AFIT/DS/ENG/99-01

Multiobjective Evolutionary Algorithms:

Classifications, Analyses, and New Innovations

DISSERTATION

David A. Van Veldhuizen
Captain, USAF

AFIT/DS/ENG/99-01

AFIT/DS/ENG/99-01

Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations

DISSERTATION

Presented to the Faculty of the Graduate School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

David A. Van Veldhuizen, A.A.S., B.G.S., M.S.
Captain, USAF

June, 1999

AFIT/DS/ENG/99-01

Multiobjective Evolutionary Algorithms

Classifications, Analyses, and New Innovations

David A. Van Veldhuizen, A.A.S., B.G.S., M.S.

Captain, USAF

Approved:

Dr Gary B. Lamont, Chairman

Dr Richard F. Deckro

Dr (Maj) Laurence D. Merkle

Dr (Maj) Thomas F. Reid

Dr Curtis H. Spenny, Dean's Representative

Dr Robert A. Calico, Jr.
Dean, Graduate School of Engineering

Acknowledgements

Completing the requirements to receive a Doctor of Philosophy degree strongly reminds me of completing a marathon. It's demanded a great deal of time, dedication, preparation, training, and the proper equipment. It's required an internal drive and desire to see the race through. Like a marathon, it's also benefited from support along the course.

Although too many to mention by name you know who you are. Past instructors, co-workers, classmates, office mates, supervisors and supervisees – you've all had an impact on me, some even positive! Your investment in me is partly responsible for this success. However, certain individuals were absolutely key to completing this marathon.

Doc Lamont – you've guided, coaxed, criticized, supported, humored, and continually motivated me for almost three years. Most of all, you've genuinely cared about my family and I. As this "lifetime of learning" continues I may encounter strange loops bringing me back to AFIT. I feel fortunate to have you as a mentor and friend. As I re-enter the Air Force mainstream know that my research hat goes with me.

Mom and Dad – you've always set a Christian example I'd do well to emulate. Your prayers that something good would finally come of me appear to be giving fruit. Thanks for teaching me to do a good job the first time, to appreciate the value of hard work, and especially for never giving up!

Loni and Jack – you've had a hard go of it as I've been constantly locked up in the study or behind a book for the past three years. Thank you so much for letting me do this "geek" thing, and especially for keeping things straight on the home front. I'm glad we're traveling together through this journey called life! I love you both. Can we go out to play now?

> ... but those who hope in the Lord will renew their strength. They will soar on wings like eagles; they will run and not grow weary, they will walk and not be faint.
> **Isaiah 40:31**

David A. Van Veldhuizen

Table of Contents

Page

Page

Page

Page

List of Figures

Figure Page

Figure | Page

Figure | Page

Figure | Page

Figure		Page

List of Tables

Table		Page

List of Acronyms

BBs	Building Blocks
DAG	Directed Acyclic Graph
DM	Decision Maker
EAs	Evolutionary Algorithms
EC	Evolutionary Computation
EP	Evolutionary Programming
ESs	*Evolutionsstrategies*, or Evolution Strategies
EVOPs	Evolutionary Operators
fmGA	fast messy GA
GAs	Genetic Algorithms
GEATbx	Genetic and Evolutionary Algorithm Toolbox
GP	Genetic Programming
mGA	messy GA
MOEAs	Multiobjective Evolutionary Algorithms
MOEP	Multiobjective Evolutionary Programming
MOES	Multiobjective Evolutionary Strategies
MOGA	Multiobjective Genetic Algorithm
MOGP	Multiobjective Genetic Programming
MOMGA	Multiobjective messy Genetic Algorithm
MOP	Multiobjective Optimization Problem
MPI	Message Passing Interface
NFL	No Free Lunch
NP	Nondeterministic Polynomial
NPGA	Niched Pareto Genetic Algorithm
NSGA	Nondominated Sorting Genetic Algorithm
OR	Operations Research
PEI	Partially Enumerative Initialization
SA	Simulated Annealing
SPEA	Strength Pareto Evolutionary Algorithm
SPMD	Single Program Multiple Data
VEGA	Vector Evaluated Genetic Algorithm

AFIT/DS/ENG/99-01

Abstract

Although computational techniques for solving Multiobjective Optimization Problems (MOPs) have been available for many years, the recent application of Evolutionary Algorithms (EAs) to such problems provides a vehicle with which to solve very large scale MOPs. This research classifies and analyzes contemporary Multiobjective Evolutionary Algorithm (MOEA) research and associated MOPs. Under the umbrella of *a priori*, progressive, and *a posteriori* algorithms, all currently known MOEAs proposed in the literature are classified and cataloged. The classification also incorporates detailed algorithmic characteristics, such as objective aggregation, interactive methods, sampling, ranking, and niching. Using a consistent MOEA terminology and notation, each cited MOEAs' key factors are presented in tabular form for ease of MOEA identification and selection. This effort currently classifies 218 distinct MOEA research efforts and applications (representing 272 separate references).

A detailed quantitative and qualitative MOEA analysis is presented. The classified efforts provide a basis for analyses about various algorithmic techniques, fitness functions, gene representations, and the problem domains within which MOEAs are applied. On a qualitative level MOEA "state of the art" is discussed, addressing topics such as MOEA characteristics, theory, additional populations, complexity, and well-engineered MOEA implementations. New theorems and definitions are also presented.

This research extends the traditional notion of *building blocks* to the MOP domain in an effort to develop more effective and efficient MOEAs. An innovative extension of an existing building block-based EA to the MOP domain (named the MOMGA), and the engineering design decisions made during its construction are presented.

The MOEA community's limited *de facto* test suites contain various MOP functions, many of whose origins and rationale for use are unknown. Thus, example MOPs from the current MOEA literature are presented in tabular form and classified based upon problem domain genotype and phenotype characteristics; these include connectivity, disjointness, concave or convex shape, constraints, and symmetry. Using general test suite guidelines,

more comprehensive MOEA test function suites are generated based upon MOP characteristics and applicable MOEA theory.

Few efforts *quantitatively* measure MOEA performance; fewer still compare MOEA results to MOPs with known optima. Using a developed MOEA test function suite, an experimental methodology incorporating known MOP solutions and appropriate test suite metrics is offered as a proposed evaluation framework allowing for absolute comparisons of specific MOEA approaches. This framework is then used in experiments with three well-known MOEAs and the MOMGA, examining their performance in regard to test MOPs. Experimental results, their statistical analyses, and other germane observations are presented. The MOMGA is shown to be at least as effective as other MOEAs tested and often more so.

Taken together, this document's classifications, analyses, and new innovations present a complete, contemporary view of current MOEA "state of the art" and possible future research. Researchers with basic EA knowledge may also use part of it as a largely self-contained introduction to MOEAs.

Multiobjective Evolutionary Algorithms:

Classifications, Analyses, and New Innovations

I. Introduction and Overview

It always takes longer than you expect, even when you take into account Hofstadter's Law.
Douglas Hofstadter, *Gödel, Escher, Bach*

1.1 Introduction

With or without conscious thought people make decisions throughout every day of their lives. These decisions may be as simple as deciding what clothes to wear or as difficult as those involved in engineering a space shuttle's design. The former decision is made in a matter of seconds while one of the latter may take years, with the attendant difficulties of changing priorities, rising costs, changing resource levels, and so on. Oftentimes these problems are viewed as minimizing cost while maximizing gain. This research focuses on complex types of these *optimization problems.*

Consider the very simple example of purchasing a car. The purchaser wishes to satisfy the following criteria: minimizing the car's cost, insurance premium, and weight (for towing behind a motor home), and maximizing its "fun." The purchaser also desires said vehicle to meet the following conditions: seats six adults (comfortably), provides all-time four-wheel drive and a "premium" stereo system, blue or black two-tone paint, and a minimum 75 miles per gallon. In mathematical terminology the available vehicles (makes and models) are the problem's *decision variables*, the conditions to be met are the *constraints*, and the process of minimizing and maximizing the criteria is called *optimization.* An *objective function* based on the decision variables is used to determine an associated vector representing how "well" some particular vehicle satisfies the criteria of vehicle and insurance cost, weight, and “fun.” Because multiple objectives are simultaneously considered this problem is termed a Multiobjective Optimization Problem (MOP).

This simple example highlights many difficulties associated with solving MOPs. Lower vehicle costs may not result in the desired paint job and stereo system. The desired ‘miles per gallon’ may not be achieved by any vehicle on the market. Constructing a mathematical model representing this situation may not be easy. For example, how does one quantify “fun” and “premium?” Perhaps this MOP is not as simple as it appears?

1.2 Research Definition

Humanity has long been solving MOPs. As both human society and its technologies progressed and became more complex, one can easily argue that real-world MOPs also became correspondingly “harder.” For instance, Darius assumed control of the Persian Empire around 500 B.C. Soon after he led his army and navy on a campaign to secure the Empire’s eastern and western frontiers. [119:pg. 16]. His 70,000 man army consisted of foot soldiers, archers, and cavalrymen, and his navy had 200-300 ships. It is obvious his campaign’s planning was rife with conflicting objectives.

Imagine his war council’s conversations. “Which frontier should be attacked first?” “Can the navy be used here instead of the army?” “Is the army or navy more effective in coastal attack?” “Since most of the foot soldiers are needed here, can this mission be accomplished by a force composed primarily of archers?” “The cavalry is the force of choice here, but isn’t their cost more expensive (in terms of logistics)?” One easily sees the conflicts and tradeoffs which often occur when attempting to simultaneously satisfy multiple conflicting and/or complementary objectives.

Fast forward 2500 years to 1991. Compare Darius’ situation to that of General Norman Schwarzkopf’s as Commander-in-Chief, Central Command, during Operation Desert Storm. As military leader of the coalition attacking Iraq (with almost 600,000 US personnel alone [15:pg. 492]), Schwarzkopf had several military force options to consider. The US could supply troops from its own Army, Air Force, Navy, and Marines. Other coalition members brought similar military forces to the battle, some with unique capabilities.

A multitude of weaponry was available. Army troops were armed with pistols, machine guns, artillery, and tanks. Coalition navies used destroyers, submarines, and carrier-based aviation. The Air Force brought precision-guided munitions and conventionally-armed cruise missiles to the fray. One easily sees this campaign's planning was a much more difficult problem than Darius faced. Many more resources were available for use, each with attendant benefits and drawbacks depending on their particular application. A mix of social, political, and military objectives was still considered here, but instead of satisfying these goals in view of a single country's interests, a coalition of countries was involved. Additionally, many coalition nations' political and military leadership were no longer embodied in the same individual. Battle maps showing coalition forces' attacks on Iraq give some appreciation for how some of the many complex military objectives were (partially) satisfied [15:pp. 515-521].

Just as instantiated MOP complexity has increased through history, so has performance and complexity of associated solution methods. Consider the post-World War II period. Here, the combination of "state of the art" algorithmic advances (e.g., linear programming, queuing theory) and the advent of electronic computation contributed to the solution of larger and more complex optimization problems. [150:pp. 3-5]. Thus, although one can easily imagine Darius and his generals clustered around an ancient "white board" manually employing a primitive version of these algorithms, we now focus on computational implementations of current "state of the art" algorithms.

Several algorithmic MOP solution approaches can be identified including enumerative, deterministic, and stochastic schemes [126]. Because many MOPs are high-dimensional, discontinuous, multimodal, and/or Nondeterministic Polynomial (NP)-Complete, stochastic methods often give better performance. This research focuses on a class of stochastic computational methods for solving real-world scientific and engineering MOPs called Evolutionary Algorithms (EAs), specifically centering on what we term Multiobjective Evolutionary Algorithms (MOEAs).

Webster's dictionary defines the term *effective* as the production of or the power to produce an acceptable result; *efficient* is defined as acting in such a way as to avoid resource loss or waste in functioning [339]. The term *engineering* is then defined as planning with more or less subtle skill. By addressing relevant issues, this research shows "well-engineered" MOEAs have the potential to solve some real-world MOPs both effectively and efficiently.

1.3 Research Goals and Objectives

This research focuses on the foundations of MOEA application to scientific and engineering MOPs. A myriad of related issues is involved in this effort, but broadly speaking, this investigation attempts to achieve three major goals: MOEA classifications, analyses, and innovations.

1.3.1 Goal 1: MOEA Classifications. Classifying any related set of items may not be a simple task as classification characteristics may be conflicting, complementary, subjective, and so forth. As both the MOP and MOEA domains are quite complex, these may be the reasons why few researchers have attempted to organize the MOEA literature into a coherent whole. This research effort attempts to place a cohesive "wrapper" around both the MOEA literature and the major factors to consider when solving MOPs with MOEAs. Research objectives supporting this goal are listed in Table 1.1.

Table 1.1. MOEA Classifications' Objectives

Goal: MOEA Classifications
Objectives:
Develop and refine a sound, extensible basis for MOEA classification
Classify known implementations
Organize key problem/algorithm domain components of classified MOEAs
Organize MOEA test functions used in the literature

1.3.2 Goal 2: MOEA Analyses. The literature has no self-contained introductory document explaining relevant issues to consider when solving MOPs with MOEAs. In addition, little literature currently exists regarding MOEA theory. As any effective and efficient MOP solution algorithm *must* incorporate problem domain knowledge and appropriate heuristics [218, 346], this study attempts to extend current MOEA theory by analyzing key problem and algorithm domain characteristics. This allows for the design and application of "well-engineered" MOEAs. Research objectives supporting this goal are listed in Table 1.2.

Table 1.2. MOEA Analyses' Objectives

Goal: MOEA Analyses
Objectives:
Critically consider current MOEA literature based upon classification effort
Analyze the MOP/MOEA domain integration process
– Identify and analyze major MOP domain characteristics
– Identify and analyze key MOEA components used in solving MOPs
Identify existing "well-engineered" MOEAs
Identify, analyze, and classify metrics for use in comparing MOP solutions

1.3.3 Goal 3: MOEA Innovations. This dissertation attempts to extend MOEA "state of the art." Its classification and analysis identifies several shortcomings in the field; the theoretical and practical innovations it offers are meant to expand the field's knowledge and to stimulate critical thinking among other researchers. Research objectives supporting this goal are listed in Table 1.3.

Table 1.3. MOEA Innovations' Objectives

Goal: MOEA Innovations
Objectives:
Define the presence and role of Building Blocks (BBs) in MOP solutions
Engineer an MOEA to explicitly manipulate BBs in solving MOPs
– Incorporate relevant analytical results in designing a BB-based MOEA
– Determine performance of the new MOEA
– Determine benefits of a parallel implementation
Substantiate and propose an MOEA test function suite
Substantiate and execute MOEA experiments
– Use developed metrics, test functions, and suitable MOEAs
Relate experimental results to MOEA application in real-world MOPs

1.4 Research Approach and Scope

This research adopts a methodical approach in accomplishing the previously defined goals and objectives. It performs an in-depth investigation into both the problem (MOP) and algorithm (MOEA) domains via an extensive literature review. Insight gained through this review is then used in engineering an innovative EA, and in designing a proposed

MOEA test suite and performance metrics. Finally, appropriate MOEA experiments are designed and executed using the developed metrics, their results analyzed, and conclusions presented.

This research's goals and objectives (see Section 1.3) clearly define its focus. However, some general comments further clarify this document's scope. First, this research assumes the reader has a basic understanding of EAs, general mathematics, and computer engineering. Second, any software developed supporting this research may not completely follow accepted software engineering practices since suitable existing software may be modified when possible. We also employ rapid prototyping, and intend to make any software implementation largely platform and operating system independent. Last, although this research focuses and reports on primarily theoretical concepts, real-world MOP issues are not ignored.

1.5 Document Organization

The remainder of this document is organized as follows. Chapter II gives an overview of MOPs, general optimization techniques, EAs, and MOEAs; it also offers new theorems and definitions. Chapter III presents in-depth analyses of MOEA "state of the art," discussing practical and theoretical algorithm design considerations. Chapter IV defines BB concepts and their use in EAs, then presents a new MOEA (called the Multiobjective messy Genetic Algorithm (MOMGA)) qualitatively different than any existing implementation. The MOMGA explicitly manipulates BBs in its search for MOP solutions. Relevant algorithmic test suite issues are discussed in Chapter V, which then substantiates/proposes MOPs for inclusion in an MOEA test suite. Chapters VI and VII provide both the experimental methodology for and analysis of experiments performed supporting this research. Chapter VIII then concludes the document's body by recapping its major contributions.

Several appendices providing background and reference information are included. Appendix A contains the extensive cataloged MOEA literature review used as the basis for much of Chapter III's presented analysis. Appendix B contains numeric MOP test functions used in the MOEA literature; Appendices C and D then present corresponding graphs for these functions showing each MOP's salient characteristics.

II. Multiobjective Optimization and Evolutionary Algorithms

In relieving the brain of all unnecessary work, a good notation sets it free to concentrate on more advanced problems, and, in effect, increases the mental power of the race.
Alfred North Whitehead

2.1 Introduction

This chapter provides an overview of the problem and algorithm domains considered within this research. Neither is straightforward. Thus, we present key concepts defining and bounding both the problem class (MOPs) and algorithms selected to solve them (MOEAs). Clearly comprehending this basic information makes it easier to grasp more detailed concepts presented later.

Section 2.2 defines the MOP domain and offers new related theorems and definitions. Section 2.3 presents an overview of general search and optimization techniques, giving a framework within which to place the algorithms focused on by this research. Key elements of these EAs/MOEAs are given in Sections 2.4 and 2.5. Finally, an MOEA literature review and technique classification scheme are described in Section 2.6.

2.2 MOP Definition and Overview

Global optimization is the process of finding the global minimum[1] within some search space. The single-objective global optimization problem is formally summarized in the following definition [17:pg. 35]:

Definition 1 (Global Minimum): *Given a function $f : \Omega \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$, $\Omega \neq \emptyset$, for $\vec{x} \in \Omega$ the value $f^* \triangleq f(\vec{x}^*) > -\infty$ is called a global minimum if and only if*

$$\forall \vec{x} \in \Omega : \quad f(\vec{x}^*) \leq f(\vec{x}) \; . \tag{2.1}$$

[1] Or maximum, since $\min\{F(x)\} = -\max\{-F(x)\}$.

Then, $\vec{x}^*$ is the global minimum solution(s), f is the objective function, and the set Ω is the feasible region. The problem of determining the global minimum solution(s) is called the global optimization problem. $\square$

Although single-objective optimization problems may have a unique optimal solution, MOPs (as a rule) present a possibly uncountable *set* of solutions, which when evaluated, produce vectors whose components represent trade-offs in objective space. A decision maker then implicitly chooses an acceptable solution (or solutions) by selecting one or more of these vectors. MOPs are mathematically defined as follows:

Definition 2 (General MOP): *In general, an MOP minimizes $F(\vec{x}) = (f_1(\vec{x}), \ldots, f_k(\vec{x}))$ subject to $g_i(\vec{x}) \leq 0,\ i = 1, \ldots, m,\ \vec{x} \in \Omega$. An MOP solution minimizes the components of a vector $F(\vec{x})$ where $\vec{x}$ is an n-dimensional decision variable vector ($\vec{x} = x_1, \ldots, x_n$) from some universe Ω.* $\square$

An MOP thus consists of n decision variables, m constraints, and k objectives, of which any or all of the objective functions may be linear or nonlinear [158]. The MOP's evaluation function, $F : \Omega \longrightarrow \Lambda$, maps decision variables ($\vec{x} = x_1, \ldots, x_n$) to vectors ($\vec{y} = a_1, \ldots, a_k$). This situation is represented in Figure 2.1 for the case $n = 2$, $m = 0$, and $k = 3$. This mapping may or may not be onto some region of objective function space, dependent upon the functions and constraints composing the particular MOP. Furthermore, all problems discussed in this dissertation are assumed to be minimization problems unless otherwise specified, and to be primitive recursive (i.e., computable) [211].

MOPs are characterized by distinct measures of performance (the objectives) which may be (in)dependent and/or non-commensurable. For example, a radio antenna's transmit power and direct monetary cost may have little dependence on each other (past a certain point); they are also measured in different units (watts vs. dollars). The multiple objectives being optimized almost always conflict, placing a partial, rather than total, ordering on the search space. In fact, finding the global optimum of a general MOP is NP-Complete [17:pg. 56]. "Perfect" MOP solutions, where all decision variables satisfy associated constraints and the objective function attains a global minimum, may not

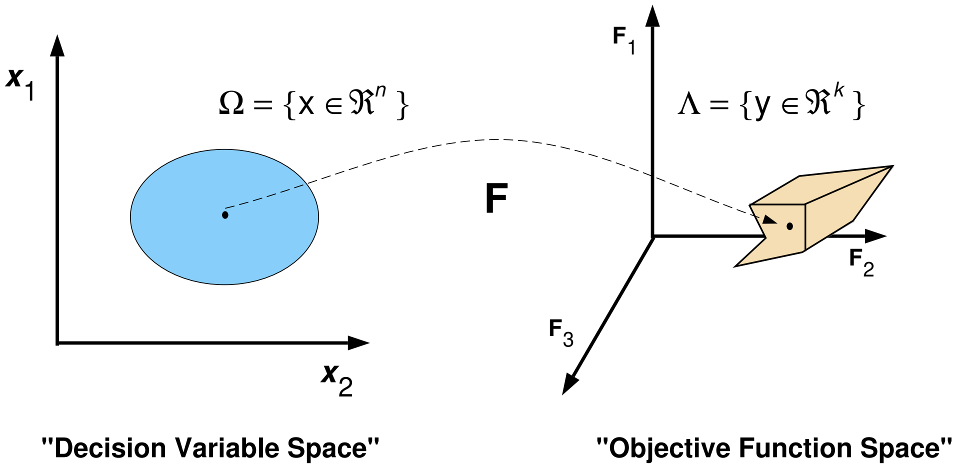


Figure 2.1. MOP Evaluation Mapping

even exist. In addition, as Horn and others do [152], we use the terms *objective*, *criteria*, and *attribute* interchangeably to represent an MOP's goals or objectives (i.e., distinct mathematical functions) to be achieved, even though they are subtly distinguished in the literature. We also use the terms *objective space* or *objective function space* to denote the coordinate space within which vectors resulting from evaluating an MOP are plotted.

Because of these characteristics (multiple objectives and constraints), MOPs may require specialized optimization techniques. Regardless of implemented technique, a key concept in determining a set of MOP solutions is that of *Pareto Optimality*.

2.2.1 Pareto Concepts. Although Pareto optimality, and its related concepts and terminology are frequently invoked, MOEA researchers often erroneously use them in the literature. To ensure understanding and consistency we define Pareto Dominance, Pareto Optimality, the Pareto Optimal Set, and the Pareto Front. An associated symbolic notation is introduced later in Section 2.5.1. Using the MOP notation presented in Definition 2 we mathematically define these key Pareto concepts [27] as follows:

Definition 3 (Pareto Dominance): *A vector* $\vec{u} = (u_1, \ldots, u_k)$ *is said to dominate* $\vec{v} = (v_1, \ldots, v_k)$ *(denoted by* $\vec{u} \preceq \vec{v}$*) if and only if u is partially less than v, i.e.,* $\forall i \in \{1, \ldots, k\},\ u_i \leq v_i \wedge \exists i \in \{1, \ldots, k\} : u_i < v_i$. $\square$

Definition 4 (Pareto Optimality): *A solution $x \in \Omega$ is said to be Pareto optimal with respect to Ω if and only if there is no $x' \in \Omega$ for which $\vec{v} = F(x') = (f_1(x'), \ldots, f_k(x'))$ dominates $\vec{u} = F(x) = (f_1(x), \ldots, f_k(x))$. The phrase "Pareto optimal" is taken to mean with respect to the entire decision variable space unless otherwise specified.* □

Definition 5 (Pareto Optimal Set): *For a given MOP $F(x)$, the Pareto optimal set ($\mathcal{P}^*$) is defined as:*

$$\mathcal{P}^* := \{x \in \Omega \mid \neg\exists\, x' \in \Omega \;\; F(x') \preceq F(x)\}. \tag{2.2}$$

□

Definition 6 (Pareto Front): *For a given MOP $F(x)$ and Pareto optimal set $\mathcal{P}^*$, the Pareto front ($\mathcal{PF}^*$) is defined as:*

$$\mathcal{PF}^* := \{\vec{u} = F(x) = (f_1(x), \ldots, f_k(x)) \mid x \in \mathcal{P}^*\}. \tag{2.3}$$

□

Pareto optimal solutions are also termed *non-inferior*, *admissible*, or *efficient* solutions [152]; their corresponding vectors are termed *nondominated*. These solutions may have no clearly apparent relationship besides their membership in the Pareto optimal set. This is the set of all solutions whose corresponding vectors are nondominated with respect to all other comparison vectors; we stress here that Pareto optimal solutions are classified as such based on their evaluated functional values. When plotted in objective space, the nondominated vectors are collectively known as the Pareto front. Again, $\mathcal{P}^*$ is a subset of some solution set. Its evaluated objective vectors form $\mathcal{PF}^*$, of which each is nondominated with respect to all objective vectors produced by evaluating every possible solution in Ω.

As an example of these Pareto concepts we present the one-variable, two-objective problem denoted as $F1$. This is the same problem used by Vincent and Grantham, Schaffer,

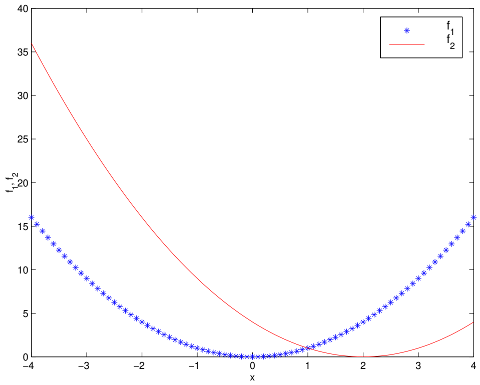


Figure 2.2. f_1 and f_2 Values vs x

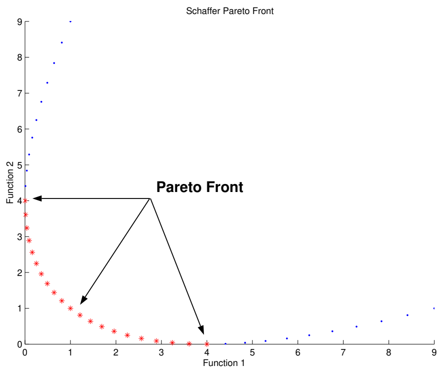


Figure 2.3. F_1's Pareto Front

and Srinivas and Deb for identical purposes [306]. This MOP is defined as:

$F1 = (f_1(x), f_2(x))$, where

$$\begin{aligned} f_1 &= x^2 \,, \\ f_2 &= (x-2)^2 \,. \end{aligned} \tag{2.4}$$

Figure 2.2 implies that the Pareto optimal set is $\{x \mid x < 0 \text{ or } x > 2\}$. The solution $x = 0$ is optimal with respect to f_1 but not f_2; the solution $x = 2$ is optimal with respect to f_2 but not f_1. Any solution $\{x \mid x \notin 0 \leq x \leq 2\}$ is not a member of the Pareto optimal set because it is not better than a solution in the set with respect to either objective. Rudolph [276] has also shown that given:

$F = (f_1(x), f_2(x))$, where

$$\begin{aligned} f_1 &= ||x||^2 \,, \\ f_2 &= ||x-z||^2 \,, \text{with } 0 \neq z \in R \,, \end{aligned} \tag{2.5}$$

the Pareto optimal set for this general MOP is:

$$\mathcal{P}^* = \{x \in R \mid x = rz,\ r \in [0,1]\} \,. \tag{2.6}$$

We point out a significant difference between Figures 2.2 and 2.3. Figure 2.2 plots the values of functions f_1 and f_2 for different values of the independent variable. However, Figure 2.3 represents the values of function f_1 plotted against those of function f_2 *for the same value of the independent variable*. In other words, Figure 2.3 is a graph in objective space displaying this MOP's vectors *as points*. The nondominated vectors (graphed as asterisks) represent $F1$'s Pareto front.

Note that the Decision Maker (DM) is often selecting solutions via choice of acceptable objective performance, represented by the Pareto front. Choosing an MOP solution that optimizes only one objective may well ignore solutions, which from an overall standpoint, are "better." The Pareto optimal set contains those better solutions. Identifying a set of Pareto optimal solutions is thus key for a DM's selection of a "compromise" solution satisfying the objectives as "best" possible. Of course, the accuracy of the decision maker's view depends on both the *true* Pareto optimal set and the set presented *as* Pareto optimal.

We note here that derived solutions of real-world MOPs often offer only a finite number of points which may or may not be truly Pareto optimal. Any time the real-(continuous) world is modeled (e.g., via objective functions) upon a computer (a discrete machine), there is a fidelity loss between reality's uncountable infinity and the implemented finite, discretized model. Complex MOPs do not generally lend themselves to analytical determination of the actual Pareto front, thus making even a computational approximation of an MOP's global optimum difficult.

2.2.2 Pareto-Related Contributions. We have developed new Pareto-based theorems and definitions to support research objectives and other theoretical results. As many MOEAs assume each generational population contains Pareto optimal solutions (with respect to that population), Theorem 1 substantiates this assumption. As the MOEA literature offers little guidance concerning possible Pareto front cardinality and dimensionality, Theorems 2 and 3 provides an upper bound. Thus, these Pareto contributions further bound both problem and algorithm domains. They are presented here for coherence.

2.2.2.1 Pareto Optimal Set Minimal Cardinality. Because of the manner in which Pareto optimality is defined, any non-empty finite solution set contains at least one Pareto optimal solution (with respect to that set). As this may be non-intuitive, and because it is assumed in many MOEA implementations, we present the following theorem for the general case.

Theorem 1: Given an MOP with feasible region Ω and any non-empty finite solution set $\omega \subseteq \Omega$, there exists at least one solution $\vec{x} \in \omega$ that is Pareto optimal with respect to ω. $\square$

Proof: Label the k-dimensional objective vectors resulting from evaluating each $\vec{x}_i \in \omega$ in non-decreasing, lexicographic order as $v_1, v_2, \ldots, v_n$ with $v_i = (v_{i,1}, v_{i,2}, \ldots, v_{i,k})$. If all v_i are equal then v_1 is nondominated. Otherwise, there exists a smallest $j \in \{1, \ldots, k\}$ such that for some $i \in \{1, \ldots, n-1\}$, $v_{1,j} = v_{2,j} = \ldots = v_{i,j} < v_{i+1,j} \leq v_{i+2,j} \leq \ldots \leq v_{n,j}$. This shows that $v_{i+1}, v_{i+2}, \ldots, v_n$ do not dominate v_1.

If $i = 1$ then we have shown v_1 is nondominated. On the other hand, if $i \neq 1$ and $j = k$ we have shown $v_1 = v_2 = \ldots = v_i$ and v_1 is again nondominated. Otherwise, there exists a smallest $j' \in \{j+1, \ldots, k\}$ such that for some $i' \in \{1, \ldots, i-1\}$, $v_{1,j'} = v_{2,j'} = \ldots = v_{i',j'} < v_{i'+1,j'} \leq v_{i'+2,j'} \leq \ldots \leq v_{i,j'}$. If either $i' = 1$, or $i' \neq 1$ and $j' = k$, then we have again shown v_1 is nondominated. Otherwise we continue this process. Because k is finite we eventually show v_1 is nondominated and therefore there is at least one solution that is Pareto optimal with respect to ω. *Q.E.D.*

2.2.2.2 Pareto Front Structure. Theoretical bounds are useful in defining a given problem domain. We now present a corollary and theorems defining the structural bounds any Pareto front may attain. Corollary 1 provides a lower bound for the cardinality of the Pareto front.

Corollary 1: Given an MOP with feasible region Ω and any non-empty finite solution set $\omega \subseteq \Omega$, its Pareto front $\mathcal{PF}^*$ is a set containing at least one vector. This result follows directly from Theorem 1. $\square$

Theorem 2 provides an upper bound on the cardinality of the Pareto front for MOPs with Euclidean objective spaces (spaces containing all n-tuples of real numbers, $(x_1, x_2, \dots, x_n)$, denoted by $\mathbb{R}^n$). This includes all MOPs of interest in this research.

Theorem 2: The Pareto front of any MOP is composed of at most an uncountably infinite number of vectors. $\square$

Proof: The Pareto front's cardinality is bounded above by the cardinality of the objective space. *Q.E.D.*

We use the following definition in bounding the Pareto front's dimensionality [6:pg. 174]:

Definition 7 (Box-Counting Dimension): *A bounded set S in $\mathbb{R}^k$ has box-counting dimension*

$$boxdim(S) = \lim_{\epsilon \to 0} \frac{\ln N(\epsilon)}{\ln(\frac{1}{\epsilon})} , \tag{2.7}$$

where the limit exists and where $N(\epsilon)$ is the number of boxes that intersect S. $\square$

Theorem 3: For a given MOP $F(x)$ and Pareto optimal set $\mathcal{P}^*$, if the Pareto front $\mathcal{PF}^*$ is bounded, then it is a set with box-counting dimension no greater than $(k-1)$. $\square$

Proof: Without loss of generality assume $\mathcal{PF}^*$ is a bounded set in $[0,1]^k$. Take S to be the closure of $\mathcal{PF}^*$. Because $[0,1]^k$ is closed, S is a bounded set in $[0,1]^k$. Let $[0,1]^k$ be partitioned by a grid of k-dimensional boxes of side-length ϵ, where the boxes' sides are parallel to the objective axes. For each $r \in R \triangleq \{0, \epsilon, 2\epsilon, \ldots, \lfloor \frac{1}{\epsilon} \rfloor \epsilon\}^{k-1}$ define $R_r = [r_1, r_1 + \epsilon] \times [r_2, r_2 + \epsilon] \times \cdots \times [r_{k-1}, r_{k-1} + \epsilon] \times [0, 1]$. If $S \cap R_r \neq \emptyset$, define p_r to be the point that minimizes f_k over R_r and B_r to be any box that includes p_r. Also define $S_\epsilon = \{p_r\}$ and $B_\epsilon = \cup_r B_r$. Then B_ϵ covers S_ϵ. Because S is closed $\lim_{\epsilon \to 0} S_\epsilon = S$, and $B \triangleq \lim_{\epsilon \to 0} B_\epsilon$ covers S. Because $\mathcal{PF}^* \subseteq \mathcal{S}$, B also covers $\mathcal{PF}^*$. Hence, $N(\epsilon) = \mid R \mid = \lceil \frac{1}{\epsilon} \rceil^{k-1}$, and the box-counting dimension of $\mathcal{PF}^*$ is

$$\begin{aligned}
\lim_{\epsilon \to 0} \frac{\ln(\lceil \frac{1}{\epsilon} \rceil^{k-1})}{\ln(\frac{1}{\epsilon})} &\leq \lim_{\epsilon \to 0} \frac{\ln((\frac{2}{\epsilon})^{k-1})}{\ln(\frac{1}{\epsilon})} \\
&= \lim_{\epsilon \to 0} \frac{(k-1)[\ln 2 + \ln(\frac{1}{\epsilon})]}{\ln(\frac{1}{\epsilon})} \\
&= \lim_{\epsilon \to 0} [\frac{(k-1)\ln 2}{\ln(\frac{1}{\epsilon})} + (k-1)] \\
&= k - 1 \qquad (2.8)
\end{aligned}$$

Q.E.D.

In practice, the Pareto front is a collection of $(k-1)$ or lower dimensional surfaces we term Pareto surfaces. The special case where $k = 2$ results in surfaces we term Pareto curves. Horn [154] and Thomas [318] state that a k-objective MOP's Pareto front *is* a $k-1$ dimensional surface. We have just shown this is incorrect; the front is *at most* $(k-1)$ dimensional surface. Although asymptotic bounds are useful, researchers must also account for the Pareto front's possible shape within those bounds. Theorem 3 then implies that any proposed MOEA benchmark test function suite should contain MOPs with Pareto fronts composed of Pareto curve(s), Pareto surface(s), or some combination of the two.

2.2.2.3 MOP Global Optimum. Defining an MOP's global optimum is not a trivial task as the "best" compromise solution may vary among DMs due to individual beliefs and biases. Solutions may also have some temporal dependence, e.g., acceptable resource expenditures may vary from month to month. Thus, there is no universally accepted definition for the MOP global optimization problem. However, we define an MOP's global optimum to substantiate later algorithmic engineering decisions.

Pareto optimal solutions are those which when evaluated, produce vectors whose performance in one dimension *cannot* be improved without adversely affecting another. The Pareto front $\mathcal{PF}^*$ determined by evaluating $\mathcal{P}^*$ is fixed by the defined MOP and does not change. Thus, $\mathcal{P}^*$ represents the "best" solutions available and allows the definition of an MOP's global optimum.

Definition 8 (MOP Global Minimum): *Given a function $F : \Omega \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^k$, $\Omega \neq \emptyset$, $k \geq 2$, for $\vec{x} \in \Omega$ the set $\mathcal{PF}^* \triangleq F(\vec{x}_i^*) > (-\infty, \ldots, -\infty)$ is called the global minimum if and only if*

$$\forall \vec{x} \in \Omega : \quad F(\vec{x}_i^*) \preceq F(\vec{x}) \; . \tag{2.9}$$

Then, $\vec{x}_i^*$, $i = 1, \ldots, n$ is the global minimum solution set (i.e., $\mathcal{P}^*$), F is the multiple objective function, and the set Ω is the feasible region. The problem of determining the global minimum solution set is called the MOP global optimization problem. □

2.3 General Optimization Algorithm Overview

We classify general search and optimization techniques into three categories: enumerative, deterministic, and stochastic (random). Although an enumerative search is deterministic we make a distinction here as it employs no heuristics. Figure 2.4 shows common examples of each type.

Enumerative schemes are perhaps the simplest search strategy. Within some defined finite search space each possible solution is evaluated. However, it is easily seen this technique is inefficient or even infeasible as search spaces become large. As many real-world problems are computationally intensive, some means of limiting the search space must be implemented to find "acceptable" solutions in "acceptable" time. Deterministic algorithms attempt this by incorporating problem domain knowledge. Many of these are considered graph/tree search algorithms and are described as such here.

Greedy algorithms make locally optimal choices, assuming optimal sub-solutions are *always* part of the globally optimal solution [42, 157]. Thus, these algorithms fail unless

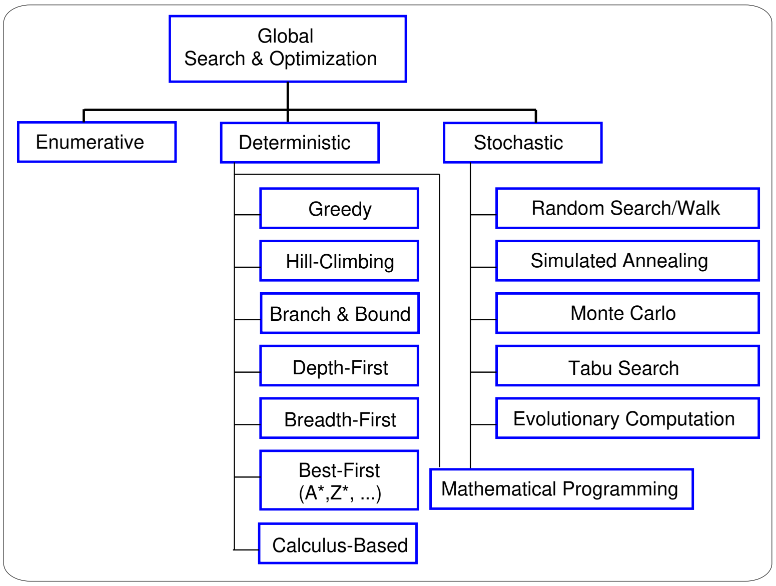


Figure 2.4. Global Optimization Approaches

that is the case. Hill-climbing algorithms search in the direction of steepest ascent from the current position. These algorithms work best on unimodal functions, but the presence of local optima, plateaus, or ridges in the fitness (search) landscape reduce algorithm effectiveness [277]. Greedy and hill-climbing strategies are *irrevocable*. They repeatedly expand a node, examine all possible successors (then expanding the "most promising" node), and keep no record of past expanded nodes [252].

Branch and bound search techniques need problem specific heuristics/decision algorithms to limit the search space [120, 252]. They compute some bound at a given node which determines whether the node is "promising;" several nodes' bounds are then compared and the algorithm branches to the "most promising" node [233]. Basic depth-first search is *blind* or *uninformed* in that the search order is independent of solution location (except for search termination). It expands a node, generates all successors, expands a successor, and so forth. If the search is blocked (e.g., it reaches a tree's bottom level) it resumes from the deepest node left behind [252]. *Backtracking* is a depth-first search variant which "backtracks" to a node's parent if the node is determined "unpromising" [233]. Breadth-first search is also uninformed. It differs from depth-first search in its actions after node expansion, where it progressively explores the graph one *layer* at a time [252]. Best-first search uses heuristic information to place numerical values on a node's "promise"; the node with highest promise is examined first [252]. A*, Z*, and others are popular best-first search variants selecting a node to expand based both on "promise" and the overall cost to arrive at that node. Finally, calculus-based search methods at a minimum require continuity in some variable domain for an optimal value to be found [13].

Greedy and hill-climbing algorithms, branch and bound tree/graph search techniques, depth- and breadth-first search, best-first search, and calculus-based methods are all deterministic methods successfully used in solving a wide variety of problems [42, 126, 233]. However, many MOPs are high-dimensional, discontinuous, multimodal, and/or NP-Complete. Deterministic methods are often ineffective when applied to NP-Complete or other high-dimensional problems because they are handicapped by their requirement for problem domain knowledge (heuristics) to direct or limit search [106, 120, 126] in these exceptionally large search spaces. Problems exhibiting one or more of these above characteristics are termed *irregular* [190].

Because many real-world scientific and engineering MOPs are irregular, enumerative and deterministic search techniques are then unsuitable. Stochastic search and optimization approaches such as Simulated Annealing (SA), Monte Carlo, Tabu search, and Evolutionary Computation (EC) techniques were developed as alternative approaches for solving these irregular problems [126, 218]. Stochastic methods require a function assigning fitness values to possible (or partial) solutions, and an encode/decode (mapping) mechanism between the problem and algorithm domains. Although some are shown to "eventually" find an optimum most cannot guarantee *the* optimal solution. They in general provide good solutions to a wide range of optimization problems which traditional deterministic search methods find difficult [126, 157].

A random search is the simplest stochastic search strategy, as it simply evaluates a given number of randomly selected solutions. A random walk is very similar, except that the next solution evaluated is randomly selected using the last evaluated solution as a starting point [333]. Like enumeration, though, these strategies are not efficient for many MOPs because of their failure to incorporate problem domain knowledge. Random searches can generally expect to do no better than enumerative ones [126:pg. 5].

SA is an algorithm explicitly modeled on an *annealing* analogy, where, for example, a liquid is heated and then gradually cooled until it freezes. Where hill-climbing chooses the *best* move from some node SA picks a *random* one. If the move improves the current optimum it is always executed, else it is made with some probability $p < 1$. This probability exponentially decreases either by time or with the amount by which the current optimum is worsened [277]. If water's temperature is lowered slowly enough it attains a lowest-energy configuration; the analogy for SA is that if the "move" probability decreases slowly enough the global optimum is found.

In general, *Monte Carlo* methods involve simulations dealing with stochastic events; they employ a pure random search where any selected trial solution is fully independent of any previous choice and its outcome [295]. The current "best" solution and associated decision variables are stored as a comparator. *Tabu search* is a meta-strategy developed to avoid getting "stuck" on local optima. It keeps a record of both visited solutions and the "paths" which reached them in different "memories." This information restricts the choice of solutions to evaluate next. Tabu search is often integrated with other optimization methods [295].

EC is a generic term for several stochastic search methods which computationally simulate the natural evolutionary process. As a recognized research field EC is young, although its associated techniques have existed for about thirty years. EC embodies the techniques of Genetic Algorithms (GAs), *Evolutionsstrategies*, or Evolution Strategies (ESs), and Evolutionary Programming (EP), collectively known as EAs. These techniques are loosely based on natural evolution and the Darwinian concept of "Survival of the Fittest" [126]. Common between them are the reproduction, random variation, competition, and selection of contending individuals within some population [104]. In general, an EA consists of a *population* of encoded solutions (*individuals*) manipulated by a set of *operators* and evaluated by some *fitness function*.

Each solution's associated *fitness* determines which survive into the next *generation*. Although sometimes considered equivalent, the terms *EA* and *EC* are used separately in this document to preserve the distinction between EAs and other EC techniques (e.g., Genetic Programming (GP) and learning classifier systems)[2].

MOP complexity and the shortcomings of deterministic search methods also drove creation of several optimization techniques by the Operations Research (OR) community. These methods (whether linear or non-linear, deterministic or stochastic) can be grouped under the rubric *mathematical programming*. These methods treat constraints as the main problem aspect [295]. *Linear programming* is designed to solve problems in which the objective function and all constraint relations are linear [150]. Conversely, *nonlinear programming* techniques solve *some* MOPs not meeting those restrictions but require convex constraint functions [295]. We note here that many problem domain assumptions must be satisfied when using linear programming, and that many real-world scientific and engineering problems may only be modeled by non-linear functions [150:pp. 138,574]. Finally, *stochastic programming* is used when random-valued parameters and objective functions subject to statistical perturbations are part of the problem formulation. Depending on the type of variables used in the problem, several variants of these methods exist (i.e., discrete, integer, binary, and mixed-integer programming) [295].

2.4 EA Overview

The following presentation defines basic EA structural terms and concepts;[3] the described terms' "meanings" are normally analogous to their genetic counterparts. A *structure* or *individual* is an encoded solution to some problem. Typically, an individual is represented as a string (or string of strings) corresponding to a biological *genotype*. This genotype defines an individual organism when it is expressed (decoded) into a *phenotype*. A genotype is composed of one or more *chromosomes*, where each chromosome is composed of separate *genes* which take on certain values (*alleles*) from some genetic alphabet. A *locus* identifies a gene's position within the chromosome. Thus, each individual decodes into a set of parameters used as input to the function under consideration. Finally, a given set of

[2]Although GP and learning classifier systems may be classified as EA techniques, we and others consider them conceptually different approaches to EC [180].

[3]There is no shortage of introductory EA texts. The general reader is referred to Goldberg [126], Michalewicz [218], or Mitchell [223]; a more technical presentation is given by Bäck [17].

chromosomes is termed a *population.* These concepts are pictured in Figure 2.5 (for both binary and real-valued chromosomes) and in Figure 2.6.

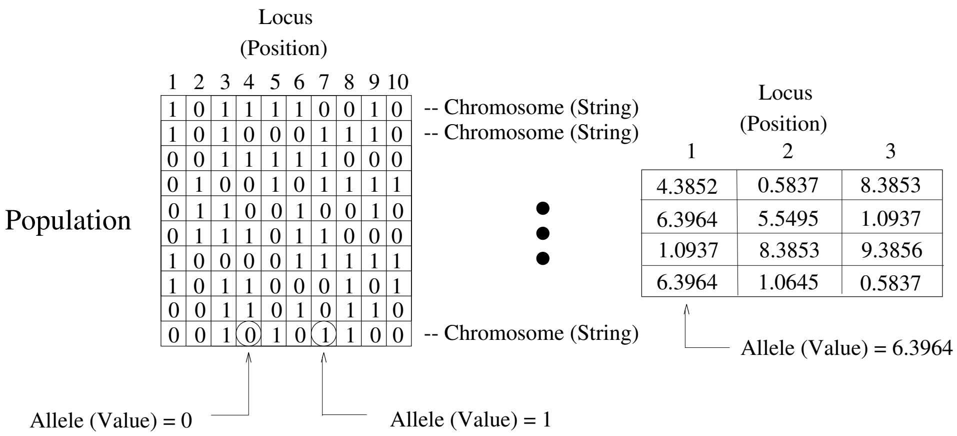


Figure 2.5. Generalized EA Data Structure and Terminology

Just as in nature, Evolutionary Operators (EVOPs) operate on an EA's population attempting to generate solutions with higher and higher fitness. The three major EVOPs associated with EAs are *mutation*, *recombination*, and *selection.* Illustrating this, Figure 2.7 shows *bitwise mutation* on an encoded string where a '1' is changed to a '0', or vice versa. Figure 2.8 shows *single-point crossover* (a form of recombination) operating on two parent binary strings; each parent is cut and recombined with a piece of the other. Above-average individuals in the population are *selected* (reproduced) to become members of the next generation more often than below-average individuals. The selection EVOP effectively gives strings with higher fitness a higher probability of contributing one or more children in the succeeding generation. The *Schema Theorem* describes this process and is discussed in Section 4.2. Figure 2.9 shows the operation of the common *roulette-wheel* selection (a *fitness proportional* selection operator) on two different populations of four strings each. Each string in the population is assigned a portion of the wheel proportional to the ratio of its fitness and the population's average fitness.

Real-valued chromosomes also undergo these same EVOPs although implemented differently. All EAs use some subset or variation of these EVOPs. Many variations on the basic operators exist; these are dependent upon problem domain constraints affecting chromosome structure and alleles [17].

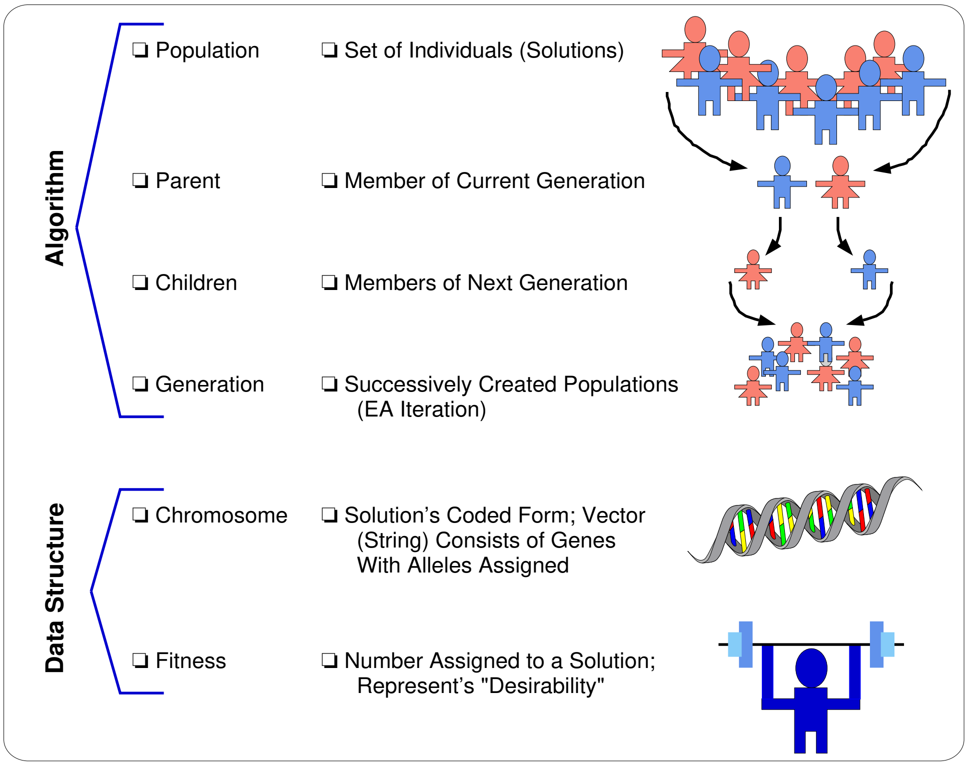


Figure 2.6. Key EA Components

An EA requires both an *objective* and *fitness* function, which are fundamentally different. The objective function defines the EA's optimality condition (and is a feature of the problem domain) while the fitness function (in the algorithm domain) measures how "well" a particular solution satisfies that condition and assigns a corresponding real-value to that solution. However, these functions are in principle identical [17:pg. 68] (e.g., numerical optimization problems).

Many other selection techniques are implemented by EAs, e.g., tournament and ranking [17]. Tournament selection operates by randomly choosing some number q individuals from the generational population and selecting the "best" to survive into the next generation. Binary tournaments ($q = 2$) are probably the most common. Ranking assigns selection probabilities solely on an individual's rank, ignoring absolute fitness values. Two other selection techniques we note in detail are the $(\mu + \lambda)$ and (μ, λ) selection strategies, where μ represents the number of parent solutions and λ the number of children. The

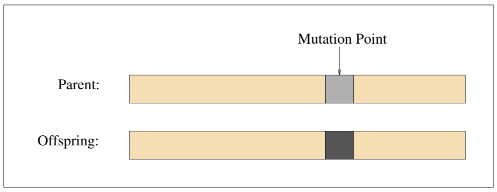


Figure 2.7. Bitwise Mutation

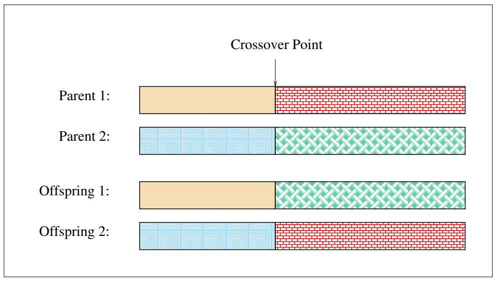
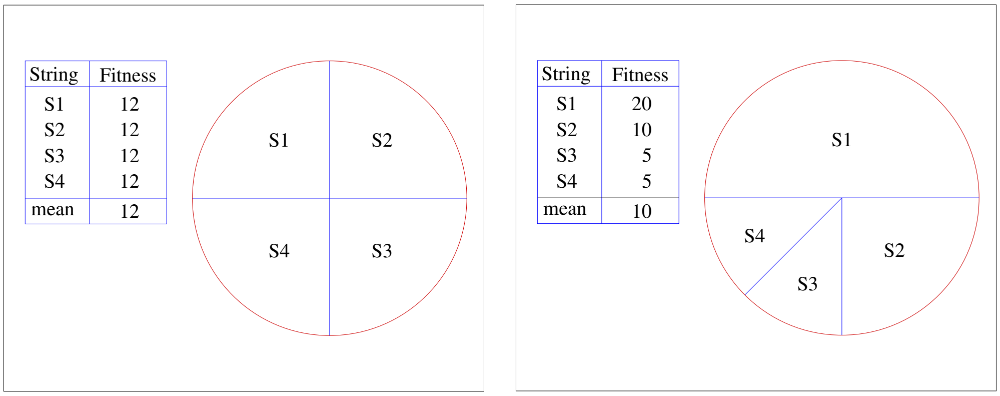


Figure 2.8. Single-Point Crossover



Equal Fitness

Unequal Fitness

Figure 2.9. Roulette Wheel Selection

former selects the μ best individuals drawing from *both* the parents and children, the latter selects μ individuals from the child population only.

Why is the choice of EA selection technique so important? Two conflicting goals are common to all EA search: *exploration* and *exploitation*. Bäck also offers the analogous terms of convergence reliability and velocity, large and small genotypic diversity, and "soft" and "hard" selection [17:pg. 165]. No matter the terminology, one goal is achieved only at the expense of another. An EA's selective pressure is the control mechanism determining the type of search performed. Bäck's analysis shows a general ordering of selection techniques (listed in order of increasing selective pressure): Proportional, linear ranking, tournament, and (μ, λ) selection [17:pg. 180]. Finally, an EA's decision function determines when execution stops. Table 2.1 highlights the major differences between the three major EC instantiations.

Table 2.1. Key EA Implementation Differences

EA Type	**Representation**	**EVOPs**
EP	Real-values	Mutation and $(\mu + \lambda)$ selection alone
ES	Real-values *and* strategy parameters	Mutation, recombination, and $(\mu + \lambda)$ or (μ, λ) selection
GA	Historically binary; Real-values now common	Mutation, recombination, and selection

It is beyond the scope of this research to provide an in-depth analysis of general EVOPs and EA components. Where appropriate, specific EA parameters and values are discussed later in this document to support design decisions.[4]

Although much room for creativity exists when selecting and defining EA instantiations (e.g., genetic representation and specific EVOPs), careful consideration must be given to the mapping from problem to algorithm domains. "Improper" representations and/or operators may have detrimental effects upon EA performance (e.g., Hamming cliffs [17:pg. 229]). Although there is no unique combination guaranteeing "good" performance [105, 346], choosing wisely may well result in more effective and efficient implementations.

