently.



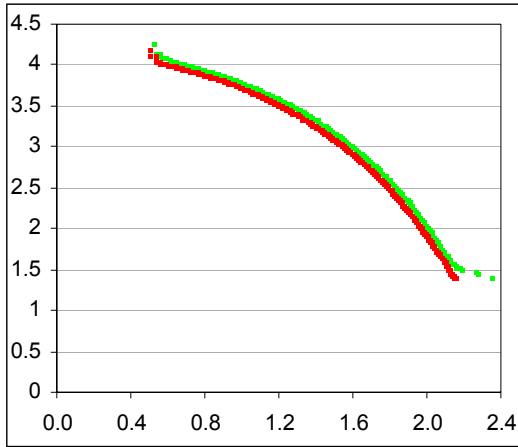
**Figure 139 — End-of-Run Frequency Matrices**

(i) Frequency matrix for Mak\_SPEA2. (ii) The red palette represents where Mak\_SPEA2 more frequently attains a given region; the blue palette indicates where SPEA2 attains an area more consistently.

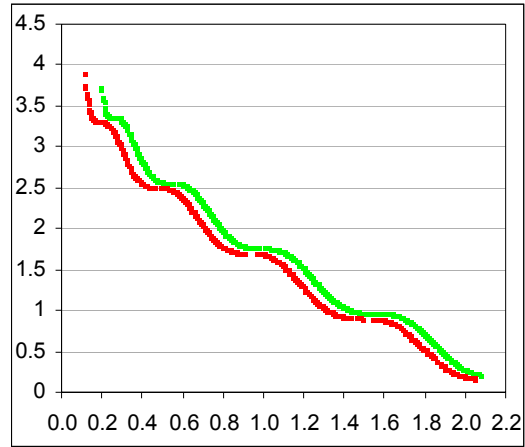


(a) *AP-1*(b) *AP-2*(c) *AP-3*(d) *AP-4*(e) *AP-5*

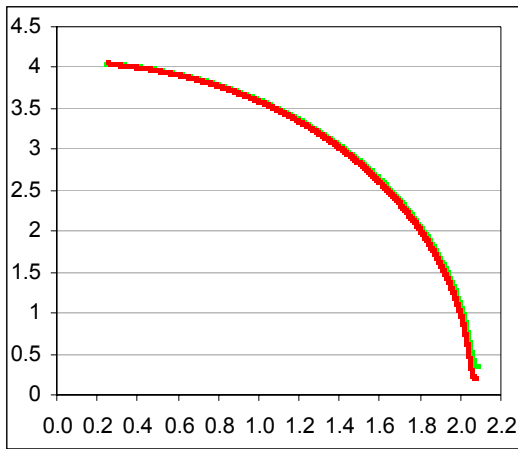
**Figure 140 — 50% Attainment Surfaces**  
 SPEA2: Green, Mak\_SPEA2: Red.  $y$ -axis is objective two;  $x$ -axis is objective one.



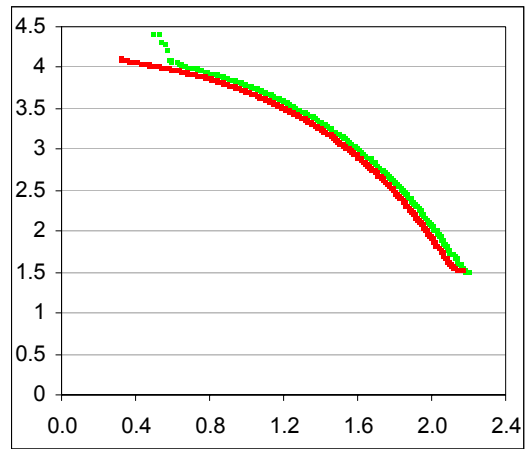
(a) *AP-15*



(b) *AP-16*



(c) *AP-17*



(d) *AP-21*

**Figure 141 — 50% Attainment Surfaces**

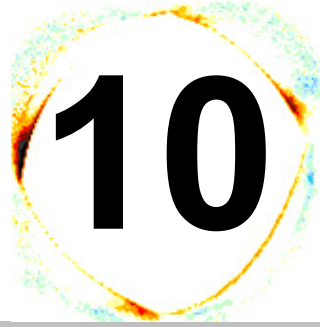
SPEA2: Green, Mak\_SPEA2: Red.  $y$ -axis is objective two;  $x$ -axis is objective one.

### 9.2.5 CONCLUSIONS

Whether it is used as the active population (as for PESA, Section 9.2.2), in a hybrid context (as for SPEA2, Section 9.2.4) or as a purely passive reference set in single- and multi-member systems (PAES, Section 9.2.1, and NSGA-II, Section 9.2.3, respectively), the integration of the Mak\_Tree into contemporary evolutionary algorithms yields impressive results. Indeed, across a diverse problem set and under a thorough statistical analysis, the extended techniques frequently achieve statistically *significant* improvements over their truncated originals. Moreover, the results illustrate a level of robustness rarely found in algorithmic extensions — the Mak\_PESA\_C, Mak\_PESA\_E, Mak\_SPEA2, Mak\_SPEA2\_KNN and Mak\_PAES systems are never *significantly* worse than the bounded originals according to any of the examined metrics, and Mak\_NSGA-II is *significantly* worse only according to a single indicator in *AP-5* (and it is worth recalling that this extension capitalises on the Mak\_Tree in only a purely passive, referential, manner). These powerful results are a clear indication that the use of unbounded archiving is more than just a theoretical curiosity — the damage caused by frontal degradation and inaccurate crowding estimations has a real practical affect on the performance of the examined contemporary algorithms. Therefore, particularly given the efficiency of the specialist Mak\_Tree (see Chapter 6, Chapter 7 and Chapter 8) and its relative simplicity, the replacement of truncated techniques with unbounded approaches in the bi-objective domain is strongly encouraged.

This chapter paints only part of the picture however. If extension of existing bounded systems yields positive results, then the development of novel specialist algorithms designed explicitly for use with unbounded sets may afford even more impressive performance gains. Before this is pursued however, it is useful to first explore more thoroughly the behaviour of existing contemporary algorithms. With such an analysis in hand, novel techniques can be designed to exploit the strengths of those systems and address any weaknesses that may emerge.

# Chapter



# Analysing Contemporary Optimisers

*"It is not only by the questions we have answered that progress may be measured, but also by those we are still asking."*

*Freda Adler — Educator and Writer*

*"Jumping to conclusions is not half as good an exercise as digging for facts."*

*Anonymous*

## 10 ANALYSING CONTEMPORARY OPTIMISERS

It is surprising that the renewed interest in multiobjective test suites (see, for instance, [41, 154, 158, 162, 164, 166, 172, 233, 234]) and performance metrics (see [103, 216, 218, 219, 224, 235-237]) has not led to a similar rebirth of comparative studies. While work exists that capitalises on Pareto compliant performance indicators (see, for instance, [62, 81, 91, 137]), there exists no single study that uses these indicators to analyse a rich set of contemporary optimisers across a diverse collection of real-valued test problems. The likely reason for the dearth of valuable comparative works (like that produced by Zitzler *et al.* [92] during the field's infancy) is related to the impressive growth of multiobjective optimisation research (see Figure 2.1 in Coello *et al.*'s [51] thorough text). Where early studies could provide a comprehensive insight into the field as a whole by examining a small number of algorithms across a narrow set of target domains — the breadth and diversity of contemporary research precludes such a simplistic goal: there is no concise set of algorithms and problems that offers a thorough and complete picture of the current state-of-play. The key then is not to abandon comparative studies, but to reform them under the knowledge that the dynamics of the field have changed. The goal of complete field-wide analysis may be infeasible, but contemporary researchers now have the tools to offer thorough and statistically sound analyses of distinct areas of contemporary multiobjective thought. The production of such studies may offer only a sliver of light into an understanding of the field as a whole, but produce enough slivers and greater truths may be exposed.

With this in mind, the following section will describe the performance of the leading contemporary elitist multiobjective genetic algorithms on a broad set of real-valued, static, unconstrained, bi-objective test problems. Specifically, PAES, PESA, NSGA-II, SPEA2 and IBEA (the Indicator Based Evolutionary Algorithm) are all examined under the nine functions proposed in Section 9.1.2.1 with the Pareto-compliant methods outlined in Section 9.1.1. Note that these algorithms are intended *only* to represent some of the leading choices from the collection of *core* evolutionary algorithms — that is, those techniques that feature a *single* evolutionary process that is perpetuated throughout the length of the run. Their selection does not infer superiority over algorithms produced under the banners of Simulated Annealing [238-241], Ant Colonies [40, 58, 242-244], Swarm Optimisers [89, 149, 174, 245,

246], Artificial Life [169, 247, 248], Memetics [62, 85, 249-251] or Messy Systems [222, 252-254] (amongst others), and indeed, it is hoped that this comparative study will inspire similar works in those distinct areas.

Before commencing the core analysis though, it is first necessary to explore the Indicator Based Evolutionary Algorithm in more detail, since it has received only cursory attention (see Section 3.1.6) thus far.

## **10.1 INDICATOR BASED EVOLUTIONARY ALGORITHMS**

In retrospect, the foundation of the Indicator Based Evolutionary Algorithm (IBEA) [91] is obvious. If Pareto-compliant performance metrics offer insight into the quality of the frontal set (and its constituent members), then a performance indicator can be used *during* the evolutionary process to apply selection pressure. If a Pareto-compliant performance measure is used as the only indicator of preferability, the evolutionary algorithm is charged with a singular task — produce solutions that improve the estimated quality of the prevailing front. This simple notion represents a marked, if not quite paradigmatic, shift in conventional multiobjective thought — rather than trying to explicitly balance the distinct goals of diversity, spread and accuracy, these can be merged via a singular, scalar, performance indicator.

While both impressive and appealingly simple, the IBEA algorithm is not without its flaws. Though the algorithm offers a way to effectively merge the distinct goals of multiobjective optimisation, it is an inherently biased merger — the technique explicitly favours accuracy by manipulating the way the performance measure is used — and requires a scaling factor, which offsets a key advantage afforded by the exclusion of independent diversity approximations (namely, the definition of an extra parameter). Moreover, assessing performance on a per-generation basis can carry a substantial run-time overhead that effectively limits the size of the truncated elite set on which the algorithm operates. Still, the technique represents an exciting new approach to multiobjective optimisation and merits further analysis against its contemporaries.

## **10.2 DESCRIBING IBEA**

The analysis in Section 10.4 explores the performance of two distinct adaptive IBEA systems — one defined by its use of hypervolume (IBEA\_H) and another that

capitalises on the additive epsilon measure (IBEA\_E). It is important to note that the adaptive form of the IBEA algorithm uses scaled objective-scores and indicators, such that all values lie in the range [0,1]. With this in mind, the IBEA system proceeds as outlined in Algorithm 22, with fitness for solution  $\mathbf{a}$  given as:

$$\mathbf{a}_{fitness} = \sum_{\mathbf{b} \in (P-\mathbf{a})} -e^{\left(\frac{-I(\mathbf{a},\mathbf{b})}{v \times c}\right)} \quad (85)$$

where  $v$  is a system parameter,  $c$  is the maximum absolute value of  $I$  for all population members, and  $I$  is a Pareto compliant binary performance indicator. In IBEA\_H,  $I$  represents the volume of objective-space (up to a given reference point) dominated by  $\mathbf{b}$ , but not by  $\mathbf{a}$ , while in IBEA\_E,  $I$  indicates the minimum distance in any dimension that  $\mathbf{a}$  must be shifted to at least weakly dominate  $\mathbf{b}$ .

Note that, structurally at least, the IBEA technique is not markedly different to the SPEA2 and NSGA-II algorithms — all three feature constantly sized truncated archives, binary tournaments between members of the archive and a truncation

#### Algorithm 22 — The IBEA Algorithm

##### Inputs:

$b$	The maximum number of solutions allowed in the breeding pool.
$s$	The maximum archive size.
1: $Arch := \text{generateRandomPop}()$	Initialising the archive with random members.
2: $Parents := \emptyset$	
3: while (terminationConditionMet() $\neq$ true)	
4:   calculateFitness( $Arch$ )	Calculate the fitness of all members according to binary performance indicators.
5:   while ( $ Arch  > s$ )	
6:     extractLeastFit( $Arch$ )	Trim the archive to size by extracting the least fit members of the set.
7:     calculateFitness( $Arch$ )	
8:     while ( $ Parents  < b$ )	Perform binary tournament selection with members of the archive.
9: $first := \text{selectRandom}(Arch)$	
10: $second := \text{selectRandom}(Arch)$	Winners are those with superior, indicator-derived, fitness scores.
11:       if ( $first^{fitness} < second^{fitness}$ )	
12: $Parents := Parents \cup \{first\}$	
13:       else	
14: $Parents := Parents \cup \{second\}$	
15: $Children := \text{reproduce}(Parents)$	Create the next generation of solutions and add this to the archive.
16: $Parents := \emptyset$	
17: $Arch := Arch \cup Children$	

system that is predicated on preferability. The only point of differentiation, though it is an important one, is in IBEA's use of numerical performance indicators to derive solution fitness.

### **10.3 EMPIRICAL ANALYSIS METHODOLOGY**

As with preceding analyses, the performance of the contemporary elite systems is compared with hypervolume and epsilon numerical metrics taken over the course of each run (with twenty distinct runs per optimiser for each indicator). Since this is a multi-system comparative study, in the interests of brevity, pair-wise visual analyses are included only when they offer meaningful insight into the differences of two algorithms. Results for PAES, PESA, NSGA-II and SPEA2 are as per those seen earlier in this chapter (see Section 9.2.1, Section 9.2.2, Section 9.2.3 and Section 9.2.4 respectively), with an equivalent set of IBEA\_H and IBEA\_E runs included to expand the scope of the study. System parameters are as per those defined in preceding sections, with the IBEA techniques featuring  $v = 0.01$  and IBEA\_H using a scaled reference point of (2,2) (which is guaranteed to be greater than the normalised maximum of any function and is the recommendation of Zitzler and Kunzli [91]). It is worth noting that, for consistency, all techniques use an archival threshold of fifty and identical mutation and crossover rates (where appropriate)<sup>76</sup>. All relevant result graphs and tables are presented at the completion of this sub-section (pages 331–345).

### **10.4 EMPIRICAL ANALYSIS**

The explicit aim of many conventional comparative studies is to establish a clear hierarchy of preferability amongst the examined algorithms (see [92], for instance). While this is an honourable pursuit, it is equally important to establish which algorithms are suited to particular problems — armed with such thorough domain-specific analysis, decision makers should be better able to select an approach that matches their current task. Thus, this chapter will explore both domain-specific *and* general performance issues, with a view to better defining the nature of each examined algorithm.

---

<sup>76</sup> Obviously, PAES has no crossover as it is a purely asexual system.



### 10.4.1 PAES PERFORMANCE

The epsilon (Figure 147, Figure 148, Figure 149, Figure 150 and Figure 151) and hypervolume (Figure 142, Figure 143, Figure 144, Figure 145 and Figure 146) results for PAES are an excellent example of how strongly the characteristics of a given test function can affect the apparent quality of an algorithm. The performance of PAES on *AP-15* and *AP-21* is particularly poor, with PAES being *significantly* worse than all algorithms under the hypervolume metric in *AP-21* (Figure 145 and Figure 146), and, according to both metrics, *significantly* worse than at least three of the five other algorithms on *AP-15* and *AP-21* (Figure 145, Figure 146, Figure 150 and Figure 151). The likely reason for such underwhelming performance is related to the non-separability of these functions and the underlying collateral noise (see Section 0) that this non-separability creates. Since PAES, at its core, is a hill-climbing algorithm with a heavy priority assigned to Pareto dominance, there is a tendency for it to have a particularly narrow search focus — recall that the search will only expand along the front if a newly generated solution is in less densely occupied space than the primary (parent) solution. The consequence of such a focussed aggressive approach in non-separable problem domains is that particular genes and dependencies can be over-emphasised in order to generate temporary gains. The problem, of course, is that by implicitly focussing on a sub-set of the solution, the development and improvement of other linkages must suffer. Moreover, the nature of collateral noise makes a latter-day improvement of such neglected alleles unlikely — since gains in other gene areas may degrade the overall utility of the solution (temporarily), the hill-climber is unlikely to orient the search in that direction. Thus, multi-member systems with a predilection towards diversity — such as NSGA-II — are better suited to highly non-separable problems, as they are less likely to pursue a single solution archetype.

The performance of PAES is similarly poor on *AP-16*, where it is *significantly* worse than every tested algorithm on both performance metrics (Figure 145, Figure 146, Figure 150 and Figure 151). This result highlights the inefficiency of single-member hill-climbing algorithms in zero-utility gradient spaces — since the mutation of a single member will rarely result in a distinct improvement, the search will often stall or degenerate into a random walk around these flat regions of space. Multi-member systems such as NSGA-II, SPEA2 and IBEA\_E are better able to avoid such

problems as they typically feature well-distributed leading fronts. This property reduces the likelihood of all members lying in low-gradient spaces and, as such, facilitates continuing evolutionary pressure that can drive the search.

Still, the parent/child hill-climbing system is not without its advantages against the more common multi-member systems. In particular, it is interesting to note that, relative to all other tested algorithms, the PAES technique has *significantly* better end-of-run results on *AP-5* according to both hypervolume (Figure 145 and Figure 146) and epsilon (Figure 150 and Figure 151) metrics, and demonstrates better early-generation performance on both *AP-1* and *AP-2* (Figure 142 and Figure 147). The differences here illustrate a difference in priorities — PAES will always pursue a dominant child and so explicitly biases Pareto dominance over diversity concerns, whereas the multi-member systems are designed to maintain a well distributed front throughout the course of the run. The consequence is that PAES is faster at locating the optimal front in *AP-1* and *AP-2*, but is considerably worse at effectively developing that front. The same is true of *AP-5*, but the biased nature of that problem aids the convergence rates of the PAES system (it offers a useful gradient that the hill-climber can capitalise upon)<sup>77</sup>.

Performance on *AP-4* is also interesting — the simple, base-line, PAES algorithm is better able to address multi-frontality than its more complex, multi-member, successors. The key to this disparity lies with frontal convergence. When a multi-member system converges onto a front, it is typically charged with ensuring an even distribution of members along that front. While this is an appropriate technique if the front is optimal, it can severely hamper performance if it is not. Consider a multi-member population situated along a false front — the set will typically be composed of many well-distributed incomparable proposals. If a solution lying beyond the front is discovered, it will solicit only a small minority of the search focus in subsequent generations — it may be the most promising avenue of escape, but the nature of binary tournaments precludes it from dominating the evolutionary

---

<sup>77</sup> It is important to note that this statement is not intended to reflect the superiority of PAES on biased problems in general. The biased objective-space in *AP-5* is fortuitously arranged in a manner that facilitates a sharp and narrow gradient that can be pursued by PAES without penalty. This aggressive exploration of locally optimal regions at the expense of diversity concerns will be extremely detrimental to the performance of PAES when the objective-space features misleading bias, as evidenced by its particularly poor performance in *AP-17*.

process<sup>78</sup>. In contrast, the hill-climbing PAES algorithm can devote the *entirety* of the search effort towards escaping the front when the opportunity presents itself — so long as the discovered solution is preferable to the parent (as it typically will be), it will become the primary source of subsequent evolutionary exploration. The ability to puncture false fronts in this manner is an interesting feature of hill-climbing systems and, given the difficulty that multi-frontality typically causes multiobjective optimisers, may rest as an interesting avenue of future research.

## 10.4.2 PERFORMANCE OF MULTI-MEMBER SYSTEMS

### 10.4.2.1 DIVERSITY-RELATED PERFORMANCE

A principal difference between the competing evolutionary algorithms is the selected diversity mechanism — with techniques ranging from cuboid,  $\kappa^{\text{th}}$  neighbour and cell-based approaches to the implicit indicator-derived estimations of the IBEA systems. Given such a breadth of available options, the best way of maintaining and developing a rich set of disparate solutions is clearly an important point of contention in the field. One way to address such a debate is to examine the practical performance of each approach in problem domains whose principal complexity is derived from a need for diversity. Consider *AP-1*, *AP-2* and *AP-3* — each of these problems feature simple objective-spaces, bereft of the types of deception, multi-frontality, bias and collateral noise that may inhibit a search. As such, optimisation of these problems does not so much test the ability of an optimiser to locate the front, but rather, its capacity to develop a rich set of solutions along that front. Thus, high end-of-run performance in these algorithms suggests the utility of the corresponding diversity process — whether such basic diversity is valuable in more complex problem spaces is the subject of another debate (as discussed in Section 10.4.1, the maintenance of a diverse set can *inhibit* performance in multi-frontal problems like *AP-4*).

Hypervolume results (Figure 145 and Figure 146) indicate that the IBEA\_E algorithm (and, thus, its epsilon-based diversity estimation) is *significantly* better than the other examined systems on the simple *AP-1*, *AP-2* and *AP-3* problems, while epsilon indicators (Figure 150 and Figure 151) suggest a *significant* improvement in

---

<sup>78</sup> The problem is exacerbated the larger the population is, since this will further dilute the impact of independent members.

*AP-2* and minor advantages in *AP-1* and *AP-3*. Equally importantly, IBEA\_E is never *significantly* worse than any other tested system on these functions, irrespective of the selected performance indicator. Such results suggest a clear differential between the implicit diversity maintenance techniques employed by IBEA\_E and the approaches utilised by the other tested systems. Similarly crisp orderings do not exist in the remaining techniques, though the results are no less meaningful.

Consider the performance of IBEA\_H (Figure 145, Figure 146, Figure 150 and Figure 151): it is particularly impressive in *AP-1* and *AP-3*, *significantly* bettering all other non-IBEA systems under the hypervolume metric. Such strong results are supported by performance in *AP-3* under the epsilon metric, where it is again *significantly* better than all non-IBEA systems, and in the epsilon analysis of *AP-1*, where it is only *significantly* outperformed by NSGA-II. However, its performance in the *AP-2* function is extremely poor — with *significantly* better results achieved by *every other* tested system according to both metrics. This lends support to the notion posited by Zitzler and Thiele [137] that hypervolume estimations may bias convex regions of objective-space — the concave optimal front in *AP-2* clearly affects the capacity of the IBEA\_H optimiser to distribute the search effectively.

With respect to the remaining multi-member techniques, the NSGA-II cuboid approach appears preferable to the  $\kappa^{\text{th}}$  neighbour and cell-based strategies, with NSGA-II achieving *significantly* better results than both in *AP-1* and *AP-3* under hypervolume metrics (Figure 145 and Figure 146), and in *AP-3* according to epsilon indicators (Figure 150 and Figure 151). Moreover, the NSGA-II approach is never *significantly* worse than either technique under any metric for these three basic problems. The results are interesting — even though SPEA2 uses a relatively small value for  $\kappa$  (very near to the suggestion made by Zitzler *et al.* [81]<sup>79</sup>), its performance indicates that even a minor bias away from locally uncrowded solutions and towards a more expansive analysis can have a detrimental affect on maintaining a well-distributed search. This is a somewhat unintuitive claim — the natural assumption is that the richer the density analysis, the better the estimations are likely to be (up to a point). The reason for such unusual conclusions is again attributable to the dual

---

<sup>79</sup> Following these guidelines a  $\kappa$  value of 10, rather than 11, is recommended. It is *very* unlikely that such a small difference would significantly affect the performance of the SPEA2 algorithm.

nature of the truncated set — the goal of the archive is to maintain uncrowded members *and* offer a reasonably well-distributed approximation of the front (lest the accuracy of dominance comparisons becomes severely eroded). However, the  $\kappa^{\text{th}}$  neighbour approach is inherently biased towards members lying at the extremes of objective-space (the larger the value of  $\kappa$ , the greater the bias) and, as such, the density-based truncation operation can lead to discontinuities in the centre of the archival front and an over-emphasis of exterior solutions — clearly this will affect how evenly distributed the archival set is. Perhaps more importantly, the  $\kappa^{\text{th}}$  neighbour technique estimates how crowded the *region* is in which a solution resides and can therefore recommend the extraction of locally uncrowded members during truncation. The  $\kappa^{\text{th}}$  neighbour estimations may therefore give a more accurate picture of the front, but are detrimental to the archive’s role as a dominance comparison set.

Thus, the superiority of the cuboid approach over the SPEA2 technique on the simple *AP-1*, *AP-2* and *AP-3* test functions implies that, of the dual tasks that a truncated archive is charged with, the maintenance of a well-distributed archival set is of primary importance. By using diversity estimation techniques that more readily allow poorly distributed archives, the SPEA2 algorithm is more susceptible to frontal degradation and the acceptance of weak solutions — the consequence of which is less evolutionary drive and clearly inferior results (as emphasised by both hypervolume and epsilon indicators).

Similar problems beset the PESA approach. Since PESA-based truncation removes a random member of the most occupied cell, there is the potential for shrinking fronts (since the extreme members of the prevailing front are not protected from deletion) and the introduction of frontal-discontinuities (particularly if solutions lying on the boundaries of a cell are removed). Problems also exist if the grid is too evenly distributed or fine grained<sup>80</sup> (where a large number of singly occupied cells can reduce much of the truncation process to random extractions), or if the grid is too coarse (where a small number of heavily occupied cells can incur similarly unpredictable deletions). Examining the average post-truncation cell sizes (Figure 156) of the PESA system, it is also clear that the crowding mechanism (under the selected grid resolution) provides little information to the binary tournament in these

---

<sup>80</sup> This runs counter to the claim of Khare that “very fine grids... [produce] good performance in terms of diversity” [75].

tests — reducing many of the matches to random selections. Thus, there is potential for the truncated archive to suffer in terms of both distribution and crowding estimation. Such deficiencies clearly underline the problems inherent in grid-based crowding — unless the cells are correctly sized, the procedure can collapse.

#### 10.4.2.2 **PERFORMANCE AGAINST COMPLEX PROBLEM SPACES**

Performance on *AP-15*, *AP-16*, *AP-17* and *AP-21*, in particular, illustrates a clear differential between the PESA algorithm and the other tested multi-member systems in complex problem domains. Indeed, PESA is *significantly* worse than *all* other examined population-based algorithms on *AP-15*, *AP-17* and *AP-21* under epsilon metrics (Figure 150 and Figure 151), and *significantly* worse on *AP-15*, *AP-16* and *AP-17* according to hypervolume metrics (Figure 145 and Figure 146). Moreover, it is never *significantly* better according to any metric when examining the complex *AP-15*, *AP-16*, *AP-17* and *AP-21* problems. The poor performance of the PESA algorithm is likely attributable to the inadequacy of the selected grid mechanism (see Section 10.4.2.1) — the truncated set in each case is primarily composed of solutions sharing an identical crowding score. Clearly, this removes the impetus for distributing the search effectively, with the evolutionary process operating on little more than a random sub-set of the (at best) leading front.

PESA fares better in the multi-modal *AP-4* function. Indeed, the PESA algorithm is never *significantly* worse than any other system on either metric, and is *significantly* better than IBEA\_H according to both metrics. This somewhat surprising result — in league with the poor performance of IBEA\_E in this domain — lends further weight to the notion that maintaining and exploring a well-distributed frontal set in multi-modal problem domains can have an adverse affect on overall performance (as outlined in Section 10.4.1). Indeed, it appears that the weaknesses of the grid-based system (namely, the potential for shrinking fronts, discontinuities and randomised selections) introduces sufficient noise to encourage exploration away from each misleading front and thus inhibits premature convergence.

Outside of the already discussed *AP-4* function, the performance of IBEA\_E on the complex test problems is particularly impressive (see Figure 145, Figure 146, Figure 150 and Figure 151, in particular). It is never *significantly* worse under either the hypervolume or epsilon metrics. Moreover, it is *significantly* better than all multi-

member systems on *AP-5* for both metrics; *significantly* outperforms SPEA2 according to at least one metric on all problems; is *significantly* superior over NSGA-II on at least one indicator in *AP-5*, *AP-15* and *AP-21*; and is *significantly* better than PESA on every function other than *AP-4* with respect to both epsilon and hypervolume metrics. Thus, together with the results illustrated in Section 10.4.2.1, the IBEA\_E approach is the dominant technique with respect to the majority of functions addressed in this study. This further solidifies the notion that the integration of performance indicators into multiobjective optimisers can yield powerful results.

However, the relatively disappointing performance of IBEA\_H demonstrates that the inclusion of an indicator alone does not guarantee unmitigated success — indeed, the choice of indicator is fundamentally important. Empirical results suggest that the hypervolume indicator used in IBEA\_H fails to adequately encourage a well-spread search (as most obviously evidenced in the *AP-15* frequency matrix illustrated in Figure 155), the consequence of which is a system whose end-of-run performance is *significantly* worse (Figure 145, Figure 146, Figure 150 and Figure 151) than IBEA\_E on every tested function bar *AP-21* (according to at least one metric). Moreover, it suffers against the cuboid and neighbourhood based approaches of NSGA-II and SPEA2 — on at least one metric, it is *significantly* worse than either or both of these algorithms on six of the nine tested functions. Given the apparent sensitivity of the IBEA approach to the underlying indicator, this thesis recommends that future works explore a richer set of Pareto-compliant techniques (such as *R* indicators [186]) with a view to establishing differences in performance characteristics.

Of the popular NSGA-II and SPEA2 approaches, results suggest that NSGA-II is superior across a broad range of complex problem domains, with *significant* improvements seen in *AP-5*, *AP-15* and *AP-16* under the hypervolume metric and similarly statistically important gains made in *AP-5* and *AP-21* on the epsilon metric (Figure 145, Figure 146, Figure 150 and Figure 151). Moreover, NSGA-II is never *significantly* worse than the SPEA2 approach on any tested function. Such impressive performance relative to SPEA2 on problem domains with challenging objective-spaces illustrates the superiority of the NSGA-II system in guiding and

distributing the search and infers the preferability of the cuboid density estimation technique and the frontality-based fitness function.

#### **10.4.2.3      *GENERAL PERFORMANCE***

While the No Free Lunch [161] theorem intimates, and the implicit biases of each technique suggest, that there can be no single optimal approach to bi-objective optimisation and it is therefore impossible to establish any kind of true general hierarchy of the available algorithms, it is possible to draw preliminary conclusions about the suitability of algorithms to particular classes of problem domains (as seen throughout this section). Should a system perform well across a set of domains that are consistent with the type of problems encountered in the real world, then it is reasonable to assume that the approach rests as a good choice for those decision makers (in particular) with limited intimate *a priori* knowledge of the problem at hand.

Of the tested algorithms, IBEA\_E presents the most consistently impressive performance — effectively addressing varying frontal shapes, non-separability, deception and zero-utility gradient spaces. Though PAES offers preferable results in both *AP-4* and *AP-5*, that algorithm's inability to perform adequately (under the settings used here) on the complex later test functions, and its relatively poor performance in the simple *AP-1*, *AP-2* and *AP-3* problems, effectively precludes PAES as a viable general-use choice. Similarly, though NSGA-II significantly improves on the performance of IBEA\_E on *AP-4*, IBEA\_E is significantly better (according to at least one metric) in five of the nine examined functions. Thus, given a general continuous bi-objective problem, this thesis offers a preliminary recommendation of the IBEA\_E system as the best of the tested *truncated* approaches. Future testing, particularly focussing on the variation of PAES, PESA and IBEA settings, should further explore this claim and endeavour to provide useful heuristics for parameter settings.

Given the broad range of core evolutionary algorithms examined and the diverse set of test problems explored, it is also possible to begin appraising which domain features best inhibit the performance of multiobjective optimisers. Most notable is the difficulty associated with multi-frontality — with attainment surfaces (Figure 108, Figure 109, Figure 118, Figure 119, Figure 128, Figure 129, Figure 140, Figure



141, Figure 152 and Figure 153) and frequency matrices (Figure 116, Figure 117, Figure 126, Figure 127, Figure 138, Figure 139, Figure 154 and Figure 155) indicating an inability to consistently escape sub-optimal fronts for all algorithms (particularly in *AP-4* and, to a lesser extent, *AP-21*). Obviously the number and intensity of false fronts will affect the extent to which optimisers are damaged by multi-frontality, but the likes of *AP-4* — with its frequent and well-defined fronts — will likely pose strong obstacles to efficiency (and the applicability of termination criteria). Of the remaining domain characteristics, high levels of non-separability strongly affect all of the examined systems (the median attainment surfaces in *AP-15* and *AP-21* are well-removed from the true front — see Figure 153), while zero-utility gradients can be problematic (see *AP-16* in Figure 153), particularly for hill-climbing systems, which require a well-defined fitness slope to drive the evolutionary process effectively. Finally, frontal shape remains a problem for some systems (concavity affects *IBEA\_H*, in particular), but is otherwise a minor issue, while misleading bias appears to most affect systems that fail to maintain a sufficiently diverse population set (which better equips them to escape locally promising, but globally inferior, regions). Thus, this thesis recommends that a greater focus be applied to research in the areas of multi-frontality and, to a lesser extent, non-separability and flat fitness spaces — improving performance of algorithms in these important areas is a key to solidifying the successes of multiobjective optimisation in real-world tasks.

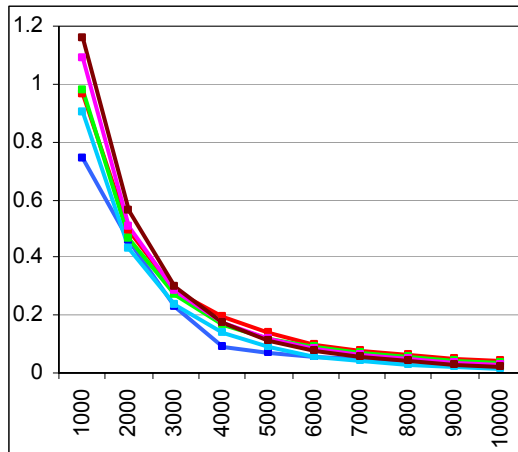
## 10.5 CONCLUSIONS

This chapter has addressed the lack of thorough comparative studies in the area of contemporary core evolutionary algorithms by offering a rich analysis of NSGA-II, SPEA2, IBEA, PESA and PAES across a diverse problem set. By examining performance under hypervolume and epsilon Pareto-compliant indicators and by exploring the statistical differences between results, the behaviour of each algorithm is placed in an unbiased contemporary context. Importantly, the results speak not only to the general behaviour of each approach, but elucidate the influence of domain characteristics and explore the contributing factors that lead to both the successes and failures of each technique. It is such information that will provide the base for the development of powerful new unbounded systems in the following chapter. A summary of the key findings is offered in Table 34, with the breadth of observations

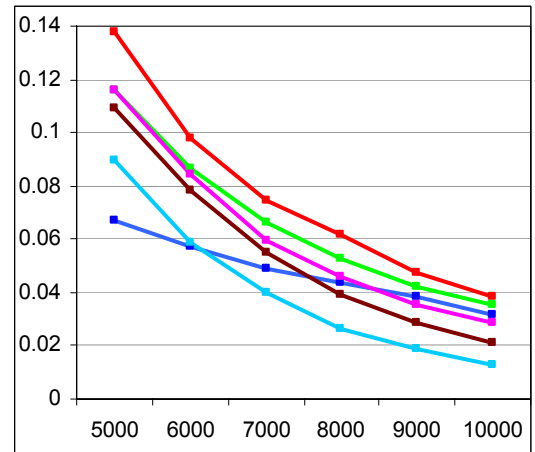
serving as an illustration of the importance of comparative studies and as a recommendation for continued work in the area.

**Table 34 - Summarising Key Findings in the Comparative Study of Contemporary Evolutionary Algorithms**

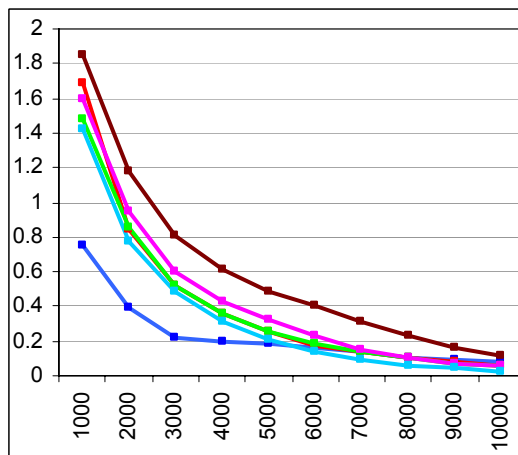
Parameter Selection Issues
Resolution of PAES and PESA grid is a key factor in performance. When cells typically contain only a single member, selection from, and truncation of, the archive can tend towards randomness.
The selection of the underlying indicator for IBEA-based optimisation is important. Empirical results suggest that the epsilon indicator is markedly better, in general, than the hypervolume indicator.
Domain-Related Issues
Multi-frontality causes problems for all optimisation techniques, particularly in <i>AP-4</i> . Noise in the optimisation process may be the key to avoiding premature convergence and encouraging exploration away from the front. The types of penetration offered by single-member, PAES-like, systems may also be important.
Zero-utility gradient spaces are particularly problematic for single-member hill-climbing algorithms, since there is little contextual information with which to orient the search. The lack of additional members also inhibits escape from such regions.
Concave optimal fronts affect the performance of the hypervolume-based IBEA_H approach. This is due to an implicit bias in the approach that favours convex objective-spaces.
Non-separability causes difficulties for multiobjective optimisers in general, and particularly for the single-member PAES system. Diversity appears to be central to improving performance, as dominance-centric approaches are prone to over-emphasising particular gene dependencies at the expense of others.
Crowding Estimation Issues
Results suggest that the cuboid diversity measure is typically preferable to the $\kappa^{\text{th}}$ neighbour approach. The cuboid technique is more able to maintain an evenly distributed archival set, which better prevents both the inclusion of weak solutions and, in-turn, frontal degradation.
The use of cell-based diversity estimates in PESA can be extremely detrimental to the quality of the archival set — truncation affords no protection of extreme members and may promote frontal shrinkage, while large frontal discontinuities can be introduced if the set is evenly distributed or built on inadequately sized cells.
Overall Performance
IBEA_E achieves the best performance in general, despite inefficiencies in <i>AP-4</i> and <i>AP-5</i> .
The PAES system is competitive in simple objective-spaces and multi-frontal domains, but suffers considerably under the introduction of non-separability and zero-utility gradient spaces (in particular).
Of the popular NSGA-II and SPEA2 systems, NSGA-II is generally superior.



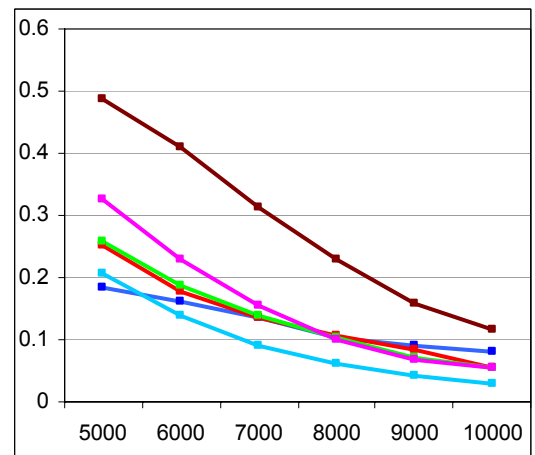
(a) AP-1



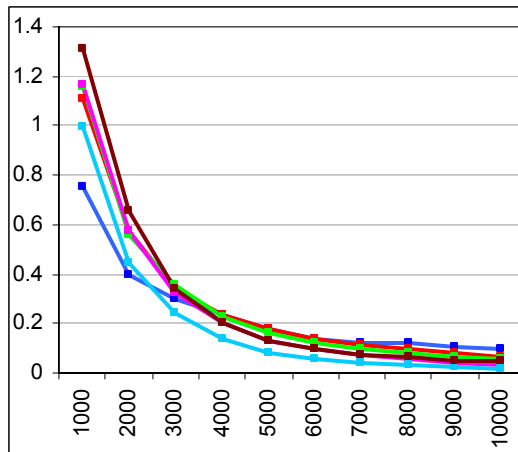
(b) AP-1



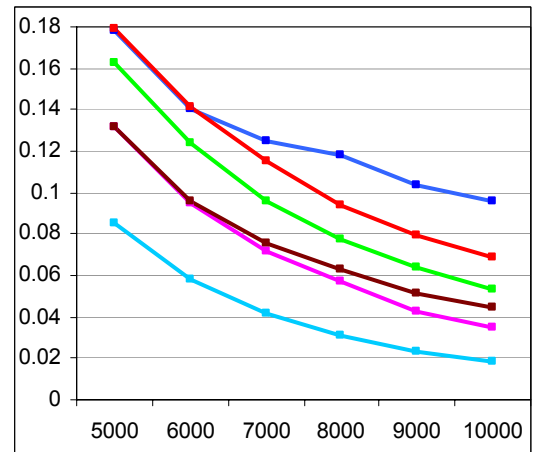
(c) AP-2



(d) AP-2



(e) AP-3

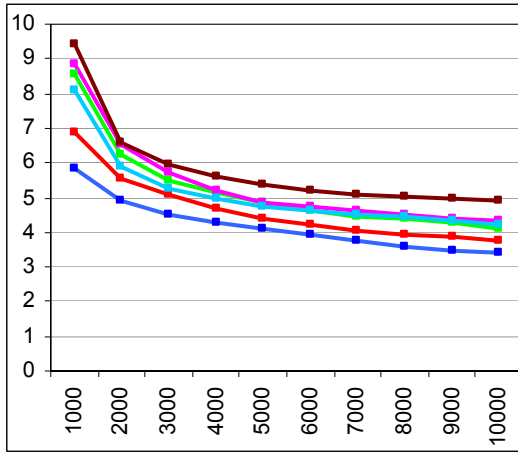


(f) AP-3

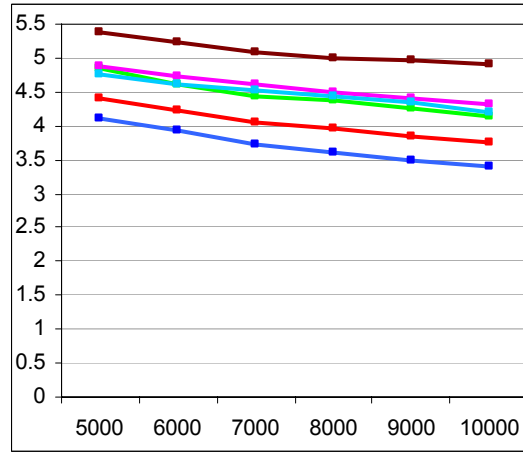


Figure 142 — Progressive Hypervolume Averages

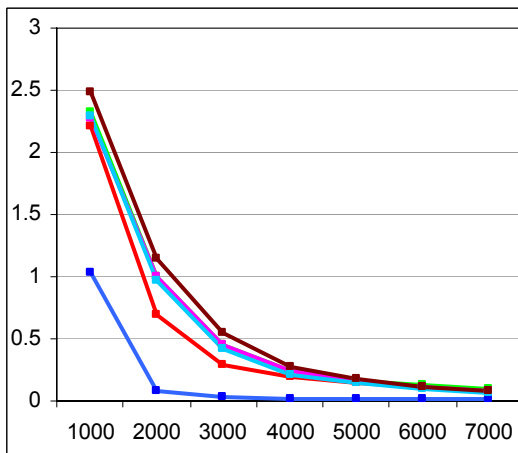
*y-axis*: average hypervolume performance; *x-axis*: number of evaluations executed.



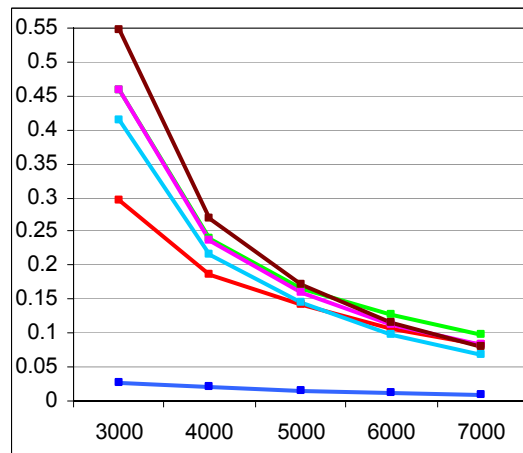
(a) AP-4



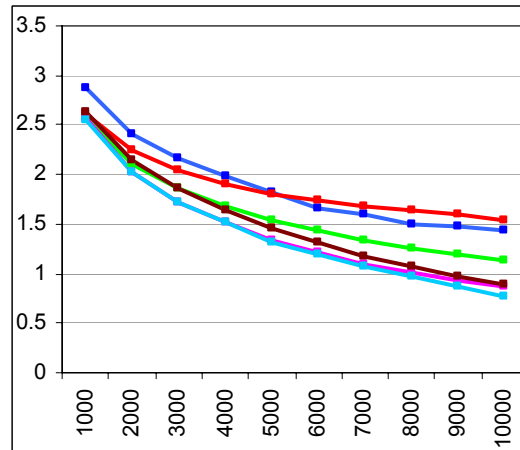
(b) AP-4



(c) AP-5



(d) AP-5

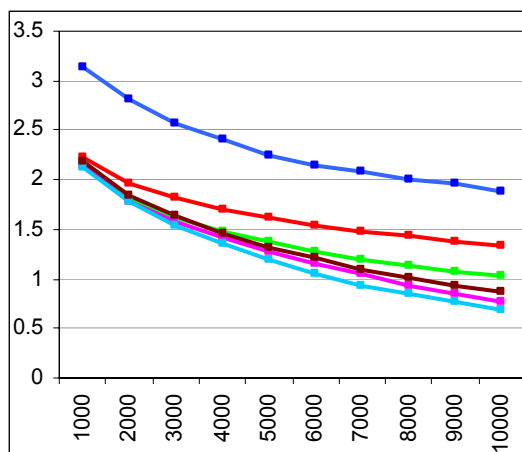
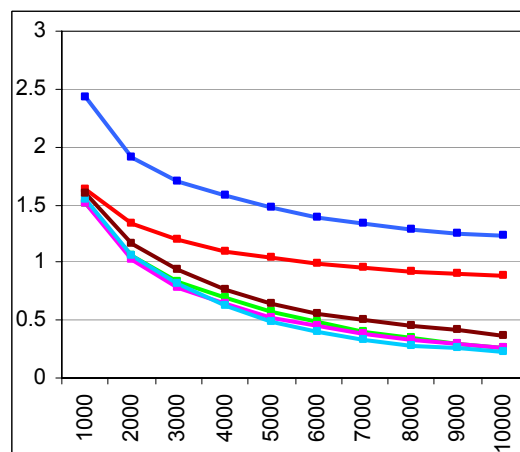
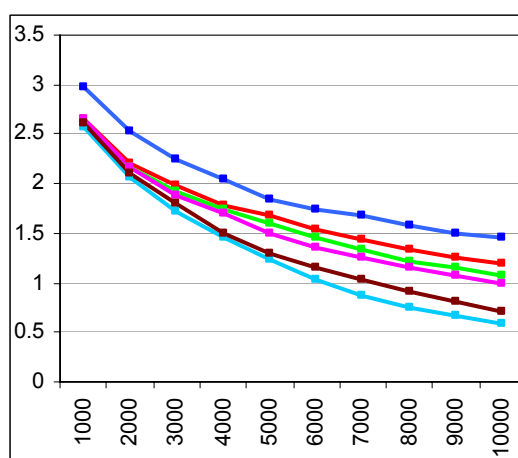
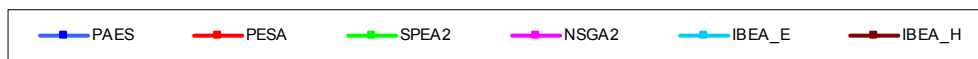


(f) AP-15

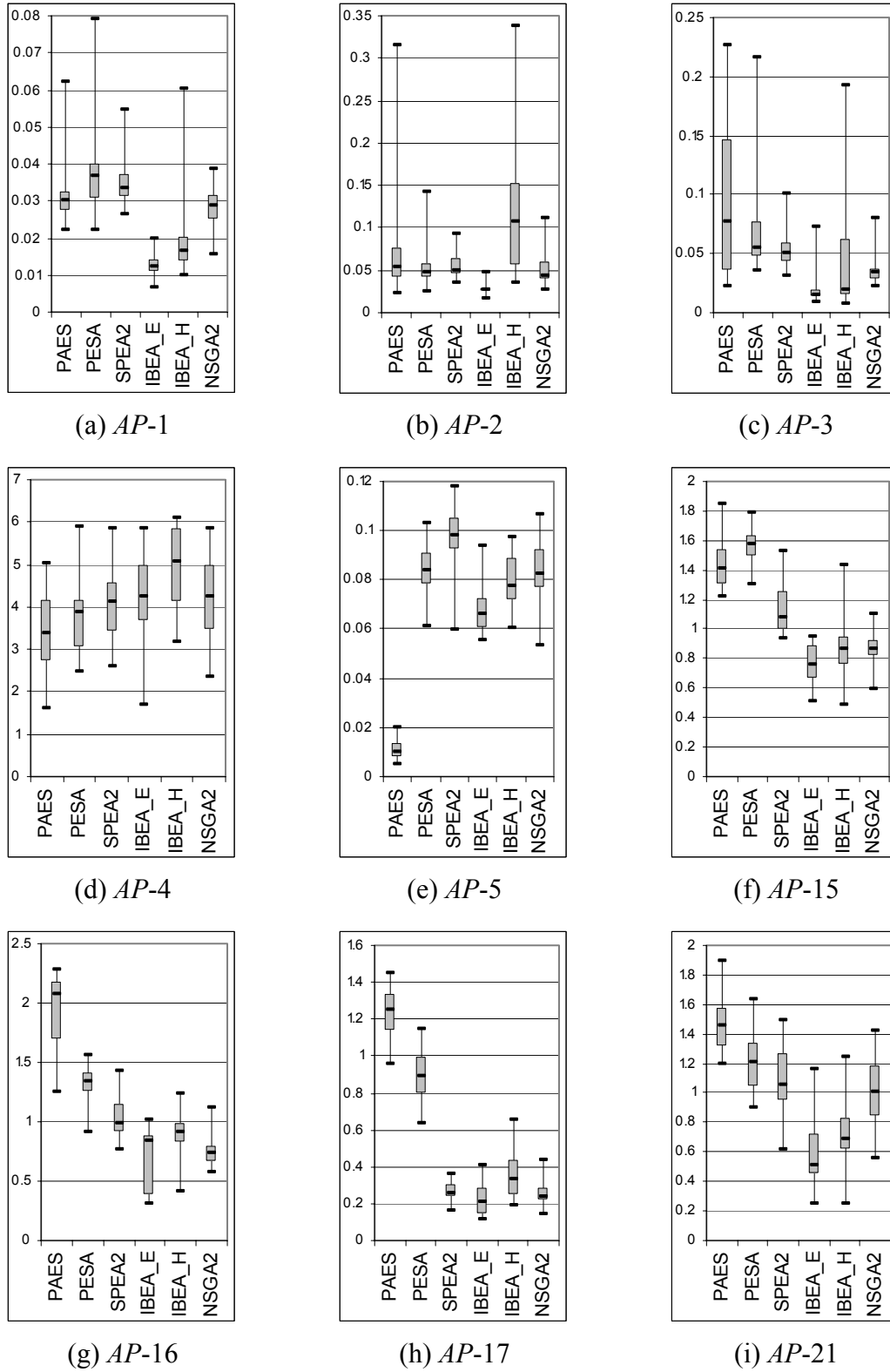


**Figure 143 — Progressive Hypervolume Averages**

*y-axis*: average hypervolume performance; *x-axis*: number of evaluations executed.

(a) *AP-16*(b) *AP-17*(c) *AP-21***Figure 144 — Progressive Hypervolume Averages**

*y-axis*: average hypervolume performance; *x-axis*: number of evaluations executed.

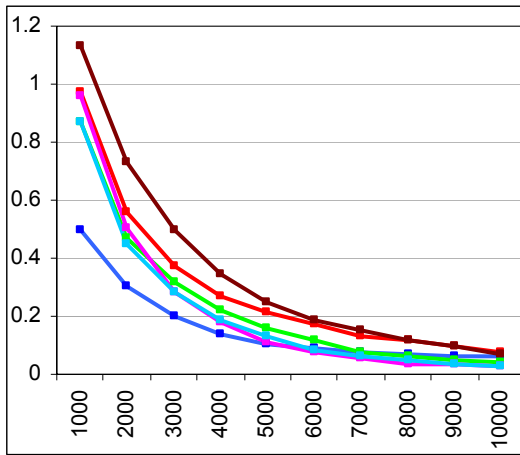


**Figure 145 — End-of-Run Hypervolume Box-Plots**

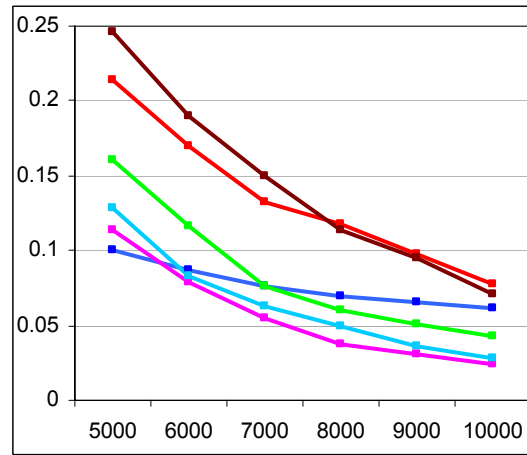
*y*-axis is the hypervolume performance at the end of the run (7,000 evaluations in *AP-5*, 10,000 evaluations in all remaining functions); *x*-axis indicates the selected optimiser.

	PAES	PESA	SPEA2	NSGA-II	IBEA_E	IBEA_H	(a) AP-1
PAES	-	<b><i>2.189E-02</i></b>	<b><i>2.604E-02</i></b>	2.451E-01	<b><i>3.935E-19</i></b>	<b><i>1.135E-10</i></b>	
PESA	2.876E-01	-	9.461E-01	<b><i>6.018E-04</i></b>	<b><i>7.668E-27</i></b>	<b><i>3.248E-17</i></b>	
SPEA2	8.878E-01	2.286E-01	-	<b><i>7.647E-04</i></b>	<b><i>1.317E-26</i></b>	<b><i>5.242E-17</i></b>	
NSGA-II	1.537E-01	7.154E-01	1.171E-01	-	<b><i>1.537E-15</i></b>	<b><i>7.180E-08</i></b>	
IBEA_E	<b><i>3.707E-11</i></b>	<b><i>1.593E-08</i></b>	<b><i>1.574E-11</i></b>	<b><i>1.071E-07</i></b>	-	<b><i>3.822E-03</i></b>	
IBEA_H	<b><i>2.706E-02</i></b>	<b><i>1.137E-03</i></b>	<b><i>3.834E-02</i></b>	<b><i>3.090E-04</i></b>	<b><i>1.542E-17</i></b>	-	
(b) AP-2							
	PAES	PESA	SPEA2	NSGA-II	IBEA_E	IBEA_H	(c) AP-3
PAES	-	3.673E-01	9.765E-01	<b><i>6.809E-05</i></b>	<b><i>2.043E-22</i></b>	<b><i>1.858E-10</i></b>	
PESA	2.854E-01	-	3.519E-01	<b><i>1.299E-06</i></b>	<b><i>1.782E-25</i></b>	<b><i>7.371E-13</i></b>	
SPEA2	<b><i>2.167E-02</i></b>	2.166E-01	-	<b><i>7.667E-05</i></b>	<b><i>2.561E-22</i></b>	<b><i>2.208E-10</i></b>	
NSGA-II	<b><i>7.838E-03</i></b>	1.090E-01	7.123E-01	-	<b><i>2.285E-10</i></b>	<b><i>1.067E-02</i></b>	
IBEA_E	<b><i>1.352E-02</i></b>	1.582E-01	8.602E-01	8.473E-01	-	<b><i>7.850E-05</i></b>	
IBEA_H	<b><i>9.024E-06</i></b>	<b><i>6.433E-04</i></b>	<b><i>2.773E-02</i></b>	6.631E-02	<b><i>4.266E-02</i></b>	-	
(d) AP-4							
	PAES	PESA	SPEA2	NSGA-II	IBEA_E	IBEA_H	(e) AP-5
PAES	-	<b><i>1.037E-30</i></b>	<b><i>1.551E-44</i></b>	<b><i>2.329E-29</i></b>	<b><i>2.995E-12</i></b>	<b><i>3.021E-24</i></b>	
PESA		-	<b><i>1.807E-04</i></b>	7.039E-01	<b><i>2.664E-08</i></b>	6.503E-02	
SPEA2			-	<b><i>3.963E-05</i></b>	<b><i>7.975E-19</i></b>	<b><i>3.953E-08</i></b>	
NSGA-II				-	<b><i>1.889E-07</i></b>	1.422E-01	
IBEA_E					-	<b><i>1.352E-04</i></b>	
	PAES	PESA	SPEA2	NSGA-II	IBEA_E	IBEA_H	(f) AP-15
PAES	-	3.077E-01	<b><i>2.055E-04</i></b>	<b><i>9.495E-19</i></b>	<b><i>8.977E-26</i></b>	<b><i>2.227E-17</i></b>	
PESA	<b><i>8.846E-03</i></b>	-	<b><i>2.777E-06</i></b>	<b><i>4.581E-22</i></b>	<b><i>2.404E-29</i></b>	<b><i>1.263E-20</i></b>	
SPEA2	<b><i>5.122E-12</i></b>	<b><i>7.223E-06</i></b>	-	<b><i>2.530E-08</i></b>	<b><i>8.562E-14</i></b>	<b><i>2.397E-07</i></b>	
NSGA-II	<b><i>7.784E-28</i></b>	<b><i>5.538E-19</i></b>	<b><i>1.029E-06</i></b>	-	<b><i>3.514E-02</i></b>	6.617E-01	
IBEA_E	<b><i>2.315E-31</i></b>	<b><i>3.189E-22</i></b>	<b><i>6.305E-09</i></b>	3.204E-01	-	<b><i>1.114E-02</i></b>	
IBEA_H	<b><i>9.754E-21</i></b>	<b><i>9.563E-13</i></b>	<b><i>3.983E-03</i></b>	<b><i>3.736E-02</i></b>	<b><i>2.221E-03</i></b>	-	
(g) AP-16							
	PAES	PESA	SPEA2	NSGA-II	IBEA_E	IBEA_H	(h) AP-17
PAES	-	<b><i>3.263E-02</i></b>	<b><i>2.250E-25</i></b>	<b><i>6.196E-29</i></b>	<b><i>1.460E-34</i></b>	<b><i>1.241E-16</i></b>	
PESA	<b><i>7.472E-03</i></b>	-	<b><i>2.746E-18</i></b>	<b><i>1.400E-21</i></b>	<b><i>6.465E-27</i></b>	<b><i>1.373E-10</i></b>	
SPEA2	<b><i>2.341E-05</i></b>	1.093E-01	-	3.079E-01	<b><i>1.001E-02</i></b>	<b><i>7.745E-03</i></b>	
NSGA-II	<b><i>7.620E-08</i></b>	<b><i>5.049E-03</i></b>	2.237E-01	-	1.172E-01	<b><i>2.552E-04</i></b>	
IBEA_E	<b><i>3.190E-23</i></b>	<b><i>1.039E-14</i></b>	<b><i>2.529E-10</i></b>	<b><i>1.897E-07</i></b>	-	<b><i>2.632E-07</i></b>	
IBEA_H	<b><i>1.624E-18</i></b>	<b><i>9.622E-11</i></b>	<b><i>5.689E-07</i></b>	<b><i>1.207E-04</i></b>	1.504E-01	-	
(i) AP-21							

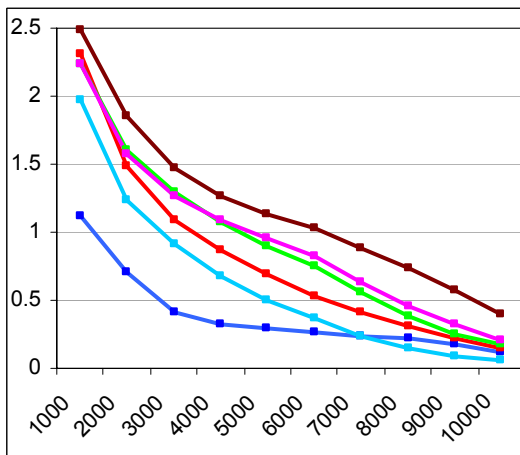
Figure 146 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Hypervolume Results  
 Bold italics indicate significant differences at the 5% level.



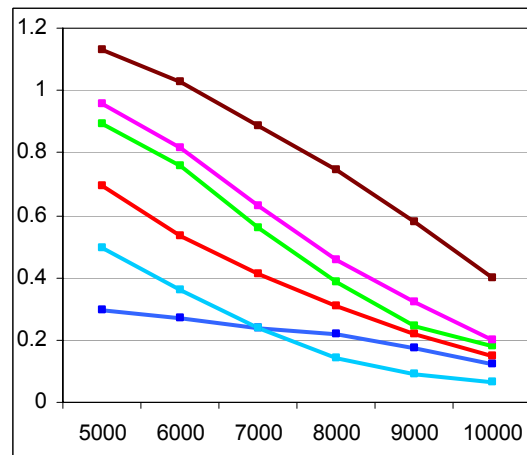
(a) *AP-1*



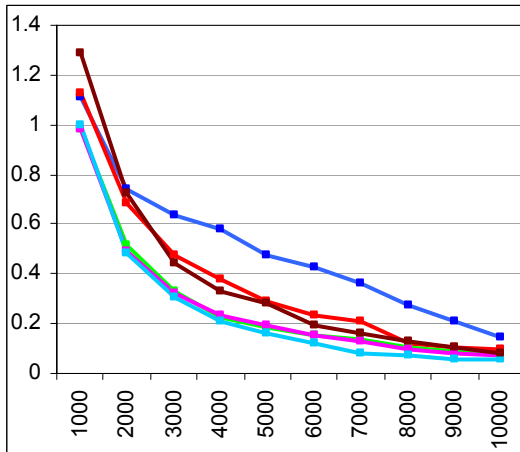
(b) *AP-1*



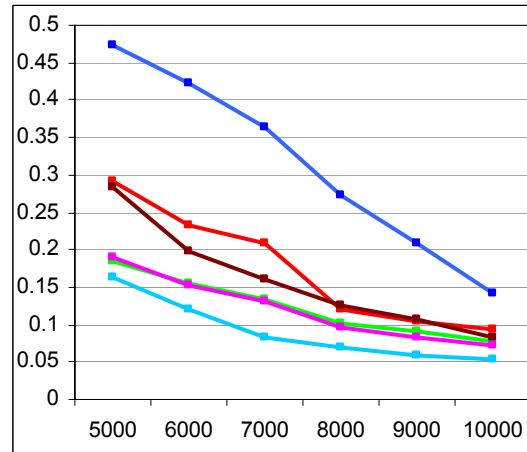
(c) *AP-2*



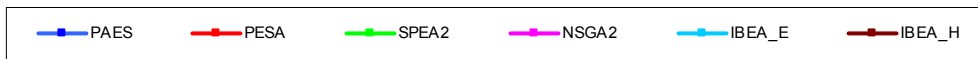
(d) *AP-2*



(e) *AP-3*



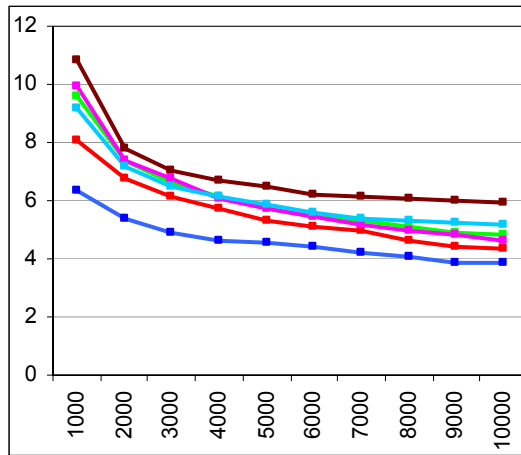
(f) *AP-3*



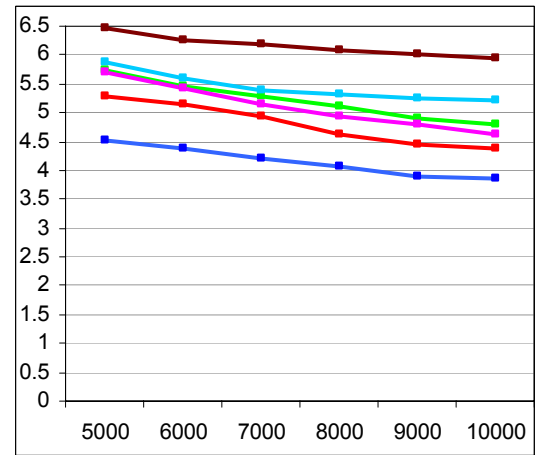
**Figure 147 — Progressive Epsilon Averages**

*y-axis*: average epsilon performance; *x-axis*: number of evaluations executed.

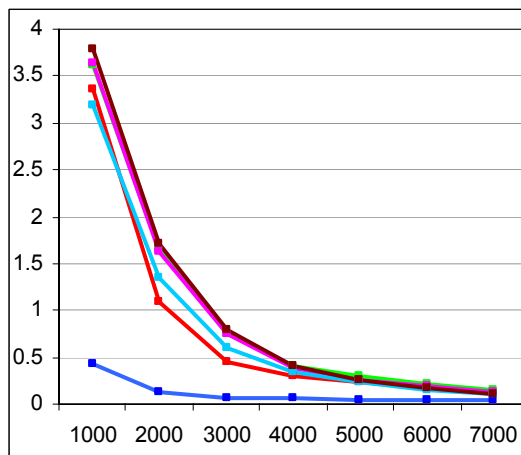




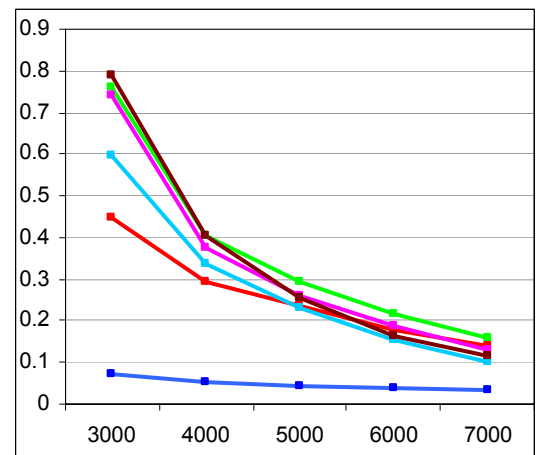
(a) AP-4



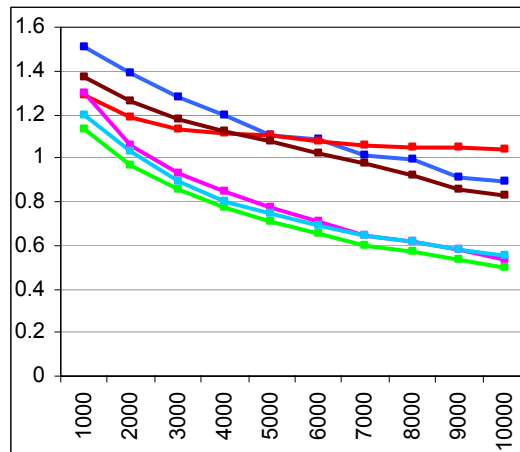
(b) AP-4



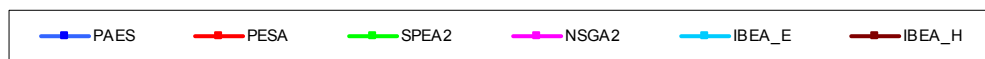
(c) AP-5



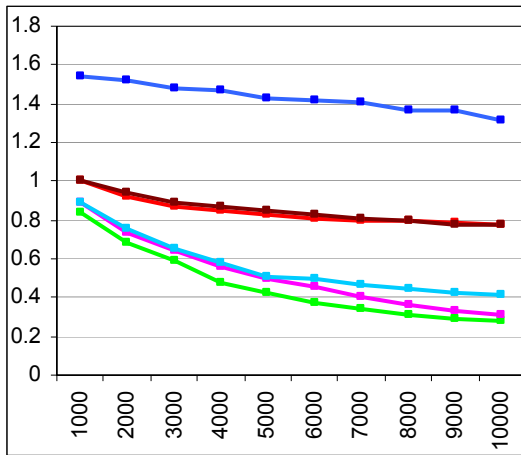
(d) AP-5



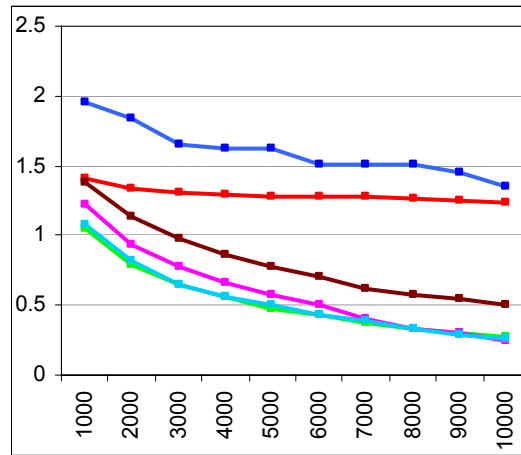
(f) AP-15



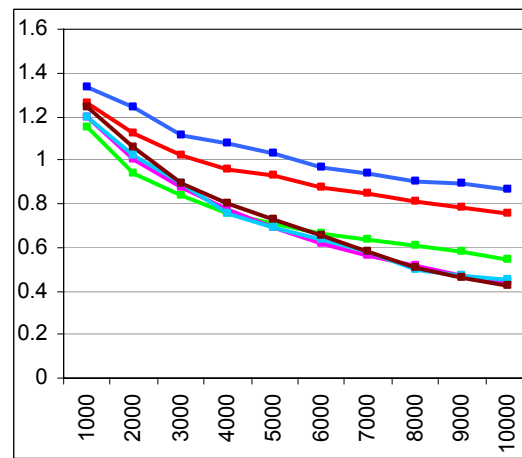
**Figure 148 — Progressive Epsilon Averages**  
*y-axis:* average epsilon performance; *x-axis:* number of evaluations executed.



(a) *AP-16*



(b) *AP-17*

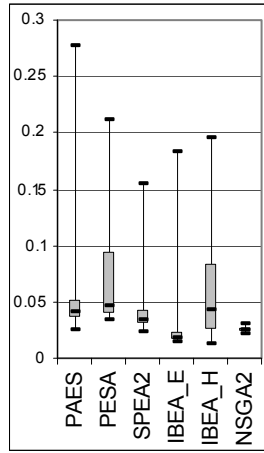
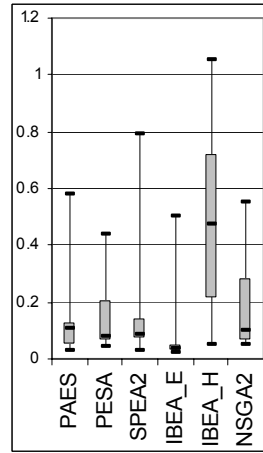
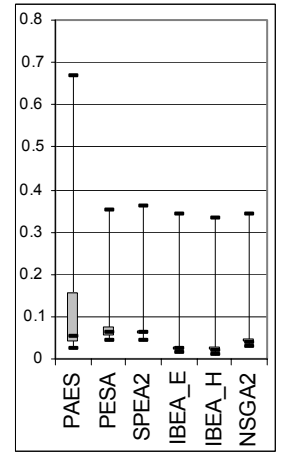
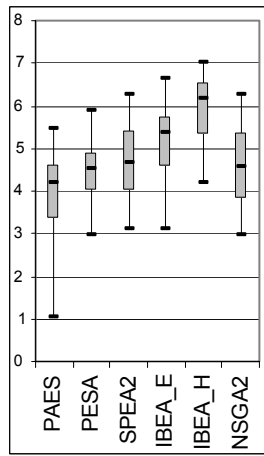
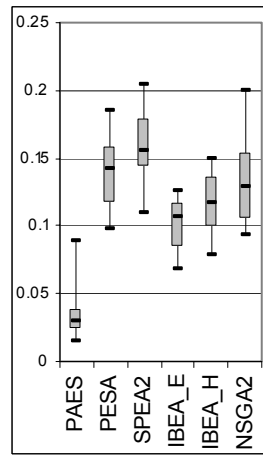
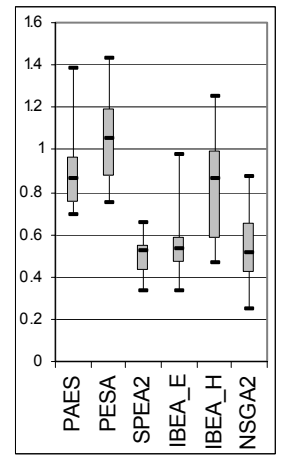
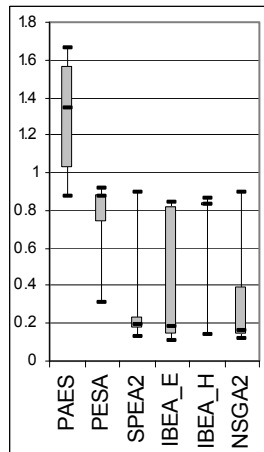
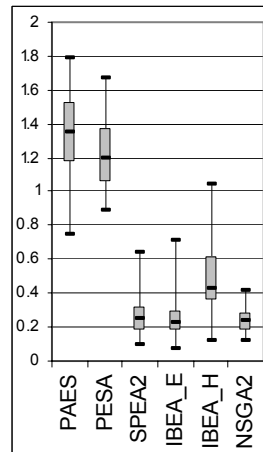
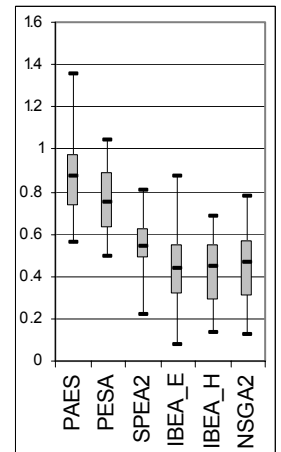


(c) *AP-21*



**Figure 149 — Progressive Epsilon Averages**

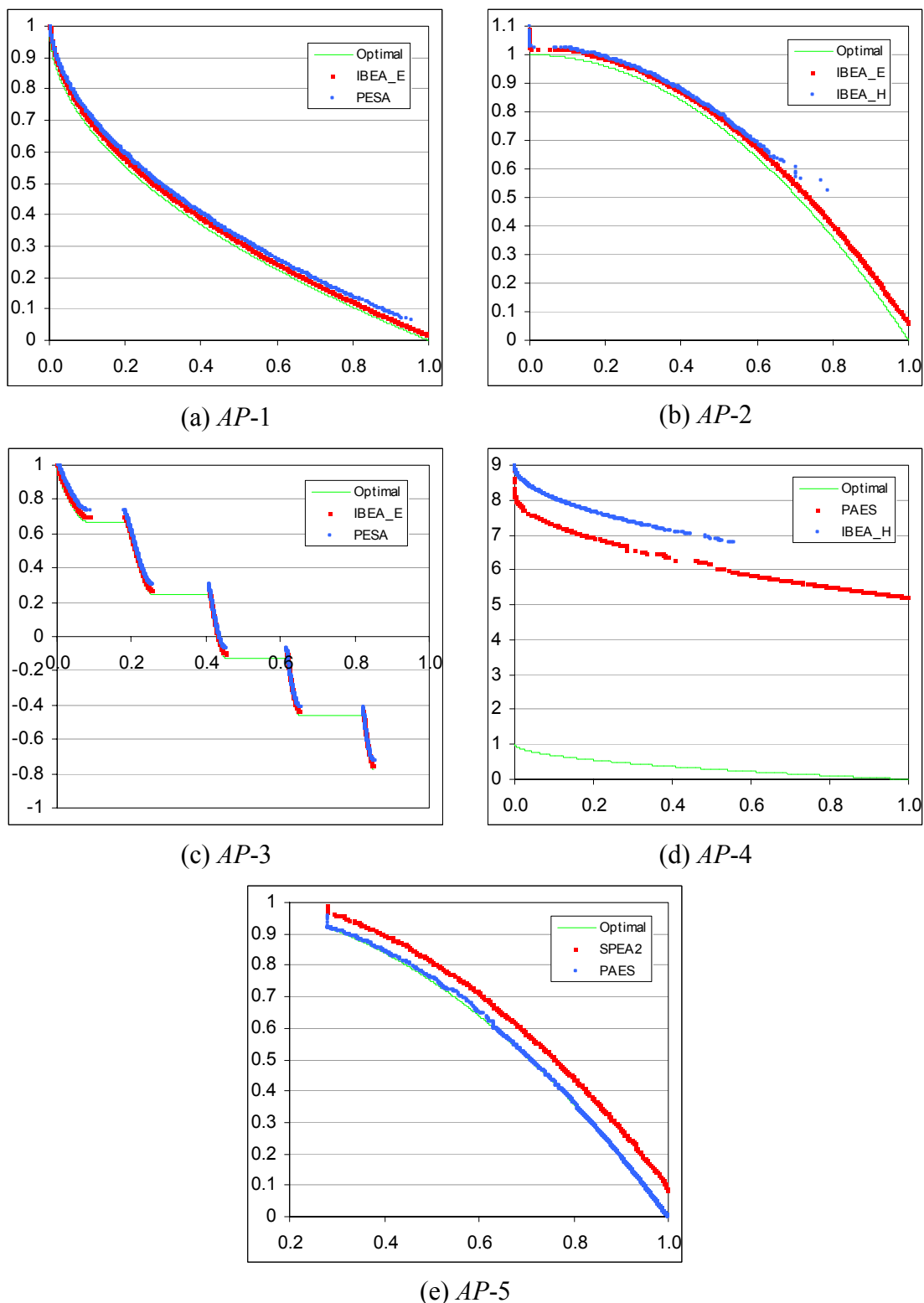
*y-axis*: average epsilon performance; *x-axis*: number of evaluations executed.

(a) *AP-1*(b) *AP-2*(c) *AP-3*(d) *AP-4*(e) *AP-5*(f) *AP-15*(g) *AP-16*(h) *AP-17*(i) *AP-21***Figure 150 — End-of-Run Epsilon Box-Plots**

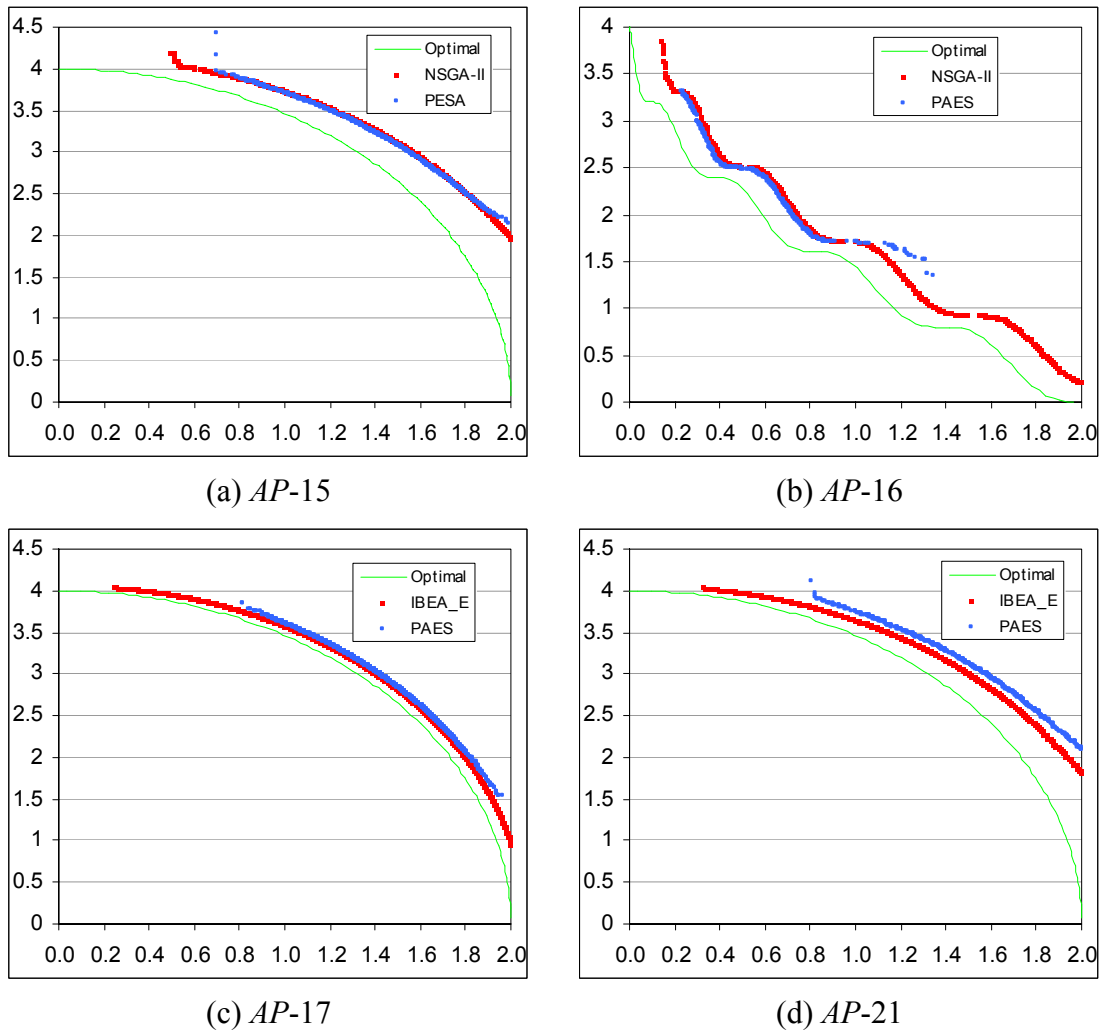
*y-axis* is the epsilon performance at the end of the run (7,000 evaluations in *AP-5*, 10,000 evaluations in all remaining functions); *x-axis* indicates the selected optimiser.

	PAES	PESA	SPEA2	NSGA-II	IBEA_E	IBEA_H	
PAES	-	1.123E-01	1.084E-01	<b>1.531E-11</b>	<b>1.027E-16</b>	1.016E-01	(a) AP-1
PESA	7.044E-01	-	<b>1.513E-03</b>	<b>4.618E-16</b>	<b>8.645E-22</b>	<b>1.358E-03</b>	
SPEA2	4.373E-01	6.909E-01	-	<b>1.274E-07</b>	<b>4.518E-12</b>	9.745E-01	
NSGA-II	1.643E-01	3.112E-01	5.380E-01	-	7.160E-02	<b>1.499E-07</b>	
IBEA_E	<b>6.399E-07</b>	<b>9.681E-08</b>	<b>1.200E-08</b>	<b>3.839E-10</b>	-	<b>5.513E-12</b>	
IBEA_H	<b>8.045E-05</b>	<b>3.470E-04</b>	<b>1.415E-03</b>	<b>9.624E-03</b>	<b>1.673E-17</b>	-	
(b) AP-2							
	PAES	PESA	SPEA2	NSGA-II	IBEA_E	IBEA_H	
PAES	-	1.210E-01	9.491E-02	8.549E-02	<b>4.212E-10</b>	<b>4.717E-08</b>	(c) AP-3
PESA	1.472E-01	-	9.044E-01	<b>1.163E-03</b>	<b>2.630E-14</b>	<b>6.540E-12</b>	
SPEA2	<b>8.060E-03</b>	2.254E-01	-	<b>7.667E-04</b>	<b>1.179E-14</b>	<b>3.096E-12</b>	
NSGA-II	5.898E-02	6.584E-01	4.405E-01	-	<b>3.284E-06</b>	<b>1.262E-04</b>	
IBEA_E	<b>4.937E-05</b>	<b>8.060E-03</b>	1.472E-01	<b>2.682E-02</b>	-	3.910E-01	
IBEA_H	<b>6.171E-09</b>	<b>8.278E-06</b>	<b>9.972E-04</b>	<b>5.433E-05</b>	6.216E-02	-	
(d) AP-4							
	PAES	PESA	SPEA2	NSGA-II	IBEA_E	IBEA_H	
PAES	-	<b>1.526E-28</b>	<b>3.855E-38</b>	<b>2.034E-24</b>	<b>1.390E-09</b>	<b>1.458E-15</b>	(e) AP-5
PESA		-	<b>8.146E-03</b>	2.350E-01	<b>2.743E-09</b>	<b>9.937E-05</b>	
SPEA2			-	<b>1.433E-04</b>	<b>1.289E-16</b>	<b>1.859E-10</b>	
NSGA-II				-	<b>1.264E-06</b>	<b>6.192E-03</b>	
IBEA_E					-	<b>2.909E-02</b>	
	PAES	PESA	SPEA2	NSGA-II	IBEA_E	IBEA_H	
PAES	-	7.968E-02	<b>1.907E-13</b>	<b>3.632E-11</b>	<b>1.707E-10</b>	1.594E-01	(f) AP-15
PESA	<b>1.323E-03</b>	-	<b>1.006E-18</b>	<b>3.923E-16</b>	<b>2.351E-15</b>	<b>1.692E-03</b>	
SPEA2	<b>1.884E-16</b>	<b>8.085E-08</b>	-	4.026E-01	2.729E-01	<b>1.035E-09</b>	
NSGA-II	<b>3.781E-19</b>	<b>7.332E-10</b>	3.868E-01	-	7.948E-01	<b>9.530E-08</b>	
IBEA_E	<b>1.296E-18</b>	<b>1.903E-09</b>	4.857E-01	8.662E-01	-	<b>3.518E-07</b>	
IBEA_H	<b>5.964E-07</b>	6.318E-02	<b>3.196E-04</b>	<b>9.476E-06</b>	<b>1.968E-05</b>	-	
(g) AP-16							
	PAES	PESA	SPEA2	NSGA-II	IBEA_E	IBEA_H	
PAES	-	5.896E-01	<b>1.936E-28</b>	<b>1.338E-30</b>	<b>1.178E-30</b>	<b>1.508E-11</b>	(h) AP-17
PESA	2.820E-01	-	<b>1.494E-26</b>	<b>1.098E-28</b>	<b>9.680E-29</b>	<b>3.746E-10</b>	
SPEA2	<b>5.349E-06</b>	<b>4.330E-04</b>	-	5.424E-01	5.321E-01	<b>1.881E-07</b>	
NSGA-II	<b>2.369E-11</b>	<b>1.145E-08</b>	<b>2.108E-02</b>	-	9.876E-01	<b>7.755E-09</b>	
IBEA_E	<b>1.323E-10</b>	<b>5.240E-08</b>	<b>4.310E-02</b>	7.739E-01	-	<b>7.127E-09</b>	
IBEA_H	<b>7.764E-12</b>	<b>4.238E-09</b>	<b>1.292E-02</b>	8.554E-01	6.387E-01	-	
(i) AP-21							

Figure 151 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Epsilon Results  
 Bold italics indicate significant differences at the 5% level.

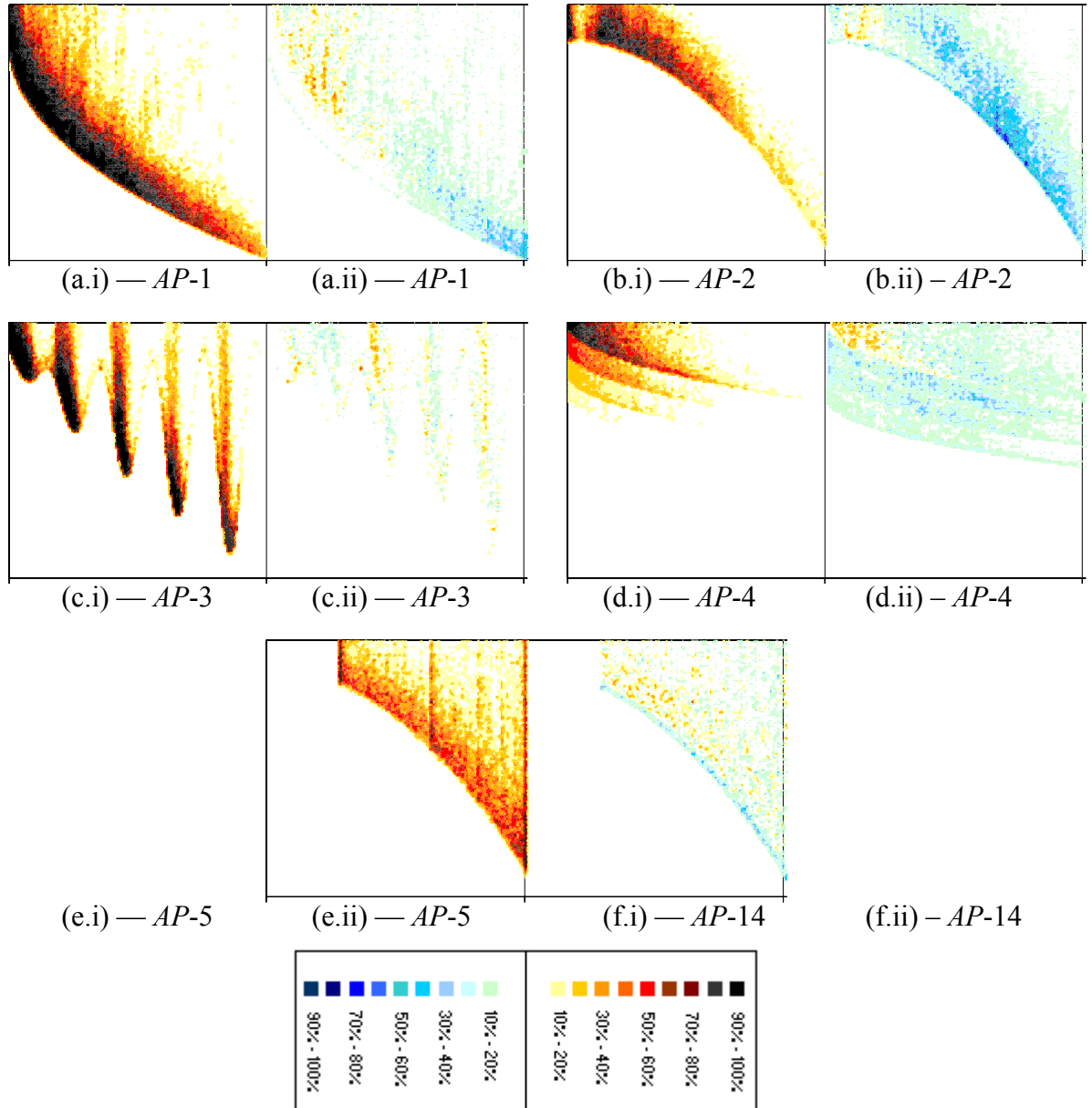


**Figure 152 —Best and Worst Median End-of-Run 50% Attainment Surfaces for Each Problem**  
*y-axis*: objective one; *x-axis*: objective two. Best and worst systems derived from median hypervolume scores provided in Figure 145.



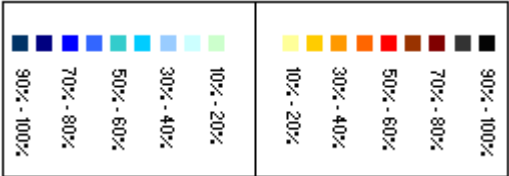
**Figure 153 —Best and Worst End-of-Run 50% Attainment Surfaces for Each Problem**

*y-axis*: objective one; *x-axis*: objective two. Best and worst systems derived from median hypervolume scores provided in Figure 145.



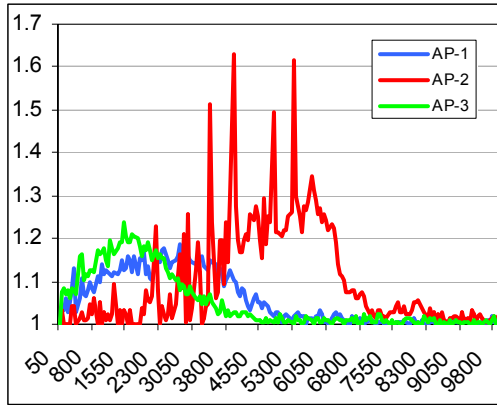
**Figure 154 — End-of-Run Frequency Matrices**

(i) Frequency matrix for IBEA\_H. (ii) The red palette represents where IBEA\_H more frequently attains a given region; the blue palette indicates where IBEA\_E attains an area more consistently.

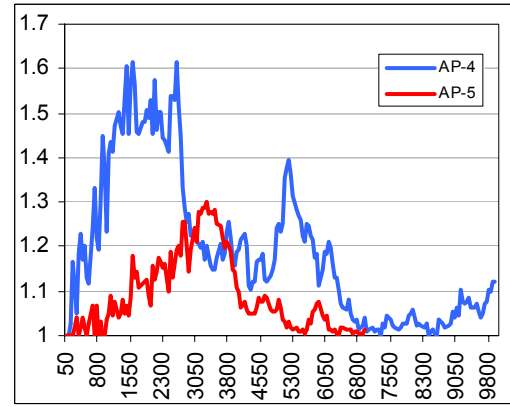


(i) Frequency matrix for IBEA\_H. (ii) The red palette represents where IBEA\_H more frequently attains a given region; the blue palette indicates where IBEA\_E attains an area more consistently.

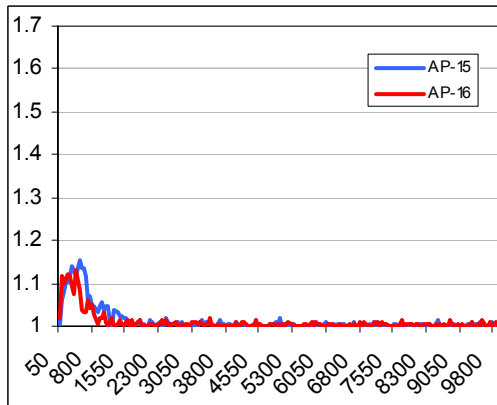




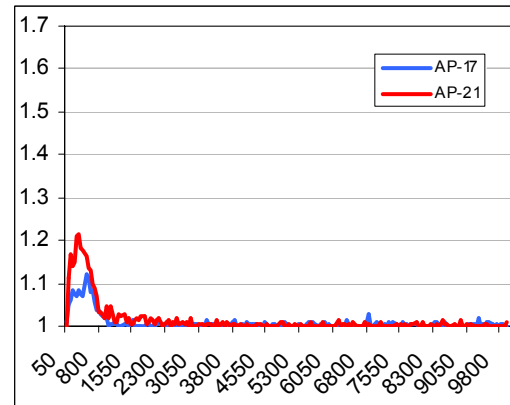
(a)



(b)



(c)



(d)

**Figure 156 —Average Size of Post-Truncation Cells Using PESA**  
*y-axis:* average cell size; *x-axis:* number of evaluations that PESA has performed.

# Chapter



Novel

Unbounded

Optimisers

*"There is no comparison  
between that which is lost by  
not succeeding and that lost  
by not trying"*

*Francis Bacon, Sr. —  
Lawyer and Philosopher*

*"A minute's success pays  
the failure of years."*

*Robert Browning — Poet*

*"If we do not succeed then  
we run the risk of failure."*

*Dan Quayle — Former Vice  
President of The United  
States of America*

## 11 NOVEL UNBOUNDED OPTIMISERS

In an effort to express the power and versatility of the Mak\_Tree, this section proposes and then explores two simple, novel, approaches to multiobjective optimisation. By focussing on the weaknesses seen in the contemporary evolutionary algorithms explored in Chapter 0, the techniques offer not only an insight into how researchers can harness the power of unbounded sets in interesting new ways, but also how some of the more problematic domain characteristics can be addressed.

### 11.1 INTRODUCING DIVERSITY\_PAES

Preceding sections have established a number of key flaws in the PAES approach, not least of which are an inability to effectively distribute the search along the optimal front (as illustrated in the *AP-1*, *AP-2* and *AP-3* results) and inefficiencies in zero-utility gradient spaces (as seen in *AP-16*). Clearly, these are important issues that limit the applicability of PAES in real-world studies. As such, this section proposes a novel new algorithm — Diversity\_PAES — which maintains the core mechanics of the basic PAES approach, but augments these with the inclusion of an unbounded Mak\_Tree.

#### 11.1.1 DESCRIBING DIVERSITY\_PAES

Where Mak\_PAES simply replaced the truncated PAES archive with an unbounded Mak\_Tree, Diversity\_PAES more thoroughly integrates the archival set into the evolutionary process. By capitalising on an extended Mak\_Tree with cuboid-based crowding annotations, the Diversity\_PAES algorithm has access to the least-crowded solution in the archival set during each iteration of the core evolutionary process. Rather than using this uncrowded solution purely for reference purposes, it is used to asexually produce one child per iteration, with the offspring integrated into the basic PAES system as per Algorithm 23. Note that the core mechanics remain largely unchanged — the principal difference is that the primary solution is now compared against a set of two alternatives.

As intimated by the name, the principal advantage of the Diversity\_PAES algorithm is that it expressly encourages the pursuit of elite uncrowded spaces. The traditional

## Algorithm 23 — The Diversity\_PAES Algorithm

1: <b>primary</b> := generateRandomSolution()	Start the search at random locations.
2: <b>remote</b> := generateRandomSolution()	
3: while (terminationConditionMet() $\neq$ true)	
4: <b>child</b> := asexualReproduction( <b>primary</b> )	Produce two children from the primary
5: <b>remoteChild</b> := asexualReproduction( <b>remote</b> )	and remote solutions.
6: updateArchiveAndGrid( <b>child</b> )	Insert the remote and primary children into
7: updateArchiveAndGrid( <b>remoteChild</b> )	the archive, where appropriate.
8: if ( <b>child</b> $\prec$ <b>primary</b> )	If the primary child dominates the primary
9:   if ( <b>remoteChild</b> $\prec$ <b>child</b> )	solution, but is dominated by the remote
10: <b>primary</b> := <b>remoteChild</b>	child, the remote child becomes primary.
11:   else if $\left( \begin{array}{l} \text{isInArchive}(\text{remoteChild}) \wedge \\ (\text{remoteChild} \sim \text{child}) \end{array} \right)$	Otherwise, if the remote child is non-
12:     if ( $\text{grid.crowd}(\text{child}) \leq \text{grid.crowd}(\text{remoteChild})$ )	dominated and incomparable to the
13: <b>primary</b> := <b>child</b>	primary child, then make the solution
14:     else	lying in the least crowded cell the
15: <b>primary</b> := <b>remoteChild</b>	primary solution.
16:     else	
17: <b>primary</b> := <b>child</b>	If the remote child is the only of the
18:   else if ( <b>remoteChild</b> $\prec$ <b>primary</b> )	children to dominate the primary
19: <b>primary</b> := <b>remoteChild</b>	solution, then it becomes the primary.
20:   else if (isInArchive( <b>child</b> ))	If neither child dominates the primary
21:     if ( $\text{grid.crowd}(\text{child}) \leq \text{grid.crowd}(\text{primary})$ )	solution, select the least crowded non-
22: <b>primary</b> := <b>child</b>	dominated child (if one exists).
23:     if (isInArchive( <b>remoteChild</b> ))	If the selected child is in less crowded
24:       if ( $\text{grid.crowd}(\text{remoteChild}) \leq \text{grid.crowd}(\text{primary})$ )	space than the primary solution, then
25: <b>primary</b> := <b>remoteChild</b>	it becomes the primary.
26:     else if (isInArchive( <b>remoteChild</b> ))	
27:       if ( $\text{grid.crowd}(\text{remoteChild}) \leq \text{grid.crowd}(\text{primary})$ )	
28: <b>primary</b> := <b>remoteChild</b>	The remote parent is the least-crowded
29: <b>remote</b> := selectLeastCrowded(Archive)	solution in the archive.

PAES technique suffers in this respect as it must happen upon uncrowded regions in order to capitalise upon them — if uncrowded space is far-removed from the current location of the primary solution, navigating to that region is unlikely to be efficient. Furthermore, the incorporation of uncrowded solutions aids in the prevention of stagnation and offers an escape from deceptive regions. If a basic hill-climbing approach like PAES descends a steep fitness gradient into deceptive isolated space<sup>81</sup>, it is inherently difficult to escape, as the fitness space surrounding the primary

<sup>81</sup> In single-objective optimisation, this is typically referred to as a *fitness well*, since once a hill-climber falls in, it is difficult to get back out.

solution will not correctly orient the search. By maintaining a link with uncrowded space, the search is provided with another avenue of escape. The fact that the alternative will typically lie away from the deceptive region — since this will become increasingly crowded as the search focus intensifies — only aids in production of viable escape routes.

Thus, Diversity\_PAES remains true to the low-population, asexual, cell-based mechanics of the original system, but incorporates elite uncrowded solutions to better drive diversity and aid performance in problematic domains. Moreover, by capitalising on an unbounded archive, the selection of the least crowded elite solution is guaranteed to be correct — the proposal will *always* be a member of the leading front and reside in the least-occupied cell.

## 11.2 INTRODUCING MAK\_OS

Of all of the examined problem domain characteristics, multi-frontality appears to be the most difficult for conventional multi-member systems to overcome. An interesting mechanism designed to diminish such problems is the introduction of noise into the evolutionary process (as proffered in Section 10.4.2.2) — the explicit goal of which is to encourage a broader search that is less likely to prematurely converge. Capitalising on this notion, the new Mak\_OS (Over-use Selection) algorithm is designed to offer both an extremely simple approach to unbounded optimisation and a system that may better address multi-frontal concerns.

### 11.2.1 DESCRIBING MAK\_OS

Designed exclusively for use with an unbounded archive, the Mak\_OS system (as described in Algorithm 24) differentiates itself from the standard methodologies of those multi-member techniques examined throughout this chapter in a number of key areas. In particular, all solutions involved in the breeding process are guaranteed to reside in the prevailing front<sup>82</sup> (since the breeding pool is composed only of members of the unbounded elite archive); the initial selection set is based on unbounded crowding estimations (which avoid the inaccuracies of crowding inherent in a

---

<sup>82</sup> A similar strategy is employed in PESA, the Single Front Genetic Algorithm (SFGA) and Parallel SFGA (PSFGA) systems [255], though their use of truncated archives precludes guarantees about frontal membership.

**Algorithm 24 — The Mak\_OS Algorithm**

---

**Inputs:**

$\alpha$	The fraction of the archive to be included in the breeding pool ( $0 < \alpha \leq 1$ ).
$b$	The maximum number of solutions allowed in the breeding pool.
1: $Children := generateRandomSolutions()$	Start the search at random locations.
2: while (terminationConditionMet() $\neq$ true)	
3: $Parents := \emptyset$	Insert the children into the
4: updateArchive( $Children$ )	archive (where appropriate).
5: $Pool := selectLeastCrowded(Archive, (\alpha \times  Archive ))$	Fill the pool with $(\alpha \times  Archive )$ uncrowded
6: while ( $ Pool  > (0.5\alpha \times  Archive )$ )	solutions and truncate according to usage.
7: removeMostUsed( $Pool$ )	
8: while ( $ Parents  < b$ )	Perform binary tournament selection
9: $first := selectRandom(Pool)$	based on solution usage. If solution usage
10: $second := selectRandom(Pool)$	is equivalent, then use crowding estimation
11: if (usage( $first$ ) < usage( $second$ ))	to distinguish preferability.
12: $Parents := Parents \cup \{first\}$	
13: incrementUsage( $first$ )	Update the usage rate of the selected
14: else if (usage( $second$ ) < usage( $first$ ))	solution.
15: $Parents := Parents \cup \{second\}$	
16: incrementUsage( $second$ )	Update the usage rate of the selected
17: else if (crowd( $second$ ) < crowd( $first$ ))	solution.
18: $Parents := Parents \cup \{second\}$	
19: incrementUsage( $second$ )	
20: else	
21: $Parents := Parents \cup \{first\}$	
22: incrementUsage( $first$ )	
23: $Children := reproduce(Parents)$	Reproduce with the parental set.

---

truncated environment); the cardinality of the selection set varies according to the size of the true non-dominated front (based on the  $\alpha$  parameter); and the fitness of selection-set members is defined by the number of times they have bred in the past (with lower scores being preferable). Importantly, the application of usage-based fitness for selection set truncation and binary tournaments not only varies the constituents of the breeding pool under stagnation conditions, but introduces a level of noise to the evolutionary process (since the least crowded solutions are not always selected). Additionally, by varying the size of the selection set (and, in-turn, the breeding pool), the Mak\_OS system takes on the properties of a simple hill-climbing system while the front is small and a multi-member optimiser when the front is large, thus encouraging aggressive dominance-based searching early in the run, and diversification when the front is suitably developed.

Thus, the Mak\_OS system represents a simple new approach to multiobjective optimisation that is tailored to the use of unbounded archival sets and the requirements implicit of multi-frontal domains. It is expected that its core mechanics should present scope for interesting future research.

### 11.3 EMPIRICAL ANALYSIS METHODOLOGY

The performance of the Mak\_OS and Diversity\_PAES techniques are contrasted with the contemporary evolutionary algorithms explored in Section 0 — namely, NSGA-II, SPEA2, PESA, PAES, IBEA\_E and IBEA\_H<sup>83</sup>. All Mak\_OS and Diversity\_PAES systems are run twenty times per performance indicator (adhering to the methodologies outlined in Section 9.1.2.1), with results for other systems extracted from preceding sections. The Mak\_OS technique uses  $\alpha = 0.2$ , the Diversity\_PAES technique uses the same grid as in basic PAES (with 10,000 distinct cells), and all systems feature matched mutation and (where appropriate) crossover rates to maintain consistency. Continuing the exploration of crowding estimations in optimiser performance, the Mak\_OS system is implemented with NSGA-II-style cuboid approximations (henceforth referred to simply as Mak\_OS) and also an averaged  $\kappa$  nearest-neighbours ( $\kappa = 20$ ) approach (referenced as Mak\_OS\_KNN). As per Chapter 0, in the interests of brevity and clarity, frequency and attainment surfaces will be used sparingly in this analysis, while graphs and tables are presented *en masse* at the end of this chapter (pages 361–373).

## 11.4 EMPIRICAL ANALYSIS

### 11.4.1 DIVERSITY\_PAES PERFORMANCE

Relative to the basic PAES approach, the performance of the new Diversity\_PAES algorithm is particularly impressive. Indeed, Diversity\_PAES is *significantly* better than the contemporary asexual system on every function other than *AP-4* according to end-of-run epsilon indicators (Figure 166, Figure 167 and Figure 168) and is *significantly* better on all bar *AP-2*, *AP-4* and *AP-21* when drawing inferences from the hypervolume results (Figure 160, Figure 161 and Figure 162). Moreover, Diversity\_PAES is never *significantly* worse than the original PAES system on any tested function according to both examined indicators.

---

<sup>83</sup> The performance of the new systems against the Mak extensions seen in Section 9.2 is described later (see Section 12).

Looking at specific domains, the improvement seen in Diversity\_PAES on *AP-1*, *AP-2* and *AP-3*, particularly in the later stages of each run (Figure 157 and Figure 163), suggests that the incorporation of the remote solution into the evolutionary process better encourages a well-distributed search than the simple PAES approach when nearing convergence. Moreover, the hill-climbing Diversity\_PAES system is competitive against the existing multi-member systems. Specifically, it is *significantly* better than SPEA2, NSGA-II and PESA according to at least one metric (Figure 160, Figure 161, Figure 166 and Figure 167) for all three functions (though NSGA-II is *significantly* better under one metric for *AP-1*), and is otherwise only *significantly* worse than the impressive IBEA\_E on *AP-2* and *AP-3*<sup>84</sup>. Such results indicate that even small asexual systems are capable of generating well-distributed sets so long as they are driven by sufficiently powerful and accurate crowding mechanisms.

While Diversity\_PAES offers a *significant* improvement over the basic PAES technique on the *AP-16* function — likely because the inclusion of the remote solution offers an alternative means of escape — it is still *significantly* outperformed by SPEA2, NSGA-II and IBEA\_E under both selected indicators (Figure 160, Figure 162, Figure 166 and Figure 168). The implication is that well-distributed multi-member systems are more equipped to handle zero-utility gradients, principally because it is less likely for all members to lie in flat fitness space and, as such, the search is better able to orient itself.

The sharp improvement on the *AP-5* function throughout the run (Figure 158 and Figure 164) suggests that when a steep utility gradient exists, the Diversity\_PAES technique is able to effectively capitalise on it, while the improved end-of-run results (Figure 160, Figure 161, Figure 166 and Figure 167) — where Diversity\_PAES is *significantly* better than all examined truncated systems under both metrics — further illustrate that the new technique can effectively distribute the search on convergence. There is debate though as to whether such an aggressive pursuit of promising gradients can be detrimental to the behaviour of the optimiser in less favourably arranged fitness-spaces, where the likes of deception could lead to evolutionary

---

<sup>84</sup> IBEA\_E significantly outperforms Diversity\_PAES on *AP-1* under the hypervolume metric, but, conversely, is significantly bettered according to the epsilon indicator. This infers that the two algorithms produce incomparable fronts with respect to this problem.



dead-ends. In particular, PAES (see Section 10.4.1) performed impressively in *AP-5*, but was the worst of all tested approaches in the deceptive *AP-17* function. It is reassuring then that Diversity\_PAES offers not only a *significant* improvement over the basic PAES approach in the complex *AP-17* problem, but also over every other tested contemporary system (with all bar SPEA2 being *significantly* outperformed on *both* indicators — see Figure 166, Figure 168, Figure 160 and Figure 162). Such powerful results suggest that the inclusion of a remote solution into the evolutionary process is sufficient to prevent an over-emphasis on deceptive spaces (so long as the remote solution lies beyond the deceptive region, which is likely) without foregoing the efficiencies familiar to basic hill-climbing systems — namely, the capacity to exploit valuable search gradients.

A particularly surprising result is seen in the highly non-separable *AP-15* function: the asexual Diversity\_PAES system is not just competitive with the examined multi-member systems, but frequently preferable. Indeed, Diversity\_PAES is *significantly* better than all examined systems with respect to the hypervolume metric (Figure 160 and Figure 162) and similarly preferable over every system other than SPEA2 under the epsilon indicator (Figure 166 and Figure 168). The results suggest that the diversity mechanism employed in the new technique is sufficient to develop and maintain multiple valuable gene-dependencies simultaneously (unlike the conventional PAES system, see Section 10.4.1) and that the asexual reproduction system may be better suited to highly non-separable tasks than the popular sexual technique. Indeed, it is possible that sexual crossover is of lower utility in problems like *AP-15* since it can disrupt cross-gene dependencies — by manipulating only a small number of alleles (typically one) during each reproduction, the asexual system is better able to maintain connections in the chromosome and provides a more incremental approach to change. This is an interesting idea, but one which certainly requires further investigation — as such, it rests as a potential avenue of future work.

With respect to the multi-frontal *AP-4* problem, Diversity\_PAES *significantly* outperforms PESA, SPEA2, NSGA-II, IBEA\_E and IBEA\_H under the hypervolume metric (Figure 160 and Figure 161) and similarly betters SPEA2, NSGA-II, IBEA\_E and IBEA\_H given epsilon analysis (Figure 166 and Figure 167). Such positive results illustrate that although the Diversity\_PAES system incorporates a remote solution with the express purpose of better distributing the search, it is still capable of

effectively punctuating false fronts due to its capacity to quickly refocus the majority of the search around dominating children (as per PAES, see Section 10.4.1).

The worst performance of the Diversity\_PAES system relative to the contemporary multi-member systems is in *AP-21*, where it is *significantly* bettered by IBEA\_E, IBEA\_H and NSGA-II under epsilon indicators (Figure 166 and Figure 168) and SPEA2, NSGA-II, IBEA\_H and IBEA\_E according to hypervolume measures (Figure 160 and Figure 162). Due to the overlapping domain characteristics present in this complex function, it is difficult to provide a definitive statement as to the cause of such performance degradation, but it is likely due to an interaction between multi-frontality and isolation. Should Diversity\_PAES puncture a false front into an isolated (or, worse, deceptive) region, the additional search emphasis in this low-yield area is unproductive. Interestingly then, the very mechanisms which are responsible for improved performance in simple multi-frontal problems like *AP-4*, may rest as a burden in certain configurations of multi-frontal space. This notion merits further work.

#### **11.4.2      MAK\_OS VERSUS MAK\_OS\_KNN**

With respect to the two examined crowding estimation mechanisms used in the Mak\_OS technique, the averaged  $\kappa$  nearest-neighbours approach yields generally preferable results, with Mak\_OS\_KNN demonstrating *significantly* better end-of-run performance than the cuboid-based approach on *AP-3*, *AP-15* and *AP-17* under hypervolume metrics (Figure 160, Figure 161 and Figure 162) and on *AP-15* and *AP-17* when considering epsilon metrics (Figure 166, Figure 167 and Figure 168). Moreover, the  $\kappa$  neighbours-based system is never *significantly* worse than the cuboid technique on any examined problem under either chosen metric. To understand the reasons for such disparity, it is worthwhile examining the performance of each approach on the simple *AP-1*, *AP-2* and *AP-3* test functions, where the basic utility of each diversity estimation mechanism is best observed (as described in Section 10.4.2.1). As illustrated in the corresponding progressive epsilon (and, to a lesser extent, hypervolume) graphs (Figure 163 and Figure 157), the cuboid approach produces preferable performance early in the run, when the population is small, but is inferior nearer to the termination point. This lends further empirical weight to an important point — the utility of the underlying diversity

mechanism in multiobjective optimisers is intrinsically tied to the size of the population from which the estimations are drawn. The cuboid approach is preferable early, as the population levels are too low for the high valued  $\kappa$  mechanism to function effectively — it will suffer from biases around the extremities of the front and poor emphasis of uncrowded *individuals*<sup>85</sup>. The cuboid approach suffers later in the run, particularly around end-of-run convergence, as the technique is too narrow in focus — emphasising locally uncrowded solutions, when the development of these regions may not lead to a markedly better distribution of the frontal set as a whole.

Given the significance of front cardinality in diversity estimation, this thesis recommends further investigation into dynamically changing diversity estimators. An interesting approach that can be of relatively low cost is to maintain multiple crowding-based annotations in the Mak\_Tree, with selection operating on the most appropriate annotation given the current population size. As an illustration — if the Mak\_Tree maintains annotations for cuboid, small-neighbourhood-based and large-neighbourhood-based crowding approximations, all that is required is a basic heuristic for selecting the most appropriate estimator given the current frontal size. Clearly the development of such heuristics also rests as an interesting piece of future work — not just for the formation of appropriate triggers in dynamic settings, but as a guide for researchers working with truncated archives, where the maximum size of the frontal set will be known in advance (as discussed earlier, Zitzler *et al.* [81] offers some work in this area with respect to the appropriate setting of  $\kappa$  in  $\kappa^{\text{th}}$  nearest-neighbour estimators, but this is not particularly well supported with empirical proof of its utility to multiobjective optimisers).

### 11.4.3 MAK\_OS\_KNN PERFORMANCE

Since the  $\kappa^{\text{th}}$  nearest-neighbour approach yields generally better results than the cuboid technique for the Mak\_OS methodology, subsequent analyses focus on the behaviour of Mak\_OS\_KNN. Results illustrate that the performance of this system is particularly impressive on the simple *AP-1*, *AP-2* and *AP-3* functions (see Figure 160, Figure 161, Figure 166 and Figure 167) where it is *significantly* better than all

---

<sup>85</sup> The inclusion of noise (via the usage mechanic) in the Mak\_OS\_KNN system diminishes the influence of such regional-bias and encourages a more diverse collection of selected solutions upon which to build a search. It is likely that such noise reduces the negative impact of the  $\kappa$  *nearest-neighbours* mechanism at lower population levels (unlike in the elitist Mak\_SPEA2\_KNN technique — see Section 9.2.4).

examined (non-OS) algorithms other than IBEA\_E under both epsilon and hypervolume metrics. Moreover, it is never *significantly* worse than IBEA\_E and *significantly* outperforms it according to both metrics in AP-2 and under the epsilon metric in AP-1. Similarly strong performance is seen in the Mak\_OS\_KNN system on AP-5 (again, see Figure 160, Figure 161, Figure 166 and Figure 167), where it is *significantly* better than all examined pre-existing optimisers according to both numerical metrics.

The key to such impressive results is related to the dual nature of the Mak\_OS\_KNN system — when the search is progressing rapidly, it takes on the characteristics of a simple hill-climbing system with a prioritisation of dominance over diversity (due to the small size of the prevailing front – see Figure 169); when the search convergences, it takes on more familiar multi-member properties, with an increased bias towards diversification (due to the increased size of the archival set and the influence of crowding estimates in solution selection). In simple objective-spaces like AP-1, AP-2, AP-3 and AP-5, this dual nature means that search gradients can be effectively exploited to locate optimal regions of space rapidly, with diversification taking a secondary role until expansion along that optimal region becomes necessary. Furthermore, since Mak\_OS\_KNN capitalises on an unbounded archive, the search is guaranteed to follow only non-dominated solutions (unlike in the basic PAES system, where the non-domination of primary solutions cannot be assumed).

However, as discussed previously (see Section 10.4.1), an aggressive pursuit of high-yield gradients at the expense of early diversification can be disadvantageous, particularly in the presence of strong deception — this was particularly obvious in the AP-17 function, where the PAES hill-climbing strategy faltered. The Diversity\_PAES results (see Section 11.4.1) suggested that the key to avoiding the lures of such deceptive spaces with simple hill-climbing systems lay principally with the explicit inclusion of remote solutions to encourage a more expansive search. The *significant* improvement seen in the Mak\_OS\_KNN technique over the simple PAES approach (Figure 160, Figure 162, Figure 166 and Figure 168) similarly suggests that while the priority of the Mak\_OS\_KNN search rests with dominance during the low-population phase of optimisation, it also encourages the pursuit of sufficiently diverse regions to avoid becoming fixated on deceptive spaces.

While the Mak\_OS\_KNN system may be capable of delivering preferable results to the basic hill-climbing system, the integration of noise can make it inferior to more conventional systems. In particular, Mak\_OS\_KNN has *significantly* worse end-of-run performance in AP-17 (Figure 160, Figure 162, Figure 166 and Figure 168) against NSGA-II, SPEA2, IBEA\_E and Diversity\_PAES — the likely cause of which is the application of usage-based fitness measures, which can diminish the influence of uncrowded solutions and reduce the frequency with which these regions are explored. If uncrowded solutions represent regions away from alluring deceptive spaces (as they may often do), this relatively minor reduction in the importance of remote solutions may reduce performance in biased domains.

The inclusion of noise though does come with some benefits — the most obvious of which is visible in the multi-frontal AP-4 problem. *Significantly* outperforming every examined non-hill-climbing system (Figure 160, Figure 161, Figure 166 and Figure 167) under both epsilon and hypervolume metrics (improvement over PAES is *insignificant*), the Mak\_OS\_KNN technique achieves better frontal progression in this difficult problem space principally due to the inclusion of usage-based fitness (and, to a lesser extent, smaller early populations that better facilitate frontal punctuation). When the optimiser converges onto a false front, the usage-derived fitness scores encourage variation in the breeding pool, such that a broader range of progenitors may be explored. The effect is that the Mak\_OS\_KNN technique is better equipped to search *along* the front when convergence occurs, while the biasing of newly created solutions better focuses the search around puncturing proposals when they are produced. In contrast, by focussing on the least-crowded solutions only, contemporary multi-member systems are limiting the scope of the search when convergence occurs — this is reasonable when diversification of an optimal frontal set is key, but becomes less appropriate when progression beyond the front is the primary concern.

Given the simplicity of the approach, the performance of the Mak\_OS\_KNN system on AP-16 is also impressive (see Figure 160, Figure 162, Figure 166 and Figure 168). In particular, Mak\_OS\_KNN is only *significantly* bettered by NSGA-II<sup>86</sup> (under the epsilon metric); shows a *significant* improvement over PAES, Diversity\_PAES and

---

<sup>86</sup> SPEA2 is preferable under epsilon metrics, but inferior under hypervolume metrics, thus indicating an incomparability between the sets produced by each system

PESA under both indicators; and yields *significantly* better end-of-run results than IBEA\_H according to hypervolume. The results suggest that even under relatively small populations (see Figure 169), the fitness mechanism is sufficient to encourage the type of diversity necessary to escape zero-utility gradient surfaces. The result also lends weight to the notion that multi-member systems are better suited to problem domains with flat objective-spaces, as illustrated by the superiority of Mak\_OS\_KNN over both PAES and Diversity\_PAES.

The Mak\_OS\_KNN system is less successful in the non-separable *AP*-15 problem, but again remains competitive (see Figure 160, Figure 162, Figure 166 and Figure 168). Specifically, it is *significantly* worse than Diversity\_PAES and IBEA\_E on at least one metric, though it is *significantly* better on at least one metric when compared with IBEA\_H, PESA, PAES and SPEA2. The small degradation in performance relative to other successful multi-member systems is again attributable largely to noise. By reducing the selective pressure on diversification, a smaller set of valuable dependencies is likely to be explored. The effect is similar to that seen in PAES, where over-specialisation can become problematic, though it occurs on a considerably smaller scale and is thus less damaging.

Performance on the complex *AP*-21 function (see Figure 160, Figure 162, Figure 166 and Figure 168) is particularly note-worthy, given that the Mak\_OS\_KNN system is only *significantly* bettered by IBEA\_E (under the hypervolume metric). Beyond this single system, Mak\_OS\_KNN produces *significantly* better end-of-run results than Diversity\_PAES, PAES, SPEA2 and PESA under both performance indicators, while also illustrating a *significant* improvement over NSGA-II under hypervolume measures. This impressive performance shows the adaptability of the simple Mak\_OS\_KNN system and again illustrates that smaller selection sets (see Figure 169) are capable of generating impressive results even in complex, multi-faceted, domains.

#### 11.4.4 SUMMARISING KEY FINDINGS

Table 35 offers a summary of the key properties of the novel systems discussed in this chapter and provides a brief overview of some of the more important issues that have been elucidated by their application. The summary again emphasises the importance of a thorough algorithmic study across a rich problem set.

**Table 35 - Summarising Key Findings in the Comparative Study of Novel and Contemporary Evolutionary Algorithms**

<b>General Properties of Novel Systems</b>
The incorporation of a globally-remote (non-dominated) solution into the hill-climbing process improves search capabilities, particularly when nearing convergence.
Diversity_PAES encourages aggressive pursuit of positive utility gradients, while the incorporation of additional diversification pressure curbs the potential for convergence in deceptive regions (a limitation in conventional hill-climbing systems like PAES).
The dual nature of the Mak_OS system means that it adapts aggressive hill-climbing properties early in the run, and a more search-centric, population-based, approach when nearing frontal convergence.
<b>Domain-Related Issues</b>
Further empirical evidence that though smaller populations are well suited to multi-frontal domains (where they may puncture false fronts more effectively), they are inappropriate for use in zero-utility gradient spaces (where an insufficient number of alternative search paths are provided).
Preliminary results suggest that asexual breeding may improve performance on non-separable problems by better maintaining gene dependencies.
Results indicate that selection-noise does improve performance on multi-frontal problems by encouraging a more thorough search along false fronts. The reduction in importance of remote solutions may degrade performance when pursuit of uncrowded regions is key however (as in non-separable or deceptive problems).
<b>Crowding Estimation Issues</b>
Additional empirical evidence that the selection of an appropriate crowding operator is contingent upon the size of the population. The neighbourhood technique in Mak_OS_KNN is preferred later in the run where the population is large, while the cuboid approach is preferable when the non-dominated set is small.

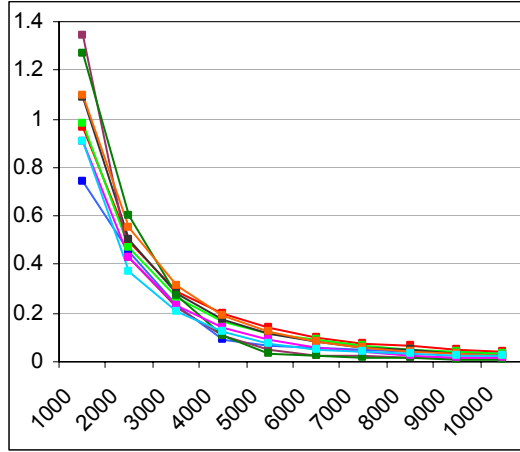
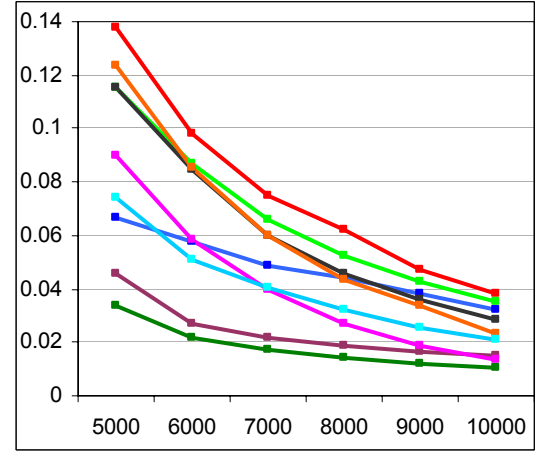
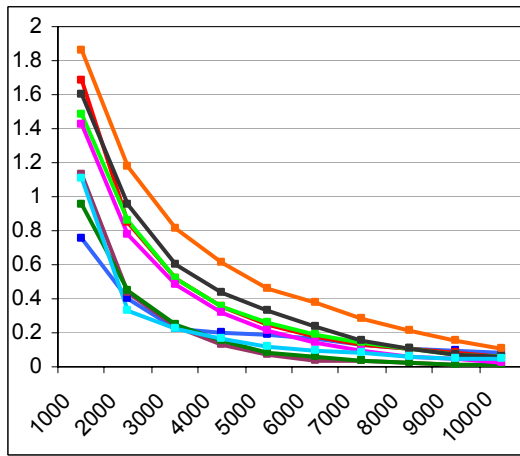
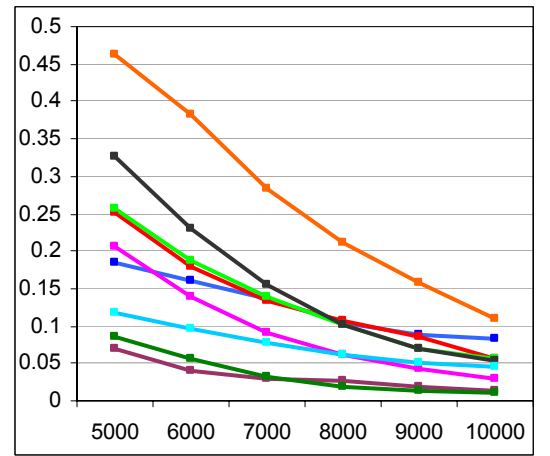
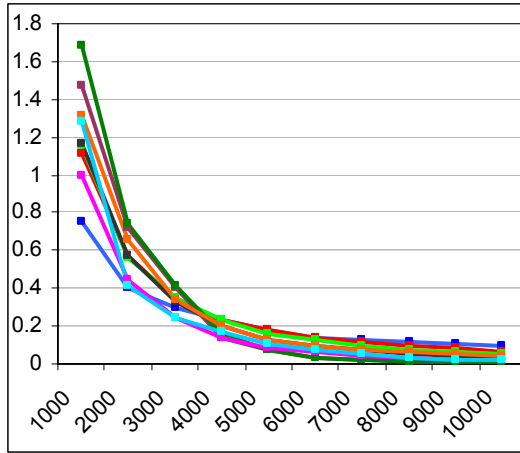
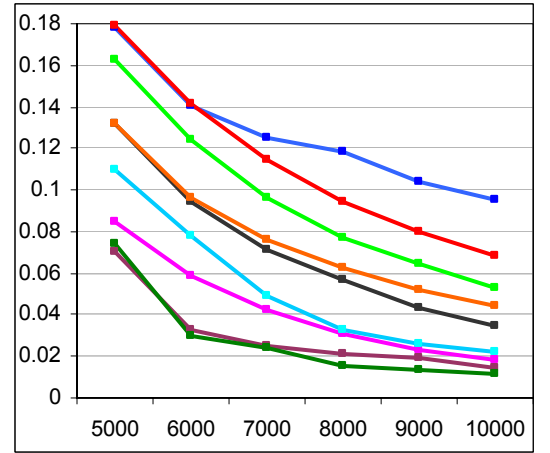
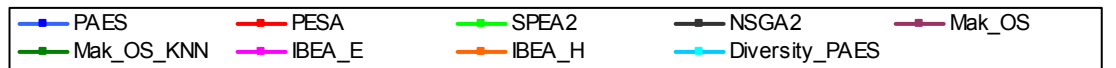
## 11.5 CONCLUSIONS

Via the integration of unbounded archiving and the application of novel techniques, the Mak\_OS and Diversity\_PAES systems are both simple approaches capable of generating particularly powerful results. Indeed, though Diversity\_PAES is essentially a low-cost hill-climbing algorithm augmented with explicit diversification mechanisms, it consistently *significantly* outperforms the original PAES system and is noticeably more robust — in particular, offering marked improvements in deceptive problem domains (where the increased diversity is important) and non-separable functions (where both diversity and asexual reproduction appear to be pivotal). Moreover, this simple technique is competitive with, and often better than, a host of contemporary population-based algorithms — further illustrating the capacity of unbounded archiving to deliver powerful performance gains.

The new Mak\_OS\_KNN approach yields similarly impressive results. By capitalising on an adaptively sized binary tournament that is predicated on usage statistics, the unbounded technique offers an aggressive, dominance-centric, search while the archival set is small (typically early in the run) and a more expansive (potentially noisy) search when the set grows. The consequence is a system that performs particularly well in simple problem spaces, where the aggressive search is beneficial, and in multi-frontal domains, where the noise and initially narrow search are important. Moreover, excluding *AP-17*, across the remaining problem domains it is generally competitive with, and often better than, the popular pre-existing systems — illustrating a level of consistency that is lacking from all of the observed systems other than IBEA\_E.

Such promising results should be read as an open invitation to researchers. If the development of simple novel algorithms predicated on the use of Mak\_Trees can yield impressive gains, then the scope for more intricate unbounded systems is considerable. As an illustration, by capitalising on adaptive diversity estimation techniques, integrating trailing fronts in fitness approximations or seeking more fine-grained mechanisms for adaptive selection procedures (such as new ways to appropriately size binary tournaments), a host of new or augmented unbounded algorithms become available — all of which merit at least some further investigation. The pursuit of such work is strongly encouraged.



(a) *AP-1*(b) *AP-1*(c) *AP-2*(d) *AP-2*(e) *AP-3*(f) *AP-3***Figure 157 — Progressive Hypervolume Averages**

*y-axis*: average hypervolume performance; *x-axis*: number of evaluations executed.

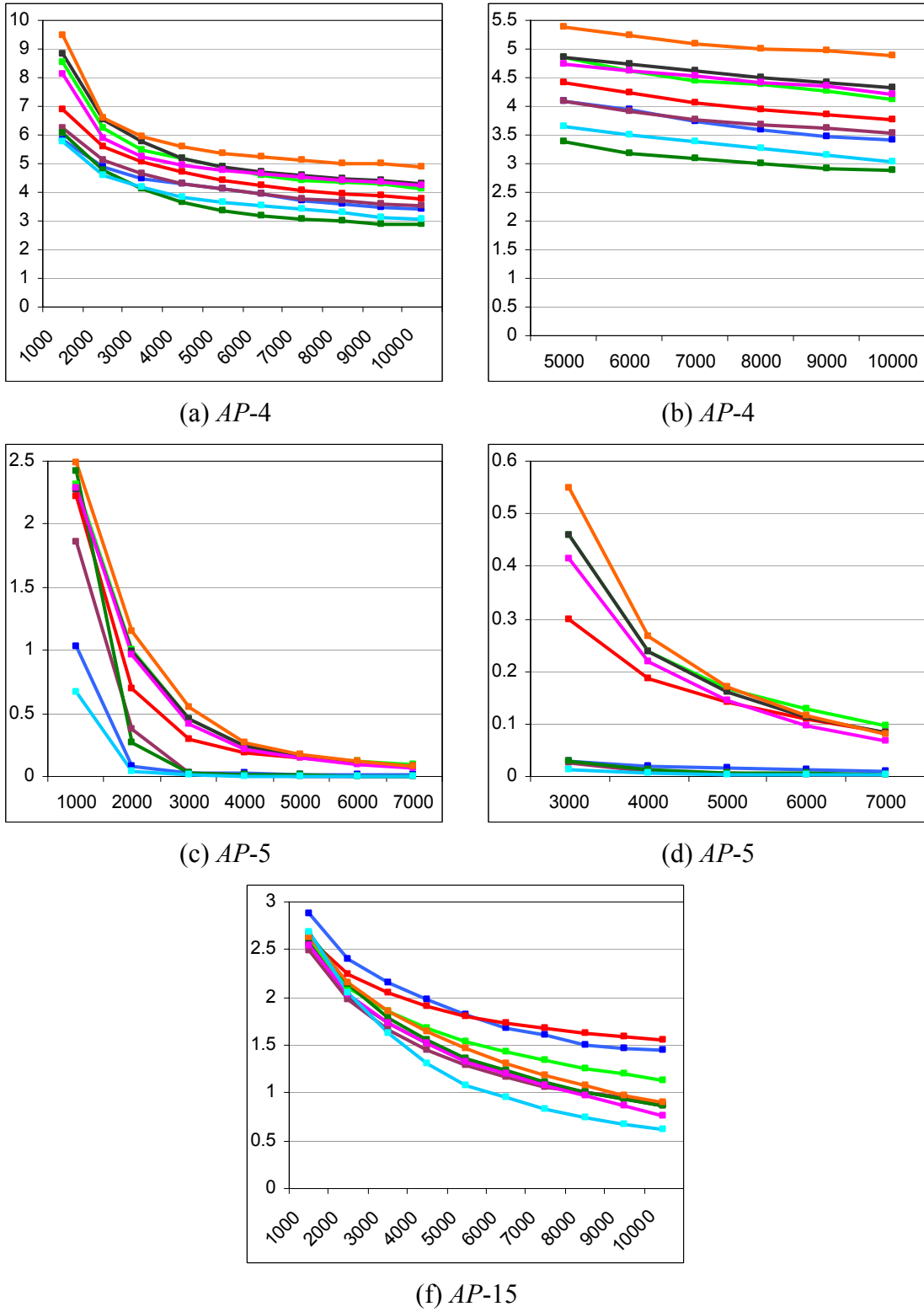
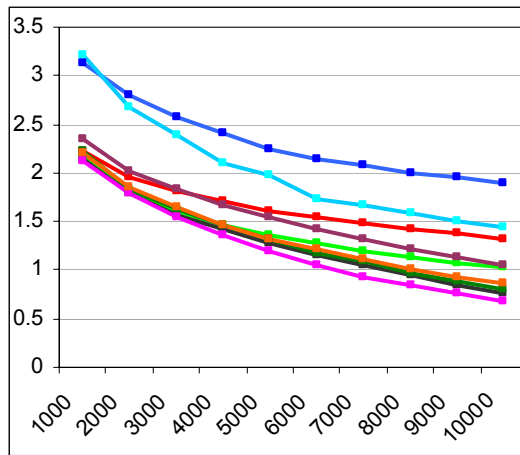
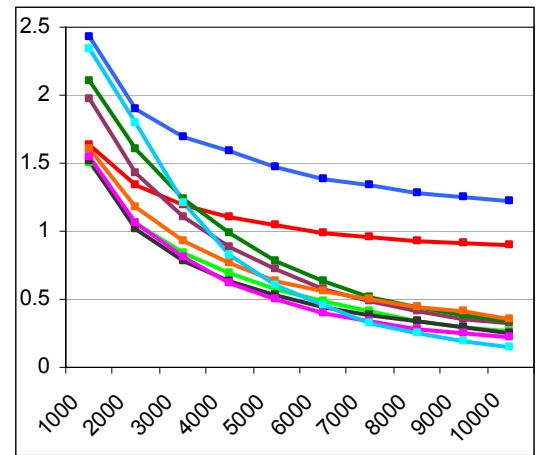
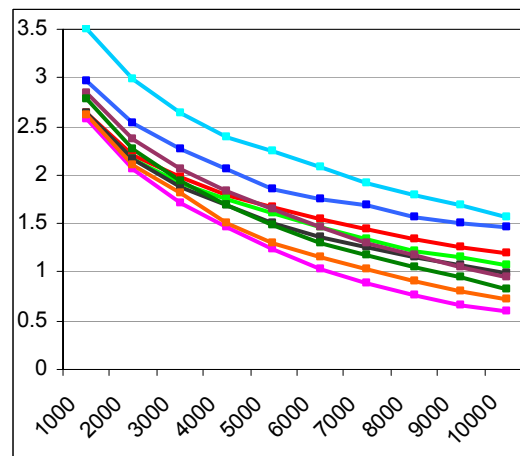
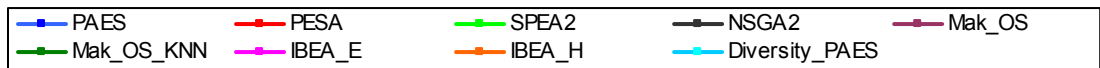
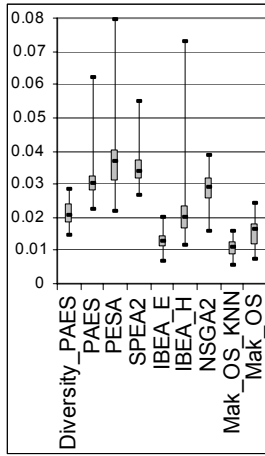
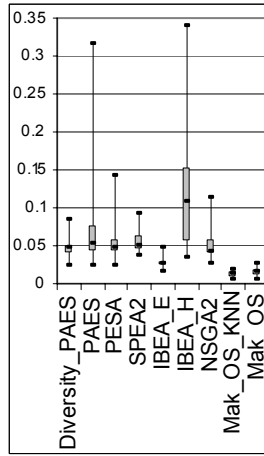
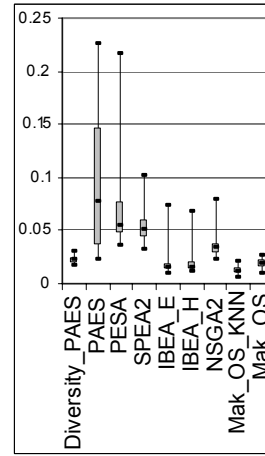
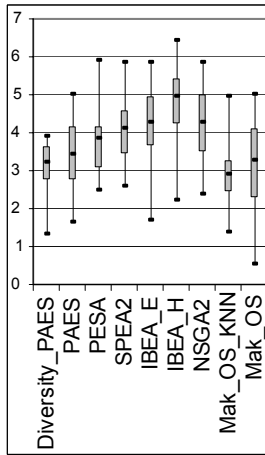
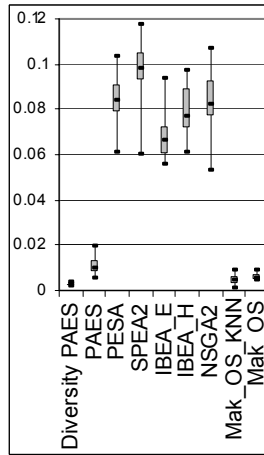
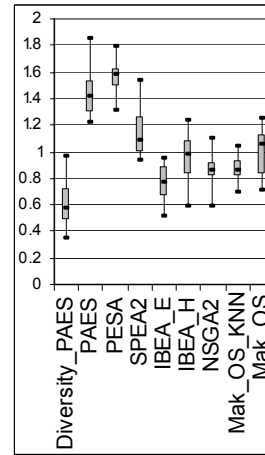
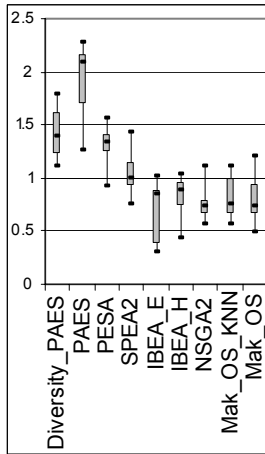
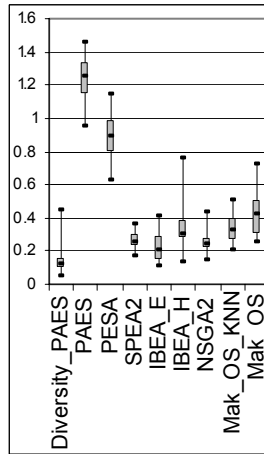
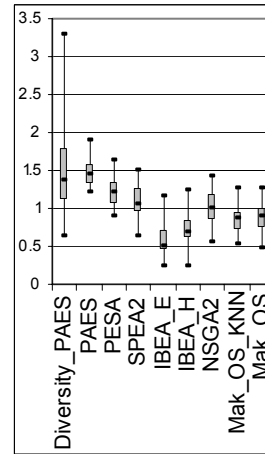


Figure 158 — Progressive Hypervolume Averages

*y-axis*: average hypervolume performance; *x-axis*: number of evaluations executed.

(a) *AP-16*(b) *AP-17*(c) *AP-21***Figure 159 — Progressive Hypervolume Averages**

*y-axis*: average hypervolume performance; *x-axis*: number of evaluations executed.


 (a) *AP-1*

 (b) *AP-2*

 (c) *AP-3*

 (d) *AP-4*

 (e) *AP-5*

 (g) *AP-15*

 (h) *AP-16*

 (i) *AP-17*

 (j) *AP-21*
**Figure 160 — End-of-Run Hypervolume Box-Plots**

*y-axis* is the hypervolume performance at the end of the run (7,000 evaluations in *AP-5*, 10,000 evaluations in all remaining functions); *x-axis* indicates the selected optimiser.

	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	1.66E-08	-	1.45E-14	2.28E-14	5.16E-06	3.09E-03	1.87E-06	1.76E-04	3.77E-01
Mak_OS	1.39E-16	3.09E-03	4.94E-24	8.38E-24	3.52E-13	-	6.06E-02	4.14E-01	3.67E-02
Mak_OS KNN	1.29E-22	1.87E-06	1.34E-30	2.33E-30	8.08E-19	6.06E-02	-	2.87E-01	8.65E-05

(a) AP-1

	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	8.05E-02	-	4.92E-01	5.91E-02	7.46E-01	2.54E-16	1.37E-18	5.37E-07	8.88E-05
Mak_OS	6.79E-22	2.54E-16	1.89E-18	2.30E-22	2.60E-17	-	4.64E-01	4.05E-04	1.55E-29
Mak_OS KNN	2.34E-24	1.37E-18	8.42E-21	7.73E-25	1.28E-19	4.64E-01	-	2.23E-05	3.77E-32

(b) AP-2

	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	5.35E-16	-	8.70E-19	6.56E-16	7.64E-06	1.67E-01	3.40E-04	4.45E-02	4.77E-02
Mak_OS	2.50E-20	1.67E-01	2.57E-23	3.11E-20	8.16E-09	-	2.57E-02	5.26E-01	8.47E-04
Mak_OS KNN	6.33E-28	3.40E-04	4.02E-31	8.03E-28	9.14E-15	2.57E-02	-	1.09E-01	4.66E-08

(c) AP-3

	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	2.43E-01	-	2.58E-02	5.81E-04	1.47E-04	7.32E-01	4.41E-01	3.05E-04	3.12E-08
Mak_OS	4.08E-01	7.32E-01	5.86E-02	1.88E-03	5.28E-04	-	2.66E-01	1.04E-03	1.81E-07
Mak_OS KNN	5.31E-02	4.41E-01	2.83E-03	2.89E-05	5.83E-06	2.66E-01	-	1.36E-05	4.43E-10

(d) AP-4

	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	1.21E-11	-	1.64E-56	5.73E-70	3.93E-55	2.17E-06	6.59E-04	1.45E-35	9.27E-50
Mak_OS	2.63E-02	2.17E-06	8.48E-39	1.00E-52	2.09E-37	-	1.65E-01	7.41E-19	4.51E-32
Mak_OS KNN	3.43E-04	6.59E-04	6.56E-44	8.88E-58	1.64E-42	1.65E-01	-	2.04E-23	4.09E-37

(e) AP-5

Figure 161 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Hypervolume Results  
 Bold italics indicate significant differences at the 5% level.