2.4.1 EA Mathematical Definition. To formally define an EA its general algorithm is described in mathematical terms, allowing for exact specification of various EA instantiations. In this framework, each EA is associated with a non-empty set I called the EA's *individual space*. Each *individual* $\mathbf{a} \in I$ normally represents a candidate solution to the problem being solved by the EA. Individuals are often represented as a vector ($\vec{\mathbf{a}}$) where the vector's dimensions are analogous to a chromosome's genes. The general framework leaves each individual's dimensions unspecified; an individual ($\mathbf{a}$) is simply that and is modified as necessary for the particular EA instance.

[4]For further information, the interested researcher is directed to the Handbook of Evolutionary Computation [19], probably the most comprehensive collection of articles discussing EC, its instantiations, and applications.

When defining (generational) population transformations Bäck denotes the resulting collection of μ individuals via I^{μ}, and denotes population transformations by the following relationship: $T : I^{\mu} \rightarrow I^{\mu}$, where $\mu \in \mathbb{N}$ [17]. However, some EA variants obtain resulting populations whose size is *not* equal to their predecessors. Thus, this general framework represents a population transformation via the relationship $T : I^{\mu} \rightarrow I^{\mu'}$, indicating succeeding populations may contain the same *or* different numbers of individuals. This framework also represents all population sizes, evolutionary operators, and parameters as ***sequences*** [216]. This is due to the fact that different EAs use these factors in slightly different ways. The general algorithm thus recognizes and explicitly identifies this nuance. Having discussed the relevant background terminology, an EA is then defined as [216][17:pg. 66]:

Definition 9 (Evolutionary Algorithm): *Let I be a non-empty set (the individual space), $\{\mu^{(i)}\}_{i \in \mathbb{N}}$ a sequence in $\mathbb{Z}^+$ (the parent population sizes), $\{\mu'^{(i)}\}_{i \in \mathbb{N}}$ a sequence in $\mathbb{Z}^+$ (the offspring population sizes), $\Phi : I \longrightarrow \mathbb{R}$ a fitness function, $\iota : \bigcup_{i=1}^{\infty} (I^{\mu})^{(i)} \longrightarrow$* {true, false} *(the termination criterion), $\chi \in$* {true, false}*, r a sequence $\{r^{(i)}\}$ of recombination operators $r^{(i)} : \mathbb{X}_r^{(i)} \longrightarrow \mathcal{T}\left(\Omega_r^{(i)}, \mathcal{T}\left(I^{\mu^{(i)}}, I^{\mu'^{(i)}}\right)\right)$, m a sequence $\{m^{(i)}\}$ of mutation operators $m^{(i)} : \mathbb{X}_m^{(i)} \longrightarrow \mathcal{T}\left(\Omega_m^{(i)}, \mathcal{T}\left(I^{\mu'^{(i)}}, I^{\mu'^{(i)}}\right)\right)$, s a sequence $\{s^{(i)}\}$ of selection operators $s^{(i)} : \mathbb{X}_s^{(i)} \times \mathcal{T}(I, \mathbb{R}) \longrightarrow \mathcal{T}\left(\Omega_s^{(i)}, \mathcal{T}\left(\left(I^{\mu'^{(i)} + \chi\mu^{(i)}}\right), I^{\mu^{(i+1)}}\right)\right)$, $\Theta_r^{(i)} \in \mathbb{X}_r^{(i)}$ (the* recombination parameters*), $\Theta_m^{(i)} \in \mathbb{X}_m^{(i)}$ (the* mutation parameters*), and $\theta_s^{(i)} \in \mathbb{X}_s^{(i)}$ (the* selection parameters*). Then the algorithm shown in Figure 2.10 is called an* Evolutionary Algorithm*.* □

2.5 MOEA Overview

MOEAs are a recently developed algorithmic tool with which to solve MOPs. Their popularity can be attributed to several desirable characteristics. For example, Horn notes that many optimization approaches in Section 2.3 were developed for searching intractably large spaces, but that traditional MOP solution techniques generally assume small, enumerable search spaces [152]. More simply, some MOP solution approaches focus on search and others on multiobjective decision making. MOEAs are then very attractive MOP

$$
\begin{array}{l}
t := 0; \\
\textit{initialize } P(0) := \{\mathbf{a_1}(0), \ldots, \mathbf{a}_\mu(0)\} \in I^{\mu^{(0)}}; \\
\textbf{while } (\iota(\{P(0), \ldots, P(t)\}) \neq \text{true}) \textbf{ do} \\
\quad \textit{recombine: } P'(t) := r^{(t)}_{\Theta_r^{(t)}}(P(t)); \\
\quad \textit{mutate: } P''(t) := m^{(t)}_{\Theta_m^{(t)}}(P'(t)); \\
\quad \textit{select:} \\
\quad\quad \textbf{if } \chi \\
\quad\quad\quad \textbf{then } P(t+1) := s^{(t)}_{(\theta_s^{(t)}, \Phi)}(P''(t)); \\
\quad\quad\quad \textbf{else } P(t+1) := s^{(t)}_{(\theta_s^{(t)}, \Phi)}(P''(t) \cup P(t)); \\
\quad\quad \textbf{fi} \\
\quad t := t + 1; \\
\textbf{od}
\end{array}
$$

Figure 2.10. Evolutionary Algorithm Outline

solution techniques because they address *both* search and multiobjective decision making. Additionally, they have the ability to search partially ordered spaces for several alternative trade-offs. Probably most important, however, is the capability for an MOEA to track several solutions simultaneously via its population, whereas traditional MOP solution techniques offer only one solution per "run." Many researchers have successfully used MOEAs to find good solutions for complex MOPs (see Appendix A).

An MOEA's defining characteristic is the set of multiple objectives being simultaneously optimized. Otherwise, a task decomposition clearly shows little structural difference between the MOEA and its single-objective EA counterparts. The following definition and figures explain this relationship.

Definition 10 (Multiobjective Evolutionary Algorithm): *Let $\Phi : I \longrightarrow \mathbb{R}^k$, ($k \geq 2$, a multiobjective fitness function). If this multiobjective fitness function is substituted for the fitness function in Definition 2.10 then the algorithm shown in Figure 2.10 is called a* Multiobjective Evolutionary Algorithm. □

Figures 2.11 and 2.12 respectively show a general EA's and MOEA's task decomposition. The major differences are noted as follows. By definition, Task 2 in the MOEA case computes k (where $k \geq 2$) fitness functions. In addition, because MOEAs expect a *single* fitness value with which to perform selection, additional processing is sometimes required to transform MOEA solutions' fitness *vectors* into a scalar (Task 2a). Although the various transformation techniques vary in their algorithmic impact (see Section 3.3.4) the remainder of the MOEA is structurally identical to its single-objective counterpart. However, this does not imply the differences are insignificant.

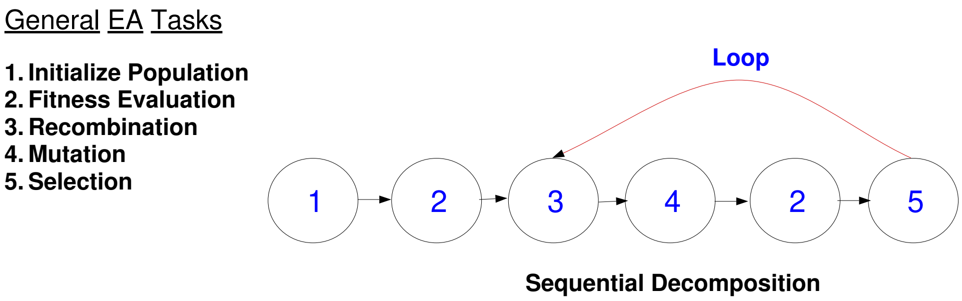


Figure 2.11. Generalized EA Task Decomposition

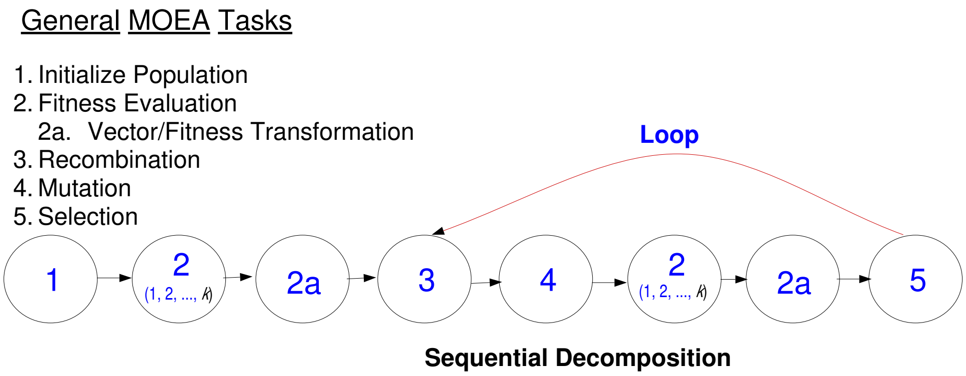


Figure 2.12. MOEA Task Decomposition

2.5.1 Pareto Notation. An MOEA's algorithmic structure can easily lead to confusion (e.g., multiple, *unique* populations) when identifying or using Pareto concepts. In fact, MOEA researchers have erroneously used Pareto terminology in the literature suggesting a more precise notation is required. During MOEA execution, a "current" set of Pareto optimal solutions (with respect to the *current* MOEA generational population) is determined at each EA generation and termed $P_{current}(t)$, where t represents the generation number. Many MOEA implementations also use a secondary population storing nondominated solutions found through the generations [326] (see also Section 3.3.3). Because a solution's classification as Pareto optimal depends upon the context within which it is evaluated (i.e., the given set of which it's a member), corresponding vectors of this set must be (periodically) tested and solutions whose associated vectors are dominated removed.

We term this secondary population $P_{known}(t)$. This term is also annotated with t to reflect its possible changes in membership during MOEA execution. $P_{known}(0)$ is defined as the empty set ($\emptyset$) and P_{known} alone as the *final* set of solutions returned by the MOEA at termination. Different secondary population storage strategies exist; the simplest is when $P_{current}(t)$ is added at each generation (i.e., $P_{current}(t) \cup P_{known}(t-1)$). At any given time, $P_{known}(t)$ is thus the set of Pareto optimal solutions *yet found by the MOEA through generation* t. Of course, the *true* Pareto optimal set (termed P_{true}) is not explicitly known for problems of any difficulty. P_{true} is implicitly defined by the functions composing an MOP; it is fixed and does not change. Because of the manner in which Pareto optimality is defined $P_{current}(t)$ is always a non-empty solution set (see Theorem 1).

$P_{current}(t)$, P_{known}, and P_{true} are sets of MOEA genotypes;[5] each set's corresponding phenotypes form a Pareto front. We term the associated Pareto front for each of these solution sets as $PF_{current}(t)$, PF_{known}, and PF_{true}. Thus, when using an MOEA to solve MOPs, the implicit assumption is that one of the following holds: $P_{known} = P_{true}$, $P_{known} \subset P_{true}$, or $\{\vec{u}_i \in PF_{known}, \vec{u}_j \in PF_{true} \mid \forall i,\ \forall j\ \min[distance(\vec{u}_i, \vec{u}_j)] < \epsilon\}$, where distance is defined over some norm (Euclidean, RMS, etc.).

2.5.2 MOEA Convergence.

If there is no chance of a search algorithm finding the desired solution(s), it makes no sense to implement it. Given that $\vec{x}$ is a decision variable, I the space of all feasible decision variables, Φ a fitness function, and t the generation number, Bäck proves [17:pg. 129] that an EA converges with probability one if it fulfills the following conditions:

$$\forall \vec{x},\ \vec{x}' \in I,\ \vec{x}' \text{ is reachable from } \vec{x} \text{ by means of mutation and recombination;} \qquad (2.10)$$

[5] Horn [152] uses P_{online}, $P_{offline}$, and P_{actual} instead of $P_{current}(t)$, P_{known}, and P_{true}. Our notation is more precise, allowing for each set's generational specification. We also note that $P_{true} = \mathcal{P}^*$ and $PF_{true} = \mathcal{PF}^*$.

and the population sequence $P(0), P(1), \ldots$ is monotone, i.e.,

$$\forall t: \ \min\{\Phi(\vec{x}(t+1) \mid (\vec{x}(t+1) \in P(t+1)\} \leq \min\{\Phi(\vec{x}(t) \mid (\vec{x}(t) \in P(t)\} \tag{2.11}$$

Bäck's definition of monotonicity, appropriate in the context of single objective EAs, is fitness based and assumes that the objective space is totally ordered. Neither of these restrictions is appropriate in the context of MOEAs. A solution's Pareto-based fitness depends on the set within which it is evaluated, and consequently may vary from one generation to the next. Also, the objective space for an MOEA is partially and not necessarily totally ordered. Thus, a convergence theorem for MOEAs requires a more general definition of monotonicity that is both fitness independent and appropriate for objective spaces that are not totally ordered.

One such definition is given by the condition

$$P_{known}(t) = \{\vec{x} \in P_{current}(t) \cup \mid \forall \vec{x'} \in P_{current}(t) \cup \ \text{ s.t. } \ F(x) \preceq F(x')\} \tag{2.12}$$

with $P_{known}(0) = \emptyset$. It can be shown by induction on t that under this condition, $P_{known}(t)$ consists of the set of solutions evaluated through generation t that are Pareto optimal with respect to the set of all such solutions. Thus, $P_{known}(t+1)$ either retains or improves upon solutions in $P_{known}(t)$. In this sense, Condition (2.12) ensures that $P_{known}(t)$ monotonically moves towards P_{true}.

Theorem 4: An MOEA satisfying (2.10) and (2.12) converges to the global optimum of an MOP (PF_{true}) with probability one, i.e.,

$$Prob\{\lim_{t \longrightarrow \infty} \{P_{true} = P(t)\}\} = 1 \ ,$$

where $P(t) = P_{known}(t)$. □

Proof: An MOEA may be viewed abstractly as a Markov chain consisting of two states. In the first state, $P_{true} = P_{known}(t)$, and in the second state this is not the case. By

Condition (2.12), there is zero probability of transitioning from the first state to the second state. Thus, the first state is absorbing. By Condition (2.10), there is a non-zero probability of transitioning from the second state to the first state. Thus, the second state is transient. The theorem follows immediately from Markov chain theory [4]. *Q.E.D.*

2.5.2.1 Other Convergence Proofs. Other research also addresses the desired MOEA convergence. Rudolph's [275] Corollary 2 guarantees that given a countably infinite MOEA population and an MOP, at least one decision variable (x_k) sequence exists such that $f(x_k)$ converges in the mean to PF_{true}, although it appears his nomenclature is inconsistent with accepted definitions.

Rudolph [276] also independently proved that a specific multiobjective ($\mu + \lambda =$ 1+1) ES converges with probability one to a member of P_{true} of the MOP specified by Equation 2.5. His distance metric is in the genotype domain, as compared to ours and his previous work, which is phenotypically based. The EVOPs in his model are not able to search the entire space (in a probabilistic sense) since a step size restriction is placed upon the probabilistic mutation operator. Thus, convergence only occurs when the ES's step size is proportional to the distance to the Pareto set as shown in the elaborate proof. However, this distance is obviously unknown in problems of high complexity which is typical of most real-world problems.

We note his variation kernel (i.e., transition probability function) is equivalent to our reachability condition (appropriate mutation and recombination operators allowing every point in the search space to be visited). He also refers to at least one sequence leading to an associated point on P_{true}, as compared to this work which indicates that through Pareto ranking *all* decision variable sequences lead towards P_{true}; likewise, these variables' phenotypical expressions lead towards PF_{true}.

Rudolph's theorems are for a specific EA *and* MOP instantiation with constrained EVOPs while ours requires a less-specific EA. Both theorems show that what we seek is possible – given MOEAs do converge to an optimal *set*, although Rudolph defines a genotypic optimum and we a phenotypic one. Using phenotypical information is often more appropriate as a decision maker's costs and profits are more accurately reflected in attribute space.

We note here the more important issue is the rate at which an MOEA converges to PF_{true}, and whether $PF_{known}(t)$ uniformly represents PF_{true} as $t \rightarrow \infty$. The MOEA literature is largely silent on these issues, although Rudolph shows the convergence rate for the specific (1+1) EA above is sub-exponential [276]. In this document, Chapter VI presents metrics for possible use in experimentally determining MOEA convergence rate, and Chapter VII shows results for selected experimental problems.

2.6 MOEA Literature Review and Analysis

MOEAs are receiving renewed interest by EA researchers. Although the first MOEA was published in 1984 [288] and a substantial MOEA literature has since developed, there have been only three notable surveys published. Of these, two contain little technical detail of the various MOEA techniques and almost no reference at all to the OR methods from which the techniques were derived!

The reviews by Fonseca and Fleming [111] and by Horn [152] (published in 1995 and 1997) quickly examine major MOEA techniques. The former additionally provides many relevant MOP issues from an MOEA perspective. Both classify existing MOEA approaches differently: Fonseca and Fleming from a broad algorithmic perspective, and Horn from a DM's. More recently, in 1999 Coello Coello [61] presents an MOEA review which classifies implementations from a detailed algorithmic standpoint and adds discussions of the strengths and weaknesses of each technique.

The literature survey conducted as part of this dissertation research offers much more. First, it expands upon previous reviews by classifying and cataloging all known (to date) MOEA efforts and considers more recent and related MOEA citations. Proposed algorithmic approaches are grouped by technique (from a DM's perspective) and key elements of each effort identified in a condensed summary. These results are listed in tabular form, allowing for quick access and easy perusal of past research by technique or approach characteristic. The classification structure used was first proposed by Horn [152];

we substantiate and extend its use. This cataloged presentation highlights previously unnoticed MOEA research trends, clearly distinguishes the various implemented techniques, and identifies distinctive characteristics of each.

Second, the classification structure and cataloged components allow easy identification of "suitable" MOEA techniques for a given MOP. A high-level discussion describes each technique and its mathematical formulation for fitness assignment and/or selection is presented.

Finally, this detailed survey and associated analysis (Appendix A and Chapter III) allows interested researchers to quickly construct MOEAs for investigating MOPs. The classification structure allows quick identification of a (possibly) effective technique(s), EVOPs, and representations. The proposed test suite in Chapter V then allows these MOEA's performance to be compared over selected numerical MOPs.

Freeman Dyson once said, "A good engineer is a person who makes a design that works with as few original ideas as possible." This survey and analysis helps an engineer hold those original ideas to a minimum for some MOP of interest. Scanning the survey's tables may locate similar efforts within some particular problem domain. The tables also provide examples in the form of previously used fitness functions and chromosomal representations. In quick order, an engineer is then able to identify and incorporate appropriate concepts in a new MOEA instantiation. This reference capability is not available in any other MOEA paper. Researchers with basic EA knowledge can use this survey as a largely self-contained introduction to MOEAs.

The review formalizes an algorithmic framework for the important and rapidly expanding research in MOEAs. This listing is *not* complete; no matter the effort spent collecting and evaluating references any proposed listing from this dynamic research field is soon outdated. Although many applications might remain unpublished for confidentiality reasons we conjecture the reported data is representative of the field's direction(s). We now detail the survey's technique classification structure as it is often referred to in succeeding chapters.

2.6.1 MOEA Classification. Many successful MOEA approaches are predicated upon previously implemented mathematical MOP solution techniques. For example, the OR field proposed several methods well before 1985 [70, 158, 308]. Their Multiple Objective Decision Making (MODM) problems are closely related to design MOPs. These problems' common characteristics are a set of quantifiable objectives, a set of well-defined constraints, and a process of obtaining trade-off information between the stated objectives (and possibly also between stated or non-stated non-quantifiable objectives) [158].

Various MODM techniques are commonly classified from a DM's point of view (i.e., how the DM performs search and decision making). Cohon [69] further distinguishes methods between two types of DM: a single DM/group or multiple DMs with conflicting decisions. Here we consider the DM to be either a single DM or a group, but a group united in its decisions.

Because the set of solutions a DM is faced with are often "compromises" between the multiple objectives some specific compromise choice(s) must be made from the available alternatives. Thus, the final MOP solution(s) results from both *optimization* (by some method) and *decision* processes. We choose to classify MOEA-based MOP solution techniques as many OR researchers do, defining three variants of the decision process [70, 158] where the final solution(s) results from a DM's preferences being made known either before, during, or after the optimization process. This is more formally declared as follows [158]:

A Priori **Preference Articulation.** (*Decide* $\longrightarrow$ *Search*) DM combines the differing objectives into a scalar cost function. This effectively makes the MOP single-objective prior to optimization.

Progressive **Preference Articulation.** (*Search* $\longleftrightarrow$ *Decide*) Decision making and optimization are intertwined. Partial preference information is provided upon which optimization occurs, providing an "updated" set of solutions for the decision maker to consider.

A Posteriori **Preference Articulation.** (*Search* $\longrightarrow$ *Decide*) DM is presented with a set of efficient (defined in Section 2.2.1) candidate solutions and chooses from that set.

Basic techniques below this top level of the MODM hierarchy may be common to several algorithmic research fields, however, we limit discussion to implemented MOEA techniques. A hierarchy of the known MOEA techniques is shown in Figure 2.13 where each is classified by the different ways in which the fitness function and/or selection is treated. See Cohon [70] and Duckstein [93] for other multiobjective techniques which may be suitable for but have not yet been implemented in MOEAs.

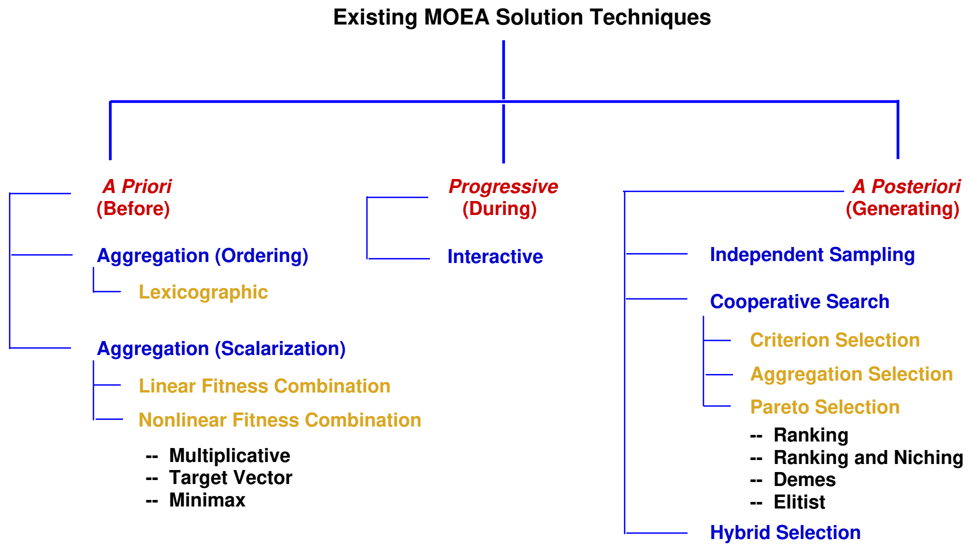


Figure 2.13. MOEA Solution Technique Classification

2.7 Research Assumptions

Only one assumption is made *a priori*, and that involves Pareto optimality. The definition of Pareto optimality implies no particular objective can be further optimized without worsening another objective (with respect to some function). Thus, the Pareto optimal solutions represent *optimal* compromise solutions. Since it makes no sense (theoretically) to accept a sub-optimal solution we define these solutions (P^*) to be the MOP's global optimum solution set. That set is the goal of any MOEA algorithm proposed by or used in this research.

2.8 Summary

This chapter provides an overview of the problem and algorithm domains focused on by this research. MOPs are defined, Pareto concepts introduced, and new theorems and definitions offered. An introduction to general search and optimization techniques is presented along with a broad overview of both EAs and MOEAs. The literature review supporting this effort is described and organized by a new classification structure. Having presented technical definitions and overviews, we now place this effort in context by reviewing related MOEA work in the next chapter.

III. MOEA Analysis and Design

There once was a man who said: "God
Must think it exceedingly odd
If he finds that this tree
Continues to be
When there's no one about in the Quad."
Monsignor Ronald Knox

Dear Sir, your astonishment's odd.
I am always about in the Quad.
And that's why the tree
Will continue to be
Since observed by yours faithfully, GOD.
Anonymous

3.1 Introduction

A conference reviewer once called a particular MOEA implementation "straightforward;" it was also evident the reviewer did not completely understand crucial MOP domain concepts. Conversations with other MOEA researchers indicate they have encountered similar situations. They agree that much time and effort is expended defining and defending MOEA concepts in conference and journal submissions, as it seems many EA practitioners do not have an adequate understanding of basic MOP issues. We hesitate to call any MOEA implementation straightforward, at least as far as achieving effective and efficient performance is concerned.

Appendix A and this chapter together address the many issues involved in MOP and MOEA domain integration. A detailed survey is located in Appendix A which mathematically defines known MOEA solution techniques for MOPs. Each citation therein is cataloged by recording key elements of its approach, and classified using the structure defined in Section 2.6.1. This database currently contains 218 entries representing 272 separate MOEA-based citations from the literature.

This chapter presents a quantitative and qualitative analysis of currently known published MOEA research. Many relevant meta-level topics are addressed, highlighting several MOEA topics which are treated lightly or even ignored in the literature. For example, we discuss MOEA fitness functions, application problem domains, theory, complexity, and other selected topics.

A quantitative and qualitative analysis of known MOEA research is presented in Sections 3.2 and 3.3. Section 3.4 recommends several well-engineered MOEA implementations for possible use. Finally, we highlight what we feel to currently be significant MOEA research contributions in Section 3.5.

3.2 MOEA Research Quantitative Analysis

This section details past MOEA research and is concerned primarily with analyzing raw data, while Section 3.3 presents analysis of a more observational nature. We are concerned in this section with issues such as the number of MOEA research efforts, practicality of the various implemented techniques, fitness functions and chromosomal representations used in MOEA research, and the problem domains in which MOEAs have been applied. This treatment of major MOEA research issues shows the interested practitioner where and how the field has focused its energies.

3.2.1 MOEA Citations. Three graphs quantifying the cataloged citations are presented here:[1] Figure 3.1 shows the number of citations by year, Figure 3.2 by technique, and Figure 3.3 by type. We immediately see that the initial transformations of EAs into the multiobjective domain did not spark any real interest for several years (Figure 3.1). We also note here that although Schaffer "invented" the first MOEA, Fourman too deserves credit for his different MOEA implementations published about the same time. Not until the mid 1990's is there a noticeable increase in published MOEA research. However, this increase is substantial as almost three times as many MOEA approaches were published in the last six years (1994-1999) as in the first ten (1984-1993). The sheer number of recent publications indicates an active research community interest in MOEAs.

As noted in Section 2.6.1, we have classified MOEA approaches into three major categories. These categories and the specific techniques they embody are listed below.

[1] As noted in Section A.1.2, a few efforts are classified under two MOP techniques reflecting dual approaches proposed in the same citation. Additionally, some efforts have multiple citations indicating a great deal of duplication between the cited papers. We ignore these minor anomalies and deal here with the total number of classified efforts within each technique; the interest is in identifying MOEA research *trends* rather than absolute values.

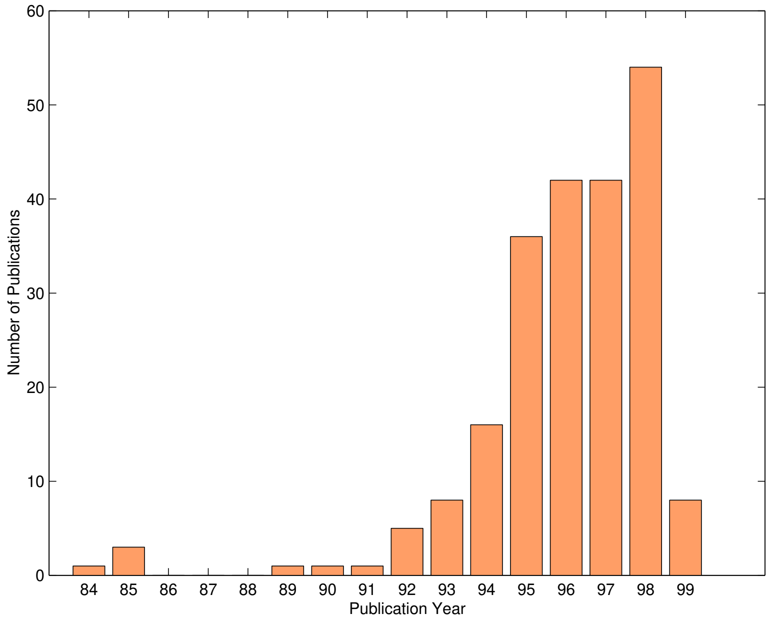


Figure 3.1. MOEA Citations by Year

***A Priori* Techniques:** Lexicographic, linear fitness combination, and nonlinear fitness combination

***Progressive* Techniques:** Progressive

***A Posteriori* Techniques:** Independent sampling, criterion selection, aggregation selection, Pareto-based selection, Pareto rank- and niche-based selection, Pareto deme-based selection, Pareto elitist-based selection, and hybrid selection

Comparing citations by technique highlights the popularity of *a posteriori* techniques (Figure 3.2). Over twice as many citations occur in that category as in the *a priori* and progressive categories combined. Does this imply a willingness by DMs to select solutions from (possibly) unbiased searches? Or is it that DMs are unwilling (or unable) to assign priorities to objectives without further information? At least in real-world problems, it seems reasonable for DMs to expend the necessary resources to first perform a search for possible solutions. Making a decision *a posteriori* could well be less expensive in the long run than making decisions without the additional knowledge gained through initial or interactive search.

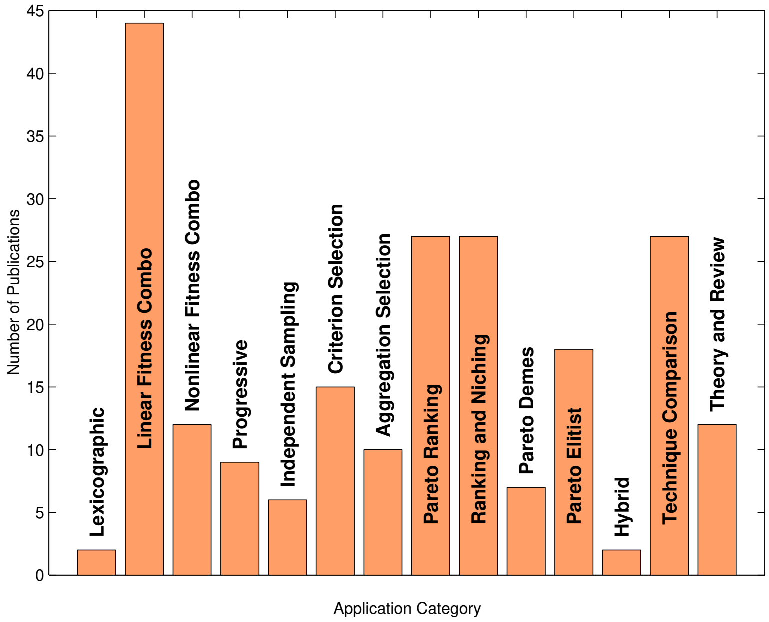


Figure 3.2. MOEA Citations by Technique

When considering the *a posteriori* techniques, almost twice as many Pareto sampling approaches exist as the others combined. The number of papers comparing MOEAs is a healthy sign of skepticism, in that researchers are seeking to compare proposed algorithms on a variety of problems.

Note that MOEA theory noticeably lags behind applications, at least in terms of published papers. This is even clearer when noting few of these categorized papers (see Section A.5.2) *concentrate* on MOEA theoretical concerns. The others discuss some MOEA theory but do so only as regarding various parameters of their respective approaches. This quantitative lack of theory is not necessarily bad but indicates further theoretical development is necessary to (possibly) increase the effectiveness and efficiency of existing MOEAs. Section 3.3.2 discusses many MOEA theoretical issues in detail.

Finally, Figure 3.3 shows the most popular MOEA implementation *by far* is a Multiobjective Genetic Algorithm (MOGA).[2] This is nine times the number of implementations

[2]As noted in Section 2.3, the terms *EA* and *EC* embody several specific techniques. Figure 3.3 tracks the following: Multiobjective Evolutionary Programming (MOEP), Multiobjective Evolutionary Strategies (MOES), MOGA, and Multiobjective Genetic Programming (MOGP).

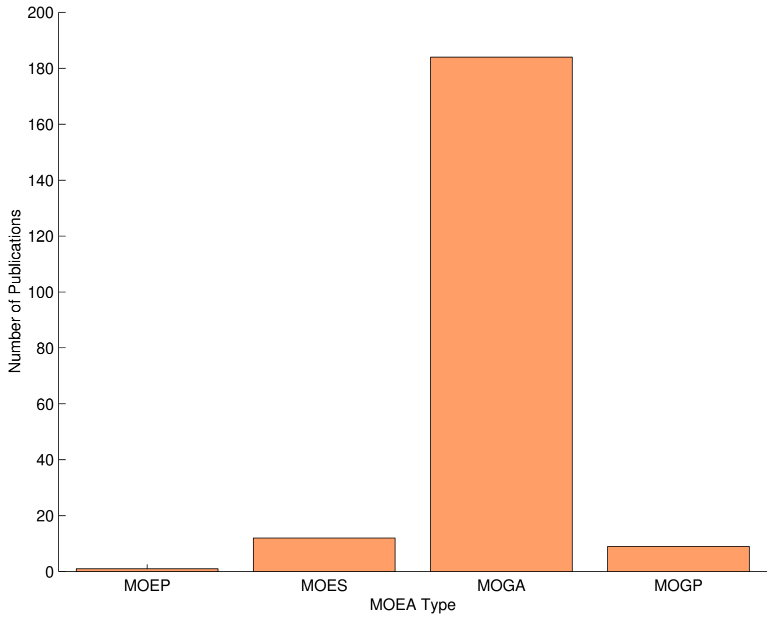


Figure 3.3. MOEA Citations by Type

as all other types combined. Also, observe that only one MOEP is reported in the literature.

3.2.2 MOEA Technique Discussions. Real estate agents claim three major factors set the price one can reasonably expect when buying or selling a home: location, location, and location! There is a direct analogy when using MOEAs to solve MOPs. To wit, three factors determine the effectiveness and/or efficiency of a particular MOEA: the problem domain, the problem domain, and the problem domain! An MOEA should be applied only when the problem requires it. A particular problem instance may also determine MOEA performance. This is no different than is the case with single-objective EAs but bears mentioning.

Many MOEA implementations are currently available. Selecting an appropriate technique and approach is dependent upon meticulous examination of the problem domain; ensuring derived solutions are the best available requires careful integration of both problem and algorithm domains. Identifying MOEA techniques and approaches which have and have not historically "worked" should improve future MOEAs. Thus, this section presents general observations about the categorized approaches. General comments about each high-level technique are given and followed by detailed discussions of the approaches cataloged within that technique.[3]

3.2.2.1 A Priori *Techniques.* By definition, these techniques require objective importance to be defined before search occurs. In real-world scientific and engineering problems this is a non-trivial task. The ramifications of "bad" objective prioritization choices are easy to see: the DM's "cost" (no matter how defined) could be greater than necessary as more "acceptable" solutions are missed. No matter the optimization algorithm used, this is an inescapable consequence of *a priori* MOEA techniques, which we now examine in detail.

Lexicographic techniques have not found favor with MOEA researchers, as only two implementations are reported. This may be due to the fact this technique explores objective space unequally, in the sense that priority is given to solutions performing well in one objective over another(s). Or in other words, one objective is optimized at all costs.

The lexicographic technique appears most suitable *only* when the importance of each objective (in comparison to the others) is clearly known. However, trade-offs do exist. On one hand, any reported solutions are Pareto optimal (by definition and with respect to all solutions evaluated) and are thus part of the global optimum. On the other hand, when is such an "all costs" goal necessary or even appropriate? If one objective is to be optimized regardless of the others' expense, it seems more appropriate to instead use a single objective EA which does not incur the additional overhead of an MOEA.

The linear fitness combination technique is a popular approach despite its identified shortfalls, probably due to its simplicity. Section A.2.2 reflects its application to many real-world scientific and engineering problems where it is often incorporated with "variations on a theme." A basic weighted sum MOEA is both easy to understand and implement; the fitness combination technique is also computationally efficient. If the problem domain is "easy" and a sense of each objective's relative worth is known and can be quantified, or even if the time available for search is short, this may be a suitable method to discover

[3]The interested reader is referred to Coello Coello [61] for a more complete description and discussion of attendant strengths/weaknesses for many of these approaches.

an acceptable MOP solution. However, this technique has a major disadvantage due to certain MOP characteristics.

Fonseca and Fleming [107] explain that for any positive set of weights and fitness function Φ (see Equation A.6 in Section A.2.2), the returned global optimum is always a Pareto optimal solution (with regard to all others identified during search). However, if PF_{true} is nonconvex, optima in that portion of the front can *not* be found via this method. Thus, blindly using this technique guarantees that some solutions in P_{true} can not be found when it is applied to certain MOPs. Fonseca and Fleming also state that linear fitness combination is the most popular MOEA technique. Figure 3.2 clearly indicates over twice as many implemented Pareto-based approaches. Thus, their statement is no longer true.

Researchers appear leery of applying nonlinear combination techniques. For example, of the two cited multiplicative efforts only one reports actually *implementing* the technique. This may be due to the overhead involved in determining appropriate probability of acceptance or utility functions, and to the various conditions which these objective functions must meet [177]. This additional overhead may not justify resulting solutions' "quality."

A target vector fitness combination (goal programming) approach is incorporated into four MOEAs. If a DM is certain of each objective's desired levels this technique may produce acceptable solutions. Just as in all *a priori* techniques, though, specifying exact goals or weights before search may unnecessarily limit the search space and therefore "miss" desirable solutions. Algorithmic overhead is minimal when implementing this technique because the desired goal levels are directly incorporated into the fitness function. These comments also hold true for the cited minimax techniques. Finally, we again find that using these techniques to minimize Φ does *not* guarantee resulting solutions are members of P_{true} [107].

It appears that these *a priori* MOEA techniques may be undesirable for general use. If a DM is expending resources to search for MOP solutions, it stands to reason optimal (or "good") solutions are desired (expected?). Because these techniques arbitrarily limit the search space they can not find all solutions in P_{true}. Additionally, as is shown in

Section 3.4, implementing "more" effective and efficient MOEAs might not be as difficult and involve less overhead than imagined.

3.2.2.2 Progressive Techniques. The lack of cited interactive search efforts in the MOEA literature is surprising. It seems that no matter what MOP solution technique is implemented, close interaction between the DM and "searchers" can only increase the efficiency (or "desirability") of discovered solutions. It is understandable that a DM's time is at a premium. At least to some level, though, more interaction certainly implies "better" results. Although either *a priori* or *a posteriori* techniques may be used interactively, the latter are more suited to MOPs because they offer a *set* of solutions rather than just one. There is a limit to how much information a DM can process at one time, but surely some greater number of choices vice one or two is generally more advantageous.

Incorporating DM preferences within and through an interactive search and decision making process may benefit all involved. Do researchers and/or practitioners feel they don't have the time? Or is it the DM who balks at the additional effort? Real-world applications should surely use this interactive process as the economic implications can be quite significant. In fact, several MOEAs [108, 99, 156] are able to explicitly incorporate DM preferences within search.

3.2.2.3 A Posteriori *Techniques.* As indicated in Section A.4 these techniques are explicitly seeking P_{true}. An MOEA search process is executed with resultant solutions and their evaluations (P_{known} and PF_{known}) provided to the relevant DM. We now examine these techniques in detail.

Several independent sampling approaches are reported but we question their overall effectiveness (see Section A.4.1). All cited efforts use some fitness combination technique where the weights assigned to each objective are uniformly varied between a number of separate MOEA runs. This technique may have limited utility if only two objectives are being considered. For example, assume an MOEA using a linear fitness combination approach. If each objective's weight varies from 0 to 1 by 0.05 increments, only 21 MOEA runs are necessary to explore the possible weight combinations and give some picture of PF_{known}. However, even varying the weights at this coarse resolution results in the required number of runs combinatorially increasing with the number of objectives. Thus, its overall usefulness seems quite limited especially as the arbitrary weight combinations may well prevent discovery of some solutions in P_{true}, and also in view of other techniques' strengths.

Schaffer's Vector Evaluated Genetic Algorithm (VEGA) [289] is an example of a criterion selection technique where fractions of succeeding populations are selected based on *separate* objective performance. This is the first time we see an MOEA's population capability fully used in that the MOEA returns a number of solutions *within a single run*. However, some criterion techniques are faulted for ignoring solutions performing "acceptably" in *all* dimensions in favor of those performing "well" in only one [152].

Crossley et al. [76, 237] believe this technique reduces the diversity of any given $PF_{current}(t)$. They implement elitist selection to ensure $PF_{known}(t)$ endpoints (or in other words, $PF_{known}(t)$'s extrema) survive between generations, noting that otherwise the MOEA converges to a single design rather than maintaining a number of alternatives. In other attempts to preserve diversity in $PF_{current}(t)$ they also employ a VEGA variant. Here, "k"-branch tournaments (where k is the number of MOP objectives) allow each solution to compete once in each of k tournaments, where each set of tournaments selects $1/k$ of the next population [170].

Aggregative selection MOEAs incorporate a variety of techniques to solve MOPs. Section A.4.3 shows weighted sums, constraint and objective combinations, and hybrid search approaches used. However, rather than using static weight combinations for the objectives throughout an MOEA run, the weights are varied between generations and/or each function evaluation. Sometimes the weights are assigned randomly, sometimes they are functions of the particular solution being evaluated, and in other cases are encoded in the chromosome as genes where EVOPs act upon the them also.

The major advantage of both criterion and aggregation selection techniques is the *set* of solutions returned by each MOEA run. Thus, P_{known} and PF_{known} may be reasonable approximations to P_{true} and PF_{true}, and have required only one MOEA run. These methods are not without their disadvantages, however. When using the weighted sum technique we know certain members of PF_{true} may be missed. Both the constraint/objective combination and hybrid search approaches have significant overhead (e.g., solving a linear system of equations to determine an appropriate hyperplane [356]). Thus, a fitness assignment or selection technique able to "easily" find all members of P_{true} and PF_{true} is desired. Pareto sampling offers this capability.

Almost 90% of reported Pareto-based MOEAs are applied to real-world scientific and engineering problems. This certainly implies Pareto techniques are suitable for a number of different engineering problem domains. Additionally, rather than the usual two objective functions, several Pareto-based approaches used three, four, seven, or more. The Pareto methodology handles this increased number of functions easily.

Figure 3.2 shows the major body of MOEA research centering upon approaches exploring "equally" in all objective dimensions (the Pareto sampling techniques). Furthermore, judging merely by the number of published efforts, more interest is evident in either Pareto-based or Pareto rank- and niche-based selection techniques as either has more citations than Pareto deme- and elitist based selection. As no direct comparisons have yet been made attesting to the efficacy of these various Pareto approaches, this is not to say that Pareto deme- or elitist-based selection is not worthwhile. The only existing criticism is that Pareto elitist approaches may not retain diverse enough populations to find and retain a PF_{known} truly representative of PF_{true}, as they retain only $P_{current}(t)$ between generational populations and discard all other solutions. As more and more population members are contained in $P_{current}(t)$ the remaining solutions may not provide enough diversity for effective further exploration.

The sheer number of Pareto sampling approaches indicates many researchers see merit in the basic methodology. As the global optimum of an MOP is PF_{true} [325], using a Pareto-based approach seems reasonable. However, in order to determine a particular MOEA implementation's effectiveness and efficiency, systematic comparison using appropriate metrics on carefully selected test problems should be performed. Although several MOEA comparison papers exist this has not yet been accomplished.

3.2.2.4 MOEA Comparisons. To date, most MOEA researchers' *modus operandi* is comparing some MOEA (usually the researcher's own new and improved variant) against an older MOEA (often VEGA, even with it's identified shortfalls), and analyzing results for some MOP (often Schaffer's F2 [289] or some other numeric example). Comparative results are then "clearly" shown in graphical form indicating which algorithm performed better, implying its returned PF_{known} is a better representation of PF_{true}. Only recently (1998) has any researcher proposed experimental methodologies for *general* MOEA comparative analysis [359]; we present an extensive discussion on this subject in Chapter VI. To their credit, many of these publications also compare MOEA performance on real-world applications. An argument can be made down the lines of "if it works, use it," but in general, using a test problem and/or an application's results to judge comprehensive MOEA usefulness is not conclusive.

3.2.2.5 MOEA Theory. Less than $1/10^{th}$ of published MOEA papers focus on underlying theoretical analyses of MOEAs. These papers focus mainly on MOEA parameters, behavior, and concepts. They attempt to further define the nature and limitations of Pareto optimality, the subsequent effects upon MOEA search, and discuss the characteristics and construction of an appropriate MOEA benchmark test function suite. Although other MOEA researchers often cite these works, our detailed categorizations show their efforts to often be modifications of previously implemented approaches, or perhaps the same approach applied to a different application. These papers add little or nothing to the body of MOEA theory. Fonseca and Fleming [111] and Horn [152] state that more effort is being spent designing and refining MOEA approaches than on developing accompanying theory. We not only agree with this but have clearly shown it to be a fact.

3.2.3 MOEA Fitness Functions. The cataloged research efforts provide various fitness function types used by MOEAs. Table 3.1 lists several generic fitness function types, their identifying characteristics, and examples of each drawn from the MOEA literature. These listed types are not limited to MOEA applications nor are they the only ones possible. However, MOEAs offer the exciting possibility of simultaneously employing different

Table 3.1. MOEA Fitness Function Types

Category	**Characteristic**	**Examples**
Electromagnetic	Energy transfer or reflection	[220, 225, 328]
Economic	Production growth	[137, 297]
Entropy	Information content and (dis)order	[112, 183, 274]
Environmental	Environmental benefit or damage	[5, 58, 322]
Financial	Direct monetary (or other) cost	[16, 156, 330]
Geometrical	Structural relationships	[92, 117, 167]
Physical (Energy)	Energy emission or transfer	[171, 249, 343]
Physical (Force)	Exerted force or pressure	[74, 242, 331]
Resources	Resource levels or usage	[21, 90, 297]
Temporal	Timing relationships	[108, 163, 297]

fitness functions to capture desirable characteristics of the problem domain regardless of implemented MOEA technique.

The fitness functions employed appear limited only by the practitioner's imagination and particular application; several are identified and others must surely exist. However, a fitness function's effectiveness depends on its application in appropriate situations (i.e., it measures some *relevant* feature of the studied problem). The claim by many authors that their particular MOEA implementations are successful imply the associated fitness functions are appropriate for the given problem domains.

Finally, the cataloged efforts clearly show the non-commensurability and independence of many fitness function combinations. For example, optimizing a radio antenna design may involve electromagnetic (energy transmission), geometric (antenna shape), and financial (dollar cost) objectives. The proposed antenna's shape may have no meaningful impact on its cost. Also, these objectives may be measured in megawatts, feet, and euros! These are the factors responsible for the partial ordering of the search space and the subsequent need to develop appropriate MOEA fitness assignment procedures.

3.2.4 MOEA Chromosomal Representations. Theorems exist [105] showing no intrinsic advantage exists in any given genetic representation. For any particular encoding and associated cardinality, *equivalent* evolutionary algorithms (in an input/output sense) can be generated *for each individual problem instance.* Although certain gene representations and EVOPs may be more effective and efficient in certain situations, the theorems show that no choice of representation and/or EVOPs operating on one or two parents offers any capability which can't be duplicated by another MOEA instantiation.

The No Free Lunch (NFL) theorems [346] indicate that if an algorithm performs "well" (on average) for some problem class then it must do worse on average over the remaining problems. In particular, if an algorithm performs better than random search on some problem class then it must perform *worse* than random search on the remaining problems. So, although the NFL theorems imply one MOEA may provide "better" results than another when applied to some problem these other theorems show that that MOEA is not unique. Thus, there appears to be more than one way to skin a cat (or MOP).

Genetic representation is then another MOEA component limited only by the implementor's imagination. The cited efforts indicate the most common representation is a binary string corresponding to some simple mapping from the problem domain. Real-valued chromosomes are also often used in this fashion. And, as in single-objective EAs, combinatorial optimization problems often use a permutation ordering of jobs, tasks, etc. However, some representations are more intricate and therefore notable.

Some MOEAs employ arrays as genome constructs. For example, Baita uses a matrix representation to store recessive information [21].[4] Parks and Chow also use matrices as these data structures are more natural representations of their respective problem domains' decision variables [250, 57]. The Prüfer encoding used by Gen [123] uniquely encodes a graph's spanning tree and allows easy repair of any illegal chromosome. In the known multiobjective Genetic Programming implementations (e.g., [191, 151, 278]), a program/program tree representation is used. No matter the representation employed, we again see any claims of "successful" MOEA implementations imply the associated genetic encodings are appropriate for the given problem domain.

3.2.5 MOEA Problem Domains. MOEAs operate on MOPs by definition. A more theoretical discussion of the MOP domain is given in Chapter V and elsewhere [327, 83]; we here discuss it in more general terms. When implementing an MOEA it is (implicitly) assumed that the problem domain (fitness landscape) has been examined, and a decision

[4] As a side note, only two published MOEAs use dominant and recessive genetic information [21, 189].

made that an MOEA technique is the most appropriate solution tool for the given MOP. In general, it is accepted that single-objective EAs are useful search algorithms when the problem domain is multidimensional (many decision variables), and/or the search space is very large. Most cited MOEA problem domains appear to exhibit these characteristics.

An overwhelming majority of cited efforts are applied to non-pedagogical problems. This indicates MOEA practitioners are developing and implementing MOEAs as real-world tools. As a quick glance through Appendix A shows, these implementations span several disparate scientific and engineering research areas and give credibility to the MOEA's claim as an effective and efficient general purpose search tool.

3.3 MOEA Research Qualitative Analysis

What differentiates an MOEA from a single-objective EA? What components should be included in an MOEA? When should an MOEA be used? This section addresses these questions and presents matters of a more philosophical nature raised by the preceding discussion, considering several MOEA design issues. Although not quantitatively derived, our analytical observations are based on the cataloged presentation in Appendix A and substantiated with other relevant citations from the literature.

3.3.1 MOEA Characteristics. Of course, the major MOEA defining characteristic is the set of multiple objectives being simultaneously optimized. Although the cited efforts in Sections A.2 through A.4 explain *how* various MOEAs incorporate these multiple objectives, they do not always explain *why*. This may well be due to a lack of MOEA theory.

3.3.2 MOEA Theoretical Issues. We agree with other MOEA researchers [152, 111] that MOEA theory is lagging behind MOEA implementations and applications. For example, until recently no proof was offered showing an MOEA is capable of converging to P_{true} or PF_{true} (see Section 2.5.2). We show in Figure 3.1 that although the number of MOEA implementations is significant, this fact alone does not indicate a corresponding *depth* of associated theory (as reflected by Table A.16 in Section A.5.2). This research makes absolutely clear that more effort has been spent designing new or variant MOEA approaches, and not in comprehensively reviewing the benefits and/or trade-offs of the various implementations.

Why is there such a lack of underlying MOEA theory? Although some mathematical foundations exist the current situation seems akin to Goldberg's recent comparisons of engineer and algorithmist [127]. He likens algorithms to "conceptual machines" and implies computer scientists are hesitant to move forward without exact models precisely describing their situation. On the other hand, he claims a design engineer often accepts less accurate models in order to build the design. MOEA researchers certainly seem to have taken this approach!

Realizing that simple assumptions are sometimes made in order to develop limited theoretical results, the foundations of single-objective EA theory are well-established. The *Handbook of Evolutionary Computation* [19] devotes entire chapters to theoretical EC results established during the past 20-30 years. Sample topics include EA types, selection, representation, crossover, mutation, fitness landscapes, and so on. Several foundational textbooks are also available, such as those by Goldberg [126], Michalewicz [218], and by Bäck [17]. Although much of this theory is (may be?) valid when regarding MOEAs, some is not. Thus, this section discusses current knowledge concerning selected MOEA theoretical issues.

3.3.2.1 Fitness Functions. The general manner of fitness function implementation is two-fold. This is reflected by the work of Wienke et al. [343] and Fonseca and Fleming [112], who each solved MOPs with seven fitness functions. Wienke et al. essentially used seven copies of an identical objective function, which was to meet atomic emission intensity goals for seven different elements. Although the elements and associated goals are each different the fitness functions are conceptually identical. This does not make the MOP "easier" but perhaps makes the objective space somewhat easier to understand.

On the other hand, Fonseca and Fleming's MOP's seven objectives appear both incommensurable and independent. Both P_{known} and PF_{known} are hard to visualize, as are their interrelationships. For example, when considering the mathematical polynomial

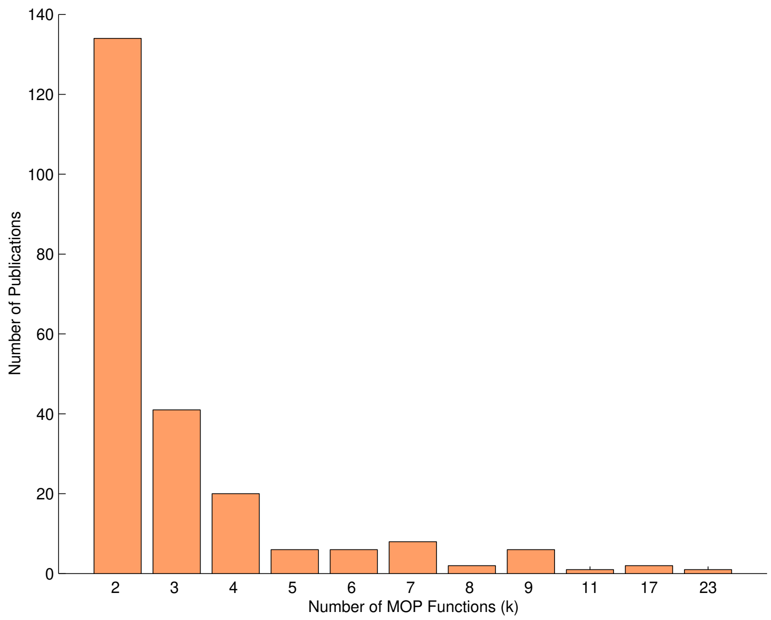


Figure 3.4. MOEA Citations by Fitness Function

model constructed by their MOEA, it is unclear how the number of terms affects the long-term prediction error and how that error may affect variance and model lag.

With that said, Figure 3.4 shows the number of citations employing a given number of fitness functions. The overwhelming majority use only two fitness functions, most probably for ease and understanding. Several use three to nine, and the currently known maximum is 23 fitness functions within a single MOEA. This approach used an MOEA to solve a heavily constrained single-objective optimization problem [62]. Thus, one objective was the fitness function and the other 22 were constraints cast as objectives. Of the two efforts using 17 objectives, one doesn't specify the specific objectives [260] and the other implements conceptually identical objectives [269]. The highest number of conceptually different implemented fitness functions is found in a linkage design problem [285] where nine objectives are used.

How many fitness functions are enough? How many objectives are generally required to adequately capture an MOP's essential characteristics? *Can* all characteristics be captured? The cataloged efforts imply most real-world MOPs are effectively solved using only two or three. There is a practical limit to the maximum number of possible objective functions, as the time to compute several complex MOEA fitness functions quickly becomes unmanageable.