	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	<b><i>1.13E-34</i></b>	-	<b><i>2.14E-38</i></b>	<b><i>1.53E-21</i></b>	<b><i>5.71E-06</i></b>	<b><i>4.63E-13</i></b>	<b><i>4.48E-06</i></b>	<b><i>1.27E-02</i></b>	<b><i>7.45E-07</i></b>
Mak_OS	<b><i>3.04E-10</i></b>	<b><i>4.63E-13</i></b>	<b><i>5.91E-13</i></b>	<b><i>6.02E-03</i></b>	<b><i>3.29E-03</i></b>	-	<b><i>3.90E-03</i></b>	<b><i>6.80E-07</i></b>	<b><i>1.21E-02</i></b>
Mak_OS_KNN	<b><i>1.41E-18</i></b>	<b><i>4.48E-06</i></b>	<b><i>6.92E-22</i></b>	<b><i>3.36E-08</i></b>	9.57E-01	<b><i>3.90E-03</i></b>	-	<b><i>3.08E-02</i></b>	7.01E-01

(a) AP-15

	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	<b><i>4.71E-02</i></b>	-	5.22E-01	<b><i>3.57E-07</i></b>	<b><i>4.69E-21</i></b>	<b><i>1.95E-19</i></b>	<b><i>1.96E-18</i></b>	<b><i>2.21E-24</i></b>	<b><i>1.45E-14</i></b>
Mak_OS	<b><i>4.26E-26</i></b>	<b><i>1.95E-19</i></b>	<b><i>2.04E-17</i></b>	<b><i>1.01E-05</i></b>	6.18E-01	-	7.53E-01	1.36E-01	1.12E-01
Mak_OS_KNN	<b><i>5.20E-25</i></b>	<b><i>1.96E-18</i></b>	<b><i>1.89E-16</i></b>	<b><i>3.89E-05</i></b>	4.16E-01	7.53E-01	-	7.13E-02	2.03E-01

(b) AP-16

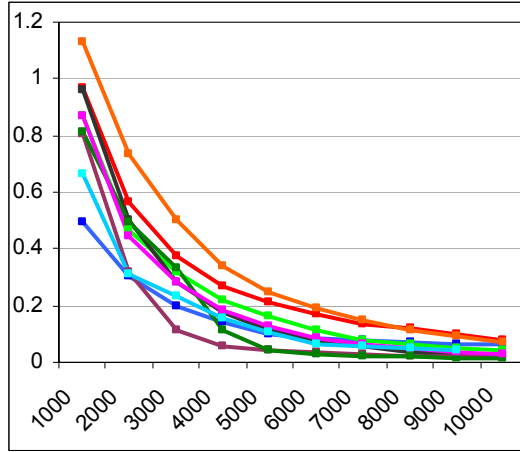
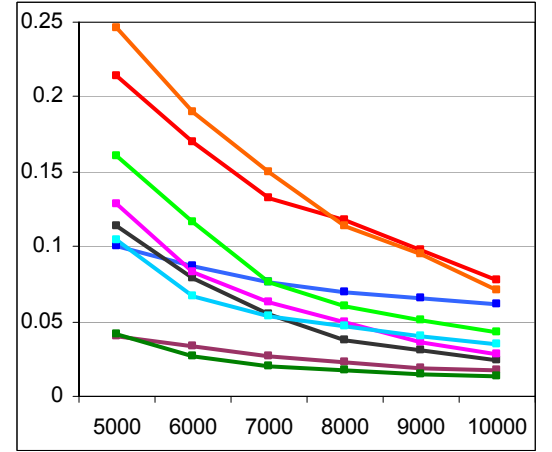
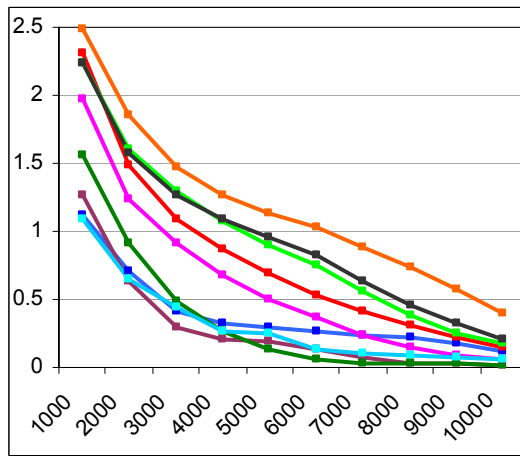
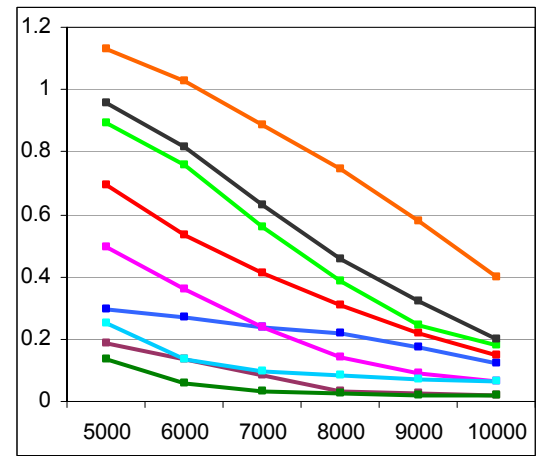
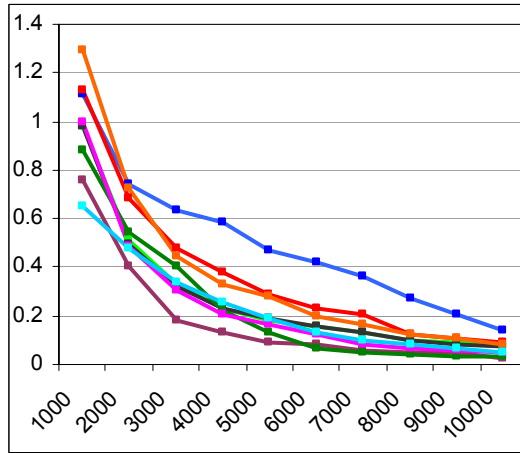
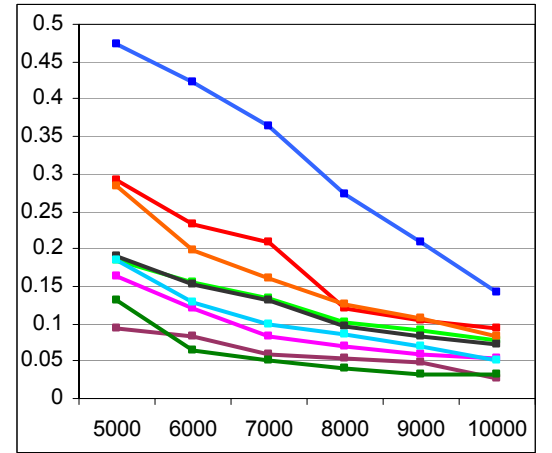
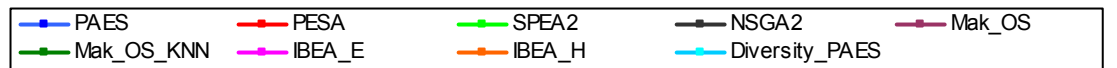
	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	<b><i>6.90E-49</i></b>	-	<b><i>5.39E-41</i></b>	<b><i>3.57E-10</i></b>	<b><i>9.40E-08</i></b>	<b><i>3.61E-24</i></b>	<b><i>9.91E-18</i></b>	<b><i>1.16E-04</i></b>	<b><i>8.91E-18</i></b>
Mak_OS	<b><i>4.45E-11</i></b>	<b><i>3.61E-24</i></b>	<b><i>3.94E-06</i></b>	<b><i>5.33E-06</i></b>	<b><i>3.66E-08</i></b>	-	<b><i>4.95E-02</i></b>	<b><i>4.39E-12</i></b>	5.12E-02
Mak_OS_KNN	<b><i>1.12E-16</i></b>	<b><i>9.91E-18</i></b>	<b><i>1.26E-10</i></b>	<b><i>8.09E-03</i></b>	<b><i>2.70E-04</i></b>	<b><i>4.95E-02</i></b>	-	<b><i>2.83E-07</i></b>	9.88E-01

(c) AP-17

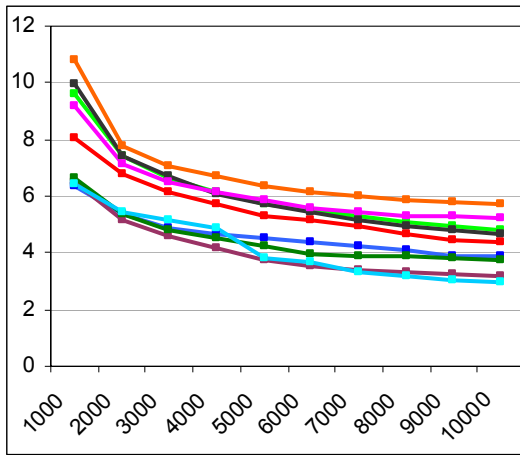
	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	1.55E-01	-	2.05E-01	<b><i>4.33E-03</i></b>	<b><i>5.49E-05</i></b>	<b><i>1.15E-08</i></b>	<b><i>4.23E-10</i></b>	<b><i>1.43E-18</i></b>	<b><i>3.27E-14</i></b>
Mak_OS	<b><i>2.81E-12</i></b>	<b><i>1.15E-08</i></b>	<b><i>6.10E-06</i></b>	<b><i>2.91E-03</i></b>	7.56E-02	-	5.54E-01	<b><i>4.39E-04</i></b>	<b><i>3.53E-02</i></b>
Mak_OS_KNN	<b><i>6.28E-14</i></b>	<b><i>4.23E-10</i></b>	<b><i>3.78E-07</i></b>	<b><i>3.82E-04</i></b>	<b><i>1.82E-02</i></b>	5.54E-01	-	<b><i>3.28E-03</i></b>	1.29E-01

(d) AP-21

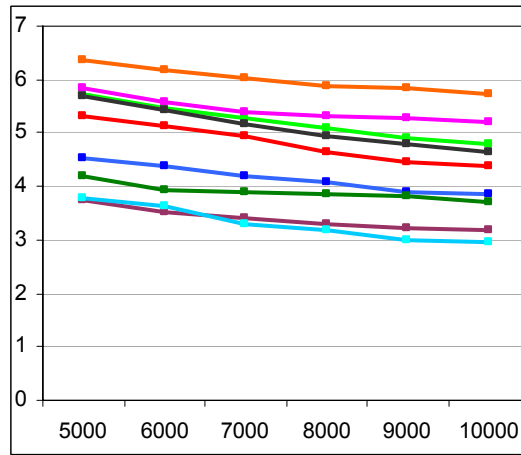
Figure 162 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Hypervolume Results  
Bold italics indicate significant differences at the 5% level.

(a) *AP-1*(b) *AP-1*(c) *AP-2*(d) *AP-2*(e) *AP-3*(f) *AP-3***Figure 163 — Progressive Epsilon Averages**

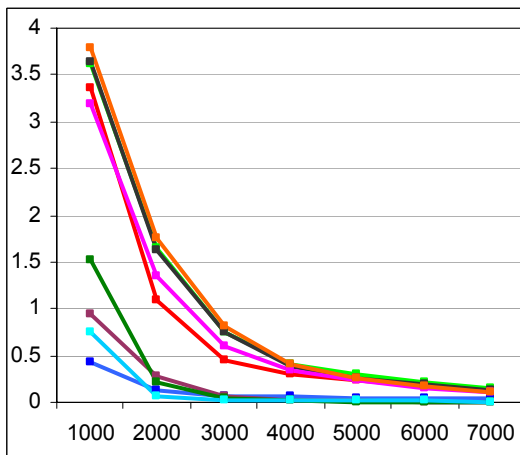
*y-axis*: average epsilon performance; *x-axis*: number of evaluations executed.



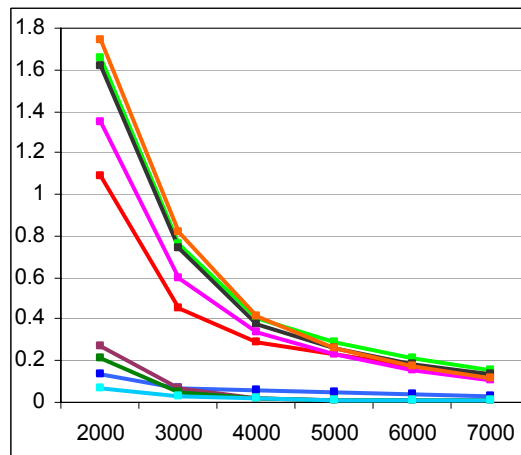
(a) AP-4



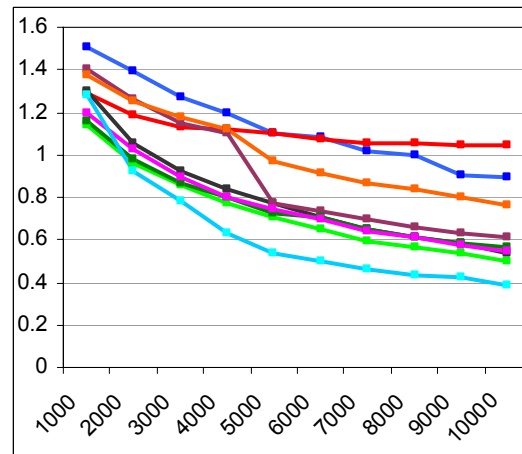
(b) AP-4



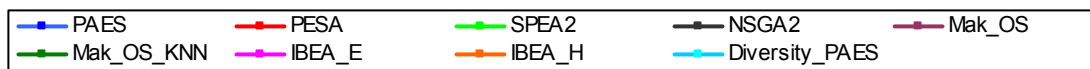
(c) AP-5



(d) AP-5



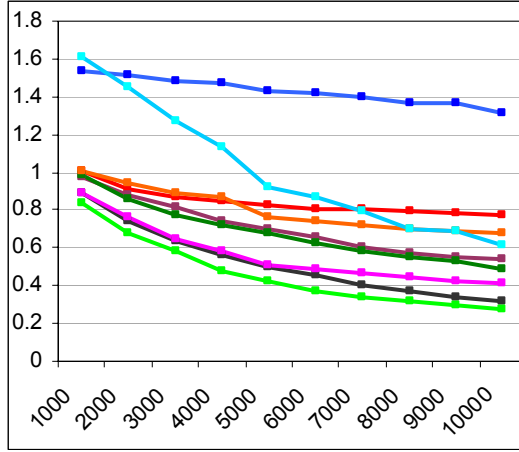
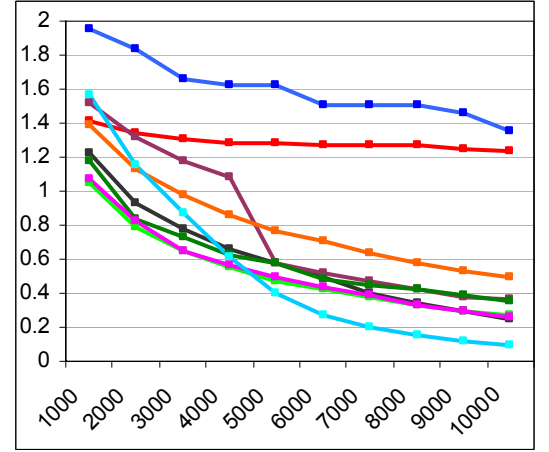
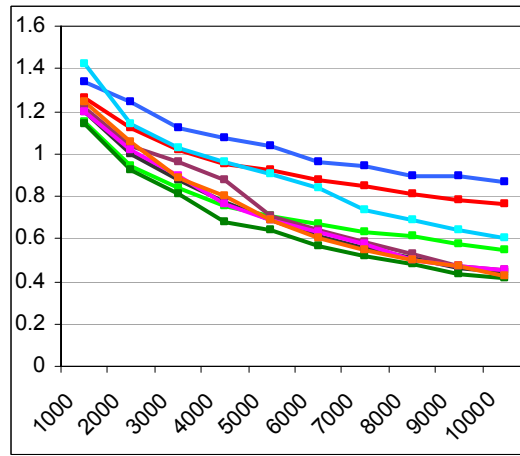
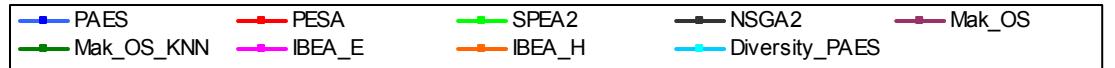
(f) AP-15



**Figure 164 — Progressive Epsilon Averages**

y-axis: average epsilon performance; x-axis: number of evaluations executed.



(a) *AP-16*(b) *AP-17*(c) *AP-21***Figure 165 — Progressive Epsilon Averages**

*y-axis*: average epsilon performance; *x-axis*: number of evaluations executed.

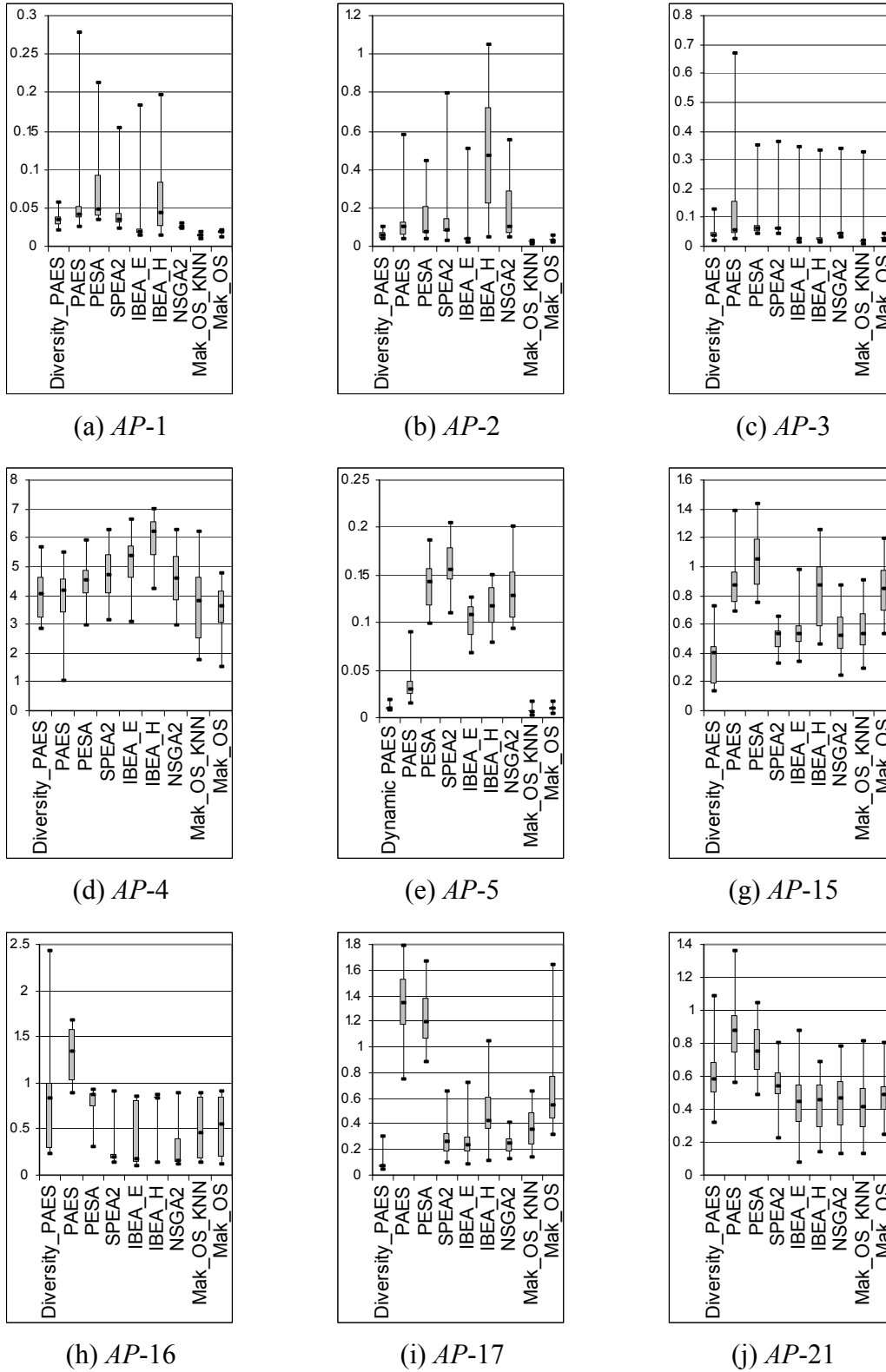


Figure 166 — End-of-Run Epsilon Box-Plots

*y-axis* is the epsilon performance at the end of the run (7,000 evaluations in *AP-5*, 10,000 evaluations in all remaining functions); *x-axis* indicates the selected optimiser. •

	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	<i>2.81E-02</i>	-	<i>1.77E-04</i>	5.51E-01	<i>2.31E-06</i>	<i>1.07E-16</i>	<i>1.96E-20</i>	<i>1.67E-10</i>	5.73E-01
Mak_OS	<i>7.93E-24</i>	<i>1.07E-16</i>	<i>2.38E-29</i>	<i>1.50E-18</i>	<i>7.76E-05</i>	-	2.37E-01	<i>2.85E-02</i>	1.90E-18
Mak_OS KNN	<i>6.48E-28</i>	<i>1.96E-20</i>	<i>1.36E-33</i>	<i>2.13E-22</i>	<i>3.90E-07</i>	2.37E-01	-	<i>8.07E-04</i>	2.72E-22

(a) *AP-1*

	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	<i>5.39E-03</i>	-	<i>1.61E-03</i>	<i>4.01E-04</i>	<i>3.59E-05</i>	<i>2.40E-05</i>	<i>3.34E-08</i>	<i>2.28E-02</i>	<i>5.94E-11</i>
Mak_OS	<i>1.00E-11</i>	<i>2.40E-05</i>	<i>9.30E-13</i>	<i>7.09E-14</i>	<i>1.12E-15</i>	-	1.67E-01	<i>4.58E-02</i>	4.94E-24
Mak_OS KNN	<i>1.21E-15</i>	<i>3.34E-08</i>	<i>8.62E-17</i>	<i>5.07E-18</i>	<i>5.57E-20</i>	1.67E-01	-	<i>7.99E-04</i>	7.94E-29

(b) *AP-2*

	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	<i>4.95E-04</i>	-	<i>6.86E-07</i>	<i>3.82E-07</i>	7.32E-02	<i>3.44E-03</i>	<i>2.86E-06</i>	<i>3.46E-03</i>	3.76E-02
Mak_OS	<i>4.15E-10</i>	<i>3.44E-03</i>	<i>2.59E-14</i>	<i>1.16E-14</i>	<i>3.25E-06</i>	-	6.89E-02	9.98E-01	3.90E-01
Mak_OS KNN	<i>4.21E-15</i>	<i>2.86E-06</i>	<i>6.56E-20</i>	<i>2.68E-20</i>	<i>2.32E-10</i>	6.89E-02	-	6.85E-02	7.62E-03

(c) *AP-3*

	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	8.73E-01	-	1.08E-01	<i>5.02E-03</i>	<i>4.07E-02</i>	1.44E-01	6.57E-01	<i>2.54E-05</i>	2.56E-09
Mak_OS	1.05E-01	1.44E-01	<i>2.28E-03</i>	<i>2.41E-05</i>	<i>4.99E-04</i>	-	3.08E-01	<i>2.33E-08</i>	3.78E-13
Mak_OS KNN	5.46E-01	6.57E-01	<i>4.05E-02</i>	<i>1.20E-03</i>	<i>1.30E-02</i>	3.08E-01	-	<i>3.60E-06</i>	2.02E-10

(d) *AP-4*

	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	<i>1.47E-05</i>	-	<i>1.50E-44</i>	<i>2.59E-54</i>	<i>3.56E-40</i>	5.73E-01	<i>2.35E-02</i>	<i>1.36E-22</i>	3.54E-30
Mak_OS	<i>1.13E-06</i>	5.73E-01	<i>1.26E-46</i>	<i>2.31E-56</i>	<i>3.00E-42</i>	-	8.79E-02	<i>1.69E-24</i>	3.39E-32
Mak_OS KNN	<i>1.20E-10</i>	<i>2.35E-02</i>	<i>6.67E-53</i>	<i>1.63E-62</i>	<i>1.51E-48</i>	8.79E-02	-	<i>1.78E-30</i>	2.06E-38

(e) *AP-5*

Figure 167 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Epsilon Results  
 Bold italics indicate significant differences at the 5% level.

	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	<b><i>7.76E-19</i></b>	-	<b><i>1.22E-24</i></b>	7.38E-02	<b><i>8.93E-03</i></b>	<b><i>3.00E-15</i></b>	<b><i>1.65E-03</i></b>	<b><i>4.11E-03</i></b>	1.52E-14
Mak_OS	2.43E-01	<b><i>3.00E-15</i></b>	<b><i>3.68E-03</i></b>	<b><i>2.59E-10</i></b>	<b><i>2.72E-08</i></b>	-	<b><i>4.36E-07</i></b>	<b><i>1.05E-07</i></b>	8.10E-01
Mak_OS_KNN	<b><i>8.86E-10</i></b>	<b><i>1.65E-03</i></b>	<b><i>1.61E-14</i></b>	1.68E-01	5.86E-01	<b><i>4.36E-07</i></b>	-	7.76E-01	1.39E-06

(a) AP-15

	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	<b><i>4.36E-05</i></b>	-	3.64E-01	<b><i>6.37E-06</i></b>	<b><i>1.00E-07</i></b>	<b><i>3.06E-03</i></b>	<b><i>7.96E-04</i></b>	<b><i>2.35E-07</i></b>	3.40E-01
Mak_OS	<b><i>7.82E-12</i></b>	<b><i>3.06E-03</i></b>	<b><i>1.22E-04</i></b>	1.08E-01	<b><i>1.38E-02</i></b>	-	6.87E-01	<b><i>2.16E-02</i></b>	4.32E-02
Mak_OS_KNN	<b><i>6.21E-13</i></b>	<b><i>7.96E-04</i></b>	<b><i>2.35E-05</i></b>	2.28E-01	<b><i>3.88E-02</i></b>	6.87E-01	-	5.75E-02	1.55E-02

(b) AP-16

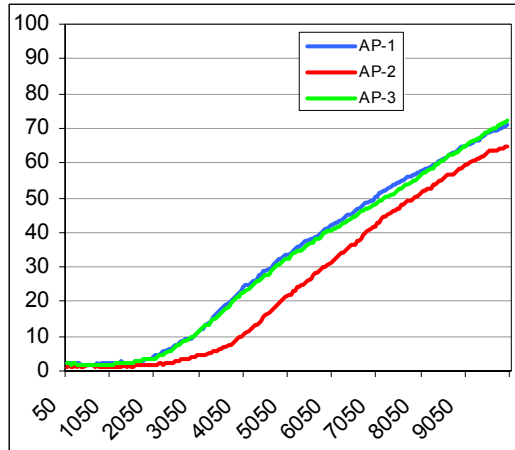
	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	<b><i>2.25E-53</i></b>	-	<b><i>2.11E-51</i></b>	<b><i>4.66E-11</i></b>	<b><i>1.62E-09</i></b>	<b><i>1.39E-33</i></b>	<b><i>2.89E-17</i></b>	<b><i>1.76E-09</i></b>	8.74E-28
Mak_OS	<b><i>1.38E-07</i></b>	<b><i>1.39E-33</i></b>	<b><i>1.90E-06</i></b>	<b><i>2.20E-11</i></b>	<b><i>4.88E-13</i></b>	-	<b><i>2.59E-06</i></b>	<b><i>4.42E-13</i></b>	1.05E-01
Mak_OS_KNN	<b><i>4.89E-21</i></b>	<b><i>2.89E-17</i></b>	<b><i>2.73E-19</i></b>	<b><i>3.07E-02</i></b>	<b><i>5.77E-03</i></b>	<b><i>2.59E-06</i></b>	-	<b><i>5.50E-03</i></b>	1.69E-03

(c) AP-17

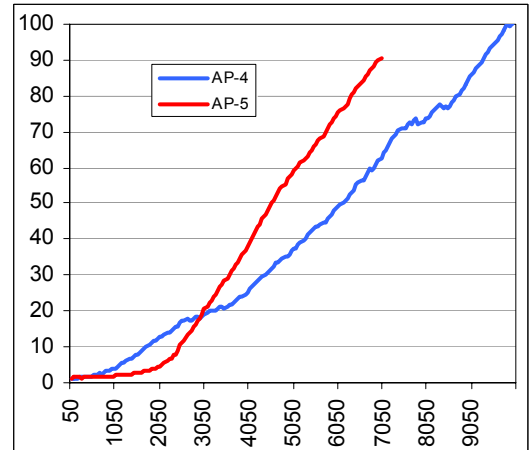
	PAES	Diversity PAES	PESA	SPEA2	NSGA-II	Mak OS	Mak OS_KNN	IBEA_E	IBEA_H
Diversity PAES	<b><i>2.20E-04</i></b>	-	<b><i>8.13E-03</i></b>	3.71E-01	<b><i>1.46E-03</i></b>	<b><i>2.10E-02</i></b>	<b><i>1.19E-04</i></b>	<b><i>3.70E-03</i></b>	7.78E-04
Mak_OS	<b><i>4.18E-09</i></b>	<b><i>2.10E-02</i></b>	<b><i>1.07E-06</i></b>	1.55E-01	3.73E-01	-	1.15E-01	5.45E-01	2.83E-01
Mak_OS_KNN	<b><i>3.20E-13</i></b>	<b><i>1.19E-04</i></b>	<b><i>2.42E-10</i></b>	<b><i>2.88E-03</i></b>	4.92E-01	1.15E-01	-	3.30E-01	6.14E-01

(d) AP-21

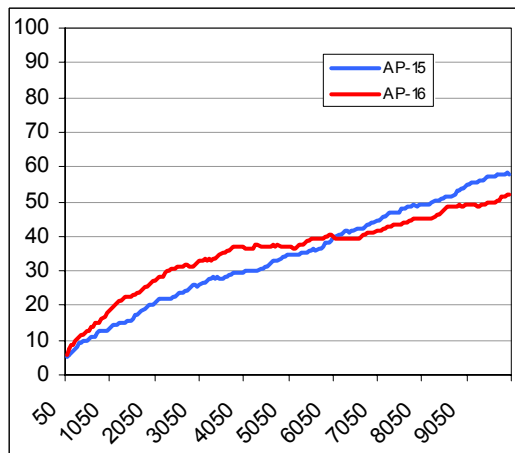
Figure 168 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Epsilon Results  
Bold italics indicate significant differences at the 5% level.



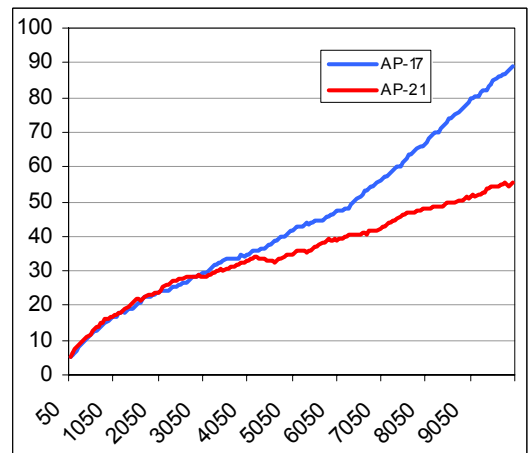
(a)



(b)



(c)



(d)

**Figure 169 — Average Number of Solutions Extracted from the Archive in Mak\_OS\_KNN**  
*y-axis:* number of solutions extracted; *x-axis:* number of evaluations executed.

# Chapter



# 12

## Completing the Analyses

*“There is merit without  
rank, but there is no rank  
without merit.”*

*Francois de la  
Rochefoucauld — Classical  
Author*

## 12 COMPLETING THE ANALYSES

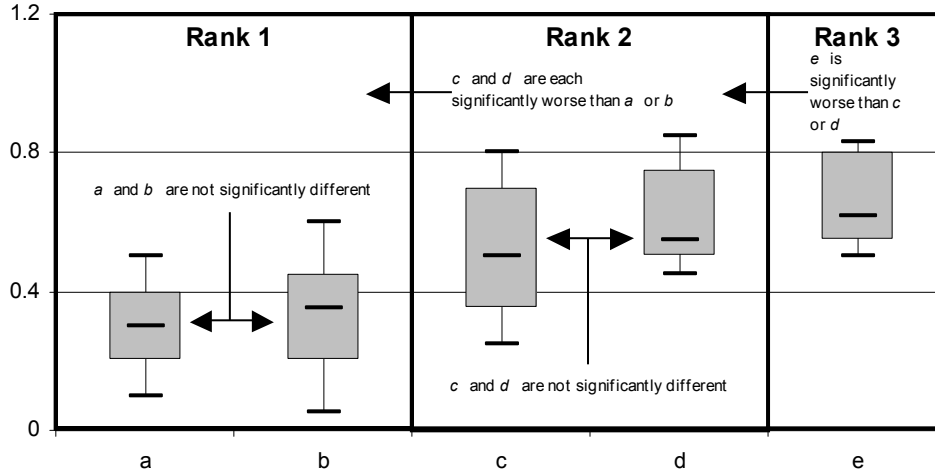
Preceding chapters have offered rich analyses of contemporary truncated evolutionary systems, a range of Mak\_Tree-based extensions and a selection of simple novel techniques. In the interests of clarity, these chapters have been largely isolated from each other, but a number of valuable insights can be made when the complete collection of results are coalesced. By broadening the scope of the investigation, the performance of the unbounded extensions can be placed in a more complete contemporary context; more general comments about the utility of unbounded archiving can be made; and domain-related peculiarities are better elucidated.

### 12.1 EMPIRICAL ANALYSIS METHODOLOGY

As this analysis is principally focussed on the recombination of results from previous sections, the chief methodological concern is not with acquisition, but rather with presentation. Beyond the now familiar progressive averages that endeavour to succinctly describe algorithmic performance on a per-problem basis, this section introduces an indicator-based ranking system that, when merged with simple box-plots, offers a clear differentiation between levels of performance on each problem. Specifically, an optimiser is given rank  $n$  on function  $f$  if it is *significantly* worse than at least one optimiser in rank  $n-1$  on  $f$ , *significantly* better than at least one optimiser in rank  $n+1$  on  $f$ , and never *significantly* better or worse than any member of rank  $n$  on  $f$  (see Equation (64)).

$$\begin{aligned}
 &\text{if } a \in \text{Rank}_n \text{ then} \\
 &\quad \left. \begin{array}{l} \exists c \in \text{Rank}_{n+1} : a \ll b \\ \forall c \in \text{Rank}_n, a \equiv c \end{array} \right\} n = 1 \\
 &\quad \left. \begin{array}{l} \exists b \in \text{Rank}_{n-1} : b \ll a \\ \forall c \in \text{Rank}_n, a \equiv c \end{array} \right\} n = |\text{Rank}| \\
 &\quad \left. \begin{array}{l} \exists b \in \text{Rank}_{n-1} : b \ll a \\ \exists c \in \text{Rank}_{n+1} : a \ll b \\ \forall c \in \text{Rank}_n, a \equiv c \end{array} \right\} \text{otherwise}
 \end{aligned} \tag{86}$$

where  $a \ll b \Rightarrow a$  is *significantly* better than  $b$   
 $a \equiv b \Rightarrow$  no *significant* difference between  $a$  and  $b$



**Figure 170 — Illustrating Ranked Box-Plots**  
*x-axis*: distinct algorithms; *y-axis*: end-of-run indicator score.

Once indicator-based rankings are completed, a powerful visualisation of performance is possible by simply arranging the box-plots such that all optimisers in a given rank appear in a contiguous region, with median indicator scores dictating the ordering of optimisers within each ranking (to offer insight into more fine-grained performance improvements). An example illustration is provided in Figure 170 for reference purposes. A more complete cross-problem analysis can be found by offering a table of normalised rankings for each system.

To further emphasise end-of-run performance, this section also introduces simple combinatorial graphs that examine both median and average indicator-based termination scores. Though it is possible to extract such results from progressive line graphs and box-plots, their combination here makes for a clearer depiction of typical performance, particularly given the large number of systems under observation (fifteen).

## 12.2 EMPIRICAL PERFORMANCE

When examining the results *in toto* (see the collected graphs and tables at the end of this chapter: pages 382–398) there can be little debating the potential power of unbounded archiving in bi-objective optimisation. The ranked box-plots illustrate that an unbounded technique is rank-one for every tested function under epsilon indicators (see Figure 181, Figure 182 and Figure 183) and is similarly ranked on all problems other than *AP-21* with respect to hypervolume (Figure 173, Figure 174 and



Figure 175). Moreover, the set of best-ranked techniques is composed *only* of Mak\_Tree-based systems in *AP-2*, *AP-5*, *AP-15*, *AP-16* and *AP-17* (five of the nine tested functions) according to hypervolume measures and in *AP-1*, *AP-2*, *AP-5* and *AP-17* (four of the nine examined problems) under epsilon indicators. The end-of-run average and median scores support such impressive results — the best performer under epsilon measures (Figure 184, Figure 185 and Figure 186) always operates on an unbounded set, while a truncated performer is preferable only in *AP-21* with respect to hypervolume scores (Figure 176, Figure 177 and Figure 178).

Given the breadth of popular contemporary truncated algorithms under examination, the superiority of unbounded techniques across such a variety of problem-space characteristics and according to such a thorough analysis suggests that unbounded archiving is a valuable tool for generic bi-objective optimisation tasks. There still remains the question though, which is the best of the unbounded techniques offered in this thesis? The answer is largely dependent on the type of problem characteristics encountered.

Considering the early functions, the Mak\_OS\_KNN approach is clearly the best of all examined algorithms. According to epsilon metrics it is rank-one in *AP-1*, *AP-2*, *AP-3*, *AP-4* and *AP-5* (Figure 181 and Figure 182), and is better than all non-Mak\_OS systems on those same functions with respect to both end-of-run averages and medians (Figure 184). Similarly, hypervolume metrics suggest that it is the best performer under average and median scores in *AP-1*, *AP-2*, *AP-3* and *AP-4* (Figure 176), and that it is rank-one in the same functions (Figure 173). This adds further weight to the claim made in Section 11.4.2 that the Mak\_OS-based technique is well suited to simple objective-spaces and those problems with multi-frontal characteristics.

In contrast, the Mak\_NSII and Mak\_SPEA2 techniques perform relatively poorly in the early *AP-1*, *AP-2*, *AP-3*, *AP-4* and *AP-5* functions — failing to achieve rank-one status under either epsilon (Figure 181 and Figure 182) or hypervolume (Figure 173 and Figure 174) metrics — but are impressive in the later test problems. In particular, according to hypervolume measures, Mak\_NSII and Mak\_SPEA2 are the only rank-one systems in *AP-16* (Figure 174); share an optimal rank with Diversity\_PAES in *AP-17* (Figure 175); and are out-ranked only by Diversity\_PAES

in *AP-15* (Figure 174). With respect to epsilon performance (see Figure 181 and Figure 182, in particular), both approaches are rank-one in *AP-21*, with Mak\_SPEA2 gaining rank-one status in *AP-16*, while Mak\_NSQA-II achieves an ideal ranking in *AP-15*. The epsilon-based box-plots also illustrate that Mak\_SPEA2 and Mak\_NSQA-II are impressive in *AP-17*, with only Diversity\_PAES achieving a preferable ranking. Thus, according to at least one indicator, the Mak\_NSQA-II approach achieves a top ranking in *AP-15*, *AP-16*, *AP-17* and *AP-21*, while Mak\_SPEA2 is similarly ranked in all bar *AP-15* (where it is rank-two). Such positive results illustrate that the extended algorithms are capable not only of outperforming their truncated originals, but also a rich set of powerful pre-existing contemporary algorithms.

More erratic results are evident in the Diversity\_PAES system. While achieving rank-one status in *AP-4*, *AP-15* and *AP-17* under both indicators and an ideal ranking for *AP-5* with respect to hypervolume, it is relatively poor under at least one metric in the remaining functions (see Figure 173, Figure 174, Figure 175, Figure 181, Figure 182 and Figure 183). The features that enable improved performance in domains with high levels of non-separability and fitness spaces with steep (valuable) gradients or multiple fronts (namely, its bias towards Pareto optimality and the use of simple asexual reproduction) are hindrances in domains where constant maintenance of a well-distributed frontal set is the key to progression (as in *AP-16*, in particular).

The worst performing of the unbounded approaches are the Mak\_PESA and Mak\_PAES techniques. Neither system achieves rank-one status under hypervolume measures (Figure 173, Figure 174, Figure 175), while the variants are of rank-one according to epsilon indicators only in *AP-4* (Figure 181, Figure 182 and Figure 183). This echoes (though generally improves upon — see Section 10.4) the poor performance of the truncated PAES and PESA systems. Still, it is again worth noting that such unsatisfactory outcomes are perhaps a product of the selected grid resolution and may not be indicative of performance in general (though if this is the case, the results do emphasise the sensitivity of the techniques to the characteristics of the underlying grid and imply the need for parameter tuning). Future work should seek to clarify this.

Considering the truncated systems, results support the claims made in Section 0 that IBEA\_E is the most consistently impressive of the contemporary systems. Under hypervolume metrics (Figure 173, Figure 174, Figure 175), it is rank-one in *AP-1*, *AP-3* and *AP-21*, with competitive, rank-two, performances in *AP-2*, *AP-15*, *AP-16* and *AP-17*. With respect to epsilon indicators (Figure 181, Figure 182 and Figure 183), IBEA\_E achieves the top rank in *AP-3*, and is rank-two in *AP-1*, *AP-2*, *AP-15*, *AP-16* and *AP-21*. Moreover, IBEA\_E is particularly poor only in *AP-4* and *AP-5* under both examined indicators — illustrating a level of consistency across domain features that is lacking in the other truncated systems. Indeed, NSGA-II, according to at least one indicator, features poor rankings in *AP-1*, *AP-2*, *AP-3*, *AP-4*, *AP-5* and *AP-21*; SPEA2 is also unsatisfactory in these functions and in *AP-15* and *AP-16*; PAES is disappointing in all functions other than *AP-4* and *AP-5*; and PESA is similarly poor in all bar *AP-4*.

With IBEA\_E established as the most consistently impressive performer of the truncated techniques and Mak\_OS\_KNN, Mak\_NSII, Mak\_SPEA2 and Diversity\_PAES each illustrating the power of unbounded archiving, it is reasonable to endeavour to elucidate which of these algorithms is preferable for general bi-objective use. In truth, such a goal is impossible to achieve with *any* comparative study, but the richer the function suite under examination, the more accurate that generalised claims are likely to be. Given that the extracted *AP* functions facilitate the examination of performance under differing frontal shapes, bias, isolation, deception, multi-frontality, non-separability and zero-utility gradient spaces, the results can speak to the likely behaviours of the examined systems in such continuous bi-objective domains and should sufficiently explore the flexibility of each system.

An interesting way to explore the relative performance and adaptability of each algorithm is to employ stacked graphs and box-plots (see Figure 175, Figure 178, Figure 183, Figure 186) with data derived from the normalised end-of-run rankings, medians and averages for each examined problem (useful summaries of rankings are also provided in Table 36 and Table 37). Examining these results, Mak\_OS\_KNN is clearly the most flexible of the examined systems: across functions, Mak\_OS\_KNN has the smallest stacked graph under both epsilon and hypervolume metrics for rankings (Figure 175 and Figure 183), averages and medians (Figure 178 and Figure

186); it has better lower and upper quartiles, medians and maximums than all other examined systems with respect to hypervolume (Figure 175) and epsilon (Figure 183) ranks; and has better median and lower quartiles than all other approaches when examining average and median end-of-run hypervolume (Figure 178) and epsilon (Figure 186) box-plots. Thus, with respect to both examined Pareto-compliant indicators, the Mak\_OS\_KNN technique offers the best general performance on the *AP* test suite. By performing very well in *AP-1*, *AP-2*, *AP-3*, *AP-4* and *AP-5*, and remaining competitive in *AP-15*, *AP-16* and *AP-21* (where it achieves rank-two status under at least one metric) it illustrates a level of consistency in performance that is lacking in the other leading techniques. Indeed, while the results indicate that the Mak\_OS\_KNN system is best suited to simple objective-spaces and problems with useful utility gradients or multi-frontal regions, it is sufficiently robust to perform well across a more complete set of domain characteristics as well.

Of the remaining systems, IBEA\_E produces impressive results according to hypervolume indicators, with rank, median and average stacked graphs and box-plots (Figure 175 and Figure 178) illustrating that it features the best general non-Mak\_OS-based performance. Results are less well defined under the epsilon indicators, however — with Mak\_NSQA-II and Mak\_SPEA2 frequently displaying comparable, and often better, suite-wide performances. Indeed, when comparing the rankings of IBEA\_E with these other techniques (Figure 181, Figure 182 and Figure 183), it is bettered by Mak\_SPEA2 on *AP-4*, *AP-16*, *AP-17* and *AP-21* (four of the nine tested functions), and is better only in *AP-1*, *AP-2* and *AP-3*; while it is inferior to Mak\_NSQA-II on *AP-15*, *AP-17* and *AP-21* (three functions) and better only in *AP-2*, *AP-3* and *AP-5*. Diversity\_PAES also features better end-of-run median and average box-plots and stacked graphs than IBEA\_E when examining epsilon indicators (Figure 186). Such differences make it difficult to delineate a sharp ordering of the systems under the epsilon metric (beyond Mak\_OS\_KNN), particularly given the disparate behavioural characteristics of each system. As such, future work in this area is recommended.

## 12.3 CONCLUSIONS

The analysis of the examined systems *en masse* provides a rich context from which general comments can be made about the relative performances of each technique.

The results again emphasise the practical power of using unbounded Mak\_Trees — with the Mak\_OS\_KNN system offering the best general performance across the examined test-suite (due to its impressive performance on simple and multi-frontal domains and competitive outcomes on all functions other than *AP-17*). Additionally, the Mak\_NSGA-II, Mak\_SPEA2\_KNN and, to a lesser extent, Diversity\_PAES techniques offer extremely competitive overall performances, with only the impressive IBEA\_E system appearing preferable (under hypervolume measures). As such, an interesting avenue of future work would be to extend the IBEA\_E system into an unbounded domain with a view to further enhancing the performance of this powerful technique. While such an extension is exciting, it is worth noting that it is not necessarily trivial — the calculation of epsilon scores will need to be modified such that it can operate efficiently in an unbounded set, lest the procedure become prohibitively expensive.

**Table 36 — Normalised Hypervolume Ranks for each Examined Problem**

Green cells indicate a rank-one performance (normalised value of 0); gold cells indicate competitive performance, where the optimiser is no worse than a middle-ranking (normalised value in the range (0,0.5]); orange cells indicate poor performance, where the optimiser is worse than a middle-ranking (normalised value in the range (0.5,1)); red cells indicate the worst rank was achieved (normalised value of 1).

	AP Problem Number									Avg
	1	2	3	4	5	15	16	17	21	
Mak_OS_KNN	0.00	0.00	0.00	0.00	0.13	0.33	0.20	0.60	0.25	0.17
Mak_OS	0.00	0.00	0.25	0.00	0.25	0.50	0.20	0.60	0.25	0.23
IBEA_E	0.00	0.25	0.00	1.00	0.63	0.17	0.20	0.20	0.00	0.27
Diversity_PAES	0.20	0.75	0.25	0.00	0.00	0.00	0.80	0.00	0.75	0.31
Mak_NSGA-II	0.40	0.75	0.50	1.00	0.75	0.17	0.00	0.00	0.25	0.42
Mak_SPEA2	0.60	0.75	0.75	0.50	0.88	0.17	0.00	0.00	0.25	0.43
IBEA_H	0.20	1.00	0.25	1.00	0.75	0.33	0.40	0.60	0.00	0.50
Mak_PESA_C	0.60	0.25	1.00	0.50	0.50	0.67	0.40	0.60	0.25	0.53
Mak_PESA_E	0.80	0.25	0.75	0.50	0.50	0.50	0.60	0.60	0.50	0.56
NSGA-II	0.60	0.75	0.75	1.00	0.75	0.33	0.20	0.20	0.50	0.56
Mak_PAES	0.60	0.50	0.75	0.50	0.13	0.83	0.80	0.80	0.75	0.63
Mak_SPEA2_KNN	1.00	0.75	1.00	1.00	1.00	0.17	0.20	0.20	0.50	0.65
PAES	0.80	0.75	1.00	0.00	0.38	0.83	1.00	1.00	1.00	0.75
SPEA2	1.00	0.75	1.00	1.00	1.00	0.67	0.60	0.40	0.50	0.77
PESA	1.00	0.75	1.00	0.50	0.75	1.00	0.80	0.80	0.75	0.82

**Table 37 — Normalised Epsilon Ranks for each Examined Problem**

Green cells indicate a rank-one performance (normalised value of 0); gold cells indicate competitive performance, where the optimiser is no worse than a middle-ranking (normalised value in the range (0,0.5]); orange cells indicate poor performance, where the optimiser is worse than a middle-ranking (normalised value in the range (0.5,1)); red cells indicate that the worst rank was achieved (normalised value of 1).

	AP Problem Number									Avg
	1	2	3	4	5	15	16	17	21	
Mak_OS_KNN	0.00	0.00	0.00	0.00	0.17	0.25	0.25	0.50	0.00	0.13
Mak_OS	0.00	0.00	0.00	0.00	0.17	0.50	0.25	0.67	0.33	0.21
IBEA_E	0.25	0.25	0.00	0.67	0.83	0.25	0.25	0.33	0.33	0.35
Mak_NSGA-II	0.25	0.75	0.33	0.67	1.00	0.00	0.25	0.17	0.00	0.38
NSGA-II	0.25	0.75	0.67	0.33	0.83	0.25	0.00	0.33	0.00	0.38
Mak_SPEA2	0.50	0.75	0.67	0.33	0.83	0.25	0.00	0.17	0.00	0.39
Diversity_PAES	0.75	0.50	0.67	0.00	0.33	0.00	0.75	0.00	0.67	0.41
Mak_SPEA2_KNN	0.75	0.75	1.00	0.67	1.00	0.25	0.00	0.17	0.33	0.55
Mak_PESA_E	0.75	0.50	0.67	0.00	0.67	0.50	0.50	0.83	0.67	0.56
IBEA_H	0.75	1.00	0.33	1.00	0.50	0.50	0.50	0.67	0.00	0.58
Mak_PESA_C	0.75	0.50	1.00	0.33	0.67	0.50	0.50	0.50	0.67	0.60
SPEA2	0.75	0.75	1.00	0.67	1.00	0.00	0.25	0.33	0.67	0.60
Mak_PAES	0.75	0.50	1.00	0.00	0.50	0.75	1.00	1.00	1.00	0.72
PAES	1.00	0.75	1.00	0.00	0.50	0.75	1.00	1.00	1.00	0.78
PESA	1.00	0.75	1.00	0.33	0.83	1.00	0.75	1.00	1.00	0.85

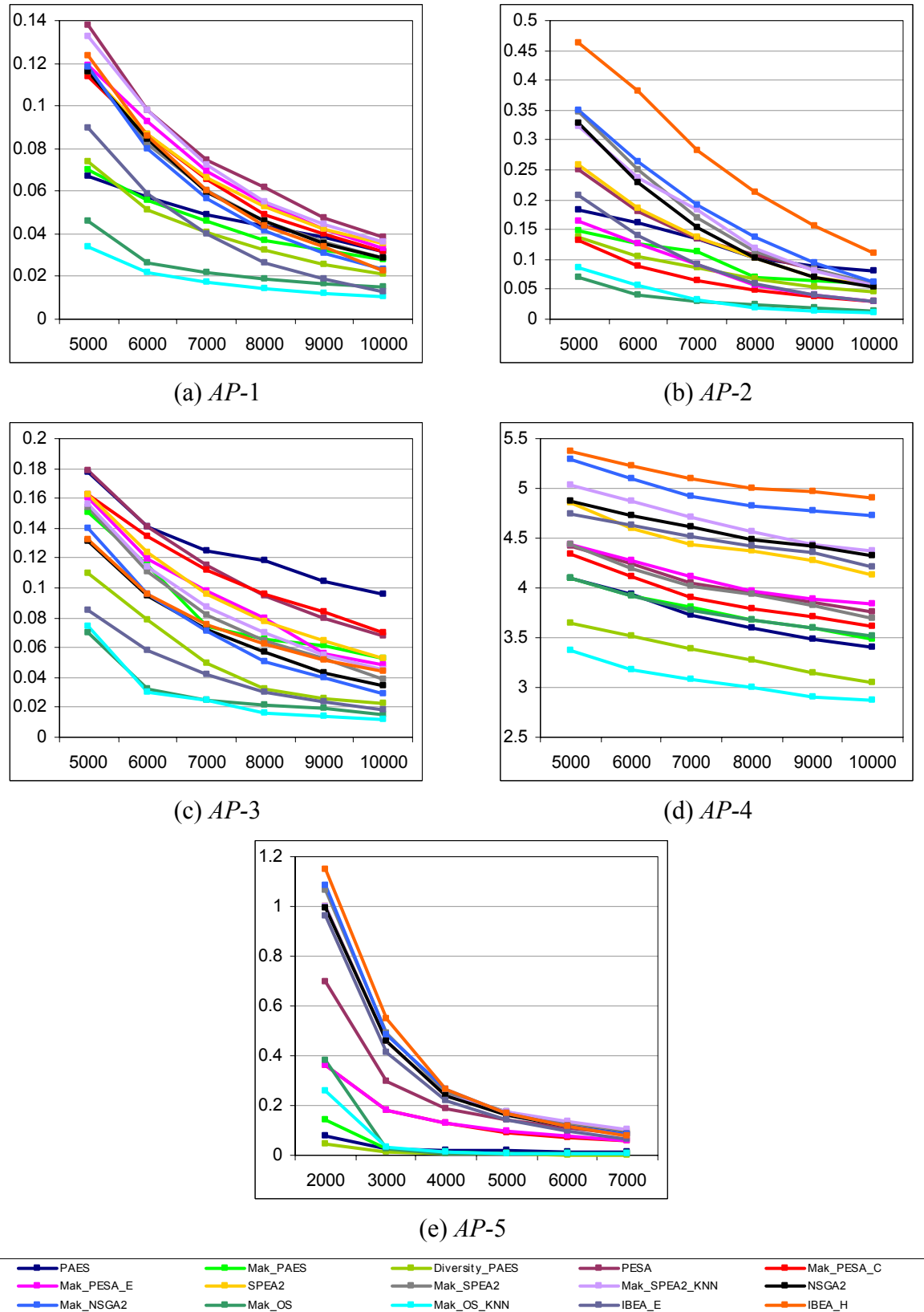
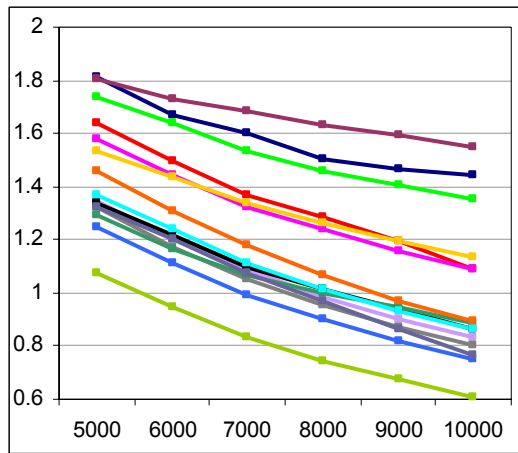
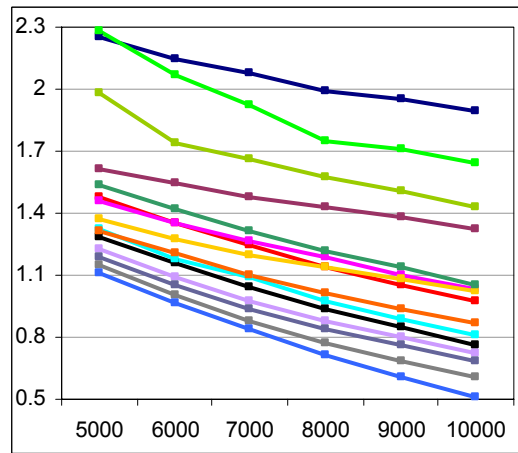


Figure 171 — Progressive Hypervolume Averages

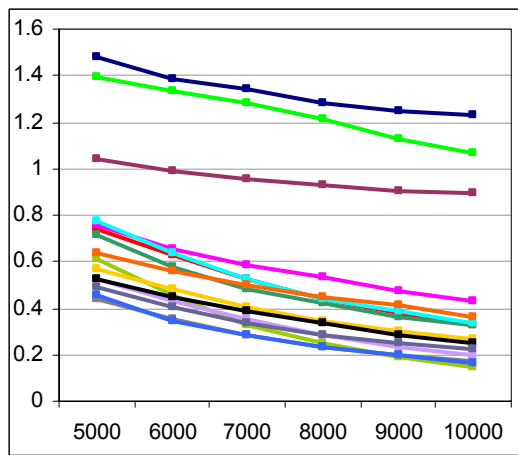
*y-axis*: average hypervolume performance; *x-axis*: number of evaluations executed.



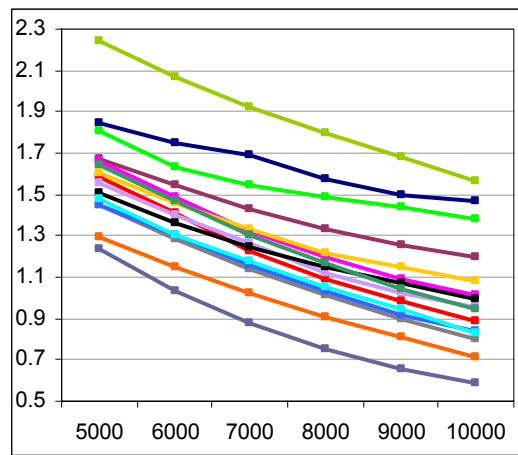
(a) AP-15



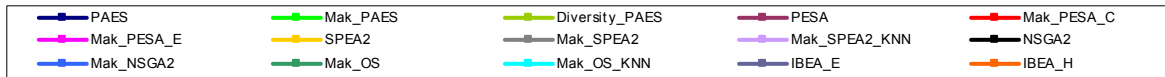
(b) AP-16



(c) AP-17



(d) AP-21



**Figure 172 — Progressive Hypervolume Averages**

*y-axis*: average hypervolume performance; *x-axis*: number of evaluations executed.



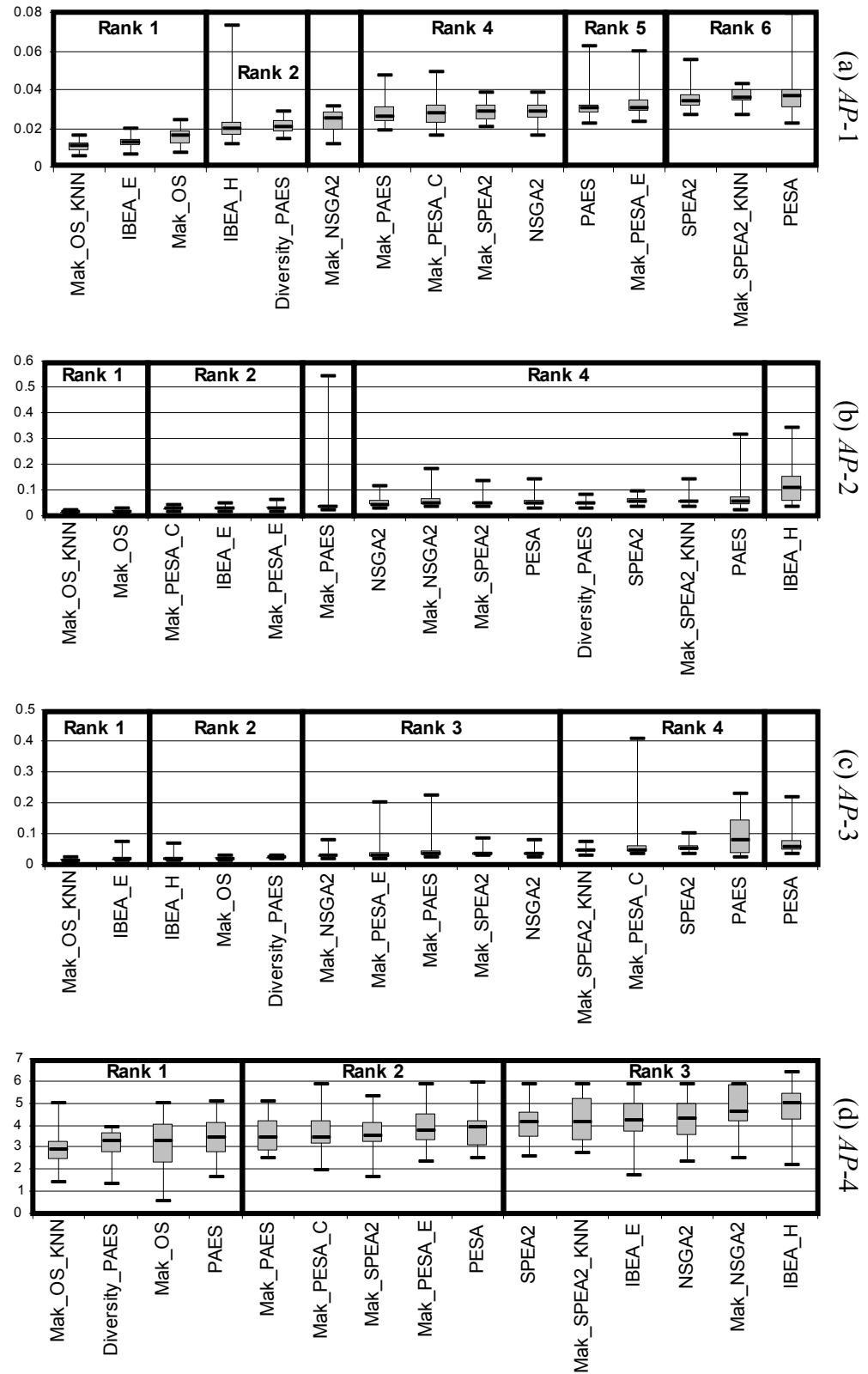


Figure 173 — End-of-Run Hypervolume Box Plots with Statistical Ranks

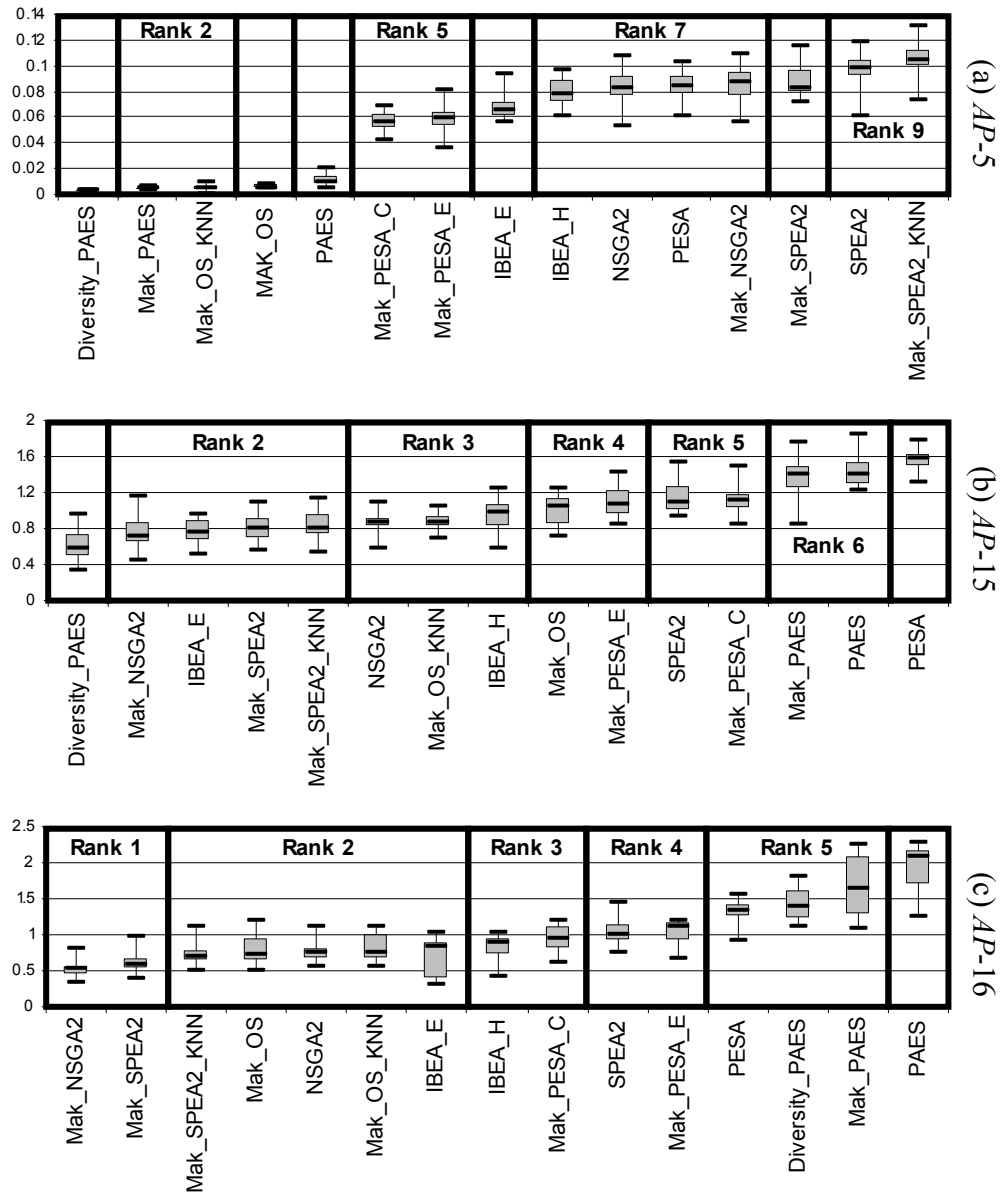


Figure 174 — End-of-Run Hypervolume Box Plots with Statistical Ranks

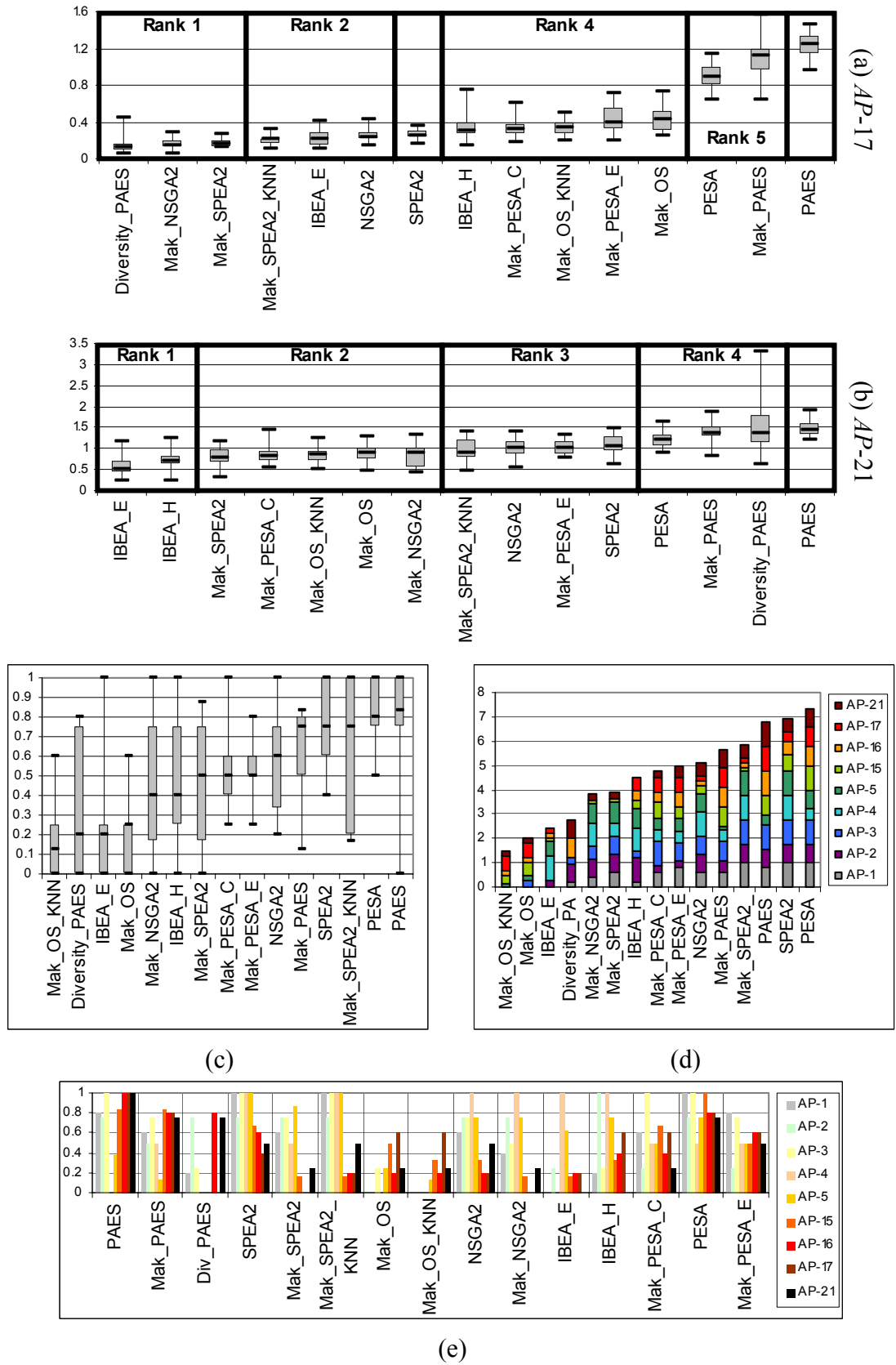
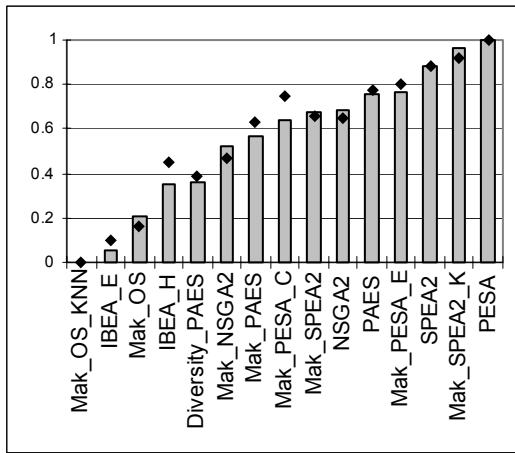
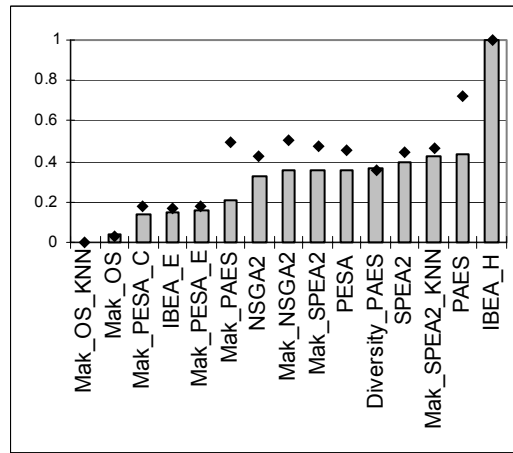


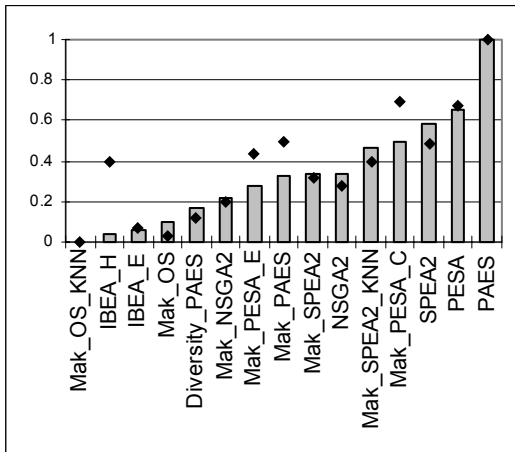
Figure 175 — End-of-Run Hypervolume Box-Plots with Statistical Ranks and Summary Hypervolume Graphs (Normalised)



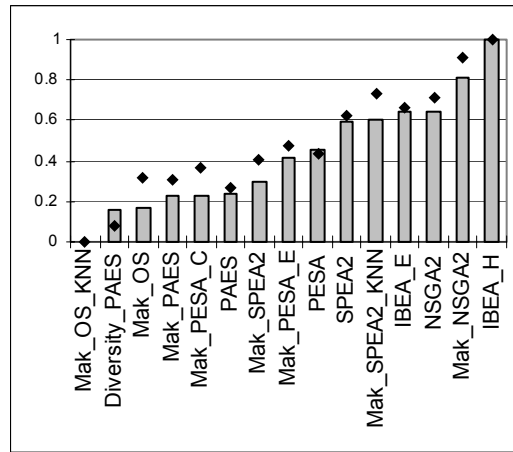
(a) AP-1



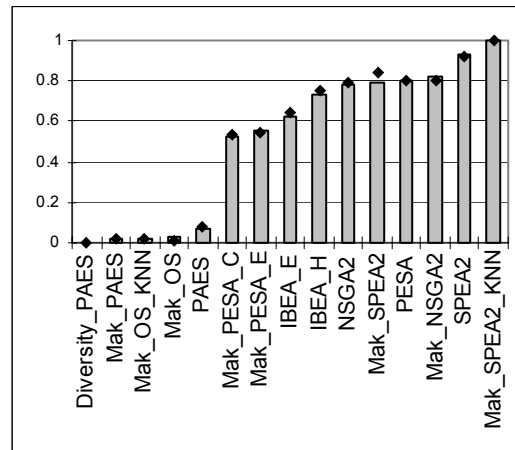
(b) AP-2



(c) AP-3



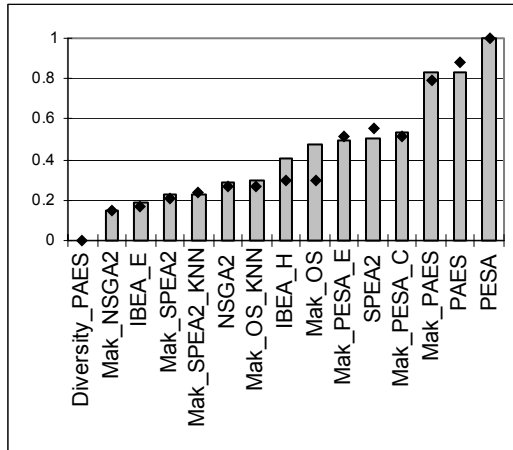
(d) AP-4



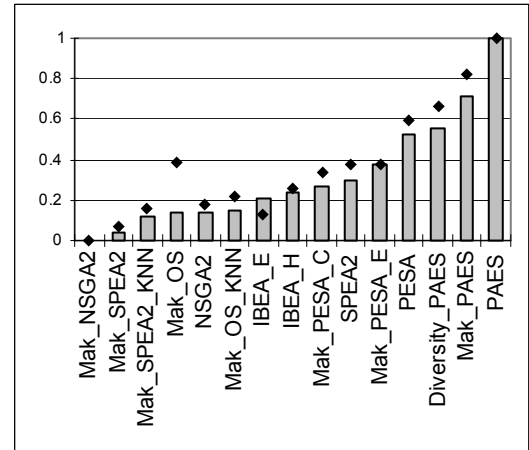
(e) AP-5

**Figure 176 — End-of-Run Hypervolume Medians and Averages**

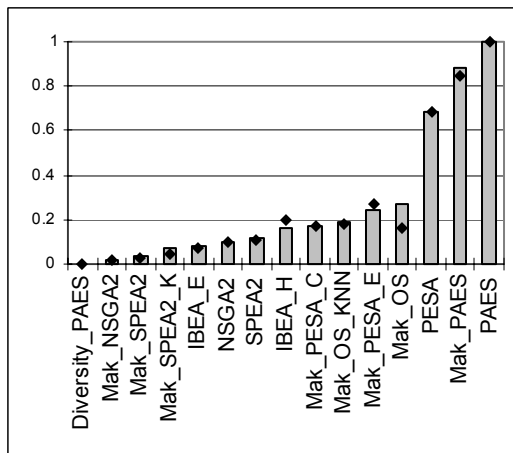
y-axis: normalised hypervolume scores; x-axis: optimiser. Columns represent medians, points represent averages.