A theoretical limit also exists as far as Pareto optimality is concerned. As additional objectives are added to an MOP more and more MOEA solutions meet the definition of Pareto optimality. Thus, as Fonseca and Fleming indicate for most Pareto MOEAs [111], the size of $P_{current}\,(t)$, $PF_{current}\,(t)$, $P_{known}\,(t)$, and $PF_{known}\,(t)$ grows, and Pareto selective pressure decreases. However, some confusion results from both their and Horn's [152] statements implying that the size of PF_{true} grows with additional objectives. We show that the Pareto front is composed of Pareto curve(s), Pareto surface(s), or some combination of the two (see Section 2.2.2.2). And, as Cantor proved [138], the infinity of points on a line, surface, cube, and so on are the same (represented by $\aleph_1$). Thus, the cardinality of PF_{true} does *not* grow with the number of objectives, only (possibly) it's topological dimension. However, since MOEAs deal with *discretized* numerical representations the number of possible solutions (and therefore the number of computable vectors composing PF_{known}) may increase as more objectives are added.

Finally, some limit to human understanding and comprehension exists. The human mind appears to have a limited capacity for simultaneously distinguishing between multiple pieces of information or concepts. Perhaps this is best noted by Miller's [222] seminal paper proposing a human one-dimensional span of judgment and immediate memory of 7 ± 2. He notes that adding objective dimensions increases this capacity but at a decreasing rate. This seems to argue a "more the merrier" viewpoint for the number of MOP objectives, but visualizing and understanding objective inter-relationships becomes more difficult as their numbers grow. Thus, certain techniques are designed to map high-dimensional information to two- or three dimensions for better understanding (e.g., Sammon mapping [284] and profiles [81]). Fonseca and Fleming [108, 112, 113] often use profiles (or tradeoff graphs) to show MOEA solution values and their interrelationships. Figure 3.5 is an example profile for an MOP with seven objectives; the lines simply connect each solution's objective values.

Past MOEA implementation results imply that two or three objectives are "satisfactory" for most problem domains. Thus, MOEA application to a given MOP should begin with two or three primary objectives in an effort to gain problem domain understanding.

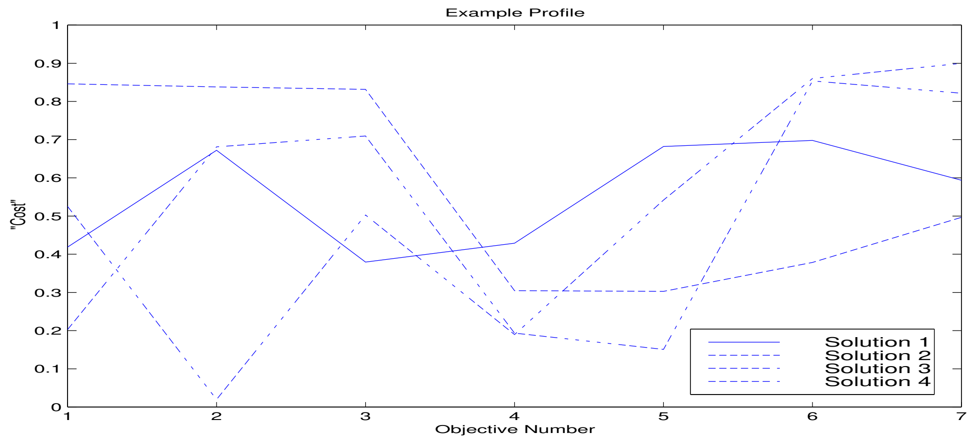


Figure 3.5. Example MOP Profile

One may be able to ascertain how the different objectives affect each other and an idea of the fitness landscape's topology. Other fitness functions may then be added in order to capture other relevant problem characteristics. Table 3.1 in Section 3.2.3 identifies several fitness function categories for this purpose.

3.3.2.2 Pareto Ranking. Two Pareto fitness assignment methods are primarily used in MOEAs although variations do exist. In general, all assign preferred (Pareto optimal) solutions the same rank and other solutions some higher (less desirable) rank. With the scheme proposed by Goldberg [126], where a solution x at generation t has a corresponding objective vector x_u, and N is the population size, the solution's rank is defined by the algorithm in Figure 3.6.

The second technique, proposed by Fonseca and Fleming [111], operates somewhat differently. As before, a solution x at generation t has a corresponding objective vector x_u. We also let $r_u^{(t)}$ signify the number of vectors associated with the current population dominating x_u; x's rank is then defined by:

$$\text{rank}(x, t) = r_u^{(t)} \ . \tag{3.1}$$

This ensures all solutions with nondominated vectors receive rank zero.

```
curr_rank = 1
m = N
while N ≠ 0 do
    For i = 1 : m do
        If x_u is nondominated
            rank(x, t) = curr_rank
    od
    For i = 1 : m do
        If rank(x, t) = curr_rank
            Store x in temporary population
            N = N − 1
    od
    curr_rank = curr_rank + 1
    m = N
od
```

Figure 3.6. Rank Assignment Algorithm

Some approaches simply split the population in two, e.g., assigning solutions with nondominated vectors rank 1 and all others rank 2 [25]. Using the same notation, this ranking scheme is defined by:

$$\mathrm{rank}(x,t) = \begin{cases} 1 & \text{if } r_u^{(t)} = 0, \\ 2 & \text{otherwise.} \end{cases} \tag{3.2}$$

When considering Goldberg's and Fonseca and Fleming's ranking schemes, it initially appears that neither is "better" than the other, although it is mentioned in the literature that Fonseca and Fleming's method, which effectively assigns a cost value to each solution, might be easier to mathematically analyze [107]. Horn [152] also notes this ranking can determine more ranks (is finer-grained) than Goldberg's (assuming a fixed population size).

One last ranking method using Pareto optimality as its basis is proposed by Zitzler and Thiele [358].[5] Their MOEA implementation uses a secondary population whose solutions are directly incorporated into the generational population's fitness assignment procedure. Effectively, each Pareto optimal solution (at each generation) is assigned a fit-

[5]Their rank assignment algorithm is lengthy. The reader is instead referred to the citation for implementation details.

ness equal to the proportion of evaluated vectors its associated vector dominates. Because of the secondary population's inclusion in the fitness assignment process this method's complexity may be significantly higher than the other methods. Additionally, this method has a known shortfall. Deb [83] presents a geometric argument that this fitness assignment method has inherent bias. Pareto optimal solutions whose associated vectors dominate more vectors (or dominate a larger portion of objective space) receive higher fitness than other Pareto optimal solutions. However, each Pareto optimal solution should receive equal fitness! This method is then biased, as it may result in some Pareto optimal solutions receiving preference over others in the selection process.

There is currently no clear evidence as to the benefit(s) of any of these ranking schemes over another. Only one experiment whose purpose is directly comparing any of these schemes is reported in the literature. Thomas compared Fonseca and Fleming's and Goldberg's Pareto ranking schemes in an MOEA applied to submarine stern design [318]. He concludes both outperformed tournament selection, and that Fonseca and Fleming's ranking appears to provide a fuller, smoother PF_{known} . However, he (and we) caution that this is a singular data point. On a similar note, only one paper in the MOEA literature presents data on the number of population "fronts" using Goldberg's ranking. Vedarajan et al. present a graph showing the number of fronts found in each generation [329]. With a population size of 300 individuals the first generation has over 40 fronts. This quickly drops and from generations 10 to 100 and oscillates between 20 and 25.

Analyzing these schemes' mathematical complexity is revealing. Table 3.2 (showing each scheme's best and worst case) and the following analysis only consider population size in computing complexity, where N is the size of the generational population and N_1 of P_{known} . We assume that as comparisons are performed appropriate counter or fitness value assignments are made or updated. Thus, the binary, Fonseca and Fleming's, and Zitzler's ranking schemes require only one "pass" through the population(s) regardless of the number of nondominated solutions. Their worst and best case complexities are identical. Goldberg's scheme, however, requires at most $N-1$ "passes" through the population if there is only one Pareto optimal solution per (reduced) population. In addition, Zitzler's scheme's complexity increases if P_{known} 's size is much larger than the generational popu-

Table 3.2. MOEA Fitness Ranking Complexities

Technique	**Best Case**	**Worst Case**
Binary	$N^2 - N$	$N^2 - N$
Fonseca	$N^2 - N$	$N^2 - N$
Goldberg	$N^2 - N$	$\frac{1}{3}(N^3 - N)$
Zitzler	$(N + N_1)^2 - N - N_1$	$(N + N_1)^2 - N - N_1$

lation's. Thus, Goldberg's and Zitzler and Thiele's ranking schemes (potentially) involve significantly more overhead than do the others.

It is also instructional to look at the possible value ranges for each ranking scheme. The binary scheme (Equation 3.2) offers only two values, $\Phi \in [0, 1]$. Both Fonseca and Fleming's (Equation 3.1) and Goldberg's scheme (Figure 3.6) offer N possible values, $\Phi \in [0, 1, \ldots, N-1]$. However, in practice Goldberg's scheme uses some subset of these values (resulting in a "coarser" ranking). Zitzler's scheme offers (possibly non-integer) values $\Phi \in [1, N)$. Using Fonseca's second function as an example (see Table B.1 in Appendix B), Figure 3.7 shows the resultant solution rankings of three Pareto ranking schemes.

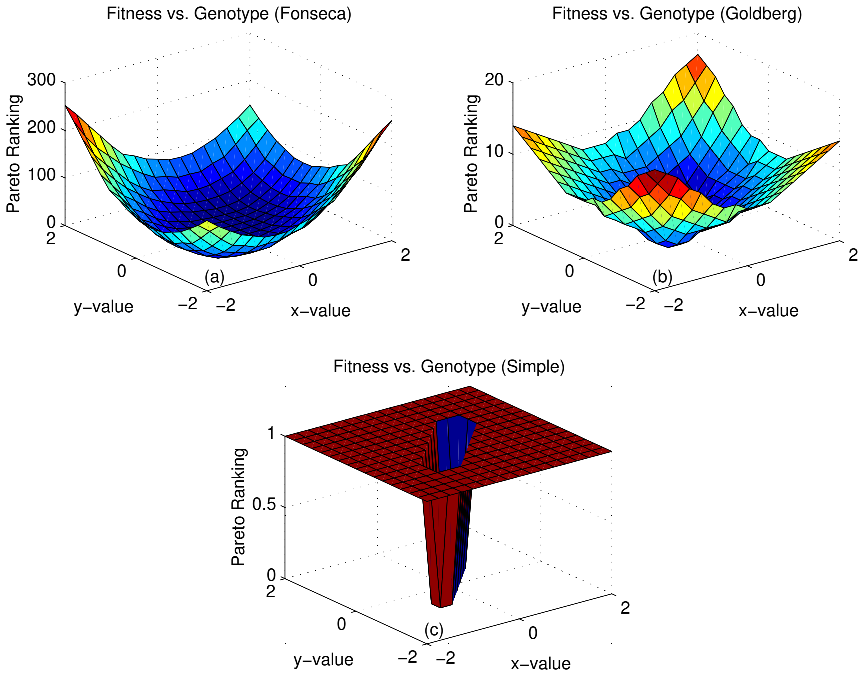


Figure 3.7. Pareto Ranking Schemes

Further clouding the issue is the fact that rank itself is often ***not*** directly used as a solution's fitness. For example, Fonseca and Fleming first used their ranking scheme in an MOEA implementation named the MOGA [108]; Srinivas and Deb were first to implement Goldberg's scheme in the Nondominated Sorting Genetic Algorithm (NSGA) [306]. Both transform assigned rank before selection occurs. The MOGA sorts solutions by rank and assigns fitness via linear or exponential interpolation, while the NSGA uses "dummy" fitness assignment, ensuring only that each "wave" of Pareto optimal solutions has a maximum fitness smaller than the preceding wave's minimum value.[6]

3.3.2.3 Pareto Niching and Fitness Sharing. Several MOEA Pareto niching and fitness sharing variants have been proposed with the same goal as in traditional single-objective optimization – finding and maintaining ***multiple*** optima. However, MOEAs use sharing in an attempt to find a uniform (equidistant) distribution of vectors ***representing*** PF_{true}, i.e., one in which PF_{known}'s shape is a "good" approximation of PF_{true}. We compare selected implementations of this concept.

Fonseca and Fleming's MOGA [114] uses restricted sharing, in the sense that fitness sharing occurs only between solutions with identical Pareto rank. They measure niching distance in phenotypic space, i.e., the distance (over some norm) between two solutions' evaluated fitness vectors is computed and compared to σ_{share} (the key sharing parameter). If the distance is less than σ_{share} the solution's associated niche count is then adjusted. Srinivas and Deb's NSGA [306] implements a slightly different scheme, where distance is measured (over some norm) in genotypic space, i.e., the distance between two solutions is compared to σ_{share}.

Horn and Nafpliotis define niching differently in their MOEA named the Niched Pareto Genetic Algorithm (NPGA) [154], which performs selection via binary Pareto domination tournaments. Solutions are selected if they dominate both the other and some small group (t_{dom}) of randomly selected solutions. However, fitness sharing occurs only in the cases where both solutions are (non)dominated. Each of the two solution's niche counts

[6] The MOGA and NSGA are used in the experiments discussed in Chapter VI. Their algorithmic implementations are further explained there.

is computed not by summing computed sharing values, but by simply counting the number of objective vectors within σ_{share} of their evaluated vectors in phenotype space. The solution with a smaller niche count (i.e., fewer phenotypical neighbors) is then selected. Horn et al. term this *equivalence class sharing* [155].[7]

Another fitness sharing variant uses the NSGA's rank assignment scheme (i.e., Goldberg's [126] Pareto ranking) but uses phenotypic-based sharing [221]; another combines both genotypic and phenotypic distances in determining niche counts [274]. Fitness sharing may also be applied to solutions regardless of rank instead of restricting sharing between equally ranked solutions.

All of these methods require setting explicit values for the key sharing parameter σ_{share}, which can affect both MOEA efficiency and effectiveness. Fitness sharing's performance is also sensitive to the population size N. Assigning appropriate values to σ_{share} is difficult as it usually requires some *a priori* knowledge about the shape and separation of a given problem's niches. However, as phenotypic-based niching attempts to obtain equidistantly spaced vectors along PF_{known}, both Fonseca and Fleming [114] and Horn [154] are able to give guidelines for determining appropriate MOEA σ_{share} values. These values are based on known phenotypical extremes (minimum and maximum) in each objective dimension. Horn also suggests appropriate values for the NPGA's tournament size parameter (t_{dom}).

To determine σ_{share}'s value using Fonseca and Fleming's method, one uses the number of individuals in the population (which implicitly determines the number of niches), scales the known attribute values, and determines the extreme attribute values in each objective dimension. These parameters are then used to derive σ_{share}. Horn's guidelines use the above parameters to define bounds for σ_{share}'s value.

How does one find each objective dimension's extreme values? One suggested approach is by computing objective values using each decision variables' minimum and maximum value. This is not feasible because decision variable extremums may not correspond to attribute extremums; the combinatorics and unknown relationships between different

[7] The NPGA is used in the experiments discussed in Chapter VI. Its algorithmic implementation is further explained there.

decision variable values is an additional factor. Thus, the minimum and maximum values of either the generational or a secondary population may be used. Fonseca and Fleming [114] indicate recomputing σ_{share} at each generation (using current generational extremums) yields good results. We also note that the MOEA's stochastic nature may not preserve these values between generations, i.e., the associated solutions may not survive. Thus, it is better to select objective extremes from the secondary population if one is incorporated in the MOEA. By definition, this population contains each objective dimension's extrema *so far*, ensuring the "ends" of PF_{known} are not lost.

As with the proposed Pareto ranking schemes, there is then no clear evidence as to the benefit(s) of one Pareto niching and sharing variant over another. Nor are experiments reported in the literature comparing key components of these different approaches (e.g., σ_{share} value assignment).

We note the following in regard to the appropriate sharing domain. Horn et al. indicate sharing should be performed in a space we "care more about" [154, 155]. Phenotypic-based sharing does make sense if one is attempting to obtain a "uniform" representation of PF_{true}. On the other hand, Benson and Sayin indicate many OR researchers "care more about" obtaining a "uniform" representation of P_{true} [28], in which case genotypic-based sharing seems appropriate. The end representation goal should drive the sharing domain.

3.3.2.4 Mating Restriction. The idea of restricted mating is not new. Goldberg [126] first mentions its use in single-objective optimization problems to prevent or minimize "low-performance offspring (lethals)." In other words, restricted mating biases how solutions are paired for recombination *in the hopes of* increasing algorithm effectiveness and efficiency. Goldberg presented an example using genotypic-based similarity as the mating criteria. Deb and Goldberg [86] implemented phenotypic-based restricted mating in their GA niching and sharing investigation. We note here these implementations only allow mating between "similar" solutions (over some metric). *Island model* GAs also implement restricted mating but in a geographic sense where solutions mate only with neighbors residing within some restricted topology [46]. It is also noted [61] that other researchers believe restricted mating should allow recombination of *dissimilar* (over some metric) individuals to prevent lethals. However defined, restricted mating is also incorporated within many MOEAs in an attempt to reduce unfit (e.g., non-Pareto optimal) offspring.

For example, Baita et al. [21], and Loughlin and Ranjithan [205], place solutions on a grid and restrict the area within which each solution may mate. Lis and Eiben [201] allow mating only between solutions of different "sexes." Jakob et al. [167] restrict mating to solutions within a particular deme. Hajela and Lin [140] implement a unique form of mating restriction. In their linear fitness combination (weighted-sum) MOEA formulation, they apply restricted mating based on a solution's associated weighting variables to prevent crossover between designs with radically different weight combinations. When considering general MOEAs phenotypic-based restricted mating between similar solutions is of more interest to us. Several MOEA researchers state in their published reports [108, 109, 359]: "Following the common practice of setting $\sigma_{mate} = \sigma_{share}$ ..."

This may be a common practice, but no background is cited in the literature. As σ_{share} attempts to define a region within which all vectors are "related," setting σ_{mate} equal to σ_{share} is intuitive. The same rationale holds in genotypic sharing and mating restriction. We currently have only empirical explanations offered for the implementation (or lack) of restricted mating in various MOEA approaches. In fact, it was recently noted [111] that "... the use of mating restriction in multiobjective EAs does not appear to be widespread." Obviously, some researchers believe restricted mating is necessary or they would not have implemented it, but others indicate it is of no value!

Zitzler and Thiele [359] state that for several different values of σ_{mate}, no improvements were noted in their test problem results (an MOP with two - four objectives) when compared against those with no mating restriction. Shaw and Fleming [297] report the same qualitative results for their application (an MOP with three objectives) whether or not mating restriction was incorporated. Horn et al. [155] offer empirical evidence directly contradicting the basis for mating restriction. They note that recombining solutions whose associated vectors are on different portions of $PF_{known}(t)$ *can* produce offspring whose vectors are on $PF_{known}(t+1)$ but between their parents. They also claim that for a specific MOP a constant (re)generation of vectors through recombination of "dissimilar" parents maintains PF_{known}. They believe most recombinations of solutions in P_{known} also yield solutions in P_{known}.

Thus, as in single-objective optimization, no clear quantitative evidence regarding restricted mating's benefits exists. The empirical evidence presented in the literature can be interpreted as an argument either for or against this type of recombination and leaves the MOEA field in an unsatisfactory predicament. This issue clearly benefits from experiments directly comparing its algorithmic inclusion/exclusion. One must also consider the NFL theorems [346], realizing that mating restriction may not always be effective (or needed) for every problem (class).

3.3.2.5 Solution Stability and Robustness. Both EAs and MOEAs search for some problem's optima. At least for MOPs, it has been noted [160] that P_{true} may not, and often *is not*, the most desirable solution set because its members are "unstable" (e.g., due to engineering tolerances, nonlinear response). It is also suggested that these solutions are often on the "edge" of optimality and/or feasibility. Thus, just as in single-objective optimization, any solutions returned as optimal must be evaluated with respect to any constraints not explicitly considered in the objective function(s). Or, perhaps a suitably defined sensitivity objective (e.g., engineering tolerances) may be incorporated into the MOEA.

3.3.3 MOEA Secondary Populations. We agree with Horn [152] that any practical MOEA implementation must include a secondary population composed of all nondominated solutions found so far ($P_{known}(t)$). This is due to the MOEA's stochastic nature which does not guarantee that desirable solutions, once found, remain in the generational population until MOEA termination. This is analogous to elitism but we stress that it is a *separate* population. The question is then how to best utilize this additional population. Is it simply a repository, continually added to and periodically culled of dominated solutions? Or is it an integrated component of the MOEA? Although several researchers indicate their use of secondary populations only a few explain its use in their implementation. As there is no consensus for its "best" use we present some of its incarnations.

A straightforward implementation stores $P_{current}(t)$ at the end of each MOEA generation (i.e., $P_{current}(t) \cup P_{known}(t-1)$). This set must be periodically culled since a solution's designation as Pareto optimal is *always* dependent upon the set within which it is evaluated. How often the population is updated is generally a matter of choice, but as determination of Pareto optimality is an $\mathcal{O}(n^2)$ algorithm, it should probably not be performed arbitrarily. As this population's size grows comparison time may become significant. This implementation does not feed solutions from $P_{known}(t)$ back into the MOEA's generational population.

Conversely, other published algorithms actively involve P_{known} in MOEA operation. For example, Zitzler and Thiele's [358] Strength Pareto Evolutionary Algorithm (SPEA) stores $P_{current}(t)$ in a secondary population and then culls dominated solutions. Solutions from both the MOEA's generational and secondary populations then participate in binary tournaments selecting the next generation. If the number of solutions in $P_{known}(t)$ exceeds a given maximum, the population is reduced by clustering which attempts to generate a representative solution subset while maintaining the original set's ($P_{known}(t)$'s) characteristics. SPEA also uses $P_{known}(t)$ in computing the main population's solutions' fitness; this effectively results in a larger generational population.

Todd and Sen [319] also insert nondominated solutions from $P_{known}(t)$ into the mating population to maintain diversity, as do Ishibuchi and Murata [163, 165, 164], and Cieniawski et al. [58]. These implementations never reduce the size of $P_{known}(t)$ except when removing dominated solutions. Parks and Miller [249] and Parks [250, 248] implement an *archive* of Pareto optimal solutions. However, solutions in $P_{current}(t)$ are not always archived; the process occurs only if a solution is sufficiently "dissimilar" from those already resident. Thus, this also is clustering. If a new solution is added any archive members no longer Pareto optimal are removed. Like SPEA, the next generation's members are selected from both $P_{known}(t)$ and the current generational population.

Some researchers use secondary populations *not* composed of Pareto optimal solutions. Bhanu and Lee [32] apply an MOEA to adaptive image segmentation; their secondary population is actually a training database from which GA population members are selected. Viennet et al. [334] use separate GAs to optimize each of the MOP's k functions independently; these "additional" populations are later combined and nondominated solutions removed to provide P_{known} .

A secondary population (of some sort) is an MOEA necessity. Because the MOEA is attempting to build up a (discrete) picture of a (possibly continuous) Pareto front, this is probably a case where at least initially, too many solutions are better than too few. It intuitively seems that a secondary population might also be useful in adding diversity to the current generation and in exploring "holes" in the known front, although how to effectively and efficiently use P_{known} in this way is unknown. Again, we suggest experiments directly comparing various secondary population implementations.

3.3.4 MOEA Complexity. It is well known that fitness function evaluation (for many real-world problems) dominates EA execution time. Thus, when discussing various MOEAs' algorithmic complexity we are concerned mainly about the number of fitness evaluations. We do consider solution comparisons and additional calculations, as this overhead is not found in simple GA (SGA) implementations. EVOP complexity is ignored for the current purpose.

MOEA complexity is generally greater than that of SGAs. After fitness evaluation in an SGA, resultant values are stored in memory and no further computation is (normally) required as far as fitness is concerned. However, an MOEA sometimes combines and/or compares these stored values which adds algorithmic complexity. As a reference we present the complexity of the various MOEA techniques in Table 3.3; SGA complexity is included for comparison. Each technique's "worst-case" was used to generate these figures.

The table's notation is as follows. Population size is denoted by n and the number of generations by G. T_f represents fitness computation time (assumed here to be equal for each objective). The number of fitness functions is designated by k and the number of solutions per processor (the Pareto demes case) by m. All table entries are based upon a single generational population, i.e., no secondary populations are used. All techniques are assumed to store a solution's evaluated fitness making selection's computational cost inconsequential. All listed techniques have the identical basic cost of T_fGnk fitness computations. Finally, independent sampling's complexity was computed using several runs

Table 3.3. MOEA Solution Technique Complexity

MOEA Technique	**Computational Complexity**
SGA	$T_f Gn$
Lexicographic	$T_f Gnk + Gn^2 k - Gnk$
Linear Combination	$T_f Gnk + Gnk - Gn$
Multiplicative	$T_f Gnk + Gnk - Gn$
Target Vector	$T_f Gnk + Gk^2 + 2Gk$
Minimax	$T_f Gnk + 3Gnk$
Independent Sampling	$c[T_f Gnk + Gnk - Gn]$
Criterion Selection	$T_f Gnk + Gn$
Aggregation Selection	$T_f Gnk + Gnk - n$
Pareto Rank	$T_f Gnk + Gn^2 k - Gnk$
Pareto Niche and Share	$T_f Gnk + Gn^2 k - Gnk + n^2$
Pareto Demes	$T_f Gnk + G\frac{m^2 k}{n^2} - G\frac{mk}{n} + \frac{m}{n} T_{comm}$
Pareto Elitist	$T_f Gnk + Gn^2 k - Gnk$

of a linear fitness combination technique. Randomly assigned weights (in the fitness functions) were used for the aggregation technique's complexity determination. Table 3.3 shows MOEA techniques explicitly incorporating Pareto concepts are the most computationally expensive; this is due primarily to the $\mathcal{O}(n^2)$ cost of determining which solutions in some set are Pareto optimal.

MOEA storage requirements are problem dependent. Like other EAs these requirements are mandated by the specific data structures used. Required storage increases linearly with the number of fitness functions used, and when a secondary population is brought into play.

We note here that MOEA complexity may be a moot issue in real-world applications. As fitness function evaluation (for many real-world problems) dominates EA execution time, the overhead involved in any of the presented techniques may be miniscule in comparison. If that is the case the complexity issue "goes away" as long as the technique appears effective and efficient.

3.3.5 MOEA Computational "Cost". When practically considered, MOP evaluation cost limits MOEA search. The most "expensive" EA component in many real-world MOPs is the fitness function evaluation. Since all algorithms must eventually terminate the number of fitness evaluations is then often selected as the finite resource expended in search, i.e., the choice is made *a priori* for an EA to execute n fitness evaluations. The "best" solution found is then returned. Assuming solutions are not evaluated more than once (no clones) a total of n points (possible solutions) in the search space are explored.

Now consider a k-objective function. Here, k fitness evaluations are performed for each possible solution (one for each objective). Assuming resources are still limited to n fitness evaluations and that each objective evaluation is equally "expensive", only $\lfloor \frac{n}{k} \rfloor$ points in the search space are now explored. All else held equal, a k-objective optimization problem may then result in a k-fold decrease in search space exploration. Note also that in the context of MOEAs, this implies using the term "fitness function evaluations" to measure computational effort may be somewhat misleading. The term "solution evaluations" is clearer.

This result implies an MOEA may require longer (than a single-objective EA) "wall clock" execution times for good performance. Further search is never guaranteed to return the optimal answer but one wishes as much exploration as possible in the time allowed. This increases the sense of confidence one has found the true, and not a local, optimum.

3.3.6 MOEA Parallelization. We have noted several parallel MOEA implementations [3, 21, 167, 210, 256, 274]. These implementations execute either several MOEAs on different processors (several independent, synchronous runs) or spread an MOEA's population among processors in a demic manner (i.e., a "master-slave" or island model [46]). However, none discuss what other parallel MOEA possibilities exist or what MOEA technique modifications may be required when implemented in parallel.

An obvious first choice for MOEA parallelization is an exact task to processor mapping, but this is not a wise choice. Each identified task in Figure 2.12 (Section 2.5) executes for varying time periods. Additionally, Task 1 executes only once. It is easy to see this proposed mapping's inefficiency. One processor completes its task and then sits idle. The other processors are also unable to operate asynchronously resulting in a much greater idle than calculation time.

The four steps in the execution loop *must* occur sequentially. Mutation cannot operate until recombination finishes. Selection does not (normally) occur until all fitnesses are computed. It is conceivable that the fitness evaluation task can operate on solutions sent immediately after mutation does/does not occur, but the overhead of opening/closing a communication channel between two processors seems prohibitively expensive compared to the minimal computational gains. Additionally, since data required by each task is resident on other processors there is an additional communication overhead associated with this implementation. We thus draw the conclusion that this implementation is not useful. "Pipelining" the algorithm's tasks is also ineffective because it is a special case of the exact task to processor mapping.

Another possibility is a Single Program Multiple Data (SPMD) implementation. One may execute several MOEAs simultaneously on different processors and compare, contrast, and/or combine the reported results. As executing a number of MOEAs sequentially achieves this same result the parallel implementation has obvious speedup. However, we also wish to consider parallelizing innate MOEA tasks.

3.3.6.1 MOEA Decomposition. Affecting the ability to effectively and efficiently parallelize an MOEA is the fact it is inherently sequential. By definition, Task 2 (Figure 2.12 in Section 2.5) in the MOEA case computes k ($k \geq 2$) fitness functions. This task can and has been parallelized.

MOEA fitness function evaluation allows for parallelism by assigning each function's evaluation to different processors, assigning subpopulations for evaluation on different processors, or assigning each individual's evaluation across several processors. These options are shown in Figure 3.8; each is discussed in turn.

Each fitness function's execution time may be radically different. Blindly assigning the entire population and each of the k functions to a different processor may then be imprudent if one fitness evaluation takes many times longer than the others (see Figure 3.8a). One could load balance these fitness computations but the effort expended may not be worthwhile. It is also possible to assign fractions of the population to different processors where identical numbers of individuals are evaluated via identical fitness func-

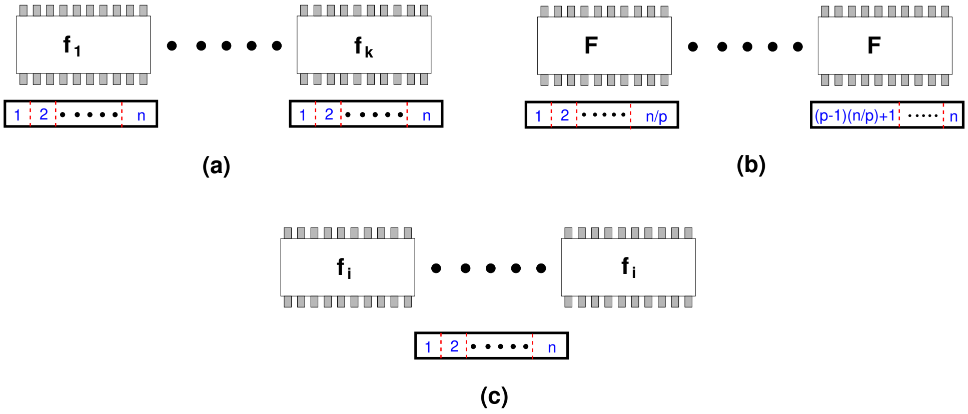


Figure 3.8. Parallel Fitness Evaluation Possibilities

tions (see Figure 3.8b). As long as communication time is not a significant fraction of each subpopulation's calculation time, this is an effective parallelization method for fitness evaluation. Jones et al. [170] use a "master–slave", dynamic load-balancing approach to distribute fitness evaluations in this manner. Finally, in the case of an extremely expensive fitness computation(s) each individual's evaluation(s) could be split among processors (see Figure 3.8c). This is most likely in problem domains such as computational electromagnetics or fluid dynamics where such parallel codes already exist.

Additional processing is sometimes required to transform the resultant fitness *vectors* into scalars. Several variants of MOEA fitness assignment and selection techniques exist (e.g., ordering, scalarization, independent sampling, and cooperative search) which may or may not be parallelizable. For instance, using a Pareto ranking and niching implementation such as Fonseca and Fleming's MOGA [108] permits the Pareto ranking and shared fitness calculations to be performed independently. As each are $\mathcal{O}(n^2)$ algorithms overall MOEA speedup is possible.

Figure 3.9 shows a parallel MOEA's task decomposition. One processor acts as the MOEA "master," executing the population initialization, recombination, mutation, and selection tasks. It also controls parallelization of the fitness evaluation/transformation tasks performed by the "slaves," easily implemented via communication libraries such as the Message Passing Interface (MPI) [247]. MPI includes communication routines that are readily incorporated into MOEA implementations, and are portable across a wide variety of

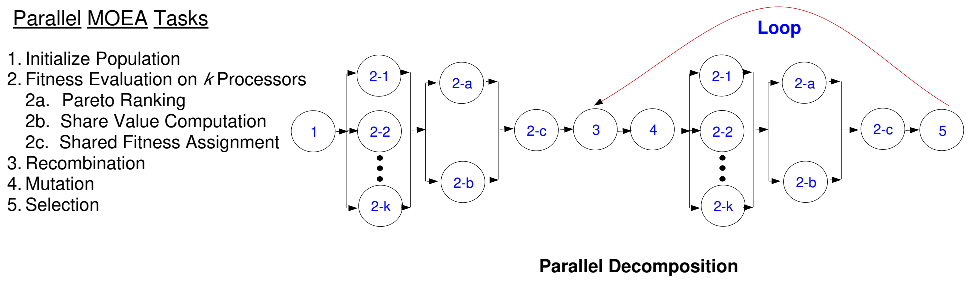


Figure 3.9. Parallel MOEA Task Decomposition

computer architectures with either homogeneous or heterogeneous processors. The master processor may or may not perform fitness calculations depending on the particular problem.

3.3.6.2 Parallel MOEA Issues. Taken as a whole, a parallel MOEA is *not* a complex algorithm. Represented as a Directed Acyclic Graph (DAG) MOEA tasks show more precedence relationships than asynchrony. In other words, the parallel MOEA has a large grain size; algorithmic decomposability is rapidly reaching its limit.

Instantiated parallel MOEAs may well benefit from applying one of the many available static or dynamic processor scheduling and load balancing techniques [97, 185]. As parallel MOEAs are applied to real-world scientific and engineering problems where the fitness calculation time is significant, these scheduling heuristics become more important. However, since the overwhelming amount of many MOEA computational efforts is spent on fitness calculation, parallelizing fitness assignment and selection may not result in large gains. The overhead involved could in fact be "more trouble than it's worth."

In broad terms, any parallel MOEA implementation should result in some speedup gains. Additionally, it offers the possibility of evaluating more candidate solutions perhaps providing a "better" view of the fitness landscape.

3.4 MOEA Design Recommendations

The tables in Sections A.2–A.4 present numerous approaches. When considering them those wishing to implement an MOEA may well be asking, "Where do I begin?" We cannot specify an "all purpose" MOEA technique nor do the NFL theorems [346] allow for one. However, we can suggest MOEAs which appear appropriate as a starting point.

Interested researchers may then select one of these MOEAs to begin their exploration of the MOP domain.

Definition 8 states that an MOP's global optimum is PF_{true}, determined by evaluating each member of P_{true}. Additionally noted in Section A.4 is the fact that many *a posteriori* approaches explicitly seek P_{true}. Thus, *a priori* techniques are not *generally* appropriate because they may not be capable of finding each member of P_{true}, and they return only a single solution per MOEA run. The DM's lack of information before search occurs is also a factor.

Although there are several *a posteriori* techniques to consider[8] we focus on those MOEAs employing Pareto rank- and niche-based selection, and specifically consider Fonseca and Fleming's MOGA [114], the NPGA [154], and the NSGA [306]. The citations give ample information to implement these algorithms.

These algorithms stand out because they incorporate known MOEA theory. The Pareto-based selection each employs explicitly seeks P_{true}. All incorporate niching and fitness sharing in an attempt to uniformly sample PF_{true}. Mating restriction may (or may not) be included in any of the three, as may a secondary population. Finally, their general algorithmic complexity is no higher than other known MOEA techniques.

Although each MOEA's authors (and rightly so) point out deficiencies in their own and other MOEAs, any algorithmic approach is bound to have some shortfalls when applied to certain problem classes (c.f., the NFL theorems [346]). These algorithms' common theme is their respect of known relevant theoretical issues, and their empirical success in both (non-)numeric MOPs and real-world applications. Appendix A shows these algorithms easily win the title "Most Often Imitated," implying other researchers also see value in them. As these MOEAs are used in experiments supporting this research we present detailed information about each in Sections 6.3.2 and 6.3.3. They are briefly described here.

[8]Progressive approaches incorporate either *a priori* or *a posteriori* techniques; any of the algorithms we recommend may be used interactively.

1. **MOGA**. Implemented by Fonseca and Fleming [114]. Used to explore incorporation of DM goals and priorities in the multiobjective search process. Employs the Pareto ranking scheme in Equation 3.1 (Section 3.3.2.2) and fitness sharing.

2. **NPGA**. Implemented by Horn et al. [154]. Used to explore benefits of providing P_{known} as input to a Multi Attribute Utility Analysis [177] process. Uses tournament selection based on Pareto optimality instead of fitness assignment based on Pareto optimality. Incorporates fitness sharing.

3. **NSGA**. Implemented by Srinivas and Deb [306]. Employs Goldberg's Pareto ranking [126] as shown in Figure 3.6 (Section 3.3.2.2). This MOEA attempts to prevent bias towards certain regions of the Pareto front and incorporates fitness sharing.

Although not straightforward, many existing EA implementations are extendable into the MOEA domain. For example, GENOCOP III [217] was readily modified to incorporate both a specialized problem domain code and linear fitness combination technique. The Genetic and Evolutionary Algorithm Toolbox (GEATbx) for use with *MATLAB*[9] [255] allowed us to quickly create both MOGA and NSGA variants; these codes are now being incorporated into the toolbox's baseline version. Upon request, other researchers have also provided their MOEA code for experimentation. Thus, initial algorithmic development should not be a barrier to solving MOPs with MOEAs.

3.5 MOEA Research Contributions

This chapter's analysis and the cataloged research in Appendix A provide a pool from which to award "MOEA Oscars" for significant and original MOEA research contributions. These awards are (of course) subjective.

Schaffer and Fourman must be recognized for their pioneering MOEA work [289, 117]. Figure 3.1 (Section 3.2.1) shows very few MOEA publications during the next six years. Goldberg deserves mention for noticing that the concept of Pareto optimality might be used to rank solutions in MOEAs [126:pg. 201]. As Deb notes [84], varying MOEA

[9] *MATLAB* is a Trademark of The MathWorks, Inc.

interpretations and implementations of Goldberg's "10-line sketch" have proved at least equal to classical approaches in many cases.

Fonseca and Fleming were the first to publish an MOEA research survey [111]. They broadly classified and critiqued known approaches presenting a solid explanation of key MOEA theoretical aspects (e.g., fitness assignment and sharing). This survey and Goldberg's book [126] are probably the most cited documents in MOEA publications. Their MOGA was one of the first Pareto-based MOEAs explicitly used to seek PF_{true} and the first to mention active DM involvement. Horn later published an updated survey [152] with a different classification structure recognizing that many implemented MOEA techniques originated in the OR field. His and Nafpliotis' NPGA [154], and Srinivas' and Deb's NSGA [306] are two other Pareto-based MOEAs built on solid theoretical results. We note our MOEA classification and technique analysis (see Appendix A) is generally more complete and up-to-date than these other surveys.

Finally, awards must be given for MOEA theory development. Three researchers deserve mention here. Rudolph brings a rigorous mathematical approach to the important issue of MOEA convergence [275, 276]. Deb realizes the lack of capability to construct MOPs with desired characteristics and analytical solutions for PF_{true} [83]. Finally, we also recognize the need for additional MOEA theory, a substantiated MOEA test function suite, and a methodology with which to quantitatively compare MOEA performance [327] (also see Chapters II, V and VI).

3.6 Summary

This chapter presents an in-depth analysis of MOEA research, discussing in detail several foundational issues such as implemented MOEA techniques and fitness functions, chromosomal representations, and application areas. More general observations are also made concerning MOEA characteristics and components. Theoretical issues relating to MOEA complexity and parallelization are discussed.

This analysis identifies appropriate MOEAs recommended for initial use in solving MOPs, and should be used when re-engineering these (or any other) MOEAs to solve particular MOPs. The chapter concludes by highlighting several significant MOEA research contributions. As a whole, this analysis and Appendix A serve as a guide to MOEA design. With this background and insight into the MOEA design process a new algorithm design is discussed and implemented in the next chapter.

IV. Building Blocks and MOEA Design

> A good scientist is a person with original ideas. A good engineer is a person who makes a design that works with as few original ideas as possible. There are no prima donnas in engineering.
>
> **Freeman Dyson**, *Disturbing the Universe*

4.1 Introduction

A primary thesis of this research is that Building Blocks (BBs) can be successfully employed in solving MOPs. This view was until now unexplored. In keeping with the above quote we first review relevant single-objective BB concepts and then extend appropriate ones to the MOP domain. Based on these results, an existing single-objective GA which explicitly manipulates BBs is made the basis for a new, innovative MOEA.

Section 4.2 gives an overview of BB concepts. A brief history of BB-based GAs is presented in Section 4.3 and Section 4.4 discusses the relationship between BBs and MOPs. Finally, Section 4.5 presents a "new" multiobjective EA (called the *MOMGA*) based on the explicit BB manipulation performed by the messy GA. Section 4.6 proposes a concurrent MOMGA implementation.

4.2 GA Building Block Overview

Theoretical GA performance analysis makes extensive use of *schemata* (singular: schema), or similarity templates.[1] A schema is a character string; its characters are drawn from some specified genetic alphabet also containing a "don't care" character (*). Since solutions are encoded as strings a schema thereby describes a subset of potential solutions. For example, the schema `1**` represents the set of all 3-bit binary strings containing a `1` in the first position, i.e., `1**` = {100, 101, 110, 111}. Likewise, the schema `1*0` represents the set of all 3-bit binary strings beginning with a `1` and ending with a `0`, i.e., `1*0` = {100,110}.

[1] This overview makes use of the concepts presented in Section 2.4.

The *defining length* ($\delta(H)$) of a schema H is the "distance" between the index of the first and last specified positions. For example, $\delta(\texttt{1*****0*}) = 7-1 = 6$ and $\delta(\texttt{1*******}) = 1-1 = 0$. The *order* ($o(H)$) of a schema H is the number of its specified positions. For example, $o(\texttt{1*******}) = 1$ and $o(\texttt{11111111}) = 8$.

These concepts provide a basis for discussing the *Fundamental Theorem of Genetic Algorithms*, also known as the *Schema Theorem.* Defining the average fitness of the strings matching some schema H as $f(H)$, the average population fitness as $\bar{f}$, and the number of strings matching the schema contained in a population at time t as $m(H,t)$, the reproduction operator's effect (assuming fitness proportional selection) is

$$m(H,t+1) = m(H,t)\frac{f(H)}{\bar{f}} \; . \tag{4.1}$$

Two types of EVOPs can disrupt schema present in the population as the GA executes. Single-point crossover disrupts a schema only when the crossover point occurs within the defining length of the schema. Thus, the probability of survival under single-point crossover for some schema in a string of length l is

$$p_s \geq 1 - p_c\frac{\delta(H)}{l-1} \; , \tag{4.2}$$

where p_c is the probability of crossover. The inequality reflects the fact that crossover may not actually disrupt the schema even when the crossover point is within the defining length.

Point mutation also disrupts a schema only when occurring within the schema's defining length. The probability of survival for the same schema under the point mutation operator is

$$p_{ms} \approx 1 - o(H)p_m, \; p_m \ll 1 \; , \tag{4.3}$$

where p_m is the probability of mutation. Combining these results and omitting negligible terms gives an estimate for the expected number of schema remaining in the next generation:

$$m(H, t+1) \geq m(H, t) \frac{f(H)}{\bar{f}} \left[1 - p_c \frac{\delta(H)}{l-1} - o(H) p_m \right] \tag{4.4}$$

Goldberg states this result implies "short, low-order, above-average schemata receive exponentially increasing trials in subsequent generations" [126]. These highly fit schemata are also referred to as BBs. Goldberg also postulates a *Building Block Hypothesis*:

> Short, low-order, and highly fit schemata are sampled, recombined, and resampled to form strings of potentially higher fitness.

BB concepts are also valid when viewed in light of real-valued EAs. Instead of an l-bit binary string assume m real-valued parameters. Given that parameter p_i's range is $l_i \leq p_i \leq u_i$, and thus a parameter space $\mathcal{P}$ defined by $\mathcal{P} = \prod_{i=1}^{m} [l_i, u_i]$, Wright defines a real-valued schema $\mathcal{S}$ by:

$$\mathcal{S} \in [\alpha_i, \beta_i] \ , \tag{4.5}$$

where $l_i \leq \alpha_i \leq \beta_i \leq u_i$ [347]. He then shows the Schema Theorem also holds when real-valued BBs are considered.

In general, EAs implicitly focus search around BBs whose role is defined by the Schema Theorem. BBs are a GA's "information" source where each has two major components: each bit (or parameter) has some specified value(s) and each is somehow dependent upon the others (*linkage*). BBs define chromosomes associated with high fitness and are used by GAs in three primary ways. First, an initial supply of BBs is provided via the starting population and used in defining individual fitness. Second, selection should cause an increase in the number of desired BBs by selecting individuals containing them for inclusion in the next generation. Finally, BBs are mixed via recombination, attempting to assemble the best BBs into a single individual. In each generation, GAs use these two EVOPs in an attempt to exploit the BBs present in their populations.[2]

4.3 Building Block-Based GAs

If true, the Building Block hypothesis means BBs are an important GA component. Few research efforts take steps to *explicitly* incorporate BB manipulation into GA operation. When considering ones that do, Goldberg et al.'s [130, 129] messy GA (mGA) and fast messy GA (fmGA) [128] are of special interest here.

4.3.1 mGA and fmGA. Goldberg et al. [130] believe too much attention is paid to "neat" GA genotype codings. They propose a coding scheme where genotypes can exhibit redundancy, over- and under- specification, and changing structure and length. They believe this GA modification forms tighter and more useful BBs than those formed by standard GAs. The resultant mGA proved successful in optimizing *deceptive functions*; these functions mislead GA search toward some local optimum when the global optimum actually lies elsewhere [130]. The mGA's pseudocode is shown in Figure 4.1.

As shown, the mGA initializes a population of BBs via a deterministic process called Partially Enumerative Initialization (PEI), producing all possible BBs of a specified size. This population size is governed by the equation

$$N = C^k \binom{l}{k} , \tag{4.6}$$

where N is the resulting population's size, C the allelic alphabet's cardinality, l the chromosomal length (in bits), and k the problems assumed BB size. Thus, for a 240-bit (binary) chromosome with $k = 3$, the initial population size is $18,202,240$. It is easily seen that population size grows exponentially with increasing k. These BBs' fitness is evaluated with respect to a *competitive template* used to fill in values of under-specified positions.

[2]Based upon a mutation probability much less than one, and a BB of any size, it is highly unlikely for BBs to be constructed via mutation. Mutation is thus considered more of an exploratory EVOP.

```
For n = 1 to k
    Perform Partially Enumerative Initialization
    Evaluate Each Population Member's Fitness (w.r.t. Template)
// Primordial Phase
    For i = 1 to Maximum Number of Primordial Generations
        Perform Tournament Thresholding Selection
        If (Appropriate Number of Generations Accomplished)
            Then Reduce Population Size
        End If
    End Loop
//Juxtapositional Phase
    For i = 1 to Maximum Number of Juxtapositional Generations
        Cut-and-Splice
        Evaluate Each Population Member's Fitness (w.r.t. Template)
        Perform Tournament Thresholding Selection
    End Loop
Update Competitive Template
End Loop
```

Figure 4.1. mGA Pseudocode

Following PEI is the *primordial phase* which contains several cycles of population growth and reduction. Next is the *juxtapositional phase* where the mGA operates on the BB population by cloning desired BBs, then recombining and selecting resulting strings with high fitness (again, with respect to the competitive template). The specialized recombination operator (called *cut-and-splice*) operates on uneven length strings. Taken together, PEI and these two phases form an *era*. The mGA executes for a user-specified k eras, returning a solution which is then optimal with respect to BB size (k) and the competitive template.

The mGA is a computationally expensive algorithm due to PEI. The fmGA is then proposed to reduce mGA complexity via probabilistic initialization schemes. The fmGA operates identically to the mGA in the juxtapositional phase. However, instead of PEI, it uses a probabilistic BB initialization technique creating a controlled number of BB clones of specified size. These BBs are then filtered, ensuring that (in a probabilistic sense) all desired BBs exist in the initial population. Goldberg et al. claim this variant is as effective as the mGA but without the initialization bottleneck caused by PEI [128].

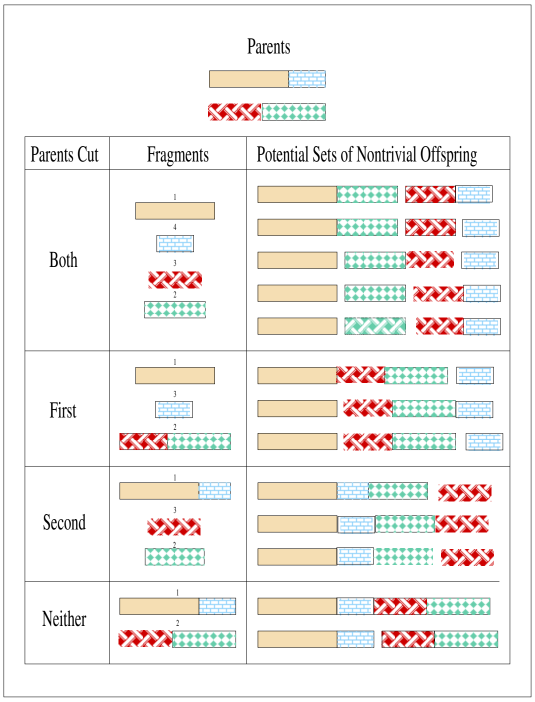


Figure 4.2. Potential "Cut and Splice" Nontrivial Offspring

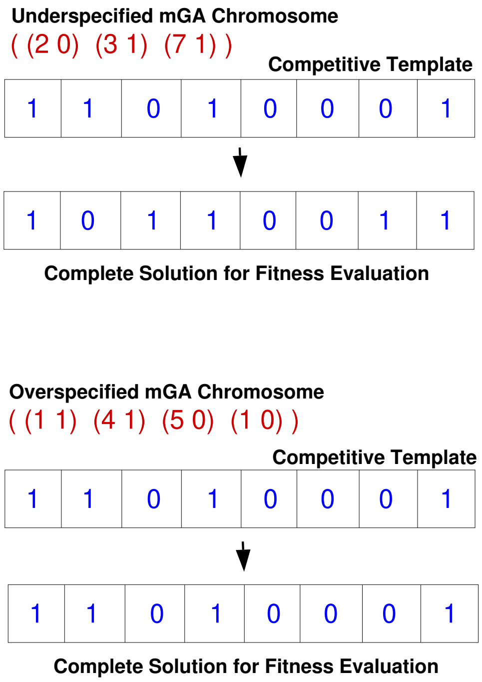


Figure 4.3. Template Fitness Examples

4.3.2 Related Building Block GAs. Several other BB-based GAs are proposed in the literature; other researchers classify them as *linkage investigating* GAs as they are specifically designed to find and propagate "tightly-linked" genes, or BBs [88]. Table 4.1 lists other BB-based GAs briefly describing what differentiates each.

Two items are of note here. First, we consider the mGA, fmGA, and gmGA as "Top-Down" approaches; the others are considered "Bottom-Up". They are classified in this fashion because of the different manner in which the algorithms attempt to determine appropriate BBs. The mGA, fmGA, and gmGA explicitly construct BBs in the initial population while the others use (or modify) randomly initialized individuals. Second, all GAs in Table 4.1 except Population-Based Incremental Learning and the Selfish Gene GA are (or are based on) work by Goldberg and his students.

4.3.3 Building Block Observations. We note that the Building Block hypothesis has never been *proved*, and may never be, although it is generally accepted to hold for cer-

Table 4.1. Building Block GAs

GA	**Brief Description**
compact GA (cGA) [146]	Virtual population; Changing probability vector determines convergence
Extended Compact GA (ECGA) [144]	Concrete population; Convergence determined by population marginal probability model
fast messy GA (fmGA) [128]	Probabilistic initialization of size k building blocks
Gene Expression Messy Genetic Algorithm (GEMGA) [172]	Completely specified chromosomes; Randomly generated initial population's size determines processed linkage size; 2 phases: *Transcription* attempts to determine linkage and *RecombinationExpression* compares/modifies the linkages of and then recombines two chromosomes
Generalized messy GA (gmGA) [214]	Replaces discrete-valued selection and filtering threshold parameters with real-valued; Probability distributions then incorporated
Linkage Learning GA (LLGA) [145]	Overspecified chromosomes; Number of introns determines processed linkage size; 2 phases: *Selection* and *Exchange*, which performs 2-point crossover, removing redundant genes from children
messy GA (mGA) [130, 129]	All size k building blocks explicitly generated
Population-Based Incremental Learning (PBIL) [22]	Incorporates hill-climbing; Changing probability vector determines convergence
Selfish Gene GA (SGGA) [72, 71]	Virtual population modeled by marginal probability vectors; Changing probability vector determines convergence

tain cases and not for others. As any EA executes, each generation's underlying probability density functions are unknown thus making such a proof difficult. Additionally, successful BB use critically depends on the EA representation's degree of *linear separability* (decomposition of the overall problem into subproblems) [264]. By definition, if a representation is not linearly separable it suffers from *epistatic* effects (epistasis is a term describing gene interrelationships). Standard EAs can cope with some degree and types of epistasis, but since exact epistatic relationships are most often unknown Goldberg's hypothesis may or may not hold in any given situation.

These "negatives" have not prevented successful EA applications based on explicit BB manipulation. For example, the mGA and fmGA are used in practical single-objective applications [95, 121, 215]. Deb also implemented a floating point mGA version that achieved good results on a numeric and cylinder design problem [82]. When considered at a meta-level, standard EAs (which are predicated upon BBs) often perform much better than random search, implying their use of BBs and problem domain knowledge is responsible for their effectiveness and/or efficiency. Thus, it appears that BB concepts are useful in *some* problem solving situations. With this background, we now focus on the explicit use of BBs when solving MOPs.

4.4 MOPs and Building Blocks

Conjecture 1: Appropriately defined building blocks can be sampled, recombined, and resampled to form "better" MOP solutions. □

This research attempts to determine Conjecture 1's validity. The preceding discussions support the practical usefulness of BBs. Although their effectiveness is not yet theoretically quantified they can be employed in MOEAs regardless of chromosome encoding. We wish to extend BB concepts successfully applied to single-objective optimization problems into the MOP domain, and use existing analogous ideas in search of more effective and efficient MOEAs.

The Schema Theorem has historically been developed, described, and analyzed in terms of single-objective functions. However, BB concepts remain applicable when extended to MOPs. We first note that BBs are not structurally modified by the simultaneous optimization of two or more functions. To illustrate, assume a binary-valued genotype of length l containing several BBs. Single-objective optimization maps this genotype to a single value; this is the genotype's associated fitness or phenotype. In MOPs the same genotype maps to a *multi*-valued fitness vector. However, *the genotype's structure and its BBs have not changed in any way!* It's simply that multiple fitness functions have been evaluated with respect to a single genotype.

Single-objective optimization attempts to find a genotype(s) mapping to "high" fitness; MOPs attempt the same. While single-objective optimization algorithms generally search for a (possibly) unique single solution, MOEAs often focus on a *set* of Pareto optimal solutions which may well have very *dissimilar* desired BBs! Thus, "good" MOP BBs should help drive search towards solutions in P_{true}.

As indicated in Section 2.2.1, P_{true} defines the MOP's trade-off surface from which some DM implicitly indicates acceptable solutions. These solutions may have no clearly apparent relationship besides their membership in the Pareto optimal set. In fact, BBs which are "good" for some solution(s) in P_{true} may be "not good" for an arbitrarily chosen other (or subset). Taking Fonseca's 2^{nd} MOP [109] as an example illustrates this phenomenon.

Minimize $F = (f_1(\vec{x}), f_2(\vec{x}))$, where

$$\begin{aligned} f_1(\vec{x}) &= 1 - \exp(-\sum_{i=1}^{n}(x_i - \frac{1}{\sqrt{n}})^2), \\ f_2(\vec{x}) &= 1 - \exp(-\sum_{i=1}^{n}(x_i + \frac{1}{\sqrt{n}})^2) \ , \end{aligned} \tag{4.7}$$

where $-4 \leq x_i \leq 4$ and $n = 2$.

Figure 4.4 shows a representation of this MOP's P_{true}, and Figure 4.5 its PF_{true} (indicated by the '$*$' symbols; dominated vectors are represented by '.').[3] When analyzing Figure 4.4 it is easily seen that when taken overall, some relationship (structure) exists between the Pareto optimal solutions.

For further insight, assume the following real-valued BBs:

$$\begin{aligned} BB_1 &= ([-0.7, -0.5] \ , [-0.7, -0.5]), \\ BB_2 &= ([0.5, 0.7] \ , [0.5, 0.7]). \end{aligned} \tag{4.8}$$

Figure 4.6 plots all solutions containing BB_1 as '+'s and BB_2 as '$\circ$'s. Figure 4.7 plots their associated vectors using the same symbols. As easily seen, solutions in the lower-left hand

[3]Figures 4.4 and 4.5 are deterministically derived; Pareto representations may slightly change when computational resolution is increased/decreased.

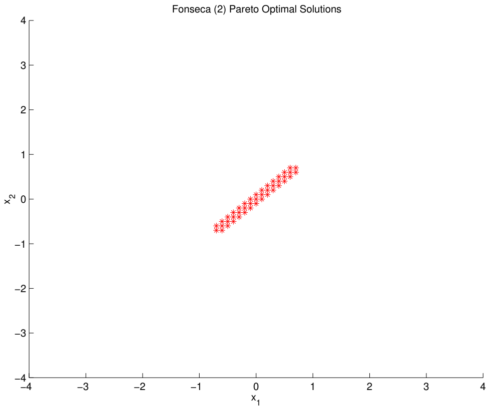


Figure 4.4. Fonseca (2) P_{true}

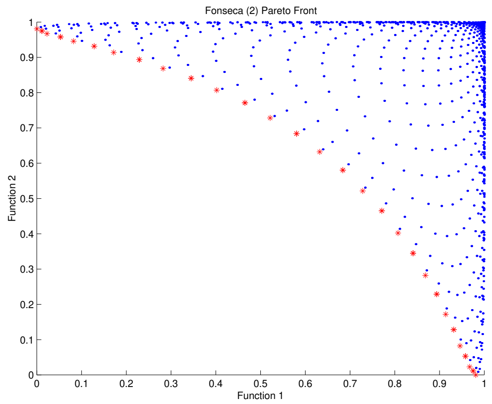


Figure 4.5. Fonseca (2) PF_{true}

corner of Figure 4.6, although closely related to others in their immediate neighborhood (i.e., they all contain BB_1), are different from solutions in the upper-right hand corner that contain BB_2. Figure 4.7 shows that in this case, the different BBs map to very different portions of objective space yet both are equally important! We wish to use BB concepts to gain insight into solving MOPs with MOEAs.

Figure 4.6. Solutions Containing BB_1 and BB_2

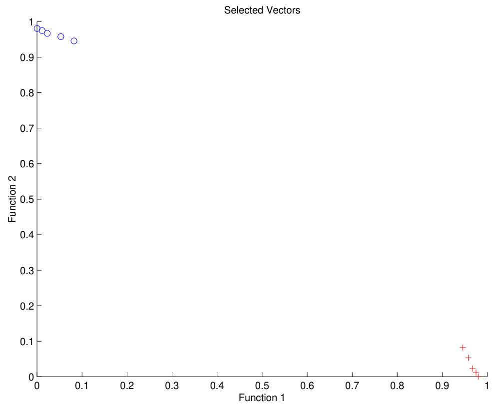


Figure 4.7. Corresponding PF_{true} Vectors

MOEAs are able to use BBs "good" for only some solutions in P_{true} , evidenced by the successful approaches predicated on sharing, crowding, and niching techniques [107, 152, 326] (also discussed in Section 3.3.2.3). Albeit in more abstract terms, other MOEA researchers also believe that MOP-BB issues are significant. Several MOEAs implement some form of mating restriction where analogous to the σ_{share} term used in computing shared fitness, a parameter is used describing the "distance" (over some norm) within which solutions (or their evaluated vectors) must reside in order to recombine (see Section 3.3.2.4). But this is nothing other than what was just stated – it suggests researchers suspect that BBs "good" for one solution in P_{known} are possibly "not good" for others!

4.4.1 Building Block Deception. Some mGA theoretical results must be viewed in a different light when extended to the MOP domain. For example, the mGA is predicated upon *a priori* definition of the suspected highest order deceptive nonlinearity present in the problem being solved [130]. Thus, if we assume a given problem is *order-k* deceptive, all order-1, order-2, ..., order-$(k-1)$ schemata direct GA search away from the *global* optimum to a deceptive one [342]. This implies all lower-order schema (i.e., order $< k$) contained as special cases of an order-k schemata have different bit-values. These types of problems are termed *deceptive*. The following theorems (similar to single-objective optimization results) help bound k's value when considering possibly deceptive MOPs.

Theorem 5: The orders of deception for the functions composing an MOP are not necessarily equal. □

Proof: Existence proof. Assume an MOP composed of two functions – one is Whitley's Deceptive Function 1 (a fully deceptive order-3 function), and the other is Whitley's Deceptive Function 2 (a fully deceptive order-4 function) [342]. *Q.E.D.*

Theorem 6: An MOP's order of deception is at worst l, where l is the number of bits encoding the chromosome. □

Proof: Existence proof. Assume an MOP, of which one function is Whitley's Deceptive Function 2 [342]. This is a fully deceptive function of order l. *Q.E.D.*

These results imply that when optimizing an MOP using a method where order is a required parameter, one must choose a value equal to or greater than the highest order of deceptiveness present in any function contained in the MOP. Selecting smaller values may well prevent discovery of solutions in P_{true} because mutation would be the only EVOP allowing discovery of the global schema. Theorem 6 also shows that an MOP's order of deception may in the worst case be l, as it may also be for single-objective optimization problems.

4.5 The Multiobjective mGA (MOMGA)

Having laid a foundation for understanding BB concepts and their use in evolutionary search, we now wish to explore the relationship between MOEA BBs by extending an existing single-objective BB-based GA to the MOEA domain. As no other known MOEA considers this approach this new algorithm is a "state of the art" contribution.

The mGA [130] is initially considered as a vehicle with which to define and investigate MOP BBs. We select this algorithm for several reasons, although primarily because its population initially contains every possible BB of a specified size corresponding to (a subset of) solutions in P_{true}. Additionally, it is designed to explicitly manipulate appropriate BBs in order to arrive at an optimal solution(s), its source code is freely available, its operation well understood, and its structure modifiable to solve MOPs. We discuss only mGA features modified in producing the MOMGA. The reader is directed to Goldberg et al.'s original papers [129, 130] for a more detailed discussion of basic mGA operation and theory. We extend the algorithm and associated theory in the following sections.

4.5.1 The mGA, MOMGA, and Fitness Functions. The mGA requires a fitness function defined over some l-bit string, each $l_i \in \{0, 1\}$. The MOMGA uses no subfunctions; fitness functions operate on the entire l-bit string. This is to focus solely on the BBs used in MOP solutions and to prevent problems determining the relationship(s) between subfunctions and a complete MOP (if any). As previously discussed, the number of simultaneously optimized fitness functions does not affect the genotype. Thus, the MOMGA evaluates each of k user-defined fitness functions taking an identical l-bit string as input.

4.5.2 The mGA, MOMGA, and Solution Evaluations. The mGA initially constructs every possible BB of a user-specified size k, resulting in a primordial population of size $2^k\binom{l}{k}$. Each of these strings requires only a *single* function evaluation. The result is then stored and used repeatedly during the primordial phase. Likewise in the juxtapositional phase, only one function evaluation is computed per generated individual string for a total of C evaluations. Thus, the mGA's algorithmic complexity is of order $\mathcal{O}(2^k\binom{l}{k}+C)$. Given identical string length, allelic cardinality, and order of deception (k), the MOMGA's population size is identical but it does require an increased number of function evaluations. However, the number of function evaluations then increases at a "manageable" (linear) rate as the MOMGA's complexity in solving a p-objective MOP is of order $\mathcal{O}(p2^k\binom{l}{k} + pC)$.[4] However, "manageable" is a subjective term. Real-world scientific and engineering MOPs often use computationally complex and time-consuming fitness calculations which may impact the use of multiple objective functions. In addition, as l and/or k grow, both the mGA's and MOMGA's complexity is of order $\mathcal{O}(2^k\binom{l}{k})$. This indicates these algorithms have a computational bottleneck due to PEI, and in fact is a primary reason for developing alternative BB-based GAs (see Section 4.3.2).

The MOMGA's storage requirements also increase linearly. Where the mGA stores a single value from each function evaluation the MOMGA stores a vector whose number of values corresponds to the number of functions being optimized.

4.5.3 The mGA, MOMGA and EVOPs. The mGA incorporates tournament selection which effectively combines selection and fitness scaling [130, 19:pg. C2.3:1]. This is implemented by choosing q solutions at random ($q \geq 2$) and selecting the solution with highest fitness for inclusion in the next generation. That solution is also removed from the selection pool. The process is repeated until the population is filled. The mGA was constructed using this selection operator because it is easily implemented and gives desirable expected performance [130]. Also, $q = 2$ is originally selected in the mGA (and is a common parameter setting) as it results in "medium" selective pressure [17:pp. 174-180]. We also select $q = 2$ in the MOMGA.

Pareto-based tournament selection (among others) has been successfully used in solving MOPs [152]. Comparing vectors based on dominance is a way of finding the "best avail-

[4]The variable p is used here to prevent confusion between the number of objectives and BB length.

able" MOP solutions. If nondominated vectors are the search target it only makes sense to use nondominance as the comparison criteria. Thus, we select *currently* Pareto optimal BBs/solutions ($P_{current}(t)$) for further processing by making the MOMGA's tournament selection operator dominance- rather than fitness-based.

However, the MOMGA implements a modified tournament selection operator directly based on the NPGA's selection scheme [154]. The NPGA randomly selects two solutions for a tournament, but also chooses a comparison set (t_{dom}) of other individuals. Each of the two candidates are compared (using Pareto dominance) against each comparison set member. If one candidate is nondominated and the other is not (with respect to the comparison set), it is selected for reproduction. If neither or both are dominated sharing is implemented. Horn et al. found that a binary tournament alone produced insufficient domination pressure resulting in poor PF_{known} representations. They then introduced the comparison set to control what they call *domination pressure*, also giving suggested values (based on empirical observation) for this parameter [155].

We show in Appendix A that many Pareto-based MOEAs employ explicit niching and fitness sharing to track several genotypes (corresponding to varied phenotype performance) at once. Sharing is also common in multimodal single-objective optimization problems, where it attempts to prevent concentration on and then loss of an optimum (a situation termed *genetic drift*). Horn et al. implement such a scheme in their NPGA [154]; we employ an identical procedure in the MOMGA.

As described above, two randomly selected candidates are compared (using Pareto dominance) against each solution in a comparison set. If neither or both's associated vectors are dominated, sharing occurs by determining the number of known vectors (the *niche count*) within some phenotypical niche radius (σ_{share}) of the two candidates. The candidate with the smaller niche count is then selected. Horn terms this *equivalence class sharing* because these solutions can be considered "equally" fit [154]. Several other niching techniques do exist, e.g., preselection, crowding, and immune system models [153]. Engineering the MOMGA to employ the NPGA-niching scheme seemed the best choice given that it already employed tournament selection.

The mGA may employ both mutation and recombination (via "cut and splice"). The MOMGA makes no changes to these EVOPs' operation. One last crucial component of successful mGA operation is the *competitive template.* Engineering this concept for MOPs is by no means straightforward.

4.5.4 The mGA, MOMGA, and Competitive Templates. The mGA uses a competitive template in both the primordial and juxtapositional phases. The primordial phase evaluates all BBs with respect to the template; the juxtapositional phase uses the template to evaluate fitness of the recombined BBs. The competitive template's purpose is to separate the value of some bit combination from an entire string without using prior functional knowledge. Thus, each partial string's assigned fitness is actually a *template fitness* where unassigned loci values are filled with the corresponding template values. Although using competitive templates allows for consistent evaluation of partial strings, a given template optimizes only one solution (itself) with respect to the available BBs.

The mGA uses templates locally optimal to the previous era. A randomly generated template is used to find the locally optimal template for era 1, the resulting "best" answer (at era 1's end) is used to find the locally optimal template for era 2, etc. The competitive template is changed by identifying the string with the highest template fitness value yet achieved; its values are substituted into the current template. This competitive template then represents the best *total* solution yet known to the mGA and it is here we arrive at the crux of the matter.

Traditional mGA search is concerned with finding a *single* ("best") answer. The competitive template limits mGA search and is thus critical to finding an *optimum* and not just an *optimal* solution. As MOPs offer a *set* of solutions the problem is how to extend the template concept in order to provide that desired set. An easy answer of using a template for each solution in P_{true} or P_{known} is not feasible. This implies determining a number of solutions *a priori* when neither of these sets' representable cardinalities is known. Furthermore, how should these (possibly quite numerous) templates be employed in the MOMGA? Combinatorial computational considerations are easily seen. Thus, the following strategy is employed.

During its primordial and juxtapositional phases the MOMGA uses a different competitive template associated with each objective function being optimized. Each time a partial string's template fitness vector is computed a random template is selected from the k available. At the end of each era the values of the "best" solution *for each objective* replace corresponding values in the respective current template. We realize that mGA competitive templates are criticized for being locally optimal [82], as is this VEGA-like approach (VEGA selection may result in strong "speciation" [107, 306]). We again note our initial focus is determining the use and role of BBs in forming MOP solutions.

4.6 MOMGA v1.0

A diagram showing our MOMGA implementation is presented in Figure 4.8. We performed all mGA modifications discussed in Sections 4.5.1 through 4.5.4, along with adding and maintaining a *secondary* solution population (P_{known}). The MOMGA pseudocode is shown in Figure 4.9.

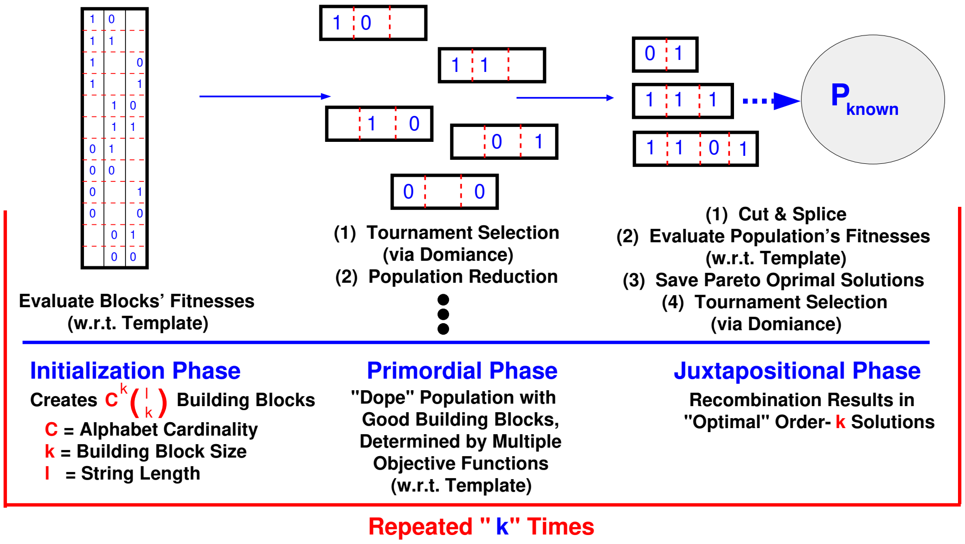


Figure 4.8. MOMGA Operation

In each era, after each juxtapositional generation, we add $P_{current}(t)$ to $P_{known}(t-1)$ (i.e., $P_{current}(t) \cup P_{known}(t-1)$). Because a solution's classification as Pareto optimal is dependent upon the context within which it is evaluated (i.e., some *current* solution set), at MOMGA termination all solutions of P_{known} are tested and those whose associated vectors

```
For n = 1 to k
    Perform Partially Enumerative Initialization
    Evaluate Each Population Member's Fitness (w.r.t. k Templates)
// Primordial Phase
    For i = 1 to Maximum Number of Primordial Generations
        Perform Tournament Thresholding Selection
        If (Appropriate Number of Generations Accomplished)
            Then Reduce Population Size
        Endif
    End Loop
//Juxtapositional Phase
    For i = 1 to Maximum Number of Juxtapositional Generations
        Cut-and-Splice
        Evaluate Each Population Member's Fitness (w.r.t. k Templates)
        Perform Tournament Thresholding Selection and Fitness Sharing
        P_known(t) = P_current(t) ∪ P_known(t − 1)
    End Loop
Update k Competitive Templates (Using Best Value Known in Each Objective)
End Loop
```

Figure 4.9. MOMGA Pseudocode

are dominated removed. Solution culling is performed at this time so as to not unnecessarily slow MOMGA execution by the $\mathcal{O}(n^2)$ complexity of dominance determination.

4.6.1 Concurrent MOMGA (cMOMGA). Parallelizing the MOMGA may lead to improved efficiency. Combining results (P_{known}) of several simultaneously executing MOMGA runs is perhaps the simplest parallel implementation. However, another possibility may be considered.

MOMGA templates are locally optimal, i.e., they focus search toward portions of the search space. Thus, instantiating several independent MOMGA runs all solving the same MOP initially focuses search in different (and more) portions of the search space. Allowing BB communication between MOMGA instantiations may then improve overall performance. As previously noted, BBs good for some (sub)set of Pareto optimal solutions may be bad for another. Ordering MOMGA runs in some manner then implies two consecutively ordered runs are searching spaces "closer" together than any other two. A cMOMGA version then shares BBs between these consecutive MOMGA instantiations in an attempt to increase performance. This process is illustrated in Figure 4.10. At termination, each particular MOMGA's P_{known} is combined, its associated vectors checked for domination, and a final P_{known} reported.

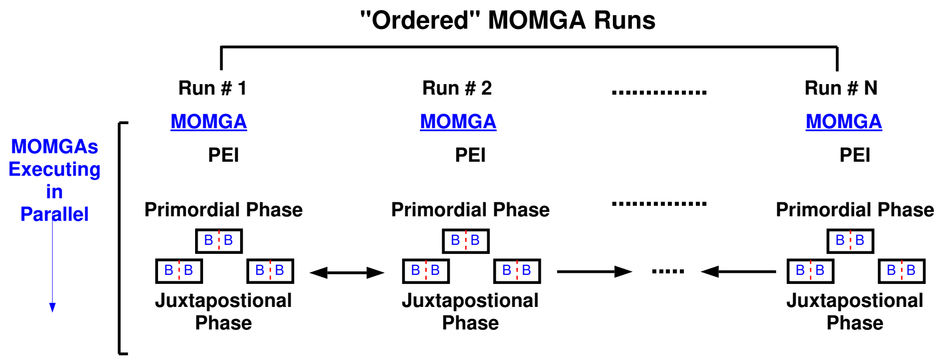


Figure 4.10. Proposed cMOMGA Operation

On the surface this approach may appear somewhat complex. High communication time is a recognized potential "show-stopper," significantly affecting parallel program efficiency [185]. However, a well-engineered cMOMGA may not add significant overhead when compared to a like number of independent runs (as regards computational expense) and may result in better performance.

Several parallel algorithm efficiency and effectiveness metrics exist; we consider primary ones [185]. *Serial run time* is the elapsed time between program execution start and finish on a sequential computer. *Parallel run time* is the elapsed time from the initial parallel computation to the last (by any processor). *Speedup* is a relative measure showing (or not) the benefit of executing an algorithm in parallel. It is defined as the ratio of serial run time to parallel run time, using the same problem instance and executing on p processors. *Efficiency* measures the fraction of time a processor is actually processing, and is defined as the ratio of speedup to the number of processors. *Cost* is the product of parallel run time and the number of processors used. These computational performance metrics are then teamed with appropriate algorithmic performance measures to determine overall cMOMGA performance. We also note that although several parallel MOEAs have been implemented (see Appendix A) no formal computational performance results are reported.

4.7 Summary

This chapter proposes a new MOEA (based on the mGA) which explicitly manipulates BBs in its search for P_{true}. The MOMGA's operation is substantiated by an overview and discussion of desired BB identification and application in the MOP domain. Although the MOMGA incorporates current MOEA theory and mGA structures that fact is no guarantor of "good" algorithm performance. In order to determine both its effectiveness and efficiency the MOMGA must be included in experiments comparing selected MOEAs' performance on appropriate MOPs.

V. MOEA Test Suite Generation and Design

When the mathematician says that such and such a proposition is true of one thing, it may be interesting, and it is surely safe. But when he tries to extend his proposition to everything, though it is much more interesting, it is also much more dangerous. In the transition from one to all, from the specific to the general, mathematics has made its greatest progress, and suffered its most serious setbacks, of which the logical paradoxes constitute the most important part.

E. Kasner and J. Newman , *Mathematics and the Imagination*

5.1 Introduction

Many research efforts use numeric MOPs as examples to show or judge MOEA performance. However, there is no comprehensive discussion of MOP landscape issues in the MOEA literature, nor is there any explanation of why (the selected) numeric MOPs may be appropriate MOEA test functions. Extensive experimentation and analyses concerning MOEA parameters, components, and approaches are also lacking.

To date, most MOEA researchers' *modus operandi* is an algorithm's comparison (usually the researcher's own new and improved variant) against some other MOEA by analyzing results for specific MOP(s) (Schaffer's VEGA and MOP-$F2$ are typical [289]). Results are often "clearly" shown in graphical form indicating the new algorithm is more effective. However, these empirical, relative experiments are incomplete as regarding general MOEA comparisons. The literature's history of visually comparing MOEA performance on non-standard and unjustified numeric MOPs does little to determine a given MOEA's actual efficiency and effectiveness. A standard suite of numeric functions exhibiting *relevant* MOP domain characteristics can provide the necessary common comparative basis.

The MOEA community's limited *de facto* test suites contain various functions, many of whose origins and rationale for use are unknown. Thus, a documented MOP test suite is an asset to MOEA research. We provide various MOPs for use in a standardized MOEA test suite. Supporting these proposals is a detailed discussion of general test suite issues and the MOP domain.

This chapter is organized as follows. Sections 5.2 and 5.3 discuss general test suite issues and relevant MOP domain characteristics. Section 5.4 then proposes appropriate numeric MOPs for an MOEA test function suite given the described MOP domain features.

5.2 An MOEA Test Function Suite

As previously indicated, the MOEA community has created limited test suites. Specific functions are often used because other researchers did so in their research, or perhaps because the MOP appears to exercise certain MOEA components. It is not clear that these specific test functions are appropriate for inclusion into an MOEA test suite. Explanation is rarely offered as to the MOP's origin or *raison d'etre*, and several appear to be relatively "easy" (see Section 5.3). Poloni et al. [257] also note the lack of complex mathematical MOEA performance assessment tests. This situation implies that identification of appropriate functions to objectively determine MOEA efficiency and effectiveness is required. Other researchers also note the need for a test suite of this type [83, 258, 331].

5.2.1 General MOEA Test Suite Issues. Generic test function suites are both condoned and condemned. Any algorithm successfully "passing" all submitted test functions has no guarantee of continued effectiveness and efficiency when applied to real-world problems, i.e., examples prove nothing. Automotive passenger airbags are a prime example; not until they were widely fielded was it discovered that airbag-babyseat interactions are sometimes deadly. Pattern recognition research recognizes the additional problem of "testing on the training data," where an algorithm is tuned for only one or a few problem instances [94]. These analogies hold when integrating the MOP and MOEA domains; new and unforeseen situations may arise resulting in undesirable consequences. An MOEA test suite is then a valuable tool only if relevant issues such as those that follow are properly considered.

The NFL theorems [346] imply that if problem domain knowledge is not incorporated into the algorithm domain no formal assurances of an algorithm's general effectiveness exist. Previously proposed EA test suites examine an EA's capability to "handle" various problem domain characteristics. These suites incorporate relevant search space features to be addressed by some particular EA instantiation. For example, De Jong [80] suggests five single-objective optimization test functions (**F1 - F5**) and Michalewicz [219] five single-objective *constrained* optimization test functions (**G1 - G5**). Whitley et al. [341] and Goldberg et al. [130] offer other formalized EA test suites; informal suites are also used [349, 350].

De Jong's test bed includes functions with the following characteristics [126]: continuous and discontinuous, convex and nonconvex, unimodal and multimodal, quadratic and nonquadratic, low- and high-dimensionality, and deterministic and stochastic. Michalewicz's test bed addresses the following issues [219]: type of objective function (e.g., linear, nonlinear, quadratic), number of decision variables and constraints, types of constraints (linear and/or nonlinear), number of active constraints at the function's optimum, and the ratio between the feasible and complete search space size. Particular EA instantiations are subjected to generic test suites like these and judged on their performance.

Test suites must contain characteristic problems from target algorithms' problem domain. Some problems should represent real-world situations. Yet others should range in difficulty from "easy" to "hard." We also consider the following guidelines suggested by Whitley et al. [341]:

- Some test suite problems should be *resistant* to simple search strategies.
- Test suites should contain nonlinear, nonseparable, and nonsymmetric problems.
- Test suites should contain scalable problems.
- Some test suite problems should have scalable evaluation cost.
- Test problems should have a canonical representation.

Note that the NFL theorems also imply incorporating too much problem domain knowledge into a search algorithm reduces its effectiveness on other problems. However, as long as a test suite involves only *major* problem domain characteristics, any search algorithm giving effective and efficient results over the test suite might remain broadly applicable to *problems from that domain*. Thus, traits common to all (most) known MOPs must be defined.

5.3 MOP Domain Features

We first assert that like single-objective EA optimization problems, numeric MOPs may be suitable representatives of real-world problems. Any modeled real-world problem is done so mathematically in a functional form, but MOPs arguably capture more information about the modeled problem as they allow incorporation of several functions (objectives). Regardless, modeling a real-world problem may result in a numeric or combinatorial MOP, one that is perhaps simple, perhaps complex. The MOP may contain continuous or discrete (e.g., integer-constrained) functions or even a mix of the two. We here restrict discussion to homogeneously continuous MOPs; other MOP types are discussed in Section 5.4.2.

It is generally accepted that EAs are useful search algorithms when the problem domain is multidimensional (many decision variables), and/or the search space is very large. Many numerical examples used by MOEA researchers do *not* explicitly meet this criteria. Of the 30 distinct numerical MOPs in the literature (both constrained and unconstrained, see Appendix B), all but three use at most two decision variables and the majority use only two objective functions. This implies that unless the search space is very large (at the least), MOEA performance claims/comparisons based on these functions may not be meaningful. The MOEA may be operating in a problem domain not particularly well-suited to its capabilities or perhaps one which is not challenging.

Some MOP test functions build upon commonly used single-objective optimization test functions. For example, Kursawe's MOP incorporates a modified Ackley's function [17:pg. 143] and a modification of one provided by Schwefel [295:pg. 341]. Poloni's MOP incorporates a modified Fletcher-Powell function [17:pg. 143]. Finally, Quagliarella's MOP uses two versions of Rastrigin's function [51]. The rationale for construction and use of these and many of the other identified MOPs is unclear.

Any proposed MOP test suite must offer functions spanning known MOP characteristics. Particularly, it must contain "MOEA challenging" functions. In order to then identify appropriate functions for inclusion relevant MOP domain characteristics must be identified and considered. We use the 30 known examples in the literature as the basis for discussion; a complete list is found in Tables B.1 and B.2 in Appendix B. These MOPs each incorporate 2-3 functions and 0-12 side constraints. Appendices C and D present a complete set of figures showing P_{true} and PF_{true} for each MOP listed in the tables. These figures are deterministically derived by computing all decision variable combinations possible at a given computational resolution. Their purpose is to highlight major structural characteristics of both P_{true} and PF_{true} for use in constructing a sound MOEA test function suite.

When implementing an MOEA it is (implicitly) assumed that the problem domain has been properly considered, and a decision made that an MOEA is an appropriate search algorithm for the given MOP. We also assume the MOEA's objective is return of P_{known}. Thus, Tables 5.1 and 5.2 identify salient MOP domain characteristics viewed from an MOEA perspective and classified under a genotype and phenotype rubric. Newly identified characteristics may be considered later. We caution that these high-level characteristics were determined from the figures presented in Appendices C and D, whose representation (and succeeding interpretation) may slightly change based upon underlying computational resolution and graphical presentation.

The table entries are explained as follows. Each row corresponds to one of the MOPs listed in Appendix B. Each column signifies some genotypic/phenotypic characteristic. P_{true}'s "shape" may be connected, disconnected, symmetric, and/or scalable. PF_{true} may be connected, disconnected, and convex or concave. MOPs exhibiting any of these characteristics are marked with an "x" in the appropriate column. Solution types are notated by the number of decision variables and their type, where "R" indicates real (continuous) decision variables. The number of functions is self-explanatory. Table 5.1 lists MOPs associated with only decision variable constraints, identifying their numbers and types. Table 5.2 lists MOPs which also contain side constraints, identifying both constraint numbers and types. Each MOPs' PF_{true}'s shape is listed, as Pareto fronts may geometrically and/or topologically differ. We also note that only two of these MOPs (Fonseca's second [109] and Schaffer's first [276]) have analytical solutions for P_{true}.

What is P_{true}'s nature? Few MOEA efforts describe an example MOP's underlying decision variable (genotype) space, i.e., the space where P_{true} resides. Since an MOP is composed of two or more functions, the solution space is obviously restricted by their

Table 5.1. MOP Numeric Test Function Characteristics

	Genotype							Phenotype				
Function	Connected	Disconnected	Symmetric	Scalable	Solution Type(s)	# Functions	Constraints	Geometry	Connected	Disconnected	Concave	Convex
Binh	x		x		2R	2	2	Curve	x			x
Binh (3)	x				2R	3	2	Point				
Fonseca	x		x		2R	2	0	Curve	x		x	
Fonseca (2)	x		x	x	nR	2	n	Curve	x		x	
Kursawe		x	x	x	nR	2	0	Curve		x	x	
Laumanns	x		x		2R	2	2	Points		x		
Lis	x		x		2R	2	2	Points		x		
Murata	x		x		2R	2	2	Curve	x		x	
Poloni		x			2R	2	2	Curves		x		
Quagliarella		x		x	nR	2	n	Points		x		
Rendon	x		x		2R	2	2	Curve	x			x
Rendon (2)	x		x		2R	2	2	Curve	x			x
Schaffer	x		x		1R	2	0	Curve	x			x
Schaffer (2)		x	x		1R	2	1	Curves		x		
Vicini	x				2R	2	2	Curve	x			
Viennet	x		x		2R	3	2	Surface	x			
Viennet (2)	x				2R	3	2	Surface	x			
Viennet (3)		x			2R	3	2	Curve	x			

combined limitations (e.g., decision variable range and side constraints). Within that space, P_{true} may be connected or disconnected, an (hyper)area or separate points, symmetric in shape, scalable, and so forth. Solutions may be discrete or continuous, and are composed of one or more decision variables. When solved computationally (and assuming feasible solutions exist), an MOP's P_{true} has only a lower bound (see Theorem 1 in Section 2.2.2.1); the upper bound is unknown and varies depending upon the underlying computational resolution.

What is PF_{true}'s nature? PF_{true} lies in objective space and as already noted, may be (dis)connected, convex or concave, and multidimensional. In fact, the structure of *any* Pareto front has theoretical dimensional limitations depending on the number of functions

Table 5.2. MOP Numeric Test Function (with side constraints) Characteristics

	Genotype							**Phenotype**				
Function	Connected	Disconnected	Symmetric	Scalable	Solution Type(s)	# Functions	Constraints	Geometry	Connected	Disconnected	Concave	Convex
Belegundu	x		x		2R	2	2 + 2S	Curve	x			x
Binh (2)	x		x		2R	2	2 + 2S	Curve	x			x
Binh (4)	x				2R	3	2 + 2S	Surface	x		x	
Jimenez	x		x		2R	2	2 + 4S	Curve	x			x
Kita		x	x		2R	2	2 + 3S	Curves		x		
Obayshi	x		x		2R	2	2 + 1S	Curve	x			x
Osyczka		x			2R	2	2 + 2S	Points		x		x
Osyczka (2)		x			6R	2	6 + 6S	Curves		x		
Srinivas		x	x		2R	2	2 + 2S	Curve	x			x
Tamaki	x		x		3R	3	3 + 1S	Surface	x			
Tanaka		x			2R	2	2 + 2S	Curves		x		
Viennet (4)		x			2R	3	2 + 3S	Surface	x			

composing the MOP (see Theorem 3 in Section 2.2.2.2). PF_{true}'s shape can range from a single vector to a collection of multi-dimensional surfaces.

Test suite functions should encompass (combinations of) all these possible characteristics. Although no guarantor of continued success, any search algorithm giving effective and efficient results over the test suite might be easily modified to target specific problems.

5.3.1 Related MOP Domain Research. Deb has recently published work which also addresses MOEA test suite issues [83, 84]. As we are cooperating with him in some MOEA research his efforts deserve critical attention, especially as he proposes a *methodology* for constructing MOPs exhibiting desired characteristics. Contrived functions may then be generated for use in MOEA test suites. We address key issues as they are ordered in Deb's tech report [83].

Deb defines both a *local* and *global* Pareto optimal set. His global Pareto optimal set is what we term P_{true}; our terminology is easily extended to denote a local Pareto optimal set, i.e., P_{local}. However, P_{local} is ill-defined and may be confusing. Consider Deb's definition:

Definition 11 (Local Pareto Optimal Set): *Given some Pareto optimal set $\mathcal{P}$, if $\forall x \in \mathcal{P}$, $\neg\exists y$ satisfying $\| y - x \|_\infty \leq \epsilon$, where ϵ is a small positive number (in principle, y is obtained by perturbing x in a small neighborhood), and for which $F(y) \preceq F(x)$, then the solutions in $\mathcal{P}$ constitute a local Pareto optimal set.* □

This definition implies that for some given set of Pareto optimal solutions, each is perturbed in some manner but no new nondominated vectors are found. Deb's purpose here is defining a set of Pareto optimal solutions whose associated front (PF_{local}) is "behind" PF_{true} for the given MOP. Although conceptually possible, any P_{local}'s existence is dependent upon the ϵ selected within which solutions are perturbed. Additionally, too large an ϵ prohibits a P_{local}, too small an ϵ may result in many local fronts.

Deb also extends the concepts of multimodality, deception, an isolated optimum, and collateral noise (well known single-objective EA difficulties) to the multiobjective domain. We dispute two of these extensions. First, he defines a deceptive MOP as one in which there are at least two optima (PF_{local} and PF_{true}) and where the majority of the search space favors PF_{local}. As stated above this concept depends on P_{local}'s existence. Secondly, Deb defines a multimodal MOP as one with multiple local fronts. This definition mixes terminology. One should use the term multimodal only when referring to a single-objective optimization function containing both local and global minima. As all vectors composing a Pareto front are "equally" optimal there is no Pareto front modality. Perhaps the term "multifrontal" is a better choice to reflect this situation.

Deb also notes some of the same MOP phenotype characteristics as we presented in Section 5.3. He points out that when computationally derived a non-uniform distribution of vectors may exist in some Pareto front. He limits his initial test construction efforts to unconstrained MOPs of only two functions; his construction methodology then places restrictions on the two component functions so that resultant MOPs exhibit desired properties. To accomplish this he defines the following generic bi-objective optimization problem:

Minimize $F = (f_1(\vec{x}), f_2(\vec{x}))$, where

$$\begin{aligned} f_1(\vec{x}) &= f(x_1, \ldots, x_m), \\ f_2(\vec{x}) &= g(x_{m+1}, \ldots, x_N)\, h(f(x_1, \ldots, x_m), g(x_{m+1}, \ldots, x_N)) \; . \end{aligned} \tag{5.1}$$

The function f_1 is a function of $(m < N)$ decision variables and f_2 a function of all N decision variables. The function g is one of $(N - m)$ decision variables which are not included in function f. The function h is directly a function of f and g function values. The f and g functions are also restricted to positive values in the search space, i.e., $f > 0$ and $g > 0$. Deb then lists five functions each for possible f and g instantiation, and four for h. These functions may then be "mixed and matched" to create MOPs with desired characteristics.

He states these functions have the following general effect:

f – This function controls vector representation uniformity along the Pareto front.

g – This function controls the resulting MOP's characteristics – whether it is multifrontal or has an isolated optimum.

h – This function controls the resulting Pareto front's characteristics (e.g., convex, disconnected, etc.)

We agree that these functions respectively influence search along and towards the Pareto front, and the shape of a Pareto front in $\mathbb{R}^2$. However, one of Deb's examples highlights a possible problem with some MOPs constructed using this methodology. Consider the following [83:pg. 9]:

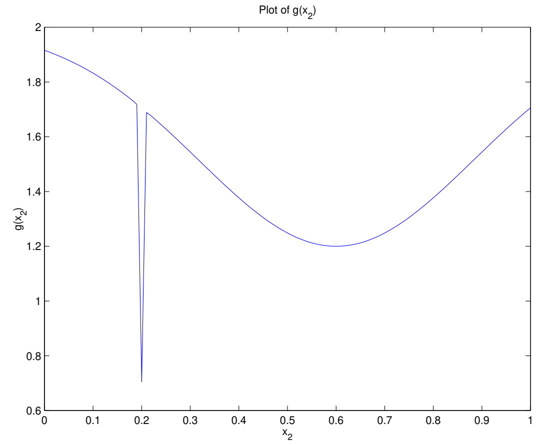


Figure 5.1. $g(x_2)$ Values

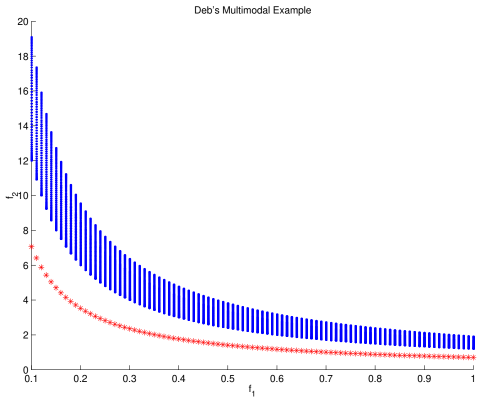


Figure 5.2. Pareto Fronts

Minimize $F = (f_1(x_1, x_2), f_2(x_1, x_2))$, where

$$\begin{aligned} f_1(x_1, x_2) &= x_1, \\ f_2(x_1, x_2) &= \frac{2.0 - \exp\{-(\frac{x_2-0.2}{0.004})^2\} - 0.8\exp\{-(\frac{x_2-0.6}{0.4})^2\}}{x_1}. \end{aligned} \tag{5.2}$$

In this MOP f_2 may also be represented as $\frac{g(x_2)}{x_1}$. Thus, $g(x_2)$ is the bimodal function represented in Figure 5.1. This function has optima of $g(0.6) \approx 1.2$ and $g(0.2) \approx 0.7057$. Figure 5.2 shows the MOP's Pareto fronts (as Deb proposes). The lower portion of the upper vector band Deb terms PF_{local} ; the lower band is PF_{true} . The solutions corresponding to P_{local} are $\{(x_1, x_2) \mid x_2 \approx 0.6\}$ and those corresponding to P_{true} are $\{(x_1, x_2) \mid x_2 \approx 0.2\}$.

Deb then implies an MOEA has difficulty finding PF_{true} because it gets "trapped" in the local optimum, namely PF_{local} . However, this is not a phenotypical effect but rather one due to the underlying genotype space. In this computational derivation function $g(x_2)$'s global optimum is in a narrow valley where fewer discretized search points exist. A pure random search results in fewer points stochastically found "close to" or in this valley, as opposed to the broad valley surrounding the local optimum containing many more points. Thus, the difficulty in finding PF_{true} is due to the number of discrete points near $g(x_2)$'s global optimum and not simply the fact that PF_{local} exists. This example is one showing deceptiveness rather than multifrontality.

This example also highlights a problem we previously alluded to – that of discretizing continuous functions (or solution spaces). The resultant mapping may not reflect reality in that the computational discretization process may introduce "errors." Additionally, a uniform discretization of decision variable space does not imply uniform mappings into objective space. In general, one must then be careful when analyzing and comparing MOEA performance in various MOP domains. Different MOEA techniques (including parameters and EVOPs) perform differently between and even within these domains (c.f., the NFL theorems [346]).

Additionally, this methodology is not the only way to construct MOPs exhibiting some set of desired characteristics. Real-world MOPs may have similar genotype and/or phenotype characteristics but look nothing at all like the examples Deb proposes. Thus, the fact an MOEA "passes" all test functions submitted using Deb's methodology may have no bearing on its performance in solving real-world MOPs. However, the same can be said of the test suite proposed in the next section. Any test functions must be carefully selected to reflect as accurately as possible the problem domain they represent.

This analysis is not meant to belittle Deb's effort. His methodology sometimes results in MOPs with analytical solutions for P_{true} or PF_{true}, allowing for absolute comparison of MOEA results and the MOP optimum. He also is attempting to generate an MOEA test suite containing functions which *in toto* consider relevant MOP genotype/phenotype characteristics. Because several distinct MOPs may be created using Deb's initial methodology [83], direct implementations of those functions are not listed in Appendix B.

5.4 Numeric MOEA Test Suite Functions

Having shown the requirement for and considered the general issues involved in an MOEA test function suite we now propose initial problems for inclusion. As discussed in the last section, a sound methodology for constructing MOPs with arbitrary complexity and characteristics still eludes us. Thus, proposed test suite MOPs are drawn from the published literature. These MOPs *in toto* address some of the issues discussed in Section 5.2 and reflect the characteristics in Table 5.1. We restrict initial functions to those with no side constraints. Their mathematical formulations (which may be slightly revised from the originals or as we elsewhere proposed [327]) are shown in Table 5.3. Figures 5.3 through 5.16 show representations of each MOPs' P_{true} and PF_{true}.[1]

Schaffer's first (unconstrained) two-objective function is selected for three primary reasons. First is its historical significance; almost all proposed MOEAs have been tested using this function. It is also an exemplar of relevant MOP concepts. Second, this MOP allows determination of an analytical expression for PF_{true} [325]. Third, as noted by Rudolph [276] this MOP's P_{true} is in closed form so solutions' membership in P_{true} is then easily determined. This MOP's PF_{true} is a single convex Pareto curve and its P_{true} a line. However, its one decision variable implies it may not be well-suited to an MOEA's search capabilities. We rename this problem **MOP1**.

Fonseca's second MOP is also selected. This two-objective function has an advantage of arbitrarily adding decision variables (scalability) without changing PF_{true}'s shape or location in objective space [109]. This MOP's PF_{true} is a single concave Pareto curve and its P_{true} an area in solution space. Additionally, a closed form for this MOP's P_{true} is claimed [109]. We rename this problem **MOP2**.

Next is Poloni's MOP, a maximization problem. This two-objective function's P_{true} is two disconnected areas in solution space while its PF_{true} is two disconnected Pareto curves. Its solution mapping into dominated objective space also appears more convoluted than other MOPs from the literature. We rename this problem **MOP3**.

Kursawe's MOP is included. This two-objective function's P_{true} is several disconnected and unsymmetric areas in solution space. Its PF_{true} is three disconnected Pareto curves. Like MOP3, its solution mapping into dominated objective space is also quite convoluted. Like MOP2, its number of decision variables is arbitrary. However, changing the number of decision variables appears to slightly change PF_{true}'s shape and does change its location in objective space.

Figure C.10 in Appendix A was derived using Kursawe's MOP with two decision variables. Compare this to Figure 5.8 which uses three decision variables. It is easily seen

[1]Note that the graphs' scales for P_{true} may be different than what is stated in Table 5.3 to show P_{true}'s "shape" more clearly.

that PF_{true} and the dominated vectors have shifted in objective space. Implementing this MOP with four decision variables resulted in another shift. We can make no conclusive claims about PF_{true}'s changing shape without increasing the computational resolution used in constructing the graphs. We rename this function **MOP4**.

We also propose Viennet's third MOP. This tri-objective function's P_{true} consists of disconnected areas in solution space, and its PF_{true} a single, convoluted three-dimensional Pareto curve. We rename this function **MOP5**.

An MOP constructed using Deb's methodology (and used by him as an example [83]) is selected. Like MOP4, this two-objective function's P_{true} and PF_{true} are disconnected, although its PF_{true} consists of four Pareto curves. Its solution mapping into dominated objective space is not as convoluted as MOP4's. This problem is used to compare MOEA performance in finding similar phenotypes produced by different MOPs (c.f., MOP4). We rename this function **MOP6**.

Finally, we propose Viennet's second MOP. This tri-objective MOP's P_{true} is a connected region in solution space. Its PF_{true} appears to be a surface and its mapping into objective space appears straightforward. This function is primarily meant to complement to MOP5. We rename this function **MOP7**.

Table 5.3 MOEA Test Suite Functions

MOP	Definition	Constraints
MOP1 P_{true} connected, PF_{true} convex	$F = (f_1(x), f_2(x))$, where $$\begin{aligned} f_1(x) &= x^2, \\ f_2(x) &= (x-2)^2 \end{aligned}$$	$-10^5 \le x \le 10^5$
MOP2 P_{true} connected, PF_{true} concave, number of decision variables scalable	$F = (f_1(\vec{x}), f_2(\vec{x}))$, where $$\begin{aligned} f_1(\vec{x}) &= 1 - \exp(-\sum_{i=1}^{n}(x_i - \frac{1}{\sqrt{n}})^2), \\ f_2(\vec{x}) &= 1 - \exp(-\sum_{i+1}^{n}(x_i + \frac{1}{\sqrt{n}})^2) \end{aligned}$$	$-4 \le x_i \le 4;\ i = 1,2,3$

Table 5.3 (continued)

MOP	Definition	Constraints
MOP3 P_{true} disconnected, PF_{true} disconnected (2 Pareto curves)	Maximize $F = (f_1(x,y), f_2(x,y))$, where $\begin{aligned} f_1(x,y) &= -[1 + (A_1 - B_1)^2 + (A_2 - B_2)^2], \\ f_2(x,y) &= -[(x+3)^2 + (y+1)^2] \end{aligned}$	$-3.1416 \le x, y \le 3.1416$, $\begin{aligned} A_1 &= 0.5 \sin 1 - 2 \cos 1 + \sin 2 - 1.5 \cos 2, \\ A_2 &= 1.5 \sin 1 - \cos 1 + 2 \sin 2 - 0.5 \cos 2, \\ B_1 &= 0.5 \sin x - 2 \cos x + \sin y - 1.5 \cos y, \\ B_2 &= 1.5 \sin x - \cos x + 2 \sin y - 0.5 \cos y \end{aligned}$
MOP4 P_{true} disconnected, PF_{true} disconnected (3 Pareto curves), number of decision variables scalable	$F = (f_1(\vec{x}), f_2(\vec{x}))$, where $\begin{aligned} f_1(\vec{x}) &= \sum_{i=1}^{n-1} (-10 e^{(-0.2)*\sqrt{x_i^2 + x_{i+1}^2}}), \\ f_2(\vec{x}) &= \sum_{i=1}^{n} (\lvert x_i \rvert^{0.8} + 5 sin(x_i)^3) \end{aligned}$	$-5 \le x_i \le 5;\ i = 1, 2, 3$
MOP5 P_{true} disconnected and unsymmetric, PF_{true} connected (a 3-D Pareto curve)	$F = (f_1(x,y), f_2(x,y), f_3(x,y))$, where $\begin{aligned} f_1(x,y) &= 0.5 * (x^2 + y^2) + sin(x^2 + y^2), \\ f_2(x,y) &= \frac{(3x - 2y + 4)^2}{8} + \frac{(x - y + 1)^2}{27} + 15, \\ f_3(x,y) &= \frac{1}{(x^2 + y^2 + 1)} - 1.1 e^{(-x^2 - y^2)} \end{aligned}$	$-30 \le x, y \le 30$
MOP6 P_{true} disconnected, PF_{true} disconnected (4 Pareto curves), number of Pareto curves scalable	$F = (f_1(x,y), f_2(x,y))$, where $\begin{aligned} f_1(x,y) &= x, \\ f_2(x,y) &= (1 + 10y) * \\ & [1 - (\frac{x}{1 + 10y})^{\alpha} - \frac{x}{1 + 10y} \sin(2\pi q x)] \end{aligned}$	$0 \le x, y \le 1$, $\begin{aligned} q &= 4, \\ \alpha &= 2 \end{aligned}$
MOP7 P_{true} connected, PF_{true} disconn ected	$F = (f_1(x,y), f_2(x,y), f_3(x,y))$, where $\begin{aligned} f_1(x,y) &= \frac{(x-2)^2}{2} + \frac{(y+1)^2}{13} + 3, \\ f_2(x,y) &= \frac{(x + y - 3)^2}{36} + \frac{(-x + y + 2)^2}{8} - 17, \\ f_3(x,y) &= \frac{(x + 2y - 1)^2}{175} + \frac{(2y - x)^2}{17} - 13 \end{aligned}$	$-400 \le x, y \le 400$

These proposed MOEA test functions address the issues mentioned in Section 5.2. MOP1 and MOP2 are arguably "easy" MOPs. MOP2 and MOP4 are scalable as regards decision variable dimensionality. MOP6 is scalable as regarding the number of Pareto

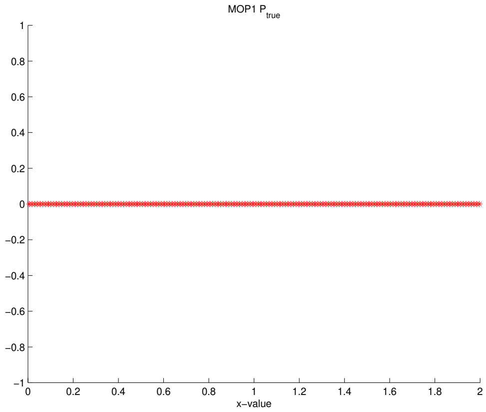


Figure 5.3. MOP1 P_{true}

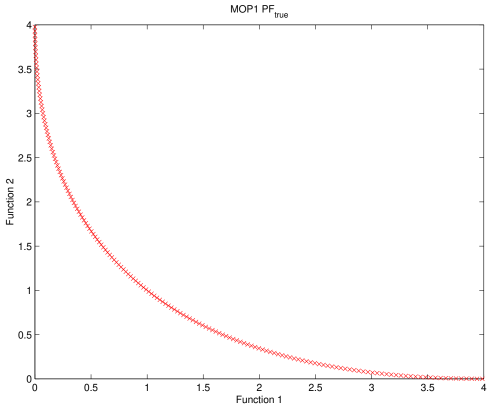


Figure 5.4. MOP1 PF_{true}

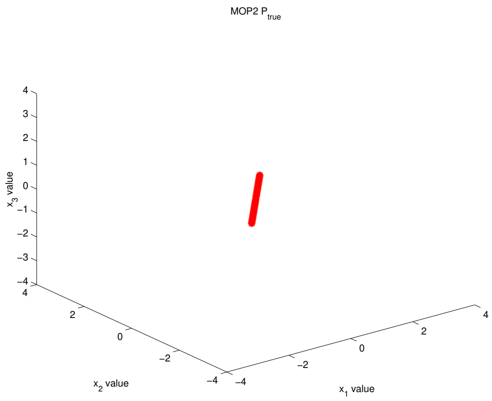


Figure 5.5. MOP2 P_{true}

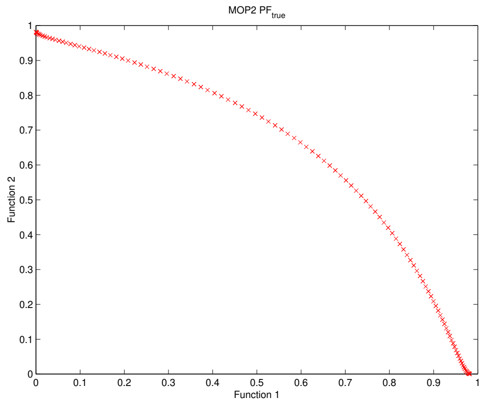


Figure 5.6. MOP2 PF_{true}

curves in PF_{true}. MOP5 and MOP7 are tri-objective MOPs. All are nonlinear, and several show a lack of symmetry in both P_{true} and PF_{true}. Taken together these MOPs begin to form a coherent basis for MOEA comparisons. However, other relevant MOP characteristics (as reflected in Tables 5.1 and 5.2) should also be addressed by further MOPs selected for test suite inclusion. These additional MOPs may need to be constructed in order to exhibit desired characteristics. Other MOP types should also be considered even though not pursued further in this research.