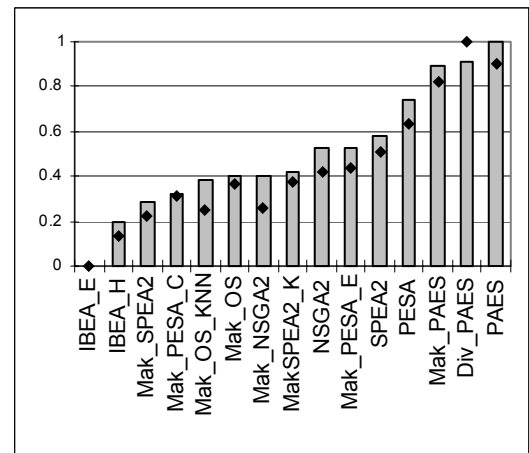
(a) AP-15



(b) AP-16



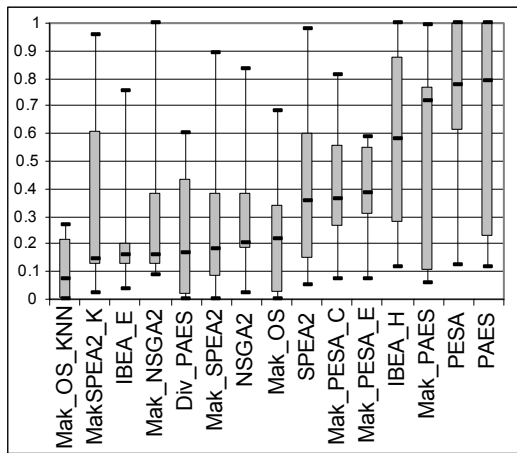
(c) AP-17



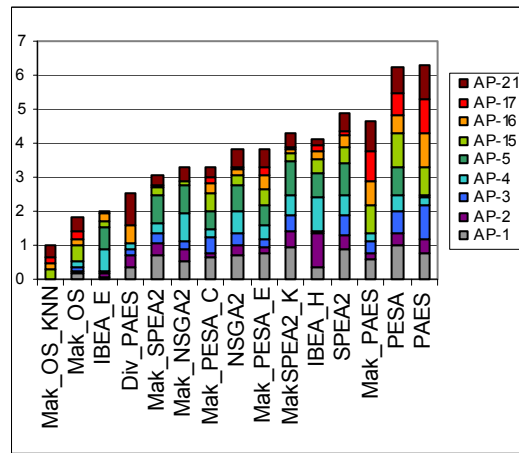
(d) AP-21

**Figure 177 — End-of-Run Hypervolume Medians and Averages**

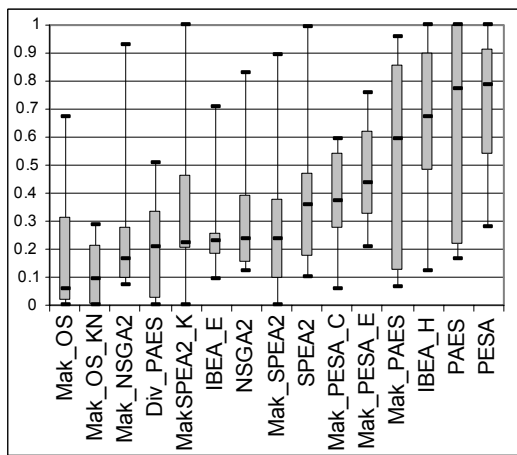
*y-axis*: normalised hypervolume scores; *x-axis*: optimiser. Columns represent medians, points represent averages.



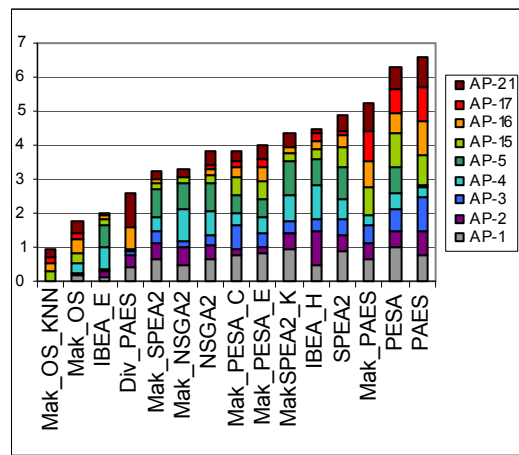
(a) Median Performance Box-Plots



(b) Median Performance Stacked Graphs



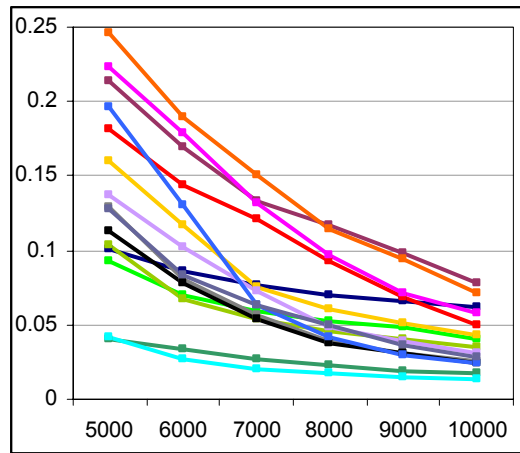
(c) Average Performance Box-Plots



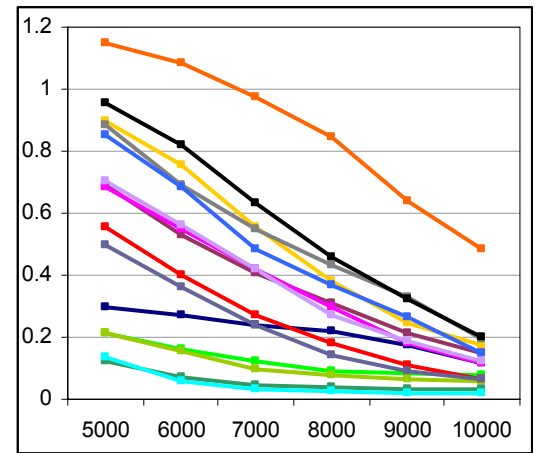
(d) Average Performance Stacked Graphs

**Figure 178 — End-of-Run Hypervolume Medians and Averages**

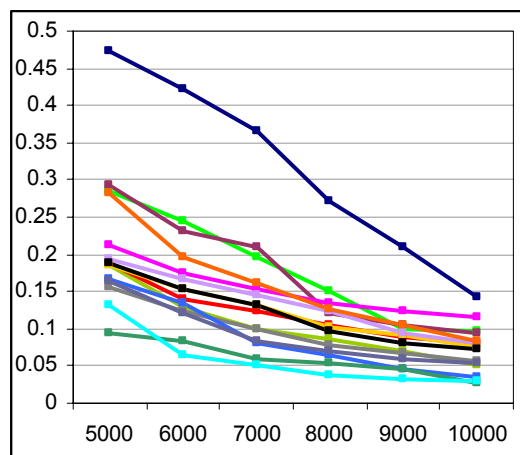
*y-axis*: normalised hypervolume scores; *x-axis*: optimiser. Results reflect performance over all examined test functions.



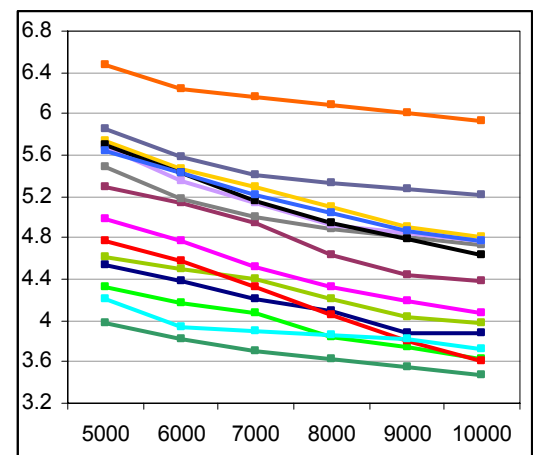
(a) AP-1



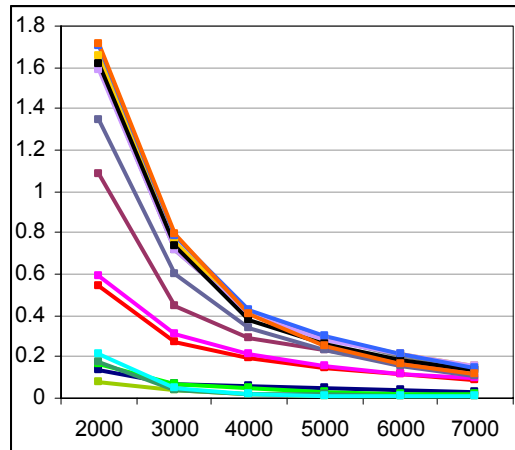
(b) AP-2



(c) AP-3



(d) AP-4



(e) AP-5

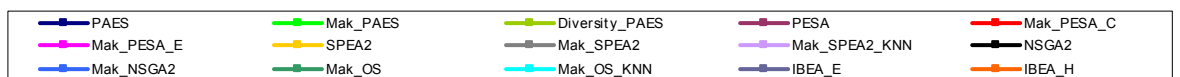
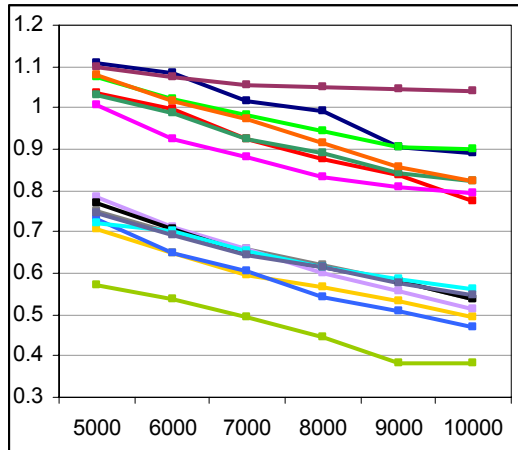
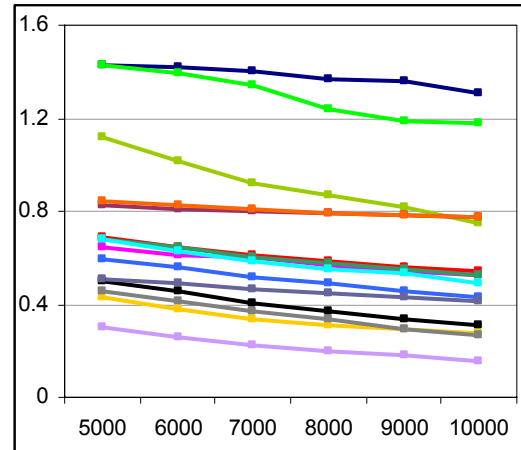


Figure 179 — Progressive Epsilon Averages

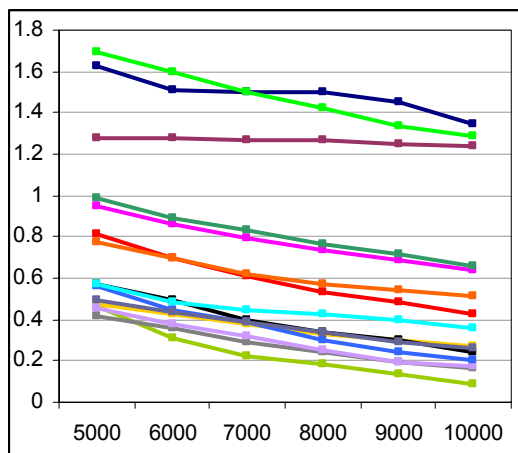
*y-axis*: average epsilon performance; *x-axis*: number of evaluations executed.



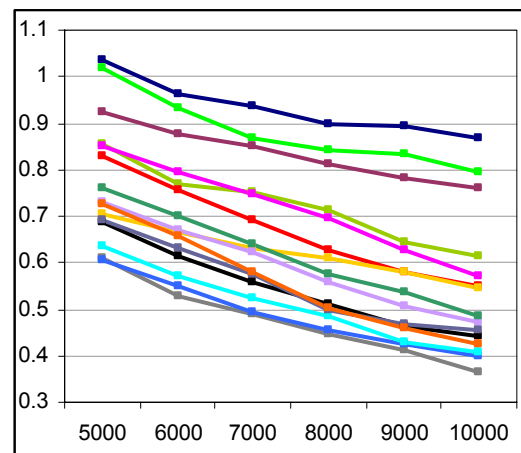
(a) *AP*-15



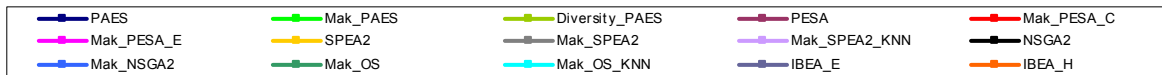
(b) *AP*-16



(c) *AP-17*



(d) *AP-21*



### Figure 180 — Progressive Epsilon Averages

*y*-axis: average epsilon performance; *x*-axis: number of evaluations executed.



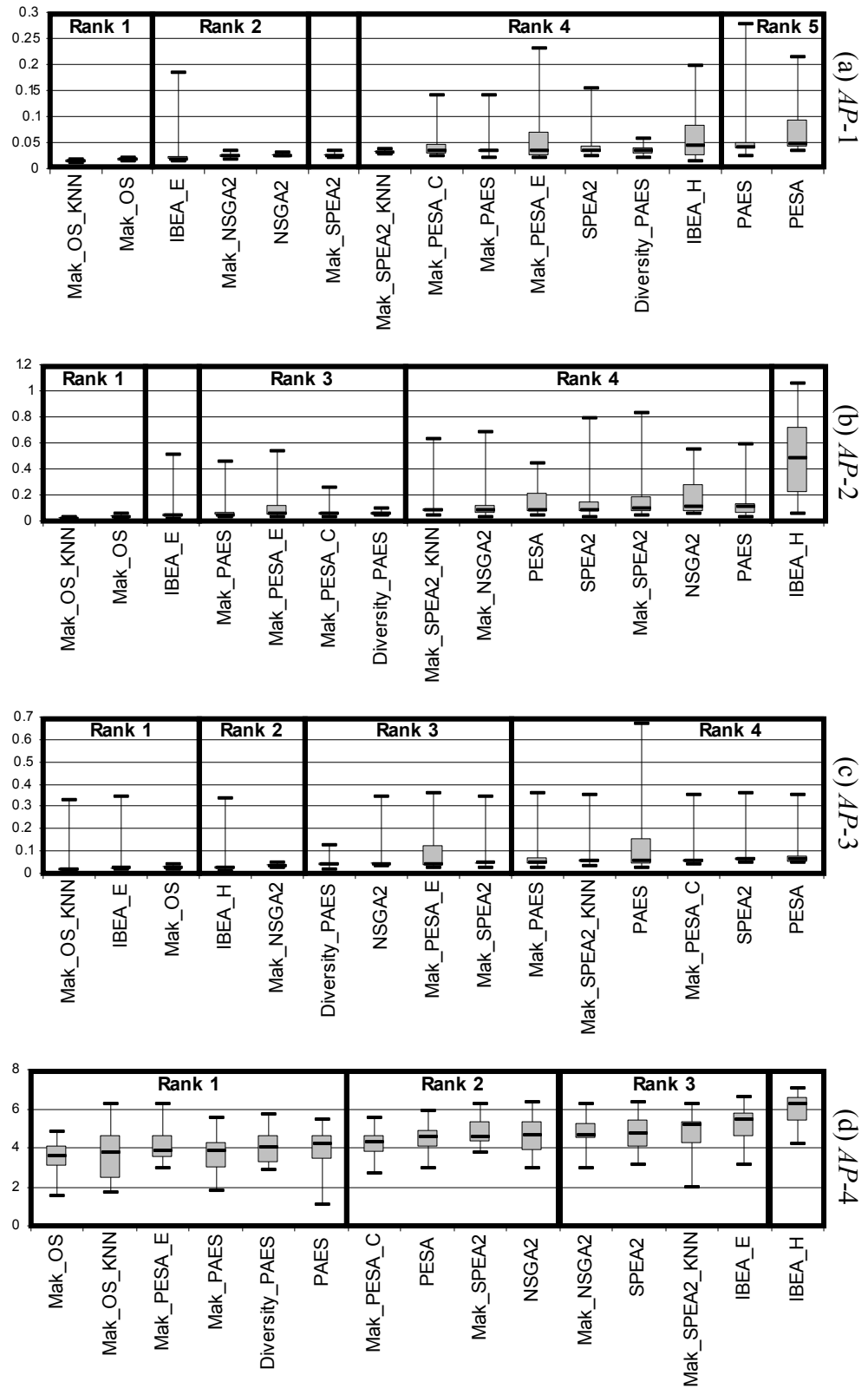


Figure 181 — End-of-Run Epsilon Box Plots with Statistical Ranks

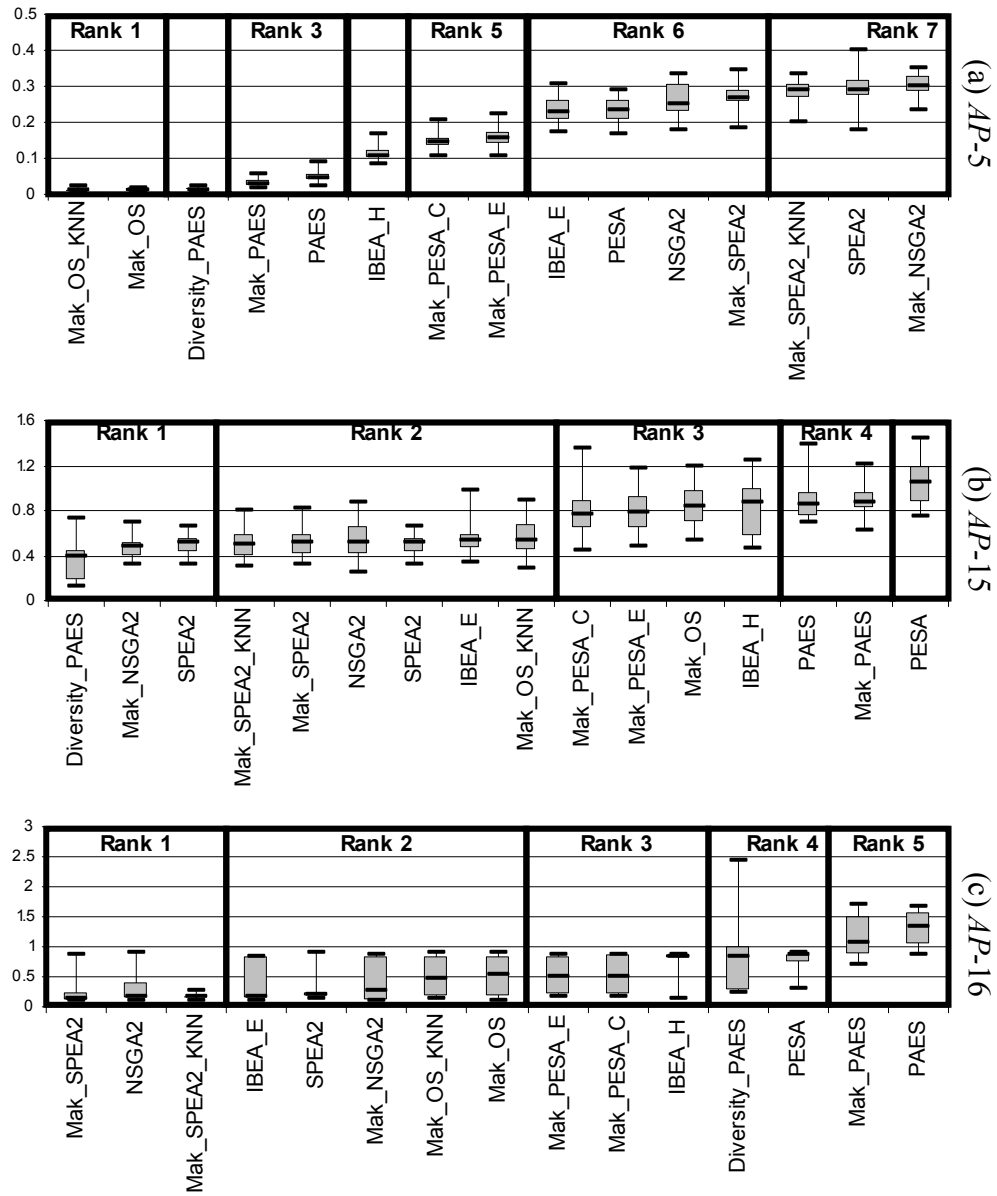


Figure 182 — End-of-Run Epsilon Box Plots with Statistical Ranks

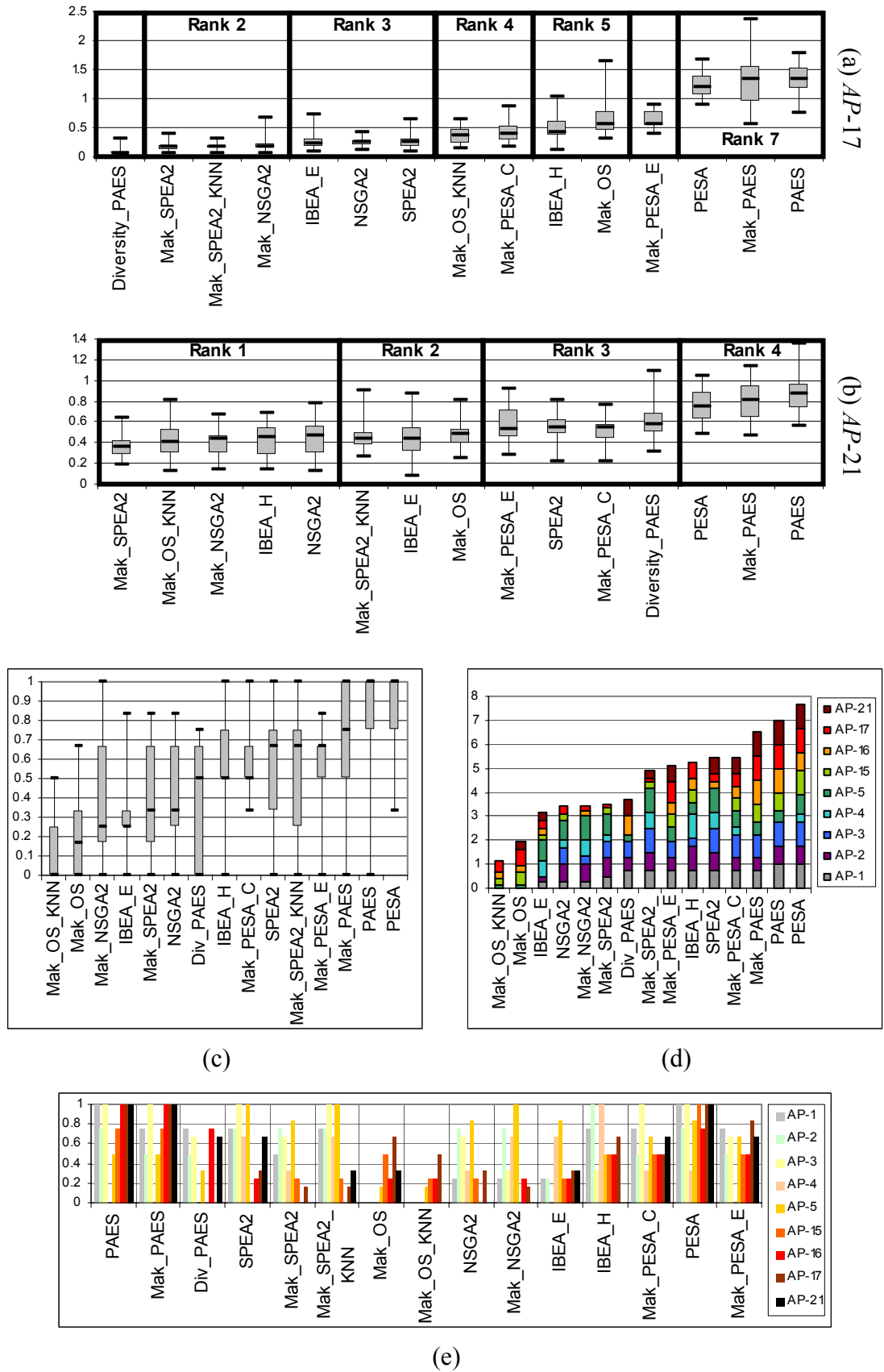
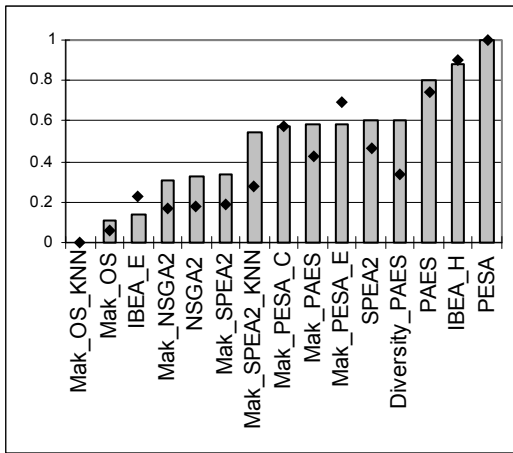
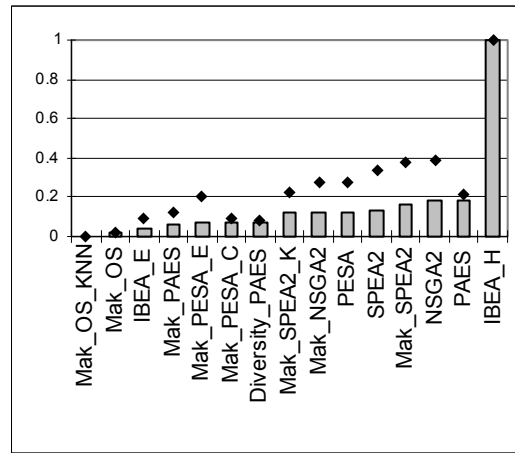


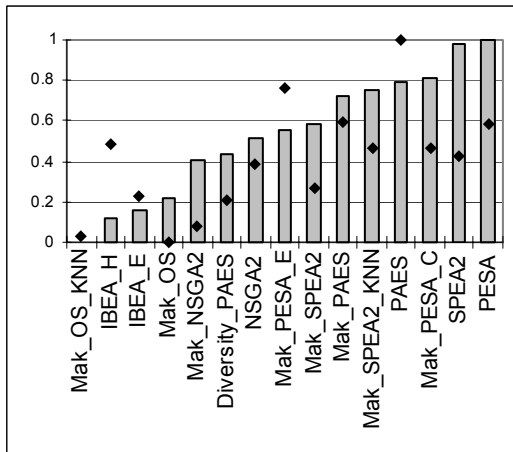
Figure 183 — End-of-Run Epsilon Box Plots with Statistical Ranks ( $AP-17$ ,  $AP-21$ ) and Summary Epsilon Graphs (Normalised)



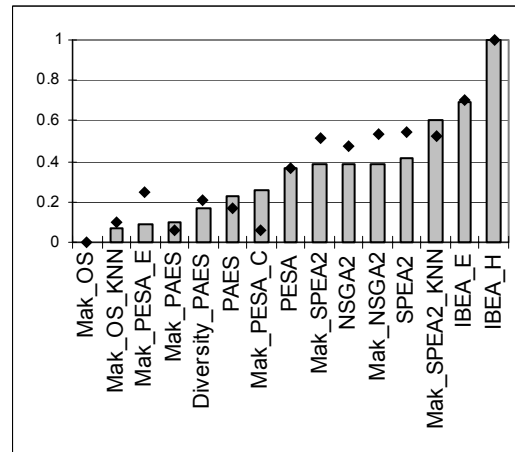
(a) AP-1



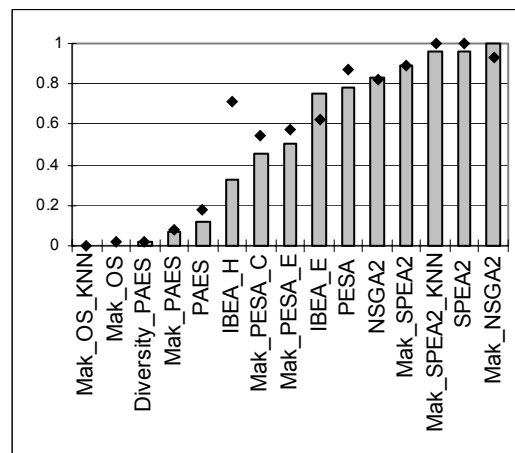
(b) AP-2



(c) AP-3



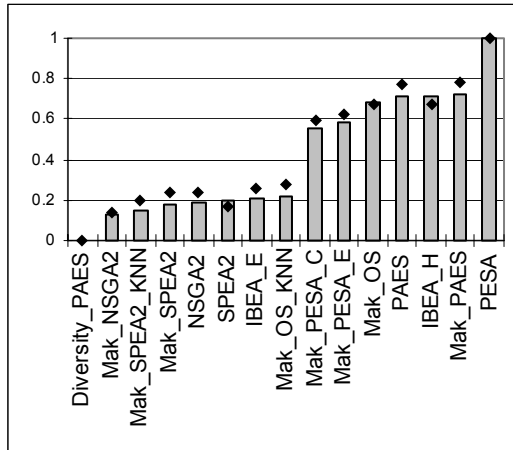
(d) AP-4



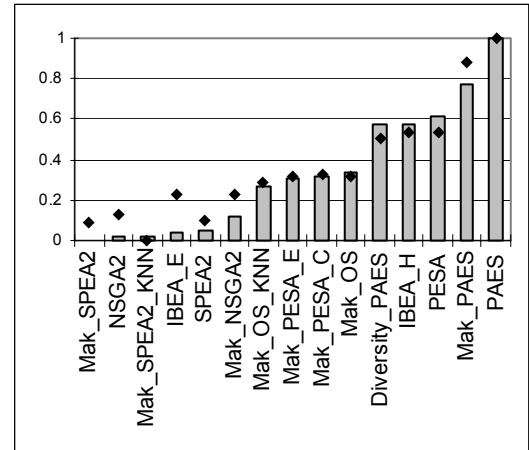
(e) AP-5

**Figure 184 — End-of-Run Epsilon Medians and Averages**

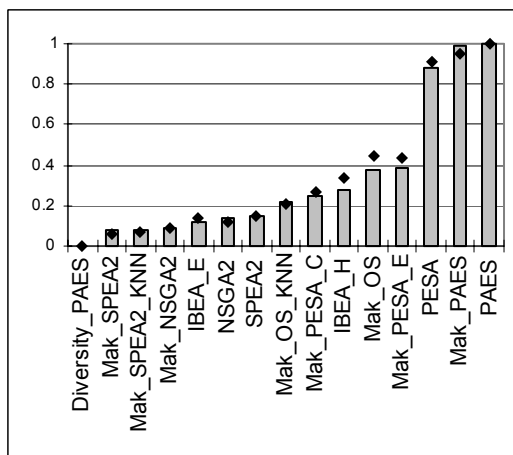
*y-axis*: normalised epsilon scores; *x-axis*: optimiser. Columns represent medians, points represent averages.



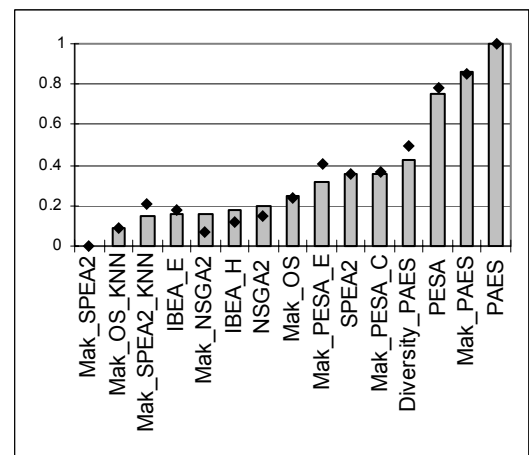
(a) AP-15



(b) AP-16



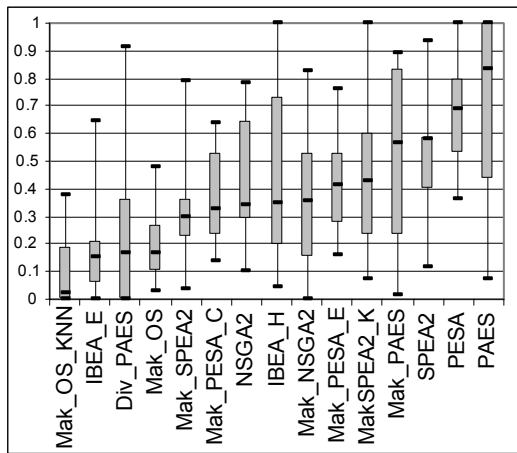
(c) AP-17



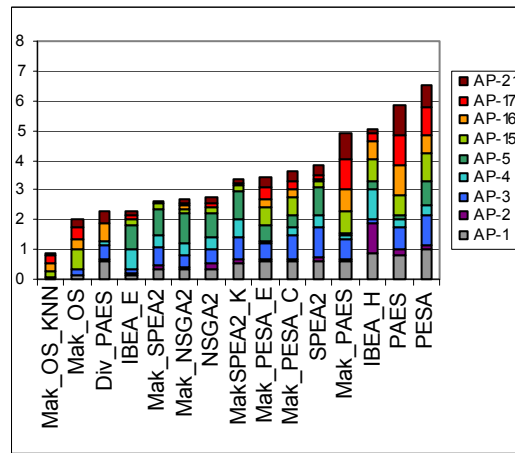
(d) AP-21

**Figure 185 — End-of-Run Epsilon Medians and Averages**

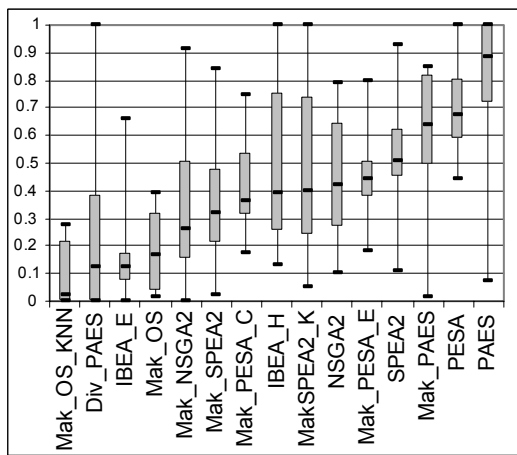
*y-axis*: normalised epsilon scores; *x-axis*: optimiser. Columns represent medians, points represent averages.



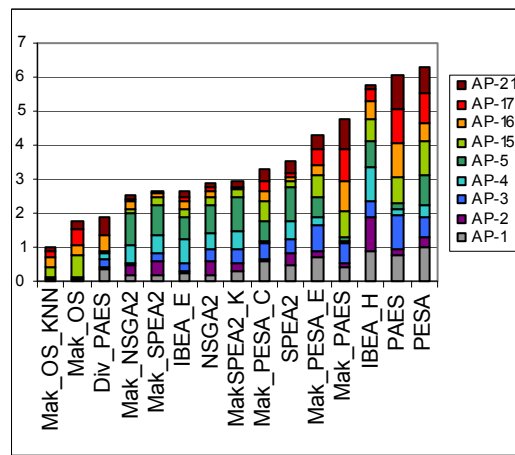
(a) Median Performance Box-Plots



(b) Median Performance Stacked Graphs



(c) Average Performance Box-Plots



(d) Average Performance Stacked Graphs

**Figure 186 — End-of-Run Epsilon Medians and Averages**

*y-axis*: normalised epsilon scores; *x-axis*: optimiser. Results reflect performance over all examined test functions.

# Chapter



# 13

## Conclusions and Future Work

*“Reasoning draws a conclusion, but does not make the conclusion certain, unless the mind discovers it by the path of experience.”*

*Roger Bacon – Philosopher*

*“A conclusion is the place where you got tired of thinking.”*

*Doctor Martin Henry Fischer – Author*

*“My interest is in the future because I am going to spend the rest of my life there.”*

*Charles F Kettering – Engineer*

## 13 CONCLUSIONS AND FUTURE WORK

### 13.1 CONCLUSIONS

A review of the literature, theoretical discussions and empirical evidence all suggest that the truncated archiving techniques that have become such a powerful standard in contemporary multiobjective evolutionary algorithms are subject to serious deficiencies. Given that the archive is intended to provide a reference to the best solutions produced thus far, the potential for frontal degradation and the loss of valuable solutions is of extreme detriment to its function, while inaccurate crowding estimates inhibit the pursuit of ill-explored spaces. By defining and then exploiting the characteristics of the non-dominated bi-objective set, this work offers specialist approaches that ameliorate such deficiencies by unlocking access to unbounded elite stores.

The basic unbounded bi-objective Mak\_Tree developed in this thesis emphasises the power that such specialisation can impart. Indeed, the new construct offers markedly improved run-times over pre-existing generalist approaches to unbounded archiving from the perspective of both big-oh theoretical estimates and empirical observations across a diverse set of problems. Moreover, the Mak\_Tree produces competitive, and frequently preferable, empirical run-time performances when compared against even a tightly bound truncated methodology ( $n = 50$ ). Such results are both important and impressive. Using the bi-objective specialisation frees an optimiser of the need for artificial binding, and it does so with no (or very little) practical efficiency overhead.

Perhaps more significantly, extending the Mak\_Tree to incorporate crowding estimates, annotations and selections — pivotal provisions for use with contemporary optimisers — results in an archiving strategy that is *faster* than strictly bound ( $n = 50$ ) systems with equivalent crowding mechanisms. Efficient extensions into cell-based Mak\_Trees also facilitate multi-faceted crowding analyses which may diminish the bias/variance dilemma.

Still, having access to the specialised bi-objective data structure means little if the removal of archival bounds imparts no practical advantage for optimisers. As such, via the integration of the extended Mak\_Tree into existing techniques, the



development of novel algorithms and the comprehensive analysis of these approaches and their truncated contemporaries, this thesis has offered a rich investigation of bi-objective optimisation in continuous domains that clearly reflects the power of unbounded optimisers. Whether used as a reference set, as the active population or in a hybridised setting, the unbounded elite set frequently produces *significant* improvements over the truncated originals. Moreover, such power is not limited merely to extensions. The development of simple novel algorithms built around the Mak\_Tree offers particularly impressive results. The Mak\_OS\_KNN approach, in particular, capitalises on the flexibility of the Mak\_Tree to deliver an adaptable, noisy, system that achieves the most robust and consistent performance of any of the examined techniques — bounded or otherwise. Given the depth of the analysis, with respect to algorithms, test functions, performance indicators and statistical inferences, such results are a strong statement as to the damaging effects of artificial truncation and the utility of the new unbounded techniques proposed.

The thesis also offers an alternative use for the Mak\_Tree as an offline repository of solutions which is appropriate for background use in any type of optimiser. Having efficient and continuous access to such a store of elite solutions provides an important foundation for the development of effective, intuitive, simple and autonomous run-termination criteria. Indeed, the existence of high-fidelity criteria is contingent on the guarantee of set quality offered only in unbounded stores, and the proposed Mak\_Terminator is amongst the first fully-realised termination systems that efficiently harnesses such a repository.

Access to a store with guaranteed quality also enables the production of better presentation sets (distillations of  $P_{estimate}$ ) than is possible when relying only on truncated approximations. Moreover, the developed Mak\_Presentation algorithm produces reduced sets that are *significantly* better distributed (in objective-space) than alternative techniques operating on both bounded and unbounded archives.

Beyond elite archiving in particular, results illustrate a number of domain-related properties that may be important for multiobjective optimisation in general. In particular, multi-frontality — which rests as amongst the most difficult domain features in multiobjective optimisation — appears to be best addressed by narrow-search systems (such as hill-climbers), which are better able to pursue solutions that

puncture the front, and noisy systems, which encourage a more thorough search along the false front or away from the front. A similarly complex characteristic is non-separability, with improved performances offered by systems emphasising the importance of diversity — likely because other approaches can become fixated on a particular subset of gene dependencies. Interestingly, initial results also suggest that asexual reproduction is beneficial in such domains, largely because sexual crossover is prone to the disruption of gene dependencies and is more likely to introduce collateral noise. Zero-utility gradient spaces and isolated deceptive regions pose few problems for effective multi-member systems, but can cause difficulties for small hill-climbing systems, where smaller populations simply lack the number of escape routes offered by more diverse sets. The least problematic feature tends to be frontal shape, so long as the system lacks a strong bias towards convexity or concavity.

A number of interesting observations can also be drawn from the comprehensive analysis of today's leading core truncated evolutionary systems. The results suggest that while the IBEA algorithm is clearly impressive (IBEA\_E offers the best general performance of the examined truncated systems) it is sensitive to the selection of the underlying indicator. It is for this reason that there is such a great disparity between IBEA\_E and IBEA\_H — the hypervolume indicator, potentially due to its biases, is less effective in promoting a balanced search. The remaining optimisers offer insight into the importance of the crowding estimation mechanisms. For truncated systems, it appears that the cuboid technique seen in NSGA-II is preferable to both the  $\kappa^{\text{th}}$  neighbour and cell-based approaches seen in SPEA2, PAES and PESA — principally because cuboid crowding is free of parameter sensitivities, is well suited to smaller population levels and is better able to maintain an evenly distributed truncated set that is more resistant to archival degradation. The cell-based approach, in particular, is sensitive to the resolution of the grid in which it operates, with shrinking fronts, discontinuities and inaccurate density approximations all possible if the grid is too coarse or too fine (a particular problem given that the optimal resolution is rarely known in advance). The  $\kappa^{\text{th}}$  neighbour approach tends to bias both the extremities of the objective-space and uncrowded *regions*, which can also lead to discontinuities in the archival expression of objective-space, particularly if the value of  $\kappa$  is poorly chosen. In unbounded domains it appears that this problem can be ameliorated to

some extent if the influence of  $\kappa$ -neighbourhood scores is weakened by the presence of noise (as in Mak\_OS\_KNN), though this requires further investigation.

The results as a whole also emphasise important methodological concerns. In particular, it is clear that the analysis of multiple Pareto-compliant indicators across a broad set of disparate test functions is the key to a well-balanced investigation. The variability of results across domains and under differing performance indicators in this work suggests that more shallow studies, which are limited in the breadth of the functions or utility-metrics under consideration, are prone to disguising biases in multiobjective optimisers and may promote misleading conclusions as to the general quality of the systems under investigation.

## 13.2 FUTURE WORK

As with most any preliminary study of new techniques, the most immediately interesting avenue of research is in the practical application of the proposed approaches and expansion into a broader set of domains. Though the comparative studies of unbounded optimisers were thorough in their analysis of static continuous bi-objective domains, investigations into constrained spaces (such as *AP-8*, *AP-9* and *AP-10*), noisy (*AP-6* and *AP-7*), dynamic (*AP-11*, *AP-12* and *AP-13*) and discrete problems (representing, for instance, knap-sack [256-258], travelling salesmen [259] and flowshop [59, 251, 260] domains) provide opportunities for both further empirical analysis and domain-specific specialisations. The use of the Mak\_Tree-based optimisers in real-world domains will also further elucidate the flexibility of those systems and may emphasise application issues that are obscured in a purely research-based environment. It is also worth noting that the significance, sensitivity and potential automation of optimiser parameters requires further investigation as such analysis was outside the focus of this work. Additional study may be particularly important for cell-based systems, where both theory and results seem to imply that correctly setting the grid resolution rests as an extremely important performance factor, and for unbounded density estimates, where adaptive neighbourhood sizing is likely to be beneficial.

Perhaps the most exciting new work revolves around the development of additional unbounded extensions, both with a view to further establishing the power of unbounded archiving and extracting real performance gains from practical systems.

Given the prevalence of truncation in contemporary optimisers, the breadth of available techniques that may benefit from the use of truly unbounded elite sets is impressive. Moreover, using the reference, active and hybridised techniques described in Section 9.2 as a guide, the extension of these systems should be relatively straightforward. Also note that while this thesis has focussed on contemporary (core) evolutionary algorithms, there is little reason why future studies should be so constrained — swarm, ant colony, artificial life, memetic and messy systems may also benefit from the use of unbounded elite sets.

Given the impressive performance of the simple Mak\_OS\_KNN system and, to a lesser extent, Diversity\_PAES, there is also scope for continued work in the development of novel algorithms centred around the use of unbounded archival sets. An interesting idea in this regard would be to use multiple Mak\_Trees to maintain distinct fronts. With access to more than just the elite set, a more complete picture of the explored objective-space is offered, particularly early in the run when the leading front is typically small. Access to a broader set of solutions may also better enable the system to effectively navigate through zero-utility gradient spaces, escape deceptive regions and integrate noise. The development of new selection (or extraction) strategies that are better suited to unbounded sets, also rests as a potentially rich area of study. In particular, work should focus on strategies that are robust in the presence of objective-space biases and shifting archival sizes.

Though there is little debating the power of the Mak\_Tree, additional extension is required to make it a truly flexible system. In particular, the storage of solutions to constrained, dynamic and noisy functions in an unbounded elite set is not trivial. In both noisy and dynamic problems there is no guarantee that the results stored in the unbounded archive necessarily correspond to the *true* value of the stored solutions at any given time. The temporal nature of performance in these non-static domains means that unique techniques to unbounded storage are required. For dynamic problems with known time-steps, the use of a fixed number of time-specific Mak\_Trees may be sufficient, with the optimiser selecting solutions from the most relevant archive. In the case where such knowledge is unknown, it is feasible that a variable number of time-specific trees could be maintained, with archival analysis suggesting when distinct trees should be merged, deleted or created. An interesting

approach to noisy problems is the storage of Mak\_Tree nodes with performance intervals rather than precise points (perhaps extending the cell-based Mak\_Tree).

With respect to the storage of constraint violators in conventional Mak\_Trees, a number of options exist and it is important to identify which technique yields the most promising outcomes. Potential alternatives include the complete exclusion of violators, complete inclusion with violation annotations, the use of multiple Mak\_Trees (with each assigned a maximum level of allowable violation) or variations thereof.

Given the development and use of the comprehensive *AP* test suite in this thesis, a particularly valuable avenue of work is to produce further comparative studies, both to explore the relative behaviours of differing systems and to elucidate the strengths and weaknesses of the *AP* problems themselves. Particular benefit would be found in differentiating the performances of the leading non-evolutionary approaches to multiobjective optimisation, including, but not limited to, operations research techniques and swarm and ant colony systems. Expansion of the algorithms under analysis will also provide a more complete view of the state of the field as a whole and may indicate problem features that are suited to particular optimiser strategies.

On a similar note, a more focussed analysis of the peculiarities of particular domain characteristics is of great import. Preliminary results in this study indicate, for instance, that multi-frontality is best solved by narrow-search or noisy systems, and it is important to know whether this and other domain-related conclusions are true in general or simply quirks of the examined test functions. An intensive investigation of this area may further clarify the key components of multiobjective optimisers and act as a primer for decision-makers seeking well-matched systems and researchers looking to develop both specialist and generalist optimisation techniques.

There also exist a host of less pressing, though potentially valuable, options for study (many of which have already been touched upon in this thesis), including:

- The implementation of the Dynamic Range Tree and investigation of specialisation into the bi-objective domain (potentially including the use of one-dimensional range trees).

- The specialisation of the Dominated Tree into bi-objective domains, including investigation into the improvement of efficiency when deleting dominated sets (potentially by harnessing the power of binary sub-tree deletions).
- The use of the Mak\_Terminator in algorithmic studies to illustrate performance trends, particularly with respect to convergence characteristics.
- The investigation of costs associated with rebuilding adaptive grids akin to those used in PAES, PESA and the grid-based Mak\_Terminator. If the costs are high, development of new data structures, improvements on existing structures or an increased focus on the development of heuristics may be valuable.
- The analysis of the Mak\_Terminator parameters with a view to exploring the trade-offs that exist between the period of observation and the accuracy of convergence estimation. Subsequent development of practical heuristics that inform parameter settings would also be of value.
- The refinement of the Mak\_Presentation algorithm to maximise fidelity between desired and resultant set sizes. An interesting avenue of exploration is to adjust optimal spaces between solutions according to an annealing process. Analysis of Mak\_Presentation performance at multiple points in the run (rather than simply at termination) may also reveal more behavioural characteristics of the technique.
- A more thorough investigation into the use of pre-existing truncation methods for the presentation of unbounded sets (including, but not limited to, nearest-neighbour, cuboid,  $\kappa^{\text{th}}$  nearest-neighbour and grid-based approaches). Analysis should include an exploration of set distribution, computational cost and solution quality.
- The comparison of the performance of averaged  $\kappa$  nearest-neighbours crowding estimation and the  $\kappa^{\text{th}}$  nearest-neighbour technique traditionally employed in SPEA2.
- The development of the most appropriate way to capitalise on multi-faceted crowding estimations in an effort to offset the bias/variance dilemma. Possible avenues include the integration of multiple density estimates via Mak\_Tree

annotations, variable selection strategies that move between different types of crowding estimation and the development of heuristics that aid in determining the most appropriate type of estimate at a given time.

- This thesis posits that the performance of the cuboid crowding estimation technique proposed by Deb *et al.* [82] will degrade markedly in higher dimensional objective-spaces (where the correlation between the estimate and the actual crowding of solutions weakens significantly). A more formal analysis of this behaviour is important given the status and popularity of NSGA-II in the field. The replacement of cuboid-crowding with  $\kappa^{\text{th}}$  nearest-neighbour clustering or PAES-like grid estimates in NSGA-II may also yield substantial performance gains (though potentially at a loss in efficiency).
- The expansion of the analysis of the optimisers reviewed in this thesis to include other Pareto-compliant indicators, such as the  $R_2$  and  $R_3$  metrics offered by Hansen and Jaszekiewicz [186]. Such additional investigation should clarify the performance of IBEA\_E (which appeared inconsistent between hypervolume and epsilon measures) and further reduce the impact of individual biases present in each of the scalar measures.
- The impressive performance of IBEA\_E across numerous test problems suggests that the use of Pareto compliant indicators in evolutionary optimisers is a promising direction. However, the generally poor results offered by IBEA\_H suggest that compliance alone is insufficient for optimiser efficacy. As such, more work should be focussed on understanding the interaction between the indicator and the user-defined  $\nu$  parameter, the biases that exist in each of the explored indicators and the performance of other indicators (such as the  $R_2$  and  $R_3$  [186] metrics).
- Given the improvements afforded contemporary methods when they capitalise on unbounded elite stores, extension of IBEA specifically into an unbounded system may yield similar gains, though care must be taken to maximise the efficiency of the procedure.
- The poor performance of both PAES and PESA may be attributable to the selected grid resolutions for these systems. Though this emphasises the

sensitivity of the corresponding parameters (and suggests that finer grid detail does not necessarily infer improved performance), it is worthwhile re-examining these systems under more favourable configurations. Such testing should incorporate the unbounded extensions, which may also benefit from parameter tuning (note that the optimal settings for bounded and unbounded systems are likely to differ however).

- The development of novel variation techniques that endeavour to adaptively track and exploit apparent cross-gene dependencies with a view to creating more powerful and robust optimisers.



### **13.3 FINAL THOUGHTS**

The introduction of this thesis questioned whether it was feasible to create unbounded bi-objective archives and, if so, whether such constructs could deliver tangible performance improvements both within and outside of the optimisation process. By exploiting the special properties of the non-dominated bi-objective set, the newly developed Mak\_Tree and its various extensions ably demonstrate the feasibility of efficient unbounded storage in bi-objective domains, with time-costs that are not only superior to pre-existing unbounded approaches but frequently better than, and at least competitive with, the deeply flawed tightly bound strategies common throughout the field. Moreover, the Mak\_Tree facilitated the development of new termination and presentation systems that benefit from access to a complete representation of truly non-dominated solutions, while both novel and adapted optimisers emphasised that unbounded archiving imparts a real practical performance advantage in bi-objective optimisation. Thus, some 406 pages after it was first posed, there can be little doubt as to how to answer the questions of the introduction: not only is unbounded bi-objective archiving feasible but, with its capacity for real performance improvements, it is also an extremely valuable new piece towards solving the multiobjective puzzle.

# Chapter

# 14

## References

*“You will find it a very good practice always to verify your references, sir.”*

*Doctor Martin Routh –  
Scientist*

## 14 REFERENCES

- [1] Coello, C.A.C. *Handling Preferences in Evolutionary Multiobjective Optimization: A Survey*. in *2000 Congress on Evolutionary Computation*. 2000: IEEE Service Center.
- [2] Srinivas, N. and Deb, K., *Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms*. *Evolutionary Computation*, 1994. **2**(3): p. 221-248.
- [3] Deb, K., *Multi — Objective Evolutionary Algorithms: Introducing Bias Among Pareto — Optimal Solutions*. 1999, Indian Institute of Technology, Kanpur, India.
- [4] Horn, J., *Multicriterion Decision Making: Handbook of Evolutionary Computation*, Back, T., Fogel, D., and Michalewicz, Z., Editors. 1997, IOP Publishing Ltd. and Oxford University Press.
- [5] Coello, C.A.C., *A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques*. *Knowledge and Information Systems. An International Journal*, 1999. **1**(3): p. 269-308.
- [6] Coello, C.A.C. and Christiansen, A.D., *Multiobjective Optimization of Trusses using Genetic Algorithms*. *Computers and Structures*, 2000. **75**(6): p. 647-660.
- [7] Jin, H. and Wong, M., *Adaptive Diversity Maintenance and Convergence Guarantee in Multiobjective Evolutionary Algorithms*. in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*. 2003: IEEE Press.
- [8] Andersson, J., *Multiobjective Optimization in Engineering Design — Applications to Fluid Power Systems*. 2001, Division of Fluid and Mechanical Engineering Systems. Department of Mechanical Engineering. Linköping Universitet.
- [9] Deb, K., Patrap, A., and Moitra, S., *Mechanical Component Design for Multi-objective using Elitist Non-dominated Sorting GA*. 2000, Indian Institute of Technology, Kanpur, India.
- [10] Buche, D., Stoll, P., Dornberger, R., and Kourmoursakos, P., *Multiobjective Evolutionary Algorithm for the Optimization of Noisy Combustion Processes*. *IEEE Transactions on Systems, Man, and Cybernetics Part C — Applications and Reviews*, 2002. **32**(4): p. 460-473.
- [11] Coello, C.A.C., Veldhuizen, D.A.V., and Lamont, G.B., *Applications (Chapter Six)*, in *Evolutionary Algorithms for Solving Multi-Objective Problems*. 2002, Kluwer Academic Publishers.
- [12] D'Angelo, S. and Minisci, E. *Multi-Objective Evolutionary Optimization of Subsonic Airfoils by Kriging Approximation and Evolution Control*. in *IEEE Congress on Evolutionary Computation (CEC)*. 2005. Edinburgh, Scotland: IEEE Service Centre.
- [13] Willmes, L. and Back, T. *Multi-criteria Airfoil Design with Evolution Strategies*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [14] Li, J. and Satofuka, N., *Optimization Design of a Compressor Cascade Airfoil using a Navier-Stokes Solver and Genetic Algorithms*. *Journal of Power and Energy*, 2002. **216**(2): p. 195-202.

- [15] Emmerich, M. and Naujoks, B. *Metamodel Assisted Multiobjective Optimisation Strategies and their Application in Airfoil Design*. in *Adaptive Computing in Design and Manufacture VI*. 2004: Springer.
- [16] Buche, D., Guidati, G., Stoll, P., and Kourmoursakos, P. *Self-Organizing Maps for Pareto Optimization of Airfoils*. in *Parallel Problem Solving from Nature — PPSN VII*. 2002: Springer-Verlag. Lecture Notes in Computer Science No. 2439.
- [17] Benini, E. and Toffolo, A., *Development of High-Performance Airfoils for Axial Flow Compressors Using Evolutionary Computation*. Journal of Propulsion and Power, 2002. **18**(3): p. 544-554.
- [18] Pulliam, T.H., Nemec, M., Hoslt, T., and Zingg, D.W. *Comparison of Evolutionary (Genetic) Algorithm and Adjoint Methods for Multi-Objective Viscous Airfoil Optimizations*. in *41st Aerospace Sciences Meeting. Paper AIAA 2003-0298*. 2003.
- [19] Erickson, M., Mayer, A., and Horn, J., *The Niche Pareto Genetic Algorithm 2 Applied to the Design of Groundwater Remediation Systems* *First International Conference on Evolutionary Multi-Criterion Optimization*, Zitzler, E., et al., Editors. 2001, Springer-Verlag. Lecture Notes in Computer Science No. 1993. p. 681-695.
- [20] Cheung, P.B., Reis, L.F.R., Formiga, K.T.M., Chaudhry, F.H., and Ticona, W.G.C. *Multiobjective Evolutionary Algorithms Applied to the Rehabilitation of a Water Distribution System: A Comparative Study*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [21] Dorn, J.L. and Ranjithan, S.R. *Evolutionary Multiobjective Optimization in Watershed Water Quality Management*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [22] Reed, P.M., *Striking the Balance: Long-Term Groundwater Monitoring Design for Multiple Conflicting Objectives*. 2002, Graduate College of the University of Illinois at Urbana-Champaign.
- [23] Teo, J. and Abbass, H.A. *Coordination and Synchronization of Locomotion in a Virtual Robot*. in *Proceedings of the 9th International Conference on Neural Information Processing (ICONIP'02)*. 2002.
- [24] Teo, J. and Abbass, H.A. *Elucidating the Benefits of A Self-Adaptive Pareto EMO Approach for Evolving Legged Locomotion in Artificial Creatures*. in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC2003)*. 2003: IEEE Press.
- [25] Yanase, T. and Iba, H. *Evolutionary Motion Design for Humanoid Robots*. in *Genetic and Evolutionary Computation Conference (GECCO 2006)*. 2006. Washington, USA: ACM Press.
- [26] Omran, M.G.H., Engelbrecht, A.P., and Salman, A. *Differential Evolution Methods for Unsupervised Image Classification*. in *IEEE Congress on Evolutionary Computation (CEC 2005)*. 2005. Edinburgh, Scotland.
- [27] Sal, D. and Grana, M. *Hyperspectral Image Watermarking with an Evolutionary Algorithm*. in *Knowledge-Based Intelligent Information and Engineering Systems, Part One, Proceedings*. 2005: Springer, Lecture Notes in Artificial Intelligence.

- [28] Greenfield, G.R. *Evolving Aesthetic Images using Multiobjective Optimization*. in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*. 2003: IEEE Press.
- [29] Markowska-Kaczmar, U. and Wnuk-Lipinski, P. *Rule Extraction from Neural Network by Genetic Algorithm with Pareto Optimization*. in *Artificial Intelligence and Soft Computing - ICAISC 2004, 7th International Conference. Proceedings*. 2004: Springer. Lecture Notes in Computer Science. Volume 3070.
- [30] Jin, Y., Okabe, T., and Sendhoff, B. *Neural Network Regularization and Ensembling Using Multi-objective Evolutionary Algorithms*. in *2004 Congress on Evolutionary Computation (CEC 2004)*. 2004: IEEE Service Center.
- [31] Abbass, H.A., *Speeding up Backpropagation using Multiobjective Evolutionary Algorithms*. Neural Computation, 2003. **15**(11): p. 2705-2726.
- [32] Abbass, H.A. *Pareto Neuro-Evolution: Constructing Ensemble of Neural Networks Using Multi-objective Optimization*. in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*. 2003: IEEE Press.
- [33] Hatanaka, T., Kondo, N., and Uosaki, K. *Multi-Objective Structure Selection for Radial Basis Function Networks Based on Genetic Algorithm*. in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*. 2003: IEEE Press.
- [34] Coello, C.A.C. and Aguirre, A.H., *Design of Combinational Logic Circuits through an Evolutionary Multiobjective Optimization Approach*. Artificial Intelligence for Engineering, Design, Analysis and Manufacture, 2002. **16**(1): p. 39-53.
- [35] Smedt, B.D. and Gielen, G. *HOLMES: Capturing the Yield-Optimized Design Space Boundaries of Analog and RF Integrated Circuits*. in *Proceedings of Design, Automation and Test in Europe (DATE'03)*. 2003: IEEE.
- [36] Luna, E.H., Coello, C.A.C., and Aguirre, A.H. *On the Use of a Population-Based Particle Swarm Optimizer to Design Combinational Logic Circuits*. in *Proceedings of the 2004 NASA/DoD Conference on Evolvable Hardware*. 2004: IEEE Computer Society.
- [37] Abbass, H.A., *An Evolutionary Artificial Neural Networks Approach for Breast Cancer Diagnosis*. Artificial Intelligence in Medicine, 2002. **25**(3): p. 265-281.
- [38] Lahanas, M., Schreibmann, E., and Baltas, D., *Multiobjective Inverse Planning for Intensity Modulated Radiotherapy with Constraint-free Gradient-based Optimization Algorithms*. Physics in Medicine and Biology, 2003. **48**: p. 2843-2871.
- [39] Suppaitnarm, A., Parks, G.T., Shea, K., and Clarkson, P.J., *Conceptual Design of Bicycle Frames by Multiobjective Shape Annealing*. Engineering Optimization, 2004. **36**(2): p. 165-188.
- [40] Chitty, D.M. and Hernandez, M.L. *A Hybrid Ant Colony Optimisation Technique for Dynamic Vehicle Routing*. in *Genetic and Evolutionary Computation — GECCO 2004. Proceedings of the Genetic and Evolutionary Computation Conference. Part I*. 2004: Springer-Verlag, Lecture Notes in Computer Science Vol. 3102.
- [41] Farina, M., Deb, K., and Amato, P., *Dynamic Multiobjective Optimization Problems: Test Cases, Approximations, and Applications*. IEEE Transactions on Evolutionary Computation, 2004. **8**(5): p. 425-442.

- [42] Hughes, E.J., *Evolutionary Multi-objective Ranking with Uncertainty and Noise*, in *First International Conference on Evolutionary Multi-Criterion Optimization*, Zitzler, E., et al., Editors. 2001, Springer-Verlag. Lecture Notes in Computer Science No. 1993. p. 329-343.
- [43] Babbar, M., Lakshmikantha, A., and Goldberg, D.E. *A Modified NSGA-II to Solve Noisy Multiobjective Problems*. in *2003 Genetic and Evolutionary Computation Conference. Late-Breaking Papers*. 2003: AAAI.
- [44] Bui, L.T., Abbass, H.A., and Essam, D. *Fitness Inheritance for Noisy Evolutionary Multi-Objective Optimization*. in *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation*. 2005. Washington DC, USA.
- [45] Jimenez, F. and Verdegay, J.L. *Constrained Multiobjective Optimization by Evolutionary Algorithms*. in *Proceedings of the International ICSC Symposium on Engineering of Intelligent Systems (EIS'98)*. 1998.
- [46] Kurpati, A., Azarm, S., and Wu, J., *Constraint handling improvements for multi-objective genetic algorithms*. *Structural and Multidisciplinary Optimization*, 2002. **23**(3): p. 204-213.
- [47] Chafekar, D., Xuan, J., and Rasheed, K. *Constrained Multi-objective Optimization Using Steady State Genetic Algorithms*. in *Genetic and Evolutionary Computation — GECCO 2003. Proceedings, Part I*. 2003: Springer. Lecture Notes in Computer Science Vol. 2723.
- [48] Binh, T.T. and Korn, U. *Multiobjective Evolution Strategy for Constrained Optimization Problems*. in *Proc. of the 15th IMACS World Congress on Scientific Computation, Modelling and Applied Mathematics*. 1997.
- [49] Saker, R., Abbass, H.A., and Karim, S. *An Evolutionary Algorithm for Constrained Multiobjective Optimization Problems*. in *The 5th Australasia-Japan Joint Workshop on Intelligent and Evolutionary Systems (AJWIS 2001)*. 2001.
- [50] Jimenez, F., Gomez-Skarmeta, A.F., Sanchez, G., and Deb, K. *An Evolutionary Algorithm for Constrained Multi-objective Optimization*. in *Congress on Evolutionary Computation (CEC 2002)*. 2002: IEEE Service Center.
- [51] Coello, C.A.C., Veldhuizen, D.A.V., and Lamont, G.B., *Evolutionary Algorithms for Solving Multi-Objective Problems*. Vol. May. 2002: Kluwer Academic Publishers.
- [52] Pena, S.I.V., Rionda, S.B., and Aguirre, A.H. *Multiobjective Shape Optimization Using Estimation Distribution Algorithms and Correlated Information*. in *Evolutionary Multi-criterion Optimization (EMO)*. 2005. Guanajuato, Mexico: ACM Press.
- [53] Garcia-Martinez, C., Cordon, O., and Herrera, F., *A Taxonomy and an Empirical Analysis of Multiple Objective Ant Colony Optimization Algorithms for the Bi-Criteria TSP*. *European Journal of Operational Research*, 2006: p. 116-148.
- [54] Iima, H., Nakase, R., and Sannomiya, N. *Proposition of Selection Operation in a Genetic Algorithm for a Job Shop Rescheduling Problem*. in *Evolutionary Multi-criteria Optimization*. 2005. Guanajuato, Mexico: Lecture Notes in Computer Science.
- [55] Erickson, M., Mayer, A., and Horn, J., *Multi-objective optimal design of groundwater remediation systems: application of the niched Pareto genetic algorithm (NPGA)*. *Advances in Water Resources*, 2002. **25**(1): p. 51-65.

- [56] Li, W. *Finding Pareto-Optimal Set by Merging Attractors for a Bi-objective Traveling Salesmen Problem*. in *Evolutionary Multi-criterion Optimization (EMO)*. 2005. Guanajuato, Mexico: Springer, LNCS.
- [57] Paquete, L., Chiarandini, M., and Stutzle, T. *Pareto Local Optimum Sets in the Biobjective Traveling Salesman Problem: An Experimental Study*. in *Metaheuristics for Multiobjective Optimisation*. 2004: Springer. Lecture Notes in Economics and Mathematical Systems Vol. 535.
- [58] Garcia-Martinez, C., Cordon, O., and Herrera, F. *An Empirical Analysis of Multiple Objective Ant Colony Optimization Algorithms for the Bi-criteria TSP*. in *Proceedings of the 4th International Workshop on Ant Colony Optimization and Swarm Intelligence*. 2004: Springer. Lecture Notes in Computer Science Vol. 3172.
- [59] Armentano, V.A. and Claudio, J.e.E., *An Application of a Multi-Objective Tabu Search Algorithm to a Bicriteria Flowshop Problem*. *Journal of Heuristics*, 2004. **10**(5): p. 463-481.
- [60] Zhou, A., Qingfu, Z., Yaochu, J., Tsang, E., and Okabe, T. *A Model-Based Evolutionary Algorithm for Bi-objective Optimization*. in *IEEE Congress on Evolutionary Computation (CEC)*. 2005. Edinburgh, Scotland.
- [61] Lopez-Ibanez, M., Paquete, L., and Stutzle, T. *On the Design of ACO for the Biobjective Quadratic Assignment Problem*. in *The 4th International Workshop of Ant Colony Optimization and Swarm Intelligence, ANTS*. 2004. Belgium: Springer, LNCS.
- [62] Jaszkievicz, A., *A Comparative Study of Multiple-Objective Metaheuristics on the Bi-Objective Set Covering Problem and the Pareto Memetic Algorithm*. *Annals of Operations Research*, 2004. **131**: p. 135-158.
- [63] Basseur, M., Lemesre, J., Dhaenens, C. and Talbi, E. *Cooperation between Branch and Bound and Evolutionary Approaches to Solve a Bi-objective Flow Shop Problem*. in *Proceedings of the Third International Workshop on Experimental and Efficient Algorithms (WEA'04)*. 2004: Springer-Verlag.
- [64] Pappa, G.L., Freitas, A.A., and Kaestner, C.A.A. *A Multiobjective Genetic Algorithm for Attribute Selection*. in *Proceedings of the 4th International Conference on Recent Advances in Soft Computing (RASC-2002)*. 2002: Nottingham Trent University.
- [65] Rouhiainen, C.J., Tade, M.O., and West, G. *Multi-Objective Genetic Algorithm for Optimal Scheduling of Chlorine Dosing in Water Distribution Systems*. in *Advances in Water Supply Management, Proceedings of the International Conference on Computers and Control in Water Industry (CCWI 2003)*. 2003: Balkema Publishers.
- [66] Tan, K.C., Lee, T.H., Chew, Y.H., and Lee, L.H. *A Hybrid Multiobjective Evolutionary Algorithm For Solving Truck and Trailer Vehicle Routing Problems*. in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*. 2003: IEEE Press.
- [67] Gandibleux, X., Morita, H., and Katoh, N. *Use of a Genetic Heritage for Solving the Assignment Problem with Two Objectives*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [68] Hughes, E.J. *Multi-objective Binary Search Optimisation*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.