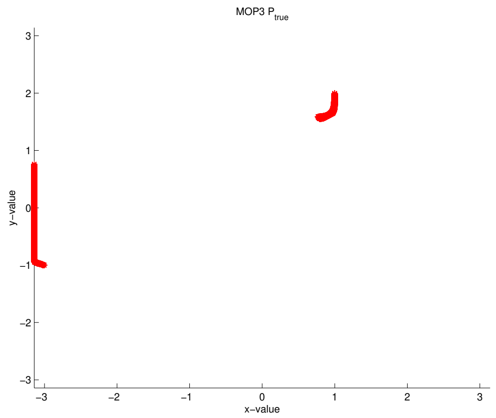


Figure 5.7. MOP3 P_{true}

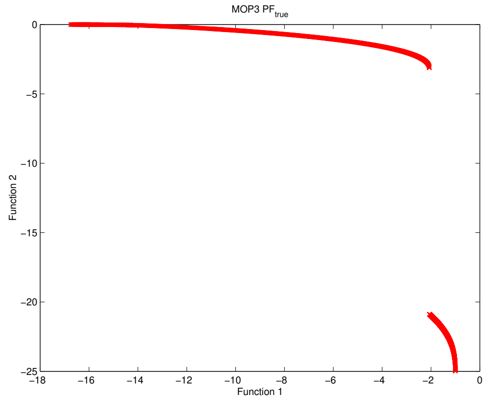


Figure 5.8. MOP3 PF_{true}

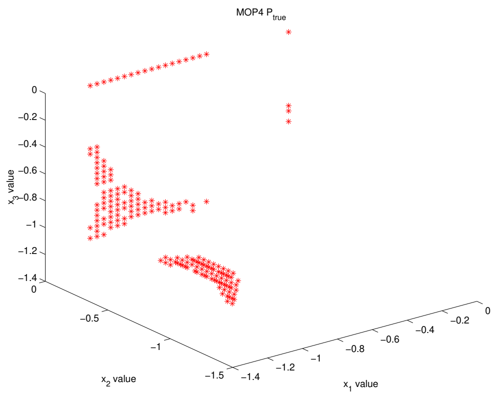


Figure 5.9. MOP4 P_{true}

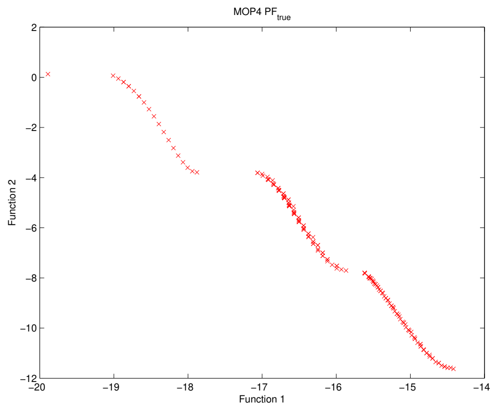


Figure 5.10. MOP4 PF_{true}

Table 5.4 Side-Constrained MOEA Test Suite Functions

MOP	Definition	Constraints
MOP-C1	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= 4x^2 + 4y^2, \\ f_2(x,y) &= (x-5)^2 + (y-5)^2 \end{aligned}$$	$0 \leq x \leq 5,\ 0 \leq y \leq 3,$ $$\begin{aligned} 0 &\geq (x-5)^2 + y^2 - 25, \\ 0 &\geq -(x-8)^2 - \\ &\quad (y+3)^2 + 7.7 \end{aligned}$$

Table 5.4 (continued)

MOP	Definition	Constraints
MOP-C2	$F = (f_1(\vec{x}), f_2(\vec{x}))$, where $$\begin{aligned} f_1(\vec{x}) &= -(25(x_1-2)^2 + (x_2-2)^2 + (x_3-1)^2 \\ &\quad + (x_4-4)^2 + (x_5-1)^2, \\ f_2(\vec{x}) &= x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2 \end{aligned}$$	$0 \le x_1, x_2, x_6 \le 10$, $1 \le x_3, x_5 \le 5$, $0 \le x_4 \le 6$, $$\begin{aligned} 0 &\le x_1 + x_2 - 2, \\ 0 &\le 6 - x_1 - x_2, \\ 0 &\le 2 - x_2 + x_1, \\ 0 &\le 2 - x_1 + 3x_2, \\ 0 &\le 4 - (x_3-3)^2 - x_4, \\ 0 &\le (x_5-3)^2 + x_6 - 4 \end{aligned}$$
MOP-C3	$F = (f_1(x,y), f_2(x,y), f_3(x,y))$, where $$\begin{aligned} f_1(x,y) &= \frac{(x-2)^2}{2} + \frac{(y+1)^2}{13} + 3, \\ f_2(x,y) &= \frac{(x+y-3)^2}{175} + \frac{(2y-x)^2}{17} - 13, \\ f_3(x,y) &= \frac{(3x-2y+4)^2}{8} + \frac{(x-y+1)^2}{27} + 15 \end{aligned}$$	$-4 \le x, y \le 4$, $$\begin{aligned} y &< -4x + 4, \\ x &> -1, \\ y &> x - 2 \end{aligned}$$

5.4.1 Side-Constrained Numeric MOEA Test Functions. Side-constrained numeric MOPs should be included in any comprehensive MOEA test function suite; we here propose suitable MOPs drawn from the published literature. However, one must be aware that solving constrained MOPs with MOEAs brings in other open research issues, most notably how the side constraints are accounted for to ensure feasible solutions.

Binh's second MOP is selected. This two-objective function's P_{true} is an area in solution space and its PF_{true} a single convex Pareto curve. We rename this problem **MOP-C1**. Next is Osyczka's second MOP, which is a heavily constrained, six decision variable problem. This two-objective function's P_{true} 's shape is currently unknown while its PF_{true} is three disconnected Pareto curves. We rename this problem **MOP-C2**. Finally, Viennet's fourth MOP is selected for inclusion. This three-objective function's P_{true} is an irregularly shaped area in solution space. Its PF_{true} is a Pareto surface. We rename this problem **MOP-C3**. These MOPs' mathematical formulations are shown in Table 5.4; figures showing representations of each MOPs' P_{true} and PF_{true} are found in Appendix D.

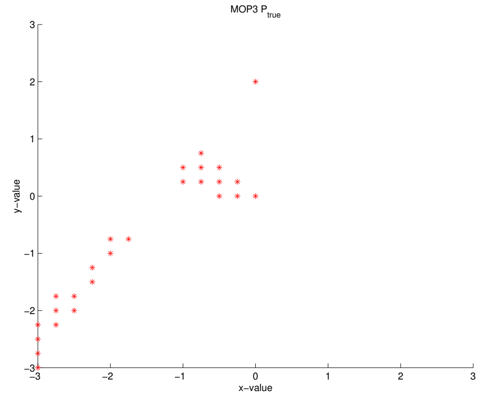


Figure 5.11. MOP5 P_{true}

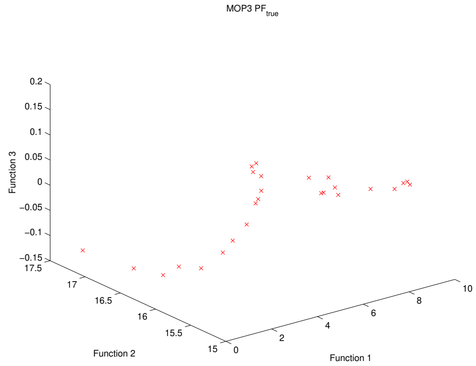


Figure 5.12. MOP5 PF_{true}

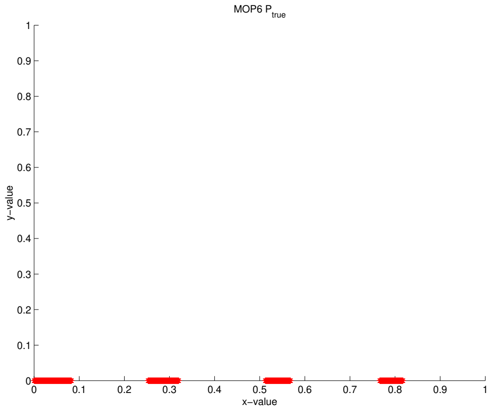


Figure 5.13. MOP6 P_{true}

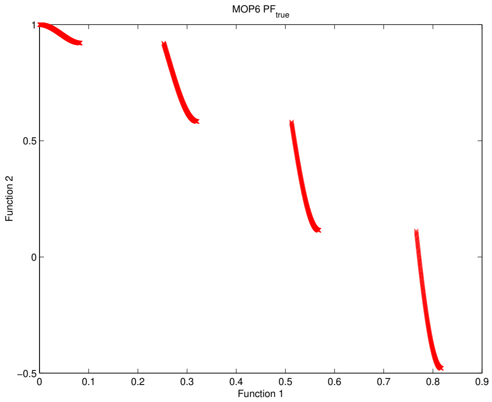


Figure 5.14. MOP6 PF_{true}

5.4.2 Combinatorial and Real-World MOEA Test Functions. Although most MOP test functions found in the MOEA literature are numeric, some combinatorial problems are used that provide differing algorithmic challenges. A combinatorial optimization problem is mathematically defined as follows: [120]

Definition 12 (Combinatorial Optimization Problem): *A combinatorial optimization problem π is either a minimization or maximization problem consisting of three parts.*

1. *A domain D_π of instantiations;*
2. *For each instance $I \in D_\pi$ a finite set $S_\pi(I)$ of candidate solutions for I; and*

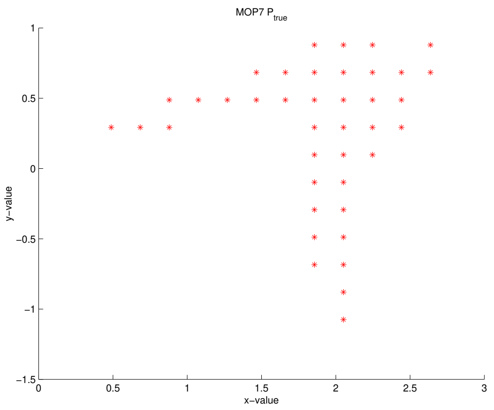


Figure 5.15. MOP7 P_{true}

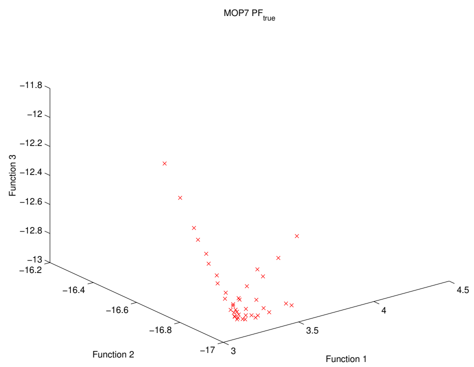


Figure 5.16. MOP7 PF_{true}

3. *A function m_π that assigns a positive rational number $m_\pi(I, \sigma)$ to each candidate solution $\sigma \in S_\pi(I)$ for each instance $I \in D_\pi$. $m_\pi(I, \sigma)$ is called the solution value for σ.*

□

An MOEA is able to search these finite (discrete) solution spaces but may require specialized EVOPs ensuring only feasible solutions (i.e., $S_\pi(I)$) are generated for evaluation. However, the phenotype domain of combinatorial MOPs is slightly different than that of its numeric counterparts. These MOPs' mapping into objective space is discrete and offers only isolated points (vectors) in objective space. As only a finite number of solutions exist only a finite number of corresponding vectors may result. Although these vectors may appear to form a continuous front when plotted, the genotype domain's discrete nature implies no solutions exist mapping to vectors between those composing PF_{true}.

Various combinatorial MOPs are reflected in the MOEA literature. Horn [154] and Deb [83] present combinatorial (unitation) MOPs. Louis converts a deceptive GA problem into an MOP [207]. NP-Complete problems are combinatorial optimization problems and many NP-Complete MOP test functions are used. For example, a group of Japanese researchers focus on the use of fuzzy logic and MOEAs in solving Multiobjective 0-1 Programming problems (e.g., [173, 280, 302]). Several efforts investigate Multiobjective Solid Transportation Problems [44, 168, 122, 198, 197]. Other traditional NP-Complete problems are also transformed into MOPs, including Multiobjective Flowshop

Table 5.5. Possible Multiobjective NP-Complete Functions

NP-Complete Problem	**Example**
0/1 Knapsack - Bin Packing	Max profit; Min weight
Traveling Salesperson	Min energy, time, and/or distance; Max expansion
Coloring	Min number of colors, number of each color
Set/Vertex Covering	Min total cost, over-covering
Maximum Independent Set (Clique)	Max set size; Min geometry
Vehicle Routing	Min time, energy, and/or geometry
Scheduling	Min time, missed deadlines, waiting time, resource use
Layout	Min space, overlap, costs
NP-Complete Problem Combinations	Vehicle scheduling and routing

Scheduling [164], Multiobjective Job Shop Scheduling [199], and Multiobjective Knapsack Problems [280, 358, 359].

Thus, we should consider the use of combinatorial MOPs in any proposed MOEA test suite. On the one hand, EAs often employ specialized representations and operators when solving these real-world problems which usually prevents a general comparison between various MOEA implementations. On the other hand, these problems' inherent difficulty should present desired algorithmic challenges and complement other test suite MOPs. Table 5.5 outlines possible NP-Complete MOPs for inclusion. However, no known solution databases such as *TSPLIB* [271], *MP-Testdata* [360], or *OR Library* [24] exist for these NP-Complete MOPs.

Finally, real-world applications should be considered for inclusion in any comprehensive MOEA test suite. These MOPs may be numeric, non-numeric, or both, and are probably more constrained (in terms of resources) than the problems we have considered here. We note that many real-world applications require extensive fitness function (e.g., computational fluid dynamics or computational electromagnetic) software requiring data interchange and mapping (c.f., [210, 170, 41, 248, 318, 240, 262]).

5.5 Summary

In the tradition of providing test suites for evolutionary algorithms we propose an extensive list of specific MOEA test functions. The development of this list is based upon accepted and historic EA test suite guidelines. Specific MOEA test suites can evolve from this proposed list based upon individual research objectives and problem domain characteristic classifications. With a generic MOEA test suite, researchers can compare their multiobjective numeric and combinatorial optimization problem results (regarding effectiveness and efficiency) with others, over a spectrum of MOEA instantiations. Using our test suite functions MOEA comparisons can be made more precise and their results more informative. We describe such an effort in the next chapter.

VI. MOEA Experiments

"Wesley, under Point Four, we'll have to close all research departments, experimental laboratories, scientific foundations, and all the rest of the institutions of that kind. They'll have to be forbidden."
... Close all those damn research laboratories – and the sooner, the better."
... "The State Science Institute, too?" asked Fred Kinnan.
"Oh, no!" said Mouch. "That's different. That's government. Besides, it's a non-profit institution. And it will be sufficient to take care of all scientific progress."
"Quite sufficient," said Dr. Ferris.

Ayn Rand, *Atlas Shrugged*

6.1 Introduction

The careful design of MOEA experiments should draw heavily from outlines presented by Barr et al. [23] and Jackson et al. [166]. These articles discuss computational experiment design for heuristic methods, providing guidelines for reporting results and ensuring their reproducibility. Specifically, they suggest a well-designed experiment follows these steps: (1) Define experimental goals; (2) Choose measures of performance (metrics); (3) Design and execute the experiment; (4) Analyze data and draw conclusions; and (5) Report experimental results.

The authors also note metrics usually fall into one of four categories: (1) Efficiency (measuring computational effort to obtain solutions, e.g., CPU time, number of evaluations/iterations), and Effectiveness (measuring the accuracy of obtained solutions); (2) Robustness (measuring how well the code recovers from improper input); (3) Reliability (measuring how large a class of problems the code can solve); and (4) Ease of use (measuring the amount of effort required to use the software).

Following these guidelines, the reasons for and goals of these experiments are presented in Section 6.2. The experimental design and performance metrics are described within the methodology proposed in Section 6.3. Experimental results and analyses are then presented in Chapter VII.

6.2 MOEA Experiments: Motivation and Objectives

The major goal of these experiments is to compare well-engineered MOEAs in terms of effectiveness and efficiency as regards *carefully selected test problems* from the same class. Jackson et al. [166] imply this should suffice to show MOEA feasibility and promise. We are not claiming that MOEAs are the only algorithms able to solve these test problems efficiently and effectively, but wish to see if one MOEA performs "better" than another over this problem domain class, and if so determine why. If all MOEAs perform equally well, we also wish to determine why, as that situation implies MOEA implementation choice may not be crucial. Other interesting observations may also arise during experiment execution and result analysis.

The first selected experimental MOEA is the MOMGA, discussed in detail in Chapter IV. It is a new, unique, and innovative extension of a single-objective EA incorporating mechanisms that should theoretically result in effective performance. The other experimental MOEAs (described in Section 6.3.2) are also based on similar theoretical mechanisms. These MOEAs have been tested on various numeric problems and used in many scientific and engineering applications. Examples prove nothing but these MOEAs have a good track record. Thus, we choose to compare these MOEAs' performance in solving carefully selected MOPs based on appropriately defined metrics.

We wish to report relevant *quantitative* MOEA performance based on appropriate experiments. Almost all comparisons cited in the current literature visually compare algorithmic results. As experimental numeric MOPs' P_{true} and PF_{true} are often not known (and almost never presented) these conclusions are then relative. The methodology described in the next section gives a basis for *absolute* conclusions regarding MOEA performance. Finally, the last experimental goal is determining how well the test problems and proposed metrics capture and report essential MOP and MOEA characteristics and performance.

6.3 Experimental Methodology

Having investigated the MOP and MOEA domains in Chapters II and III, meaningful MOEA experiments may now be conducted. Although test suite functions do provide a common basis for MOEA comparisons, results are empirical unless the global optima are known. We again note that finding a general MOP's Pareto optimal solution set is NP-Complete [17:pg. 56]. However, there is a way to determine P_{true} for certain problems! Teaming this data with appropriate metrics then allows desired quantitative MOEA comparisons.

6.3.1 MOP P_{true} Determination. When the real- (continuous) world is modeled (e.g., via objective functions) on a computer (a discrete machine), there is a fidelity loss between the (possibly) continuous mathematical model and its discrete representation. Any formalized MOP being computationally solved suffers this fate. However, at a "standardized" computational resolution and representation, MOEA results can be quantitatively compared not only against each other but against certain MOPs' PF_{true}. Thus, whether or not these selected MOPs' PF_{true} is actually continuous or discrete is not an experimental concern, as the representable P_{true} and PF_{true} are fixed based on certain assumptions.

6.3.1.1 Computational Grid. For purposes of these experiments we define a *computational grid* by placing an equidistantly spaced grid over decision variable space, allowing a uniform sampling of possible solutions. Each grid intersection point (computable solution) is then assigned successive numbers using a binary representation. For example, given a fixed length binary string, decision variable values are determined by mapping the binary (sub)string to an integer int and then solving the following for each x_i:

$$x_i = l + \frac{int * (u - l)}{2^n - 1} \,, \tag{6.1}$$

where l and u correspond to the lower and upper decision variable bounds and n is the length of the binary string (for each x_i). For example, given the binary string 1011100001, x_1 represented by the first three bits and x_2 by the last seven, and upper and lower bounds for both variables set at 4.0 and -4.0 respectively, int for $x_1 = 5$ and $x_1 = 1.714$, while int for $x_2 = 97$ and $x_2 = 2.110$.

EA binary encodings have identified shortfalls (e.g., Hamming cliffs [17:pg. 229]) so other encodings are often used. Although restricting MOEA genetic representation to binary strings may result in less effective results it does allow for the desired standard comparison between MOEAs. If one algorithm uses real-valued genes its computational grid's "fidelity" is much finer, giving it a search advantage because it is able to "reach" more discrete points in the solution space. Additionally, different computational platforms may allow different resolutions (i.e., different ϵ values – the smallest computable difference between 1 and the next smallest value) and different numbers of distinct values (i.e., how many distinct numbers can be computed).

Thus, even though a binary representation restricts a search space's size it allows for a quantitative MOEA comparison, determination of an MOP's PF_{true} (at some resolution), and an enumeration method for deterministically searching a solution space (see the next section). The underlying resolution may be increased/decreased as desired, at least up to some point where computation becomes impractical or intractable. This methodology is designed for experimentation and used to make judgments about proposed MOEAs and their implementations.

6.3.1.2 Search Space Enumeration. Our enumerative search concept is in part due to a paper suggesting that exhaustive deterministic enumeration may be the only viable approach to solving irregular or chaotic problems [235]. Its authors propose harnessing ever-expanding computational capability to obtain the desired solutions. We constructed such a program executing on parallel high-powered computers whose purpose is to find P_{true} and PF_{true} for several numeric MOPs. The resulting sets are still only a discrete representation of their continuous counterparts, but are the "best possible" at a given computational resolution.

The IBM SP computers at both the Aeronautical Systems Center's Major Shared Resource Center (ASC MSRC) and the U. S. Army Corps of Engineers Waterways Experiment Station's (CEWES) MSRC are used to deterministically enumerate all possible solutions for a given MOP at a given computational resolution as previously defined.[1]

[1]Developmental work was performed on a Sun Network of Workstations (NOWs). The program uses 64-bit accuracy and currently executes on NOWs, Silicon Graphics Origin 2000 and Power Challenge systems, and the IBM SP-2.

The program is written in "C" and uses the Message Passing Interface (MPI) to distribute function evaluations among many processors. A parallel implementation is selected to efficiently process large solution spaces, e.g., 2^{24} and larger. For a given MOP, each processor evaluates some subset of solutions and stores the resultant Pareto optimal solutions and their corresponding nondominated vectors on disk. Noting that Pareto optimality places a partial ordering on the search space, combining the separate solutions/vectors from different processors and again comparing the vectors results in P_{true} ***at that particular computational resolution***. Figure 6.1 illustrates this process; P_{local} is the Pareto optimal set as regards the solutions evaluated by each processor. Program timing and processor loadings may also be recorded to determine problem scaling. This program easily "solves" bi- and tri-objective MOPs of size $2^{24} - 2^{26}$ using 32 or more processors on the SP-2.

Using the P_{true} database various MOEA results may be compared not only against each other, but also against the ***true*** MOP optimum. However, these MOEAs must use a binary encoding and mapping as explained in Section 6.3.1.1. At least for selected MOPs a true quantitative comparison is then possible. This methodology allows absolute performance observations.

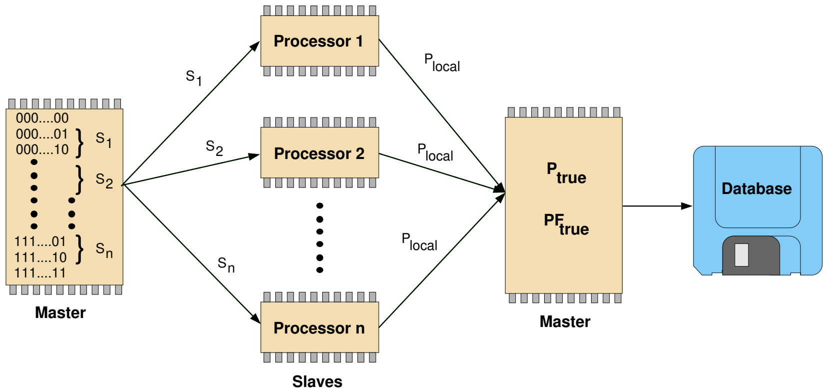


Figure 6.1. Deterministic Enumeration Process

6.3.2 MOEA Test Algorithms. Four MOEAs were selected for testing. These algorithms and their original ***raison d'etre*** are:

1. **MOGA**. Implemented by Fonseca and Fleming [114]. Used to explore incorporation of decision maker goals and priorities in the multiobjective search process.

2. **MOMGA**. Implemented by Van Veldhuizen and Lamont (see Chapter IV). Used to explore use of BBs in constructing MOP solutions where BB desirability may differ with respect to the k functions.

3. **NPGA**. Implemented by Horn et al. [154]. Used to explore benefits of providing P_{known} as input to a Multi Attribute Utility Analysis [177] process.

4. **NSGA**. Implemented by Srinivas and Deb [306]. Used to explore Goldberg's Pareto ranking [126] and preventing bias in exploring the Pareto front.

Rather than describe these algorithms in detail the reader is referred to the literature (for the MOGA [114], NPGA [154], and NSGA [306]) and to Chapter IV. However, we note here that these algorithms were selected because they specifically incorporate what appear to be key theoretical problem/algorithm domain aspects such as Pareto ranking, niching, and fitness sharing (see Section 3.3.2). Other researchers appear to share these thoughts as the MOGA, NPGA, and NSGA (or variants thereof) are the literature's most cited and imitated (see Section 3.4).

The MOGA, NPGA, and NSGA are based on "traditional" GAs; the MOMGA is based on the mGA and can be considered non-standard. However, the conceptual evolutionary process modeled by each algorithm is the same and gives the basis for their direct comparison. Table 6.1 lists each MOEAs' key characteristics which are explained in the next section. Figures 6.2 through 6.4 show the pseudocode for the MOGA, NPGA, and NSGA implementations; MOMGA pseudocode is shown in Figure 4.9 in Section 4.6.

We consider three other algorithms for inclusion in these experiments. These are random search, VEGA, and SPEA. As several MOEA comparisons have shown random search performs *much* worse than other tested algorithms (see Table A.15 in Section A.5.1) we choose not to include it. VEGA is excluded because it is biased towards solutions performing "well" in only one dimension [152], and because several efforts indicate VEGA performs "worse" than their proposed MOEA (see Section A.5.1). Finally, we choose not to include SPEA because of its explicit incorporation of a secondary population in the fitness assignment process [358] which may unfairly impact performance (see Section 3.3.3). Of course, these and other alternative MOEAs may be considered in later experiments.

Table 6.1. Key Experimental MOEA Characteristics

Algorithm	**EVOPs**	**Fitness Assignment**	**Sharing and Niching**	**Population**
MOGA	Crossover and Mutation ($p_c = 1$, $p_m = \frac{1}{0.042}$)	Linear interpolation using Fonseca's [108] Pareto ranking	Phenotypic (σ_{share} - Fitness)	Randomly initialized; $N = 50$
MOMGA	"Cut and splice" ($p_{cut} = 0.02$, $p_{splice} = 1$)	Tournament ($t_{dom} = 3$)	Phenotypic (σ_{share} - Domination)	Deterministically initialized; $N = 100$
NPGA	Crossover and Mutation ($p_c = 1$, $p_m = \frac{1}{0.042}$)	Tournament ($t_{dom} = 5$)	Phenotypic (σ_{share} - Domination)	Randomly initialized; $N = 50$
NSGA	Crossover and Mutation ($p_c = 1$, $p_m = \frac{1}{0.042}$)	"Dummy" fitness using Goldberg's [126] Pareto ranking	Phenotypic (σ_{share} - Fitness)	Randomly initialized; $N = 50$

Although the NFL theorems [346] show there is no "best" EA, certain EAs have been experimentally shown to be more likely effective than others for some real-world problems. Nothing like this has yet been shown for MOEAs. Additionally, no studies have been performed showing which parameters (or parameter values) are key to good performance. In the next section many crucial MOEA components are described in the context of the parameter settings used in these experiments.

6.3.3 Key Algorithmic Parameters. Many EA experiments vary key algorithmic parameters in an attempt to determine the most effective and efficient implementation for a particular problem instantiation or class. A parameter analysis investigating effects of differing parameter values is beyond the scope of these experiments. These experiments' purpose is to determine general MOEA performance and to explore the algorithm domain, not to "tune" MOEAs for good performance on some problem class. These algorithms are then executed with default parameter values as reported in the literature, implementing each MOEA "out of the box" as it were. However, using the term "default" is somewhat of a misnomer as no MOEA parameter value studies are known.

```
Initialize Population
Evaluate Objective Values
Assign Rank Based on Pareto Dominance
Compute Niche Count
Assign Linearly Scaled Fitness
Assign Shared Fitness
For i = 1 to G
    Selection via Stochastic Universal Sampling
    Single Point Crossover
    Mutation
    Evaluate Objective Values
    Assign Rank Based on Pareto Dominance
    Compute Niche Count
    Assign Linearly Scaled Fitness
    Assign Shared Fitness
End Loop
```

Figure 6.2. MOGA Pseudocode

The MOEA literature typically reports using default single-objective EA parameter values, except perhaps for population size. Because MOEAs track a set of solutions, and because more objectives imply the possibility of more Pareto optimal solutions (by definition when using a discrete representation), researchers sometimes enlarge the MOEA's generational population. We again note that these experiments' purpose is MOEA performance comparison and not determination of ideal parameter settings for some (class of) MOPs. If possible, key MOEA parameter values are then kept identical. A discussion of these key parameters follows.

6.3.3.1 Population Initialization. The MOGA, NPGA, and NSGA all use a random population initialization scheme. That is, given some genetic representation, all solutions in the initial generational population are uniformly selected from the solution space. The MOMGA uses a deterministic scheme. For each era (signified by k) the MOMGA generates all possible BBs of size k. Thus, its initial population composition is always known. However, the initial competitive templates are randomly generated.

6.3.3.2 Mating Restriction. As discussed in Section 3.3.2.4, mating restriction has both its proponents and opponents. Existing empirical experimental results

```
Initialize Population
Evaluate Objective Values
For i = 1 to G
    Specialized Binary Tournament Selection
        Only Candidate 1 Dominated: Select Candidate 2
        Only Candidate 2 Dominated: Select Candidate 1
        Both Candidates Dominated or Both Not Dominated:
            Perform Specialized Fitness Sharing
                Return Candidate with Lower Niche count
    Single Point Crossover
    Mutation
    Evaluate Objective Values
End Loop
```

Figure 6.3. NPGA Pseudocode

sometimes indicate it is necessary for good performance, and at other times various MOEA implementations seem to operate well without it. These empirical results indicate the NFL theorems are alive and well [346]. As incorporating mating restriction in some experimental software required major code modifications, and because of its uncertain usefulness in the MOP domain, mating restriction is *not* incorporated in any experimental MOEA.

6.3.3.3 Fitness Assignment. The MOMGA and NPGA employ tournament selection and so require no specific solution fitness manipulation besides those values returned by the MOP fitness function. The MOGA first evaluates all solutions, then assigns fitness by sorting the population on rank ('0' being the best and 'N' the worst – see Equation 3.1 in Section 3.3.2.2). Fitness is assigned linearly to each ordered solution; final fitness is determined by averaging the fitnesses for identically ranked solutions and then performing fitness sharing. The NSGA also evaluates and sorts the population by rank. However, it assigns some large "dummy" fitness to all solutions of the best rank. After fitness sharing it assigns a "dummy" fitness smaller than the current lowest fitness to those solutions of the next best rank, and so on. We note here that all experimental MOEAs employ fitness scaling as each objective dimension's magnitude may be vastly different.

6.3.3.4 Fitness Sharing. All experimental MOEAs incorporate phenotypic-based sharing using the "distance" between objective vectors for consistency. For the

```
Initialize Population
Evaluate Objective Values
Assign Rank Based on Pareto Dominance in Each "Wave"
Compute Niche Count
Assign Shared Fitness
For i = 1 to G
    Selection via Stochastic Universal Sampling
    Single Point Crossover
    Mutation
    Evaluate Objective Values
    Assign Rank Based on Pareto Dominance in Each "Wave"
    Compute Niche Count
    Assign Shared Fitness
End Loop
```

Figure 6.4. NSGA Pseudocode

MOGA and NSGA, σ_{share} is computed and a sharing matrix formed via the standard sharing equation [126]. Finally, fitness sharing occurs only between solutions with the same rank [114, 306].

The NPGA and MOMGA use a slightly different sharing scheme. As explained in Section 4.5.3, two solutions undergoing tournament selection are actually compared against those in a small comparison set. Sharing occurs only if both solutions are dominated or nondominated with respect to the comparison set. A σ_{share} value is used, however, the associated niche count is simply the number of vectors within σ_{share} in phenotypic space rather than a degradation value applied against unshared fitness. The solution with the smaller niche count is selected for inclusion in the next generation. Horn [155] labels this *equivalence class sharing*. An identical scheme is implemented in the MOMGA as it also uses tournament selection. Per Horn's recommendation, continuously updated sharing is used by both the NPGA and the MOMGA due to the observation that chaotic niching behavior may result when combining fitness sharing and tournament selection [154].

σ_{share} represents how "close" two individuals must be in order to decrease each other's fitness. This value commonly depends on the number of optima in the search space. As this number is generally unknown, and because PF_{true}'s shape within objective space is also unknown, we assign σ_{share}'s value using Fonseca's suggested method [114]:

$$N = \frac{\prod_{i=1}^{k}(\Delta_i + \sigma_{share}) - \prod_{i=1}^{k}\Delta_i}{\sigma_{share}^{k}}, \tag{6.2}$$

where N is the number of individuals in the population, Δ_i is the difference between the maximum and minimum objective values in dimension i, and k is the number of distinct MOP objectives. As all variables but one are known σ_{share} can be easily computed. For example, if $k = 2$, $\Delta_1 = \Delta_2 = 1$, and $N = 50$, the above equation simplifies to:

$$\sigma_{share} = \frac{\Delta_1 + \Delta_2}{N - 1} = 0.041. \tag{6.3}$$

This appears a reasonable way to obtain σ_{share} values, although Horn also presents equations bounding PF_{true}'s possible size [154] but leaves the user to choose specific σ_{share} values. Finally, as each MOP's objective values may span widely disparate ranges all objective values are scaled before σ_{share} is computed. This action is meant to prevent unintentional niching bias.

6.3.3.5 Representation and EVOPs. As described in Section 6.3.1.1, the experimental methodology requires each MOEA to use a binary representation. Thus, all MOEAs use an l-bit ($l = 24$) string for each solution and identical minimum/maximum values in each decision variable dimension. Using this scheme ensures identical "reachability" of the test algorithms for a given MOP. The bit length may be increased in later experiments to examine larger search spaces.

However, the MOEAs employ different binary- to real-value mappings. The MOMGA, NPGA, and deterministic enumeration program use the mapping shown in Equation 6.1; the MOGA and NSGA execute as part of a larger program (see Section 6.3.5) that uses a different mapping. This may result in differing mapped values due to truncation or round-off errors as the schemes are implemented.

The mGA's "cut and splice" EVOPs' effect (when both are used) is intended to be similar to recombination's [130]. The MOMGA used mGA default parameters for these operators, namely $p_{cut} = 0.2$ (only one cut allowed per string) and $p_{splice} = 1.0$. There is not yet a "default" MOEA crossover rate but past experiments used crossover probabilities in the range $p_c \in [0.7, 1.0]$ [154, 113, 306]. Thus, the other experimental MOEAs used single-point crossover with $p_c = 1.0$. All but the MOMGA used a mutation rate of $p_m = \frac{1}{l}$ where l is the number of binary digits. The MOMGA did not employ mutation (i.e., $p_m = 0$). As in the original mGA presentation [130], this results in the most stringent possible testing. As mutation is not available to provide diversity and "recreate" BBs, losing a BB from the population means it is gone forever.

6.3.3.6 Termination, Solution Evaluations, and Population Size. When should an MOEA stop executing? The easy answer is after convergence occurs – but when is that? Some "best guess" is normally made and appropriate termination flags set. We do the same in this experimental series and terminate search based on the number of solution evaluations.

Like Goldberg et al. in their early mGA experiments [130], we compare experimental MOEA results derived after an identical number of solution evaluations are performed, using that factor as a measure of common computational effort. However, the number of executed solution evaluations differs between MOMGA runs (even those solving the same MOP) because of internal parameters dynamically governing its operation. In these experiments, the MOMGA is set to execute for three eras and to contain 100 individuals in each juxtapositional population. These values are the mGA defaults, reflecting our desire to execute each experimental MOEA "out of the box" and because Goldberg et al. indicate the juxtapositional generation size should be "about" that of a usual GA [130]. The MOMGA is set to execute a maximum of 20 juxtapositional generations in each era and its execution is terminated before the total number of solution evaluations for a run exceeds 65,536 (2^{16}). Thus, the total fraction of explored search space is then bounded above by $\frac{2^{16}}{2^{24}} = 0.39\%$. Historically, EAs often execute at most tens of thousands of fitness evaluations and this experimental limit is within that range. As it explores only a small fraction of the search space an MOEA's effectiveness should be readily apparent in how well its results (P_{known} and PF_{known}) compare to P_{true} and PF_{true} (if known).

Thus, for all test MOPs, the MOMGA was executed first and the number of executed solution evaluations per run determined. The other MOEAs (each with population size $N = 50$) were then set to execute almost the same number of evaluations (N multiplied by the number of generations), ensuring a very nearly equivalent computational effort for each tested MOEA.

The literature sometimes indicates that more objectives imply a larger generational population size is necessary. However, as these experiments involve only bi- and tri-objective MOPs, population size is left at the suggested single-objective GA default size of 50 [17:pg. 123]. The exception was the MOMGA, instead using the mGA default population size of 100 individuals per juxtapositional generation. We again note these experiments' purpose is to explore MOEA performance and not to determine ideal parameter settings over the test functions.

6.3.4 MOEA Experimental Metrics. What metrics might adequately measure an MOEA's results or allow meaningful comparisons of specific MOEA implementations? Appropriate metrics must be selected upon which to base MOEA performance claims, and as the literature offers few quantitative MOEA metrics, proposed metrics must be carefully defined to be useful. Additionally, no single metric can entirely capture total MOEA performance, as some measure algorithm effectiveness and others efficiency. Temporal effectiveness and efficiency may also be judged, e.g., measuring an MOEA's progress each generation. All may be considered when judging an MOEA. Following are possible metrics developed for use in analyzing these experiments, but they should not be considered a complete list.

The metrics identified in this section measure performance in the phenotype domain. Whereas Benson and Sayin indicate many OR researchers attempt to generate P_{true} (and thus implicitly measure performance in genotype space) [28], MOEA researchers have mainly focused on generating PF_{true} (and thus measure performance in phenotype space). As there is a direct correspondence between solutions in P_{true} and vectors in PF_{true} one method may not be "better" than another. However, we do note that multiple solutions may map to an identical vector.

Although here described in terms of measuring final MOEA performance, many of these metrics may also be used to track performance of generational populations. This then indicates performance during execution (e.g., rate of convergence to the MOEA optimum) in addition to an overall performance metric. Although presented using two-objective examples, these metrics may be extended to MOPs with an arbitrary number of objective dimensions.

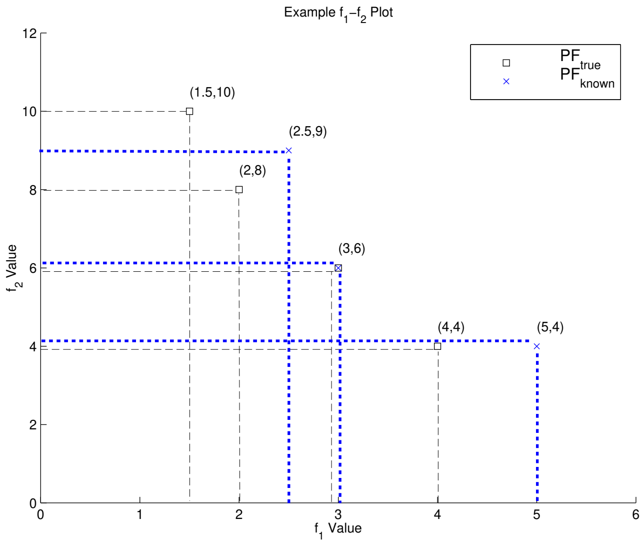


Figure 6.5. PF_{known} / PF_{true} Example

6.3.4.1 Error Ratio. An MOEA reports a finite number of vectors in PF_{known} which are or are not members of PF_{true}. If they are not members of PF_{true} the MOEA has erred or perhaps not converged. This metric is mathematically represented by:

$$E \triangleq \frac{\sum_{i=1}^{n} e_i}{n}, \tag{6.4}$$

where n is the number of vectors in PF_{known} and

$$e_i = \begin{cases} 0 & \text{if vector } i, i = (1, \ldots, n) \in PF_{true}, \\ 1 & \text{otherwise.} \end{cases} \tag{6.5}$$

For example, $E = 0$ indicates every vector reported by the MOEA in PF_{known} is actually in PF_{true} ; $E = 1$ indicates that none are. The example in Figure 6.5 has $E = \frac{2}{3}$. We also note a similar metric [359, 358] measuring the percentage of solutions in some set (e.g., P_{known}) dominated by another solution set's members (e.g., P_{true}).

6.3.4.2 Generational Distance. Used in other experiments [325] this metric is a value representing how "far" PF_{known} is from PF_{true} and is defined as:

$$G \triangleq \frac{(\sum_{i=1}^{n} d_i^p)^{1/p}}{n} , \tag{6.6}$$

where n is the number of vectors in PF_{known} , $p = 2$, and d_i is the Euclidean distance (in objective space) between each vector and the *nearest* member of PF_{true} . A result of 0 indicates $PF_{true} = PF_{known}$; any other value indicates PF_{known} deviates from PF_{true} . The example in Figure 6.5 has $d_1 = \sqrt{(2.5-2)^2 + (9-8)^2}$, $d_2 = \sqrt{(3-3)^2 + (6-6)^2}$, and $d_3 = \sqrt{(5-4)^2 + (4-4)^2}$, and $G = \sqrt{1.118^2 + 0^2 + 1^2}/3 = 0.5$.

Schott proposes a "7-Point" distance measure that is similar to our generational distance [292]. In his experiments neither P_{true} or PF_{true} are known, so he generates seven points (vectors) in objective space for comparison. Assuming a bi-objective minimization MOP and an (f_1, f_2) coordinate system with origin at (0,0), first determine the maximum value in each objective dimension. Two equidistantly spaced points are then computed between the origin and each objective's maximum value (on the objective axis). The "full" measure is then created by averaging the Euclidean distances from each of the seven axis points to the member of PF_{known} closest to each point. Given a general bi-objective minimization MOP $F(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}))$, the seven points are:

$$\begin{aligned}&\{(0, (\max f_2(\vec{x}))/3),\ (0, 2*(\max f_2(\vec{x}))/3),\ (0, (\max f_2(\vec{x}))),\ (0,0),\\ &\quad ((\max f_1(\vec{x}))/3, 0),\ (2*(\max f_1(\vec{x}))/3, 0),\ ((\max f_1(\vec{x})), 0)\}.\end{aligned} \tag{6.7}$$

6.3.4.3 Maximum Pareto Front Error. It is difficult to measure how well some set of vectors compares to another. For example, in comparing PF_{known} to PF_{true} , one wishes to determine how far "apart" the two sets are and how well they conform in shape. This metric determines a maximum error band which when considered with respect to PF_{known}, encompasses every vector in PF_{true}. Put another way, this is the largest minimum distance between each vector in PF_{known} and the corresponding closest vector in PF_{true}. This metric is defined as:

$$ME \triangleq \max_j(\min_i \mid f_1^i(\vec{x}) - f_1^j(\vec{x}) \mid^p + \mid f_2^i(\vec{x}) - f_2^j(\vec{x}) \mid^p)^{1/p}\,, \tag{6.8}$$

where $i = 1, \ldots, n_1$ and $j = 1, \ldots, n_2$ index vectors in PF_{known} and PF_{true} respectively, and $p = 2$. A result of 0 indicates $PF_{known} \subseteq PF_{true}$; any other value indicates at least one vector in PF_{known} is not in PF_{true}. The vectors in Figure 6.5's P_{known} are $1.118, 0$, and 1 units away from the closest vector in P_{true}. Thus, $ME = 1.118$.

6.3.4.4 Hyperarea and Ratio. Zitzler and Thiele propose an MOEA comparative metric [359] which we term ***hyperarea***. Hyperarea defines the area of objective value space covered by PF_{known} (i.e., the "area under the curve"). For example, a vector in PF_{known} for a two-objective MOP defines a rectangle bounded by an origin and $(f_1(\vec{x}), f_2(\vec{x}))$. The union of all such rectangles' area defined by each vector in PF_{known} is then the comparative measure and is defined as:

$$H \triangleq \{\bigcup_i a_i \mid v_i \in PF_{known}\}, \tag{6.9}$$

where v_i is a nondominated vector in PF_{known} and a_i is the hyperarea determined by the components of v_i and the origin. Using the Pareto fronts in Figure 6.5 as an example, the rectangle bounded by $(0,0)$ and $(4,4)$ has an area of 16 units. The rectangle bounded by $(0,0)$ and $(3,6)$ then contributes $(3 * (6-4)) = 6$ units to the measure, and so on. Thus, P_{true}'s $H = 16 + 6 + 4 + 3 = 29$ units^2, and PF_{true}'s $H = 20 + 6 + 7.5 = 33.5$ units^2.

Zitzler and Thiele do note that this metric may be misleading if PF_{known} is nonconvex. They also implicitly assume the MOP's objective space origin coordinates are $(0, \ldots, 0)$, but this is not always the case. The vectors in PF_{known} can be translated to reflect a zero-centered origin, but as each objective's ranges may be radically different between MOPs, optimal H values may vary widely. We thus also propose a ***hyperarea*** *ratio* metric defined as:

$$HR \triangleq \frac{H_1}{H_2}, \tag{6.10}$$

where H_1 is the hyperarea of PF_{known} and H_2 that of PF_{true}. In a minimization problem, this ratio is 1 if $PF_{known} = PF_{true}$ and greater than one if PF_{known}'s hyperarea is larger than PF_{true}'s. The example in Figure 6.5 has an $HR = \frac{33.5}{29} = 1.155$.

6.3.4.5 Spacing. We wish to measure the spread (distribution) of vectors throughout PF_{known}. The experimental MOEAs perform fitness sharing in an attempt to spread each generational population ($PF_{current}(t)$) evenly along the front. Because PF_{known}'s "beginning" and "end" are known, a suitably defined metric judges how well PF_{known} is distributed. Schott [292] proposes such a metric measuring the range (distance) variance of neighboring vectors in PF_{known}. Called *spacing*, he defines this metric as:

$$S \triangleq \sqrt{\frac{1}{n-1}\sum_{i+1}^{n}(\overline{d} - d_i)^2}\,, \tag{6.11}$$

where $d_i = \min_j(\mid f_1^i(\vec{x}) - f_1^j(\vec{x}) \mid + \mid f_2^i(\vec{x}) - f_2^j(\vec{x}) \mid)$, $i, j = 1, \ldots, n$, $\overline{d}$ is the mean of all d_i, and n is the number of vectors in PF_{known}. A value of zero for this metric indicates all members of PF_{known} are equidistantly spaced. We again note (see Section 5.3.1) that the vectors composing PF_{true} in objective space may not be uniformly spaced. The example in Figure 6.5 has an $S = 0.25$.

Some MOPs (e.g., MOP3, MOP4, and MOP6) have PF_{true}'s that are composed of two or more Pareto curves. Including the distance between the endpoints of two successive curves may skew this metric. Thus, for MOPs with this characteristic, the distance corresponding to the "breaks" in the front are removed from the spacing computation. However, this metric may also then be applied to portions of PF_{known} in isolation (those of high interest). Srinivas and Deb [306] define a similar measure expressing how well an MOEA has distributed Pareto optimal solutions over a nondominated region (the Pareto optimal set). This metric is defined as:

$$\iota \triangleq (\sum_{i=1}^{q+1} (\frac{n_i - \overline{n}_i}{\sigma_i})^p)^{1/p} , \tag{6.12}$$

where q is the number of desired optimal points and the $(q+1)$-th subregion is the dominated region, n_i is the actual number of individuals serving the ith subregion (niche) of the nondominated region, $\overline{n}_i$ is the expected number of individuals serving the ith subregion of the nondominated region, $p = 2$, and σ_i^2 is the variance of individuals serving the ith subregion of the nondominated region. They show that if the distribution of points is ideal with $\overline{n}_i$ number of points in the ith subregion, the performance measure $\iota = 0$. Thus, a low performance measure characterizes an algorithm with a good distribution capacity. This metric may be modified to measure the distribution of vectors within the Pareto front. In that case both metrics (S and ι) then measure only uniformity of vector distribution and thus complement the generational distance and maximum Pareto front error metrics.

6.3.4.6 Overall Nondominated Vector Generation and Ratio. The tested MOEAs add $P_{current}$ to P_{known} each generation, possibly resulting in different cardinalities for P_{known}. This metric then measures the total number of nondominated vectors found during MOEA execution and is defined as:

$$ONVG \triangleq \mid PF_{known} \mid . \tag{6.13}$$

Schott [292] uses this metric (although defined over the Pareto optimal set, i.e., $\mid P_{known} \mid$). Genotypically or phenotypically defining this metric is probably a matter of preference, but we again note multiple solutions may map to an identical vector, or put another way, $\mid P_{known} \mid \geq \mid PF_{known} \mid$. Although counting the number of nondominated solutions gives some feeling for how effective the MOEA is in generating desired solutions, it does not reflect on how "far" from PF_{true} the vectors in PF_{known} are. Additionally, too few vectors and PF_{known}'s representation may be poor; too many vectors may overwhelm the DM.

It is difficult to determine what good values for $\mid ONVG \mid$ might be. PF_{known}'s cardinality may change at various computational resolutions as well as differing (perhaps radically) between MOPs. Reporting the ratio of PF_{known}'s cardinality to the discretized P_{true}'s gives some feeling for the number of nondominated vectors found versus how many exist to be found. This metric is then defined as:

$$ONVGR \triangleq \frac{| PF_{known} |}{| PF_{true} |}. \tag{6.14}$$

A value of 1 indicates the MOEA has found the same number of nondominated vectors as exists in PF_{true}. The example in Figure 6.5 has an $ONVG = 3$ and an $ONVGR = 0.75$.

6.3.4.7 Progress Measure. Bäck defines a parameter used in assessing single-objective EA convergence velocity called a *Progress Measure* [17], which quantifies *relative* rather than *absolute* convergence improvement by:

$$P \triangleq \ln \sqrt{\frac{f_{max}(0)}{f_{max}(T)}}, \tag{6.15}$$

where $f_{max}(i)$ is the best objective function value in the parent population at generation i.

To account for the (possible) multiple solutions in P_{known} we modify this definition to the following:

$$RP \triangleq \ln \sqrt{\frac{G_1}{G_T}}, \tag{6.16}$$

where G_1 is the generational distance at generation 1, and G_T the distance at generation T.

6.3.4.8 Generational Nondominated Vector Generation. This metric tracks how many nondominated vectors are produced each MOEA generation and is defined as:

$$GNVG \triangleq | PF_{current}(t) | . \tag{6.17}$$

6.3.4.9 Nondominated Vector Addition. As *globally* nondominated vectors are sought, one hopes to add new nondominated vectors (that may or may not dominate existing vectors) to PF_{known} each generation. This metric is then defined as:

$$NVA \triangleq \mid PF_{known}(t) \mid - \mid PF_{known}(t-1) \mid \ . \tag{6.18}$$

However, this metric may be misleading. A single vector added to PF_{known} (t) may dominate and thus remove several others. PF_{known} (t)'s size may also remain constant for several successive generations even if $GNVG \neq 0$.

6.3.4.10 Additional MOEA Experimental Metrics. Although implemented in the phenotype domain several experimental metrics may also be defined in a genotypic fashion. For example, the error ratio, generational distance, spacing, and overall non-dominated vector generation metrics are valid when modified to reflect a genotypic basis. However, note that decision variable dimensionality may easily exceed the number of objective dimensions, which may require further metric refinement. In addition, Schott uses three other metrics in his thesis effort [292]: cost function evaluations, clone proportion, and total clones identified. These measures are not relevant to the current experiments.

This effort uses the number of function (solution) evaluations as a constant between MOEAs ensuring "equal" computational effort by each; Schott appears interested only in measuring the results of a single MOEA. We currently make no effort to identify clones (previously evaluated solutions) during execution. As shown in the next section these MOEAs execute quickly. When compared to many real-world MOPs, where each fitness evaluation may take from minutes to hours, it makes no sense to incorporate the overhead of clone identification within these experiments. Thomas' use of MOEAs in submarine stern design, where each individual's fitness evaluation took about 10 minutes, is a case where clone identification is more useful [318]. As no clones are identified in these experiments clone proportion is not considered. Later experiments can easily include these and other metrics.

6.3.5 Computational Environment and Implementation. All MOEAs are executed on the same computational platform for consistency. The host is a Sun Ultra 60 workstation with dual 300 MHz processors and 512 MB RAM, running Solaris 2.5.1. Many other computational platforms would suffice but this high-end host offers exclusive access.

The MOMGA and NPGA are extensions of existing algorithms and specific software (the mGA and SGA-C) from the Illinois Genetic Algorithms Laboratory (IlliGAL) [162]. The NPGA is the original code used by Horn in his MOEA research [154, 155]. Both the MOMGA and NPGA are written in "C" and are compiled using the Sun WorkShop Compiler version C 4.2. Much of our associated research and related experimentation employs the GEATbx v2.0 for use with *MATLAB* [255]. This toolbox offers the user several "default" EA instantiations (e.g., real- or binary-valued GA, ES, EP) and excellent visualization output to aid in analysis. GEATbx requires only a limited amount of user effort to implement a specific EA. Thus, the MOGA and NSGA are written as self-contained "m-files" using other pre-defined toolbox routines. They were constructed using definitions given in the literature [114, 306]. These MOEAs are also executed on the Sun platform described previously but within the *MATLAB* 5.2 environment.

Timing results are not of specific experimental concern. However, for all experimental MOPs, each MOEA run executes in a matter of minutes. Empirical observations indicate that the MOMGA and NPGA execute more quickly than the other MOEAs. This result is expected as the latter algorithms are executing via interpretation within the *MATLAB* environment while the former are compiled codes. Further issues are discussed in Section 7.3.2, but we note these MOEAs exhibit roughly the same computational complexity (see Tables 3.2 and 3.3).

6.3.6 Experimental Test Suite MOPs. Several MOPs are substantiated and proposed for use in an MOEA test function suite (Section 5.4). For these experiments' test functions we select the following MOPs: MOP1, MOP2, MOP3, MOP4, and MOP6. These are all bi-objective MOPs and are validated in Section 5.4.

6.4 Summary

This chapter presents an experimental methodology for quantitatively comparing MOEA performance. After motivating the experiments, key methodology components are discussed. The test algorithms and their parameter settings are presented in detail. Several appropriate metrics are proposed, classified, and analyzed, and example values derived. The chapter concludes by discussing the experimental computational environment and selected test problems.

VII. MOEA Experiment Results and Analyses

> You know that I write slowly. This is chiefly because I am never satisfied until I have said as much as possible in a few words, and writing briefly takes far more time than writing at length.
>
> **Karl Friedrich Gauss**

7.1 Introduction

The purpose of these experiments is to compare well-engineered algorithms in terms of effectiveness as regards *carefully selected test problems.* We wish to determine selected MOEA performance over the MOP domain class, to evaluate the usefulness of proposed test functions and metrics, and to record other germane observations arising during experiment execution and result analysis.

This chapter presents the experimental results derived from applying four MOEAs to the proposed MOEA test suite functions. Section 7.2 discusses the experimental results and statistical analyses for the test suite functions identified in Section 6.3.6. Section 7.3 then presents more general observations about these and related experiments.

7.2 MOEA Experiment Approach and Analyses

Appropriate metrics should be selected for use in judging experimental results as concerning MOEA effectiveness (producing an acceptable result). Thus, specific effectiveness metrics are drawn from those discussed in Section 6.3.4 and listed in alphabetical order in Table 7.1. They are initially used to compare *final* MOEA results; in Section 7.3.4 we discuss using these and/or other metrics to investigate an MOEA's efficiency (rate of convergence). These metrics are selected because they initially appear to be the most appropriate indicators of MOEA performance and thus provide a validated basis for MOEA comparison.

All MOP test functions used are formulated as stated in Table 5.3 in Section 5.4; other experimental and algorithmic parameters are as discussed in Section 6.3. For statistical

Table 7.1. Selected MOEA Experimental Metrics

Metric	**Desired Value**
Error Ratio (E)	0
Generational Distance (G)	0
Hyperarea Ratio (HR)	1
Maximum Error (ME)	0
Overall Nondominated Vector Generation ($ONVG$)	$\gg 1$ (MOP Dependent)
Overall Nondominated Vector Generation Ratio ($ONVGR$)	1 (MOP Dependent)
Spacing (S)	0

comparison purposes, the four experimental MOEAs were each executed ten times for each MOP, providing a statistical sample with which to derive metric values. Each MOEA's results for each MOP are separately analyzed followed by a discussion of their performance across all tested MOPs.

A figure containing seven individual graphs is presented for each MOP tested (all figures are located at the chapter's end for ease of comparative evaluation). Each graph's x-axis contains four entries corresponding to each MOEA. Each y-axis is labeled with the graph's measured metric. Note that the y-axis scales may change between metrics and between MOPs, and that graphs (a) through (e) represent metrics where minimum values are desired, whereas graphs (f) and (g) reflect the opposite. For each graph, the metric's value for each MOEA run is represented by a dot ('.') above the appropriate algorithm name. The error bars for each algorithm are 2σ in length ($\mu + \sigma, \mu - \sigma$, with μ the mean and σ the standard deviation). Additionally, Tables 7.3 and 7.4 (located at the chapter's end) give the mean and standard deviation for each MOEA-metric combination.

7.2.1 Bi-Objective MOP Experimental Results. The following sections (7.2.1.1 through 7.2.1.5) discuss MOEA results as applied to a single MOP. Section 7.2.2 presents observations about the experimental metrics and and MOP instantiations; Section 7.2.3 then analyzes MOEA performance across the five bi-objective test suite functions (see Section 5.4).

7.2.1.1 MOP1 Experimental Results. Figure 7.17 presents MOP1's results for each metric and MOEA. We first observe that MOP1's (and the others) reported error ratio (Figure 7.17(a)) is somewhat misleading. As noted in Section 6.3.3.5, even identical binary- to real-value mapping algorithms may give slightly different results on different architectures due to truncation or round-off errors. P_{true} was computed on one architecture (an IBM SP-2); the four MOEAs executed on another (Sun Ultra 60). Additionally, the MOGA and NSGA use a different binary- to real-value mapping than the MOGA and NSGA, as they execute under GEATbx and use its predefined routines. As all computation and metric derivation is performed using double precision, MOEA experiments with MOP1 clearly show these induced errors.

Consider some P_{true} defined as containing all solutions within a given range, i.e., $P_{true} = \{x \mid L \leq x \leq U\}$. Any solution lying within that range but not *exactly* identical to a computed P_{true} solution within it can differ only by some small ϵ. Thus, if a computed Pareto optimal solution $x_c \in P_{known}$ is close but not identical to a solution $x_p \in P_{true}$, then $x_c \in [x_p - \epsilon, x_p + \epsilon]$. Determining an appropriate ϵ value is difficult and we thus choose to evaluate the error metric as originally proposed. For MOP1, note that only the MOMGA returns vectors in PF_{true} resulting in error ratios between 97% and 100%.

The NPGA and NSGA in each run returned only one or two Pareto optimal solutions. Thus, the spacing metric (Figure 7.17(b)) is undefined in those cases because there are zero or one distances d_i; this situation is represented by a value of -1. The MOGA resulted in comparatively large spacing values (about 0.5 to 5) but this can be attributed to the large objective space and relatively few numbers of nondominated vectors in PF_{known} . All MOMGA runs resulted in spacing values between 0 and 0.5.

The generational distance values (Figure 7.17(c)) for the NPGA (near 0 to 325) and NSGA (near 0 to 41) are quite large compared to the other MOEAs whose values are almost all near 0. This is again due to the large objective space. The same holds for the MOEAs' maximum error (Figure 7.17(d)). For MOP1 these two metrics' results as regards the NPGA and NSGA are the same, as many runs returned only one Pareto optimal solution. The large objective space also skews the hyperarea ratio graph due to the NPGA's large values (Figure 7.17(e)). The NSGA's values range between near 0 and 300; MOGA and MOMGA values are between 0 and 4.5.

The $ONVGR$ values (Figure 7.17(f)) indicate only the number of PF_{known} vectors presented versus those in PF_{true}. This metric is driven by the $ONVG$ metric (Figure 7.17(g)). We see the MOMGA always returning more vectors (between 1 and 168) than the other MOEAs, and the MOGA (between 3 and 14) more than the other two. In all cases the NPGA and NSGA return only one or two vectors in PF_{known}. As all MOEAs explicitly seek nondominated vectors this is a somewhat surprising result, but one likely due to the large objective space.

MOP1 was thought to be an "easy" MOP for MOEAs to solve because its P_{true} is convex and is formulated with only one decision variable. It appears this may not be the case, likely due to the very large MOP decision variable bounds resulting in a situation similar to that described in Section 5.3.1. As in that case, the difficulty in solving this continuous MOP instantiation appears due to the extremely small number of computationally discrete points representing MOP1's P_{true}. Although the current bounds may make MOP1 too "hard", smaller bounds make it too "easy." The MOEA literature often presents MOP1 as an example but all other known instantiations use a much smaller search space. For example, Schaffer's original proposition appears to use, and Horn does use, a search space bounded by $\{x \mid x \in [-6, 6]\}$ [289, 154]; Norris and Crossley use $\{x \mid x \in [-10, 10]\}$ [237]. The largest implemented known bounds (besides ours) are $\{x \mid x \in [-1000, 1000]\}$ used by Srinivas and Deb [306]. MOP1's search space is two orders of magnitude larger, and leads to even larger metric results because the objective vectors may take on the following values:

$$\{(f_1, f_2) \mid f_1 \in [0, (10^5)^2] \wedge f_2 \in [0, (-10^5 - 2)^2]\}. \tag{7.1}$$

Analogous to executing several single-objective EAs and selecting the "best overall fitness" found as the final answer, we conclude MOP1's analysis by combining each

MOEAs' run's results, i.e.,

$$PF_{known} = \bigcup_{i=1}^{10} PF_{known_i} \ , \tag{7.2}$$

where each PF_{known_i} was returned by a single MOEA run. Thus, Figure 7.1 visually presents each MOEA's overall *qualitative* performance by plotting MOP1's PF_{true} against each MOEA's respective overall PF_{known}. This figure implies that the MOGA performed "well" in solving MOP1 and the MOMGA "very well", while the NPGA and NSGA did a poor job of "covering" PF_{true}. Table 7.5 (at the chapter's end) gives selected metric values for PF_{true} and PF_{known}.

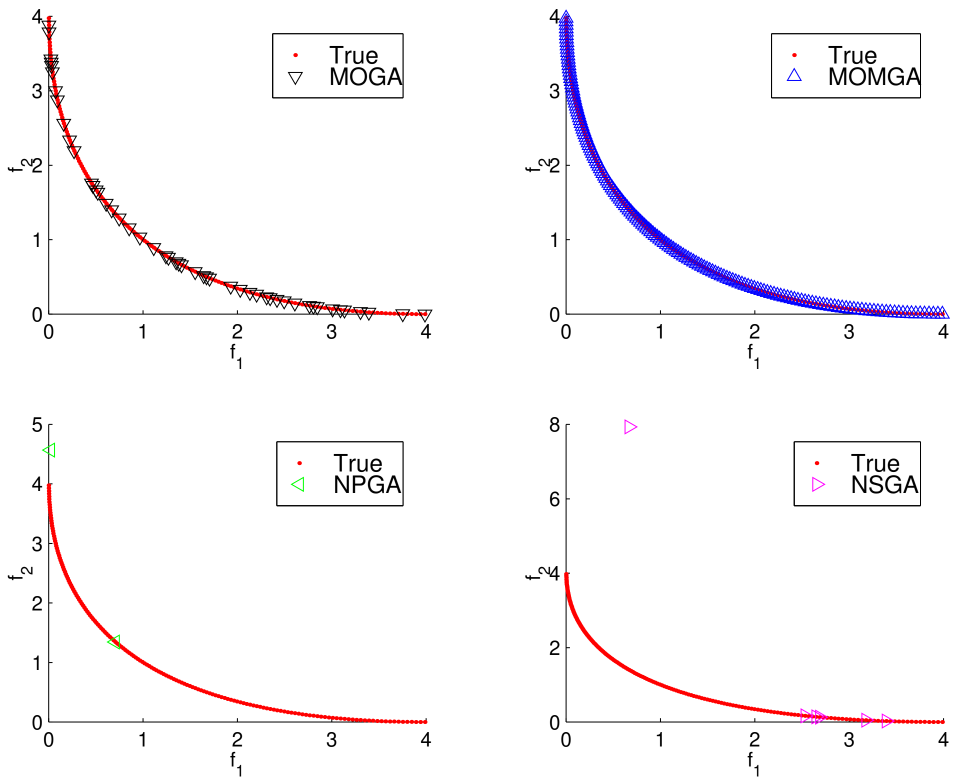


Figure 7.1. MOP1 PF_{known} Comparison

7.2.1.2 MOP2 Experimental Results. Figure 7.18 presents MOP2's results for each metric and MOEA. Only the NPGA returned any vectors in PF_{true}, with error ratios ranging from 92% to 100% (Figure 7.18(a)). As regards spacing (Figure 7.18(b)), the

NSGA consistently returned vectors less evenly distributed (values from 0.08 to 0.12) while the other MOEAs' results were all below 0.03. The same trend is seen when considering generational distance (Figure 7.18(c)). The maximum error results (Figure 7.18(d)) show more consistency in the MOGA and NPGA cases (values from 0.015 to 0.04), while both the MOMGA and NSGA show result values ranging from 0.02 to 0.17. Several MOMGA and NSGA runs returned a vector farther away from PF_{true} than any found in all the MOGA and NPGA runs.

The hyperarea ratios (Figure 7.18(e)) for all MOGA and NPGA runs are consistent and near 1.1, whereas the other two algorithms' results are more varied (from 1.01 to 1.12). However, one NSGA run returns a HR value below one, which is possible because of MOP2's concave PF_{true} (see Section 6.3.4.4). Most notable about the last two metrics (Figures 7.18(f) and 7.18(g)) is that the NSGA again returns far fewer nondominated vectors than the other MOEAs. At most the NSGA returns 20 vectors in PF_{known}, while the other three MOEAs' return from 48 to 131.

Figure 7.2 visually presents each MOEA's overall *qualitative* performance by plotting MOP2's PF_{true} against each MOEA's respective overall PF_{known}. This figure implies all MOEAs except the NSGA performed "very well" in solving MOP2, although the NSGA's returned PF_{known} does "cover" and come close to most of PF_{true}. In general, the figure reflects each MOEA's spacing, generational distance, and $ONVG$ results. However, the NSGA's "worse" results (considering those metrics) are easily seen. Its PF_{known} is not as evenly spaced, the vectors are not as close to PF_{true}, and they do not cover as much of PF_{true} as the other three. Table 7.5 (at the chapter's end) gives selected metric values for PF_{true} and PF_{known}.

7.2.1.3 MOP3 Experimental Results. Figure 7.19 presents MOP3's results for each metric and MOEA. Note that MOP3 is a maximization MOP. Here, the MOGA and NPGA return vector(s) in PF_{true} (Figure 7.19(a)); the NPGA does so consistently with error ratios ranging from 88% to 95%. All MOEAs' spacing results vary (Figure 7.19(b)), but like MOP1 this is expected due to a "larger" objective space. Values here range from near zero to 0.43. Although several of the maximum errors are between 3 and 4

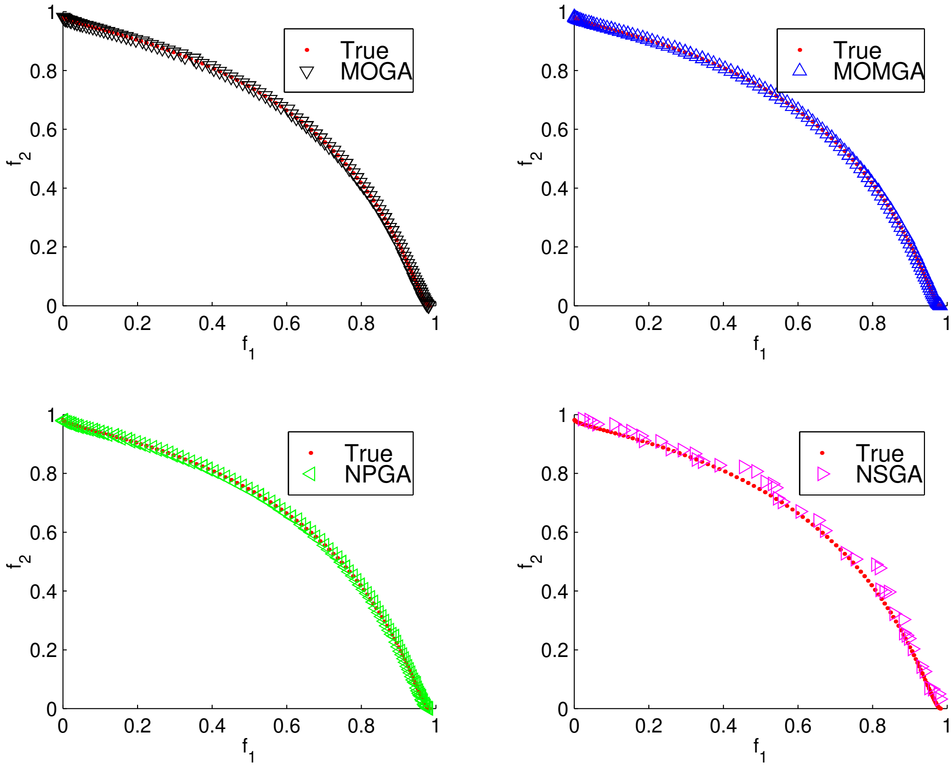


Figure 7.2. MOP2 PF_{known} Comparison

(Figure 7.19(d)), the much larger number of nondominated vectors returned by each MOEA (Figure 7.19(g)) means this metric has a smaller impact on the generational distance (Figure 7.19(c)) than it might otherwise have had. All MOEA runs resulted in "very good" hyperarea ratios (Figure 7.19(e)) near 1.001. However, note that as MOP3 is a maximization problem the metric is actually the inverse of that stated in Section 6.3.4.4.

Because of P_{true} 's cardinality and $ONVG$ values, all $ONVGR$ results (Figure 7.19(f)) are nicely clustered. The MOEAs generally return a few hundred nondominated vectors, although the NSGA in most cases again returns far fewer. Figure 7.3 visually presents each MOEA's overall *qualitative* performance by plotting MOP3's PF_{true} against each MOEA's respective overall PF_{known} . This figure implies that all MOEAs performed "very well" in solving MOP3 as each MOEA's PF_{known} solidly covers PF_{true} . Table 7.5 (at the chapter's end) gives selected metric values for PF_{true} and PF_{known} .

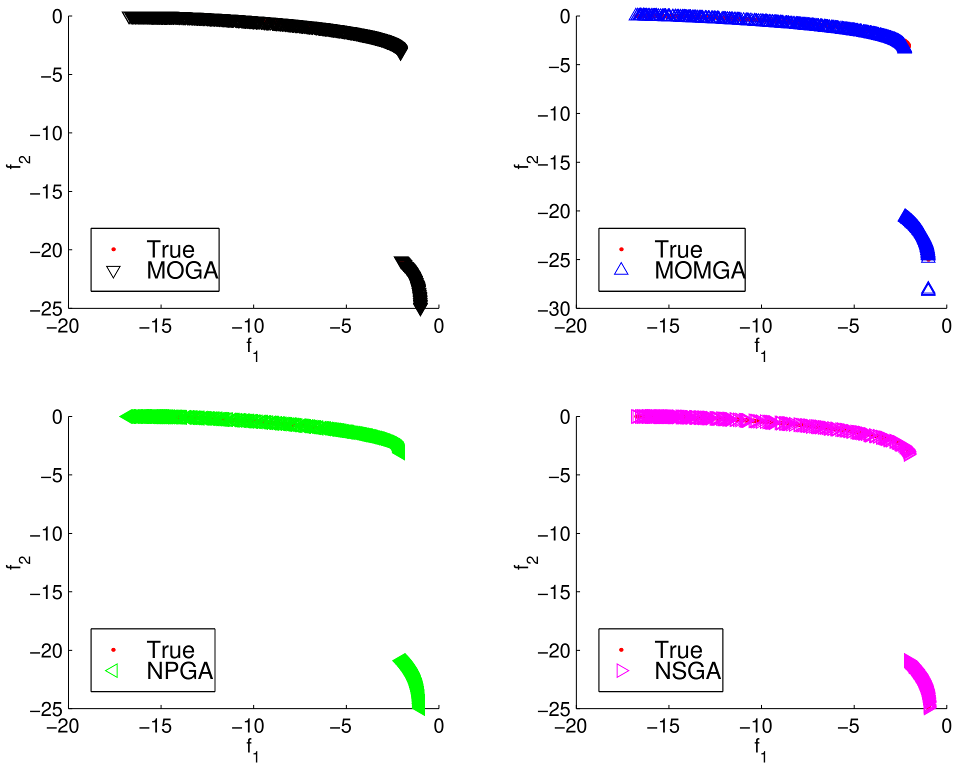


Figure 7.3. MOP3 PF_{known} Comparison

7.2.1.4 MOP4 Experimental Results. Figure 7.20 presents MOP4's results for each metric and MOEA. Only the NPGA returned vectors in PF_{true} with all runs but one returning error ratios from 82% to 95% (Figure 7.20(a)). Spacing results (Figure 7.20(b)) are fairly consistent between runs with all values between 0.1 and 0.7. As this MOP also encompasses a larger objective space the spacing, generational distance (Figure 7.20(c)), and maximum error (Figure 7.20(d)) values appear reasonable, although the MOMGA and NSGA results show more variability between runs. This is also reflected in the hyperarea ratio (Figure 7.20(e)), where all but the NSGA have consistently returned vectors close to PF_{true}. Several HR values are below one due to PF_{known}'s nonconvex shape.

The $ONVGR$ values (Figure 7.20(f)) are again driven by the number of nondominated vectors returned. All but the MOMGA have fairly well clustered $ONVG$ values (Figure 7.20(g)) although the NSGA again returns far fewer. The NSGA returns at most 23 vectors, while the other MOEAs return from 23 to 121. Figure 7.4 visually presents each

MOEA's overall *qualitative* performance by plotting MOP4's PF_{true} against each MOEA's respective overall PF_{known}. This figure implies that the MOGA and NSGA did *not* perform "well" in solving MOP4. The MOMGA and NPGA are close to and do cover PF_{true}; the MOGA and NSGA approximate PF_{true}'s entire shape but their PF_{known} becomes farther from PF_{true} (in distance) as one travels down and right. Table 7.5 (at the chapter's end) gives selected metric values for PF_{true} and PF_{known}. Note that in this case, these values do not necessarily reflect what is concluded visually regarding MOEA performance.

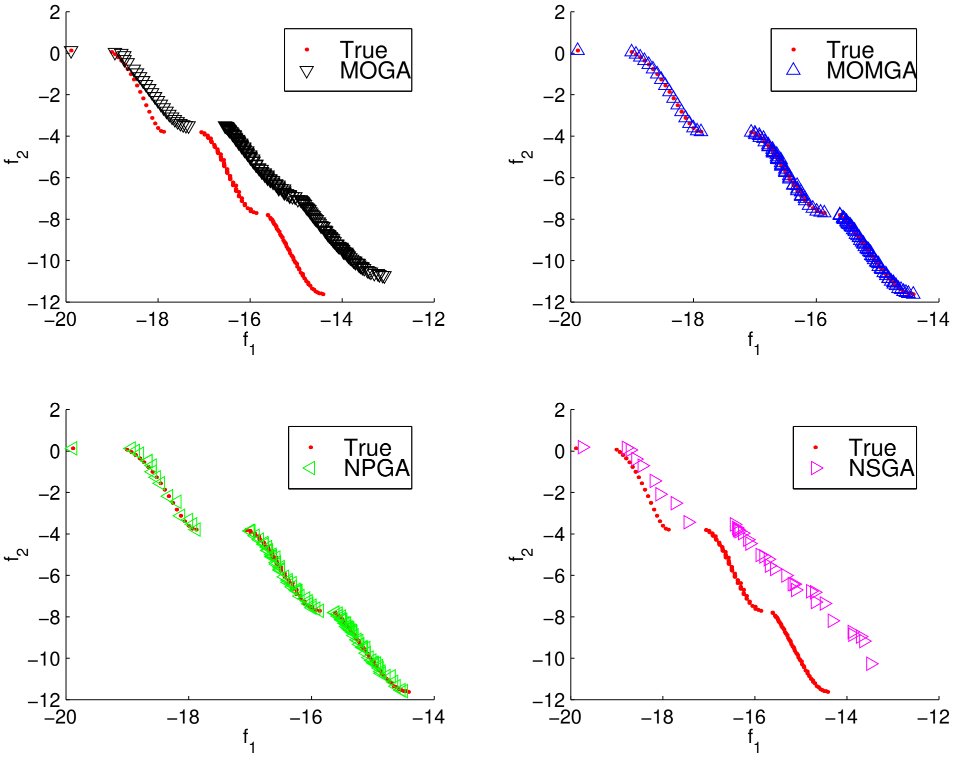


Figure 7.4. MOP4 PF_{known} Comparison

7.2.1.5 MOP6 Experimental Results. Figure 7.21 presents MOP6's results for each metric and MOEA. The MOMGA always returns vectors in PF_{true} while the NPGA often does (Figure 7.21(a)). These are the best values yet seen for the error ratio metric with the MOMGA's error ratio values ranging from 13% to 60%. Spacing results (Figure 7.21(b)) are fairly tight and below 0.1 for all algorithms but the NPGA, whose values range from 0.08 to 0.2. For MOP6, the MOGA and MOMGA return excellent values for generational distance and maximum error (Figures 7.21(c) and Figure 7.21(d)); the other two algorithms have a fairly large spread. The NPGA's and NSGA's generational distance values range from 0.01 to 0.55, and maximum error values from 0.2 to 9.4. This is also reflected by the hyperarea ratio results (Figure 7.21(e)), where we again see several HR values less than one. Finally, $ONVGR$ (Figure 7.21(f)) and $ONVG$ results (Figure 7.21(g)) show the MOMGA returning significantly more nondominated vectors (max of 443), followed by the MOGA (max of 121), NSGA (max of 39), and the NPGA (max of 27).

Figure 7.5 visually presents each MOEA's *overall* performance by plotting MOP6's PF_{true} against each MOEA's respective overall PF_{known}. This figure implies that all MOEAs performed "well" in solving MOP6, although the NPGA and NSGA report a total of three vectors in PF_{known} that are not in PF_{true}. Table 7.5 (at the chapter's end) gives selected metric values for PF_{true} and PF_{known}.

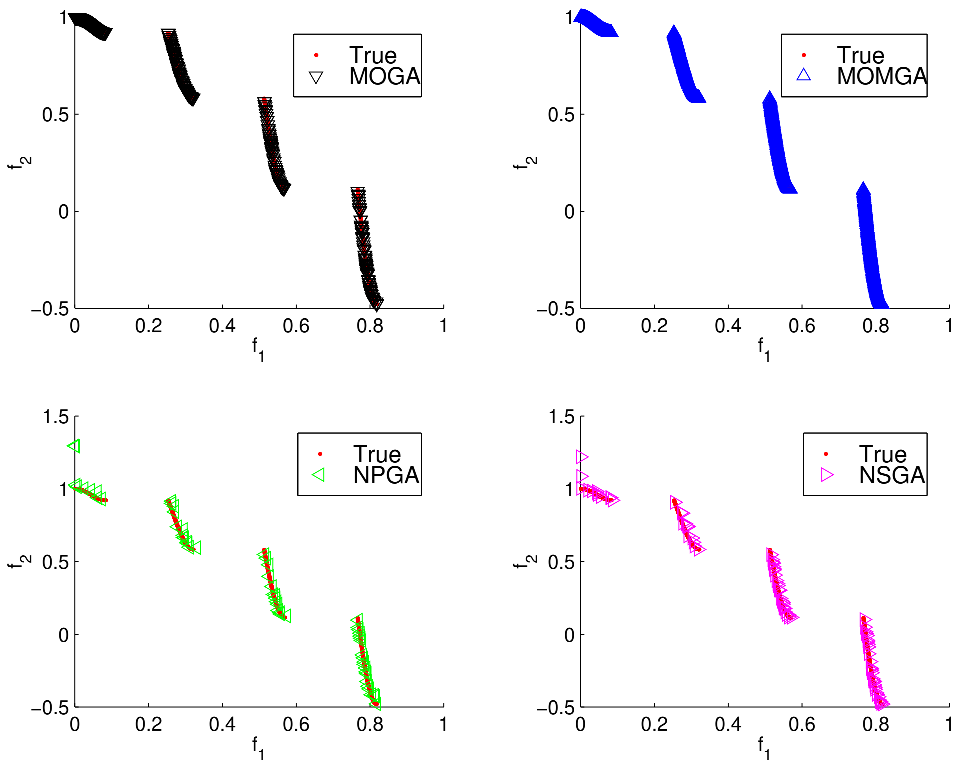


Figure 7.5. MOP6 PF_{known} Comparison

7.2.2 MOEA Experimental Metrics and MOPs. Although the experimental metrics and MOPs were previously theoretically validated, these experiments highlight practical implementation difficulties. The discussions in Sections 7.2.1.1 through 7.2.1.5 indicate some metrics appear not as valuable as others. For example, the error ratio metric reports values for only two of the four algorithms, thus preventing a general comparison. The maximum error metric shows how far one vector is from P_{true}, but the derived value does depend on the objective space's size within which distance is being measured. The hyperarea ratio metric sometimes gives misleading values (when applied to nonconvex Pareto fronts) and requires the inverse value of a maximization MOP. Finally, the $ONVGR$ is largely dependent upon the number of vectors in PF_{true}. Therefore, using their current definitions, we consider spacing, generational distance, and $ONVG$ as the most meaningful metrics for analysis. Although generational distance is dependent upon objective space size, spacing and $ONVG$ values may not be as much so.

Taken overall, the selected test suite MOPs appear useful in practice as well as in theory. We do make the following recommendations, however. Due to the difficulties discussed in Section 7.2.1.1 (primarily concerning metric values), further experiments incorporating MOP1 should use smaller decision variable bounds, e.g., $\{x \mid x \in [-1000, 1000]\}$. MOP2 should incorporate additional decision variables to introduce further dimensional complexity. MOP4 appears to be an "MOEA challenging" problem and should be investigated further. Incorporate additional decision variables in this MOP, also.

7.2.3 Overall Experimental Statistical Analyses. Figures 7.17 – 7.21 imply each algorithm's observations are *not* normally distributed, and that the variance is noticeably different for different MOEAs. This data may not satisfy necessary assumptions for parametric mean comparisons and we thus consider non-parametric statistical techniques for analyzing these experimental results. Based on the previous discussion we perform these tests only on the generational distance, $ONVG$, and spacing metrics.

The Kruskal-Wallis H-Test requires no assumptions about the probability distributions being compared [213]. However, other assumptions must be satisfied in order to apply this test: that five or more measurements are in each sample and that the samples are random and independent; and that the probability distributions from which the samples are drawn are continuous. The results presented in Sections 7.2.1.1 – 7.2.1.5 meet this criteria so we test the following hypotheses:

H_0: The probability distributions of MOGA, MOMGA, NPGA, and NSGA results applied to MOP$\mathbf{X}$ are identical.

H_a: At least two of the experimental MOEAs' result distributions differ.

Table 7.6 (at the chapter's end) shows the Kruskal-Wallis H-Test results for each MOP. The listed values are p-values, also called the observed significance levels, for the Kruskal-Wallis H-Test. We reject the null hypothesis whenever $p \leq \alpha$. Using a significance level $\alpha = 0.1$, we in all cases see there is enough evidence to support the alternative hypothesis and conclude that for each MOP and recorded metrics, at least two MOEAs' results' distributions differ.

This result allows use of the Wilcoxon rank sum test in comparing the results of MOEA "pairs," attempting now to determine which of a given two MOEAs does "better." This test assumes the sample of differences is randomly selected and that the probability distributions from which the sample of paired differences is drawn is continuous [213]. The results presented in Sections 7.2.1.1 – 7.2.1.5 meet this criteria so we test the following hypotheses:

H_0: The probability distributions of MOEA_1 and MOEA_2 results applied to MOP$\mathbf{X}$ are identical.

H_a: The probability distributions differ for the two MOEAs.

There are six possible MOEA pairings: MOGA and MOMGA, MOGA and NPGA, MOGA and NSGA, MOMGA and NPGA, MOMGA and NSGA, and NPGA and NSGA. If c Wilcoxon rank sum tests are performed with an overall level of significance α, the Bonferroni technique [234] allows us to conduct each individual test at a level of significance $\alpha^* = \alpha/c$ [213]. For these tests we select an overall significance level $\alpha = 0.2$ due to the fact we have only 10 data points per MOEA. Thus, $\alpha^* = 0.2/6 = 0.0\overline{3333}$.

Table 7.6 (at the chapter's end) also shows the Wilcoxon rank sum tests for each MOEA pair. The listed values are p-values, also called the observed significance levels, for the Wilcoxon rank sum test. We reject the null hypothesis whenever $p \leq \alpha^*$. These tests show the majority of pairwise MOEA comparisons provide enough evidence to support the alternative hypothesis, and we can thus conclude in those cases that there is a significant statistical difference between the MOEAs. We are then able to make conclusions about each MOEA's results as regards each of the three metrics. For each metric, a figure presenting mean metric performance (μ) is plotted for each MOP by algorithm with error bars as above.

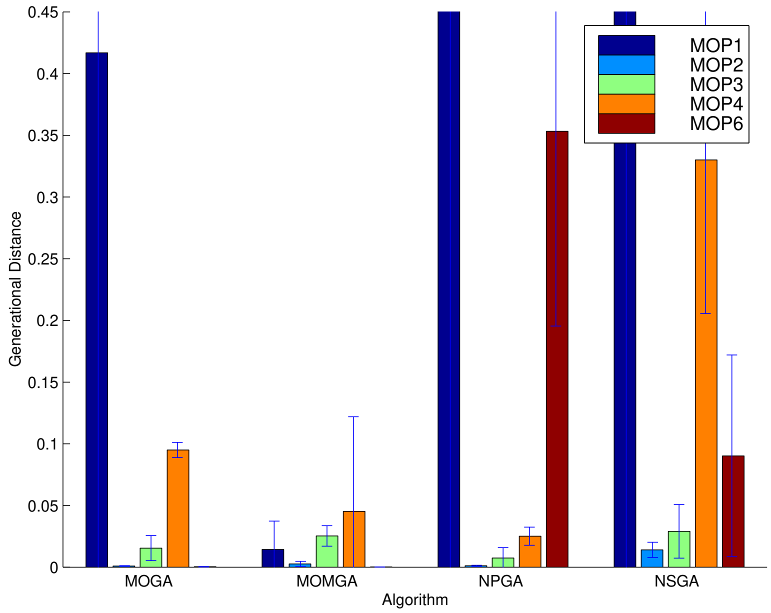


Figure 7.6. Overall Generational Distance Performance

7.2.3.1 Generational Distance Statistical Analysis. Figure 7.6 presents MOEA performance as regards generational distance. The MOP1 values for both the NPGA and NSGA were large enough to skew the results when viewed in this format – the graph truncates those two bars (their respective results are $G \approx 66$ and $G \approx 11$). The pairwise Wilcoxon rank sum tests indicate the following MOEA pairs' results are statistically insignificant: (**MOP1**) MOGA and MOMGA, MOGA and NSGA, NPGA and NSGA; (**MOP2**) MOGA and MOMGA, MOGA and NPGA, MOMGA and NPGA;

(**MOP3**) MOGA and NPGA, MOGA and NSGA, MOMGA and NSGA; and (**MOP4**) MOMGA and NPGA. All other results are statistically significant.

This allows us to state that in general, when considering generational distance the MOGA, MOMGA, and NPGA gave better results than the NSGA over the test suite problems. This is certainly true for MOP4; the MOGA and MOMGA perform much better than the other two MOEAs on MOP6.

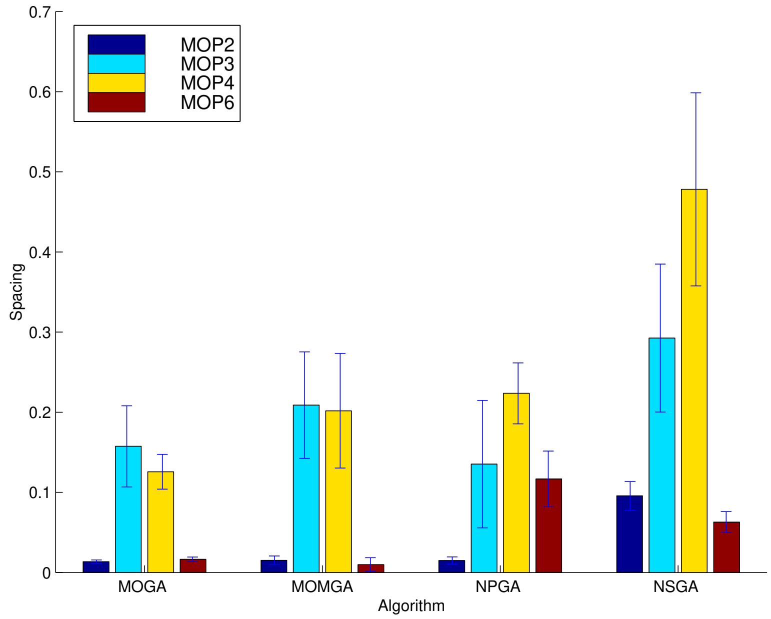


Figure 7.7. Overall Spacing Performance

7.2.3.2 Spacing Statistical Analysis. Figure 7.7 presents overall MOEA performance as regards spacing. This graph does not report MOP1 results as they are (possibly) somewhat misleading (see Section 7.2.1.1). The pairwise Wilcoxon rank sum tests indicate the following MOEA pairs' results are statistically insignificant: (**MOP2**) MOGA and MOMGA, MOGA and NPGA, MOMGA and NPGA; (**MOP3**) MOGA and NPGA, MOMGA and NPGA, MOMGA and NSGA; and (**MOP4**) MOMGA and NPGA. All other results are statistically significant.

This allows us to state that in general, when considering spacing the NSGA gave worse results over the test suite problems. This is certainly true for MOP2 and MOP4; the MOGA and MOMGA perform much better than the other two MOEAs on MOP6.

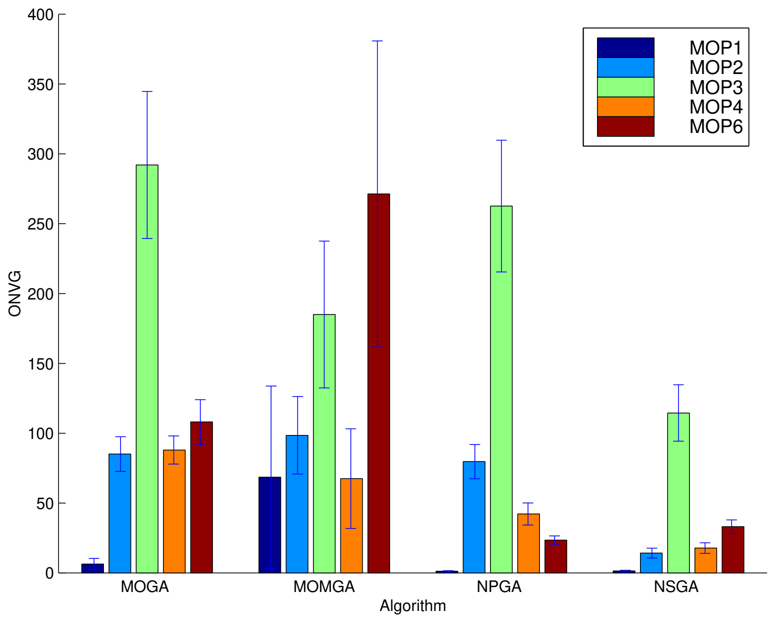


Figure 7.8. Overall $ONVG$ Performance

7.2.3.3 ONVG Statistical Analysis. Figure 7.8 presents overall MOEA performance as regards $ONVG$. The pairwise Wilcoxon rank sum tests indicate the following MOEA pairs' results are statistically insignificant: (**MOP1**) MOGA and MOMGA, NPGA and NSGA; (**MOP2**) MOGA and MOMGA, MOGA and NPGA, MOMGA and NPGA; (**MOP3**) MOGA and NPGA; and (**MOP4**) MOGA and MOMGA, MOMGA and NPGA. All other results are statistically significant.

This allows us to state that in general, when considering $ONVG$ the NSGA again gave worse results over the test suite problems. Disregarding MOP1 allows us to state the these three algorithms *always* outperformed the NSGA. This is a surprising result. Although sometimes performing worse (when considering spacing and generational distance), the NSGA generally returned values "close" to the other algorithms. In this case the NSGA consistently returns fewer (often less than half) the number of nondominated vectors than the other algorithms. We highlight this result as we are attempting to provide a DM with a number of choices represented by the nondominated vectors composing PF_{known} .

7.3 MOEA Experiment Observations

A number of related experiments are executed in support of those analyzed in this chapter. Selected results and observations gleaned through the experimental process are reported in this section.

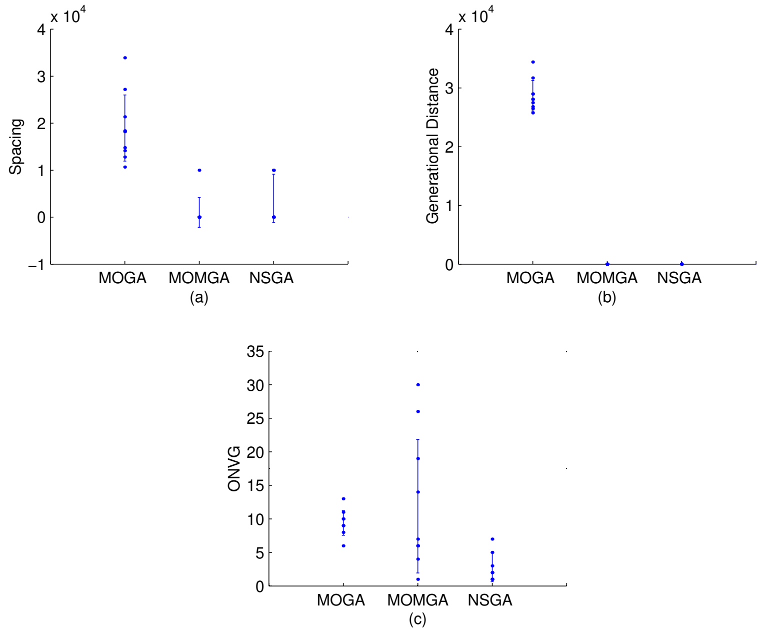


Figure 7.9. MOP7 Metrics

7.3.1 MOP7 Experimental Results. Figure 7.9 presents MOP7's results for three MOEAs as regards three metrics. MOP7 was selected as it is a tri-objective MOP, illustrating that MOEAs and experimental metrics can be extended to MOPs with more objectives. However, as the NPGA code used in these experiments is currently limited to two objectives, it was not used in solving MOP7.

Figures 7.9(a) and Figure 7.9(b) imply that MOP7's objective space is quite large. It also appears that the MOMGA and NSGA performed better than the MOGA as concerning spacing and generational distance, although further statistical analysis is necessary to state that with finality. As the scale in Figure 7.9(b) is quite large, we note that the MOMGA's results range from nearly 0 to about 1.07 and the NSGA's from 0.07 to about 20. The MOEAs in general return fewer nondominated vectors than in the other MOP experiments (Figure 7.9(c)).

Figure 7.10 visually presents each MOEA's overall *qualitative* performance by plotting MOP7's PF_{true} against each MOEA's respective overall PF_{known}. This figure implies that the MOMGA performed "well" in solving MOP7 and the others did not. Table 7.5 (at the chapter's end) gives selected metric values for PF_{true} and PF_{known}.

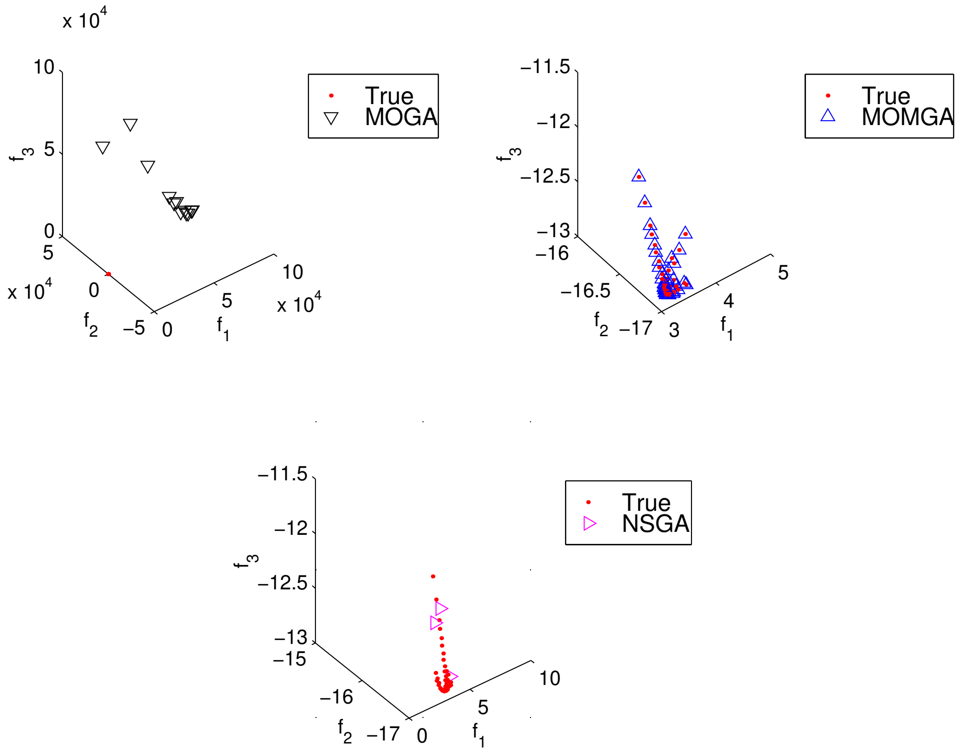


Figure 7.10. MOP7 PF_{known} Comparison

7.3.2 Experimental Timing Analysis. Section 6.3.5 intimates that the MOMGA and NPGA execute more quickly than the MOGA and NSGA. Although probably still true, further analysis determines the primary reason behind these observations. As previously indicated, the MOGA and NSGA are implemented within a *MATLAB* toolbox. Part

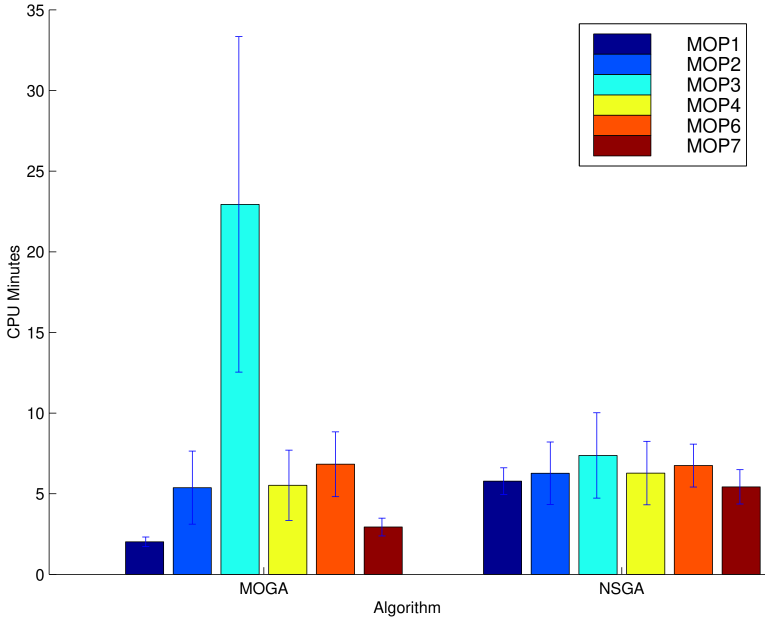


Figure 7.11. MOEA Timing

of the toolbox's default output includes CPU and wall-clock execution time. Figure 7.11 reports mean CPU timing results (μ) for each MOP tested with error bars as above.

It is seen that the MOGA's mean values are generally less than those of the NSGA, although MOP3 is an exception. A straightforward reason exists for this. All MOEAs employed a secondary population. Whereas the MOMGA and NPGA append $P_{current}$ to a file containing P_{known} each generation, the MOGA and NSGA update P_{known} as part of their generational loop. As determination of Pareto optimality is $\mathcal{O}(n^2)$, and as the MOGA always returns more nondominated vectors than the NSGA (see Figure 7.8), its actual execution times are most likely actually much lower (as are the NSGA's). Coello Coello also notes the NSGA's large execution time [67:pp. 187-189]. However, even if the MOGA's and NSGA's logic is changed to reflect that of the MOMGA and NPGA (or vice versa), we believe the MOMGA and NPGA are still the faster running MOEAs because they are machine executables.

Table 7.2. Current Experimental Code Status

Characteristic	**MOGA**	**MOMGA**	**NPGA**	**NSGA**
Encoding	Real values or Binary	Binary	Binary	Real Values or Binary
Execution Speed (Empirical)	Slower	Faster	Fastest	Slowest
# Functions	Arbitrary	2 or 3	2	Arbitrary
Language (Platform)	GEATbx ($MATLAB$)	"C"	"C"	GEATbx ($MATLAB$)
Sharing	Dynamic	Dynamic	Fixed	Dynamic

7.3.3 Experimental MOEA Implementations. Although each is instantiated in the same manner for experimental purposes, each of the current MOEA codes has different capabilities, summarized in Table 7.2. We note the following about its entries.

$MATLAB$ offers a compiler option allowing "m-files" to be directly compiled into "C" source code. Thus, the MOGA and NSGA could theoretically be translated into a "C" version. This was not attempted. Extensive modification is necessary to extend either the MOMGA or NPGA to employ real-valued genetic representations. Minor modifications would allow the MOMGA and NPGA to also handle an arbitrary number of functions composing an MOP. The execution speed ratings are empirical and based upon the *existing* implementations. Finally, the NPGA requires key sharing parameters (e.g., σ_{share}) to be entered at the beginning of each run; extensive modification is required to allow dynamic NPGA σ_{share} determination. Although the other experimental MOEAs use the NPGA's values for consistency, their current formulations allow for dynamic determination of these values each generation.

7.3.4 Additional Experimental Metrics. As discussed in Section 6.3.4, some experimental metrics may also be used to track MOEA generational performance. We provide examples here for further insight. All results are from an arbitrary MOP7 experimental run; we again note that the NPGA is not used in solving MOP7.

Figure 7.12 presents $GNVG$ results (see Section 6.3.4.8). As the MOMGA records $P_{current}$ each juxtapositional generation, we see values for 13 generations each in eras 1

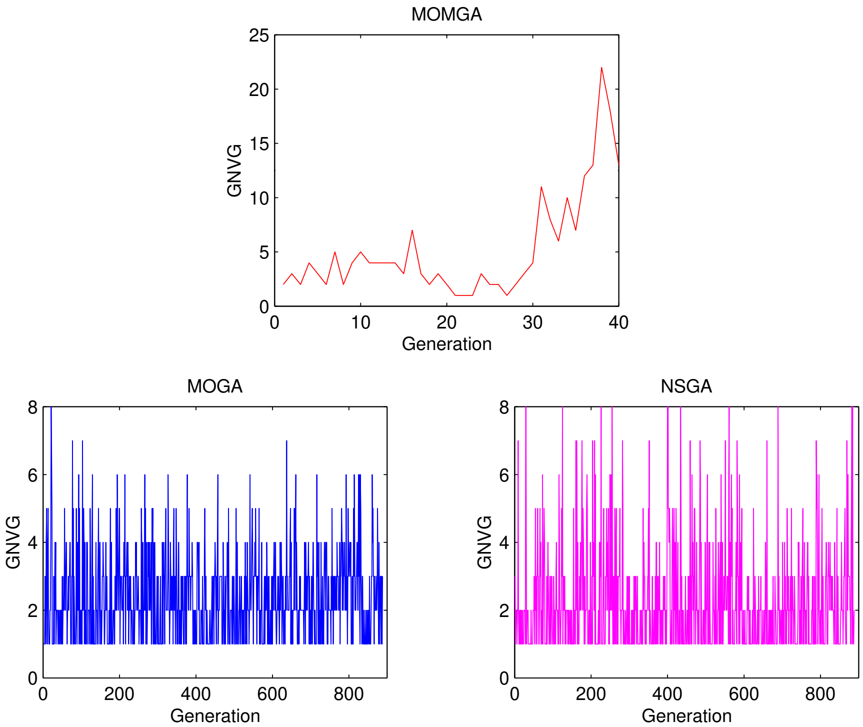


Figure 7.12. $GNVG$

and 2, and 14 in era 3. Note that the number of nondominated vectors is higher near the algorithm's end and that it may produce two to three times the number of vectors as the MOGA and NSGA. This is most likely due to the MOMGA's larger juxtapositional population size (100 individuals vs. 50). The latter MOEAs record results *every* generation. Their results are plotted using identical scales; only three NSGA generations reported a $GNVG$ above 8 (two of 9 and 1 of 11).

Only one citation in the literature [329] reports any quantitative results concerning the number of "waves" or fronts produced (per generation) by implementing Goldberg's Pareto ranking scheme (see Section 3.3.2.2). We present a similar graph (Figure 7.13) showing the number of waves may vary significantly between generations. As this computation may be significant, it is most likely the main reason why the NSGA is the slowest MOEA tested (shown in Section 7.3.3). Figure 7.14 shows NVA values during a MOGA run. As described in Section 6.3.4.9, we see this metric's value may remain steady for several generations and sometimes shows a net loss of solutions from PF_{known}.

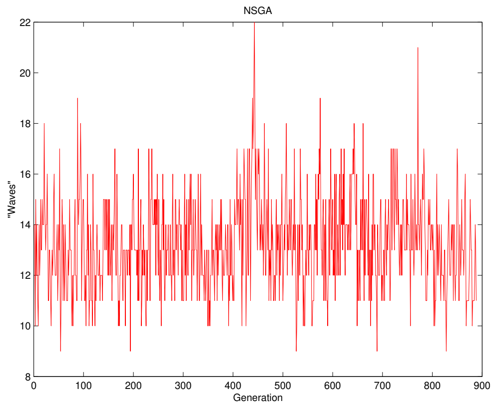


Figure 7.13. NSGA "Waves"

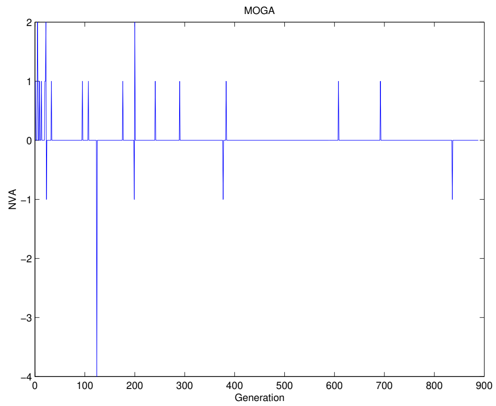


Figure 7.14. MOGA NVA

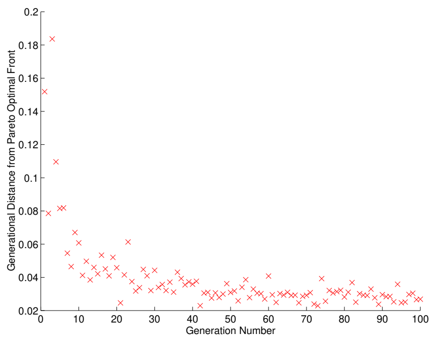


Figure 7.15. $PF_{current}$'s G

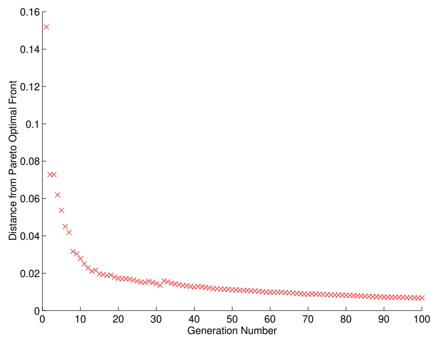


Figure 7.16. PF_{known} 's G

Figures 7.15 and 7.16 are reproduced from a previous publication [325] investigating MOEA convergence. The generational distance metric (G, discussed in Section 6.3.4.2) is used there to indicate an MOEA's convergence to PF_{true} . One expects G to generally decrease during execution if convergence is occurring, although the contextual nature of determining Pareto dominance is clearly reflected by the results in Figure 7.15.

7.4 Summary

This chapter presents MOEA experimental results and analyses. After describing the presentation approach, selected results are presented for five test functions. Detailed discussions are teamed with appropriate figures to judge MOEA effectiveness. After substantiating the choice of three practical metrics (generational distance, overall nondominated vector generation, and spacing), nonparametric statistical analyses then show the

NSGA performance to be statistically worse than the other tested MOEAs (over the test problems). Concluding the chapter are selected results and observations of related experiments.