- [69] Jin, Y. and Sendhoff, B. *Trade-Off between Performance and Robustness: An Evolutionary Multiobjective Approach*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [70] Paquete, L. and Stutzle, T. *A Two-Phase Local Search for the Biobjective Traveling Salesman Problem*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [71] Lacomme, P., Prins, C., and Sevaux, M. *Multiobjective Capacitated Arc Routing Problem*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [72] Watanabe, S., Hiroyasu, T., and Miki, M. *Multi-objective Rectangular Packing Problem and Its Applications*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [73] de Toro, F., Ros, E., Mota, S., and Ortega, J. *Non-invasive Atrial Disease Diagnosis Using Decision Rules: A Multi-objective Optimization Approach*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [74] Greiner, D., Galvan, B., and Winter, G. *Safety Systems Optimum Design by Multicriteria Evolutionary Algorithms*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [75] Khare, V., Yao, X., and Deb, K. *Performance Scaling of Multi-objective Evolutionary Algorithms*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [76] Hughes, E.J. *Evolutionary Many-Objective Optimisation: Many Once or One Many?* in *Congress on Evolutionary Computation (CEC 2005)*. 2005: IEEE Service Centre.
- [77] Purshouse, R.C. and Fleming, P.J. *Evolutionary Multi-Objective Optimisation: An Exploratory Analysis*. in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*. 2003: IEEE Press.
- [78] Farina, M. and Amato, P. *On the Optimal Solution Definition for Many-criteria Optimization Problems*. in *Proceedings of the NAFIPS-FLINT International Conference 2002*. 2002: IEEE Service Center.
- [79] Knowles, J.D. and Corne, D.W., *Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy*. *Evolutionary Computation*, 2000. 8(2): p. 149-172.
- [80] Zitzler, E. and Thiele, L., *An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach*. 1998, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH).
- [81] Zitzler, E., Laumanns, M., and Thiele, L. *SPEA2: Improving the Strength Pareto Evolutionary Algorithm*. in *EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*. 2002.



- [82] Deb, K., Agrawal, S., Pratab, A., and Meyarivan, T. *A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II*. in *Proceedings of the Parallel Problem Solving from Nature VI Conference*. 2000: Springer. Lecture Notes in Computer Science No. 1917.
- [83] Knowles, J.D., Corne, D.W., and Oates, M.J. *The Pareto-Envelope based Selection Algorithm for Multiobjective Optimization*. in *Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature (PPSN VI)*. 2000: Springer.
- [84] Corne, D.W., Jerram, N.R., Knowles, J.D., and Oates, M.J. *PESA-II: Region-based Selection in Evolutionary Multiobjective Optimization*. in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001)*. 2001: Morgan Kaufmann Publishers.
- [85] Knowles, J. and Corne, D. *M-PAES: A Memetic Algorithm for Multiobjective Optimization*. in *2000 Congress on Evolutionary Computation*. 2000: IEEE Service Center.
- [86] Costa, L. and Oliveira, P. *An Elitist Genetic Algorithm for Multiobjective Optimization*. in *Proceedings of the 4th Metaheuristics International Conference — MIC 2001*. 2001: Program Operational Ciencia, Tecnologia, Inovacao do Quadro Comunitario de Apoio III de Fundacao para a Ciencia e Tecnologia.
- [87] Costa, L. and Oliveira, P., *An Adaptive Sharing Elitist Evolution Strategy for Multiobjective Optimization*. *Evolutionary Computation*, 2003. **11**(4): p. 417-438.
- [88] Vasconcelos, J.A., Adriano, R.L.S., Vieira, D.A.G., Souza, G.F.D., and Azevedo, H.S., *NSGA with Elitism Applied to Solve Multiobjective Optimization Problems*. *Journal of Microwaves and Optoelectronics*, 2002. **2**(6): p. 59-69.
- [89] Bartz-Beielstein, T., Limbourg, P., Parsopoulos, K.E., Vrahatis, M.N., Mehnen, J., and Schmitt, K. *Particle Swarm Optimizers for Pareto Optimization with Enhanced Archiving Techniques*. in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*. 2003: IEEE Press.
- [90] Yang, S.M., Shao, D.G., and Luo, Y.J. *Dynamic Archive Evolution Strategy for Multiobjective Optimization*. in *Evolutionary Multi-Criterion Optimization (EMO 2005)*. 2005. Guanajuato, Mexico: Springer (Lecture Notes in Computer Science).
- [91] Zitzler, E. and Kunzli, S. *Indicator-based Selection in Multiobjective Search*. in *Parallel Problem Solving from Nature - PPSN VIII*. 2004: Springer-Verlag. Lecture Notes in Computer Science Vol. 3242.
- [92] Zitzler, E., Deb, K., and Thiele, L., *Comparison of Multiobjective Evolutionary Algorithms: Empirical Results*. *Evolutionary Computation*, 2000. **8**(2): p. 173-195.
- [93] Knowles, J. and Corne, D. *Bounded Pareto Archiving: Theory and Practice*. in *Metaheuristics for Multiobjective Optimisation*. 2004: Springer. Lecture Notes in Economics and Mathematical Systems Vol. 535.
- [94] Deb, K. and Goyal, T., *Controlled Elitist Non-dominated Sorting Genetic Algorithms for Better Convergence*. 2000, Indian Institute of Technology, Kanpur, India.
- [95] Jensen, M.T., *Reducing the Run-Time Complexity of Multiobjective EAs: The NSGA-II and Other Algorithms*. *IEEE Transactions on Evolutionary Computation*, 2003. **7**(5): p. 503-515.

- [96] Mostaghim, S., Teich, J., and Tyagi, A. *Comparison of Data Structures for Storing Pareto-sets in MOEAs*. in *Congress on Evolutionary Computation (CEC 2002)*. 2002: IEEE Service Center.
- [97] Fieldsend, J.E., Everson, R.M., and Singh, S., *Using Unconstrained Elite Archives for Multiobjective Optimization*. *IEEE Transactions on Evolutionary Computation*, 2003. 7(3): p. 305-323.
- [98] Laumanns, M., Thiele, L., Zitzler, E., and Deb, K. *Archiving with Guaranteed Convergence and Diversity in Multi-Objective Optimization*. in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2002)*. 2002: Morgan Kaufmann Publishers.
- [99] Anderson, M.B., Burkhalter, J.E., and Jenkins, R.M. *Missile Aerodynamic Shape Optimization Using Genetic Algorithms*. in *AIAA Aerospace Sciences Meeting and Exhibit*. 1999: AIAA Paper 99-0261.
- [100] Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., and Grunert da Fonseca, V., *Performance Assessment of Multiobjective Optimizers: An Analysis and Review*. *IEEE Transactions on Evolutionary Computation*, 2003. 7(2): p. 117-132.
- [101] Deb, K., Mohan, M., and Mishra, S. *Towards a Quick Computation of Well-Spread Pareto-Optimal Solutions*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [102] Laumanns, M., Rudolph, G., and Schwefel, H., *Approximating the Pareto Set: Concepts, Diversity Issues, and Performance Assessment*. 1999, Dortmund: Department of Computer Science/LS11.
- [103] Deb, K. and Jain, S. *Running Performance Metrics for Evolutionary Multi-Objective Optimization*. in *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning (SEAL'02)*. 2002: Nanyang Technical University.
- [104] Fonseca, C.M. and Fleming, P.J. *On the Performance Assessment and Comparison of Stochastic Multiobjective Optimizers*. in *Parallel Problem Solving from Nature — PPSN IV*. 1996: Springer-Verlag.
- [105] Knowles, J.D., *Local-Search and Hybrid Evolutionary Algorithms for Pareto Optimization*. 2002, The University of Reading, Department of Computer Science: Reading, UK.
- [106] *The Real Cost of F1*, in *F1 Racing* (Magazine Article), March, 2007, Derwent Howard.
- [107] Scarborough, C., *Technical Writer, Autosport.com*, (personal communication: e-mail). Provided sketch of Formula One vehicle, 13/6/07.
- [108] The Federation Internationale de l'Automobile (FIA), *2006 Formula One Technical Regulations* (available online at <http://www.fia.com/sport/Regulations/f1regs.html>). 2006.
- [109] Sarkar, D. and Modak, J.M., *Pareto-optimal Solutions for Multi-objective Optimization of Fed-batch Bioreactors using Nondominated Sorting Genetic Algorithm*. *Chemical Engineering Science*, 2005. 60(2): p. 481-492.
- [110] Baran, B., von Lucken, C., and Sotelo, A., *Multi-objective pump scheduling optimisation using evolutionary strategies*. *Advances in Engineering Software*, 2005. 36(1): p. 39-47.
- [111] Shengjing, M., Hongye, S., Jia, T., Yong, G., and Chu, J., *Scalable multi-objective optimization of industrial purified terephthalic acid (PTA) oxidation process*. *Computers & Chemical Engineering*, 2004. 28(11): p. 2219-2231.

- [112] Mitra, K., Majumdar, S., and Raha, S., *Multiobjective Dynamic Optimization of Epoxy Polymerization Process*. Computer and Chemical Engineering, 2004. **28**(12): p. 2583-2594.
- [113] Mitra, K. and Gopinath, R., *Multiobjective Optimization of an Industrial Grinding Operation Using Elitist Nondominated Sorting Genetic Algorithm*. Chemical Engineering Science, 2004. **59**(2): p. 385-396.
- [114] Kanazaki, M., Obayashi, S., and Nakahashi, K., *Exhaust Manifold Design with Tapered Pipes using Divided Range MOGA*. Engineering Optimization, 2004. **36**(2): p. 149-163.
- [115] Deb, K., Mitra, K., Dewri, R., and Majumdar, S., *Towards a Better Understanding of the Epoxy Polymerization Process Using Multi-objective Evolutionary Computation*. Chemical Engineering Science, 2004. **59**(20): p. 4261-4277.
- [116] Dedieu, S., Pibouleau, L., Azzaro-Pantel, C., and Domenech, S., *Design and retrofit of multiobjective batch plants via a multicriteria genetic algorithm*. Computers & Chemical Engineering, 2003. **27**(12): p. 1723-1740.
- [117] Veldhuizen, D.A.V. and Lamont, G.B., *Multiobjective Evolutionary Algorithm Research: A History and Analysis*. 1998, Van Veldhuizen and Lamont, 1998.
- [118] Fonseca, C.M. and Fleming, P.J., *An Overview of Evolutionary Algorithms in Multiobjective Optimization*. Evolutionary Computation, 1995. **3**(1): p. 1-16.
- [119] Eklund, N.H. and Embrechts, M.J., *Multi-Objective Optimization of Spectra Using Genetic Algorithms*. Journal of Illuminating Engineering Society, 2001. **30**: p. 65-72.
- [120] Jin, Y., von Seelen, W., and Sendhoff, B., *On Generating Flexible, Complete, Consistent and Compact (FC) Fuzzy Rule Systems from Data using Evolution Strategies*. IEEE Transactions on Systems, Man, and Cybernetics, 1999. **29**(4): p. 829-845.
- [121] Hussain, T., Montana, D., and Vidaver, G. *Evolution-Based Deliberative Planning for Cooperating Unmanned Ground Vehicles in a Dynamic Environment*. in *Genetic and Evolutionary Computation — GECCO 2004. Proceedings of the Genetic and Evolutionary Computation Conference. Part II*. 2004: Springer-Verlag, Lecture Notes in Computer Science Vol. 3103.
- [122] Aerts, J.C.J.H., van Herwijnen, M., and Stewart, T.J. *Using Simulated Annealing and Spatial Goal Programming for Solving a Multi Site Land Use Allocation Problem*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [123] Guntch, M. and Middendorf, M. *Solving Multi-criteria Optimization Problems with Population-Based ACO*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [124] Zitzler, E. and Thiele, L. *Multiobjective Optimization Using Evolutionary Algorithms — A Comparative Study*. in *Parallel Problem Solving from Nature V*. 1998: Springer-Verlag.
- [125] Coello, C.A.C., Veldhuizen, D.A.V., and Lamont, G.B., *Evolutionary Algorithm MOP Approaches (Chapter Two)*, in *Evolutionary Algorithms for Solving Multi-Objective Problems*. 2002, Kluwer Academic Publishers.

- [126] Laumanns, M., Zitzler, E., and Thiele, L. *A Unified Model for Multi-Objective Evolutionary Algorithms with Elitism*. in *2000 Congress on Evolutionary Computation*. 2000: IEEE Service Center.
- [127] Goldberg, D.E. and Deb, K., *A Comparative Analysis of Selection Schemes Used in Genetic Algorithms*, in *Foundations of Genetic Algorithms*. 1991, Morgan Kaufman. p. 69-93.
- [128] Glover, F., *Future Paths for Integer Programming and Links to Artificial Intelligence*. Computers and Operations Research, 1986: p. 533-549.
- [129] Glover, F., *Tabu Search - Part I*. ORSA Journal on Computing, 1989: p. 190-206.
- [130] Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P., *Optimization by Simulated Annealing*. Science, 1983: p. 671-680.
- [131] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M., Teller, A.H., and Teller, E., *Equation of State Calculations by Fast Computing Machines*. Journal of Chem. Phys., 1953: p. 1087-1092.
- [132] Holland, J.H., *Adaptation in Natural and Artificial Systems*. 1975, Ann Arbor: The University of Michigan Press.
- [133] Goldberg, D.E., *Genetic Algorithms in Search*. 1989, Reading, MA: AddisonWesley.
- [134] Dumitrescu, D., Lazzerini, B., Jain, L.C., and Dumitrescu, A., *Evolutionary Computation*. 2000: CRC Press International.
- [135] Alba, E. and Cotta, C., *Evolutionary Algorithms (Chapter One, Models and Paradigms)*, in *Handbook of Bioinspired Algorithms and Applications*, Olariu, S. and Zomaya, A., Editors. 2005, Chapman and Hall.
- [136] Corne, D.W., Knowles, J.D., and Oates, M.J. *The Pareto Envelope-based Selection Algorithm for Multiobjective Optimization*. in *Proceedings of the Parallel Problem Solving from Nature VI Conference*. 2000: Springer. Lecture Notes in Computer Science No. 1917.
- [137] Zitzler, E. and Thiele, L., *Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach*. IEEE Transactions on Evolutionary Computation, 1999. 3(4): p. 257-271.
- [138] Buche, D. and Dornberger, R. *New Evolutionary Algorithm for Multi-Objective Optimization and the Application to Engineering Design Problems*. in *Proceedings of the Fourth World Congress of Structural and Multidisciplinary Optimization*. 2001.
- [139] Tan, K.C., Yang, Y.J., Goh, C.K., and Lee, T.H. *Enhanced Distribution and Exploration for Multiobjective Evolutionary Algorithms*. in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*. 2003: IEEE Press.
- [140] Fonseca, C.M. and Fleming, P.J. *Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization*. in *Proceedings of the Fifth International Conference on Genetic Algorithms*. 1993: University of Illinois at Urbana-Champaign Morgan Kauffman Publishers.
- [141] Horn, J. and Nafpliotis, N., *Multiobjective Optimization using the Niche Pareto Genetic Algorithm*. 1993, University of Illinois at Urbana-Champaign.
- [142] Horn, J., Nafpliotis, N., and Goldberg, D.E. *A Niche Pareto Genetic Algorithm for Multiobjective Optimization*. in *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*. 1994: IEEE Service Center.

- [143] Knowles, J.D. and Corne, D.W. *The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Multiobjective Optimisation*. in *1999 Congress on Evolutionary Computation*. 1999: IEEE Service Center.
- [144] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T., *A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II*. *IEEE Transactions on Evolutionary Computation*, 2002. 6(2): p. 182-197.
- [145] Zitzler, E., Laumanns, M., and Thiele, L., *SPEA2: Improving the Strength Pareto Evolutionary Algorithm*. 2001, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich.
- [146] Schaffer, J.D. *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*. in *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*. 1985: Lawrence Erlbaum.
- [147] Fonseca, C.M. and Fleming, P.J., *An Overview of Evolutionary Algorithms in Multiobjective Optimization*, Department of Automatic Control and Systems Engineering, University of Sheffield.
- [148] Deb, K., *Multi-Objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems*. *Evolutionary Computation*, 1999. 7(3): p. 205-230.
- [149] Xiaodong, L. *Better Spread and Convergence: Particle Swarm Multiobjective Optimization Using the Maximin Fitness Function*. in *Genetic and Evolutionary Computation — GECCO 2004. Proceedings of the Genetic and Evolutionary Computation Conference. Part I*. 2004: Springer-Verlag, Lecture Notes in Computer Science Vol. 3102.
- [150] Maneeratana, K., Boonlong, K., and Chaiyaratana, N. *Multi-objective Optimisation by Co-operative Co-evolution*. in *Parallel Problem Solving from Nature - PPSN VIII*. 2004: Springer-Verlag. Lecture Notes in Computer Science Vol. 3242.
- [151] Iorio, A.W. and Xiaodong, L. *A Cooperative Coevolutionary Multiobjective Algorithm Using Non-dominated Sorting*. in *Genetic and Evolutionary Computation — GECCO 2004. Proceedings of the Genetic and Evolutionary Computation Conference. Part I*. 2004: Springer-Verlag, Lecture Notes in Computer Science Vol. 3102.
- [152] Veldhuizen, D.A.V., *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. 1999.
- [153] Jin, Y. and Branke, J., *Evolutionary Optimization in Uncertain Environments - A Survey*. *IEEE Transactions on Evolutionary Computation*, 2005. 9(3).
- [154] Jin, Y. and Sendhoff, B. *Constructing Dynamic Optimization Test Problems Using the Multi-objective Optimization Concept*. in *Applications of Evolutionary Computing. Proceedings of Evoworkshops 2004: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoMUSART, and EvoSTOC*. 2004: Springer. Lecture Notes in Computer Science Vol. 3005.
- [155] Koppen, M., Franke, K., and Nickolay, B. *Fuzzy-Pareto-Dominance Driven Multiobjective Genetic Algorithm*. in *Proceedings of the 10th IFSA World Congress (IFSA 2003)*. 2003.
- [156] Coello, C.A.C., Veldhuizen, D.A.V., and Lamont, G.B., *Multi-Criteria Decision Making (Chapter Four)*, in *Evolutionary Algorithms for Solving Multi-Objective Problems*. 2002, Kluwer Academic Publishers.
- [157] Bellman, R. and Dreyfus, S., *Applied Dynamic Programming*. 1962, Princeton, New Jersey: Princeton University Press.

- [158] Huband, S., Barone, L., While, L., and Hingston, P. *A Scalable Multi-objective Test Problem Toolkit*. in *Evolutionary Multi-Criterion Optimization (EMO): Third International Conference*. 2005: Springer-Verlag.
- [159] Fonseca, C.M. and Fleming, P.J., *Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms I: A Unified Formulation*. 1995, University of Sheffield.
- [160] Wolpert, D.H. and Macready, W.G., *No Free Lunch Theorems for Optimization*. IEEE Transactions on Evolutionary Computation, 1997. **1**(1): p. 67-82.
- [161] Corne, D.W. and Knowles, J.D. *No Free Lunch and Free Leftovers Theorems for Multiobjective Optimisation Problems*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [162] Veldhuizen, D.A.V. and Lamont, G.B. *Multiobjective Evolutionary Algorithm Test Suites*. in *Proceedings of the 1999 ACM Symposium on Applied Computing*. 1999: ACM.
- [163] Deb, K., Thiele, L., Laumanns, M., and Zitzler, E., *Scalable Test Problems for Evolutionary Multi-Objective Optimization*. 2001, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH).
- [164] Deb, K., Pratap, A., and Meyarivan, T. *Constrained Test Problems for Multi-objective Evolutionary Optimization*. in *First International Conference on Evolutionary Multi-Criterion Optimization*. 2001: Springer-Verlag. Lecture Notes in Computer Science No. 1993.
- [165] Gaspar-Cunha, A. and Covas, J.A. *A Real-World Test Problem for EMO Algorithms*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [166] Okabe, T., Jin, Y., Olhofer, M., and Sendhoff, B. *On Test Functions for Evolutionary Multi-objective Optimization*. in *Parallel Problem Solving from Nature - PPSN VIII*. 2004: Springer-Verlag. Lecture Notes in Computer Science Vol. 3242.
- [167] Adra, S.F., Griffin, I., and Fleming, P.J. *Hybrid Multiobjective Genetic Algorithm with a new Adaptive Local Search Process*. in *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation (GECCO)*. 2005. Washington DC, USA.
- [168] Reyes-Sierra, M. and Coello, C. *Dynamic Fitness Inheritance Proportion for Multi-Objective Particle Swarm Optimization*. in *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation (GECCO)*. 2006. Washington, USA.
- [169] Berry, A. and Vamplew, P. *The Combative Accretion Model - Multiobjective Optimisation Without Explicit Pareto Ranking*. in *Evolutionary Multi-criteria Optimisation: EMO 2005*. 2005.
- [170] Tanaka, M., Watanabe, H., Furukawa, Y., and Tanino, T. *GA-Based Decision Support System for Multicriteria Optimization*. in *Proceedings of the International Conference on Systems, Man, and Cybernetics*. 1995: IEEE.
- [171] Coello, C.A.C., Veldhuizen, D.A.V., and Lamont, G.B., *MOEA Test Suites (Chapter Three)*, in *Evolutionary Algorithms for Solving Multi-Objective Problems*. 2002, Kluwer Academic Publishers.

- [172] Farina, M., Deb, K., and Amato, P. *Dynamic Multiobjective Optimization Problems: Test Cases, Approximation, and Applications*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [173] Huband, S., Hingston, P., White, L., and Barone, L. *An Evolution Strategy with Probabilistic Mutation for Multi-Objective Optimisation*. in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*. 2003: IEEE Press.
- [174] Fieldsend, J.E. and Singh, S. *A Multi-Objective Algorithm based upon Particle Swarm Optimisation, an Efficient Data Structure and Turbulence*. in *Proceedings of the 2002 U.K. Workshop on Computational Intelligence*. 2002.
- [175] Deb, K., Agrawal, S., Pratab, A., and Meyarivan, T., *A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II*. 2000, Indian Institute of Technology, Kanpur, India.
- [176] Coello, C.A.C., *Treating Constraints as Objectives for Single-Objective Evolutionary Optimization*. *Engineering Optimization*, 2000. **32**(3): p. 275-308.
- [177] Coello, C.A.C., *Constraint-handling using an evolutionary multiobjective optimization technique*. *Civil Engineering and Environmental Systems*, 2000. **17**: p. 319-346.
- [178] Aguirre, A.H., Rionda, S.B., Lizarraga, G.L., and Coello, C.A.C. *IS-PAES: A Constraint-Handling Technique Based on Multiobjective Optimization Concepts*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [179] Jensen, M.T. *Guiding Single-Objective Optimization Using Multi-objective Methods*. in *Applications of Evolutionary Computing. Evoworkshops 2003: EvoBIO, EvoCOP, EvoIASP, EvoMUSART, EvoROB, and EvoSTIM*. 2003: Springer. Lecture Notes in Computer Science Vol. 2611.
- [180] Knowles, J.D., Watson, R.A., and Corne, D.W., *Reducing Local Optima in Single-Objective Problems by Multi-objectivization*. *First International Conference on Evolutionary Multi-Criterion Optimization (EMO)*, 2001: p. 268-282.
- [181] Fonseca, C.M. and Fleming, P.J. *Multiobjective Genetic Algorithms*. in *IEE Colloquium on Genetic Algorithms for Control Systems Engineering*. 1993: IEE.
- [182] Deb, K. and Goel, T. *Controlled Elitist Non-dominated Sorting Genetic Algorithms for Better Convergence*. in *First International Conference on Evolutionary Multi-Criterion Optimization*. 2001: Springer-Verlag. Lecture Notes in Computer Science No. 1993.
- [183] Iorio, A.W. and Xiaodong, L. *Rotated Test Problems for Assessing the Performance of Multi-objective Optimization Algorithms*. in *Genetic and Evolutionary Computation Conference (GECCO)*. 2006. Seattle, Washington, USA: ACM Press.
- [184] Iorio, A.W. and Xiaodong, L. *Incorporating Directional Information within a Differential Evolution Algorithm for Multi-objective Optimization*. in *Genetic and Evolutionary Computation Conference (GECCO)*. 2006. Seattle, Washington, USA: ACM Press.

- [185] Deb, K., Chaudhuri, S., and Miettinen, K. *Towards Estimating Nadir Objective Vector Using Evolutionary Approaches*. in *Genetic and Evolutionary Computation Conference (GECCO)*. 2006. Seattle, Washington, USA: ACM Press.
- [186] Hansen, M.P. and Jaszkiewicz, A., *Evaluating the quality of approximations to the non-dominated set*. 1998, Technical University of Denmark.
- [187] Jaszkiewicz, A. *On the computational effectiveness of multiple objective metaheuristics*. in *Proceedings of the Fourth International Conference on Multi-Objective Programming and Goal Programming MOPGP'00. Theory & Applications*. 2000: Springer-Verlag.
- [188] Zitzler, E., Deb, K., and Thiele, L. *Comparison of Multiobjective Evolutionary Algorithms on Test Functions of Different Difficulty*. in *Proceedings of the 1999 Genetic and Evolutionary Computation Conference. Workshop Program*. 1999.
- [189] Mehlhorn, K. and Naher, S., *Dynamic Fractional Cascading*. *Algorithmica*, 1990. **5**: p. 215-241.
- [190] Adelson-Veskiy, G. and Landis, Y., *An Algorithm for the Organization of Information (Translated)*. SSSR (Moscow), 1962. **16**(2): p. 263-266.
- [191] Foster, C.C. *Information Storage and Retrieval Using AVL Trees*. in *ACM 20th National Conference*. 1965.
- [192] Guibas, L.J. and Sedgwick, R., *A Dichromatic Framework for Balanced Trees*. *Proceedings of the 19th Annual Symposium on Foundations of Computer Science*, 1978: p. 8-21.
- [193] Bayer, R., *Symmetric Binary B-Trees: Data Structures and Maintenance Algorithms*. *Acta Informatica*, 1972: p. 290-306.
- [194] Baer, J. and Schwab, B., *A Comparison of Tree-Balancing Algorithms*. *Communications of the ACM*, 1977. **20**(5): p. 322-330.
- [195] Karlton, P.L., Fuller, S.H., Scroggs, R.E., and Kaehler, E.B., *Performance of Height-Balanced Trees*. *Communications of the ACM*, 1976. **19**(1): p. 23-28.
- [196] Chiang, Y. and Tamassia, R., *Dynamic Algorithms in Computational Geometry (Revised Version)*. *Proceedings of IEEE Special Issue on Computational Geometry*, 1992: p. 362-381.
- [197] Cormen, T.H., Leiserson, C.E., Rivest, R.L., and Stein, C., *Amortized Analysis*, in *Introduction to Algorithms*. 2001, MIT Press and McGraw-Hill. p. 405-430.
- [198] Fieldsend, J.E., *Author of the Dominated Tree and multiobjective researcher*, (personal communication: e-mail). Provided source code for Dominated Tree structures, 26/1/06.
- [199] Veldhuizen, D.A.V. and Lamont, G.B., *Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art*. *Evolutionary Computation*, 2000. **8**(2): p. 125-147.
- [200] Deb, K. and Goyal, T., *Multi-Objective Evolutionary Algorithms for Engineering Shape Design*. 2000, Indian Institute of Technology, Kanpur, India.
- [201] Zitzler, E., *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. 1999, Swiss Federal Institute of Technology (ETH).
- [202] Kruskal, W.H. and Wallis, W.A., *Use of Ranks in One-criterion Variance Analysis*. *Journal of the American Statistical Association*, 1952: p. 583-621.



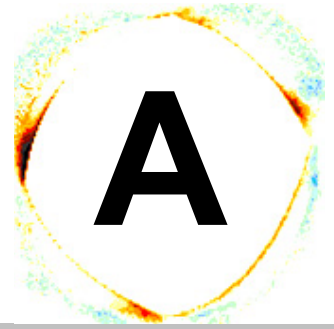
- [203] Conover, W.J., *Practical Nonparametrics Statistics*. Third ed. 1999, New York, New York: John Wiley & Sons.
- [204] Bleuler, S., Laumanns, M., Thiele, L., and Zitzler, E. *PISA — A Platform and Programming Language Independent Interface for Search Algorithms*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [205] Fonesca, C., Knowles, J.D., Thiele, L., and Zitzler, E., *PISA Performance Assessment* (<http://www.tik.ee.ethz.ch/sop/pisa/?page=assessment.php>). 2007, The Swiss Federal Institute of Technology (Zurich) - Computer Engineering and Networks Laboratory.
- [206] Knowles, J.D., Corne, D.W., and Fleischer, M. *Bounded Archiving using the Lebesgue Measure*. in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*. 2003: IEEE Press.
- [207] Veldhuizen, D.A.V., Zydallis, J.B., and Lamont, G.B., *Considerations in Engineering Parallel Multiobjective Evolutionary Algorithms*. IEEE Transactions on Evolutionary Computation, 2003. 7(2): p. 144-173.
- [208] Silverman, B.W., *Density Estimation For Statistics and Data Analysis*, in *School of Mathematics University of Bath, UK*. 1986.
- [209] Scott, D.W. and Sain, S.R., *Multi-dimensional Density Estimation*. Handbook of Statistics - Data Mining and Computational Statistics, 2004. 24: p. 229-261.
- [210] Lu, H. and Yen, G.G. *Dynamic Population Size in Multiobjective Evolutionary Algorithms*. in *Congress on Evolutionary Computation (CEC 2002)*. 2002: IEEE Service Center.
- [211] Moreno-Seco, F., Mico, L., and Oncina, J. *A Fast Approximately K-Nearest-Neighbour Search Algorithm for Classification Tasks*. in *Proceedings of the Joint IAPR International Workshops on Advances in Pattern Recognition*. 2000.
- [212] Liao, Y. and Vemuri, V.R., *Use of K-Nearest Neighbor Classifier for Intrusion Detection*. Computers and Security, 2002: p. 439-448.
- [213] Denoeux, T., *A k-Nearest Neighbor Classification Rule Based on Dempster-Shafer Theory*. IEEE Transactions on Systems, Man and Cybernetics, 1995.
- [214] Mugica, F. and Nebot, A., *A Specialization of the K-Nearest Neighbor Classification Rule for the Prediction of Dynamical Systems Using FIR*. Advances in Artificial Intelligence and Engineering Cybernetics, 1996: p. 130-136.
- [215] Borges, C.C.H. and Barbosa, H.J.C. *A Non-generational Genetic Algorithm for Multiobjective Optimization*. in *2000 Congress on Evolutionary Computation*. 2000: IEEE Service Center.
- [216] Veldhuizen, D.A.V. and Lamont, G.B. *On Measuring Multiobjective Evolutionary Algorithm Performance*. in *2000 Congress on Evolutionary Computation*. 2000: IEEE Service Center.
- [217] Deb, K., *Multi-Objective Optimization using Evolutionary Algorithms*. 2001: John Wiley & Sons.
- [218] Knowles, J.D., Thiele, L., and Zitzler, E. *A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers*. 2005.
- [219] Knowles, J. and Corne, D. *On Metrics for Comparing Nondominated Sets*. in *Congress on Evolutionary Computation (CEC 2002)*. 2002: IEEE Service Center.

- [220] Zitzler, E., Laumanns, M., Thiele, L., Fonseca, C.M., and da Fonseca, V.G. *Why Quality Assessment of Multiobjective Optimizers Is Difficult*. in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2002)*. 2002: Morgan Kaufmann Publishers.
- [221] Fleischer, M. *The Measure of Pareto Optima. Applications to Multi-objective Metaheuristics*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [222] Veldhuizen, D.A.V. and Lamont, G.B. *Multiobjective Optimization with Messy Genetic Algorithms*. in *Proceedings of the 2000 ACM Symposium on Applied Computing*. 2000: ACM.
- [223] Coello, C.A.C. and Cortes, N.C. *An Approach to Solve Multiobjective Optimization Problems Based on an Artificial Immune System*. in *First International Conference on Artificial Immune Systems (ICARIS 2002)*. 2002: University of Kent at Canterbury, UK.
- [224] da Fonseca, V.G., Fonseca, C.M., and Hall, A.O. *Inferential Performance Assessment of Stochastic Optimisers and the Attainment Function*. in *First International Conference on Evolutionary Multi-Criterion Optimization*. 2001: Springer-Verlag. Lecture Notes in Computer Science No. 1993.
- [225] Deb, K. and Agrawal, R.B., *Simulated Binary Crossover for Continuous Search Space*. *Complex Systems*, 1995. **9**: p. 115 - 148.
- [226] Sun, D., Benekohal, R.F., and Waller, S.T. *Multi-objective Traffic Signal Timing Optimization Using Non-dominated Sorting Genetic Algorithm II*. in *Genetic and Evolutionary Computation — GECCO 2003. Proceedings, Part II*. 2003: Springer. Lecture Notes in Computer Science Vol. 2724.
- [227] Kasat, R.B., Kunzru, D., Saraf, D.N., and Gupta, S.K., *Multiobjective Optimization of Industrial FCC Units using Elitist Nondominated Sorting Genetic Algorithm*. *Industrial & Engineering Chemistry Research*, 2002. **41**(19): p. 4765-4776.
- [228] Barreto, W.J., Price, R.K., Solomatine, D.P., and Vojinovic, Z. *Approaches to Multi-Objective Multi-Tier Optimization in Urban Drainage Planning*. in *7th International Conference on Hydroinformatics (HIC 2006)*. 2006. Nice, France.
- [229] Shie-Yui, L., Al-Fayyaz, T.A., and Sai, L.K., *Application of Evolutionary Algorithm in Reservoir Operations*. *Journal of The Institution of Engineers, Singar*, 2004. **44**.
- [230] Deb, K., Pratab, A., and Moitra, S. *Mechanical Component Design for Multiple Objectives Using Elitist Non-dominated Sorting GA*. in *Proceedings of the Parallel Problem Solving from Nature VI Conference*. 2000: Springer. Lecture Notes in Computer Science No. 1917.
- [231] Deviredy, V. and Reed, P. *An Efficient Design Methodology for the Nondominated Sorted Genetic Algorithm-II*. in *2003 Genetic and Evolutionary Computation Conference. Late-Breaking Papers*. 2003: AAAI.
- [232] Zitzler, E., Deb, K., and Thiele, L., *Comparison of Multiobjective Evolutionary Algorithms: Empirical Results*. 1999, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich.
- [233] Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. *Scalable Multi-Objective Optimization Test Problems*. in *Congress on Evolutionary Computation (CEC 2002)*. 2002: IEEE Service Center.

- [234] Veldhuizen, D.A.V. and Lamont, G.B. *MOEA Test Suite Generation, Design & Use*. in *Proceedings of the 1999 Genetic and Evolutionary Computation Conference. Workshop Program*. 1999.
- [235] Farhang-Mehr, A. and Azarm, S. *Minimal Sets of Quality Metrics*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [236] Jin, W. and Azarm, S., *Metrics for Quality Assessment of a Multiobjective Design Optimization Solution Set*. Transactions of the ASME, Journal of Mechanical Design, 2001. **123**: p. 18-25.
- [237] Okabe, T., Jin, Y., and Sendhoff, B. *A Critical Survey of Performance Indices for Multi-Objective Optimization*. in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*. 2003: IEEE Press.
- [238] Shu, L.-S., Shinn-Jang, H.o., Shinn-Ying, H.o., Chen, J.-H., and Hung, M.-H. *A Novel Multi-objective Orthogonal Simulated Annealing Algorithm for solving Multi-objective Optimization Problems with a Large Number of Parameters*. in *Genetic and Evolutionary Computation — GECCO 2004. Proceedings of the Genetic and Evolutionary Computation Conference. Part I*. 2004: Springer-Verlag, Lecture Notes in Computer Science Vol. 3102.
- [239] Suppaitnarm, A., Seffen, K.A., Parks, G.T., and Clarkson, P.J., *A simulated annealing algorithm for multiobjective optimization*. Engineering Optimization, 2000. **33**(1): p. 59-85.
- [240] Czyzak, P. and Jaszkievicz, A., *Pareto simulated annealing — a metaheuristic technique for multiple-objective combinatorial optimization*. Journal of Multi-Criteria Decision Analysis, 1998. **7**: p. 34-47.
- [241] Suman, B., *Study of simulated annealing based algorithms for multiobjective optimization of a constrained problem*. Computers & Chemical Engineering, 2004. **28**: p. 1849-1871.
- [242] Doerner, K., Gutjahr, W.J., Hartl, R.F., Strauss, C., and Stummer, C., *Pareto Ant Colony Optimization: A Metaheuristic Approach to Multiobjective Portfolio Selection*. Annals of Operations Research, 2004. **131**: p. 79-99.
- [243] Romero, C.E.M. and Manzanares, E.M. *MOAQ an Ant-Q Algorithm for Multiple Objective Optimization Problems*. in *Genetic and Evolutionary Computing Conference (GECCO 99)*. 1999: Morgan Kaufmann.
- [244] Iredi, S., Merkle, D., and Middendorf, M. *Bi-Criterion Optimization with Multi Colony Ant Algorithms*. in *First International Conference on Evolutionary Multi-Criterion Optimization*. 2001: Springer-Verlag. Lecture Notes in Computer Science No. 1993.
- [245] Coello, C.A.C., Pulido, G.T., and Lechuga, M.S., *Handling Multiple Objectives With Particle Swarm Optimization*. IEEE Transactions on Evolutionary Computation, 2004. **8**(3): p. 256-279.
- [246] Mostaghim, S. and Teich, J. *Strategies for Finding Good Local Guides in Multi-objective Particle Swarm Optimization (MOPSO)*. in *2003 IEEE Swarm Intelligence Symposium Proceedings*. 2003: IEEE Service Center.
- [247] Berry, A. and Vamplew, P. *A Simplified Artificial Life Model for Multiobjective Optimisation: A Preliminary Report*. in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*. 2003: IEEE Press.
- [248] Laumanns, M., Rudolph, G., and Schwefel, H.. *A Spatial Predator-Prey Approach to Multi-Objective Optimization: A Preliminary Study*. in *Parallel Problem Solving From Nature — PPSN V*. 1998: Springer-Verlag.

- [249] Knowles, J.D. and Corne, D.W. *A Comparison of Diverse Approaches to Memetic Multiobjective Combinatorial Optimization*. in *Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program*. 2000.
- [250] Jaszekiewicz, A., *A comparative study of multiple-objective metaheuristics on the bi-objective set covering problem and the Pareto memetic algorithm*. 2001, Institute of Computing Science, Poznan University of Technology.
- [251] Ishibuchi, H., Yoshida, T., and Murata, T., *Balance Between Genetic Search and Local Search in Memetic Algorithms for Multiobjective Permutation Flowshop Scheduling*. IEEE Transactions on Evolutionary Computation, 2003. **7**(2): p. 204-223.
- [252] Zydallis, J.B., Lamont, G.B., and Veldhuizen, D.A.V. *Messy Genetic Algorithm Based Multi-Objective Optimization: A Comparative Statistical Analysis*. in *PPSN/SAB Workshop on Multiobjective Problem Solving from Nature (MPSN)*. 2000.
- [253] Day, R.O., Kleeman, M.P., and Lamont, G.B. *Multi-Objective fast messy Genetic Algorithm Solving Deception Problems*. in *2004 Congress on Evolutionary Computation (CEC 2004)*. 2004: IEEE Service Center.
- [254] Veldhuizen, D.A.V. and Lamont, G.B. *Genetic Algorithms, Building Blocks, and Multiobjective Optimization*. in *Proceedings of the 1999 Genetic and Evolutionary Computation Conference. Workshop Program*. 1999.
- [255] de Toro, F., Ortega, J., Fernandez, J., and Diaz, A. *PSFGA: A Parallel Genetic Algorithm for Multiobjective Optimization*. in *10th Euromicro Workshop on Parallel, Distributed and Network-Based Processing*. 2002: IEEE.
- [256] Jaszekiewicz, A., *On the Computational Efficiency of Multiple Objective Metaheuristics. The Knapsack Problem Case Study*. European Journal of Operational Research, 2004. **158**(2): p. 418-433.
- [257] Gandibleux, X. and Freville, A., *Tabu Search Based Procedure for Solving the 0-1 Multi-Objective Knapsack Problem: The Two Objectives Case*. Journal of Heuristics, 2000. **6**(3): p. 361-383.
- [258] Grosan, C. *Improving the Performance of Evolutionary Algorithms for the Multiobjective 0/1 Knapsack Problem using  $\epsilon$ -dominance*. in *2004 Congress on Evolutionary Computation (CEC 2004)*. 2004: IEEE Service Center.
- [259] Yan, Z., Zhang, L., Kang, L., and Lin, G. *A New MOEA for Multi-objective TSP and Its Convergence Property Analysis*. in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. 2003: Springer. Lecture Notes in Computer Science. Volume 2632.
- [260] Murata, T., Ishibuchi, H., and Tanaka, H., *Multi-Objective Genetic Algorithm and Its Application to Flowshop Scheduling*. Computers and Industrial Engineering, 1996. **30**(4): p. 957-968.

# Appendix

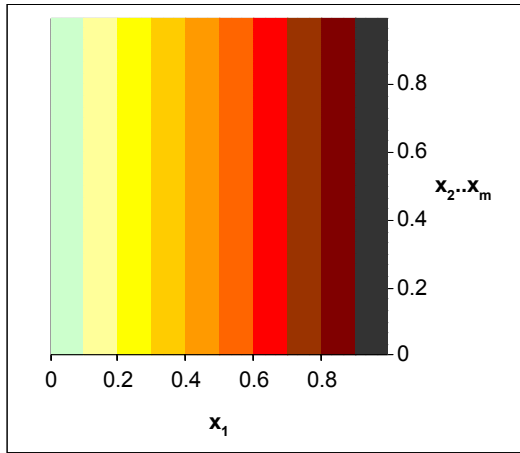
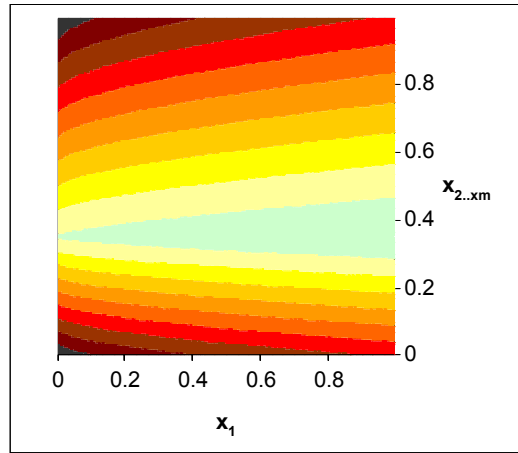
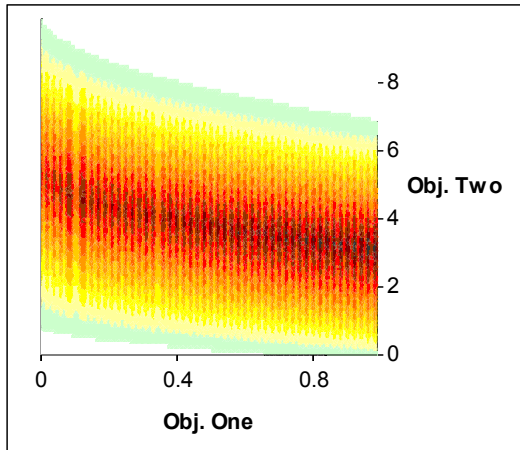


Test Problem

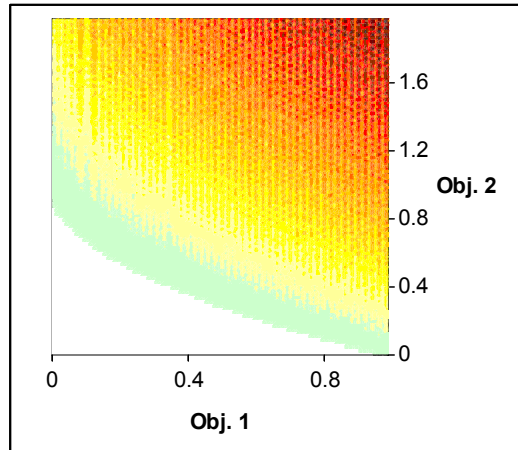
Appendix

# A TEST PROBLEM APPENDIX

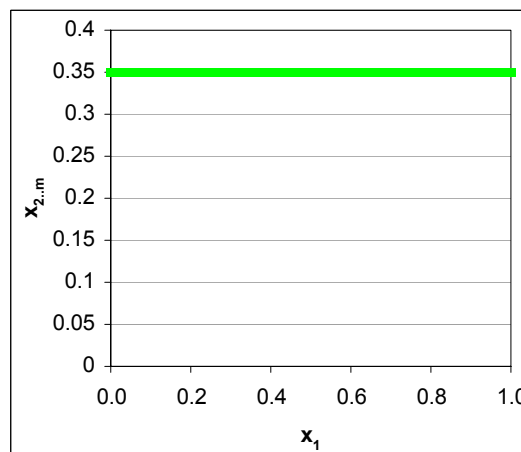
## A.1 EXAMINING *AP-1*


 (a) Solution-Space  $\rightarrow$  Objective-One

 (b) Solution-Space  $\rightarrow$  Objective-Two


(c) Objective-Space



(d) Objective-Space (Pareto Front)

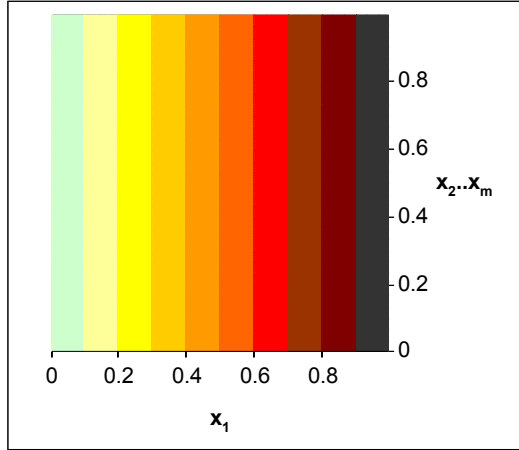


(e) Pareto Optimal Solution Values

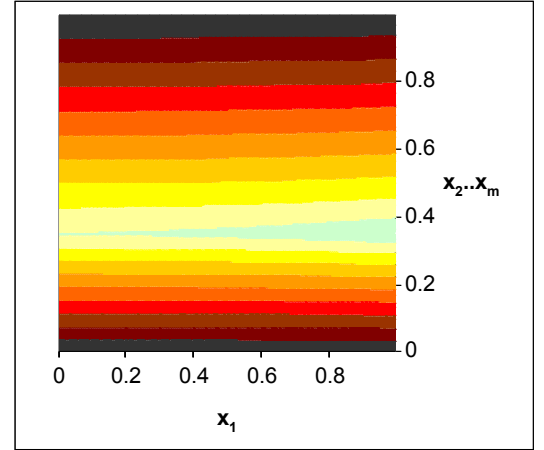
**Figure A1 — Problem Characteristics for *AP-1***

Objective-one is optimal when  $x_1 = 0$ ; objective-two is optimal when  $x_{2..m} = 0.35$  and  $x_1 = 1$ . Note that the Pareto optimal front is convex — with extreme points of (0,1) and (1,0). The optimal front is formed by setting  $x_{2..m}$  to 0.35 and varying  $x_1$ .  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .

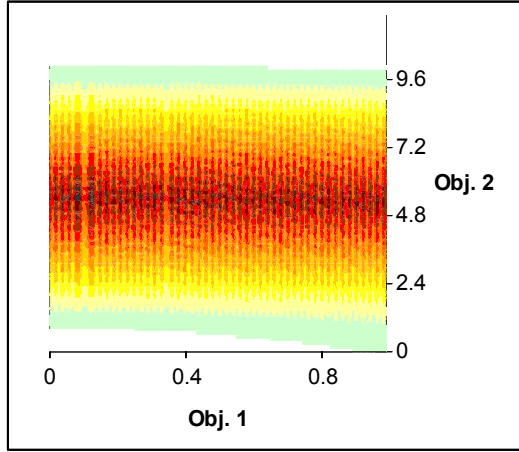
## A.2 EXAMINING *AP-2*



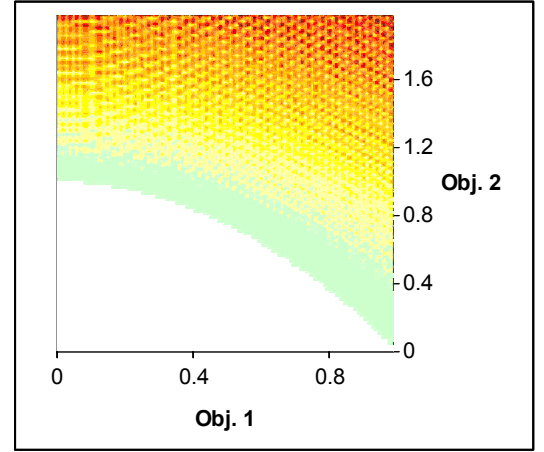
(a) Solution-Space  $\rightarrow$  Objective-One



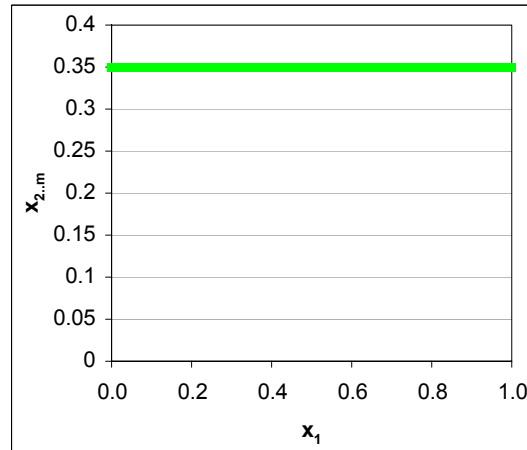
(b) Solution-Space  $\rightarrow$  Objective-Two



(c) Objective-Space



(d) Objective-Space (Pareto Front)



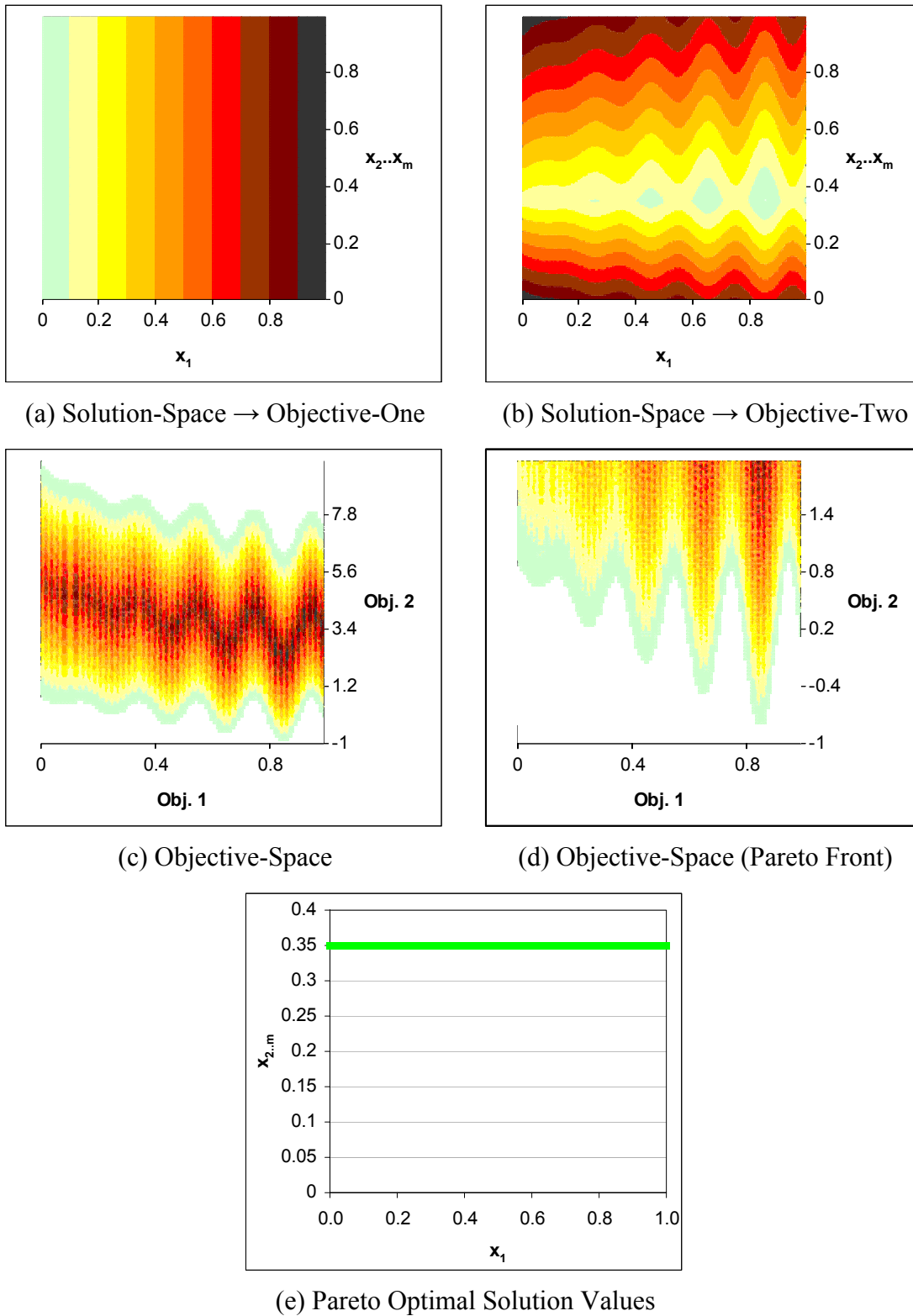
(e) Pareto Optimal Solution Values

**Figure A2 — Problem Characteristics for *AP-2***

Note that objective-one is optimal when  $x_1 = 0$ ; objective-two is optimal when  $x_{2..m} = 0.35$  and  $x_1 = 1$ .

The Pareto optimal front is concave — with extreme points of  $(0,1)$  and  $(1,0)$  — and is formed by setting  $x_{2..m}$  to 0.35 and varying  $x_1$ .  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .

### A.3 EXAMINING *AP-3*

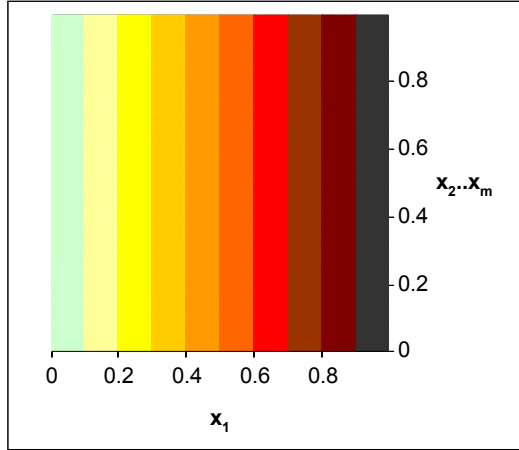


**Figure A3 — Problem Characteristics for *AP-3***

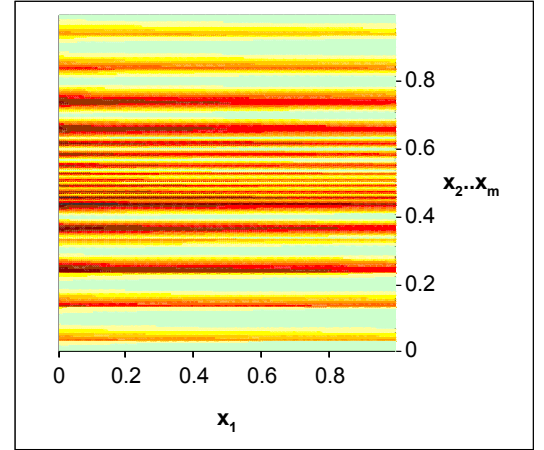
Note that objective-one is optimal when  $x_1 = 0$ ; objective-two is optimal when  $x_{2..m} = 0.35$  and  $x_1 = 0.85$ . The Pareto optimal front is a disconnected series of convex curves — with extreme points of  $(0,1)$  and  $(0.85,-0.77)$  — and is formed by setting  $x_{2..m}$  to 0.35 and varying  $x_1$ .  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .



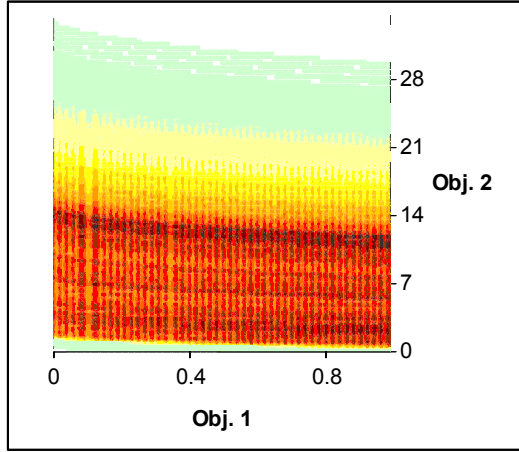
## A.4 EXAMINING *AP-4*



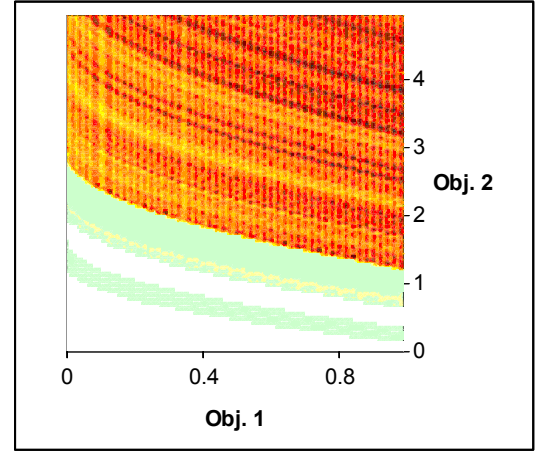
(a) Solution-Space  $\rightarrow$  Objective-One



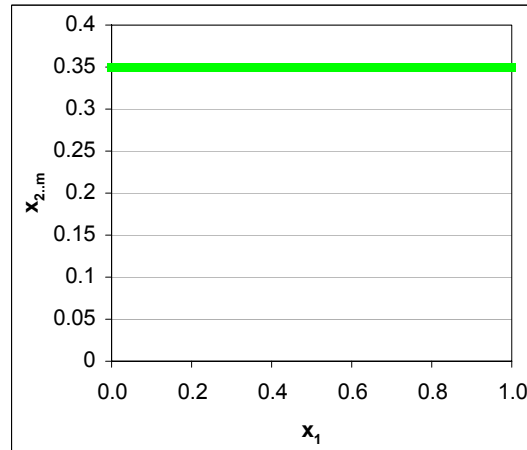
(b) Solution-Space  $\rightarrow$  Objective-Two



(c) Objective-Space



(d) Objective-Space (Pareto Front)

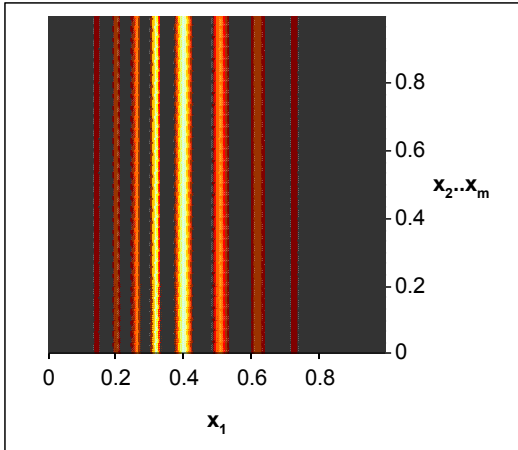


(e) Pareto Optimal Solution Values

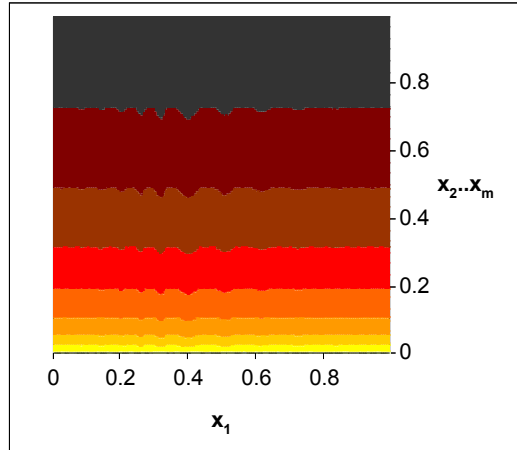
**Figure A4 — Problem Characteristics for *AP-4***

Objective-one is optimal when  $x_1 = 0$ ; objective-two is optimal when  $x_{2..m} = 0.35$  and  $x_1 = 1$ . Both (a) and (b) feature normalised solution values for  $x_{2..m}$ . Note that the Pareto optimal front is convex — with extreme points of (0,1) and (1,0) — and is formed by setting  $x_{2..m}$  to 0.35 and varying  $x_1$ . (d) shows the presence of at least two well-defined and disseminated false optimal fronts.  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .

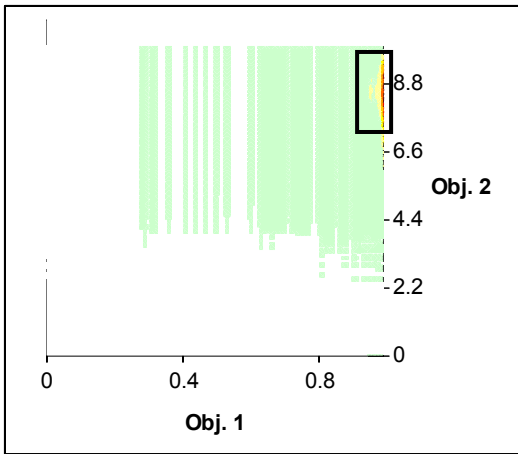
## A.5 EXAMINING *AP-5*



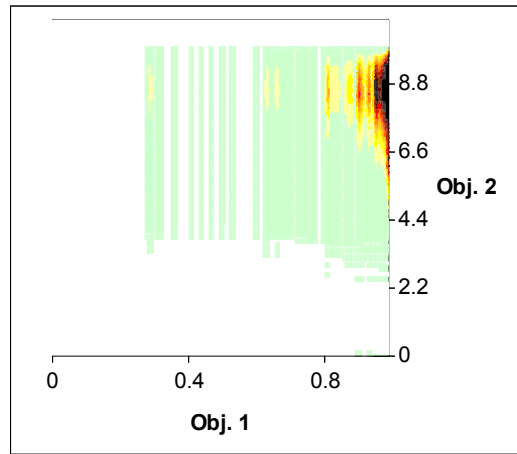
(a) Solution-Space → Objective-One



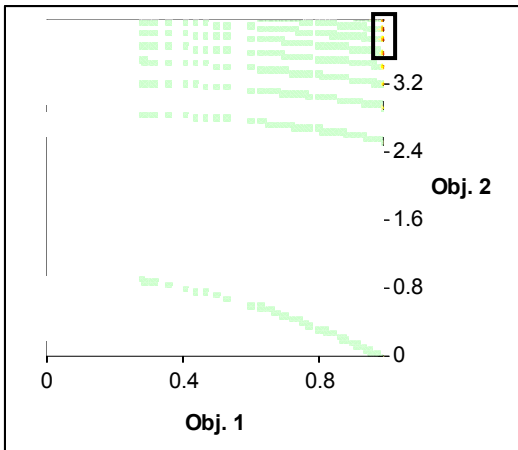
(b) Solution-Space → Objective-Two



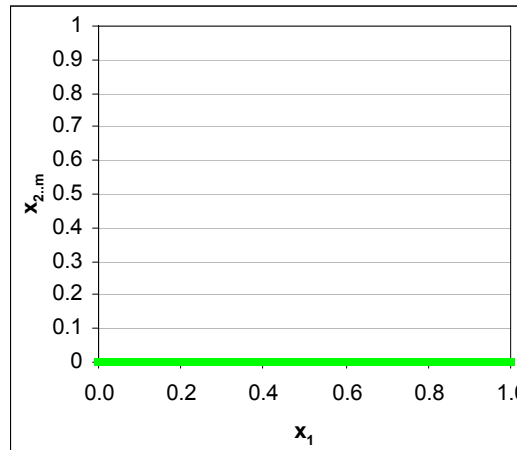
(c) Objective-Space



(d) Objective-Space (New Palette)



(e) Obj-Space (Pareto Front; New Palette)

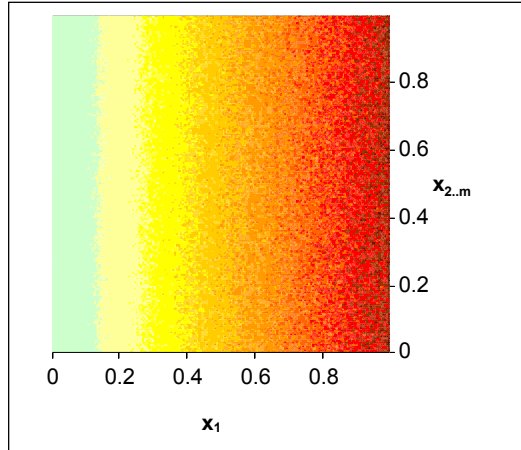


(f) Pareto Optimal Solution Values

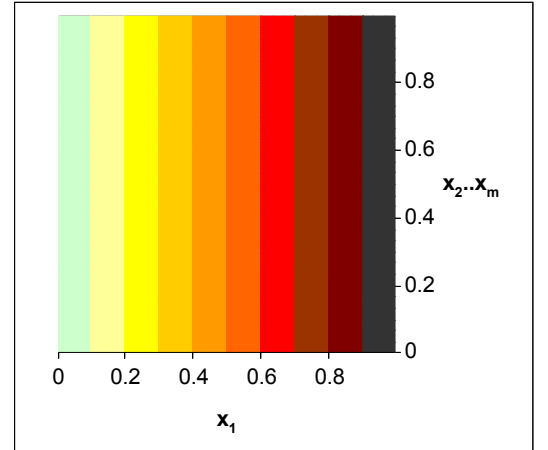
**Figure A5 — Problem Characteristics for *AP-5***

Objective-one is optimal when  $x_1 = 0.35$ ; objective-two is optimal when  $x_{2..m} = 0$  and  $x_1 = 1$ . Both (a) and (b) feature normalised solution values for  $x_{2..m}$ . The Pareto optimal front is concave — with extreme points of  $(0.28, 0.92)$  and  $(1, 0)$  — and is formed by setting  $x_{2..m}$  to 0 and varying  $x_1$ . Note how poorly populated the optimal front is in (e) and how isolated it is from the remainder of the heavily biased space. (d) normalises frequencies in the range of 0% - 10%, any frequency  $\geq 10\%$  is black. The boxes in (c) and (e) highlight biased regions.  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .

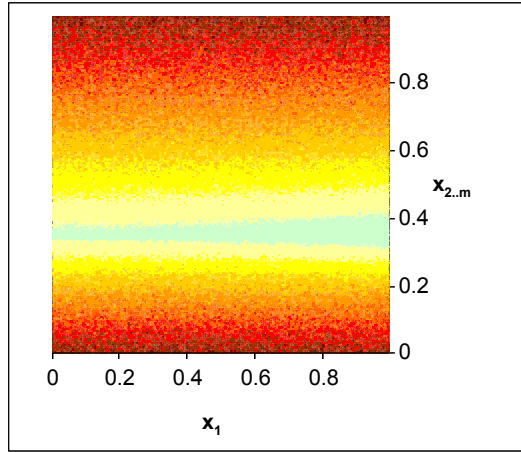
## A.6 EXAMINING *AP-6*



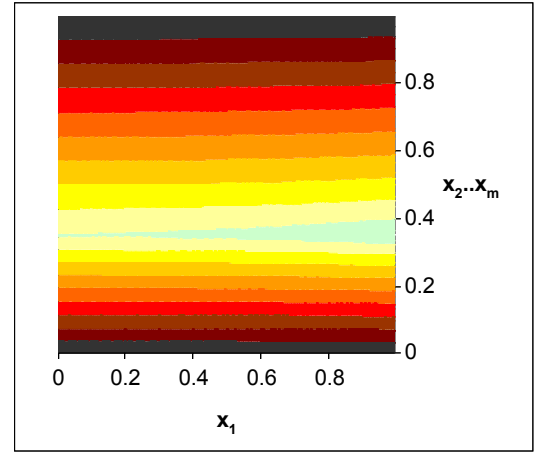
(a) Apparent Solution-Space → Obj. One



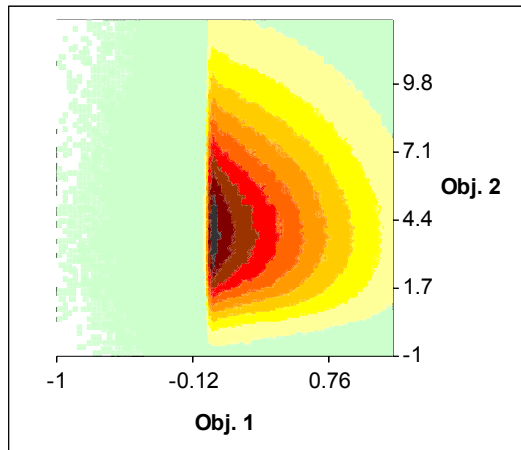
(b) True Solution-Space → Obj. One



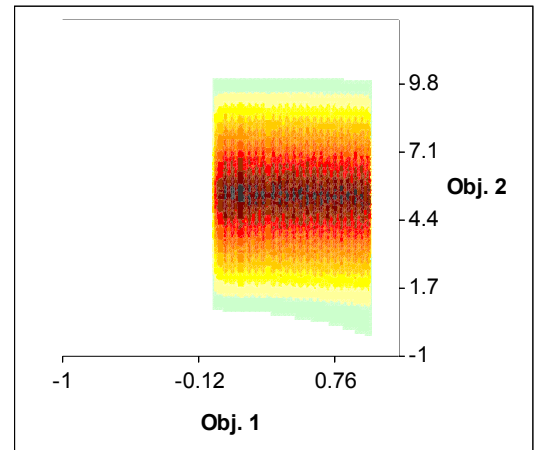
(c) Apparent Solution-Space → Obj. Two



(d) True Solution-Space → Obj. Two



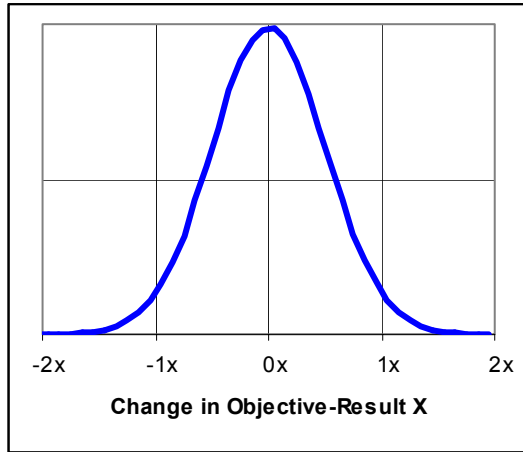
(e) Apparent Objective-Space



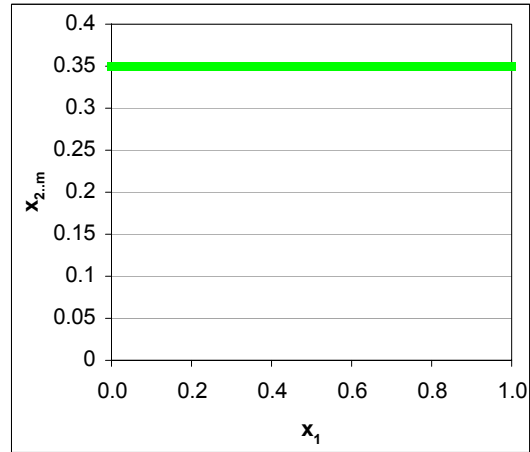
(f) True Objective-Space

**Figure A6 — Problem Characteristics for *AP-6***

(a), (c) and (e) illustrate the distorted search- and objective-spaces that a search algorithm will encounter under the presence of the noisy Gaussian function. (b), (d) and (f) each depict the true spaces that the noise obscures. As per *AP-2*, the true Pareto optimal front is a concave curve with extreme points of (0,1) and (1,0). Objective-one is optimal when  $x_1 = 0$ ; objective-two is optimal when  $x_{2..m} = 0.35$  and  $x_1 = 1$ .  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .



(a) Distribution of Noise in *AP-6*

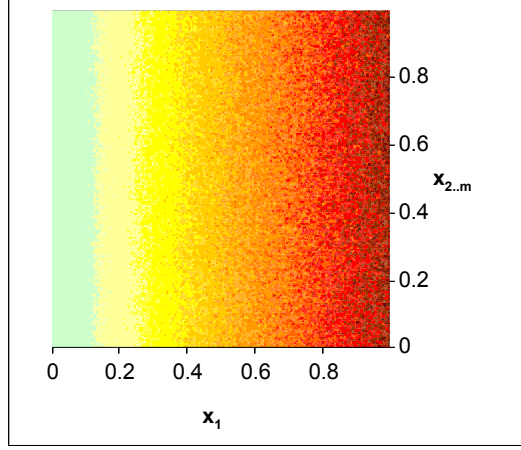
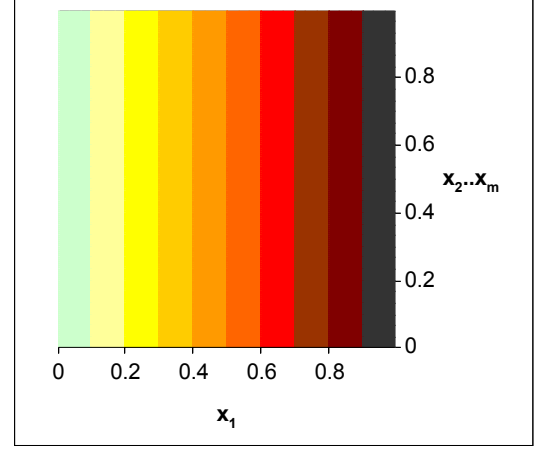
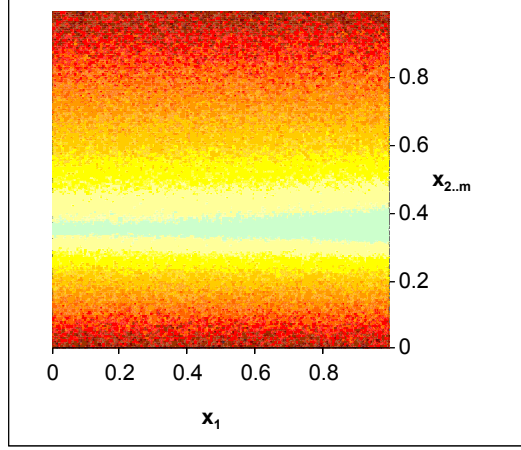
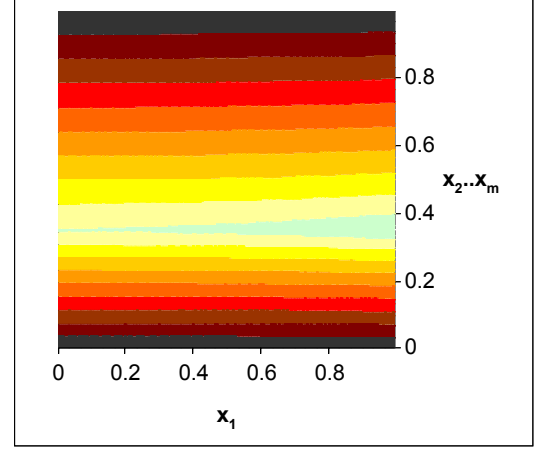
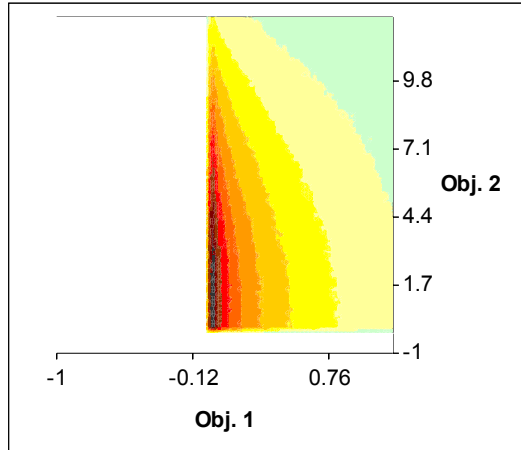


(e) Pareto Optimal Solution Values

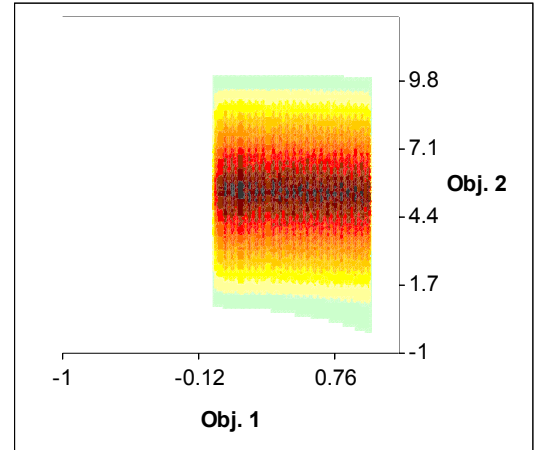
**Figure A7 — Problem Characteristics for *AP-6***

(a) indicates the type of objective-space noise encountered in *AP-6* — over 95% of all noise causes original objective-scores to lie in the range of  $[0, 2x]$ , while over 65% results in values lying in  $[0.5x, 1.5x]$ . (e) indicates that the concave Pareto optimal front is formed by setting  $x_{2..m}$  to 0.35 and varying  $x_1$ .

## A.7 EXAMINING *AP-7*


 (a) Apparent Solution-Space  $\rightarrow$  Obj. One

 (b) True Solution-Space  $\rightarrow$  Obj. One

 (c) Apparent Solution-Space  $\rightarrow$  Obj. Two

 (d) True Solution-Space  $\rightarrow$  Obj. Two


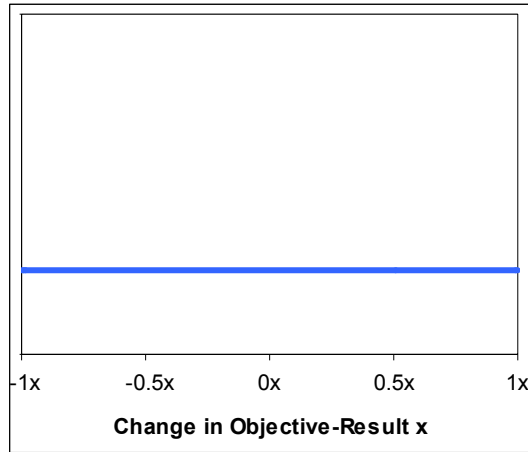
(e) Apparent Objective-Space



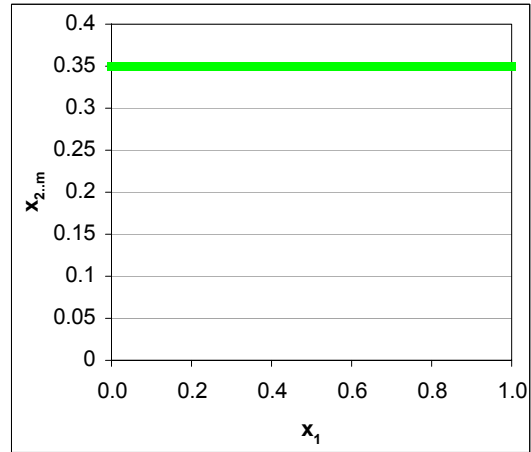
(f) True Objective-Space

**Figure A8 — Problem Characteristics for *AP-7***

(a), (c) and (e) illustrate the distorted search- and objective-spaces that a search algorithm will encounter under the presence of the noisy non-Gaussian function. (b), (d) and (f) each depict the true spaces that the noise obscures. As per *AP-2*, the true Pareto optimal front is a concave curve with extreme points of (0,1) and (1,0). Objective-one is optimal when  $x_1 = 0$ ; objective-two is optimal when  $x_{2..m} = 0.35$  and  $x_1 = 1$ .  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .



(a) Distribution of Noise in *AP-6*

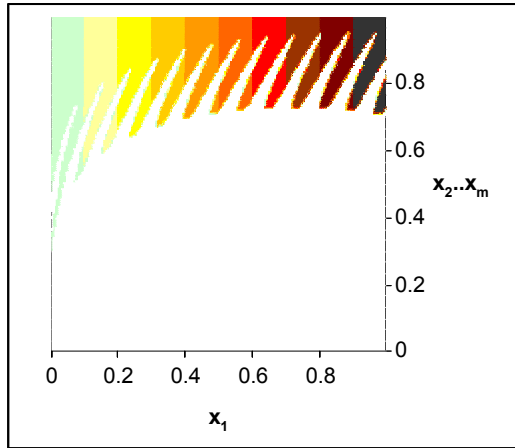
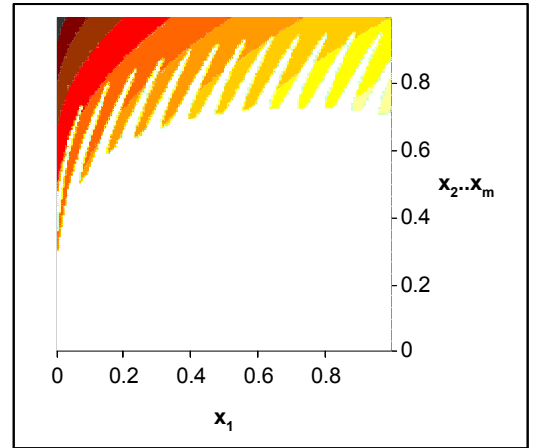
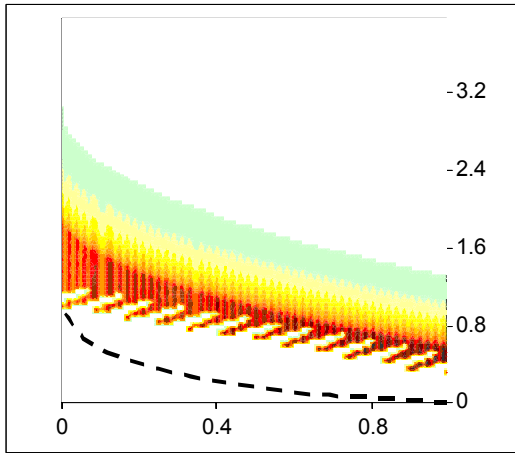


(e) Pareto Optimal Solution Values

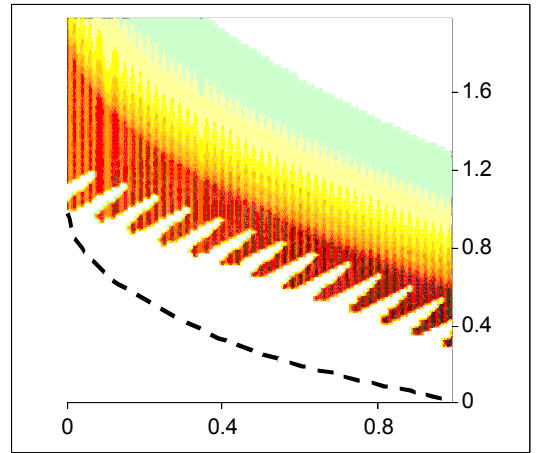
**Figure A9 — Problem Characteristics for *AP-7***

(a) indicates the type of objective-space noise encountered in *AP-7* — original objective-scores are recast such that there is an even likelihood of lying anywhere in the range  $[0, 2x]$ . (e) indicates that the concave Pareto optimal front is formed by setting  $x_{2,m}$  to  $0.35$  and varying  $x_1$ .

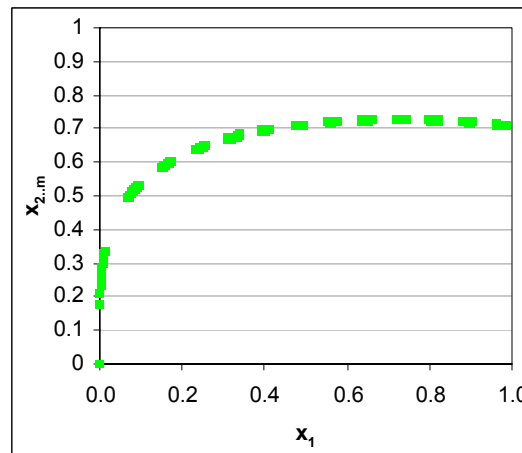
## A.8 EXAMINING *AP-8*


 (a) Solution-Space  $\rightarrow$  Objective-One

 (b) Solution-Space  $\rightarrow$  Objective-Two


(c) Objective-Space



(d) Objective-Space



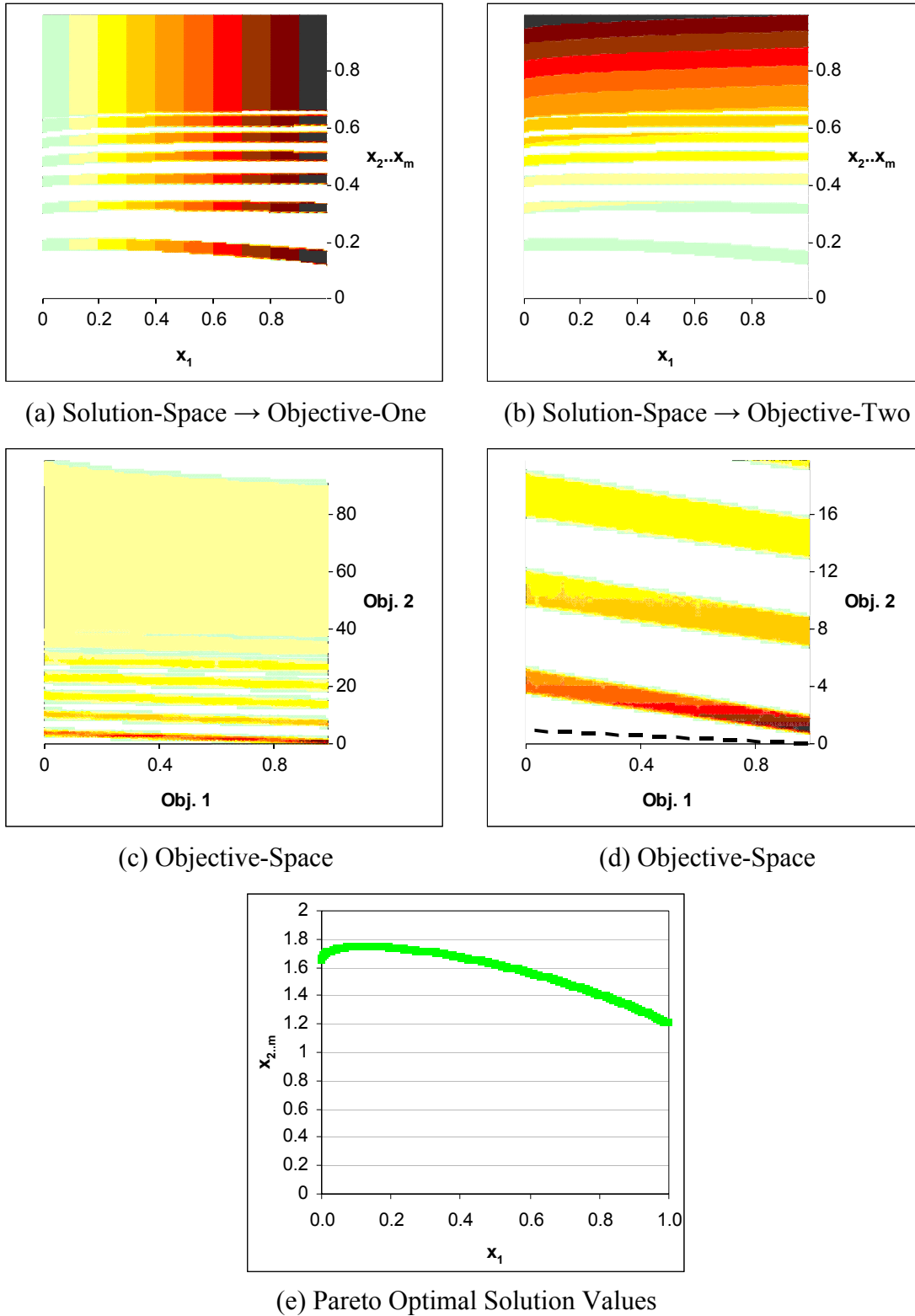
(e) Pareto Optimal Solution Values

**Figure A10 — Problem Characteristics for *AP-8***

Objective-one is optimal when  $x_1 = 0$ ; objective-two is optimal when  $x_{2..m} = 0.71$  and  $x_1 = 0.984$ . Note that the feasible Pareto optimal front is a disconnected line — with extreme points of  $(0,1)$  and  $(1,0)$ .

The dashed convex curve in (c) and (d) represents the infeasible optimal front.  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .

## A.9 EXAMINING *AP-9*

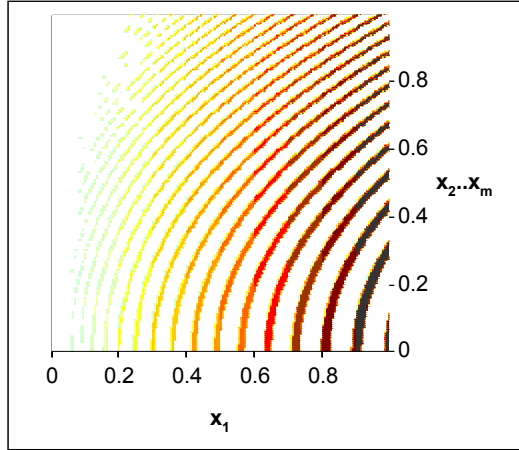
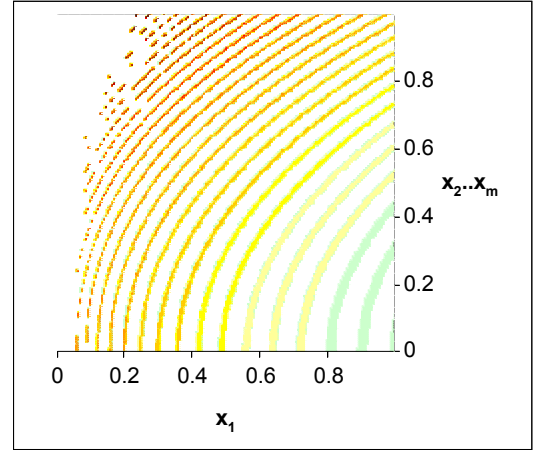
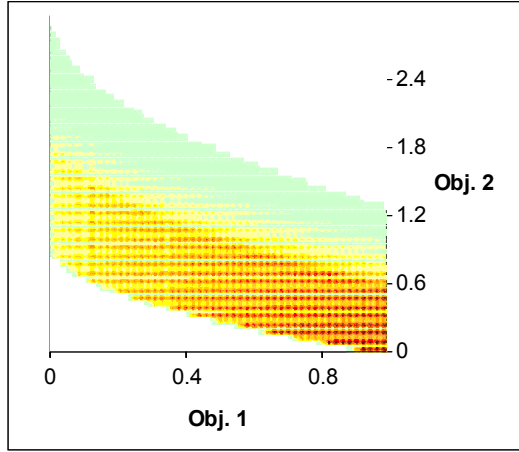


**Figure A11 — Problem Characteristics for *AP-9***

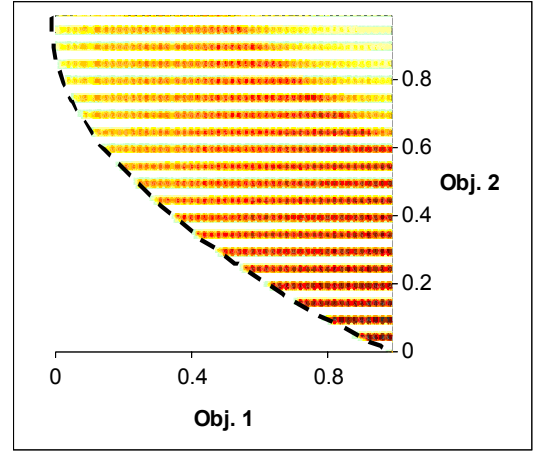
Objective-one is optimal when  $x_1 = 0$ ; objective-two is optimal when  $x_{2..m} = 1.2$  and  $x_1 = 1$ . Both (a) and (b) feature normalised solution values for  $x_{2..m}$ . The feasible Pareto optimal front is a continuous line — with extreme points of (0,3.7) and (1,0.88). The dashed convex curve in (d) represents the infeasible optimal front.  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .



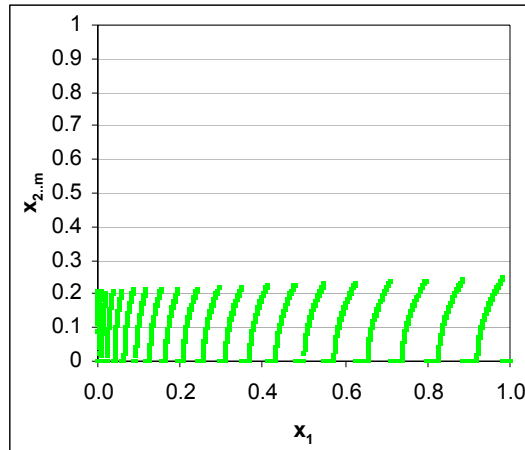
## A.10 EXAMINING *AP*-10


 (a) Solution-Space  $\rightarrow$  Objective-One

 (b) Solution-Space  $\rightarrow$  Objective-Two


(c) Objective-Space



(d) Objective-Space

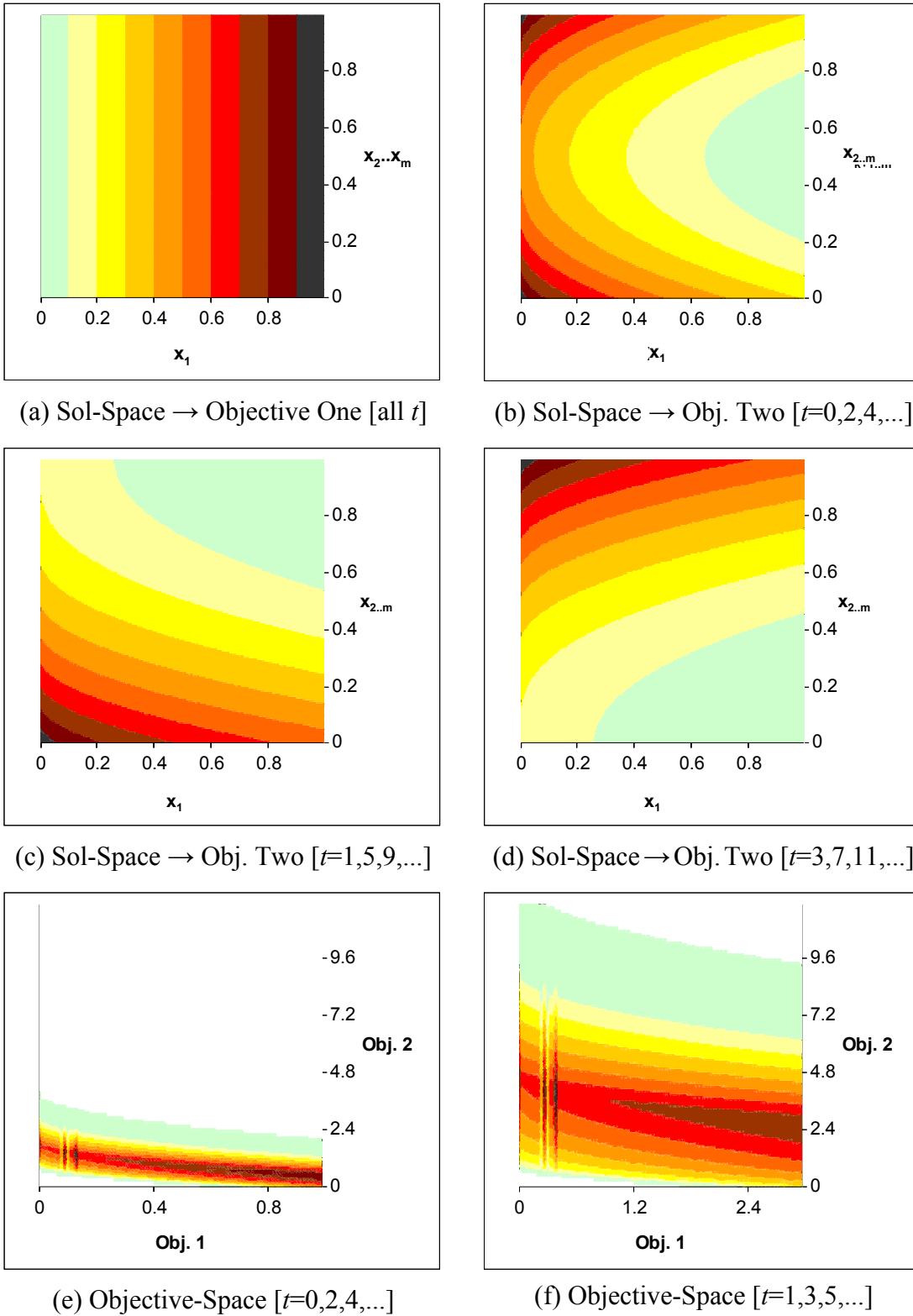


(e) Pareto Optimal Solution Values

**Figure A12 — Problem Characteristics for *AP*-10**

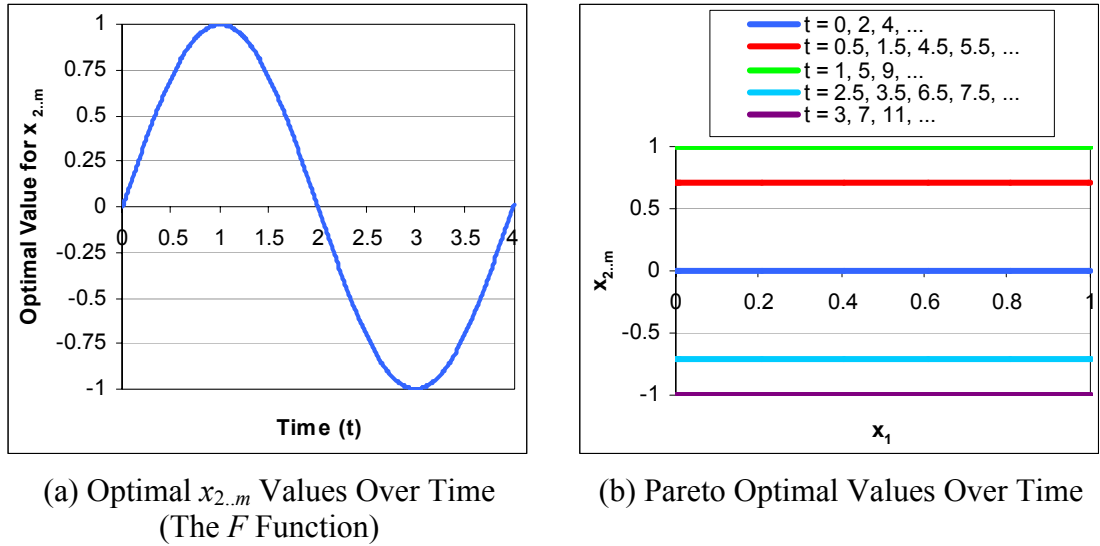
Objective-one is optimal when  $x_1 = 0$ ; objective-two is optimal when  $x_{2..m} = 0$  and  $x_1 = 1$ . The feasible Pareto optimal front is a discontinuous line — with extreme points of (0,1) and (1,0). The dashed convex curve in (d) represents the infeasible optimal front.  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .

## A.11 EXAMINING *AP-11*



**Figure A13 — Problem Characteristics for *AP-11***

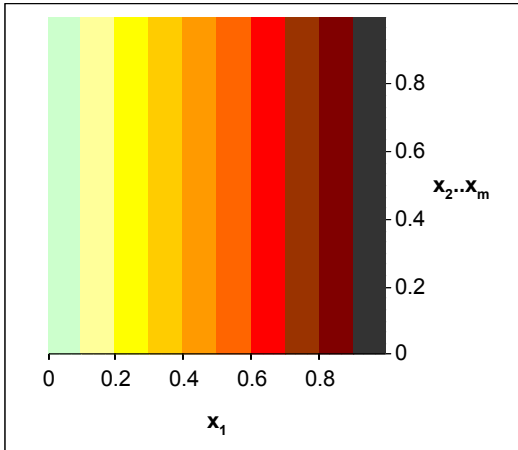
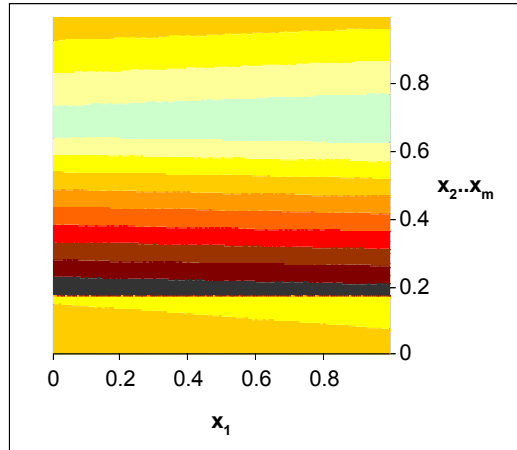
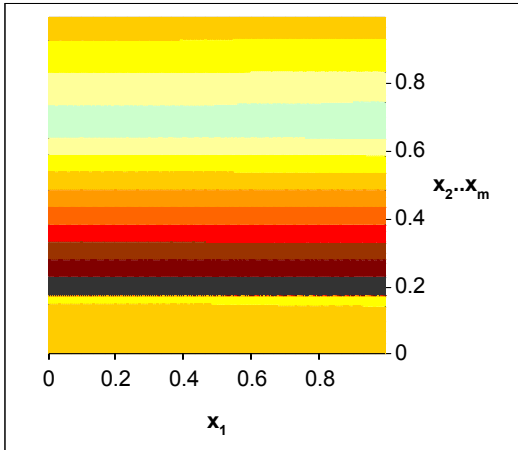
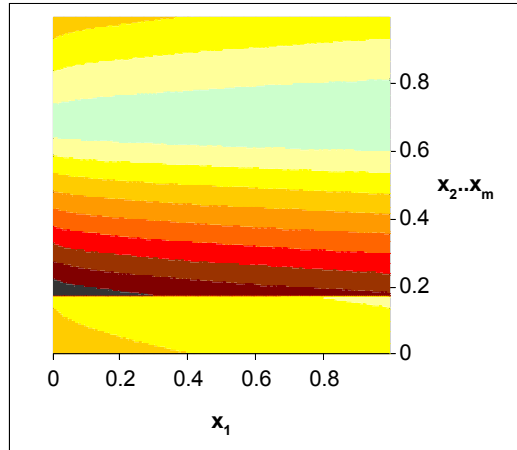
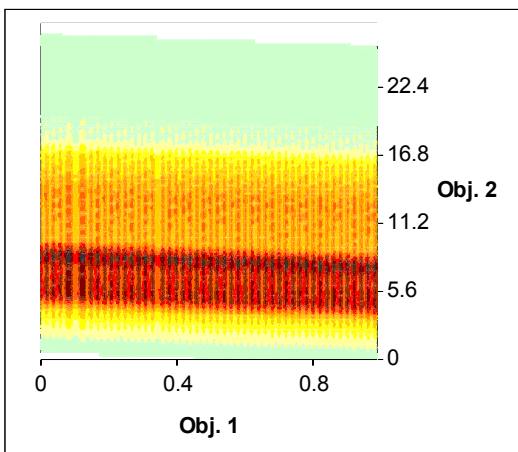
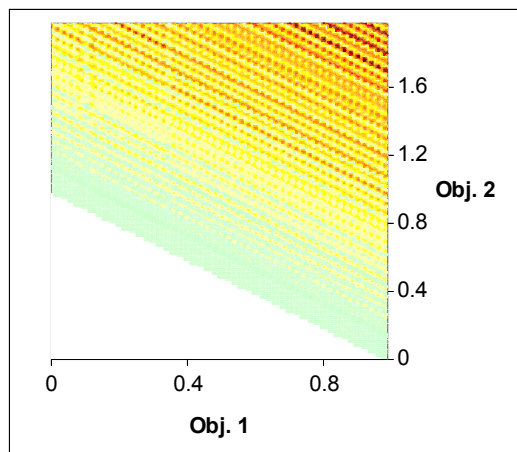
The solution-space for objective-one is static; the solution-space for objective-two changes over time, with a moving optimal region. The Pareto optimal front is a static convex curve — with extreme points of (0,1) and (1,0) — embedded in a variably sized objective-space (with size oscillating between (e) and (f)). Note that (a), (b), (c) and (d) feature normalised values for  $x_{2..m}$ .  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .



**Figure A14 — Problem Characteristics for AP-11**

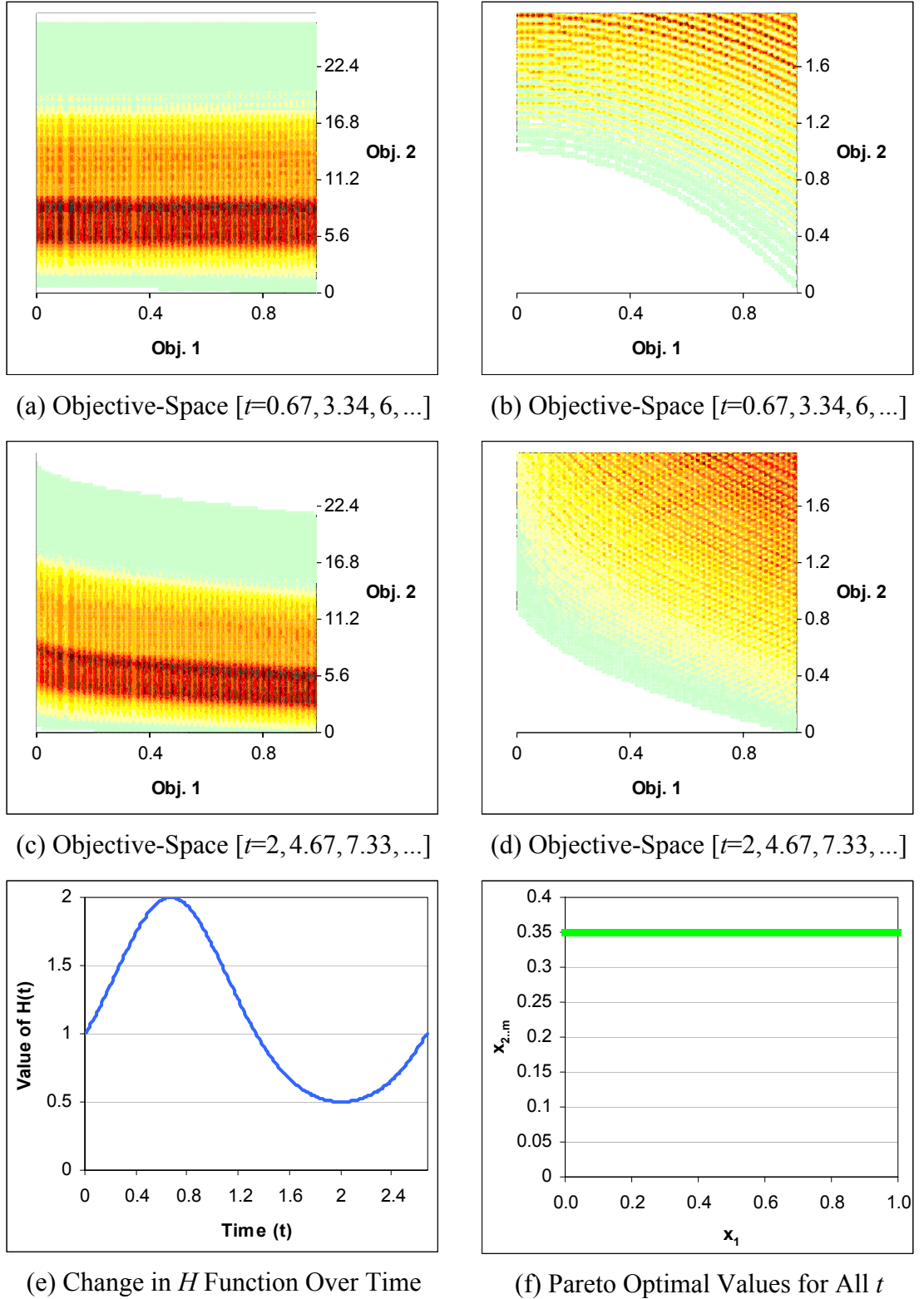
The Pareto optimal front is produced by varying  $x_1$  and setting  $x_{2..m}$  to the appropriate time-dependent value defined in (a), as illustrated at several key time points in (b). Performance on objective-one is time-independent and is optimal whenever  $x_1 = 0$ ; performance on objective-two is ideal only when  $x_1 = 1$  and  $x_{2..m}$  is set appropriately (again, see (a)). The periodic function described in (g) also dictates the size of the objective-space — it is maximal when  $|F(t)| = 1$  and minimal when  $F(t) = 0$ . The period of  $F$  is  $t = 4$ .

## A.12 EXAMINING *AP*-12


 (a) Sol-Space  $\rightarrow$  Objective One [all  $t$ ]

 (b) Sol-Space  $\rightarrow$  Objective Two  
[ $t=0, 2.67, 5.33, \dots$ ]

 (c) Sol-Space  $\rightarrow$  Objective Two  
[ $t=0.67, 3.34, 6, \dots$ ]

 (d) Sol-Space  $\rightarrow$  Objective Two  
[ $t=2, 4.67, 7.33, \dots$ ]

 (e) Objective-Space [ $t=0, 2.67, 5.33, \dots$ ]

 (f) Objective-Space [ $t=0, 2.67, 5.33, \dots$ ]

**Figure A15 — Problem Characteristics for *AP*-12**

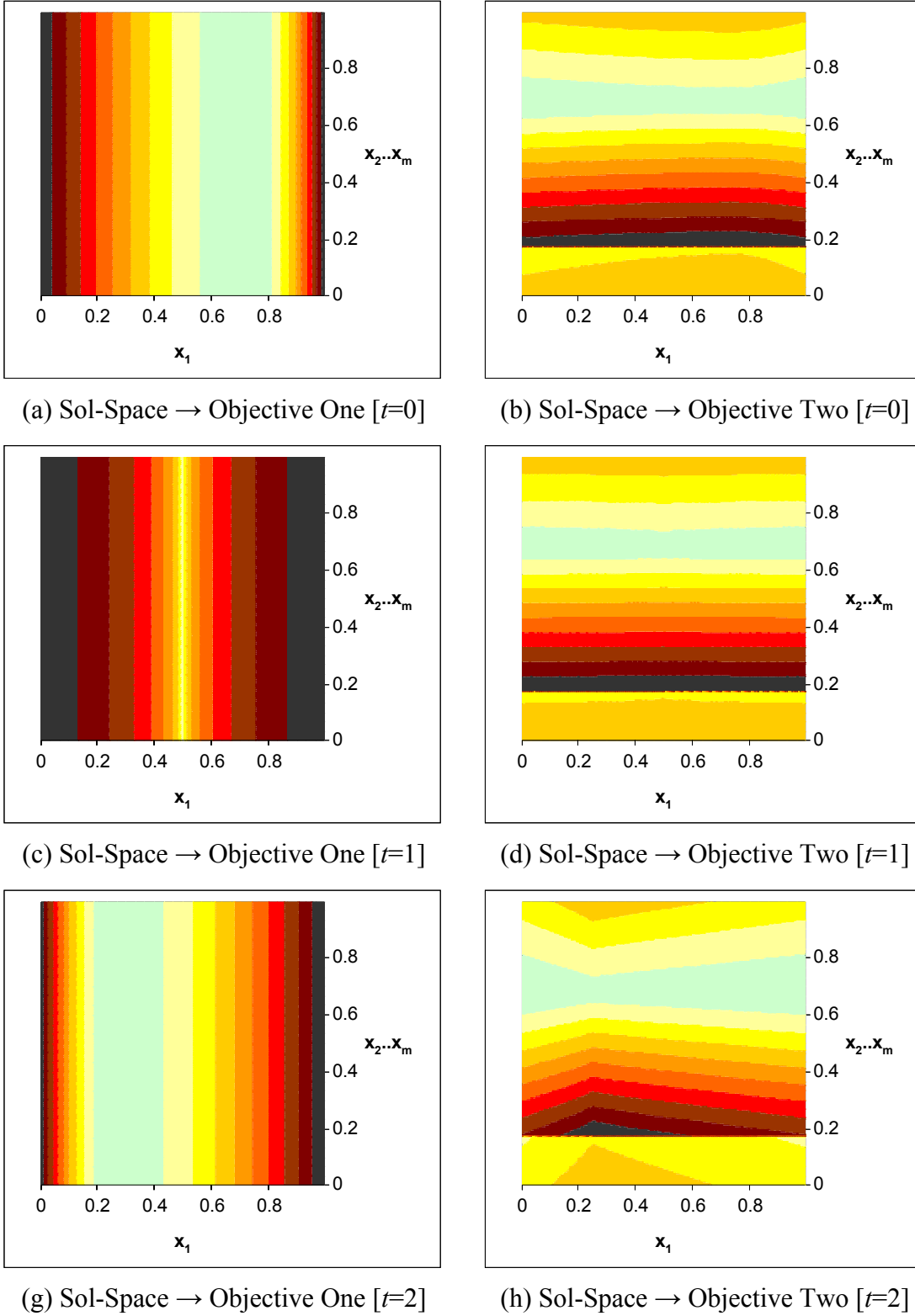
The solution-space mappings are constant for objective-one; mappings for objective-two vary subtly over time, though the optimal region is static. The Pareto optimal front varies shape with time, with static extreme points at (0,1) and (1,0). Note that (a-d) feature normalised values for  $x_{2..m}$ .  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .



**Figure A16 — Problem Characteristics for AP-12**

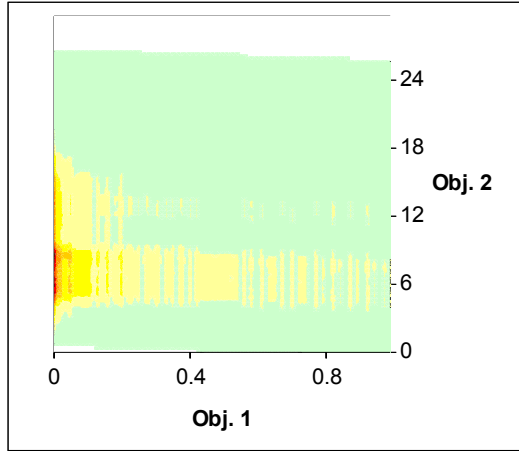
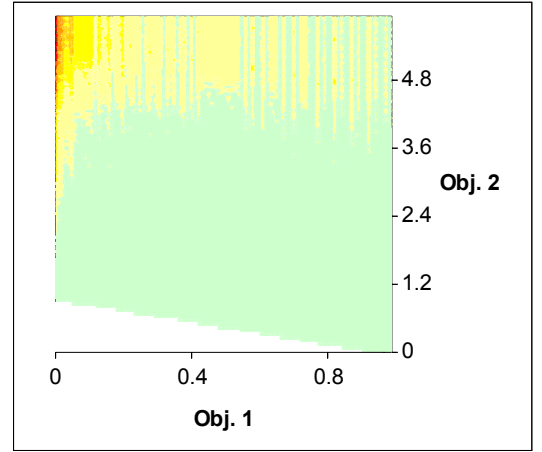
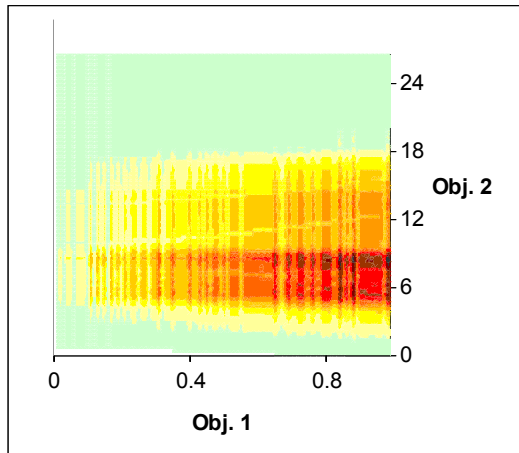
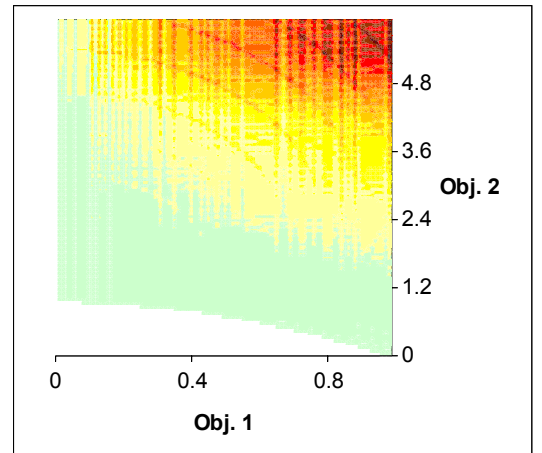
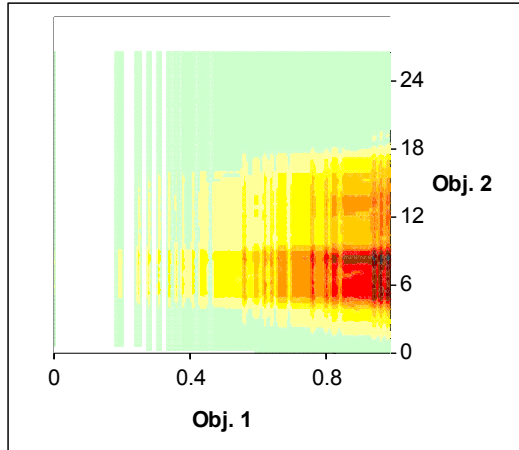
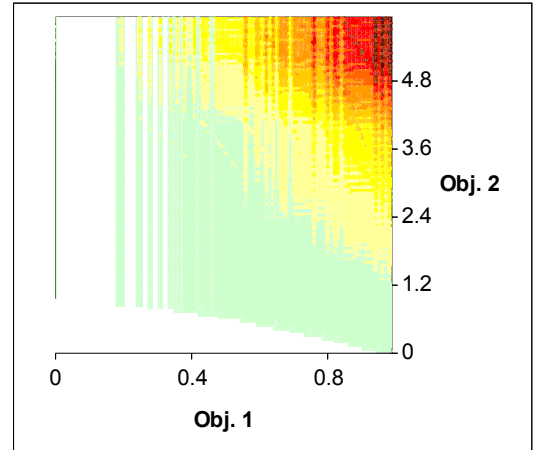
The Pareto optimal front is produced by varying  $x_1$  and setting  $x_{2,m}$  to 0.35. When the  $H$  function described in (e) is greater than one, the Pareto optimal front is concave; when  $H$  is less than one, the front is convex; when  $H$  equals one, the front is linear. The period of the  $H$  function is  $t \approx 2.67$ . Regardless of variation in shape, all optimal fronts share the same extremal points of (1,0) and (0,1).

### A.13 EXAMINING *AP-13*



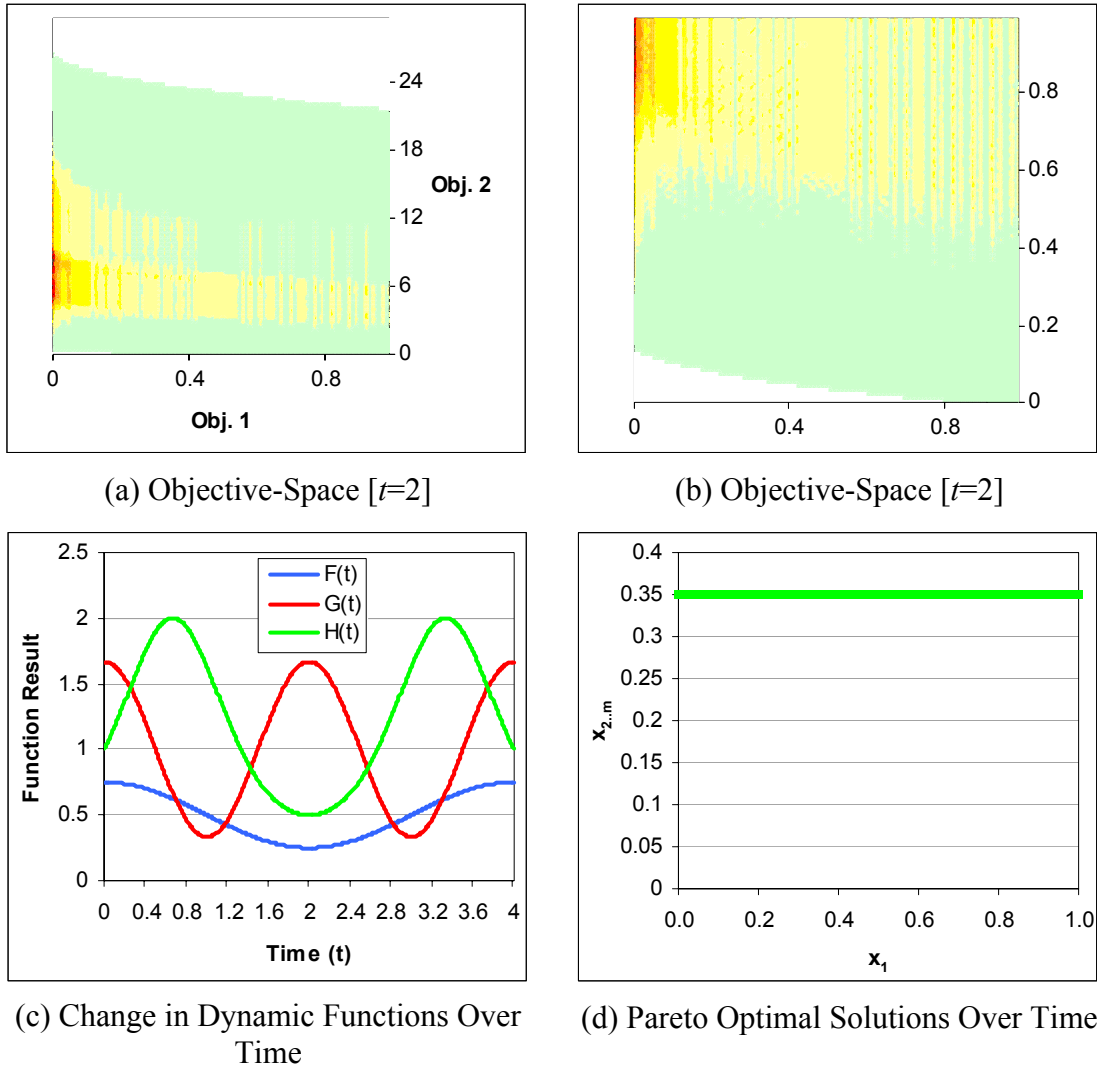
**Figure A17 — Problem Characteristics for *AP-13***

The search-gradient and optimal region varies markedly over time for objective-one. In contrast, the optimal region is fixed for objective-two and the solution-space varies only subtly across time. Note that all of the above graphs feature normalised values for  $x_{2..m}$ .  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .


 (i) Objective-Space [ $t=0$ ]

 (i) Objective-Space [ $t=0$ ]

 (j) Objective-Space [ $t=0.7$ ]

 (j) Objective-Space [ $t=0.7$ ]

 (k) Objective-Space [ $t=1$ ]

 (k) Objective-Space [ $t=1$ ]

**Figure A18 — Problem Characteristics for AP-13**

The objective-space bias varies with time, as does the shape of the Pareto optimal front. Regardless of variation in shape, all optimal fronts share the same extremal points of (1,0) and (0,1).

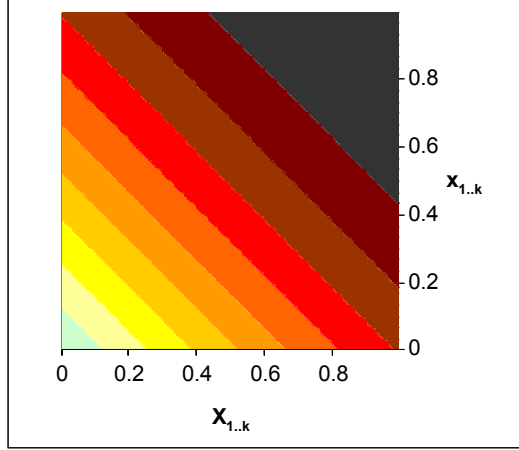
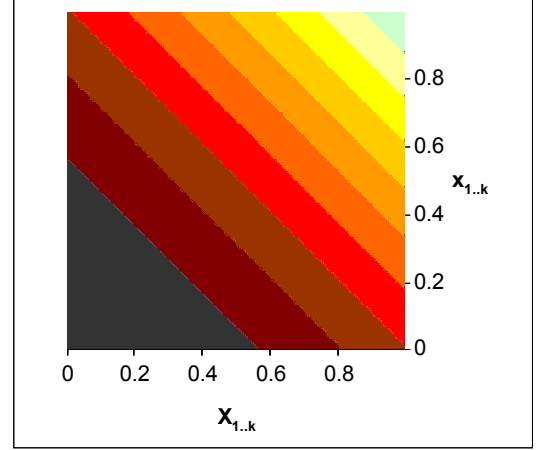
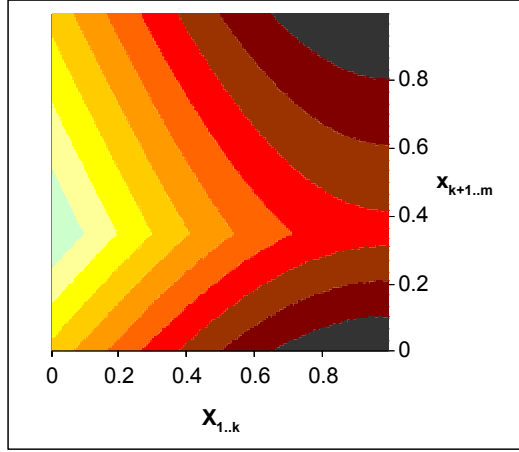
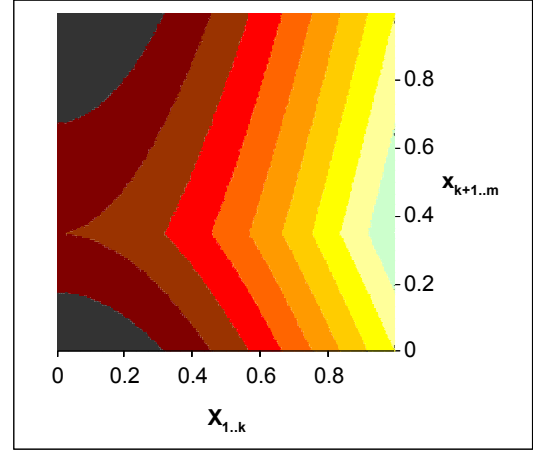
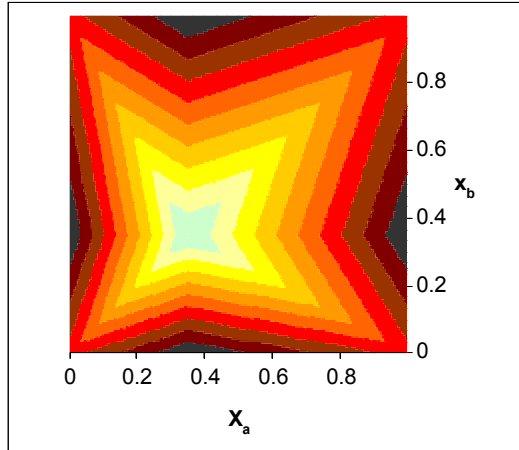
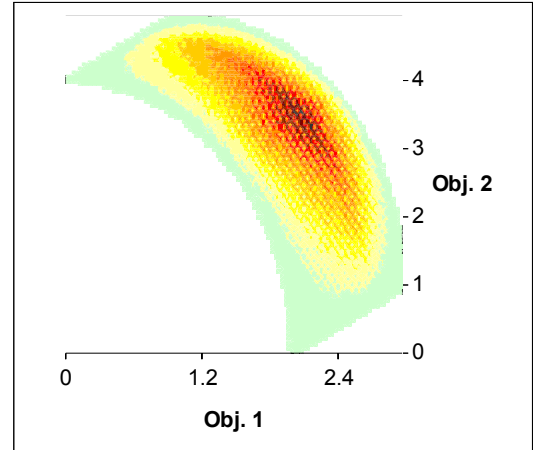


**Figure A19 — Problem Characteristics for AP-13**

Objective-one is optimal when  $x_1 = F(t)$  (see (d)); objective-two is optimal when  $x_1 = 0$  and  $x_{2,m} = 0.35$ . When the  $H$  function described in (c) is greater than one, the Pareto optimal front is concave; when  $H$  is less than one, the front is convex; when  $H$  equals one, the front is linear. Regardless of variation in shape, all optimal fronts share the same extremal points of (1,0) and (0,1). Objective-space bias is controlled by the  $G$  function—when  $G$  is less than one, the objective-space is biased towards optimal objective-one performance; when  $G$  is greater than one, the objective-space favours regions with poor objective-one performance. The period of  $F$  is  $t = 4$ ; the period of  $G$  is  $t = 2$ ; the period of  $H$  is  $t \approx 2.67$ .



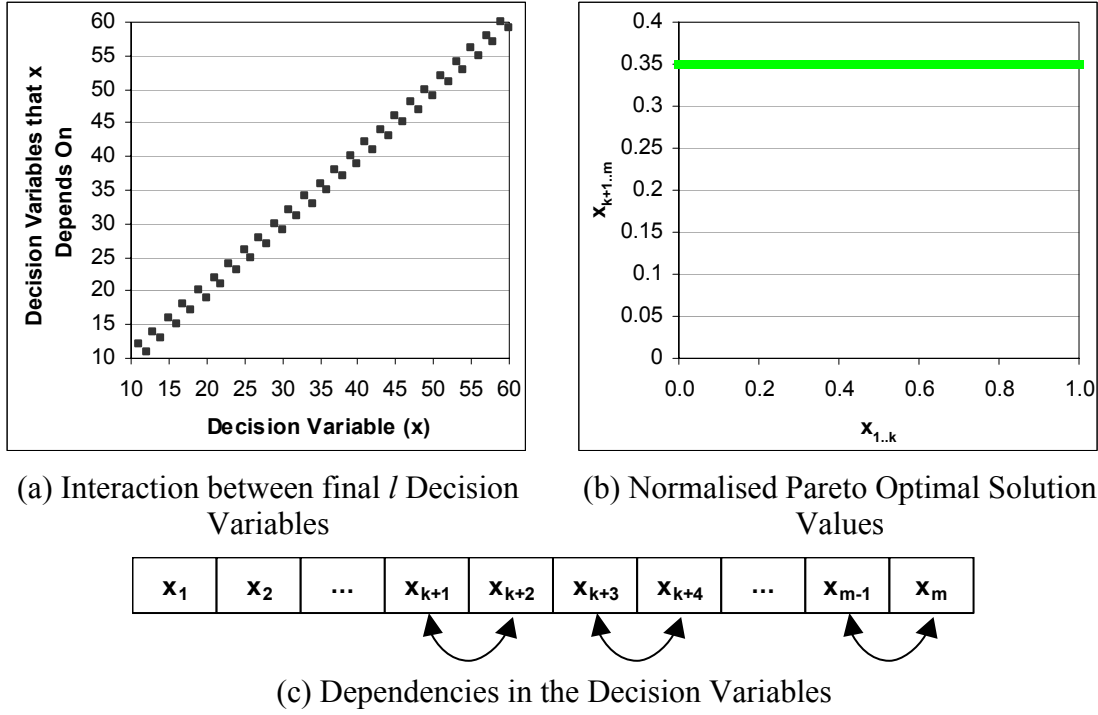
## A.14 EXAMINING *AP*-14


 (a) Solution-Space  $\rightarrow$  Objective-One

 (b) Solution-Space  $\rightarrow$  Objective-Two

 (c) Solution-Space  $\rightarrow$  Objective-One

 (d) Solution-Space  $\rightarrow$  Objective-Two

 (e) Solution-Space  $\rightarrow$  Either Objective  
 ( $a$  and  $b$  are dependent neighbours, where  $k < a, b \leq m$ )


(f) Objective-Space

**Figure A20 — Problem Characteristics for *AP*-14**

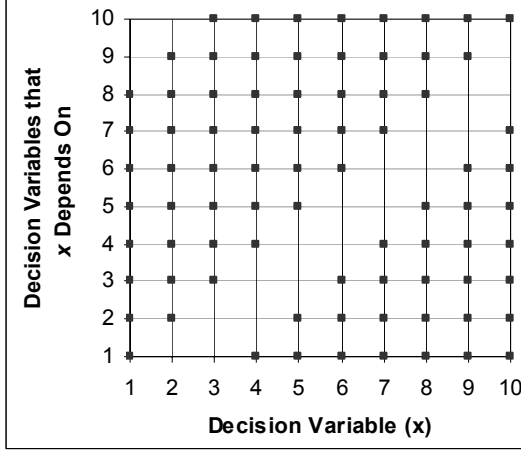
Considering normalised decision-variable values, objective-one is optimal when  $x_{1..k} = 0$  and  $x_{k+1..m} = 0.35$  objective-two is optimal when  $x_{1..k} = 1$  and  $x_{k+1..m} = 0.35$ . The Pareto optimal front is a concave curve with extreme points at (0,4) and (2,0). (e) illustrates the affect of pair-wise dependencies on the search-space gradients. (a), (b), (c), (d) and (e) all illustrate normalised variable values.  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .



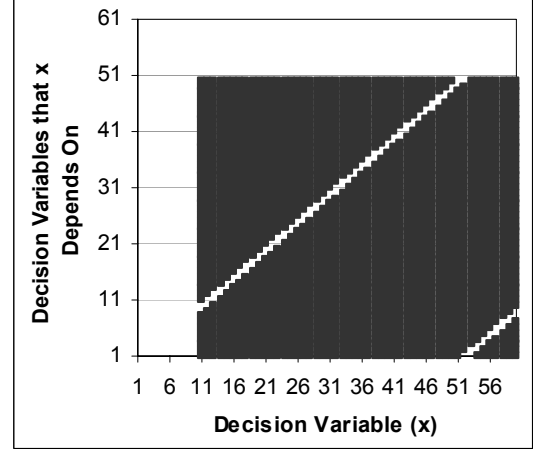
**Figure A21 — Problem Characteristics for AP-14**

(a) and (c) illustrate that dependencies only exist in local pair-wise couplings in the final  $l$  decision variables. (b) shows that the Pareto optimal front is formed by setting  $x_{k+1..m}$  to 0.35 (normalised) and varying  $x_{1..k}$ .

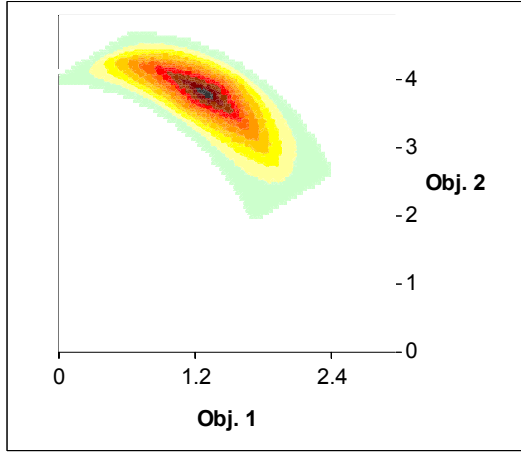
## A.15 EXAMINING *AP-15*



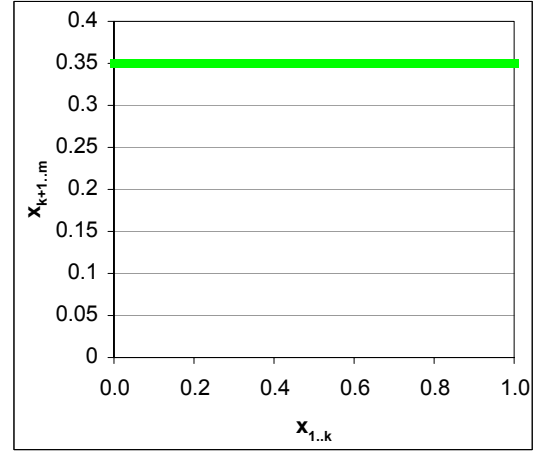
(a) Solution-Space Dependencies for  $x_{1..k}$



(b) Solution-Space Dependencies for  $x_{k+1..m}$



(c) Objective-Space

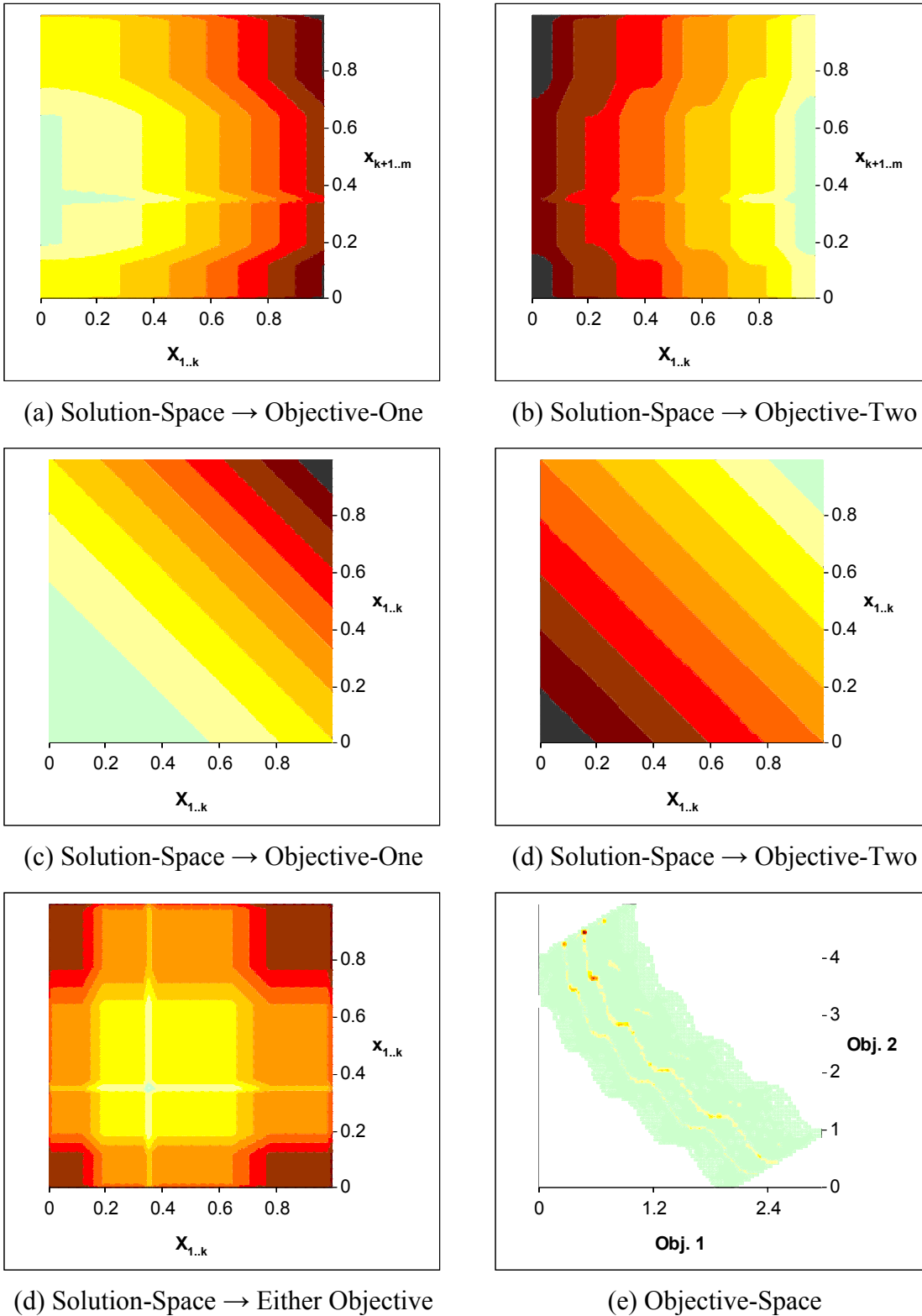


(d) Normalised Pareto Optimal Solution Values

**Figure A22 — Problem Characteristics for *AP-15***

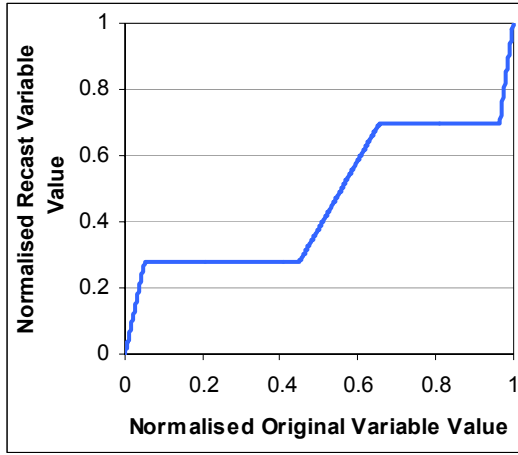
(a) illustrates that the first  $k$  decision variables are highly inter-dependent and feature mutual dependencies. (b) shows that all  $x_{k+1..m-k}$  are highly inter-dependent, include mutual dependencies and are affected by position parameters. (b) also demonstrates that all  $x_{m-k+1..m}$  have numerous dependencies on both position- and distance-related parameters, but lack any mutual dependencies. The resultant objective-space is a concave surface, with extent that is affected by the number and type of parameters chosen (due to the non-separability that exists between position and distance variables). The Pareto optimal front is formed by setting the normalised values of  $x_{k+1..m}$  to 0.35 and varying  $x_{1..k}$ .

## A.16 EXAMINING *AP-16*

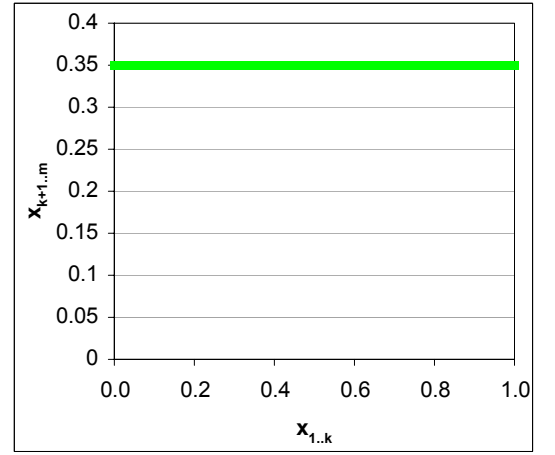


**Figure A23 — Problem Characteristics for *AP-16***

Considering normalised decision-variable values, objective-one is optimal when  $x_{1..k} = 0$  and  $x_{k+1..m} = 0.35$  objective-two is optimal when  $x_{1..k} = 1$  and  $x_{k+1..m} = 0.35$ . The Pareto optimal front is of mixed-shape with extreme points at (0,4) and (2,0). (d) illustrates the lack of gradient information in the final  $l$  decision variables.  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .