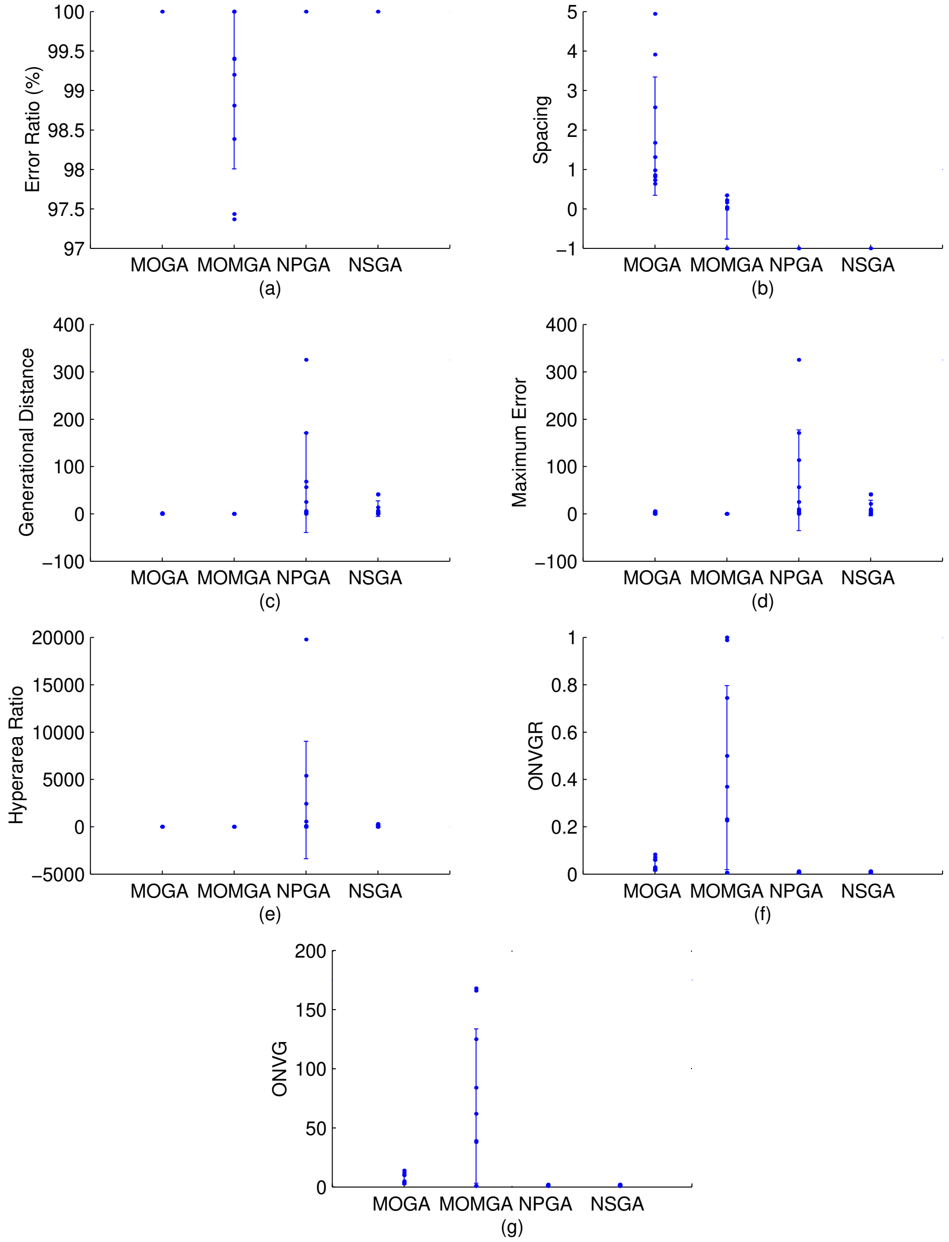


Figure 7.17. MOP1 Metrics

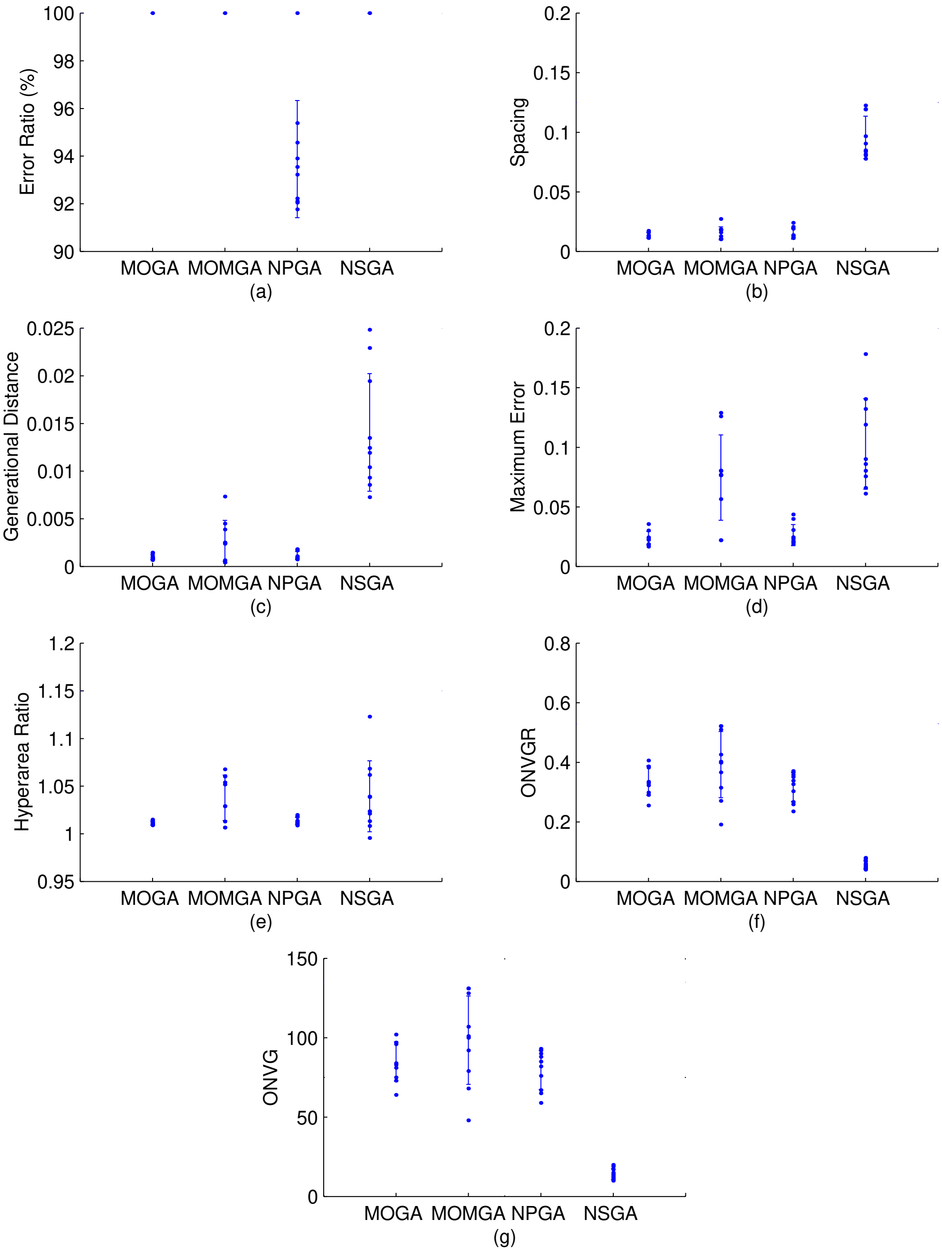


Figure 7.18. MOP2 Metrics

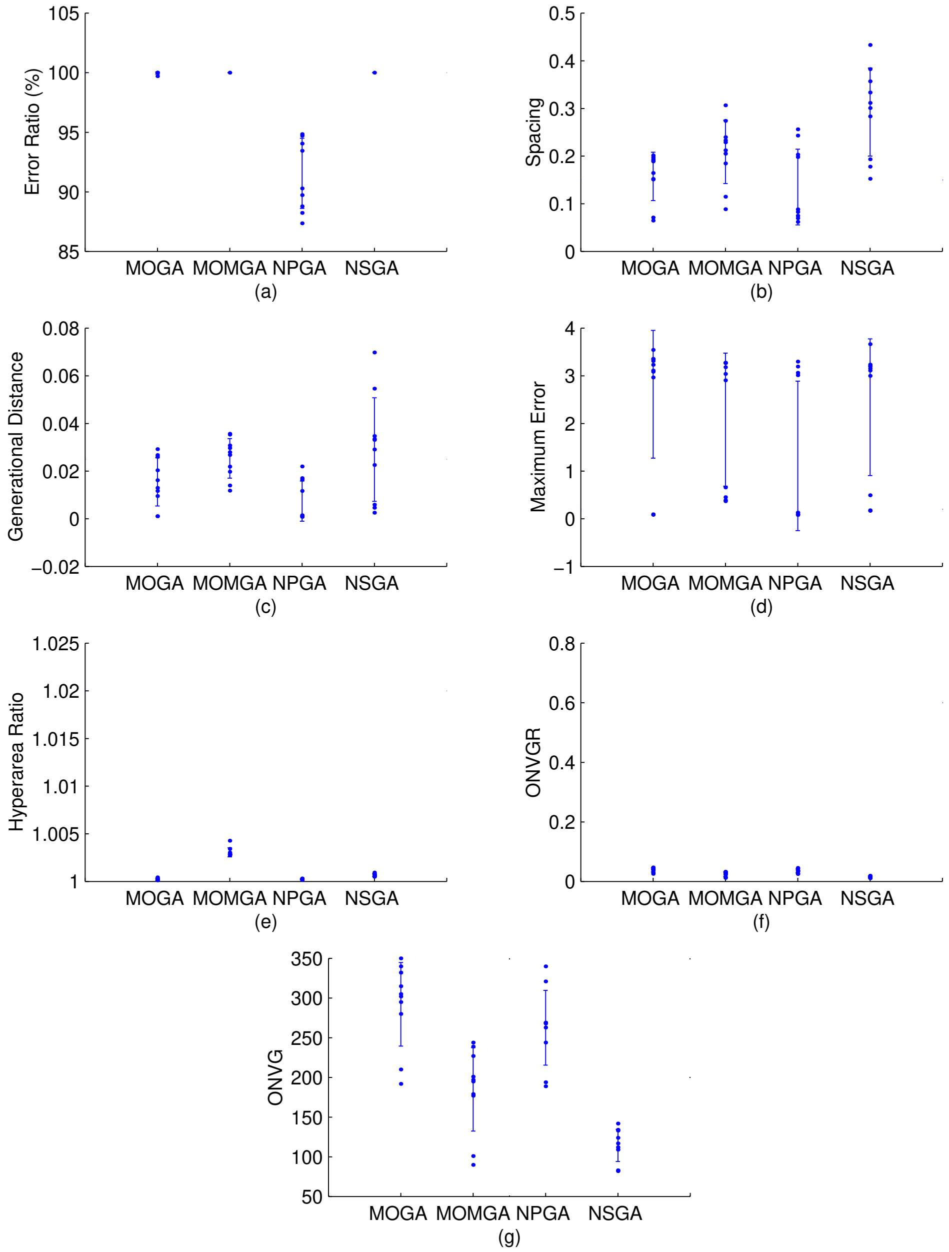


Figure 7.19. MOP3 Metrics

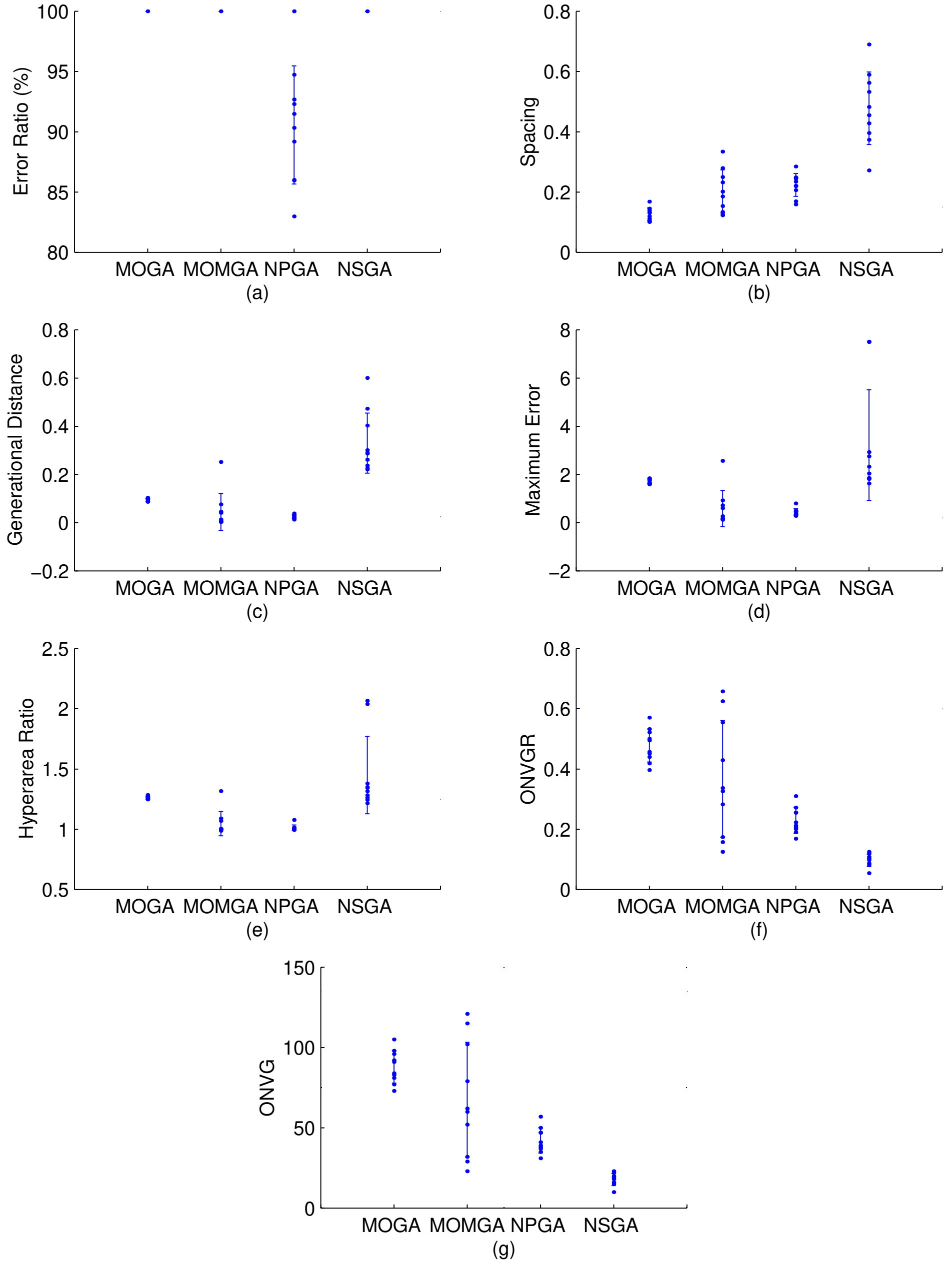


Figure 7.20. MOP4 Metrics

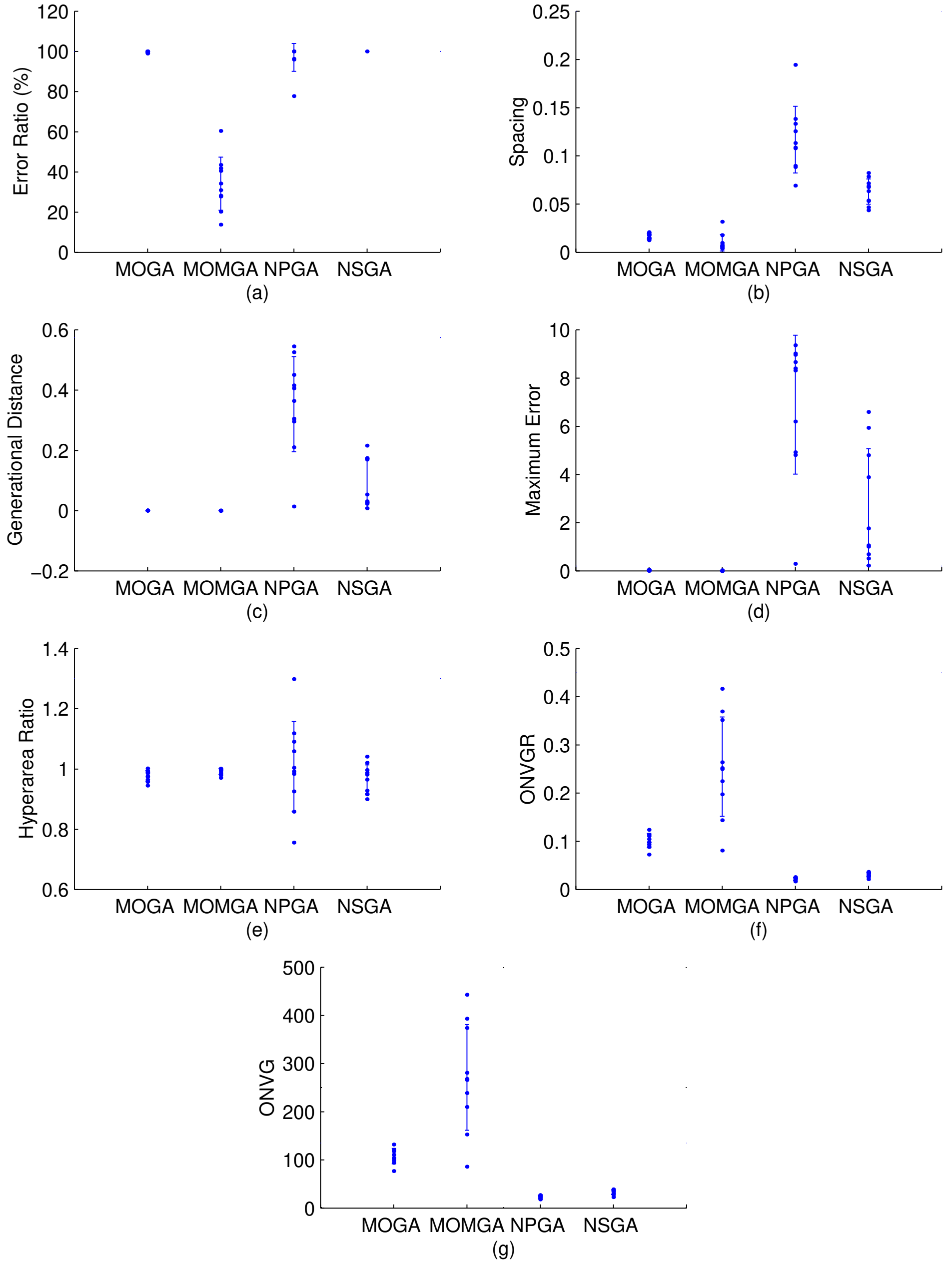


Figure 7.21. MOP6 Metrics

Table 7.3. Experimental Statistics (1)

MOP	MOEA	E		G		HR		ME	
		μ	σ	μ	σ	μ	σ	μ	σ
MOP1	MOGA	1.0000	0.0000	0.4169	0.6201	1.2768	1.3887	1.6173	2.0470
	MOMGA	0.9900	0.9934	0.0143	0.0230	0.6026	0.4450	0.0191	0.0246
	NPGA	1.0000	0.0000	66.0867	105.4002	2830.1963	6207.2673	71.0911	106.1471
	NSGA	1.0000	0.0000	11.1514	16.3387	71.9118	119.9754	12.5316	16.3872
MOP2	MOGA	1.0000	0.0000	0.0010	0.0003	1.0113	0.0020	0.0237	0.0057
	MOMGA	1.0000	0.0000	0.0027	0.0021	1.0378	0.0238	0.0747	0.0358
	NPGA	0.9387	2.4602	0.0012	0.0004	1.0136	0.0037	0.0263	0.0088
	NSGA	1.0000	0.0000	0.0141	0.0062	1.0393	0.0372	0.1029	0.0380
MOP3	MOGA	0.9997	0.0930	0.0155	0.0101	1.0002	0.0001	2.6134	1.3397
	MOMGA	1.0000	0.0000	0.0254	0.0083	1.0031	0.0005	2.0811	1.3952
	NPGA	0.9155	2.9437	0.0074	0.0084	1.0002	0.0000	1.3215	1.5682
	NSGA	1.0000	0.0000	0.0291	0.0218	1.0007	0.0001	2.3410	1.4345
MOP4	MOGA	1.0000	0.0000	0.0950	0.0062	1.2618	0.0122	1.7142	0.0986
	MOMGA	1.0000	0.0000	0.0453	0.0767	1.0468	0.1006	0.5869	0.7515
	NPGA	0.9056	4.9024	0.0251	0.0073	1.0123	0.0238	0.4374	0.1448
	NSGA	1.0000	0.0000	0.3301	0.1245	1.4499	0.3218	3.2195	2.2967
MOP5	MOGA	0.9981	0.4024	0.0004	0.0001	0.9777	0.0177	0.0266	0.0166
	MOMGA	34.1612	13.2212	0.0001	0.0001	0.9920	0.0100	0.0076	0.0081
	NPGA	0.9700	6.9441	0.3533	0.1579	1.0085	0.1490	6.8957	2.8793
	NSGA	1.0000	0.0000	0.0902	0.0818	0.9653	0.0483	2.6476	2.4228

Table 7.4. Experimental Statistics (2)

MOP	**MOEA**	*ONVG*		*ONVGR*		*S*	
		μ	σ	μ	σ	μ	σ
MOP1	MOGA	6.3000	4.0838	0.0375	0.0243	1.8436	1.4984
	MOMGA	68.5000	65.2776	0.4077	0.3886	-0.2179	0.5502
	NPGA	1.2000	0.4216	0.0071	0.0025	-1.0000	0.0000
	NSGA	1.4000	0.5164	0.0083	0.0031	-1.0000	0.0000
MOP2	MOGA	85.1000	12.3868	0.3390	0.0493	0.0135	0.0023
	MOMGA	98.5000	27.8219	0.3924	0.1108	0.0152	0.0055
	NPGA	79.7000	12.2751	0.3175	0.0489	0.0150	0.0045
	NSGA	14.2000	3.4897	0.0566	0.0139	0.0957	0.0178
MOP3	MOGA	292.1000	52.6444	0.0393	0.0071	0.1575	0.0506
	MOMGA	185.0000	52.5378	0.0249	0.0071	0.2089	0.0664
	NPGA	262.6000	47.1668	0.0353	0.0063	0.1353	0.0794
	NSGA	114.5000	20.2498	0.0154	0.0027	0.2926	0.0923
MOP4	MOGA	88.0000	10.0775	0.4783	0.0548	0.1257	0.0217
	MOMGA	67.5000	35.7313	0.3668	0.1942	0.2018	0.0715
	NPGA	42.2000	7.8853	0.2293	0.0429	0.2236	0.0380
	NSGA	17.8000	3.7653	0.0967	0.0205	0.4781	0.1204
MOP5	MOGA	108.1000	15.9614	0.1016	0.0150	0.0166	0.0028
	MOMGA	271.3000	109.5973	0.2550	0.1030	0.0099	0.0087
	NPGA	23.5000	2.9907	0.0221	0.0028	0.1169	0.0346
	NSGA	33.1000	4.8637	0.0311	0.0046	0.0629	0.0132

Table 7.5. MOEA Overall Results

MOP	**Pareto Front**	**ONVG**	**Spacing**	**Generational Distance**
MOP1	PF_{true}	168 (0.0010%)	5.93e-15	0
	PF_{known} (MOGA)	49	0.1188	3.4132e-12
	PF_{known} (MOMGA)	168	2.1745e-11	1.8349e-12
	PF_{known} (NPGA)	2	-1	0.2956
	PF_{known} (NSGA)	6	3.7569	0.6688
MOP2	PF_{true}	251 (0.0015%)	0.0102	0
	PF_{known} (MOGA)	145	0.0103	1.7928e-04
	PF_{known} (MOMGA)	144	0.0101	8.9255e-05
	PF_{known} (NPGA)	133	0.0099	2.0287e-04
	PF_{known} (NSGA)	45	0.0254	0.0030
MOP3	PF_{true}	7,439 (0.0443%)	0.0039	0
	PF_{known} (MOGA)	970	0.0211	1.4611e-04
	PF_{known} (MOMGA)	629	0.1289	0.0011
	PF_{known} (NPGA)	919	0.0262	1.8701e-04
	PF_{known} (NSGA)	453	0.0495	6.5920e-04
MOP4	PF_{true}	184 (0.0011%)	0.0912	0
	PF_{known} (MOGA)	212	0.0635	0.0604
	PF_{known} (MOMGA)	122	0.0910	6.0801e-04
	PF_{known} (NPGA)	96	0.1243	0.0043
	PF_{known} (NSGA)	36	0.2929	0.1523
MOP6	PF_{true}	1,064 (0.0063%)	0.0013	0
	PF_{known} (MOGA)	385	0.0054	3.7862e-05
	PF_{known} (MOMGA)	884	0.0017	3.4246e-06
	PF_{known} (NPGA)	57	0.0323	0.0074
	PF_{known} (NSGA)	83	0.0222	0.0029
MOP7	PF_{true}	40 (0.0002%)	0.1151	0
	PF_{known} (MOGA)	12	1.5786e04	2.2217e04
	PF_{known} (MOMGA)	40	0.1151	8.7334e-16
	PF_{known} (NSGA)	3	2.1961	1.0079

Table 7.6. Nonparametric Statistical Test Results

MOP	MOEA	Generational	ONVG	Spacing
MOP1	Kruskal-Wallis	0.0007	0.0001	
$\alpha = 0.1$	MOGA – MOMGA	0.0390	0.1284	
$\alpha^* = 0.0\overline{3333}$	MOGA – NPGA	0.0027	0.0001	
	MOGA – NSGA	0.0848	0.0002	
	MOMGA – NPGA	0.0009	0.0081	
	MOMGA – NSGA	0.0213	0.0210	
	NPGA – NSGA	0.1390	0.3415	
MOP2	Kruskal-Wallis	$< .0001$	$< .0001$	$< .0001$
$\alpha = 0.1$	MOGA – MOMGA	0.1287	0.1733	0.8797
$\alpha^* = 0.0\overline{3333}$	MOGA – NPGA	0.5379	0.4055	0.6769
	MOGA – NSGA	0.0002	0.0002	0.0002
	MOMGA – NPGA	0.1291	0.0537	0.5452
	MOMGA – NSGA	0.0002	0.0002	0.0002
	NPGA – NSGA	0.0002	0.0002	0.0002
MOP3	Kruskal-Wallis	0.0023	$< .0001$	0.0026
$\alpha = 0.1$	MOGA – MOMGA	0.0233	0.0019	0.0284
$\alpha^* = 0.0\overline{3333}$	MOGA – NPGA	0.1204	0.1035	0.8798
	MOGA – NSGA	0.0962	0.0002	0.0065
	MOMGA – NPGA	0.0015	0.0045	0.0342
	MOMGA – NSGA	0.7624	0.0101	0.0588
	NPGA – NSGA	0.0041	0.0002	0.0041
MOP4	Kruskal-Wallis	$< .0001$	$< .0001$	$< .0001$
$\alpha = 0.1$	MOGA – MOMGA	0.0025	0.1509	0.0082
$\alpha^* = 0.0\overline{3333}$	MOGA – NPGA	$< .0001$	0.0002	0.0002
	MOGA – NSGA	0.0002	0.0002	0.0002
	MOMGA – NPGA	0.4963	0.1304	0.3258
	MOMGA – NSGA	0.0004	0.0002	0.0003
	NPGA – NSGA	0.0002	0.0002	0.0002
MOP6	Kruskal-Wallis	$< .0001$	$< .0001$	$< .0001$
$\alpha = 0.1$	MOGA – MOMGA	0.0008	0.0019	0.0082
$\alpha^* = 0.0\overline{3333}$	MOGA – NPGA	0.0001	0.0002	0.0002
	MOGA – NSGA	0.0001	0.0002	0.0002
	MOMGA – NPGA	0.0001	0.0002	0.0002
	MOMGA – NSGA	0.0001	0.0002	0.0002
	NPGA – NSGA	0.0025	0.0011	0.0004

VIII. Conclusion

I can remember Bertrand Russell telling me of a horrible dream. He was in the top floor of the University Library, about A.D. 2100. A library assistant was going round the shelves carrying an enormous bucket, taking down book after book, glancing at them, restoring them to the shelves, or dumping them into the bucket. At last he came to three large volumes which Russell could recognize as the last surviving copy of *Principia Mathematica*. He took down one of the volumes, turned over a few pages, seemed puzzled for a moment by the curious symbolism, closed the volume, balanced it in his hand and hesitated....

G. H. Hardy, *A Mathematician's Apology*

8.1 Introduction

Of all research conducted supporting this effort this document presents the most important results. Taken as a whole, this research is largely original and should be of great interest to the MOEA research community. Portions are "state of the art" and expand MOEA knowledge frontiers. Section 8.2 highlights this research's major contributions, substantiating their originality and relating them to the overall research goals. Based on this research, Section 8.3 then identifies promising research areas within which further work may be beneficial.

8.2 Dissertation Contributions

Did this research meet its goals? This section presents evidence backing our claim it did. This research produced several original contributions satisfying research goals, including an MOEA classification, additional Pareto-based MOEA theory and terminology, an innovative MOEA, a proposed MOEA test function suite (with guidelines for adding further functions), and an experimental methodology allowing for quantitative comparisons based on the developed metrics. These contributions are now summarized (not necessarily in order of importance), their originality substantiated, and their specific relationship to the research goals and supporting objectives (as shown in Table 8.1) identified.

8.2.1 MOEA Technique Classification, Catalogue, and Analysis. A framework with which to classify MOEA implementations and applications was defined and used in

Table 8.1. Research Goals and Objectives

Goal: MOEA Classifications
Objectives:
Develop/refine a sound, extensible basis for MOEA classification
Classify known implementations
Organize key problem/algorithm domain components of classified MOEAs
Organize MOEA test functions used in the literature
Goal: MOEA Analyses
Objectives:
Critically consider current MOEA literature based upon classification effort
Analyze the MOP/MOEA domain integration process
– Identify and analyze major MOP domain characteristics
– Identify and analyze key MOEA components used in solving MOPs
Identify existing "well-engineered" MOEAs
Identify/analyze metrics for use in comparing MOP solutions
Goal: MOEA Innovations
Objectives:
Define the presence and role of BBs in MOP solutions
Engineer an MOEA to explicitly manipulate building blocks in solving MOPs
– Incorporate relevant analytical results in designing a BB-based MOEA
– Determine performance of the new MOEA
– Determine benefits of a parallel implementation
Substantiate and propose an MOEA test function suite
Substantiate and execute MOEA experiments
– Use developed metrics, test functions, and suitable MOEAs
Relate experimental results to MOEA application in real-world MOPs

cataloging key facts of known MOEA-based research efforts. New citations are easily placed in this framework; new techniques are easily accommodated by extending it. The catalogue is a "one-stop shopping" resource for the interested practitioner and allows several ways to cross-reference information of interest. Its structure and accompanying analysis allows EA practitioners to quickly become familiar with key MOEA issues and easily implement theoretically sound algorithms. As part of the cataloging effort this research attempted to capture all currently known MOEA citations in its database. Recently, a related effort was identified that is constructing an on-line MOEA bibliography and repository [68]. We contributed a large number of citations to this list helping bring the current total to over 300 separate references.

Based on our review an extensive and detailed analysis discussed several key MOEA topics, offering many original observations such as general MOEA complexity and design recommendations. This analysis substantiated and profiled heretofore unseen research trends and "disconnects" in the literature, rebutting some assertions and validating others. Finally, we collected, classified, and analyzed all known MOEA test functions found in the literature.

The MOEA classification framework is an extension of one previously appearing in the literature [152]. We independently justified its structure in light of past MOEA research and indicated several other multiobjective solution techniques may yet be implemented within an MOEA. Our tabular presentation is also unique. Although other (complementary) MOEA reviews exist, as do similar cataloging efforts focusing on single-objective EAs (e.g., EAs in management applications [236]), this is the only MOEA resource with this extensive detail. Other MOEA researchers recognize its value as they cite it in their own work [61, 83].

Although "bits and pieces" of this analysis are addressed elsewhere this is the only collocated, and most comprehensive, discussion currently available. These contributions satisfy all objectives supporting the MOEA classifications goal (see Table 8.1). Except for metric identification, they also fulfill all objectives supporting the MOEA analyses goal.

8.2.2 Pareto Theory, Terminology, and Notation. The literature review supporting this research indicates many MOEA researchers do not clearly understand the MOP domain, and highlights the fact that many use Pareto terminology inconsistently. Thus, we have developed several Pareto definitions, theorems, and a corollary specifically relating to MOEA implementation and use. These definitions and theorems are either not found in the MOEA literature (or citations within it) or were developed independently; together they more clearly define the MOP and MOEA domains. We also developed a consistent terminology and associated notation differentiating between Pareto instantiations within MOEAs. Although it had its genesis elsewhere [152] we have refined, specified, and implemented the notation for general MOEA use. This easily extendable notation is a tool with which researchers can consistently and unambiguously explain and refer to Pareto concepts instantiated in their algorithms. Taken collectively these results further define and refine the domain within which MOEAs operate.

8.2.3 MOMGA Implementation. The MOMGA, a new and innovative MOEA, was justified and implemented. This algorithm extends a previously proposed single-objective EA (the mGA) to the MOP domain, incorporating known MOEA theory and further insights gained through this research in an attempt to construct a more efficient and effective MOEA.

The MOMGA exemplifies a new paradigm, as it is the only known MOEA explicitly manipulating BBs in the search for MOP solutions. In fact, of all known MOEA implementations, only one makes any mention of even considering mGA concepts within MOEAs [142]. Because of its unique algorithmic structure and operators the MOMGA provides the means with which to identify and "follow" BBs as they are used in constructing MOP solutions. This new MOEA may be parallelized which should result in performance gains.

The MOMGA's successful operation and performance over this problem class implies other BB-based EAs may also be extended to the MOP domain. These alternative BB-based implementations may then offer further insight. The MOMGA, and its supporting analysis and justification, satisfy the first two objectives supporting the MOEA innovations goal (see Table 8.1).

8.2.4 MOEA Test Function Suite. An MOEA benchmark test function suite was substantiated and proposed. Although *de facto* MOEA benchmark test functions exist, no standardized and/or truly justified test functions were available; the lack of an MOEA test suite was a glaring omission. Thus, a range of MOP genotypical and phenotypical characteristics were identified and related to the MOEA domain. Consideration of appropriate issues then led to suitable test functions (with and without side-constraints, and drawn from the literature) suggested for use. We also addressed recent related work proposing an MOEA test suite construction methodology [83], indicating some possible shortfalls.

This document reflects one of only two collocated discussions of and proposals for an MOEA benchmark test function suite [326, 83]. Unlike some test functions used in the MOEA literature the proposed MOPs *do* test relevant MOEA domain characteristics. This now serves as a foundation for further definition of problem domain characteristics, and for constructing functions of desired complexity/characteristics. The test function suite lays the groundwork for "meaningful" MOEA comparisons, and satisfies an objective supporting the MOEA innovations goal (see Table 8.1).

8.2.5 MOEA Experimental Methodology and Metrics. A quantitative MOEA experimental methodology was substantiated and implemented. Several new metrics were also defined, which when used with other metrics drawn from the literature, allow for absolute and/or relative quantitative MOEA comparisons. A parallelized program allowing for deterministic enumeration of arbitrarily large numeric solution spaces was developed supporting this methodology, and may also be used to "solve" combinatorial optimization problems. Experimental results highlighted the strengths/weakness of selected MOEA implementations and identified issues for further study.

This quantitative MOEA experimental methodology stands as a new foundation for future MOEA comparisons. The overwhelming majority of cited MOEA "experiments" consists of testing MOEAs on unjustified numeric MOPs and/or specific applications, often visually comparing each MOEA's PF_{known}. Very few efforts report quantitative metrics for judging MOEA performance [292, 359, 358]. This methodology with its associated quantitative metrics allows for statistical *and* quantitative conclusions concerning tested MOEAs' efficiency and or effectiveness. We can now make absolute versus relative performance statements. Even if P_{true} or PF_{true} is unknown, some metrics can still indicate (lack of) desired performance; they may also be used in determining algorithmic convergence to PF_{true}. As a whole, this contribution satisfies the last objective supporting the MOEA analyses goal (see Table 8.1), and the remaining objectives supporting the MOEA innovations goal.

8.3 Future MOEA/MOP Research

This research accomplished its major goals. In addition to producing several original contributions, the MOMGA, MOEA test function suite, and experimental metrics are "state of the art." The contributions we have noted substantiate further exploration into selected topics.

A primary interest is the role of BBs in determining a *set* of solutions. Extending another appropriate BB-based single-objective EA into the MOP domain produces another tool with which to study their role and address the relationship between BBs and P_{true}, and allows more complex MOPs to be solved with these BB-based MOEAs.

The experimental methodology is an excellent framework within which to compare MOEA implementations and their performance. Further appropriate additions to the MOEA test function suite (e.g., MOPs with real-world-sized search spaces *and* analytical P_{true} solutions, side-constrained and NP-Complete MOPs) increase the suite's benefits to the MOEA community. Further experimentation using the test functions also allows for MOEA parameter sensitivity analysis. Additionally, some of the proposed metrics may be used to explore inter-generational behavior, i.e., they can be used to measure the convergence characteristics of various MOEAs.

The MOEA software used in this research is largely developmental. We used *MATLAB* to our advantage in the rapid prototyping and execution of experimental software and supporting research. Although reengineering all experimental MOEAs may increase execution time, translating the *MATLAB*-implemented MOEAs to C++ has in at least one case shown a minimum one order of magnitude improvement; parallelizing the C++ code provides even greater performance gains [143:p. 95].

Finally, other algorithmic approaches may be suitable for implementation within the MOP domain. For example, single-objective, *Immune* EAs may be implemented within Computer Virus Immune Systems [212]. An Immune EA's objective function is used to condition the EA's population (antibodies) to recognize a broad range of solutions (antigens), which may have widely differing characteristics. For example, Immune EAs are candidates for solving constrained single-objective optimization problems [139, 141].

As Pareto optimal solutions have been shown to (possibly) have differing characteristics, a multiobjective Immune EA approach may be viable and result in an alternative algorithmic solution technique for MOPs.

Appendix A. MOEA Classification and Technique Analysis

A.1 Introduction

This chapter contains the tables classifying and cataloging all known (to date) MOEA implementations. Relevant background information is first presented, followed by tables for each of the three major techniques, and finally the tables cataloging related MOEA publications.

These tables, background information, and associated discussion are originally published in a technical report [326]. Because the volume of information is so large, the background information and tables are placed in this appendix with the analysis in the body.

A.1.1 Mathematical Notation. Definition 2 (Section 2.2) defines a formal MOP model; we also employ the following associated notation to mathematically represent various MOEA techniques.

$$\Phi \quad : \quad \Omega \longrightarrow \Lambda \tag{A.1}$$

$$f(a_i) \quad \triangleq \quad F(f_1(a_i), \ldots, f_k(a_i)) \tag{A.2}$$

$$\Psi \quad : \quad P \longrightarrow P' \tag{A.3}$$

$$\mathbb{P} \quad = \quad \{a_i | \forall a_i, a_i \in P_{known}\} \tag{A.4}$$

Equation A.1 describes a particular technique's domain (Ω) and range (Λ). Equation A.2 is a solution's scalar fitness value derived via some defined equation. Equation A.3 is a generation transition function indicating that the particular technique incorporates specialized selection EVOPs, perhaps *not* relying on a directly computed overall solution fitness. Finally, Equation A.4 describes the set of solutions returned to a DM such that every solution in the set is a member of P_{known}.

A.1.2 Presentation Layout. A brief explanation of each major technique in Figure 2.13 (Section 2.6.1) is presented, and includes a mathematical description such as Fonseca and Fleming present [107]. Also, a table cataloging relevant MOEA research efforts incorporating that technique is shown. Each table, for each effort, lists five key algorithm and problem domain components which are:

Approach. Name or type of MOEA used, citation, and year results published

Description. Approach specific information of interest, e.g., operators, methodology, etc.

Application. Problem domain (if any) in which the MOEA is applied

Objectives. Number of objectives and their description

Chromosome. Representation used and gene correspondence (if noteworthy)

These components capture essential information about each approach and are not meant as a complete description. Because of the manner in which the research efforts were classified (according to MOEA fitness assignment and/or selection), the "Approach" and "Description" categories contain information such as the specific MOEA type, parallelized implementation, specialized EVOPs, etc. Finally, each table's entries are chronologically ordered by year published.

The next few sections present key points of known MOEA approaches. Some assignments of an approach to a particular category are necessarily subjective, as several approaches incorporate or report results from several MODM techniques. Thus, some approaches are classified more than once; their classifications correspond to the categories identified in Figure 2.13.

A.2 A Priori *MOEA Techniques*

A priori MOEA techniques expect DM input before the EA search process begins, and result in an optimal *solution* presented to the DM. Ordering, linear, and nonlinear combination techniques are discussed in this section.

A.2.1 Lexicographic Techniques. Lexicographic selection (ordering) is based on each objective's DM-assigned priority *prior* to optimization. The highest priority objective is used first when comparing solutions; if a tie results the next highest-priority objective is compared, etc. All objectives $f_1, \ldots, f_k$ are assumed sorted in order of increasing priority. This is termed *lexicographic* ordering [27] and is mathematically represented by:

$$\begin{aligned} \Phi \quad &: \quad \mathbb{R}^n \longrightarrow \{0, 1, \ldots, \mu - 1\} \\ f(a_i) \quad &\triangleq \quad \sum_{j=1}^{\mu} \chi \left(f(a_j) \; l\!< \; f(a_i) \right) , \end{aligned} \tag{A.5}$$

where $f(a_j) \; l\!< \; f(a_i)$ if and only if

$$\exists p \in \{1, \ldots, k\} : \forall q \in \{p, \ldots, k\} \quad , f_q(a_j) \leq f_q(a_i) \wedge f_p(a_j) < f_p(a_i) \; ,$$

and where

$$\chi = \begin{cases} 1 & \text{if } (f(a_j) \; l\!< \; f(a_i)) \\ 0 & \text{otherwise} \end{cases}$$

This technique is best used with rank-based selection. Table A.1 lists the known lexicographic MOEA approaches.

Table A.1 Lexicographic Techniques

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [117] (1985)	Heuristically prioritizes objectives	Silicon layout compaction	(3) Bounding box size; Design rule violations; Rectangle placement	Variable length; Genes are lists of layout constraints
Global Evolutionary Planning and Obstacle Avoidance system (GEPOA) [92] (1998)	Fuzzy tournament selection algorithm implements fuzzy lexicographic preferences	Motion planning and obstacle avoidance	(3) Euclidean distance; Sum of path slope changes; Average slope change	Real Values; Genes represent x-y coordinates

A.2.2 Linear Fitness Combination Techniques. Linear fitness combination is a scalar aggregation of several distinct fitnesses; a DM assigns a strictly positive scalar weight to each objective reflecting its relative importance to the final solution. The weighting vector, $\lambda = (w_1, \ldots, w_k) \in R^k$, is often normalized so that its elements sum to unity [308].

This technique is mathematically represented by:

$$
\begin{aligned}
\Phi \quad &: \quad \mathbb{R}^n \longrightarrow \mathbb{R} \\
f(a_i) \quad &\triangleq \quad \sum_{j=1}^{k} w_j f_j(a_i) \ ,
\end{aligned}
\tag{A.6}
$$

where w_j is the weight assigned to objective f_j. This technique can be used in fitness proportional, tournament, or rank-based selection. Table A.2 lists the known linear fitness combination MOEA approaches.

Table A.2 Linear Fitness Combination

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [310] (1991)	Hybrid GA implementation; Incorporates a schedule builder and evaluator	Laboratory resource scheduling	Not stated	Permutation task ordering
GA-based learning system [167] (1992)	Structured populations; Parameterized mating only within overlapping demes; Parallelized	Machine learning (route planning and vehicle control)	(5) Distance; Required time; Path deviation; Collision monitor activations; Emergency monitor activations	"Action Chain;" Genes are lists of actions for robotic task
GA [171] (1993)	Linear normalized fitness and weighted penalties	3-D structure conformational search	(2) Match penalty; Energy penalty	Binary string; Genes are rotation angles
GP [278] (1994)	***Pygmies and Civil Servants***; "Normal" fitness computation for servants; Pygmy fitness based on chromosome length; Recombination uses one parent from each	Network sorting	(2) Efficiency; Length	Program
GA [32] (1994)	Specialized crossover; GA population selected from training database; One, some, or all GA population members replace least fit database members	Adaptive image segmentation	(5) Edge-border coincidence, Boundary consistency, Pixel classification, Object overlap, Object contrast	Binary string; Genes are fitness, image conditions, and parameters; EVOPs operate ***only*** on parameters
GA_FTP1 [44] (1994)	Specialized EVOPS; Objectives are fuzzified	Multiobjective solid transportation problem	(2)	3-D integer array
GA [199] (1994)	Uses specialized EVOPs	Job shop scheduling	(2) Mean flow time; Mean lateness	Integer string; Genes are sequenced jobs
GP [159] (1994)	Incorporates minimum description length principle	Pattern recognition & Time series prediction	(2) Tree coding length; Exception coding length	Decision tree

Table A.2 continued

Approach	Description	Application	Objectives (#)	Chromosome
Multi-Niche Crowding (MNC) GA [330, 47] (1995,1997)	Fitness obtained by summing individual rank *in each objective*; Phenotypic-based crowding; Integrated with flow-transport simulation code	Groundwater pollution containmant monitoring; Also tested on multimodal, dynamic function	(3) Cost; Contanimant removal; Containimant leakage	Variable length integer string; Genes are geographic nodes
GA [279] (1995)	Each solution's fitness based on how "well" it fits its race's ideal	None	(2) Numeric optimization (one objective is always "race" ideal)	Implies binary string
GA [16] (1995)	Repair procedure encodes valid chromosomes; Presents unique bit string representation of flow-network paths	Computer Aided Process Planning	(2) Cost; Quality	Binary string; Chromosome is an encoded flow network
GA [225] (1995)	Standard GA	Pot core transformer design	(2) Device area; Magnetic flux density	Binary string
GA [43] (1995)	Crowding-based selection; GA deceptive problem	Food distribution center management	(2) Quality loss; Storage utilization	Binary string; Genes are cluster capacity and time utilized
GA [122][1] (1995)	Unknown	Unknown	Unknown	Unknown
GA [11] (1995)	Weights selected to explicitly focus search	Wing design	(4) Lift/drag; Lift/weight; Area; Lift	Binary string
Parallel GA [196] (1996)	Decomposition splits problem into (independent) sub-problems which are solved in parallel	Rotor blade design	(3) Unknown	Binary string; 179 bits; 42 design variables
Multiobjective Fuzzy GA [321] (1996)	Linear weighted sum of fuzzified objectives and constraints	Depth control system	(2) Original objective; Constraints	Binary string; Gray coded
GP [355] (1996)	Sum of two objective's weighted values; Function is adaptive	Water pollution prediction & Laser prediction	(2) Error; Complexity	Neural trees
GP [304] (1996)	Progressive fitness measure	Tetris	(2) Computation time; Game time	Terminal set
Markowitz Model GA [303] (1996)	Representation guarantees feasible solutions	Portfolio selection	(2) Portfolio variance; Rate of return	Binary string; Genes are stocks
GA [136] (1996)	Apparently sums two objectives; Incorporates penalty function	Truck packing	(2) Volume; Center of gravity	Binary string; Genes are box spacings
GA [197, 198] (1996)	Specialized EVOPs; Directs search from "negative" to "positive" ideal point; Elitist selection; Fuzzy numbers and ranking	Multicriterion solid transportation problem	(3)	3-D integer array
Multiobjective GA [30] (1996)	Steady-state GA; Indirect representation and mapping allows smaller chromosomes	Table design & Prism design	(5) Size; Mass; Flat surface; Stability; Supportiveness & Unknown	Unknown

[1] Cited by Li [198]; in Japanese.

Table A.2 continued

Approach	Description	Application	Objectives (#)	Chromosome
ES [7] (1996)	Hybridized; Compares results against other stochastic and deterministic optimization algorithms	Superconducting magnetic energy storage system	(2) Energy; Magnetic flux density RMS error	Implies real values
GA [168] (1996)	Specialized EVOPs; Only feasible individuals created	Interval multi-objective solid transportation problem	(3)	3-D real-valued array
Multiobjective GA [14] (1996)	Chromosomal representation compatible with common CAD tools	Circuit design	(3) Function; Signal delay; Circuit area	Unknown; Genes are library cells and attributes
GA [305] (1996)	Two sub-populations initialized via NEH and RC heuristics; Uses crowding	Flowshop/Cellular manufacturing system scheduling	(3) Makespan; Flowtime; Idletime	Integer string; Genes are job sequences
GA [161] (1997)	Steady-state GA; Results appear to use only two criteria	Selective laser sintering build cylinder packing	(3) Part overlap; Packing "tightness"; Part containment in cylinder	List of lists; Permutation integer ordering in one dimension; integers in others
Multi-Sexual GA [201] (1997)	Individuals are "sex" coded (one for each function); Recombination uses one parent from each sex; Individuals evaluated by their sex's function	None	(2) Numeric optimization	Binary string; "Sex" marker at end
GA [300, 299] (1997)	Compares results of various scheduling heuristics	Production scheduling	(3) Rejected jobs; Order lateness; Run variation	Integer string; Indirect representation
GA [260] (1997)	Uses weighted sum of goal deviations; Compares results against those derived by goal programming	Upgrading/new road link projects	(17) Unknown	Integer string; Permutation ordering
GA [283] (1998)	Integrated with finite element method approximation code	Superconducting magnetic energy storage	(2) Energy; Magnetic induction	Real values
(1 + 1) ES [41] (1998)	Incorporates specialized problem domain code	Permanent magnet synchronous motor	(3) Harmonic content; Fundamental component; Cogging torque	Real values
(1 + 1) ES [179] (1998)	Uses "shaking" to escape local minima; Incorporates FEM code	Electric vehicle induction motor	(2) Engine efficiency; Weight	Real values
GA [328] (1998)	Integrated two GAs with electromagnetic evaluation code; Fitness mapping (scaling)	Wire antenna geometry design	(4) Antenna gain; Radiation symmetry; Resistance; Reactance	Real values; Gene triplets represent wire endpoints in 3-D space
GA [227] (1998)	Weights are functions of objective functions' max and min values yet found	Computer aided process planning	(2) Processing and transportation time; Workstation load variation	Integer string; Genes are plans producing certain parts

Table A.2 continued

Approach	Description	Application	Objectives (#)	Chromosome
GA [352] (1998)	Steady-state GA; Specialized EVOPs and population re-initialization	Telephone operator scheduling	(2) Operator shortage; Operator surplus	Integer string; Genes are partial schedules composed of shift time, and number and time of rest breaks
GA [57] (1998)	Specialized crossover; $\frac{3}{4}$ population: tournament selection, $\frac{1}{4}$ roulette wheel and fitness scaling	Non-chromatic rectangle boards	(4) Distribution of colors; # Red, white, and blue chromatic rectangles	2-D array of integer values; Genes are colored squares
GA [229] (1998)	Uses "Pitt" approach in evolving classification rules; Fitness depends on both weighted sum of conjunctive attribute test and simulated trading results; Chromosomally encoded weights sometimes operated on by GA	Portfolio stock selection	(9) Conjunctive attribute rule tests	Integer (0,1), character (#), and real values; Genes are attribute tests and associated weights
GA [243] (1998)	Compared different encodings and population policies	Automotive water pump design	(3) Exit pressure; Exit flow; Input power	Real values and grey coded binary string
GA [203] (1998)	Four weight parameter combinations examined; "Fine-tuning" after GA convergence	45-bar truss design	(3) LQR cost; Robustness; Controllability	Binary string; Genes are structural and actuator variables (both continuous and discrete)

A.2.3 Nonlinear Fitness Combination Techniques. Nonlinear fitness combination is also a scalar aggregation of distinct fitnesses; several EA-based variants have been implemented. This aggregation incorporates nonlinear terms which are normally derived in some "trial and error" fashion.

For example, penalty functions *penalize* solutions when a constraint is not met. Two variants are common in EA research: general penalty functions like that defined by Goldberg [126], and transforming constraints "into" objectives. According to Cohon [69] the latter method changes the MOP into a single-objective optimization problem. One objective is arbitrarily selected for optimization and the other $k-1$ objectives are constrained to a maximum value represented by ϵ_i, where $i = 2, \ldots, k$.

Penalty functions have been implemented as part of cooperative population searches and are thus classified in Sections A.4.3 and A.4.4. Other implemented nonlinear combination techniques are now addressed in turn.

A.2.3.1 Multiplicative Fitness Combination Techniques. Multiplicative fitness combination is a scalar aggregation of distinct fitnesses where individual objective values are combined through multiplication. This technique's general form is mathematically represented by:

$$\begin{aligned} \Phi &: \mathbb{R}^n \longrightarrow \mathbb{R} \\ f(a_i) &\triangleq \prod_{j=1}^{k} f_j(a_i) \; . \end{aligned} \tag{A.7}$$

This technique can be used in fitness proportional, tournament, or rank-based selection. Table A.3 lists the known multiplicative fitness combination MOEA approaches.

Table A.3 Multiplicative Techniques

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
Multi Attribute Utility Analysis (MAUA)-GA [154, 153] (1993)	*Proposes* MAUA to determine fitness function	None	None	None
GA [338] (1996)	Probability of acceptance is fitness; Overall fitness is logarithm of all multiplied probabilities; Penalty function used	Two-member truss Design	(6) Stress safety factor and diameter for each bar (4); Buckling safety factor; Cost	Binary string

A.2.3.2 Target Vector Fitness Combination Techniques. The "target vector" technique is a scalar aggregative method which can be thought of as using the "distance to the target" as the fitness metric. A DM assigns performance goals to each objective, whereupon solutions are evaluated by measuring the distance (over some norm) from their respective goals in criteria space. This technique is mathematically represented by:

$$\begin{aligned} \Phi &: \mathbb{R}^n \longrightarrow \mathbb{R} \\ f(a_i) &\triangleq \| \, [f(a_i) - g]\mathbf{W}^{-1} \, \|_\alpha \; , \end{aligned} \tag{A.8}$$

where $g = (g_1, \ldots, g_k)$ is a vector representing the desired goals, $\mathbf{W}$ is a weighting matrix accounting for differing variance between the k goals, and α is most often the Euclidean distance ($\alpha = 2$) [343]. This technique can be used in fitness proportional, tournament, or rank-based selection. Table A.4 lists the known target vector fitness combination MOEA approaches.

Table A.4 Target Vector Techniques

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [343] (1992)	Attempts to achieve desired criterion goals (goal programming)	Atomic emission spectroscopy	(7) Atomic emission intensities of seven atomic elements	Binary string; Represents NaCl concentration and current intensity
GA [285] (1994)	Attempts to achieve desired criterion goals (nonlinear goal programming)	3-Bar truss & Linkage design & 10-Bar truss	(2) Weight; Stress & (9) Distances (5); Size and weight; Velocity change; Path sensitivity (2) & (7) Weight; Stress (3); Displacement (2) Number of beams	Unknown
Fuzzy Logic-Based Multiobjective GA [269] (1997)	Fuzzy logic-based fitness; Uses NPGA [155] fitness sharing	Beryllium powder micromechanical model	(17) Calculated/reference densification data point deltas	Binary string (266 bits); Genes are model parameters
Fuzzy Logic-Based Multiobjective GA [270] (1997)	Fuzzy logic-based fitness; Uses NPGA [155] fitness sharing	Copper powder micromechanical model	(6) Calculated/reference densification data point deltas	Binary string (224 bits); Genes are model parameters

A.2.3.3 Minimax Fitness Combination Techniques. Minimax is a scalar aggregative method minimizing the maximum (weighted) difference between the objectives and DM-specified goals. This technique is mathematically represented by:

$$\begin{aligned} \Phi &: \mathbb{R}^n \longrightarrow \mathbb{R} \\ f(a_i) &\triangleq \max_{j=1,\ldots,k} \frac{f_j(a_i) - g_j}{w_j} , \end{aligned} \qquad \text{(A.9)}$$

where g_j is the performance goal to be reached or bettered for objective f_j, and w_j is a weight indicating the desired search direction in objective space, where w_j is often set to $\| g_j \|$ [107]. This technique can be used in fitness proportional, tournament, or rank-based selection. Table A.5 lists the known minimax fitness combination MOEA approaches.

Table A.5 Minimax Techniques

Approach	Description	Application	Objectives (#)	Chromosome
GA [265] (1993)	*Proposes* supercriterion method of combining objectives (game-theoretic approach)	None	None	None
GA [280] (1994)	Maximizes minimum linear membership function value; Specialized EVOPs	Fuzzy 2-objective knapsack: 1-D & 4-D	(2) & (2)	Binary string & Diploid binary string
GA [58] (1995)	Tchebycheff weighting, Uniformly varies key parameter	Groundwater containmant monitoring	(2) Undetected plumes; Contaminated area	Fixed-length integer string
GA [65, 59] (1995)	Objectives optimized in turn, then used to optimize weighted min-max formulation	Robot arm balancing	(4) Joint torque (2); Reaction force (2)	Real values
GA [323] (1995)	Original problem fuzzified; Max-min formulation	Fuzzy multiobjective double sampling	(3) Cost; Quality; Covariance	Binary string; Genes are sample sizes and acceptance numbers
Fuzzy Interval GA [49] (1997)	Incorporates decision maker's (fuzzy) goals into search	Nonlinear mixed integer programming	(2) Numeric optimization	Unknown

A.3 Progressive MOEA Techniques

The progressive techniques presented in this section involve direct interaction with the DM during the EA search process. Either cycles of decision making and search, or of search and decision making, are performed in pursuit of acceptable solutions. Both *a priori* and *a posteriori* techniques may be used in the search portion of this interactive decision making process; thus, no specific mathematical representation is given. However, as explained in Section A.4, *a posteriori* techniques provide a *set* of solutions instead of a single one. This situation is often preferable for MOPs. Table A.6 lists the known progressive MOEA approaches; papers are cited here *only* if the authors explicitly mentioned DM incorporation in the MOP solution process.

Table A.6 Interactive Techniques

Approach	Description	Application	Objectives (#)	Chromosome
Multiple Objective Genetic Algorithm (MOGA) [108, 110, 114, 115] (1993, 1995, 1998)	Fonseca's [111] ranking; Incorporates niching and goals (preferences)	Step response of gas turbine engine	(4) "Reach" time; "Settle" time; Overshoot; Error	Binary string; Genes are controller parameters

Table A.6 (continued)

Approach	Description	Application	Objectives (#)	Chromosome
GA [316] (1995)	Initial population contains ***only*** solutions in P_{known}; DM selects preferred returned solutions, used as basis for further exploration	None	(2) Numeric optimization	Binary string
Multiple Objective GA [297] (1996)	Uses Fonseca's MOGA [108]; Compares to weighted-sum approach	Meal production line scheduling	(3) Rejected orders; Batch lateness; Shift/staff balancing	Permutation ordering
Explorer [99] (1996)	GUI- and Pareto rank-based; Interactive; Incorporates user preferences and local search (hill climbing)	Floor planning	(4) Area; Aspect ratio; Routing congestion; Path delay	5 components: Integer string; Real values; Alphabet string; Binary string; Integer string
Evolutionary Co-Design (EvoC) [156, 133] (1996,1997)	Preference info classifies solutions; Pareto ranking on preferences	Hardware and software co-design	(3) Component cost; Critical excess MIPs; Feasibility factor	Binary string; Genes are implementation type and processor
MOGA [112] (1997)	Specialized EVOPs	Non-linear system identification (polynomial model)	(7) Residual variance; Long-term prediction error; Number of terms; Model lag; Model degree; Auto-correlation; Cross-correlation	Variable length integer string
GA [173, 302, 282, 281, 175, 176, 174] (1997)	Defines "fuzzy" Pareto optimality; DM (re)selects fuzzy membership levels; Decomposition procedure and specialized coding ensure feasible solutions; Minimax approach	Multiobjective 0-1 Programming	(3)	Binary string
GP [273] (1997)	Incorporates problem domain code; Extends MOGA [114] to the GP domain	Nonlinear system polynomial models	(7) Number of terms; Model degree; Model lag; Residual Variance; Long-term prediction error; Auto-correlation; Cross-correlation	Unknown
GA [226] (1998)	Uses domain-specific representation and operators, heuristic initialization, dynamic rescheduling, and shared (human) control; Linearly combines the objectives	Field service scheduling & Military land move scheduling	(7) [All costs] Missed target, Travel, Slack, Return home, Parts order, Unscheduled, Skills mismatch & (2) [Both costs] Staging; Link overuse	"Ordered pair;" Genes are indices into resource and tasking lists & String based; Genes are resource mappings

A.4 A Posteriori *MOEA Techniques*

A Posteriori MOEA techniques perform an MOP search process resulting in a ***set*** of identified solutions for DM selection. These techniques include both independent sampling and cooperative population search. We agree with Horn [152] that these approaches, whether implicitly or explicitly, are seeking the Pareto optimal solution set denoted by P_{true}. By definition, this set contains ***all*** possible optimal solutions – assuming a rational DM. Once a "satisfactory" P_{known} is discovered for a particular problem instance, a new DM (e.g., a new production supervisor) does not require a repeated search. Also, if P_{known} is "small enough" the additional overhead incurred by DM interaction is perhaps unnecessary.