(a) Recasting Variable Values in the Final  $l$  Decision Variables

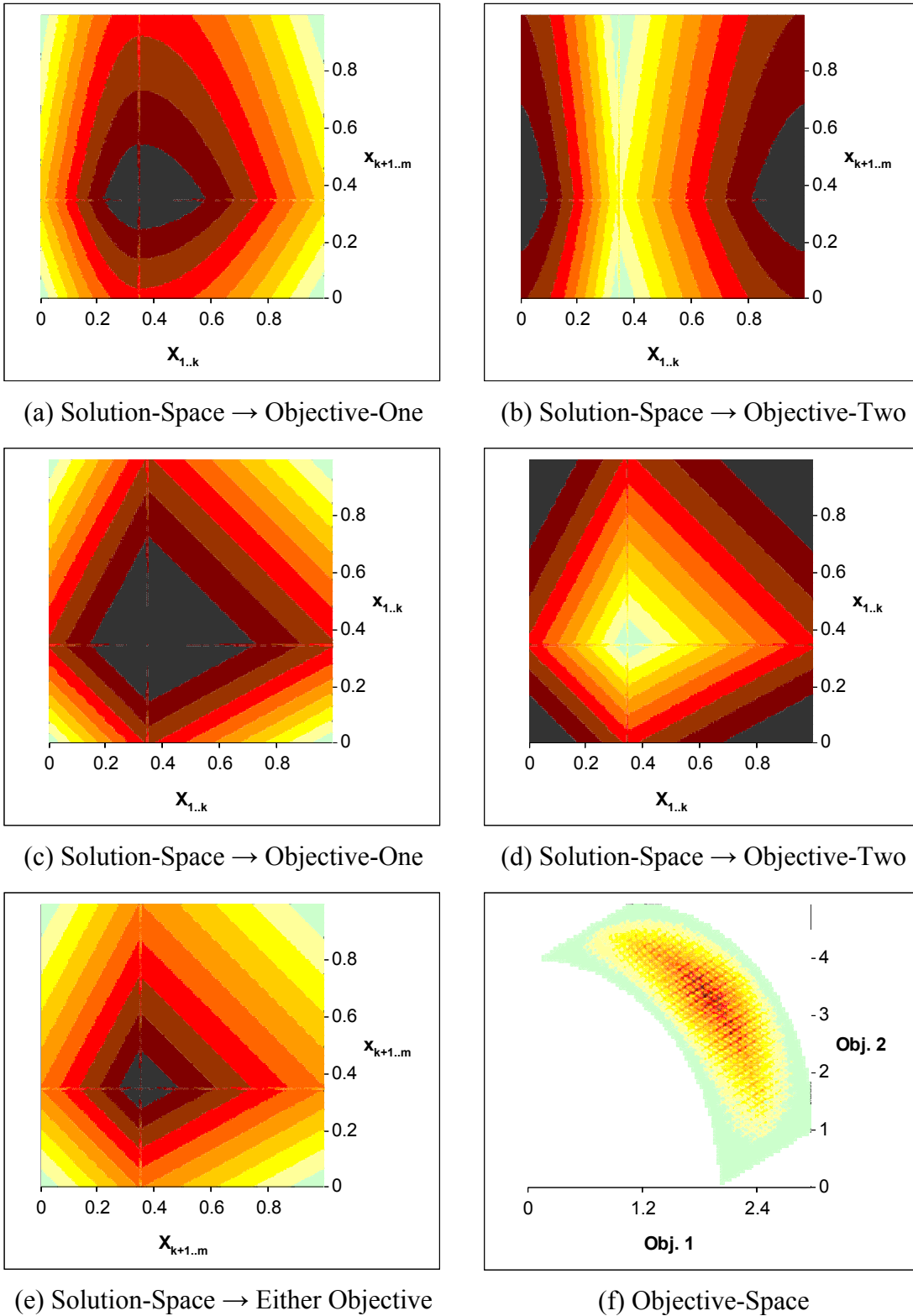


(b) Normalised Pareto Optimal Solution Values

**Figure A24 — Problem Characteristics for AP-16**

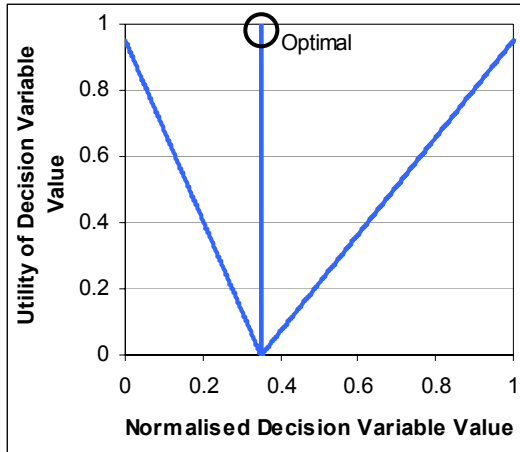
(a) demonstrates that all normalised  $x_{k+1..m}$  values between 0.05 and 0.45 are mapped to a single point, as are values between 0.657 and 0.965. (b) shows that the Pareto optimal front is formed by setting  $x_{k+1..m}$  to 0.35 (normalised) and varying  $x_{k+1..m}$ .

## A.17 EXAMINING *AP-17*

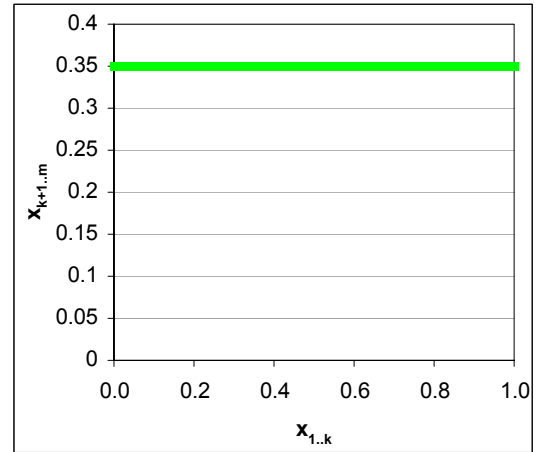


**Figure A25 — Problem Characteristics for *AP-17***

Considering normalised decision variable values, objective-one is optimal when  $x_{1..k} = 0$  and  $x_{k+1..m} = 0.35$ ; objective-two is optimal when  $x_{1..k} = 1$  and  $x_{k+1..m} = 0.35$ . The solution-space mappings clearly illustrate that the search gradients aggressively discourage the exploration of these optimal areas in all bar the final  $l$  decision variables for objective-two. The Pareto optimal front is a concave curve with extreme points at (0,4) and (2,0).  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .



(a) Utility of the final  $l$  Decision Variables for Objective-Two and All Decision Variables for Objective-One (Higher Utility is Better)

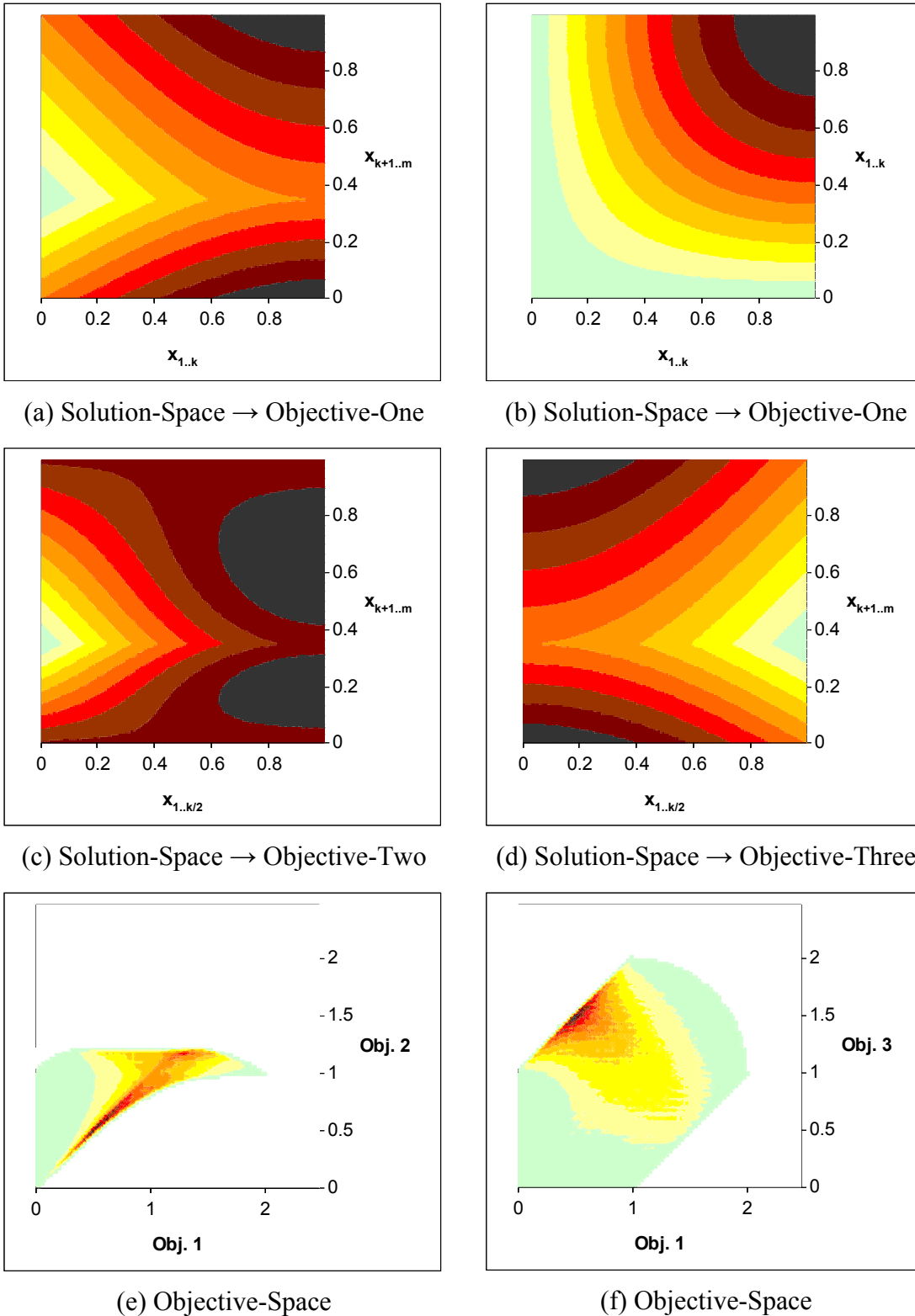


(b) Pareto Optimal Solution Values

**Figure A26 — Problem Characteristics for AP-17**

(a) demonstrates the lack of correlation between the search gradient and the location of the optimal value when deception is applied. (b) shows that the Pareto optimal front is formed by setting  $x_{k+1..m}$  to 0.35 (normalised) and varying  $x_{1..k}$ .

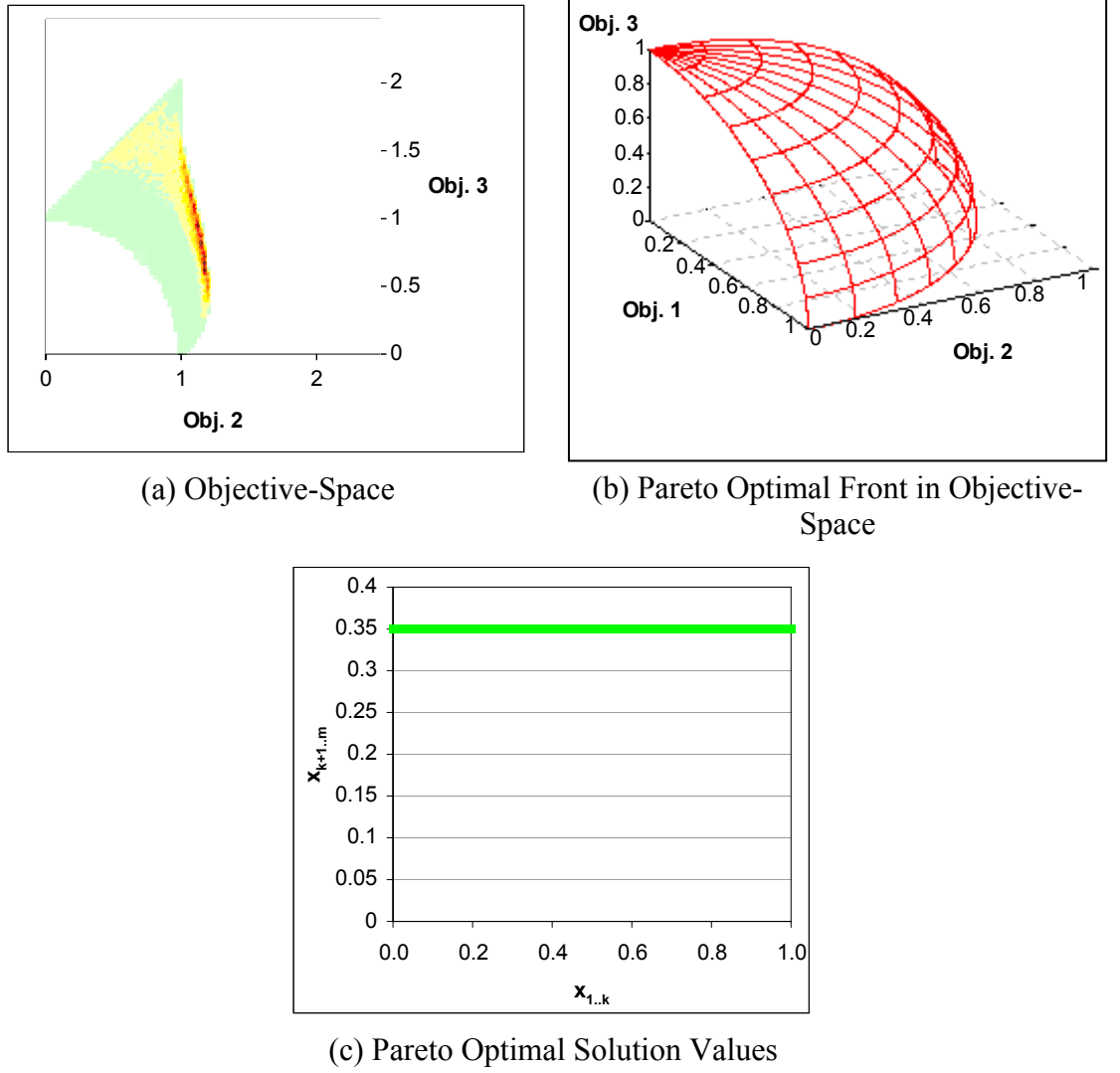
## A.18 EXAMINING *AP*-18



**Figure A27 — Problem Characteristics for *AP*-18**

Considering normalised decision variable values, objective-one is optimal when  $x_{1..k} = 0$  and  $x_{k+1..m} = 0.35$ ; objective-two is optimal when  $x_{1..k/2} = 0$  and  $x_{k+1..m} = 0.35$ ; objective-three is optimal when  $x_{1..k} = 1$  and  $x_{k+1..m} = 0.35$ . Note that objective-one performance depends on  $x_{1..m}$ , while objective-two and objective-three depend on only  $x_{1..k/2}$  and  $x_{k+1..m}$ . (e) and (f) illustrate slices of the objective-space and indicate the presence of bias.  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .

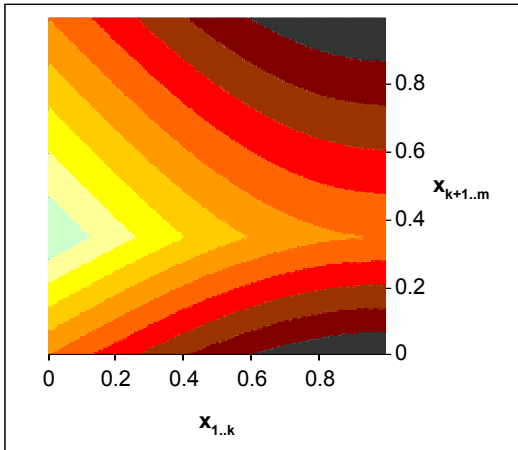
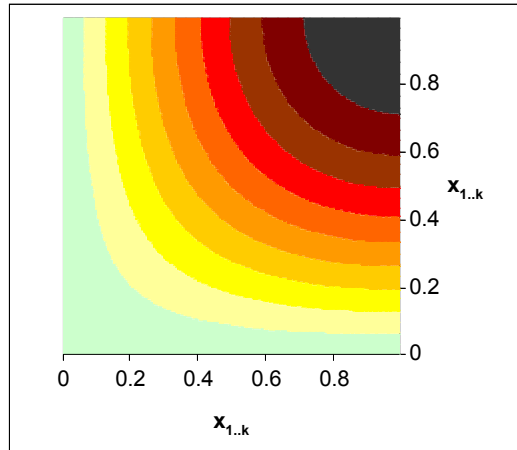
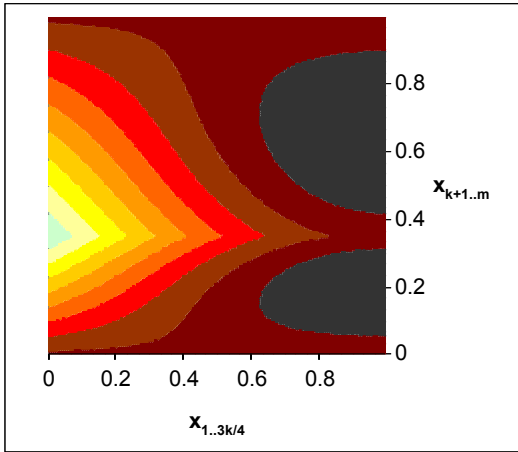
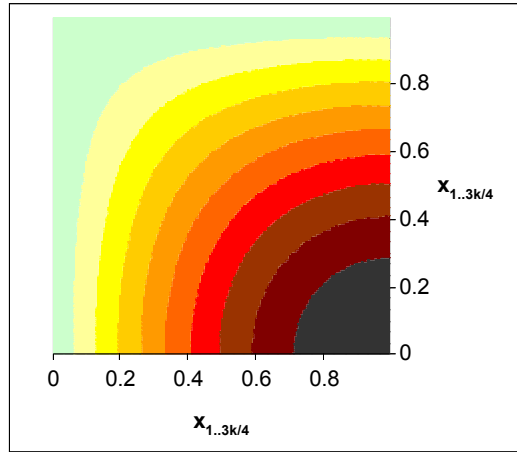
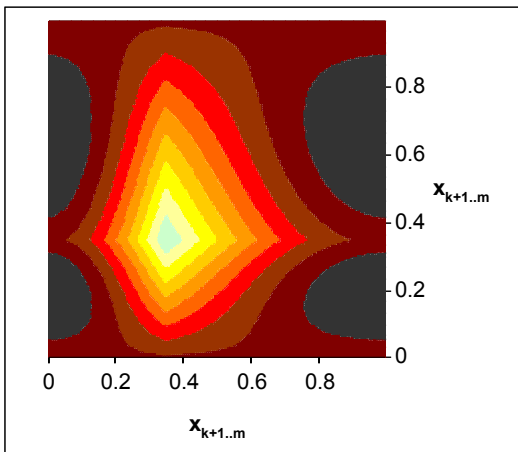
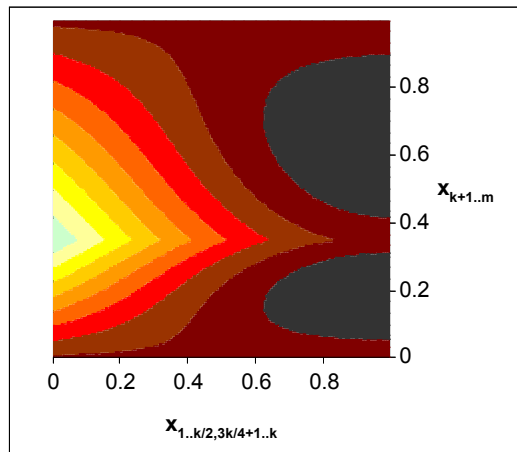




**Figure A28 — Problem Characteristics for AP-18**

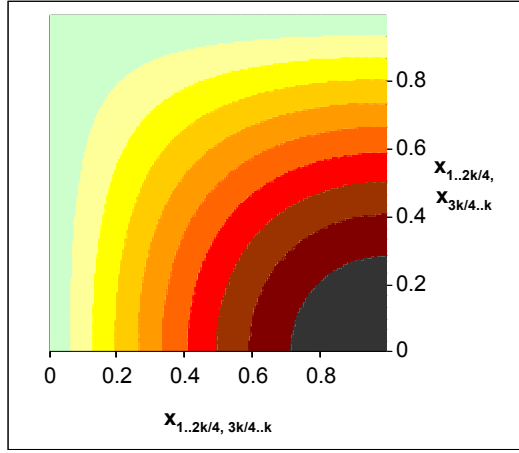
(a) further describes the presence of objective-space bias. (b) illustrates the *shape* of the concave three-dimensional Pareto optimal front — with extremes at (1,0,0), (0,1,0) and (0,0,1). Note that the diagram is not intended to impart density information. (c) shows that the Pareto optimal front is formed by setting  $x_{k+1..m}$  to 0.35 (normalised) and varying  $x_{1..k}$ .

## A.19 EXAMINING AP-19

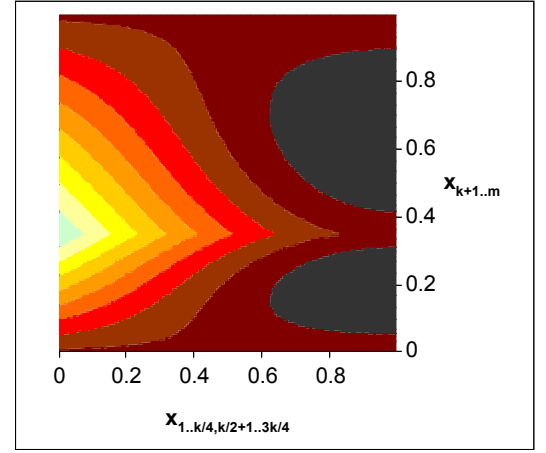

 (a) Solution-Space  $\rightarrow$  Objective-One

 (b) Solution-Space  $\rightarrow$  Objective-One

 (c) Solution-Space  $\rightarrow$  Objective-Two

 (d) Solution-Space  $\rightarrow$  Objective-Two

 (e) Solution-Space  $\rightarrow$  Objective-Two

 (f) Solution-Space  $\rightarrow$  Objective-Three

**Figure A29 — Problem Characteristics for AP-19**

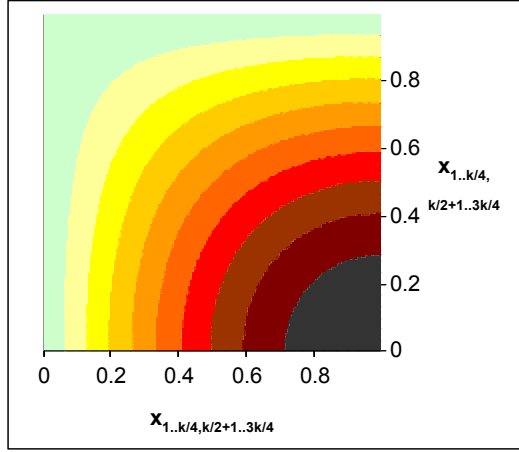
Considering normalised decision variable values, objective-one is optimal when  $x_{1..k} = 0$  and  $x_{k+1..m} = 0.35$ ; objective-two is optimal when  $x_{1..3k/4+1} = 0$  and  $x_{k+1..m} = 0.35$ ; objective-three is optimal when  $x_{1..k/2, 3k/4+1..k} = 0$  and  $x_{k+1..m} = 0.35$ . Note that objective-one performance depends on  $x_{1..m}$ , while objective-two depends on all variables bar  $x_{3k/4+1..k}$  and objective-three is affected by  $x_{k/2+1..m}$ .  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .



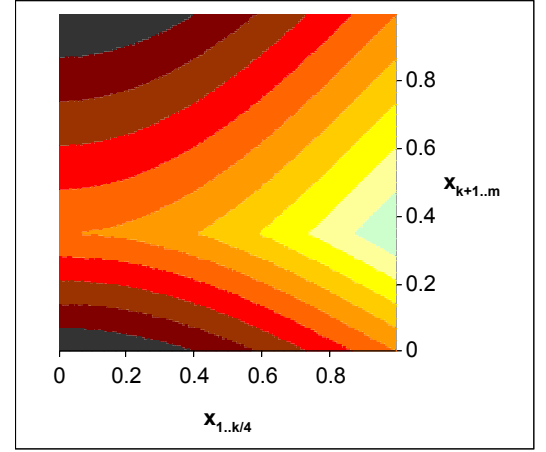
(a) Solution-Space → Objective-Three



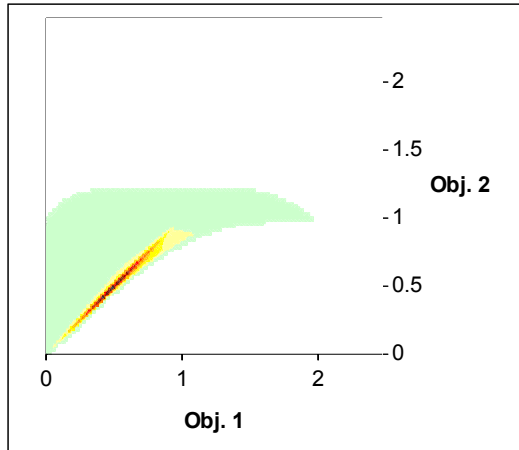
(b) Solution-Space → Objective-Four



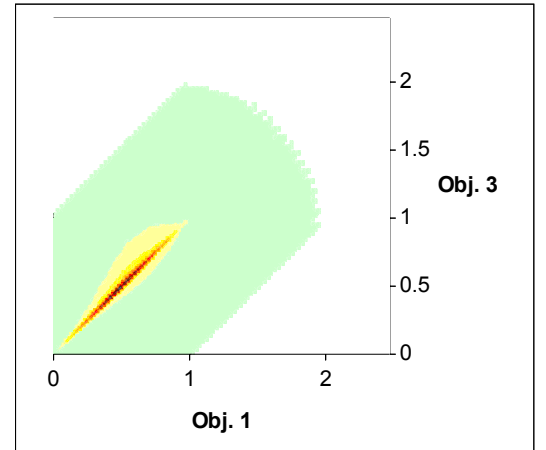
(c) Solution-Space → Objective-Four



(d) Solution-Space → Objective-Five



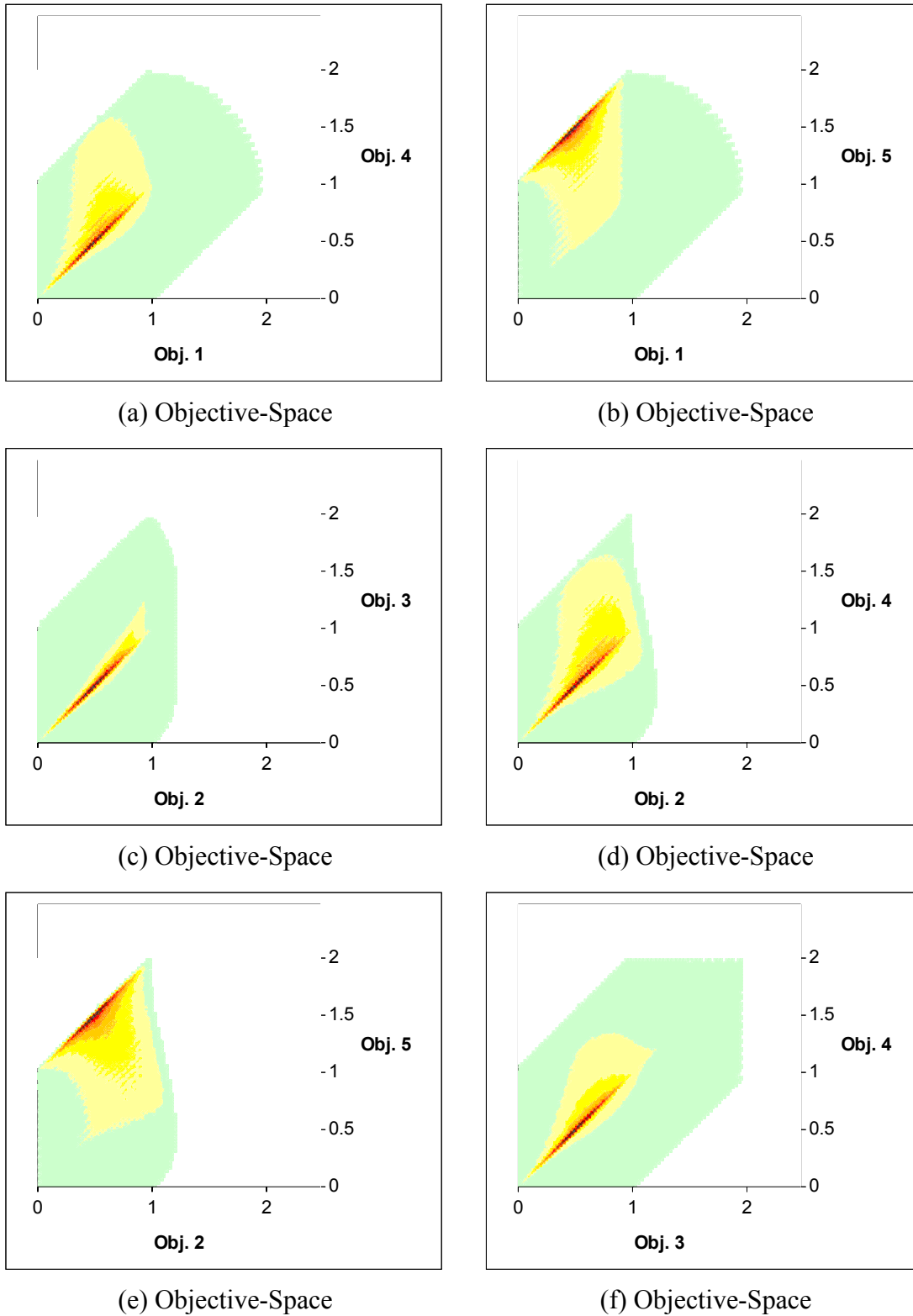
(e) Objective-Space



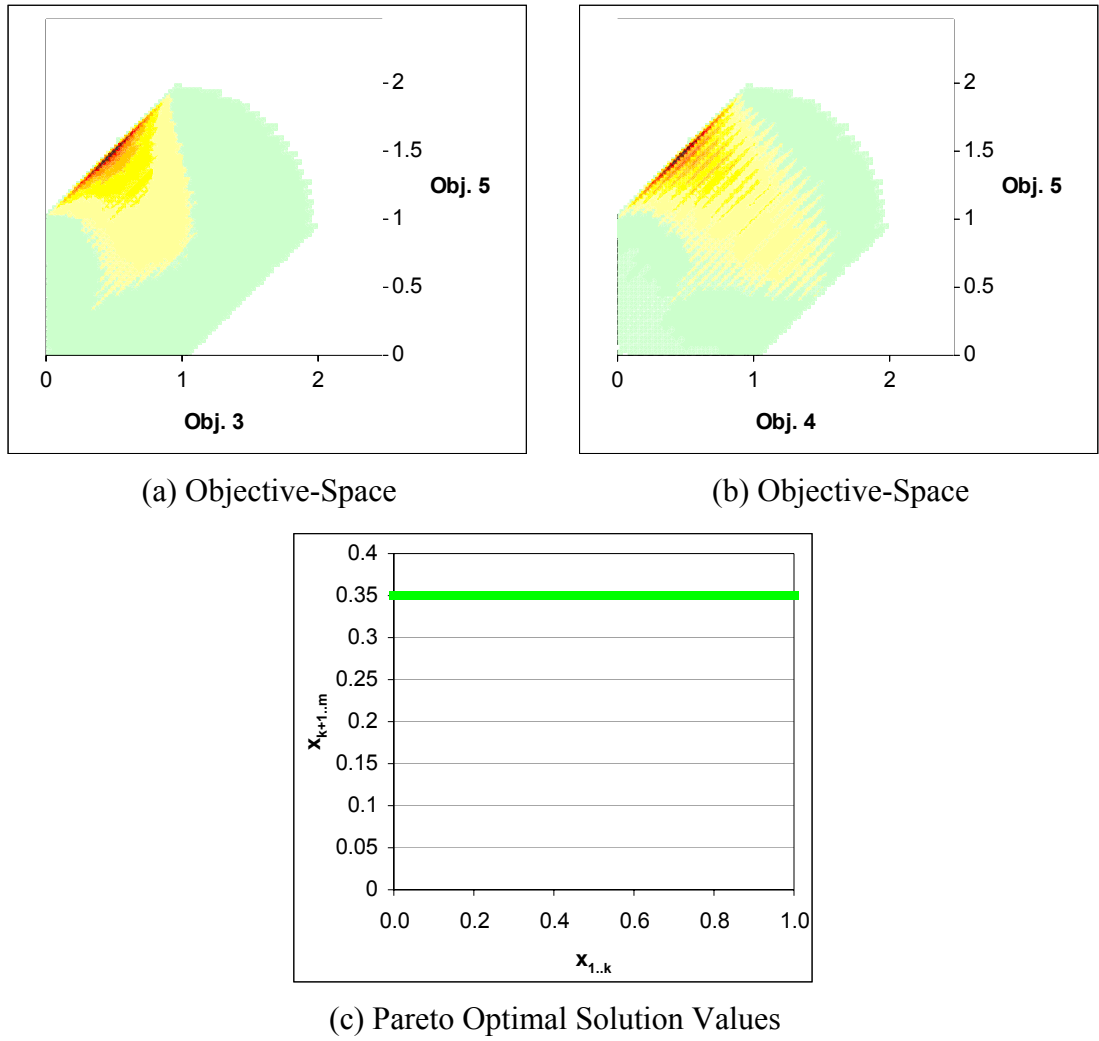
(f) Objective-Space

**Figure A30 — Problem Characteristics for AP-19**

Considering normalised decision variable values, objective-four is optimal when  $x_{1..k/4, 3k/4+1..k} = 0$  and  $x_{k+1..m} = 0.35$ ; and objective-five is optimal when  $x_{1..k/4} = 0$  and  $x_{k+1..m} = 0.35$ . Note that objective-four performance depends on all variables bar  $x_{k/2+1..3k/4}$  and objective-five is affected by only  $x_{1..k/4}$  and  $x_{k+1..m}$ . (e) and (f) illustrate slices of the objective-space — each with obvious bias.  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .



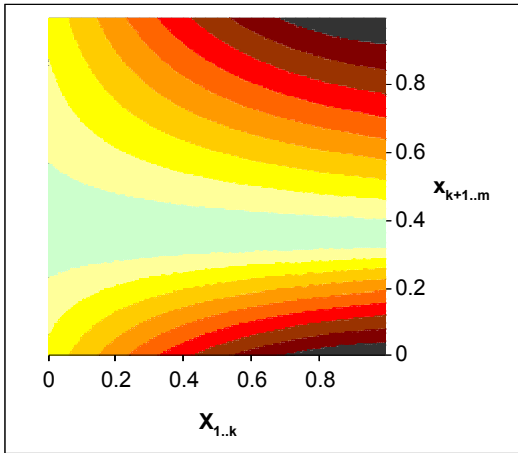
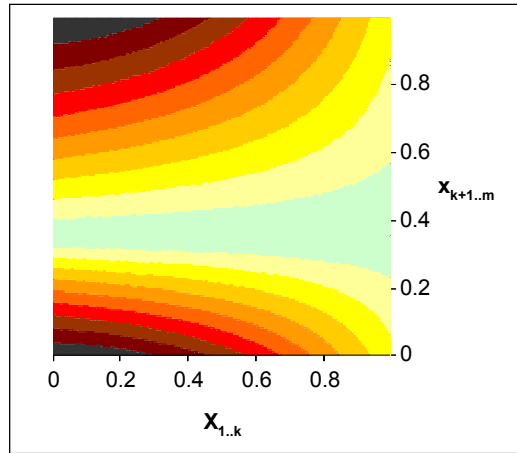
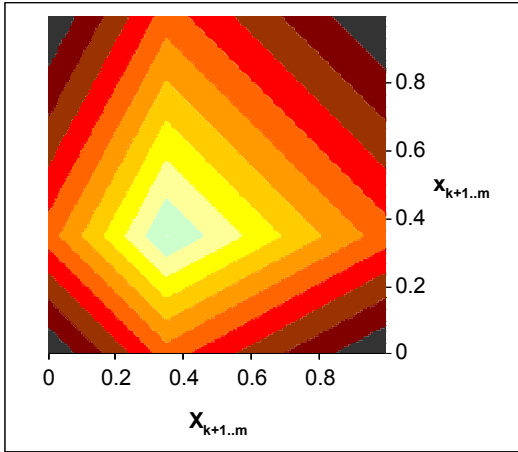
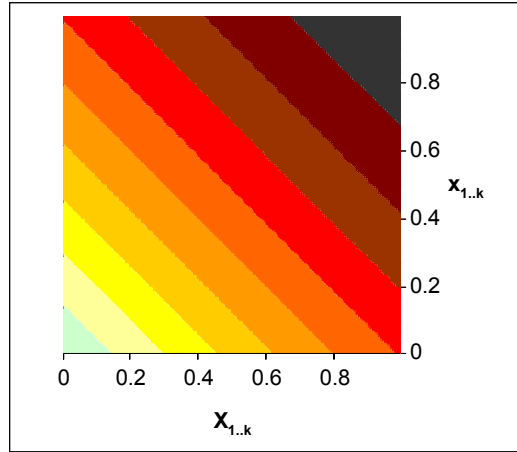
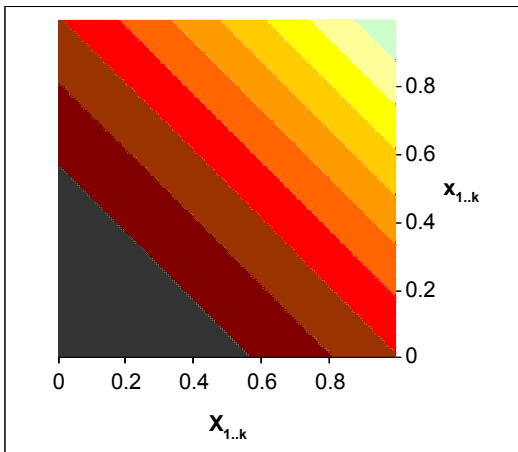
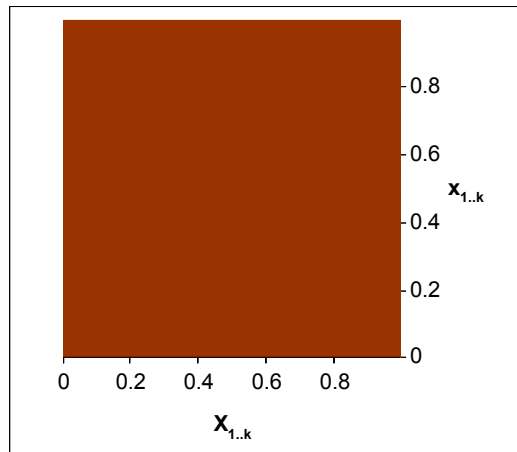
**Figure A31 — Problem Characteristics for AP-19**  
Slices of the objective-space — each with obvious bias.



**Figure A32 — Problem Characteristics for AP-19**

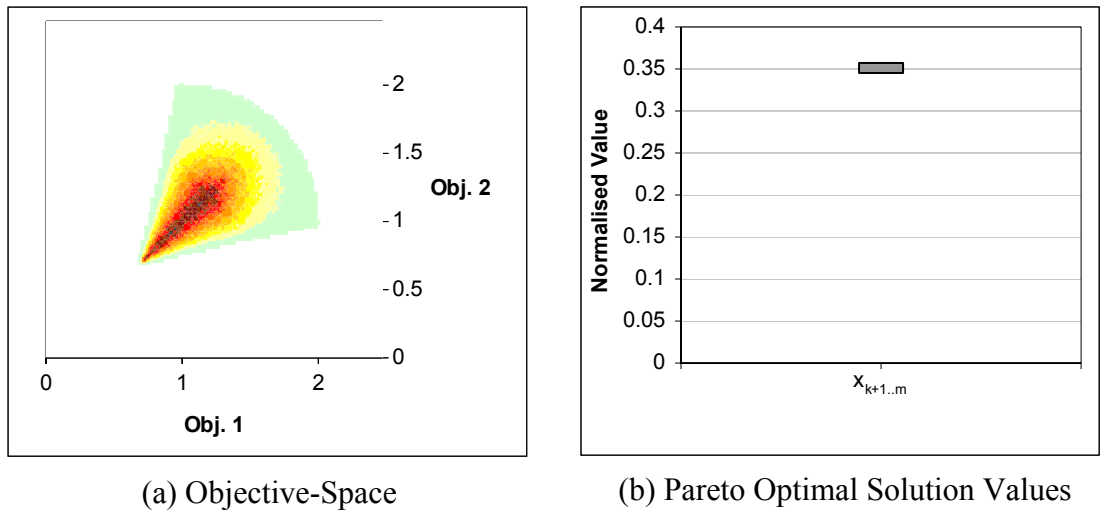
(a) and (b) are slices of the objective-space — each with obvious bias. (c) illustrates that the concave optimal front is formed by setting  $x_{k+1..m}$  to 0.35 (normalised) and varying  $x_{1..k}$ .

## A.20 EXAMINING *AP-20*


 (a) Solution-Space  $\rightarrow$  Objective-One

 (b) Solution-Space  $\rightarrow$  Objective-Two

 (c) Solution-Space  $\rightarrow$  Either Objective

 (d) Solution-Space  $\rightarrow$  Objective-One  
(when  $x_{k+1..m}$  is sub-optimal)

 (e) Solution-Space  $\rightarrow$  Objective-Two

 (f) Solution-Space  $\rightarrow$  Objective-One  
(when  $x_{k+1..m}$  is optimal)

**Figure A33 — Problem Characteristics for *AP-20***

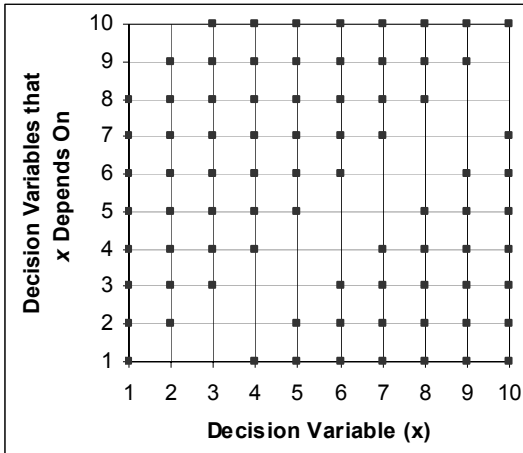
As  $x_{k+1..m}$  progressively approaches better performing space, the importance of  $x_{1..k}$  diminishes until eventually — when  $x_{k+1..m}$  is optimal at 0.35 — its value is meaningless (see the search-gradient in (f)).  
 $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .



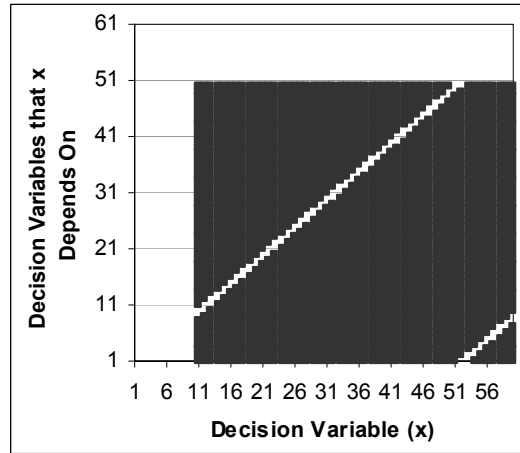
**Figure A34 — Problem Characteristics for AP-20**

The final Pareto optimal front is a single *point* in objective-space at (0.707, 0.707). The front is formed by setting  $x_{k+1..m}$  to 0.35 (optimal performance is completely independent of the value of  $x_{1..k}$ ).

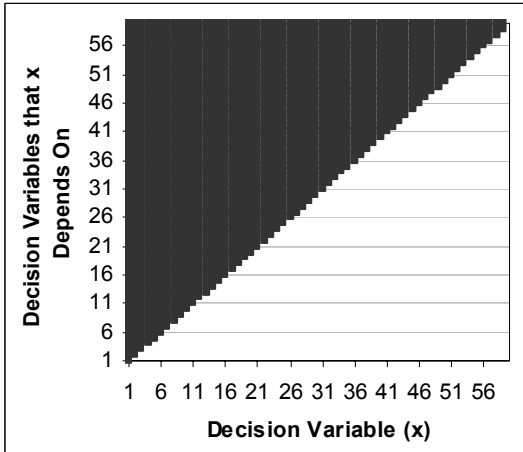
## A.21 EXAMINING *AP-21*



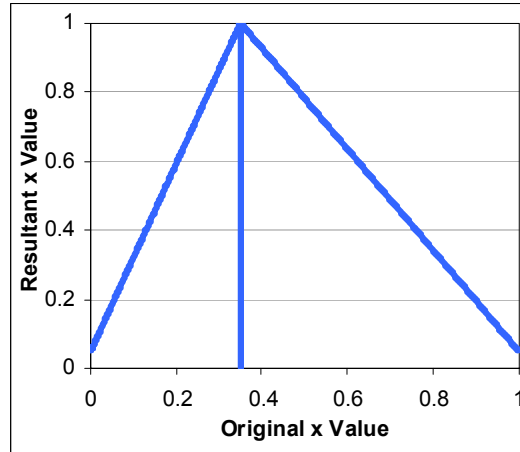
(a) Solution-Space Dependencies for  $x_{1..k}$  Caused by Non-Separable Function



(b) Solution-Space Dependencies for  $x_{k+1..m}$  Caused by Non-Separable Function



(c) Solution-Space Dependencies for  $x_{1..k}$  Caused by Biasing Function

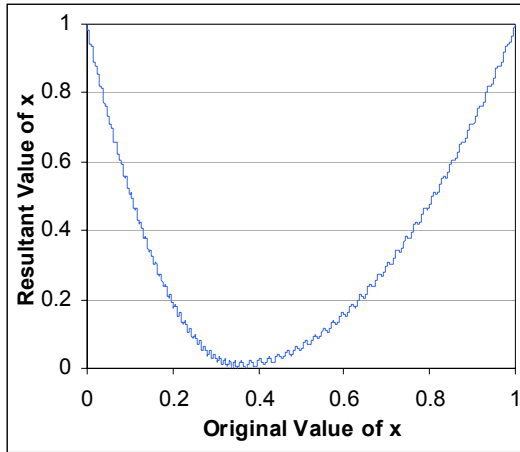


(d) Effect of Deception on  $x_{1..k}$  Values

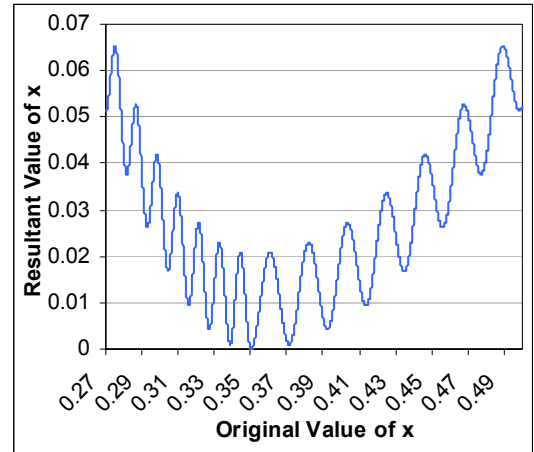
**Figure A35 — Problem Characteristics for *AP-21***

(a) and (c) illustrate that the first  $k$  decision variables are highly inter-dependent and feature mutual dependencies — caused both by the explicit non-separable function and the parameter biasing function. (b) shows that all  $x_{k+1..m-k}$  are highly inter-dependent, include mutual dependencies and are affected by position parameters. (b) also demonstrates that all  $x_{m-k+1..m}$  have numerous dependencies on both position- and distance-related parameters, but lack any mutual dependencies. (d) demonstrates the lack of correlation between the search gradient and the location of the optimal value when deception is applied to  $x_{1..k}$ .

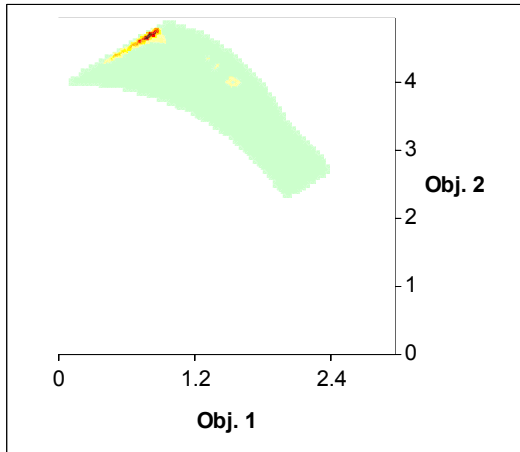




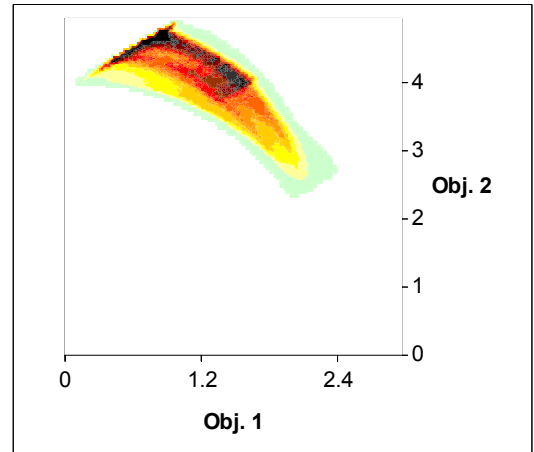
(a) Effect of Multi-Modality on  $x_{k+1..m}$  Values



(b) Effect of Multi-Modality on  $x_{k+1..m}$  Values



(c) Objective-Space



(d) Objective-Space

**Figure A36 — Problem Characteristics for AP-21**

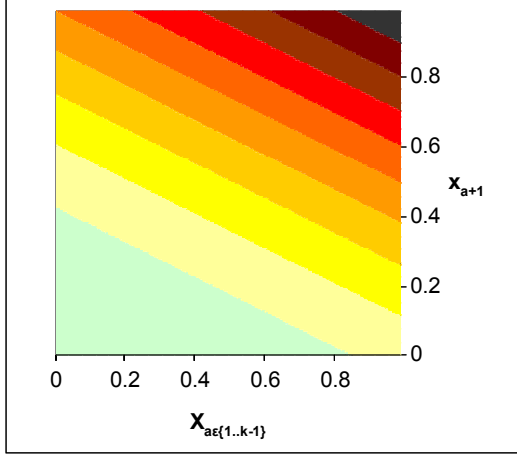
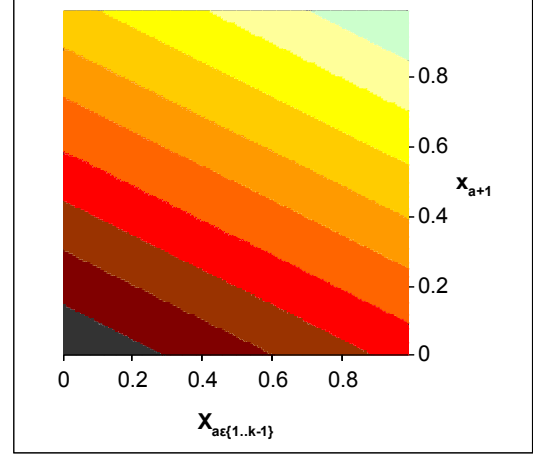
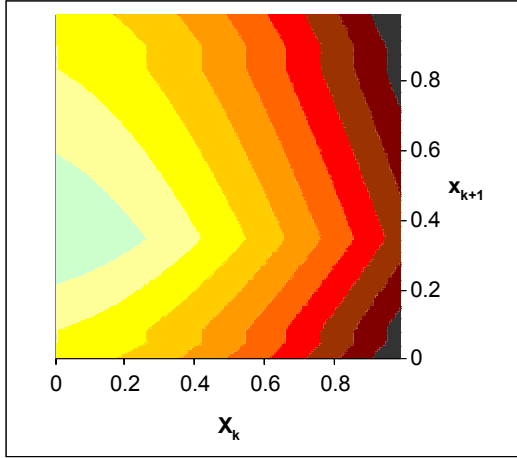
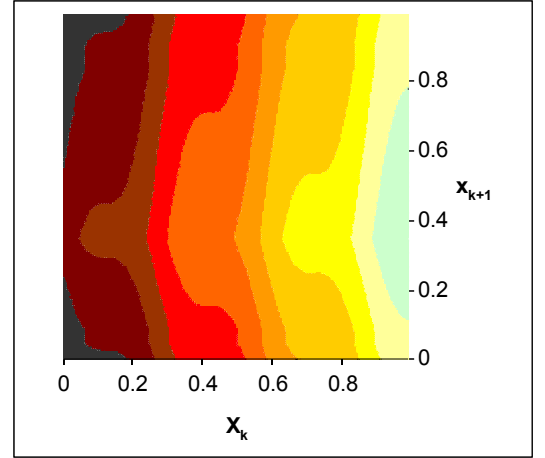
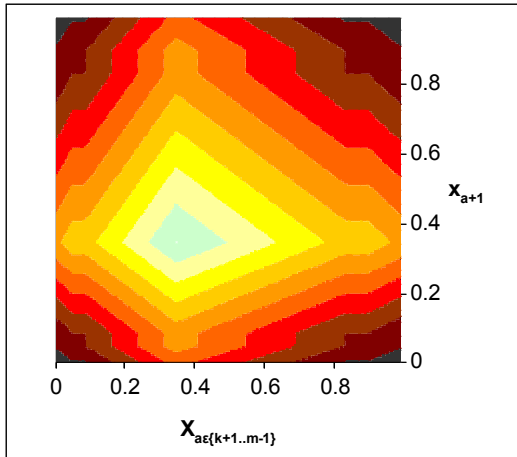
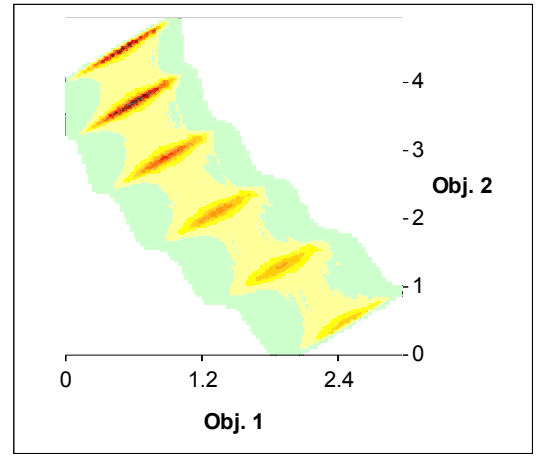
(a) and (b) highlight the fine-grained multi-modality that affects  $x_{k+1..m}$ . (c) illustrates the bias of the objective-space, with (d) normalising frequencies to the range of 0% - 10% to better elucidate secondary influences (any frequency  $\geq 10\%$  is black). The Pareto optimal front is a concave surface, with extent that is affected by the number and type of parameters chosen (due to the non-separability that exists between position and distance variables).

## A.22 AN AUXILIARY TEST PROBLEM — *P1*

$$\begin{aligned}
 x'_{i=1..k} &= x_i \\
 x'_{i=k+1..m} &= \text{b\_flat}(x_i, 0.8, 0.75, 0.85) \\
 x''_1 &= \sum_{i=1}^k \frac{2i \times x'_i}{k^2 + k}, \quad x''_2 = \sum_{i=k+1}^m \frac{2i \times x'_i}{m^2 + m - k^2 - k} \\
 f_1(X'') &= 2 + x''_2 - 2 \cos\left(\frac{\pi x''_1}{2}\right) \\
 f_2(X'') &= x''_2 + 4 \left(1 - x''_1 - \cos\left(\frac{10\pi x''_1 + 0.5\pi}{10\pi}\right)\right) \\
 \text{where } X &= \{x_1, x_2, \dots, x_k, x_{k+1}, \dots, x_m\}, \quad X' = \{x'_{1..k+l/2}\}, \quad X'' = \{x''_1, x''_2\}, \\
 m &= 60, \quad k = 10, \quad l = 50, \quad x_{i=1..m} \in [0, 1]
 \end{aligned} \tag{87}$$

This auxiliary multi-faceted problem (constructed from the functions provided in [158]) features a mixed-shape Pareto optimal front and parameter biasing that favours the later decision variables (see Figure A37 for spectral graph analyses). The problem is further complicated by the presence of a small zero-utility gradient that affects the final  $l$  solution parameters.

The problem is included as an alternative to the particularly complex multi-faceted *AP-21* problem and is appropriate for use when the investigation of high-levels of non-separability is not required.


 (a) Solution-Space  $\rightarrow$  Objective-One

 (b) Solution-Space  $\rightarrow$  Objective-Two

 (c) Solution-Space  $\rightarrow$  Objective-One

 (d) Solution-Space  $\rightarrow$  Objective-Two

 (e) Solution-Space  $\rightarrow$  Either Objective


(f) Objective-Space

**Figure A37 — Problem Characteristics for Auxiliary Problem P1**

Objective-one is optimal when  $x_{1..k} = 0$  and  $x_{k+1..m} = 0.35$ ; objective-two is optimal when  $x_{1..k} = 1$  and  $x_{k+1..m} = 0.35$ . Note that the solution-spaces in these examples consider neighbouring decision variables only — increased spacings result in similar gradients, though they are increasingly affected by parameter weights. The Pareto optimal front is of mixed-shape with extreme points at (0,4) and (2,0).  $a \rightarrow b$  indicates the mapping of  $a$  to  $b$ .

## A.23 AUXILIARY FUNCTIONS FOR TEST PROBLEMS

### A.23.1 S\_LINEAR

$$s\_linear(x, opt) = \frac{|x - opt|}{|opt - x| + opt} \quad (A1)$$

Taken from Huband *et al.* [158], the  $s\_linear$  function recasts the variable  $x$  such that it maps to zero when at  $opt$  according to Figure A38.

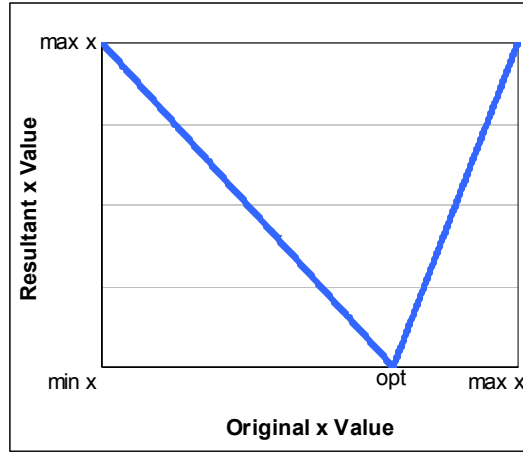


Figure A38 — Illustrating the  $s\_linear$  Function

### A.23.2 R\_FLAT

$$r\_flat(x, a, b, c) = a + \min(0, \lfloor x - b \rfloor) \times \frac{a \times (b - x)}{b} - \min(0, \lfloor c - x \rfloor) \times \frac{(1 - a) \times (y - c)}{1 - c} \quad (A2)$$

Taken from Huband *et al.* [158], the  $r\_flat$  function takes a variable  $x$  and maps it such that it equals  $a$  when initially between  $b$  and  $c$  (see Figure A39).

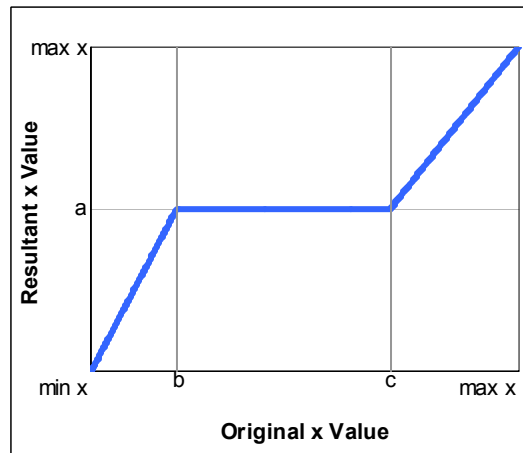


Figure A39 — Illustrating the  $r\_flat$  Function

### A.23.3 INTERDEPEND

$$\text{interdepend}(X) = \frac{\sum_{i=1}^{|X|} X_i + \sum_{j=1}^{|X|} |X_i - X_j|}{0.5 \times |X| \times (1 + |X|)} \quad (\text{A3})$$

Specialised from Huband *et al.*'s [158] *r\_nonsep*, the *interdepend* function takes a set of solutions  $X$  and forms pair-wise dependencies as per Figure A40.

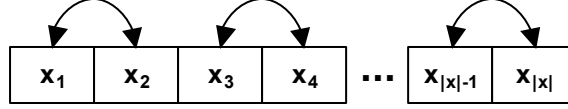


Figure A40 — Illustrating Dependencies caused by the *Interdepend* function  
Arrows represent dependencies.

### A.23.4 S\_DECEPT

$$\begin{aligned} f(x, opt, b) &= 1 + (|x - opt| - b) \\ g(x, opt, b, c) &= \frac{|x - opt + b| \left( 1 - c + \frac{opt - b}{b} \right)}{opt - b} \\ h(x, opt, b, c) &= \frac{|opt + b - x| \left( 1 - c + \frac{1 - opt - b}{b} \right)}{1 - opt - b} \\ s\_decept(x, opt, b, c) &= f(x, opt, b) \times \left( g(x, opt, b, c) + h(x, opt, b, c) + \frac{1}{b} \right) \end{aligned} \quad (\text{A4})$$

Taken from Huband *et al.* [158], the *s\_decept* function takes a variable  $x$  and recasts it according to Figure A41.

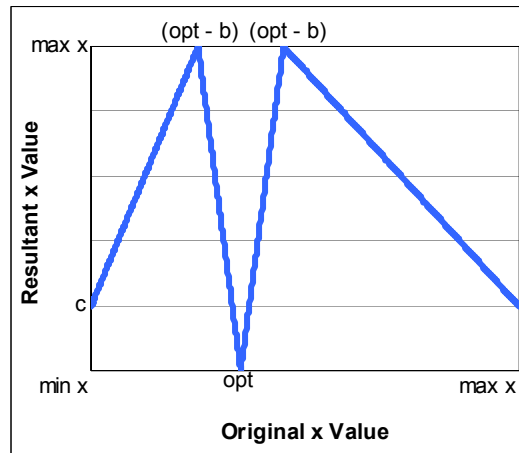


Figure A41 — Illustrating the *s\_decept* Function

### A.23.5 NONSEP\_SPECIAL

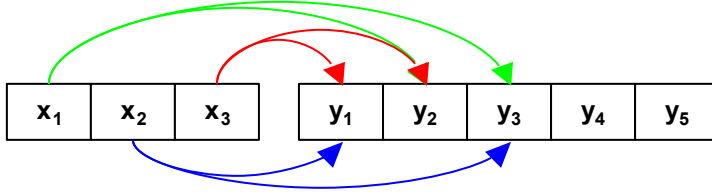
$$\text{nonsep\_special}(X, Y, a) = \frac{\sum_{i=1}^{|X|} X_i + \sum_{j=0}^{|Y|-a-1} |X_i - Y_{1+(i+j) \bmod (|X|)}|}{0.5 \times |X| \times (1 + |X|)} \quad (\text{A5})$$

where  $|Y| \geq |X|$ ,  $|Y| > a$ ,  $(|Y| - a) < |X|$

A newly defined function (inspired by Huband *et al.*'s *r\_nonsep* [158]), *nonsep\_special* takes two sets ( $X$  and  $Y$ ) and maps dependencies according to Equation (A6). Note that the function ensures that each member in  $X$  has exactly  $(|Y|-a)$  dependencies and that those dependencies can only come from the first  $|X|$  members of  $Y$  (see Figure A42 for an example). The function leads to particularly interesting forms of non-separability when  $X$  is a subset of  $Y$  — specifically, when  $(1+|Y|-a) = |X|$ , the first  $|X|$  members are mutually dependent upon each other; when  $(1+|Y|-a) < |X|$  the first  $|X|$  members are either completely indirectly dependent upon each other (when  $|Y|-a < 0.5 \times |X|$ ) or a mix of indirect and mutual dependence (when  $|Y|-a > 0.5 \times |X|$ ).

$$X_i \rightarrow Y_{1+(i+j) \bmod (|X|)} \quad \forall i \in \{1, 2, \dots, |X|\}, j \in \{1, 2, \dots, (|Y| - a)\} \quad (\text{A6})$$

where  $\rightarrow$  represents a dependency,

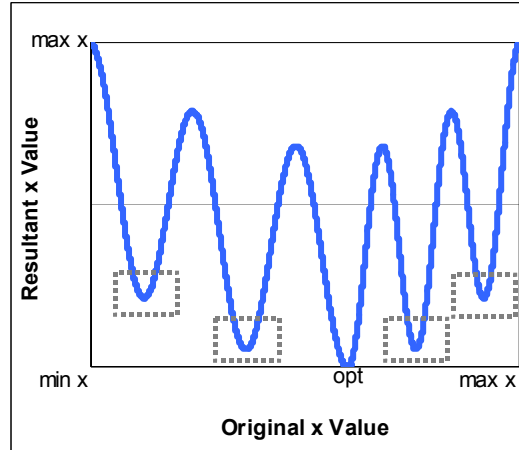


**Figure A42 — Illustrating Example Dependencies Caused by the *nonsep\_special* Function**  
Arrows represent dependencies. In this example,  $a = 3$ .

### A.23.6 S\_MULTI

$$\begin{aligned}
 f(x, a, opt) &= \cos \left( (4a + 2)\pi \times \left( 0.5 - \frac{|x - opt|}{2 \times (|opt - x| + opt)} \right) \right) \\
 g(x, b, opt) &= 4b \left( \frac{|x - opt|}{2 \times (|opt - x| + opt)} \right)^2 \\
 s\_multi(x, a, b, opt) &= \frac{1 + f(x, a, opt) + g(x, b, opt)}{b + 2}
 \end{aligned} \tag{A7}$$

Taken from Huband *et al.* [158], the  $s\_multi$  function recasts the variable  $x$  such that it maps to zero when at  $opt$ , and is affected by the presence of  $2a$  local minima when not at  $opt$ . The  $b$  parameter is used to control the depth of the gullies leading to local (and global) minima — with smaller settings resulting in lower local minima<sup>87</sup> and higher local maxima. Figure A43 offers an example use of the  $s\_multi$  function.



**Figure A43 — Illustrating an Example of Using  $s\_multi$**

Dashed boxes indicate local minima. In this example,  $opt = 0.7$ ,  $a = 2$  and  $b = 1$ . Dashed boxes illustrate deceptive minima (optima).

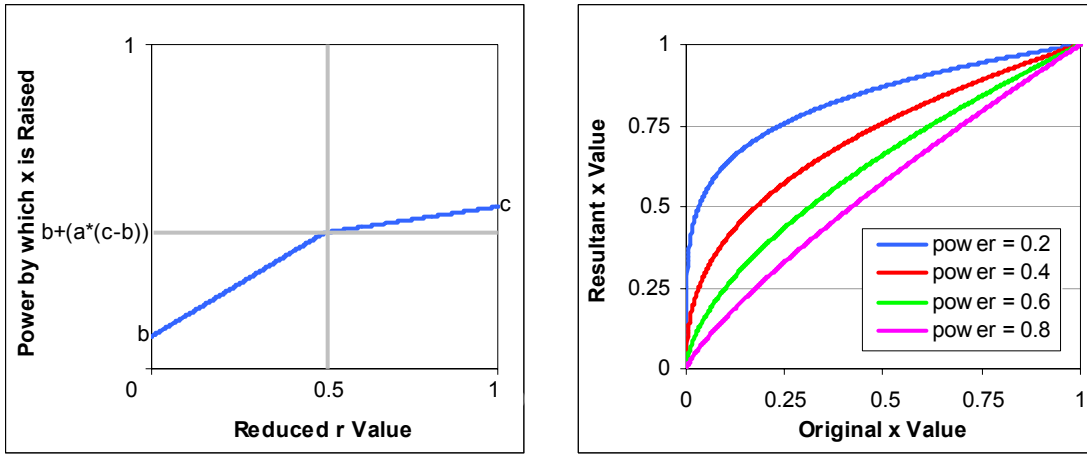
### A.23.7 B\_PARAM

$$\begin{aligned}
 f(r, a) &= a - (1 - 2r) \times \left| 0.5 - r \right| + a \\
 power(r, a, b, c) &= (b + (c - b) \times f(r, a)) \\
 b\_param(x, r, a, b, c) &= x^{power(r, a, b, c)}
 \end{aligned} \tag{A8}$$

Taken from Huband *et al.* [158], the  $b\_param$  function dictates how dependent  $x$  is upon other solution variables (represented here as a previously reduced value,  $r$ ). The  $a$ ,  $b$ ,  $c$  and  $r$  parameters dictate the power by which the original  $x$  value is raised — as per Figure A44. Higher values of  $b$  and  $c$  reduce the dependence of  $x$  on  $r$  by

<sup>87</sup> When  $b = 0$ , the local minima become global minima with values of zero.

encouraging higher *power* values. For clarity, the result of *b\_param* when applied to *x* for various *power* settings is illustrated in Figure A44.



(a) Determining the Result of *power*

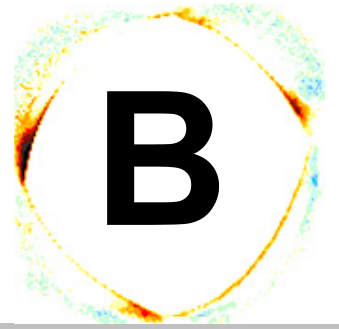
(b) How Applying *power* Affects *x*

**Figure A44 — Illustrating *b\_param***

Describes how the *power* component of *b\_param* is derived and applied.



# Appendix



Algorithm

Particulars

## B ALGORITHM PARTICULARS

### B.1 NSGA-II SETTINGS

#### B.1.1 CONFIGURATION ONE

Table A1 — Key Parameter Settings for NSGA-II

<b>Crossover Rate</b>	0.9 (using Simulated Binary Crossover)
<b>Mutation Rate</b>	$1/m$
<b>Population Size</b>	100, unless otherwise defined
<b>Constraints</b>	All feasible solutions are given preference; if required, infeasible solutions are ranked by their violation scores (with ties broken by dominance)

#### B.1.2 CONFIGURATION TWO

Table A2 — Key Parameter Settings for NSGA-II

<b>Crossover Rate</b>	0.9 (using Simulated Binary Crossover)
<b>Mutation Rate</b>	$1/m$
<b>Population Size</b>	50
<b>Constraints</b>	All feasible solutions are given preference; if required, infeasible solutions are ranked by their violation scores (with ties broken by dominance)

## B.2 MUTATION OPERATOR USED FOR ALL OPTIMISERS

All optimisers described in the thesis use an identical mutation operator (described in Equation (A9)) to ensure consistency. The  $\text{rand}()$  function returns an evenly distributed random number between zero and one, while  $e$  is Euler's number.

$$\text{mutate}(x, \min, \max) = \begin{cases} \text{mutate}^+(x, \omega, \max) & \text{rand}() < 0.5 \\ \text{mutate}^-(x, \omega, \min) & \text{otherwise} \end{cases} \quad (\text{A9})$$

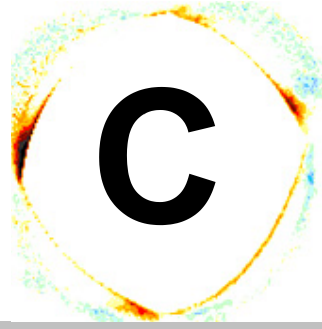
where:

$$\omega = e^{3.2 \times \text{rand}()} \times \frac{\text{range}}{100} - 0.0045$$

$$\text{mutate}^+(x, \omega, \max) = \begin{cases} x + \omega & (x + \omega) \leq \max \\ \text{rand}() \times (\max - x) + x & \text{otherwise} \end{cases}$$

$$\text{mutate}^-(x, \omega, \min) = \begin{cases} x - \omega & (x - \omega) \geq \min \\ \text{rand}() \times (x - \min) + \min & \text{otherwise} \end{cases}$$

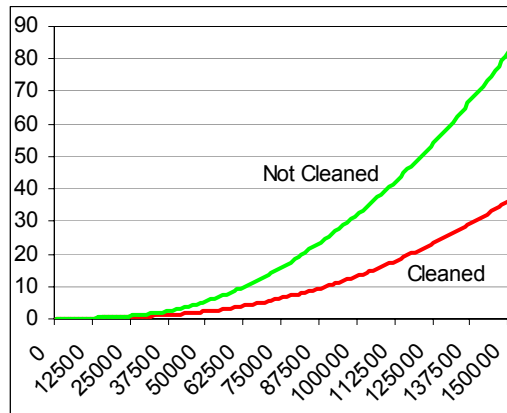
# Appendix



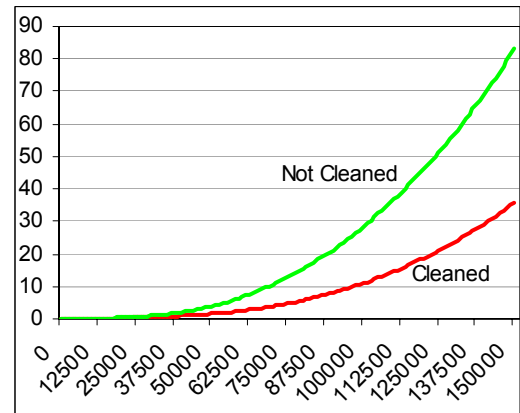
Supplemental  
Results

## C SUPPLEMENTAL RESULTS

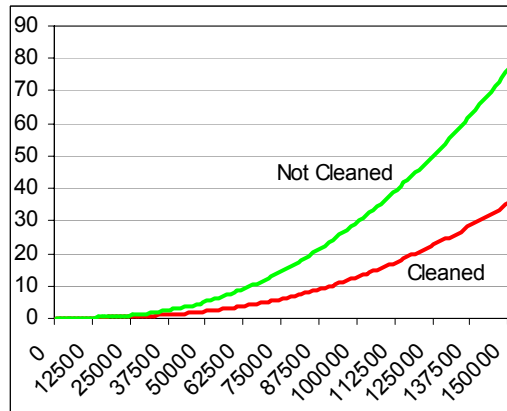
### C.1 THE EFFECT OF CLEANING DOMINATED TREES



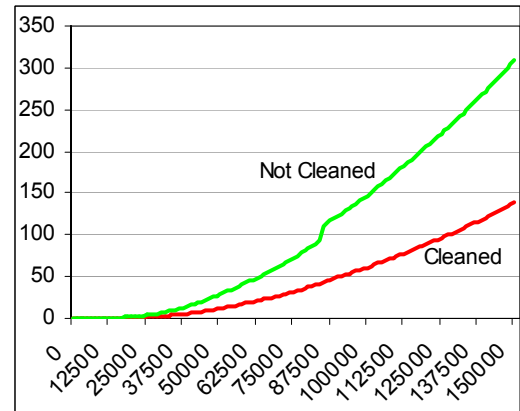
(a) AP-1



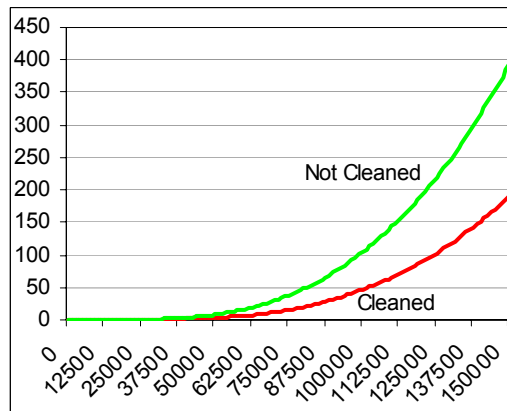
(b) AP-2



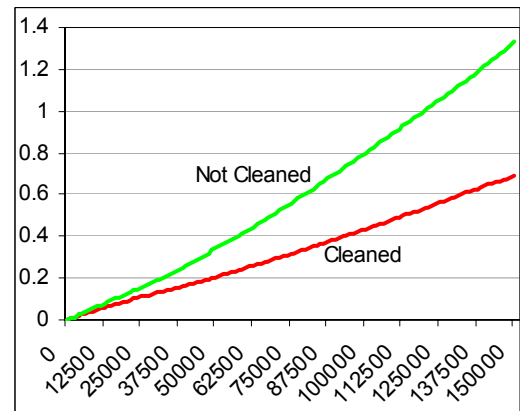
(c) AP-3



(d) AP-4



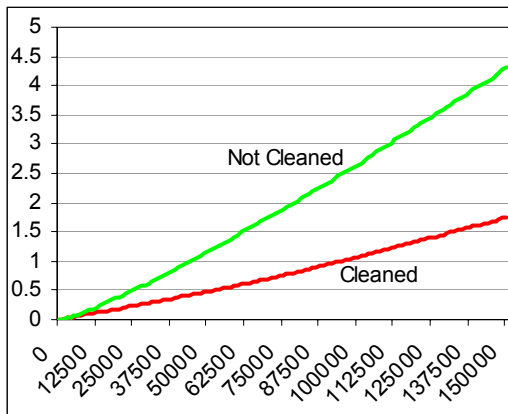
(e) AP-5



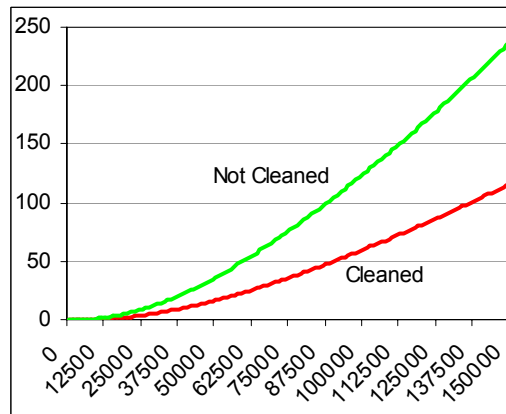
(f) AP-8

**Figure A45 — Comparative Cumulative Time Costs for Dominated Trees With and Without Cleaning**

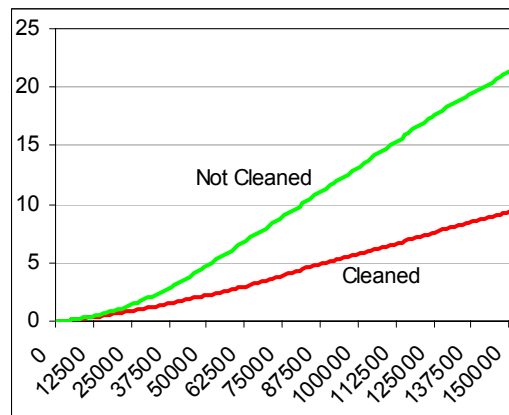
For all graphs, the *x-axis* represents the number of solutions presented to the archive and the *y-axis* is the average cumulative processing time in seconds



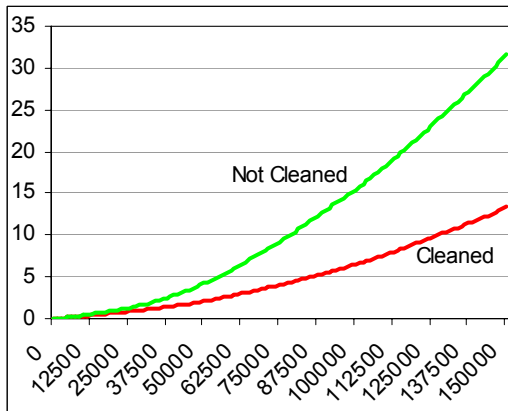
(a) AP-9



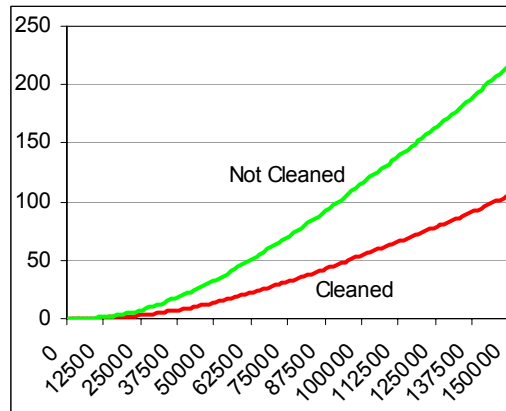
(b) AP-10



(c) AP-15



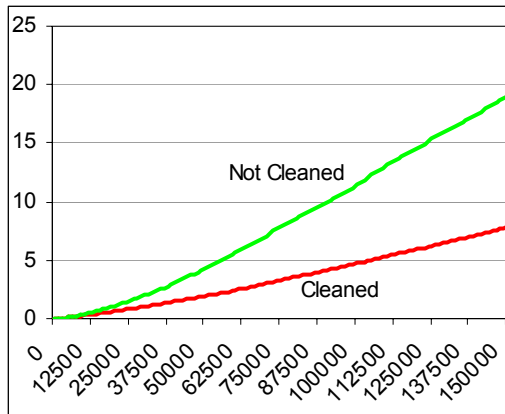
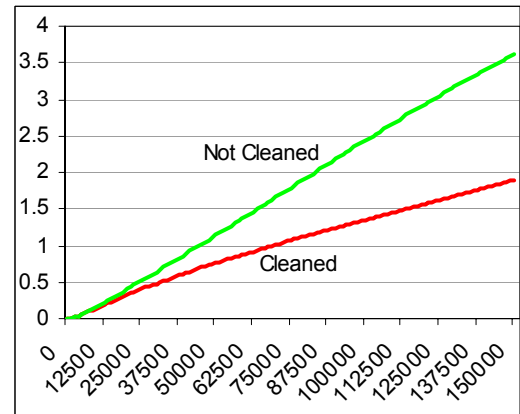
(e) AP-16



(f) AP-17

**Figure A46 — Comparative Cumulative Time Costs for Dominated Trees With and Without Cleaning**

For all graphs, the *x-axis* represents the number of solutions presented to the archive and the *y-axis* is the average cumulative processing time in seconds

(e) *AP-21*(f) *F-1*

**Figure A47 — Comparative Cumulative Time Costs for Dominated Trees With and Without Cleaning**

For all graphs, the *x-axis* represents the number of solutions presented to the archive and the *y-axis* is the average cumulative processing time in seconds

# Appendix



Nomenclature



## D NOMENCLATURE

### D.1 STYLE GUIDE

Table A3 — Style Guide Listing

$\mathbf{a}$	<b><i>Bold italics</i></b>	The solution vector $\mathbf{a}$ .
$a_i$	<i>Subscript italics</i>	The $i^{\text{th}}$ decision variable in solution $\mathbf{a}$ .
$A$	<i>Capital italics</i>	The collection $A$ .
$A_i$	<i>Subscript italics</i>	Element $i$ of collection $A$ .
$a^{\text{prop}}$	<i>Superscript italics</i>	$a$ has the property <i>prop</i> .
<code>fun (a, b)</code>	Courier	<i>fun</i> is a function, program or algorithm accepting parameters $a$ and $b$ .

### D.2 SPECIAL NOTATIONS

Table A4 — Special Notations Listing

$\mathbf{a}_{1..k}$	All decision variables of $\mathbf{a}$ with indexes in the range of $[1:k]$ .
$x \Rightarrow y$	The statement $x$ logically leads to the statement $y$ .
$v \approx w$	The value of $v$ is approximately equivalent to the value of $w$ .
$\tilde{a}$	The in-order list-predecessor of $a$ .
$\vec{a}$	The in-order list-successor of $a$ .
<i>significant</i>	Any <i>italicised</i> use of the word significant infers statistical significance (at the 5% level) according to a two-tailed Kruskal-Wallis test.

### D.3 TERMS AND OPERATORS

Table A5 — Terms and Operators Listing

$Y$	The complete decision-space.
$F$	The complete objective-space.
$f$	A multiobjective function (returns a list of objective-scores).
$f_i$	The $i^{\text{th}}$ multiobjective function (returns a single objective-score).
$f_i(\mathbf{a})$	The $i^{\text{th}}$ objective-score for solution $\mathbf{a}$ .
$\mathbf{a} \prec \mathbf{b}$	The vector performance of solution $\mathbf{a}$ strongly (strictly) dominates the vector performance of solution $\mathbf{b}$ .
$\mathbf{a} \preceq \mathbf{b}$	The vector performance of solution $\mathbf{a}$ weakly dominates the vector performance of solution $\mathbf{b}$ .
$\mathbf{a} = \mathbf{b}$	Solution $\mathbf{a}$ shares the same point in objective-space as solution $\mathbf{b}$ (equivalent vector performance).
$\mathbf{a} \sim \mathbf{b}$	The vector performance of solution $\mathbf{a}$ is incomparable with the vector performance of solution $\mathbf{b}$ .
$P_{\text{front}}$	The true (complete) Pareto front.
$P_{\text{estimate}}$	A non-dominated collection of solutions that estimate the Pareto front.
$P_{\text{local}}$	The non-dominated collection of solutions produced up to a given point in the optimisation process.
$\beta$	The number of objectives to be optimised.
$m$	The number of decision variables.
$n$	The size of the current population of solutions.
$\kappa$	The size of the neighbourhood for nearest-neighbour calculations.
$i_{\text{terminal}}$	The first member of the dominated set.
$j_{\text{terminal}}$	The last member of the dominated set.
$\mathcal{F}_1$	The leading (supposedly non-dominated) front in the NSGA-II archive.

## D.4 MAK\_TREE NODE PROPERTIES

All Mak\_Tree nodes (*node*) have access to (and maintain) the properties outlined in Table A6.

Table A6 — Mak\_Tree Properties Listing

$node^{parent}$	The parent of <i>node</i> or <code>null</code> if it does not exist.
$node^{leftChild}$	The left child of <i>node</i> or <code>null</code> if it does not exist.
$node^{rightChild}$	The right child of <i>node</i> or <code>null</code> if it does not exist.
$node^{obj1\_label}$	The objective one score associated with this <i>node</i> .
$node^{obj2\_label}$	The objective two score associated with this <i>node</i> .
$\overleftarrow{node}$	The in-order list-predecessor of <i>node</i> .
$\overrightarrow{node}$	The in-order list-successor of <i>node</i> .

Extended Mak\_Tree nodes (*node*) may have access to (and maintain) the properties outlined in Table A6.

Table A7 — Extended Mak\_Tree Properties Listing

$node^{annotation}$	The crowding annotation for <i>node</i> .
$node^{\kappa nnScore}$	The $\kappa$ nearest-neighbour score of <i>node</i> .
$node^{leftmostNeighbour}$	The leftmost neighbour of <i>node</i> in <i>node</i> 's collection of nearest neighbours; <code>null</code> if there is no such boundary neighbour.
$node^{rightmostNeighbour}$	The rightmost neighbour of <i>node</i> in <i>node</i> 's collection of nearest neighbours; <code>null</code> if there is no such boundary neighbour.
$node^{breadcrumb}$	An identifier used to determine which nodes have been visited during annotation updates.

## D.5 ACRONYMS

Table A8 and Table A9 define the acronyms used throughout this work.

**Table A8 — Acronyms (Part One)**

AP	The newly developed Alternative Problem suite.
CTP	Constrained Test Problem suite.
Diversity_PAES	The novel diversity-based hill-climbing system, founded on the principles of the Pareto Archived Evolutionary Strategy.
DMOEA	The Dynamic Multi-Objective Evolutionary Algorithm.
F1	Formula One.
FDA	Farina, Deb and Amato's dynamic test suite.
GA	Genetic Algorithm.
IBEA	The Indicator Based Evolutionary Algorithm.
IBEA_E	IBEA using an epsilon performance indicator.
IBEA_H	IBEA using a hypervolume performance indicator.
Mak_NSGA-II	The unbounded elitist Non-dominated Sorting Genetic Algorithm.
Mak_OS	The unbounded Overuse Selection algorithm with cuboid crowding.
Mak_OS_KNN	The unbounded Overuse Selection algorithm with averaged $\kappa$ nearest-neighbours crowding.
Mak_PAES	The unbounded Pareto Archived Evolutionary Strategy.
Mak_PESA	The unbounded Pareto Envelope Selection Algorithm.
Mak_SPEA2	The unbounded improved Strength Pareto Evolutionary Algorithm with cuboid crowding.
Mak_SPEA2_KNN	The unbounded improved Strength Pareto Evolutionary Algorithm with averaged $\kappa$ nearest-neighbours crowding..
MOEA	Multi-Objective Evolutionary Algorithm (generic term).
MOGA	The Multi-Objective Genetic Algorithm (specific algorithm).
NPGA	The Niche Pareto Genetic Algorithm.
NSGA	The Non-dominated Sorting Genetic Algorithm.
NSGA-II	The elitist Non-dominated Sorting Genetic Algorithm.

**Table A9 — Acronyms (Part Two)**

PAES	The Pareto Archived Evolutionary Strategy.
PESA	The Pareto Envelope Selection Algorithm.
PQRS	Partitioned Quasi-Random Selection scheme.
SPEA	The Strength Pareto Evolutionary Algorithm.
SPEA2	The improved Strength Pareto Evolutionary Algorithm.
WFG	The Walking Fish Group test suite.
ZDT	Zitzler, Deb and Thiele's test suite.

# Appendix



Full Listing of  
Epsilon and  
Hypervolume  
Statistical  
Inferences

## E FULL LISTING OF EPSILON AND HYPERVOLUME STATISTICAL INFERENCES

### E.1 HYPERVOLUME INFERENCES

	PAES	Mak PAES	Div. PAES	PESA	Mak PESA C	Mak PESA E	SPEA2	Mak SPEA2	Mak SPEA2 KNN	NSGA-II	Mak NSGA-II	Mak OS	Mak OS KNN	IBEA E	IBEA H
PAES	-	<b>4.28E-02</b>	<b>1.66E-08</b>	<b>2.19E-02</b>	6.36E-02	4.80E-01	<b>2.60E-02</b>	3.20E-01	<b>3.28E-03</b>	2.45E-01	<b>4.89E-05</b>	<b>1.39E-16</b>	<b>1.29E-22</b>	<b>3.93E-19</b>	<b>1.13E-10</b>
Mak PAES	<b>1.59E-05</b>	-	<b>1.93E-04</b>	<b>1.99E-05</b>	8.63E-01	<b>6.51E-03</b>	<b>2.65E-05</b>	3.00E-01	<b>1.00E-06</b>	3.85E-01	<b>3.75E-02</b>	<b>7.74E-11</b>	<b>3.96E-16</b>	<b>4.93E-13</b>	<b>4.82E-06</b>
Diversity PAES	8.05E-02	<b>8.82E-03</b>	-	<b>1.45E-14</b>	<b>9.89E-05</b>	<b>3.22E-10</b>	<b>2.28E-14</b>	<b>2.39E-06</b>	<b>1.58E-16</b>	<b>5.16E-06</b>	9.27E-02	<b>3.09E-03</b>	<b>1.87E-06</b>	<b>1.76E-04</b>	3.77E-01
PESA	2.88E-01	<b>9.97E-04</b>	4.92E-01	-	<b>4.09E-05</b>	1.11E-01	9.46E-01	<b>1.09E-03</b>	5.09E-01	<b>6.02E-04</b>	<b>5.39E-10</b>	<b>4.94E-24</b>	<b>1.34E-30</b>	<b>7.67E-27</b>	<b>3.25E-17</b>
Mak PESA C	<b>1.23E-13</b>	<b>7.64E-04</b>	<b>4.82E-09</b>	<b>9.41E-11</b>	-	<b>1.07E-02</b>	<b>5.40E-05</b>	3.87E-01	<b>2.25E-06</b>	4.86E-01	<b>2.45E-02</b>	<b>2.76E-11</b>	<b>1.19E-16</b>	<b>1.62E-13</b>	<b>2.19E-06</b>
Mak PESA E	<b>6.99E-11</b>	<b>1.76E-02</b>	<b>8.95E-07</b>	<b>2.80E-08</b>	3.11E-01	-	1.27E-01	8.97E-02	<b>2.47E-02</b>	6.23E-02	<b>2.22E-06</b>	<b>8.84E-19</b>	<b>5.25E-25</b>	<b>2.02E-21</b>	<b>1.50E-12</b>
SPEA2	8.88E-01	<b>8.59E-06</b>	5.91E-02	2.29E-01	<b>4.86E-14</b>	<b>3.00E-11</b>	-	<b>1.37E-03</b>	4.67E-01	<b>7.65E-04</b>	<b>7.94E-10</b>	<b>8.38E-24</b>	<b>2.33E-30</b>	<b>1.32E-26</b>	<b>5.24E-17</b>
Mak SPEA2	3.89E-01	<b>4.86E-04</b>	3.73E-01	8.39E-01	<b>2.79E-11</b>	<b>9.45E-09</b>	3.16E-01	-	<b>9.42E-05</b>	8.66E-01	<b>1.94E-03</b>	<b>1.18E-13</b>	<b>2.34E-19</b>	<b>4.80E-16</b>	<b>2.98E-08</b>
Mak SPEA2 KNN	8.39E-01	<b>6.53E-06</b>	5.13E-02	2.06E-01	<b>3.23E-14</b>	<b>2.06E-11</b>	9.51E-01	2.88E-01	-	<b>4.76E-05</b>	<b>1.06E-11</b>	<b>2.68E-26</b>	<b>5.74E-33</b>	<b>3.69E-29</b>	<b>2.74E-19</b>
NSGA-II	1.54E-01	<b>3.33E-03</b>	7.46E-01	7.15E-01	<b>7.86E-10</b>	<b>1.83E-07</b>	1.17E-01	5.70E-01	1.03E-01	-	<b>3.34E-03</b>	<b>3.52E-13</b>	<b>8.08E-19</b>	<b>1.54E-15</b>	<b>7.18E-08</b>
Mak NSGA-II	3.05E-01	<b>8.73E-04</b>	4.68E-01	9.70E-01	<b>7.50E-11</b>	<b>2.28E-08</b>	2.44E-01	8.69E-01	2.20E-01	6.87E-01	-	<b>4.62E-06</b>	<b>2.60E-10</b>	<b>8.95E-08</b>	<b>1.06E-02</b>
Mak OS	<b>6.79E-22</b>	<b>4.03E-09</b>	<b>2.54E-16</b>	<b>1.89E-18</b>	<b>8.03E-03</b>	<b>2.74E-04</b>	<b>2.30E-22</b>	<b>4.31E-19</b>	<b>1.43E-22</b>	<b>2.60E-17</b>	<b>1.43E-18</b>	-	6.06E-02	4.14E-01	<b>3.67E-02</b>
Mak OS KNN	<b>2.34E-24</b>	<b>5.97E-11</b>	<b>1.37E-18</b>	<b>8.42E-21</b>	<b>7.64E-04</b>	<b>1.42E-05</b>	<b>7.73E-25</b>	<b>1.82E-21</b>	<b>4.75E-25</b>	<b>1.28E-19</b>	<b>6.32E-21</b>	4.64E-01	-	2.87E-01	<b>8.65E-05</b>
IBEA E	<b>3.71E-11</b>	<b>1.32E-02</b>	<b>5.37E-07</b>	<b>1.59E-08</b>	3.64E-01	9.16E-01	<b>1.57E-11</b>	<b>5.31E-09</b>	<b>1.08E-11</b>	<b>1.07E-07</b>	<b>1.30E-08</b>	<b>4.05E-04</b>	<b>2.23E-05</b>	-	<b>3.82E-03</b>
IBEA H	<b>2.71E-02</b>	<b>1.85E-10</b>	<b>8.88E-05</b>	<b>1.14E-03</b>	<b>1.99E-20</b>	<b>3.28E-17</b>	<b>3.83E-02</b>	<b>2.24E-03</b>	<b>4.44E-02</b>	<b>3.09E-04</b>	<b>1.30E-03</b>	<b>1.55E-29</b>	<b>3.77E-32</b>	<b>1.54E-17</b>	-

(a) AP-1 (white) and AP-2 (dark grey)

	PAES	Mak PAES	Div. PAES	PESA	Mak PESA C	Mak PESA E	SPEA2	Mak SPEA2	Mak SPEA2 KNN	NSGA-II	Mak NSGA-II	Mak OS	Mak OS KNN	IBEA E	IBEA H
PAES	-	<b>9.49E-04</b>	<b>5.35E-16</b>	3.67E-01	4.81E-01	<b>1.98E-05</b>	9.76E-01	<b>1.31E-03</b>	2.30E-01	<b>6.81E-05</b>	<b>7.66E-10</b>	<b>2.50E-20</b>	<b>6.33E-28</b>	<b>2.04E-22</b>	<b>1.86E-10</b>
Mak PAES	8.67E-01	-	<b>2.82E-07</b>	<b>2.98E-05</b>	<b>8.87E-03</b>	3.18E-01	1.05E-03	9.25E-01	3.35E-02	4.83E-01	2.69E-03	1.54E-10	7.25E-17	3.27E-12	1.20E-03
Diversity PAES	2.43E-01	1.82E-01	-	<b>8.70E-19</b>	<b>6.26E-14</b>	<b>2.77E-05</b>	<b>6.56E-16</b>	<b>1.76E-07</b>	<b>1.56E-12</b>	<b>7.64E-06</b>	<b>2.63E-02</b>	1.67E-01	<b>3.40E-04</b>	<b>4.45E-02</b>	<b>4.77E-02</b>
PESA	2.85E-01	3.68E-01	<b>2.58E-02</b>	-	1.09E-01	<b>3.07E-07</b>	3.52E-01	<b>4.42E-05</b>	<b>3.60E-02</b>	<b>1.30E-06</b>	<b>3.46E-12</b>	<b>2.57E-23</b>	<b>4.02E-31</b>	<b>1.78E-25</b>	<b>7.37E-13</b>
Mak PESA C	5.76E-01	6.96E-01	8.46E-02	6.10E-01	-	<b>3.29E-04</b>	5.00E-01	<b>1.16E-02</b>	6.18E-01	<b>9.58E-04</b>	<b>3.64E-08</b>	<b>4.42E-18</b>	<b>1.78E-25</b>	<b>4.23E-20</b>	<b>9.88E-09</b>
Mak PESA E	2.11E-01	2.78E-01	<b>1.59E-02</b>	8.54E-01	4.88E-01	-	<b>2.24E-05</b>	2.75E-01	<b>1.89E-03</b>	7.66E-01	<b>4.36E-02</b>	<b>3.99E-08</b>	<b>6.63E-14</b>	<b>1.27E-09</b>	<b>2.38E-02</b>
SPEA2	<b>2.17E-02</b>	<b>3.31E-02</b>	<b>5.81E-04</b>	2.17E-01	8.14E-02	2.93E-01	-	<b>1.45E-03</b>	2.41E-01	<b>7.67E-05</b>	<b>9.06E-10</b>	<b>3.11E-20</b>	<b>8.03E-28</b>	<b>2.56E-22</b>	<b>2.21E-10</b>
Mak SPEA2	4.65E-01	5.73E-01	5.81E-02	7.35E-01	8.63E-01	6.01E-01	1.16E-01	-	<b>4.21E-02</b>	4.26E-01	<b>1.98E-03</b>	<b>8.79E-11</b>	<b>3.72E-17</b>	<b>1.81E-12</b>	<b>8.65E-04</b>
Mak SPEA2 KNN	<b>1.03E-02</b>	<b>1.64E-02</b>	<b>2.12E-04</b>	1.32E-01	<b>4.41E-02</b>	1.86E-01	7.85E-01	<b>6.54E-02</b>	-	<b>4.86E-03</b>	<b>4.55E-07</b>	<b>1.53E-16</b>	<b>8.96E-24</b>	<b>1.66E-18</b>	<b>1.34E-07</b>
NSGA-II	<b>7.84E-03</b>	<b>1.26E-02</b>	<b>1.47E-04</b>	1.09E-01	<b>3.50E-02</b>	1.56E-01	7.12E-01	5.27E-02	9.23E-01	-	<b>2.08E-02</b>	<b>8.16E-09</b>	<b>9.14E-15</b>	<b>2.29E-10</b>	<b>1.07E-02</b>
Mak NSGA-II	<b>3.29E-04</b>	<b>6.07E-04</b>	<b>2.49E-06</b>	<b>1.08E-02</b>	<b>2.30E-03</b>	<b>1.79E-02</b>	1.86E-01	<b>3.98E-03</b>	2.93E-01	3.39E-01	-	<b>3.52E-04</b>	<b>1.27E-08</b>	<b>2.87E-05</b>	8.07E-01
Mak OS	4.08E-01	3.20E-01	7.32E-01	5.86E-02	1.66E-01	<b>3.81E-02</b>	<b>1.88E-03</b>	1.20E-01	<b>7.44E-04</b>	<b>5.28E-04</b>	<b>1.16E-05</b>	-	<b>2.57E-02</b>	5.26E-01	<b>8.47E-04</b>
Mak OS KNN	5.31E-02	<b>3.57E-02</b>	4.41E-01	<b>2.83E-03</b>	<b>1.29E-02</b>	<b>1.55E-03</b>	<b>2.89E-05</b>	<b>7.93E-03</b>	<b>8.94E-06</b>	<b>5.83E-06</b>	<b>5.68E-08</b>	2.66E-01	-	1.09E-01	<b>4.66E-08</b>
IBEA E	<b>1.35E-02</b>	<b>2.12E-02</b>	<b>3.05E-04</b>	1.58E-01	5.52E-02	2.20E-01	8.60E-01	8.07E-02	9.23E-01	8.47E-01	2.51E-01	<b>1.04E-03</b>	<b>1.36E-05</b>	-	<b>7.85E-05</b>
IBEA H	<b>9.02E-06</b>	<b>1.87E-05</b>	<b>3.12E-08</b>	<b>6.43E-04</b>	<b>9.42E-05</b>	<b>1.22E-03</b>	<b>2.77E-02</b>	<b>1.84E-04</b>	5.34E-02	6.63E-02	3.76E-01	<b>1.81E-07</b>	<b>4.43E-10</b>	<b>4.27E-02</b>	-

(b) AP-3 (white) and AP-4 (dark grey)

Figure A48 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Hypervolume Results  
 Bold Italics indicate significant differences at the 5% level.

	PAES	Mak PAES	Div. PAES	PESA	Mak PESA C	Mak PESA E	SPEA2	Mak SPEA2	Mak SPEA2 KNN	NSGA-II	Mak NSGA-II	Mak OS	Mak OS KNN	IBEA E	IBEA H
PAES	-	2.16E-05	1.21E-11	1.04E-30	3.72E-05	1.85E-06	1.55E-44	2.03E-34	4.63E-51	2.33E-29	6.33E-31	2.63E-02	3.43E-04	3.00E-12	3.02E-24
Mak PAES		-	6.37E-03	1.81E-46	1.00E-15	8.40E-18	2.79E-60	2.96E-50	1.39E-66	4.54E-45	1.09E-46	3.77E-02	4.87E-01	8.66E-26	1.16E-39
Diversity PAES			-	1.64E-56	1.43E-24	6.46E-27	5.73E-70	3.16E-60	4.84E-76	3.93E-55	9.95E-57	2.17E-06	6.59E-04	1.45E-35	9.27E-50
PESA				-	1.12E-16	1.20E-14	1.81E-04	3.03E-01	5.84E-08	7.04E-01	9.52E-01	8.48E-39	6.56E-44	2.66E-08	6.50E-02
Mak PESA C					-	4.97E-01	2.63E-29	6.50E-20	1.08E-35	1.57E-15	7.36E-17	5.59E-10	1.07E-13	2.10E-03	2.16E-11
Mak PESA E						-	6.46E-27	9.35E-18	3.17E-33	1.48E-13	8.02E-15	9.75E-12	1.11E-15	1.60E-02	1.17E-09
SPEA2							-	6.09E-03	7.66E-02	3.96E-05	2.27E-04	1.00E-52	8.88E-58	7.97E-19	3.95E-08
Mak SPEA2								-	8.29E-06	1.59E-01	3.32E-01	1.39E-42	1.06E-47	8.11E-11	4.23E-03
Mak SPEA2 KNN									-	7.76E-09	7.96E-08	3.63E-59	3.94E-64	1.09E-24	1.32E-12
NSGA-II										-	6.60E-01	2.09E-37	1.64E-42	1.89E-07	1.42E-01
Mak NSGA-II											-	5.11E-39	3.94E-44	1.94E-08	5.68E-02
Mak OS												-	1.65E-01	7.41E-19	4.51E-32
Mak OS KNN													-	2.04E-23	4.09E-37
IBEA E														-	1.35E-04

(a) AP-5 (white)

	PAES	Mak PAES	Div. PAES	PESA	Mak PESA C	Mak PESA E	SPEA2	Mak SPEA2	Mak SPEA2 KNN	NSGA-II	Mak NSGA-II	Mak OS	Mak OS KNN	IBEA E	IBEA H
PAES	-	2.22E-01	1.13E-34	3.08E-01	1.76E-04	4.81E-06	2.06E-04	4.51E-23	5.75E-21	9.50E-19	5.00E-27	3.04E-10	1.41E-18	8.98E-26	2.23E-17
Mak PAES	1.53E-01	-	2.87E-30	2.54E-02	1.05E-02	6.76E-04	1.18E-02	4.63E-19	4.48E-17	5.26E-15	7.74E-23	2.29E-07	7.57E-15	1.24E-21	9.70E-14
Diversity PAES	4.71E-02	5.75E-01	-	2.14E-38	2.10E-21	1.25E-18	1.53E-21	1.09E-03	1.04E-04	5.71E-06	3.28E-02	4.63E-13	4.48E-06	1.27E-02	7.45E-07
PESA	8.85E-03	2.30E-01	5.22E-01	-	2.30E-06	3.25E-08	2.78E-06	1.45E-26	2.21E-24	4.58E-22	1.25E-30	5.91E-13	6.92E-22	2.40E-29	1.26E-20
Mak PESA C	4.60E-17	7.18E-13	2.37E-11	1.03E-09	-	3.90E-01	9.67E-01	1.69E-11	7.18E-10	3.14E-08	1.02E-14	6.80E-03	4.17E-08	1.12E-13	2.94E-07
Mak PESA E	8.04E-12	2.95E-08	5.10E-07	9.97E-06	7.02E-02	-	3.68E-01	2.56E-09	7.66E-08	2.23E-06	2.72E-12	6.31E-02	2.86E-06	2.53E-11	1.58E-05
SPEA2	5.12E-12	2.01E-08	3.57E-07	7.22E-06	8.21E-02	9.42E-01	-	1.31E-11	5.67E-10	2.53E-08	7.73E-15	6.02E-03	3.36E-08	8.56E-14	2.40E-07
Mak SPEA2	1.35E-39	2.26E-34	2.41E-32	4.74E-30	3.30E-10	3.41E-15	5.58E-15	-	5.26E-01	1.87E-01	2.49E-01	2.47E-05	1.69E-01	4.28E-01	7.92E-02
Mak SPEA2 KNN	1.36E-30	1.43E-25	1.18E-23	1.64E-21	7.15E-05	1.36E-08	2.01E-08	1.36E-02	-	4.92E-01	7.45E-02	3.09E-04	4.58E-01	1.54E-01	2.61E-01
NSGA-II	7.78E-28	6.44E-23	4.69E-21	5.54E-19	1.29E-03	7.26E-07	1.03E-06	1.23E-03	4.36E-01	-	1.38E-02	3.29E-03	9.57E-01	3.51E-02	6.62E-01
Mak NSGA-II	2.13E-45	3.93E-40	4.50E-38	9.87E-36	1.71E-14	4.39E-20	7.54E-20	1.16E-01	6.30E-05	2.11E-06	-	1.13E-07	1.19E-02	7.18E-01	3.82E-03
Mak OS	4.26E-26	2.93E-21	1.95E-19	2.04E-17	6.33E-03	7.34E-06	1.01E-05	2.04E-04	2.02E-01	6.18E-01	1.89E-07	-	3.90E-03	6.80E-07	1.21E-02
Mak OS KNN	5.20E-25	3.14E-20	1.96E-18	1.89E-16	1.55E-02	2.87E-05	3.89E-05	5.87E-05	1.12E-01	4.16E-01	3.74E-08	7.53E-01	-	3.08E-02	7.01E-01
IBEA E	2.32E-31	2.59E-26	2.21E-24	3.19E-22	2.95E-05	4.23E-09	6.30E-09	2.41E-02	8.30E-01	3.20E-01	1.48E-04	1.36E-01	7.13E-02	-	1.11E-02
IBEA H	9.75E-21	3.14E-16	1.45E-14	9.56E-13	2.48E-01	3.17E-03	3.98E-03	1.76E-07	4.39E-03	3.74E-02	2.75E-11	1.12E-01	2.03E-01	2.22E-03	-

(b) AP-15 (white) and AP-16 (dark grey)

Figure A49 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Hypervolume Results  
Bold Italics indicate *significant* differences at the 5% level.



	PAES	Mak PAES	Div. PAES	PESA	Mak PESA C	Mak PESA E	SPEA2	Mak SPEA2	Mak SPEA2 KNN	NSGA-II	Mak NSGA-II	Mak OS	Mak OS KNN	IBEA E	IBEA H
PAES	-	3.01E-01	<b>6.90E-49</b>	<b>3.26E-02</b>	<b>9.15E-18</b>	<b>9.81E-12</b>	<b>2.25E-25</b>	<b>3.44E-44</b>	<b>1.70E-39</b>	<b>6.20E-29</b>	<b>1.01E-45</b>	<b>4.45E-11</b>	<b>1.12E-16</b>	<b>1.46E-34</b>	<b>1.24E-16</b>
Mak PAES	3.36E-01	-	<b>4.44E-45</b>	2.68E-01	<b>1.22E-14</b>	<b>4.16E-09</b>	<b>7.10E-22</b>	<b>2.22E-40</b>	<b>1.04E-35</b>	<b>2.53E-25</b>	<b>6.57E-42</b>	<b>1.60E-08</b>	<b>1.26E-13</b>	<b>7.83E-31</b>	<b>1.38E-13</b>
Diversity PAES	1.55E-01	6.46E-01	-	<b>5.39E-41</b>	<b>1.21E-16</b>	<b>2.51E-23</b>	<b>3.57E-10</b>	2.02E-01	<b>1.11E-02</b>	<b>9.40E-08</b>	3.90E-01	<b>3.61E-24</b>	<b>9.91E-18</b>	<b>1.16E-04</b>	<b>8.91E-18</b>
PESA	<b>7.47E-03</b>	8.46E-02	2.05E-01	-	<b>1.51E-11</b>	<b>1.24E-06</b>	<b>2.75E-18</b>	<b>2.57E-36</b>	<b>1.07E-31</b>	<b>1.40E-21</b>	<b>7.78E-38</b>	<b>3.94E-06</b>	<b>1.26E-10</b>	<b>6.46E-27</b>	<b>1.37E-10</b>
Mak PESA C	<b>7.23E-13</b>	<b>2.60E-10</b>	<b>3.56E-09</b>	<b>2.29E-06</b>	-	<b>3.87E-02</b>	<b>2.14E-02</b>	<b>6.42E-13</b>	<b>1.42E-09</b>	<b>9.63E-04</b>	<b>4.30E-14</b>	<b>2.08E-02</b>	7.24E-01	<b>1.56E-06</b>	7.13E-01
Mak PESA E	<b>1.71E-07</b>	<b>1.55E-05</b>	<b>1.04E-04</b>	<b>8.10E-03</b>	<b>3.18E-02</b>	-	<b>1.59E-05</b>	<b>3.91E-19</b>	<b>3.31E-15</b>	<b>1.32E-07</b>	<b>1.79E-20</b>	8.04E-01	8.59E-02	<b>2.03E-11</b>	8.86E-02
SPEA2	<b>2.34E-05</b>	<b>9.60E-04</b>	<b>4.33E-03</b>	1.09E-01	<b>1.44E-03</b>	2.90E-01	-	<b>3.40E-07</b>	<b>1.01E-04</b>	3.08E-01	<b>4.12E-08</b>	<b>5.33E-06</b>	<b>8.09E-03</b>	<b>1.00E-02</b>	<b>7.74E-03</b>
Mak SPEA2	<b>7.91E-15</b>	<b>4.11E-12</b>	<b>6.81E-11</b>	<b>7.92E-08</b>	4.92E-01	<b>4.75E-03</b>	<b>1.17E-04</b>	-	2.02E-01	<b>3.55E-05</b>	6.78E-01	<b>6.23E-20</b>	<b>6.52E-14</b>	<b>9.00E-03</b>	<b>5.91E-14</b>
Mak SPEA2 KNN	<b>2.65E-09</b>	<b>4.13E-07</b>	<b>3.65E-06</b>	<b>6.38E-04</b>	1.71E-01	4.32E-01	6.58E-02	<b>4.04E-02</b>	-	<b>3.74E-03</b>	9.14E-02	<b>6.01E-16</b>	<b>1.86E-10</b>	1.77E-01	<b>1.71E-10</b>
NSGA-II	<b>7.62E-08</b>	<b>7.73E-06</b>	<b>5.49E-05</b>	<b>5.05E-03</b>	<b>4.66E-02</b>	8.73E-01	2.24E-01	<b>7.64E-03</b>	5.31E-01	-	<b>5.89E-06</b>	<b>3.66E-08</b>	<b>2.70E-04</b>	1.17E-01	<b>2.55E-04</b>
Mak NSGA-II	<b>4.98E-13</b>	<b>1.85E-10</b>	<b>2.57E-09</b>	<b>1.75E-06</b>	9.54E-01	<b>2.75E-02</b>	<b>1.18E-03</b>	5.29E-01	1.54E-01	<b>4.06E-02</b>	-	<b>2.75E-21</b>	<b>4.05E-15</b>	<b>2.54E-03</b>	<b>3.66E-15</b>
Mak OS	<b>2.81E-12</b>	<b>8.98E-10</b>	<b>1.15E-08</b>	<b>6.10E-06</b>	8.30E-01	5.30E-02	<b>2.91E-03</b>	3.67E-01	2.49E-01	7.56E-02	7.85E-01	-	<b>4.95E-02</b>	<b>4.39E-12</b>	5.12E-02
Mak OS KNN	<b>6.28E-14</b>	<b>2.78E-11</b>	<b>4.23E-10</b>	<b>3.78E-07</b>	7.06E-01	<b>1.18E-02</b>	<b>3.82E-04</b>	7.56E-01	8.15E-02	<b>1.82E-02</b>	7.50E-01	5.54E-01	-	<b>2.83E-07</b>	9.88E-01
IBEA E	<b>3.19E-23</b>	<b>4.83E-20</b>	<b>1.43E-18</b>	<b>1.04E-14</b>	<b>9.43E-04</b>	<b>8.45E-08</b>	<b>2.53E-10</b>	<b>8.41E-03</b>	<b>3.82E-06</b>	<b>1.90E-07</b>	<b>1.15E-03</b>	<b>4.39E-04</b>	<b>3.28E-03</b>	-	<b>2.63E-07</b>
IBEA H	<b>1.62E-18</b>	<b>1.47E-15</b>	<b>3.27E-14</b>	<b>9.62E-11</b>	5.85E-02	<b>6.40E-05</b>	<b>5.69E-07</b>	2.27E-01	<b>1.20E-03</b>	<b>1.21E-04</b>	6.66E-02	<b>3.53E-02</b>	1.29E-01	1.50E-01	-

(a) AP-17 (white) and AP-21 (dark grey)

Figure A50 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Hypervolume Results  
Bold Italics indicate *significant* differences at the 5% level.

## E.2 EPSILON INFERENCES

	PAES	Mak PAES	Div. PAES	PESA	Mak PESA C	Mak PESA E	SPEA2	Mak SPEA2	Mak SPEA2 KNN	NSGA-II	Mak NSGA-II	Mak OS	Mak OS KNN	IBEA E	IBEA H
PAES	-	8.53E-02	<b>2.81E-02</b>	1.12E-01	5.20E-02	<b>1.43E-02</b>	1.08E-01	<b>2.02E-10</b>	<b>2.76E-03</b>	<b>1.53E-11</b>	<b>4.13E-12</b>	<b>7.93E-24</b>	<b>6.48E-28</b>	<b>1.03E-16</b>	1.02E-01
Mak PAES	<b>2.12E-03</b>	-	6.31E-01	<b>1.02E-03</b>	8.23E-01	4.62E-01	9.07E-01	<b>1.84E-06</b>	1.97E-01	<b>2.29E-07</b>	<b>7.80E-08</b>	<b>3.49E-18</b>	<b>5.19E-22</b>	<b>9.29E-12</b>	9.33E-01
Diversity PAES	<b>5.39E-03</b>	7.67E-01	-	<b>1.77E-04</b>	7.98E-01	7.98E-01	5.51E-01	<b>1.59E-05</b>	4.17E-01	<b>2.31E-06</b>	<b>8.49E-07</b>	<b>1.07E-16</b>	<b>1.96E-20</b>	<b>1.67E-10</b>	5.73E-01
PESA	7.04E-01	<b>5.78E-04</b>	<b>1.61E-03</b>	-	<b>4.60E-04</b>	<b>6.43E-05</b>	<b>1.51E-03</b>	<b>8.76E-15</b>	<b>6.01E-06</b>	<b>4.62E-16</b>	<b>1.05E-16</b>	<b>2.38E-29</b>	<b>1.36E-33</b>	<b>8.65E-22</b>	<b>1.36E-03</b>
Mak PESA C	<b>2.20E-03</b>	9.92E-01	7.75E-01	<b>6.00E-04</b>	-	6.09E-01	7.34E-01	<b>5.14E-06</b>	2.86E-01	<b>6.88E-07</b>	<b>2.43E-07</b>	<b>1.75E-17</b>	<b>2.85E-21</b>	<b>3.64E-11</b>	7.58E-01
Mak PESA E	1.01E-01	1.46E-01	2.47E-01	<b>4.39E-02</b>	1.49E-01	-	3.94E-01	<b>4.67E-05</b>	5.79E-01	<b>7.40E-06</b>	<b>2.83E-06</b>	<b>6.37E-16</b>	<b>1.32E-19</b>	<b>7.36E-10</b>	4.12E-01
SPEA2	4.37E-01	<b>1.30E-04</b>	<b>4.01E-04</b>	6.91E-01	<b>1.36E-04</b>	<b>1.61E-02</b>	-	<b>1.06E-06</b>	1.60E-01	<b>1.27E-07</b>	<b>4.27E-08</b>	<b>1.50E-18</b>	<b>2.13E-22</b>	<b>4.52E-12</b>	9.74E-01
Mak SPEA2	1.68E-01	<b>1.07E-05</b>	<b>3.79E-05</b>	3.17E-01	<b>1.12E-05</b>	<b>2.71E-03</b>	5.47E-01	-	<b>4.05E-04</b>	6.67E-01	5.20E-01	<b>1.28E-05</b>	<b>4.40E-08</b>	<b>2.59E-02</b>	<b>1.23E-06</b>
Mak SPEA2 KNN	8.19E-01	<b>9.83E-04</b>	<b>2.64E-03</b>	8.80E-01	<b>1.02E-03</b>	6.21E-02	5.84E-01	2.50E-01	-	<b>7.76E-05</b>	<b>3.25E-05</b>	<b>2.76E-14</b>	<b>7.59E-18</b>	<b>1.60E-08</b>	1.69E-01
NSGA-II	1.64E-01	<b>1.01E-05</b>	<b>3.59E-05</b>	3.11E-01	<b>1.06E-05</b>	<b>2.60E-03</b>	5.38E-01	9.90E-01	2.45E-01	-	8.32E-01	<b>7.76E-05</b>	<b>3.90E-07</b>	7.16E-02	<b>1.50E-07</b>
Mak NSGA-II	7.35E-01	<b>6.10E-03</b>	<b>1.42E-02</b>	4.73E-01	<b>6.30E-03</b>	1.93E-01	2.65E-01	8.66E-02	5.71E-01	8.43E-02	-	<b>1.79E-04</b>	<b>1.09E-06</b>	1.12E-01	<b>5.04E-08</b>
Mak OS	<b>1.00E-11</b>	<b>8.13E-05</b>	<b>2.40E-05</b>	<b>9.30E-13</b>	<b>7.80E-05</b>	<b>1.06E-07</b>	<b>7.09E-14</b>	<b>1.23E-15</b>	<b>2.41E-12</b>	<b>1.12E-15</b>	<b>7.73E-11</b>	-	2.37E-01	<b>2.85E-02</b>	<b>1.90E-18</b>
Mak OS KNN	<b>1.21E-15</b>	<b>1.54E-07</b>	<b>3.34E-08</b>	<b>8.62E-17</b>	<b>1.46E-07</b>	<b>4.85E-11</b>	<b>5.07E-18</b>	<b>6.13E-20</b>	<b>2.48E-16</b>	<b>5.57E-20</b>	<b>1.20E-14</b>	1.67E-01	-	<b>8.07E-04</b>	<b>2.72E-22</b>
IBEA E	<b>6.40E-07</b>	<b>4.72E-02</b>	<b>2.28E-02</b>	<b>9.68E-08</b>	<b>4.61E-02</b>	<b>6.46E-04</b>	<b>1.20E-08</b>	<b>4.14E-10</b>	<b>2.08E-07</b>	<b>3.84E-10</b>	<b>3.15E-06</b>	<b>4.58E-02</b>	<b>7.99E-04</b>	-	<b>5.51E-12</b>
IBEA H	<b>8.05E-05</b>	<b>9.84E-12</b>	<b>5.94E-11</b>	<b>3.47E-04</b>	<b>1.05E-11</b>	<b>3.99E-08</b>	<b>1.41E-03</b>	<b>9.27E-03</b>	<b>1.97E-04</b>	<b>9.62E-03</b>	<b>1.99E-05</b>	<b>4.94E-24</b>	<b>7.94E-29</b>	<b>1.67E-17</b>	-

(a) AP-1 (white) and AP-2 (dark grey)

	PAES	Mak PAES	Div. PAES	PESA	Mak PESA C	Mak PESA E	SPEA2	Mak SPEA2	Mak SPEA2 KNN	NSGA-II	Mak NSGA-II	Mak OS	Mak OS KNN	IBEA E	IBEA H
PAES	-	7.47E-01	<b>4.95E-04</b>	1.21E-01	7.47E-01	7.36E-02	9.49E-02	5.77E-02	9.34E-01	8.55E-02	<b>3.46E-05</b>	<b>4.15E-10</b>	<b>4.21E-15</b>	<b>4.21E-10</b>	<b>4.72E-08</b>
Mak PAES	2.61E-01	-	<b>1.52E-03</b>	6.13E-02	5.19E-01	1.42E-01	<b>4.66E-02</b>	1.15E-01	6.85E-01	1.62E-01	<b>1.27E-04</b>	<b>2.59E-09</b>	<b>3.67E-14</b>	<b>2.63E-09</b>	<b>2.44E-07</b>
Diversity PAES	8.73E-01	3.35E-01	-	<b>6.86E-07</b>	<b>1.48E-04</b>	8.50E-02	<b>3.82E-07</b>	1.07E-01	<b>3.66E-04</b>	7.32E-02	4.95E-01	<b>3.44E-03</b>	<b>2.86E-06</b>	<b>3.46E-03</b>	<b>3.76E-02</b>
PESA	1.47E-01	<b>1.04E-02</b>	1.08E-01	-	2.19E-01	<b>9.13E-04</b>	9.04E-01	<b>6.20E-04</b>	1.42E-01	<b>1.16E-03</b>	<b>2.14E-08</b>	<b>2.59E-14</b>	<b>6.56E-20</b>	<b>2.63E-14</b>	<b>6.54E-12</b>
Mak PESA C	6.22E-01	1.06E-01	5.14E-01	3.38E-01	-	<b>3.50E-02</b>	1.77E-01	<b>2.66E-02</b>	8.10E-01	<b>4.14E-02</b>	<b>8.65E-06</b>	<b>6.23E-11</b>	<b>4.60E-16</b>	<b>6.32E-11</b>	<b>8.47E-09</b>
Mak PESA E	9.44E-01	2.92E-01	9.29E-01	1.29E-01	5.73E-01	-	<b>5.98E-04</b>	9.12E-01	6.13E-02	9.44E-01	<b>1.65E-02</b>	<b>4.47E-06</b>	<b>3.49E-10</b>	<b>4.52E-06</b>	<b>1.66E-04</b>
SPEA2	<b>8.06E-03</b>	<b>1.81E-04</b>	<b>5.02E-03</b>	2.25E-01	<b>3.05E-02</b>	<b>6.56E-03</b>	-	<b>4.01E-04</b>	1.12E-01	<b>7.67E-04</b>	<b>1.13E-08</b>	<b>1.16E-14</b>	<b>2.68E-20</b>	<b>1.18E-14</b>	<b>3.10E-12</b>
Mak SPEA2	<b>2.12E-02</b>	<b>6.62E-04</b>	<b>1.38E-02</b>	3.88E-01	6.93E-02	<b>1.76E-02</b>	7.26E-01	-	<b>4.77E-02</b>	8.57E-01	<b>2.21E-02</b>	<b>7.32E-06</b>	<b>6.58E-10</b>	<b>7.41E-06</b>	<b>2.52E-04</b>
Mak SPEA2 KNN	<b>7.03E-03</b>	<b>1.51E-04</b>	<b>4.35E-03</b>	2.08E-01	<b>2.71E-02</b>	<b>5.71E-03</b>	9.63E-01	6.91E-01	-	7.16E-02	<b>2.45E-05</b>	<b>2.57E-10</b>	<b>2.40E-15</b>	<b>2.61E-10</b>	<b>3.06E-08</b>
NSGA-II	5.90E-02	<b>2.74E-03</b>	<b>4.07E-02</b>	6.58E-01	1.62E-01	5.03E-02	4.41E-01	6.74E-01	4.13E-01	-	<b>1.36E-02</b>	<b>3.25E-06</b>	<b>2.32E-10</b>	<b>3.28E-06</b>	<b>1.26E-04</b>
Mak NSGA-II	<b>1.05E-02</b>	<b>2.56E-04</b>	<b>6.60E-03</b>	2.62E-01	<b>3.81E-02</b>	<b>8.57E-03</b>	9.27E-01	7.95E-01	8.90E-01	4.96E-01	-	<b>2.42E-02</b>	<b>5.56E-05</b>	<b>2.43E-02</b>	1.61E-01
Mak OS	1.05E-01	6.17E-01	1.44E-01	<b>2.28E-03</b>	<b>3.49E-02</b>	1.21E-01	<b>2.41E-05</b>	<b>1.01E-04</b>	<b>1.97E-05</b>	<b>4.99E-04</b>	<b>3.54E-05</b>	-	6.89E-02	9.98E-01	3.90E-01
Mak OS KNN	5.46E-01	6.03E-01	6.57E-01	<b>4.05E-02</b>	2.73E-01	5.94E-01	<b>1.20E-03</b>	<b>3.76E-03</b>	<b>1.02E-03</b>	<b>1.30E-02</b>	<b>1.63E-03</b>	3.08E-01	-	6.85E-02	<b>7.62E-03</b>
IBEA E	<b>4.94E-05</b>	<b>3.02E-07</b>	<b>2.54E-05</b>	<b>8.06E-03</b>	<b>3.38E-04</b>	<b>3.70E-05</b>	1.47E-01	7.22E-02	1.61E-01	<b>2.68E-02</b>	1.24E-01	<b>2.33E-08</b>	<b>3.60E-06</b>	-	3.91E-01
IBEA H	<b>6.17E-09</b>	<b>8.82E-12</b>	<b>2.56E-09</b>	<b>8.28E-06</b>	<b>8.43E-08</b>	<b>4.20E-09</b>	<b>9.97E-04</b>	<b>2.82E-04</b>	<b>1.17E-03</b>	<b>5.43E-05</b>	<b>7.25E-04</b>	<b>3.78E-13</b>	<b>2.02E-10</b>	<b>6.22E-02</b>	-

(b) AP-3 (white) and AP-4 (dark grey)

Figure A51 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Epsilon Results  
 Bold Italics indicate *significant* differences at the 5% level.

	PAES	Mak PAES	Div. PAES	PESA	Mak PESA C	Mak PESA E	SPEA2	Mak SPEA2	Mak SPEA2 KNN	NSGA-II	Mak NSGA-II	Mak OS	Mak OS KNN	IBEA E	IBEA H
PAES	-	1.09E-01	1.47E-05	1.53E-28	4.42E-05	2.05E-06	3.86E-38	6.52E-30	3.50E-39	2.03E-24	2.97E-33	1.13E-06	1.20E-10	1.39E-09	1.46E-15
Mak PAES		-	5.46E-03	2.67E-34	2.19E-08	4.55E-10	4.73E-44	1.05E-35	4.24E-45	4.99E-30	4.09E-39	8.72E-04	6.93E-07	7.31E-14	1.37E-20
Diversity PAES			-	1.50E-44	7.20E-16	5.17E-18	2.59E-54	5.65E-46	2.39E-55	3.56E-40	2.11E-49	5.73E-01	2.35E-02	1.36E-22	3.54E-30
PESA				-	5.71E-15	5.72E-13	8.15E-03	6.99E-01	3.45E-03	2.35E-01	1.88E-01	1.26E-46	6.67E-53	2.74E-09	9.94E-05
Mak PESA_C					-	4.85E-01	2.02E-23	4.03E-16	2.19E-24	1.23E-11	5.16E-19	1.36E-17	3.86E-23	3.54E-02	2.28E-05
Mak PESA_E						-	4.29E-21	4.61E-14	4.94E-22	7.77E-10	7.88E-17	8.34E-20	1.61E-25	1.58E-01	3.65E-04
SPEA2							-	2.35E-02	7.76E-01	1.43E-04	1.80E-01	2.31E-56	1.63E-62	1.29E-16	1.86E-10
Mak SPEA2								-	1.09E-02	1.16E-01	3.51E-01	4.75E-48	2.59E-54	3.03E-10	2.02E-05
Mak SPEA2 KNN									-	4.59E-05	1.04E-01	2.16E-57	1.59E-63	1.72E-17	3.42E-11
NSGA-II										-	1.26E-02	3.00E-42	1.51E-48	1.26E-06	6.19E-03
Mak NSGA-II											-	1.80E-51	1.07E-57	1.04E-12	2.72E-07
Mak OS												-	8.79E-02	1.69E-24	3.39E-32
Mak OS KNN													-	1.78E-30	2.06E-38
IBEA_E														-	2.91E-02

(a) AP-5 (white)

	PAES	Mak PAES	Div. PAES	PESA	Mak PESA C	Mak PESA E	SPEA2	Mak SPEA2	Mak SPEA2 KNN	NSGA-II	Mak NSGA-II	Mak OS	Mak OS KNN	IBEA E	IBEA H
PAES	-	8.35E-01	7.76E-19	7.97E-02	4.11E-02	9.42E-02	1.91E-13	4.14E-11	6.56E-13	3.63E-11	1.37E-15	2.43E-01	8.86E-10	1.71E-10	1.59E-01
Mak PAES	3.78E-01	-	1.68E-19	1.22E-01	2.46E-02	6.01E-02	4.88E-14	1.17E-11	1.71E-13	1.02E-11	3.24E-16	1.69E-01	2.67E-10	4.97E-11	1.07E-01
Diversity PAES	4.36E-05	1.21E-03	-	1.22E-24	9.97E-13	9.00E-14	7.38E-02	8.38E-03	4.80E-02	8.93E-03	2.92E-01	3.00E-15	1.65E-03	4.11E-03	1.52E-14
PESA	1.32E-03	1.89E-02	3.64E-01	-	1.70E-04	6.73E-04	1.01E-18	4.57E-16	4.04E-18	3.92E-16	4.15E-21	3.68E-03	1.61E-14	2.35E-15	1.69E-03
Mak PESA_C	1.82E-09	1.98E-07	4.00E-02	3.21E-03	-	7.10E-01	3.43E-08	2.41E-06	9.27E-08	2.18E-06	5.90E-10	3.78E-01	2.45E-05	7.11E-06	5.22E-01
Mak PESA_E	3.33E-10	4.32E-08	1.89E-02	1.21E-03	7.66E-01	-	4.64E-09	4.08E-07	1.31E-08	3.66E-07	6.67E-11	6.11E-01	4.80E-06	1.29E-06	7.89E-01
SPEA2	1.88E-16	7.48E-14	6.37E-06	8.08E-08	1.17E-02	2.59E-02	-	3.90E-01	8.48E-01	4.03E-01	4.60E-01	2.59E-10	1.68E-01	2.73E-01	1.03E-09
Mak SPEA2	7.09E-24	6.18E-21	3.32E-11	1.09E-13	2.14E-06	8.24E-06	2.20E-02	-	5.04E-01	9.83E-01	1.11E-01	3.06E-08	6.01E-01	8.12E-01	1.07E-07
Mak SPEA2 KNN	3.32E-25	3.19E-22	3.03E-12	8.23E-15	3.30E-07	1.39E-06	7.51E-03	6.97E-01	-	5.18E-01	3.52E-01	7.82E-10	2.34E-01	3.65E-01	3.03E-09
NSGA-II	3.78E-19	2.10E-16	1.00E-07	7.33E-10	7.60E-04	2.09E-03	3.87E-01	1.52E-01	6.89E-02	-	1.16E-01	2.72E-08	5.86E-01	7.95E-01	9.53E-08
Mak NSGA-II	2.17E-18	1.11E-15	3.34E-07	2.83E-09	1.72E-03	4.46E-03	5.31E-01	9.49E-02	3.98E-02	8.11E-01	-	2.96E-12	3.46E-02	6.71E-02	1.32E-11
Mak OS	7.82E-12	1.44E-09	3.06E-03	1.22E-04	3.56E-01	5.31E-01	1.08E-01	1.13E-04	2.30E-05	1.38E-02	2.59E-02	-	4.36E-07	1.05E-07	8.10E-01
Mak OS KNN	6.21E-13	1.41E-10	7.96E-04	2.35E-05	1.85E-01	3.04E-01	2.28E-01	5.17E-04	1.19E-04	3.88E-02	6.74E-02	6.87E-01	-	7.76E-01	1.39E-06
IBEA_E	1.30E-18	6.78E-16	2.35E-07	1.90E-09	1.36E-03	3.58E-03	4.86E-01	1.10E-01	4.71E-02	8.66E-01	9.43E-01	2.16E-02	5.75E-02	-	3.52E-07
IBEA_H	5.96E-07	3.21E-05	3.40E-01	6.32E-02	2.69E-01	1.61E-01	3.20E-04	8.02E-09	9.18E-10	9.48E-06	2.66E-05	4.32E-02	1.55E-02	1.97E-05	-

(b) AP-15 (white) and AP-16 (dark grey)

Figure A52 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Epsilon Results

Bold Italics indicate *significant* differences at the 5% level.

	PAES	Mak PAES	Div. PAES	PESA	Mak PESA C	Mak PESA E	SPEA2	Mak SPEA2	Mak SPEA2 KNN	NSGA- II	Mak NSGA- II	Mak OS	Mak OS KNN	IBEA E	IBEA H
<b>PAES</b>	-	5.50E-01	<b>2.25E-53</b>	5.90E-01	<b>5.94E-16</b>	<b>9.30E-07</b>	<b>1.94E-28</b>	<b>5.36E-42</b>	<b>2.87E-41</b>	<b>1.34E-30</b>	<b>5.60E-38</b>	<b>1.38E-07</b>	<b>4.89E-21</b>	<b>1.18E-30</b>	<b>1.51E-11</b>
<b>Mak PAES</b>	3.86E-01	-	<b>3.45E-51</b>	9.54E-01	<b>3.41E-14</b>	<b>1.42E-05</b>	<b>2.38E-26</b>	<b>8.44E-40</b>	<b>4.50E-39</b>	<b>1.76E-28</b>	<b>8.53E-36</b>	<b>2.49E-06</b>	<b>4.19E-19</b>	<b>1.55E-28</b>	<b>5.24E-10</b>
<b>Diversity PAES</b>	<b>2.20E-04</b>	<b>4.33E-03</b>	-	<b>2.11E-51</b>	<b>1.83E-22</b>	<b>5.44E-35</b>	<b>4.66E-11</b>	<b>2.13E-03</b>	<b>1.10E-03</b>	<b>1.62E-09</b>	<b>3.63E-05</b>	<b>1.39E-33</b>	<b>2.89E-17</b>	<b>1.76E-09</b>	<b>8.74E-28</b>
<b>PESA</b>	<b>2.82E-01</b>	<b>8.33E-01</b>	<b>8.13E-03</b>	-	<b>2.32E-14</b>	<b>1.10E-05</b>	<b>1.49E-26</b>	<b>5.16E-40</b>	<b>2.75E-39</b>	<b>1.10E-28</b>	<b>5.23E-36</b>	<b>1.90E-06</b>	<b>2.73E-19</b>	<b>9.68E-29</b>	<b>3.75E-10</b>
<b>Mak PESA_C</b>	<b>4.13E-07</b>	<b>2.19E-05</b>	1.51E-01	<b>5.27E-05</b>	-	<b>4.18E-04</b>	<b>1.85E-04</b>	<b>6.58E-13</b>	<b>2.31E-12</b>	<b>1.54E-05</b>	<b>5.13E-10</b>	<b>1.62E-03</b>	1.07E-01	<b>1.44E-05</b>	1.21E-01
<b>Mak PESA_E</b>	<b>1.05E-05</b>	<b>3.49E-04</b>	4.58E-01	<b>7.46E-04</b>	4.87E-01	-	<b>1.99E-12</b>	<b>4.80E-24</b>	<b>2.25E-23</b>	<b>3.88E-14</b>	<b>2.10E-20</b>	6.98E-01	<b>4.07E-07</b>	<b>3.50E-14</b>	<b>4.47E-02</b>
<b>SPEA2</b>	<b>5.35E-06</b>	<b>1.97E-04</b>	3.71E-01	<b>4.33E-04</b>	5.86E-01	8.79E-01	-	<b>2.18E-04</b>	<b>4.55E-04</b>	5.42E-01	<b>8.51E-03</b>	<b>2.20E-11</b>	<b>3.07E-02</b>	5.32E-01	<b>1.88E-07</b>
<b>Mak SPEA2</b>	<b>2.01E-16</b>	<b>7.21E-14</b>	<b>1.01E-06</b>	<b>2.84E-13</b>	<b>4.37E-04</b>	<b>2.84E-05</b>	<b>5.32E-05</b>	-	8.43E-01	<b>1.90E-03</b>	2.74E-01	<b>9.79E-23</b>	<b>9.37E-09</b>	<b>1.99E-03</b>	<b>1.76E-17</b>
<b>Mak SPEA2 KNN</b>	<b>1.36E-10</b>	<b>1.77E-08</b>	<b>3.75E-03</b>	<b>5.37E-08</b>	1.39E-01	<b>3.01E-02</b>	<b>4.36E-02</b>	<b>3.88E-02</b>	-	<b>3.58E-03</b>	3.70E-01	<b>4.48E-22</b>	<b>2.70E-08</b>	<b>3.76E-03</b>	<b>7.17E-17</b>
<b>NSGA-II</b>	<b>2.37E-11</b>	<b>3.63E-09</b>	<b>1.46E-03</b>	<b>1.14E-08</b>	7.70E-02	<b>1.40E-02</b>	<b>2.11E-02</b>	7.55E-02	7.70E-01	-	<b>4.23E-02</b>	<b>4.88E-13</b>	<b>5.77E-03</b>	9.88E-01	<b>7.76E-09</b>
<b>Mak NSGA-II</b>	<b>6.00E-14</b>	<b>1.48E-11</b>	<b>4.23E-05</b>	<b>5.29E-11</b>	<b>6.95E-03</b>	<b>7.28E-04</b>	<b>1.23E-03</b>	4.02E-01	2.17E-01	3.46E-01	-	<b>3.74E-19</b>	<b>2.32E-06</b>	<b>4.39E-02</b>	<b>3.33E-14</b>
<b>Mak OS</b>	<b>4.18E-09</b>	<b>3.86E-07</b>	<b>2.10E-02</b>	<b>1.07E-06</b>	3.79E-01	1.16E-01	1.55E-01	<b>7.86E-03</b>	5.48E-01	3.73E-01	6.71E-02	-	<b>2.59E-06</b>	<b>4.42E-13</b>	1.05E-01
<b>Mak OS KNN</b>	<b>3.20E-13</b>	<b>6.99E-11</b>	<b>1.19E-04</b>	<b>2.42E-10</b>	<b>1.44E-02</b>	<b>1.75E-03</b>	<b>2.88E-03</b>	2.74E-01	3.28E-01	4.92E-01	7.97E-01	1.15E-01	-	<b>5.50E-03</b>	<b>1.69E-03</b>
<b>IBEA_E</b>	<b>1.32E-10</b>	<b>1.73E-08</b>	<b>3.70E-03</b>	<b>5.24E-08</b>	1.38E-01	<b>2.98E-02</b>	<b>4.31E-02</b>	<b>3.92E-02</b>	9.96E-01	7.74E-01	2.19E-01	5.45E-01	3.30E-01	-	<b>7.13E-09</b>
<b>IBEA_H</b>	<b>7.76E-12</b>	<b>1.31E-09</b>	<b>7.78E-04</b>	<b>4.24E-09</b>	5.13E-02	<b>8.41E-03</b>	<b>1.29E-02</b>	1.10E-01	6.35E-01	8.55E-01	4.47E-01	2.83E-01	6.14E-01	6.39E-01	-

(a) *AP*-17 (white) and *AP*-21 (dark grey)

**Figure A53 — Two-Tailed Kruskal-Wallis Tests on End-of-Run Epsilon Results**  
Bold Italics indicate *significant* differences at the 5% level.

# Appendix



## Miscellaneous

## F MISCELLANEOUS

### F.1 AVAILABILITY OF SOURCE CODE FOR RED-BLACK TREE IMPLEMENTATIONS

Table A10 lists websites offering full implementations of the basic Red-Black tree for a variety of contemporary programming languages. The resources are reliable as at March 2007.

**Table A10 - Online Sources for Red-Black Tree Implementations**

C
<a href="http://web.mit.edu/~emin/www/source_code/red_black_tree/index.html">http://web.mit.edu/~emin/www/source_code/red_black_tree/index.html</a> <a href="http://sourceforge.net/projects/libredblack/">http://sourceforge.net/projects/libredblack/</a>
C#
<a href="http://www.codeplex.com/NGenerics">http://www.codeplex.com/NGenerics</a>
C++
<a href="http://sourceforge.net/projects/libredblack/">http://sourceforge.net/projects/libredblack/</a> <a href="http://web.mit.edu/~emin/www/source_code/red_black_tree/index.html">http://web.mit.edu/~emin/www/source_code/red_black_tree/index.html</a>
Java
As <i>TreeMap</i> from the standard library (Java 1.2–1.6: <i>current version as at the time of writing</i> ): <a href="http://java.sun.com/javase/downloads/index.jsp">http://java.sun.com/javase/downloads/index.jsp</a>  As a stand-alone implementation: <a href="http://gee.cs.oswego.edu/dl/classes/collections">http://gee.cs.oswego.edu/dl/classes/collections</a>

## F.2 FREQUENCY MATRIX SETTINGS

All displayed frequency matrices from Chapter 9 onwards represent attained regions lying in the space bounded by the objective-one and objective-two range values (Table A11).

Table A11 - Frequency Matrix Dimensions

Function	Objective-One Range	Objective-Two Range
AP-1	[0,1]	[0,1.2]
AP-2	[0,1]	[0,1.2]
AP-3	[0,1]	[0,1.2]
AP-4	[0,1]	[0,10]
AP-5	[0,1]	[0,1.2]
AP-15	[0,2.5]	[0,4.5]
AP-16	[0,2.5]	[0,4.5]
AP-17	[0,2.5]	[0,4.5]
AP-21	[0,2.5]	[0,4.5]

## F.3 FURTHER ILLUSTRATIONS

### F.3.1 LOCATING DOMINATED NODES

Figure A54 illustrates an additional example of inserting a dominating solution into the basic Mak\_Tree.

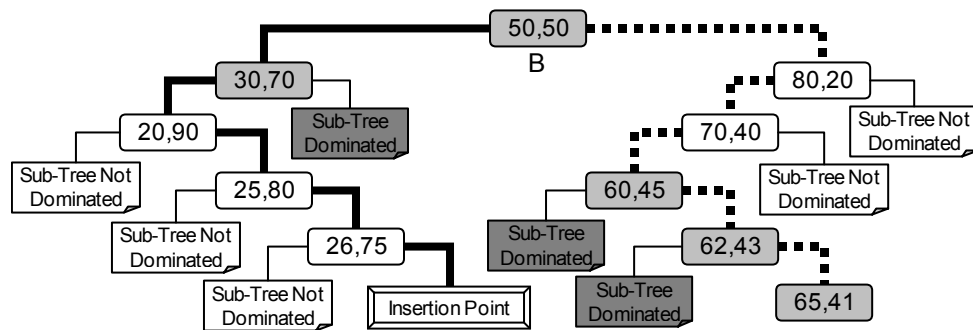


Figure A54 — Locating Dominated Nodes in an Example Mak\_Tree

Assume that a solution with results (29,40) is being inserted into the tree and that *B* is the first dominated node to be identified. The dashed bold line represents the  $O(\log n)$  path taken to identify all dominated nodes to the right of *B*. The solid bold line represents the  $O(\log n)$  path taken to both identify all dominated nodes to the left of *B* and insert the solution at the correct location in the tree. For clarity, individual dominated nodes are lightly shaded, while dominated sub-trees feature heavy shading.