A.4.1 Independent Sampling Techniques. Independent sampling is a technique using ***multiple*** single-criterion searches; each individual search optimizes different objective aggregations. Over time, P_{known} and PF_{known} emerge and are presented to the DM, as P_{true} and PF_{true} are often unknown for problems of any complexity. These techniques are mathematically represented by:

$$\begin{aligned} \Phi_i &: \mathbb{R}^n \longrightarrow \mathbb{R} \\ \mathbb{P} &= \{a_i | \forall a_i, a_i \in P_{known}\} , \end{aligned} \tag{A.10}$$

where Φ_i is some fitness function assigning solution fitness for MOEA "run" i (e.g., the multiple functions' associated weights change between runs), and $\mathbb{P}$ (i.e., P_{known}) is returned to the DM where each a_i is the optimum found in run i. Table A.7 lists the known independent sampling MOEA approaches.

Table A.7 Independent Sampling Techniques

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [117] (1985)	Composite strategies sample the trade-off surface	Silicon layout compaction	(3) Bounding box size; Design rule violations; Rectangle placement	Variable length; Genes are lists of layout constraints
GA [220] (1993)	Multiple GA runs use different function weights; Crowding replacement	Radar absorbent material coating design	(2) Coating reflection; Coating thickness	Binary string; Genes are material type and thickness
Multiple Objective GA [272] (1994)	*Proposes* multiple GA runs optimizing one criterion at a time, then varying the constraints	None	None	None

Table A.7 (continued)

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [58] (1995)	Tchebycheff weighting, uniformly varies key parameter	Groundwater monitoring	(2) Undetected plumes; Contaminated area	Fixed-length integer string
GA [48] (1995)	Multiple runs uniformly varies weights; Fitness scaling	Firing angles in railway traction substations	(2) Power supply; Uniform load sharing	Binary string
GAA and GAA2 [322] (1995)	Hybrid GA/SA; Linearly normalized weighted functions uniformly varied over several runs	Economic-Environmental Power Dispatch	(2) Cost; Weighted-sum of pollutants' emissions	Real-values; Genes are generator loadings

A.4.2 Criterion Selection Techniques. Criterion selection techniques are the first discussed to ***directly*** utilize an MOEA's population. Here, fractions of succeeding populations are selected using various of the k objectives. These techniques are able to find multiple members of P_{known} within a single EA run, and are mathematically represented by:

$$\begin{aligned} \Psi \quad &: \quad P \longrightarrow P' \\ \mathbb{P} \quad &= \quad \{a_i | \forall a_i, a_i \in P_{known}\} \ , \end{aligned} \tag{A.11}$$

where Ψ is some generation transition function selecting solutions based on their performance in some objective, and $\mathbb{P}$ (i.e., P_{known}) is returned to the DM. Table A.8 lists the known criterion selection MOEA approaches.

Table A.8 Criterion Selection Techniques

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
Vector Evaluated GA (VEGA) [288, 289] (1984)	$\frac{1}{k}$ of new population selected using each of the k objectives	None	(2) Numeric optimization	Binary string; Contains genes and objective performance information
Vector Evaluated GA (VEGA) [290] (1985)	$\frac{1}{k}$ of new population selected using each of the k objectives	Multiclass pattern discrimination	(2) (3) (4) (5) Number of classes	Unknown; Genes are rules
GA [117] (1985)	One criteria randomly selected as comparator	Silicon layout compaction	(3) Bounding box size; Design rule violations; Rectangle placement	Variable length; Genes are lists of layout constraints
ES [189] (1990)	Objectives' associated probabilities used as selection criteria; Polyploid individuals	None	(2) Numeric optimization	Both decision and stepsize variables have dominant and recessive chromosomes

Table A.8 (continued)

Approach	Description	Application	Objectives (#)	Chromosome
ES (μ+λ) [5] (1992)	Assigns "gender" to each function; Each sex judged only on its respective function; *No results presented*	Pipeline construction	(2) Cost; Biodiversity destruction	Binary string
GA [251, 34] (1994)	VEGA isolates feasible values of constrained parameters; Secondary GA searches hypercube based on returned values	Gas turbine engine cooling hole geometry	(3) Metal temperature; Cooling hole area; Coolant flow rate	Unknown
GA [32] (1994)	Specialized crossover; GA population selected from training database; One, some, or all GA population members replace least fit database members; VEGA selection; Implies only nondominated GA population solutions retained	Adaptive image segmentation	(2) Global (weighted sum of edge-border coincidence and boundary consistency); Local (weighted sum of pixel classification, object overlap and object contrast)	Binary string; Genes are fitness, image conditions, and parameters; *Only* parameters affected by EVOPs
Multiobjective GA [331] (1997)	Both nondominated- and roulette wheel (one objective)- based selection	Transonic airfoil design	(2) Mach number; Lift coefficient	Binary string; Genes are airfoil parameters
Nash GA [254, 253] (1997)	Incorporates Nash equilibrium concept; Population split into k; Each subpopulation optimizes solutions with respect to a different objective (subject to different solution constraints)	Electromagnetic backscattering	(2) RCS 1; RCS 2	Binary string; Genes are antenna locations
Multiobjective GA [237] (1998)	"Two-branch" tournament selection; Individuals compete in *only* one of 2 tournaments; Linear penalty functions	Non-collocated control	(2) Control error of Disk 1 rotational position; Same for Disk 2	Binary string; Genes are controller gains
Multiobjective GA [98] (1998)	"Two-branch" tournament selection; Individuals compete once in each of 2 tournaments; External penalty functions	Satellite constellation design	(2) Constellation altitude; Number of satellites	Binary string
Multiobjective GA [76] (1998)	"Two-branch" tournament selection; Individuals compete once in each of 2 tournaments; Scaled penalty functions	Two 10-bar truss designs	(2) Weight; Vertical displacement	Binary string
Parallel GA [170] (1998)	"k-branch" tournament selection; Parallel implementation; Integrated with XFOIL and WOPWOP codes; Penalty functions enforce constraints	Airfoil optimization	(2) Drag coefficient; Overall Averaged Sound Pressure Level	Binary string; Gray coded

Table A.8 (continued)

Approach	Description	Application	Objectives (#)	Chromosome
ES [195] (1998)	"Predator-prey" model; Predators "attack" based on one of k objectives	None	(2) Numeric optimization	Real values
GA [62] (1999)	Constraints cast as objectives; Uses VEGA [289] selection	Hydrostatic thrust bearing & Belleville spring & Combinatorial circuit & Himmelblau's problem & 10-bar truss	(8) & (8) & (9) & (4) & (23)	Real values

A.4.3 Aggregation Selection Techniques. Aggregation selection techniques directly utilize an MOEA's population capability. Here, succeeding populations are selected using solution fitness computed by either linear or nonlinear fitness combination techniques (which are not necessarily identical for each evaluated solution, i.e., multiple Φs exist). Thus, multiple members of P_{known} may be found within a single MOEA run. These techniques are mathematically represented by:

$$\begin{aligned} \Psi &: P \longrightarrow P' \\ \mathbb{P} &= \{a_i \mid \forall a_i, a_i \in P_{known}\} , \end{aligned} \tag{A.12}$$

where Ψ is some generation transition function selecting solutions based on their performance using an associated Φ , and $\mathbb{P}$ (i.e., P_{known}) is returned to the DM. Table A.9 lists the known aggregation selection MOEA approaches.

Table A.9 Aggregation Selection Techniques

Approach	Description	Application	Objectives (#)	Chromosome
GA [140] (1992)	Weighted sum; Weights are chromosomally encoded; Compares fitness sharing (applied only to weighting variables) and two VEGA [289] variants	Static and dynamically loaded 10-bar truss & Wing Box	(2) Structural weight; Vertical displacement & (2) Structural weight; Natural frequencies	Genes are design variables and weights; Mix of continuous and discrete alleles
Multi-Objective GAs [230, 231] (1995)	Randomly assigned weights; Pareto elitist selection	None	(2) Numeric & Scheduling & Rule selection examples	Binary string & Permutation ordering & Tri-Valued string
ES [335] (1996)	Fuzzy controller selects each solution's evaluation function	Railway network scheduling	(2) Cost; Waiting time	Unknown

Table A.9 (continued)

Approach	Description	Application	Objectives (#)	Chromosome
Multi-Objective Genetic Local Search Algorithm [163, 164] (1996, 1998)	Randomly assigned weights; Elitist selection; Local search in direction of current weights	Flowshop scheduling	(3) Makespan; Maximum tardiness; Total flowtime	Integer permutation ordering; Genes are jobs
Non-Generational GA (NGGA) [324] (1997)	Non-generational selection; Fitness calculated incrementally; k objectives transformed to 2; Weighted sum of objectives	None	(2) Numeric optimization (Effectively minimizes domination and niche count)	Binary string
Neighborhood Constraint Method [205, 204] (1997, 1998)	Indexed solutions; $N-1$ objectives converted into constraints, the other optimized; Constraint values varied among solutions; Restricted mating based on "neighborhood"	Air quality management	(2) Cost; Constraint satisfaction	Real values
GA [45] (1997)	Constraints converted into functions; Both efficient and dominated solutions determine search direction	None	(2) Numeric optimization (Original function; Constraints)	Real values
ES $(\mu + \lambda)$ [356] (1997)	Fitness determined by objective values and adaptive objective value hyperplane	Multicriteria production process planning	(2) Processing cost; Processing time	Permutation integer ordering; Genes are selected nodes for some operation
GA [75] (1997)	Kreisselmeier-Steinhauser function gives fitness; Multiple objectives and constraints combined into one unconstrained function	3-bar truss & Rotor system design	(2) Cost; Weight & (2) Power; Rotor system weight	Binary string; Genes are discrete, integer, and continuous variables
Multi-Objective Genetic Local Search Algorithm [165] (1998)	Randomly assigned weights; Elitist selection; Local search in direction of current weights	Fuzzy rule-based system rule selection	(2) Number of if-then rules; Number of correctly classified patterns	Binary string; Genes are rules
GA [123] (1998)	Specialized encoding and selection EVOP; Incorporates adaptive objective evaluation hyperplane [356] and auxiliary bi-objective problem	Topological Network Design	(2) Connection cost; Message delay	Prüfer number encoding; Integer string uniquely encodes a spanning-tree
GA [354, 353] (1998)	Adaptive objective weights; Values set with respect to objective value and user goals	Operational amplifier design	(7) Gain; GBW; Linearity; Power consumption; Area; Phase margin; Slew-rate	Integer string; Genes are transistor sizes, current, and capacitors
NSGA [85] (1998)	Uses NSGA and weighted goal programming; Adaptive objective weights	Welded beam design	(2) Cost; End deflection	Implies real values

A.4.4 Pareto Sampling Techniques. Pareto sampling directly utilizes an MOEA's population capability. Some approaches also incorporate a secondary population storing all Pareto optimal solutions yet found during MOEA execution. When using these methods, the generational population (possibly) holds several solutions of P_{known} and at least one member in $P_{current}(t)$. The secondary population is periodically updated to remove solutions which are no longer nondominated. These techniques explicitly use Pareto concepts in the selection process such that Pareto solutions are given preference over dominated solutions, but are treated equivalently among themselves.

Two types of Pareto fitness assignment are widely used. The first method, proposed by Goldberg [126], is mathematically (recursively) represented by:

$$\begin{aligned} \Phi \quad &: \quad \mathbb{R}^n \longrightarrow \{1, \ldots, \mu\} \\ f(a_i) \quad &\triangleq \quad \begin{cases} 1 & \neg(f(a_j)\; p\!<\; f(a_i))\; \forall j \in \{1, \ldots, \mu\} \\ \phi & \neg(f(a_j)\; p\!<\; f(a_i))\; \forall j \in \{1, \ldots, \mu\} | \{l : \Phi(f(a_l)) < \phi\} \,, \end{cases} \end{aligned} \tag{A.13}$$

where $f(a_j)\; p\!<\; f(a_i)$ if and only if

$$\forall k \in \{1, \ldots, n\} \;\; f_k(a_j) \leq f_k(a_i) \wedge \exists k \in \{1, \ldots, n\} : f_k(a_j) < f_k(a_i) \,,$$

and where the symbol $\neg$ denotes logical negation. In words, this method identifies all nondominated solutions within some set and assigns them rank 1. These are then removed and all nondominated solutions of the reduced solution set identified. These are then assigned rank 2. The process continues until the entire population is ranked.

The second technique, proposed by Fonseca and Fleming [111] is mathematically represented by:

$$\begin{aligned} \Phi \quad &: \quad \mathbb{R}^n \longrightarrow \{0, 1, \ldots, \mu - 1\} \\ f(a_i) \quad &\triangleq \quad \sum_{j=1}^{\mu} \chi(f(a_j)\; p\!<\; f(a_i)) \,, \end{aligned} \tag{A.14}$$

where χ (condition) $= 1$ if the condition is true, else 0. In words, this method assigns each solution a rank equal to the number of solutions it's dominated by. Thus, all nondominated solutions are assigned rank 0.

Several Pareto-based selection approaches have been implemented, selecting solutions based (at least in part) upon their domination status. These techniques are mathematically represented by:

$$\begin{aligned} \Psi &: P \longrightarrow P' \\ \mathbb{P} &= \{a_i | \forall a_i, a_i \in P_{known}\} , \end{aligned} \tag{A.15}$$

where Ψ is some generation transition function selecting solutions based on Pareto optimality, and $\mathbb{P}$ (i.e., P_{known}) is returned to the DM.

A.4.4.1 Pareto-Based Selection. These approaches base selection upon each solution's assigned fitness, which is derived *primarily* via Pareto ranking. They are characterized as such by their lack of Pareto rank-and niche-, deme-, or elitist-based selection characteristics. Table A.10 lists the known Pareto-based selection MOEA approaches.

Table A.10 Pareto Selection Techniques: Ranking

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
Thermodynamical Genetic Algorithm (TDGA) [181, 228] (1995)	Simulated annealing concepts used in selection; Attempts to balance population diversity and fitness; Goldberg's [126] ranking	None	(2) Numeric optimization	Binary string
Constrained Optimization by Multi-Objective Genetic Algorithms (COMOGA) [309] (1995)	Pareto ranks solutions by constraint violations; Binary tournament selection uses either adapting probability of pipe cost or Pareto rank as criterion	Gas network design	(2) Constraint violation; Network pipe cost	Variable cardinality (number of alleles) integer string; Genes are pipes' diameters

Table A.10 (continued)

Approach	Description	Application	Objectives (#)	Chromosome
Genetic Algorithm running on the INternet (GAIN) [307] (1995)	Constraints cast into objectives; Solutions ranked first by Pareto optimality, then lexicographically	Microprocessor Design	(3) Hardware budget; Power factor budget; Cycles per instruction	String of Architectural Parameters: Cache size, Cache line size, Cache associativity, Write buffer size, Number of issued instructions per clock cycle
Multiobjective GA [250] (1995)	Integrated with FORMOSA-P; Goldberg's [126] ranking	Pressurized water reactor reload core design	(3) Boron contamination, Discharge burnup; Power peaking	Integer matrices; Genes are fuel assembly layouts, burnable poison loadings, and orientations
Multiobjective GA (MGA) [202] (1995)	Modified Pareto ranking and selection schemes	Discrete time control system design	(2) Steady-state/robustness controller; Function response controller	Binary string; Genes are tuning parameter radii, angles, and coefficients
GA [101] (1995)	Modified Pareto ranking scheme	Aircraft flat panel design	(4) Panel buckling; Bay buckling; Weight; Number of frames and stiffeners	Unknown
GA [244] (1995)	Maintains population of Pareto solutions; New solutions' fitness determined by minimum (phenotype) distance from any current solution	Multiple disc brake design	(2) Brake mass; Stopping time	Unknown
Multi-Criteria GA [187, 188, 186, 245] (1996, 1996, 1996, 1995)	Maintains population of Pareto solutions; New solutions' fitness determined by minimum (phenotype) distance from any current solution	Non-linear control system design	(2) Control input; State variable description	Binary string
GA [296] (1996)	Goldberg's [126] ranking; GA applied to backpropagation neural network	Spinning production process	(2) Yarn strength; Yarn elongation	Binary string; Genes are neural net inputs
Diploid GA [334] (1996)	Separately minimizes each function, Dominated solutions removed from combined populations	None	(3) Numeric optimization	Implies real values
ES [36, 38, 39] (1996)	Models sharing when $P_{current}$ grows too large; Method variation incorporates constraint handling	Controller design	(2) Rise time; Settle Time	Real values

Table A.10 (continued)

Approach	Description	Application	Objectives (#)	Chromosome
Implicit Multiobjective Parameter Optimization Via Evolution (IMPROVE) [12] (1996)	Pareto-based tournament selection; Prevents "clones" within a generation	Wing design	(3) Wing area; Wing lift; Lift/drag ratio	Binary string
Pareto GA [257, 258] (1996)	Employs toroidal grid; Local Pareto-based tournament selection; Specialized crossover EVOP	Airfoil & Aircraft propulsion system design	(2) Lift; Drag & Fuel consumption; Fuel consumption; Excess power	Binary string & Binary string
GA [294] (1996)	Goldberg's [126] Pareto ranking; Incorporates local search; Fitness derived via rank and feasibility	Pump scheduling model	(2) Energy/constraint violation; Switches	Binary string; Genes are pump switches (on or off)
GA [147] (1996)	Initial population seeded with local optimum and mutated copies	Finite impulse response filter design	(2) Phase linearity error; Response magnitude error	Binary string
Multiobjective structured GA (SGA) [317] (1996)	Used in H^∞ optimization; Control genes (de)activate coefficient genes; Uses Fonseca's ranking [114]	Controller design	(4) Weighting function values	Binary string
GP [193, 192] (1996,1995)	Implies "standard" GP	List construction (using simple data structures)	(2) CPU usage; Memory usage	Unknown
ES [37] (1997)	Adds several classes of constraint violations in ranking infeasible individuals	None	(2) Numeric optimization	Real values; Genes are wing characteristics
Parallel Multiobjective GA [3] (1997)	Unspecified Pareto ranking scheme	Pairwise object recognition parameter selection	(3) Histogram distance; Variant set; Histogram area	Unknown
GA [137] (1997)	Specialized dominance definition; 224 decision variables	Automotive steering box design	(6) Assembly cost; Assembly cycle time; Product reliability; Maintenance cost; Production flexibility; Redesign/modification flexibility	Implies mix of continuous and discrete decision variables
GA [287] (1997)	Goldberg's [126] ranking; Integrated with two local search schemes; Progressive penalties	Water pump scheduling	(2) Energy cost; Pump switches	Binary string
Pareto Converging GA (PCGA) [183] (1997)	Rank-ratio histograms indicate convergence; Steady-state implementation	Pattern-space partitioning	(6) Hypersphere overlap; Hypersphere dimensionality; Data point inclusion; Hypersphere classification rate; Partition compactness; Included patterns	Binary string

Table A.10 (continued)

Approach	Description	Application	Objectives (#)	Chromosome
Multiobjective GA [52] (1997)	Fuzzy logic penalty function transforms MOP into unconstrained one; Uses bounded P_{known}	4-bar pyramid truss & 72-bar space truss & 4-bar plane truss	(2) Structural weight; Control effort & (2) Structural weight; Strain energy & (3) Structural weight; Outer and inner node vertical displacement	Binary string
Pareto GA [333] (1997)	Both nondominated-, random walk-, and roulette wheel (one objective)- based selection	Transonic airfoil design	(2) Mach number; Lift coefficient	Binary string; Genes are airfoil parameters
MOGA [8] (1998)	***Proposes*** use of Fonseca's [108] MOGA; Uses simulation to determine performance criteria	Fuzzy logic traffic signal controller	(3) CO emissions; NO_x emissions; Mean travel time	Unknown
GA [332] (1998)	Hybridized with conjugate gradient-based local search; Acts like specialized mutation EVOP	Airfoil design	(2) Drag; Pitching moment	Real values; 12 design variables
Pareto Converging GA [184] (1998)	Uses variable length chromosomes; Seeds initial population; Incorporates specialized search heuristics	Pattern space partitioning	(7) Number of hypershpheres; Learning complexity; Decision surface regularity; Included class patterns; Included patterns; Partition overlap; Surface area	Implies real values

A.4.4.2 Pareto Rank- and Niche-Based Selection. These approaches base selection upon each solution's assigned fitness, derived via Pareto ranking ***and*** shared fitness (see Goldberg [131] and Deb [82] for an in-depth fitness sharing explanation). Single objective EAs use shared fitness and niching to find and maintain multiple subpopulations defining multiple optima. Although the Pareto front is a single optima it is composed of at most an uncountably infinite number of vectors. Sharing is thus used in MOEAs to (attempt to) maintain a population uniformly spread along the Pareto front (Section 3.3.2.3 discusses the two major MOEA fitness sharing methods in detail). Table A.11 lists the known Pareto ranking- and niching-based MOEA approaches.

Table A.11 Pareto Selection Techniques: Ranking and Niching

Approach	Description	Application	Objectives (#)	Chromosome
Multiple Objective Genetic Algorithm (MOGA) [108, 110, 114] (1993, 1995, 1998)	Fonseca's [111] ranking; Incorporates niching and goals (preferences)	Step response of gas turbine engine	(4) "Reach" time; "Settle" time; Overshoot; Error	Binary string; Genes are five controller parameters
Nondominated Sorting GA (NSGA) [306] (1994)	Assigns and shares dummy fitnesses in each front; Goldberg's [126] ranking	None	(2) Numeric optimization	Binary string
Niched Pareto GA (NPGA) [155] (1994)	Specialized Pareto domination tournaments	Groundwater containmant monitoring	(2) Plumes detected; Average volume detected	Binary string; Genes are x, y, z coordinates
Pareto GA [221] (1995)	Uses the NSGA [306]	Electromagnetic absorber design	(2) Absorber layer thickness; Electromagnetic reflection	Binary string; Genes are layer's material type and thickness
NSGA [87] (1995)	NSGA [306]; Compares real valued GA with simulated binary crossover against binary encoded GA	Welded beam design	(2) Cost; End deflection	Real values
NSGA [340] (1996)	Uses the NSGA [306]; Population size of 8,000	Microwave absorber design	(2) Thickness; Reflectance	Unknown
GP [194] (1996)	First finds individual "passing" all tests, then uses multiobjective fitness; Uses variant of Horn's [154] fitness niching	Dyck language problem & Reverse Polish calculator	(2) # of correct answers; CPU time & (6) # of correct answers (5); CPU time	Programming primitives
MOGA [55, 54] (1996)	Uses Fonseca's MOGA [114]; Transcription activates only certain genes	Gas turbine engine design	(9) Rise-time (2); Settling-time (2); Overshoot (2); Channel (2); Controller complexity	Integer string
MOGA [53] (1996)	Uses Fonseca's MOGA [114]	Electromagnetic suspension control system	(7) Air gap; Passenger cabin acceleration; Control voltage; Maximum test result values (3); Unknown	Real values
Reduced Pareto Set Algorithm (RPSA) [77] (1997)	Increased selection of $P_{current}$; Pareto optimal solutions ranked according to niche count	Polymer Extrusion	(4) Mass output; Melt temperature; Screw length; Power consumption	Unknown
Multiple Criteria GA (MCGA) [319] (1997)	Selection draws from current and secondary population; Specialized EVOPs	Containership loading	(4) Proximity; Transverse center of gravity; Vertical Center of Gravity; Unloads	Integer string; Genes are possible (available) placement locations
Multiple Criteria GA (MCGA) [320] (1997)	Indirect chromosome representation; Simulation-derived fitness	Scheduling shipyard plate cutting	(2) Makespan; Penalty costs	Integer string; Genes are machine choices and job lists
GA [210] (1997)	Parallelized; Integrated with CFD and CEM codes; Uses NSGA [306] with tournament selection	Two-dimensional airfoil design	(2) Drag coefficient; Transverse magnetic radar cross section	Real values

Table A.11 (continued)

Approach	Description	Application	Objectives (#)	Chromosome
Multiobjective GA [242, 238] (1997)	Uses Fonseca's MOGA [108]; Elitist selection; Integrated with Navier-Stokes code	Cascade airfoil design	(3) Pressure rise; Flow turning angle; Pressure loss	Real values
MOGA [178] (1997)	Uses Fonseca's MOGA [108]; No apparent DM interaction	Rule-based airplane design	(3) Engine cost; Fuselage cost; Wing cost	Integer string; Genes are engine, fuselage, and wing parameters
MOGA [293] (1997)	Restricts mating; Uses "extra" objectives to retain diverse populations	Active magnetic bearing controller design	(11) Steady state error (2); Compliance (2); Current (2); Noise susceptibility (2); Complexity, Eigenvalue, Simulation length	Gray coded binary string; 402 bits
Structured messy GA [142] (1997)	Begins with 1-bit strings and gradually increases length each generation; Evaluates ***only*** active decision variables; One objective is weighted-sum of four others; Goldberg's [126] Pareto ranking	Water pipe network rehabilitation	(2) Benefit; Cost	Integer string; Genes are pipe and decision numbers
NSGA [329] (1997)	Uses NSGA [306]; Compares results to those obtained via a weighted-sum technique	Investment portfolio optimization	(2) Expected return; Risk	Binary string
Multi-Objective EP (MOEP) [232] (1998)	$(\mu + \lambda)$ elitist strategy; $P_{current}$ solutions selected with high probability	Voltage reference circuit parameter optimization	(2) Room temperature reference voltage; Temperature variation	Implies real values
MOGA [240] (1998)	Uses Fonseca's [108] MOGA; Integrated with Navier-Stokes and Squire-Young codes	Transonic wing design	(2) Lift; Drag	Real values
Multi-Objective Genetic Programming (MOGP) [151] (1998)	Uses Fonseca's [108] MOGA; Results compared to single-objective GP on same problems	Model derivation for distillation column and cooking extruder	(4) RMS error; Residual variance; Residual and output correlation; Model string length	Unknown
Strength Pareto Evolutionary Algorithm (SPEA) [358] (1998)	Actively uses secondary population in fitness assignment and selection; Uses clustering to reduce secondary population size; Pareto-based niching parameter	None	(2,3,4) Combinatorial optimization example (0/1 knapsack problem)	Binary string; Genes are items present in ith knapsack

Table A.11 (continued)

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
Multi-level MOGA [56] (1998)	Uses Fonseca's [108] MOGA to develop satisfactory controllers at discrete design points; Another MOGA then uses P_{known} to determine satisfactory overall controller	Gas turbine controller design	(7) [1st MOGA] Pole magnitude; Gain and phase margin; Rise and settle time; Overshoot; Error & [2nd MOGA] (5) Rise and settle time; Overshoot; Error; Temperature	Real values; Genes are engine gains and time constraints
Modified NSGA [224] (1998)	Weighted penalties used in *each* objective function; Simulation determines chromosome fitness	Industrial nylon 6 reactor	(2) Reaction time; Cyclic concentration	Binary string; Genes are control variable history and value
MOGA [148][2] (1998)	Uses Fonseca's [108] MOGA	Robot sensory-action neural network design	Unknown	Unknown
NSGA [84] (1999)	Defines NSGA specifics; Discusses proposed MOP test problems [83]	Welded beam design	(2) Cost; End deflection	Real values
GA [286] (1999)	Implies EVOPs guarantee feasible solutions; Compares 10 runs using different parameter values	Pulp and paper mill operation	(2) Energy consumption; Production rate change	Real values

A.4.4.3 Pareto Deme-Based Selection. The traditional EA *island* model is composed of several separate demes or subpopulations. The underlying idea is that these separate subpopulations are divisions of one overall population, but each subpopulation is evolving (somewhat) independently of the others. Each deme is interconnected by some defined topology or geographic structure used for communication; these communication channels are normally used for the occasional migration of individuals between demes.

At one extreme of the island model (all demes are fully interconnected) the island model mimics a single, large population. At the other extreme where communication is minimized the model mimics several independent EA trials. The island model can be executed on a sequential processor but its power when using multiple processors is readily apparent.

We define Pareto deme-based selection as a technique whereby an MOEA uses both a solution's Pareto ranking *and* its location within some sort of geographical structure

[2]Abstract only; the article is in Japanese.

imposed upon the population as criteria for selection. Table A.12 lists the known Pareto deme-based selection MOEA approaches.

Table A.12 Pareto Selection Techniques: Demes

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA with Redundancies [21] (1995)	85% of new population selected via local geographic mating; 15 % via binary tournament; Parallelized	Mass-Transit Vehicle Scheduling	(2) Number of vehicles; Number of "deadheading" trips	Integer matrix; Permutation ordering; Genes are vehicles assigned a trip; Recessive genes used
Genetic Programming [191] (1995)	Pareto-based tournament and demic selection; Non-elitist fitness based on primitives "passing" a series of test functions	Evolution of primitives implementing a FIFO queue	(6) Number of tests (5) passed by specific functions; Number memory cells used	Program trees; Genes are queue and shared memory primitives
Hybrid GA [256] (1995)	Local geographic selection via Pareto tournaments; Hybridized; Parallelized	Aerodynamic shape parameterization	(2) Pressure distributions	Integer string and real-valued vector
GA [2] (1997)	Island model with Pareto ranking; EVOPs operate on ***sub-populations***, not individuals	Pairwise object recognition parameter selection	(3) Histogram distance; Variant set; Histogram area	Array; Genes are histogram, type, and distance
Hybrid GA [291] (1997)	Problem specific heuristics and operators combine various constraints and objectives within single optimization step; Pareto concepts select desirable shapes; Gene pool recombination; Parallel "island model" implementation	Facility & VLSI macro-cell layout	(2) Minimal flow cost; Admissible shape constraints & (2) Layout area; Routing	Binary slicing tree; Genes are blocks defining layout or packing pattern
Virtual Subpopulation GA (VSGA) [262] (1998)	Parallel implementation; Toroidal structure; Elitist strategy; Evaluates m solutions simultaneously; Last step is selecting 1 solution from P_{known} and optimizing for drag alone	Transonic flow wing design	(2) Drag; Weight	Real values; Genes are taper ratio, chord, twist angle, and wing root thickness
Multi-OBjective Evolutionary Algorithm (MOBEA) (ES) [40, 35] (1999)	Parallel implementation; Uses preselection "window" in identifying solutions for mutation and recombination	None	(2) & (3) Numeric optimization	Unknown

A.4.4.4 Pareto Elitist-Based Selection. Elitist selection ensures the best (or n best) individuals are retained in the next generation. Pareto elitist-based techniques thus first select some number of top Pareto-ranked individuals and the remainder of the next population is filled via some other method. Thus, these approaches primarily use a solution's "elite" (in $P_{current}(t)$) status as the selection criteria. Table A.13 lists the known Pareto elitist-based selection MOEA approaches.

Table A.13 Pareto Selection Techniques: Elitist

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
Pareto Optimal Genetic Algorithm [207, 206] (1993)	Pareto optimal solutions selected from efficient set formed by parents and offspring	None	(2) Numeric optimization	Binary string
GENMO [25, 26] (1994)	Pareto optimal solutions given rank 1; Dominated and infeasible solutions given Rank 2 and discarded	Turbomachinery airfoil design & Ceramic composite	(2) Torsional flutter margin; Torsional resonant amplitude & (2) Cost; Residual stress	Binary string
GA [348] (1994)	Selects $\frac{1}{k}$ "best" values in each objective for next population; Extinction eliminates identical individuals; Immigration of randomly generated solutions	Bicriteria linear transportation problem	(2) Cost; Deterioration	Integer matrix
Genetic Algorithm [124, 125] (1995)	Design rule-set evolves; Optimizes the inverse problem to obtain attainable criteria set; Next generation formed as per Louis [207]	Beam section topology	(2) Surface area; Moment of inertia	Binary string; Genes are sets of executable rules producing a design
GA [313][3] (1995)	Retains all (or subset of) $P_{current}$	Unknown	Unknown	Unknown
MOGA [248] (1996)	Integrated with FORMOSA-P and PANTHER; Goldberg's [126] ranking; Specialized EVOPs; Active secondary population	Pressurized water reactor reload core design	(3) Feed enrichment; Discharge burnup; Radial form factor	Three 2-D integer matrices; Genes are fuel assembly layouts, burnable poison loadings, and orientations
PAReto optimal and Amalgamated induction for DEcision trees (PARADE) [351] (1996)	Attempts to unify feature subset selection, generalization, and pruning methods; Discards all non-Pareto solutions	Decision tree induction	(2) Error rate; Number of leaf nodes	S-expression representing decision tree
Parallel Diffusion GA [274] (1996)	Reproduction only with immediate neighbors; Elitist Pareto selection between offspring and one parent	Solution sensitivity analysis	(2) Solution quality change; Number of considered parameters	Binary string; Genes are problem parameters
GA [311][4] (1996)	Discards all dominated solutions; Prohibits solution duplication; Population size varies	Unknown	Unknown	Unknown

[3]Cited by Tamaki [314]; in Japanese.
[4]Cited by Tamaki [314]; in Japanese.

Table A.13 (continued)

Approach	Description	Application	Objectives (#)	Chromosome
Pareto GA [9] (1996)	Simulation derives fitness estimation	Ballistic weapon performance	(2) RMS position error; RMS Euler angle error	Binary string
GA [10] (1996)	Retains best performing solution for each individual objective each generation	Ballistic weapon design	(2) RMS position error; RMS Euler angle error	Binary string
Multiobjective GA [249] (1998)	Integrated with problem domain code; Specialized EVOPs; Secondary population used	Pressurized water reactor reload design	(3) Feed enrichment; Discharge burnup; Power peaking	Three 2-dimensional arrays
Nondominated Sorting GA [169] (1998)	No explicit fitness; Nonlinear rank selection probability assigned on basis of Pareto rank and feasibility; Uses crowding and specialized EVOPs	None	(2) Numeric optimization	Real values
GA [118] (1998)	Uses penalty functions; NPGA [155] fitness sharing; Elitist selection with "life span"; Fitness based on distance from $P_{current}$	Automotive engine design	(4) Fuel consumption; Following and starting responses; Acceleration	Real values; Genes are engine parameters
GA [132] (1998)	Specialized EVOPs; Uses three different population initialization schemes	Fuzzy modeling	(2) Quadratic mean error; # of rules	Binary, integer, and real values; Genes are rule and control set values
GA [246] (1998)	Fonseca's [108] ranking and fitness sharing; Best "N" selection; Uses Taguchi method to analyze parameter epistasis	Aerodynamic wing optimization	(2) Lift; Drag	Real values; Variables encoded in a tree structure; Genes are wing parameters
MOGA [312] (1998)	Fonseca's [108] MOGA; Simulation derived fitness; uses elitism and "coevolutionary shared niching"	Transonic wing design	(3) Drag; Wing weight; Fuel weight	Real values
MOGA [209] (1998)	NSGA [306] with tournament selection; Specialized EVOPs; All solutions in $P_{current}$ placed in next generation	Two dimensional airfoil design	(2) Drag coefficient; Backscatter wave	Real values

A.4.5 Hybrid Selection Techniques. Hybrid selection techniques directly utilize an MOEA's population capability. Here, succeeding populations are selected using two or more cooperative search techniques (which are not necessarily identical for each evaluated solution, i.e., multiple strategies may be used each generation). These techniques are mathematically represented by:

$$\begin{aligned} \Psi &: P \longrightarrow P' \\ \mathbb{P} &= \{a_i | \forall a_i, a_i \in P_{known}\} , \end{aligned} \tag{A.16}$$

where Ψ is some generation transition function selecting solutions based on their performance using a particular selection technique, and $\mathbb{P}$ (i.e., P_{known}) is returned to the DM. Table A.14 lists the known hybrid selection MOEA approaches.

Table A.14 Hybrid Selection Techniques

Approach	Description	Application	Objectives (#)	Chromosome
Fuzzy Reduction GA (FuReGA) [336] (1998)	Fuzzy logic-based selection decisions; Uses elitist, Pareto ranking, VEGA, or VEGA variant	Railway timetables	(2) Waiting time; Investment cost	Unknown
GA [357] (1999)	Selection via adaptive objective evaluation hyperplane [356] or NSGA [306]	None	(2) Numeric optimization	Prüfer number encoding; Integer string uniquely encodes a spanning-tree

A.5 MOEA Comparisons and Theory

In addition to proposed MOEA techniques, several MOEA research efforts focus on the comparison and theoretical aspects of MOEAs. This section catalogues publications classified in these categories.

A.5.1 MOEA Technique Comparisons.

Several citations not only introduce some new MOEA technique, but also compare the new approach to an existing one(s). Other citations simply apply different MOEAs to some problem and compare/contrast the results. Table A.15 lists the known efforts comparing different MOEA performances.

Table A.15 Technique Comparisons

Approach	Description	Application	Objectives (#)	Chromosome
GA [200, 149] (1990, 1989)	Compares VEGA [289] and Goldberg's ranking [126]; Specialized crossover	Set covering problem & Scheduling algorithm parameter search	(2) Cost; Violated constraints & (4) Fitness function weights	Binary string

Table A.15 (continued)

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [140] (1992)	Weighted sum with chromosomally encoded weights; Compares fitness sharing (applied only to weighting variables) and two VEGA [289] variants	Static and dynamically loaded 10-bar truss & Wing Box	(2) Structural weight; Vertical displacement & (2) Structural weight; Natural frequencies	Mix of continuous and discrete alleles; Genes are design variables and weights
GA [344] (1993)	Compares weighted-sum, goal attainment, Goldberg's [126] Pareto ranking, and VEGA [289] on identical problems	Digital filter design	(2) RMS response error; Adder cost	Binary string; Genes are coefficients
GA [315] (1994)	2 variants: Pareto elitist and VEGA selection; VEGA selection and fitness sharing	Hot rolling process scheduling	(2) Pressing order; Slab assignment	Binary string
Multiple Objective GA [272] (1994)	2 variants: VEGA; Goldberg's [126] ranking	Groundwater pollution containmant monitoring	(2) System cost; System reliability	Binary string; Genes are operating mode and pumping rate of n wells
Modified Combinatorial ES [182] (1995)	Compares "Pure" Pareto selection and "Best per Objective Selection"	Constrained facility layout (formulated as a Restricted Quadratic Assignment Problem)	(2) Cost; Violated zoning constraints (only for Pareto variants)	Permutation ordering; Genes are machine locations
GA [58] (1995)	Compares VEGA, Tchebycheff weighting, Pareto ranking, and VEGA-Pareto GA variants	Groundwater monitoring	(2) Undetected plumes; Contaminated area	Fixed-length integer string
Modified GENESIS ([135]) [301] (1995)	Compares summation, Pareto ranking, and two fuzzy logic ranking techniques; MOP also varies in complexity	Flat aircraft panel design	(5) Weight; Man-hours; Number of stiffeners and frames; Panel buckling; Bay buckling	Implies binary string
Multiple Objective GA [297] (1996)	Compares Fonseca's MOGA [108] to separate weighted-sum runs	Meal production line scheduling	(3) Rejected orders; Batch lateness; Shift/staff balancing	Permutation ordering
Multiobjective GA [73] (1996)	Compares linear combination, "two-branch-" and Pareto domination-tournament	Rotor system design	(2) Rotor system power; Rotor system weight	Binary string
GA [314] (1996)	4 variants: Parallel selection; Pareto ranking; Tournament selection with sharing; Pareto reservation	None	(2) Numeric optimization	Binary string
GA [298] (1996)	Compares weighted-sum, MOGA [114], and parallel migration results	Production scheduling	(3) Job omission; Order lateness; Staff shift lengths	Unknown
MOGAC [90] (1997)	Fonseca's [111] ranking; Implements inverse elitism and dynamic parameter adaptation	Hardware/software co-design	(2) Cost; Power consumption	Unknown

Table A.15 (continued)

Approach	Description	Application	Objectives (#)	Chromosome
GA [263] (1997)	Hybridized with conjugate gradient-based local search; Compares simple, three hybridization, and weighted sum approaches	Airfoil design	(2) Drag; Pitching moment	Real values; 18 design variables
Fuzzy Logic-based Multiobjective GA [267] (1997)	Compares fuzzy logic-based fitness assignment and NPGA [155] results	None	(2) Numeric optimization	Binary string
Fuzzy Logic-based Multiobjective GA [268] (1997)	Compares fuzzy logic-based fitness assignment and NPGA [155] results	Born-Mayer problem	(2) & (4) & (9) Sample function values (all)	Binary string; Genes are model parameters
GA [31, 29] (1997)	Compares six weighted-sum ranking methods	None	(2) Numeric optimization	Binary string
GA [91] (1998)	Compares weighted sum, Goldberg's ranking [126], and Fonseca's MOGA [108]; Specialized EVOPs	"Cell" configuration (constrained facility layout)	(2) Yearly processing; Overall Cost	Integer string; Genes represent the number of "reactors" in the corresponding cell
MOGA [241, 239] (1998)	Compares niching and elitist models; Integrates problem domain codes	Transonic wing design	(3) Aerodynamic drag; Wing weight; Fuel tank volume or aspect structure	Real values; Genes are polar-coordinate x-y pairs
GA [78, 79] (1998)	Compares Pareto ranking, lexicographic, linear combination, VEGA, and Fourman's [117] techniques	Computer aided project study	(1-9) Take off distance; Landing speed; 2 excess power measurements; 4 turn rates; Ferry range	Real values
MOGA [359, 358] (1998)	Compares random, weighted sum, NPGA, NSGA, and VEGA MOEAs	None	(2,3,4) Combinatorial optimization example (0/1 knapsack problem)	Binary string; Genes are items present in ith knapsack
GA [50] (1998)	Compares results of GAs, tabu search, and simulated annealing	Cardinality constrained portfolio optimization	(2) Return; Risk	Appears to be real values
GA [60] (1998)	Compares weighted min-max; random, and several MOEA results	Machine tool spindle	(2) Volume; Static displacement	Real values
GA [318] (1998)	Compares Fonseca's [108] and Goldberg's [126] Pareto ranking, and also tournament selection; Population has multiple, non-interbreeding species; Uses penalty function	Full stern submarine design	(2) Volume; Power	Binary string
GA [63, 64, 66, 67] (1998, 1999)	Compares weighted min-max; random, and several MOEA results	I-beam design & Machining parameters	(2) Cross-section; Static deflection & Surface roughness; Surface integrity; Tool life; Metal removal rate	Real values

Table A.15 (continued)

Approach	Description	Application	Objectives (#)	Chromosome
GA [1] (1998)	Compares distance, average, and Pareto rankings	Self-organizing fuzzy logic controller rule-base optimization	(2) Absolute error integral; Controller effort integral	Binary string; Genes are rule parameters
GA [208] (1998)	Compares Pareto and fuzzy logic-based rankings	Muscle relaxant anesthesia model	(2) Absolute error and control effort integral; Time absolute error and control effort integral	Binary string; Genes are rules

A.5.2 MOEA Theory and Reviews. Many of the preceding cited efforts at least pay "lip service" to different facets of underlying MOEA theory, but make no significant contribution when simply citing relevant issues raised by others. However, some (e.g., Fonseca [114] and Horn [154]) go into significant theoretical detail. Their work provides basic MOEA models and theories which are addressed in Section 3.3.2. Other recent papers also focus on MOEA theory and use application examples to illustrate key concepts. Finally, four major MOEA reviews exist [111, 152, 326, 61]. Table A.16 lists the known efforts discussing MOEA theory in some detail.

Table A.16 MOEA Theory

Researcher(s)	Paper Focus
Fonseca and Fleming [109] (1995)	MOEA selection, sharing, and mating parameter values
Fonseca and Fleming [111] (1995)	MOEA review and general Pareto concepts
Fonseca and Fleming [114] (1998)	MOEA parameters and values; Goal incorporation
Horn and Nafpliotis [154] (1995)	MOEA sharing and niching values
Fonseca and Fleming [107] (1997)	MOEA mathematical formulations
Horn [152] (1997)	MOEA-Pareto observations and review
Rudolph [276] (1998)	MOEA convergence
Van Veldhuizen and Lamont [325] (1998)	MOEA convergence and Pareto terminology
Van Veldhuizen and Lamont [326] (1998)	MOEA components, Pareto characteristics, and test problems
Deb [83] (1998)	Constructing bi-objective MOEA test problems
Coello [61] (1999)	MOEA technique review
Van Veldhuizen and Lamont [327] (1999)	MOEA benchmark test problems

Appendix B. MOPs in the Literature

This section contains the tables classifying and cataloging all known (to date) MOEA test functions[1]. As previously discussed, together they form a *de facto* MOEA test function suite.

Table B.1 MOP Numeric Test Functions

Researcher & Major MOP Characteristics	Definition	Constraints
Binh (1) [36, 38]; P_{true} connected, PF_{true} convex	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= x^2 + y^2, \\ f_2(x,y) &= (x-5)^2 + (y-5)^2 \end{aligned}$$	$-5 \leq x, y \leq 10$
Binh (3) [35];	$F = (f_1(x,y), f_2(x,y), f_3(x,y))$, where $$\begin{aligned} f_1(x,y) &= x - 10^6, \\ f_2(x,y) &= y - 2 * 10^{-6}, \\ f_3(x,y) &= xy - 2 \end{aligned}$$	$10^{-6} \leq x, y \leq 10^6$
Fonseca [111]; P_{true} connected, PF_{true} concave	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= 1 - \exp(-(x-1)^2 - (y+1)^2), \\ f_2(x,y) &= 1 - \exp(-(x+1)^2 - (y-1)^2) \end{aligned}$$	None
Fonseca (2) [109]; P_{true} connected, PF_{true} concave, Analytical solution stated	$F = (f_1(\vec{x}), f_2(\vec{x}))$, where $$\begin{aligned} f_1(\vec{x}) &= 1 - \exp(-\sum_{i=1}^{n}(x_i - \frac{1}{\sqrt{n}})^2), \\ f_2(\vec{x}) &= 1 - \exp(-\sum_{i=1}^{n}(x_i + \frac{1}{\sqrt{n}})^2) \end{aligned}$$	$-4 \leq x_i \leq 4$
Kursawe[2] [189]; P_{true} disconnected, PF_{true} disconnected	$F = (f_1(\vec{x}), f_2(\vec{x}))$, where $$\begin{aligned} f_1(\vec{x}) &= \sum_{i=1}^{n-1}(-10e^{(-0.2)*\sqrt{x_i^2 + x_{i+1}^2}}), \\ f_2(\vec{x}) &= \sum_{i=1}^{n}(\lvert x_i \rvert^{0.8} + 5sin(x_i)^3) \end{aligned}$$	None

[1]Because several distinct MOPs may be created using Deb's initial methodology [83], direct implementations of those functions are not listed here.

[2]Marco Laumanns indicates this MOP was misprinted in Kursawe's original paper (personal correspondence).

Table B.1 (continued)

Researcher & Major MOP Characteristics	**Definition**	**Constraints**
Laumanns [195]; P_{true} disconnected, PF_{true} convex	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= x^2+y^2, \\ f_2(x,y) &= (x+2)^2+y^2 \end{aligned}$$	$-50 \le x,y \le 50$
Lis [201]; P_{true} disconnected, PF_{true} disconnected and concave	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= \sqrt[8]{x^2+y^2}, \\ f_2(x,y) &= \sqrt[4]{(x-0.5)^2+(y-0.5)^2} \end{aligned}$$	$-5 \le x,y \le 10$
Murata[3] [230]; P_{true} connected, PF_{true} concave	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= 2\sqrt{x}, \\ f_2(x,y) &= x(1-y)+5 \end{aligned}$$	$1 \le x \le 4,\ 1 \le y \le 2$
Poloni[4] [257]; P_{true} disconnected, PF_{true} disconnected and convex	Maximize $F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= -[1+(A_1-B_1)^2+(A_2-B_2)^2], \\ f_2(x,y) &= -[(x+3)^2+(y+1)^2] \end{aligned}$$	$-\pi \le x,y \le \pi$, $$\begin{aligned} A_1 &= 0.5\sin 1 - 2\cos 1 + \\ &\quad \sin 2 - 1.5\cos 2, \\ A_2 &= 1.5\sin 1 - \cos 1 + \\ &\quad 2\sin 2 - 0.5\cos 2, \\ B_1 &= 0.5\sin x - 2\cos x + \\ &\quad \sin y - 1.5\cos y, \\ B_2 &= 1.5\sin x - \cos x + \\ &\quad 2\sin y - 0.5\cos y, \end{aligned}$$
Quagliarella [262]; P_{true} disconnected, PF_{true} convex	$F = (f_1(\vec{x}), f_2(\vec{x}))$, where $$\begin{aligned} f_1(\vec{x}) &= \sqrt{\frac{A_1}{n}}, \\ f_2(\vec{x}) &= \sqrt{\frac{A_2}{n}} \end{aligned}$$	$$\begin{aligned} A_1 &= \sum_{i=1}^{n}[(x_i)^2 - \\ &\quad 10\cos[2\pi(x_i)]+10], \\ A_2 &= \sum_{i=1}^{n}[(x_i-1.5)^2 - \\ &\quad 10\cos[2\pi(x_i-1.5)] \\ &\quad +10], \end{aligned}$$ $-5.12 \le x_i \le 5.12, n = 16$

[3] Tamaki [315] gives an almost identical function.

[4] The MOP appears to be mistyped in the cited paper; A later paper [258] also mistypes the function; it then modifies the original function by requiring: $x_i, y_j \in [-\pi/4, \pi/4], x = \sum_{i=1}^{4} x_i, y = \sum_{i=1}^{4} y_j$.

Table B.1 (continued)

Researcher & Major MOP Characteristics	Definition	Constraints
Rendon [324]; P_{true} connected, PF_{true} convex	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= \frac{1}{x^2+y^2+1}, \\ f_2(x,y) &= x^2+3y^2+1 \end{aligned}$$	$-3 \leq x, y \leq 3$
Rendon (2) [324]; P_{true} connected, PF_{true} convex	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= x+y+1, \\ f_2(x,y) &= x^2+2y-1 \end{aligned}$$	$-3 \leq x, y \leq 3$
Schaffer[5] [289]; P_{true} connected, PF_{true} convex, Analytical solution proved [276]	$F = (f_1(x), f_2(x))$, where $$\begin{aligned} f_1(x) &= x^2, \\ f_2(x) &= (x-2)^2 \end{aligned}$$	None
Schaffer (2) [306, 31]; P_{true} disconnected, PF_{true} disconnected	$F = (f_1(x), f_2(x))$, where $$\begin{aligned} f_1(x) &= -x, \text{ if } x \leq 1, \\ &= -2+x, \text{ if } 1 < x \leq 3, \\ &= 4-x, \text{ if } 3 < x \leq 4, \\ &= -4+x, \text{ if } x > 4, \\ f_2(x) &= (x-5)^2 \end{aligned}$$	$-5 \leq x \leq 10$
Vicini[6] [331]; P_{true} connected, PF_{true} convex	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= -(\sum_{i=1}^{20} H_i \exp[\frac{(x-x_i)^2+(y-y_i)^2}{2\sigma_i^2}]) \\ &\quad +3, \\ f_2(x,y) &= -(\sum_{i=1}^{20} H_i \exp[\frac{(x-x_i)^2+(y-y_i)^2}{2\sigma_i^2}]) \\ &\quad +3 \end{aligned}$$	$$\begin{aligned} 0 \leq H_i \leq 1, \\ -10 \leq x, x_i, y, y_i \leq 10, \\ 1.5 \leq \sigma_i \leq 2.5 \end{aligned}$$
Viennet [334]; P_{true} connected and symmetric, PF_{true} curved surface	$F = (f_1(x,y), f_2(x,y), f_3(x,y))$, where $$\begin{aligned} f_1(x,y) &= x^2+(y-1)^2, \\ f_2(x,y) &= x^2+(y+1)^2+1, \\ f_3(x,y) &= (x-1)^2+y^2+2 \end{aligned}$$	$-2 \leq x, y \leq 2$

[5] Jones et al. [170] and Norris [237] give almost identical functions; their modifications are intended to ease analysis.

[6] A three decision variable equation of the same form is also presented.

Table B.1 (continued)

Researcher & Major MOP Characteristics	Definition	Constraints
Viennet (2) [334]; P_{true} connected, PF_{true} disconnected	$F = (f_1(x,y), f_2(x,y), f_3(x,y))$, where $$\begin{aligned} f_1(x,y) &= \frac{(x-2)^2}{2} + \frac{(y+1)^2}{13} + 3, \\ f_2(x,y) &= \frac{(x+y-3)^2}{36} + \frac{(-x+y+2)^2}{8} - 17, \\ f_3(x,y) &= \frac{(x+2y-1)^2}{175} + \frac{(2y-x)^2}{17} - 13 \end{aligned}$$	$-4 \leq x, y \leq 4$
Viennet (3) [334]; P_{true} disconnected and unsymmetric, PF_{true} connected	$F = (f_1(x,y), f_2(x,y), f_3(x,y))$, where $$\begin{aligned} f_1(x,y) &= 0.5 * (x^2 + y^2) + sin(x^2 + y^2), \\ f_2(x,y) &= \frac{(3x-2y+4)^2}{8} + \frac{(x-y+1)^2}{27} + 15, \\ f_3(x,y) &= \frac{1}{(x^2+y^2+1)} - 1.1e^{(-x^2-y^2)} \end{aligned}$$	$-3 \leq x, y \leq 3$

Table B.2 MOP Numeric Test Functions (with side constraints)

Researcher & Major MOP Characteristics	Definition	Constraints
Belegundu [25]; P_{true} connected, PF_{true} connected	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= -2x + y, \\ f_2(x,y) &= 2x + y \end{aligned}$$	$0 \leq x \leq 5,\ 0 \leq y \leq 3,$ $$\begin{aligned} 0 &\geq -x + y - 1, \\ 0 &\geq x + y - 7 \end{aligned}$$
Binh (2) [37]; P_{true} connected, PF_{true} convex	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= 4x^2 + 4y^2, \\ f_2(x,y) &= (x-5)^2 + (y-5)^2 \end{aligned}$$	$0 \leq x \leq 5,\ 0 \leq y \leq 3,$ $$\begin{aligned} 0 &\geq (x-5)^2 + y^2 - 25, \\ 0 &\geq -(x-8)^2 - \\ &\quad (y+3)^2 + 7.7 \end{aligned}$$
Binh (4) [39];	$F = (f_1(x,y), f_2(x,y), f_3(x,y))$, where $$\begin{aligned} f_1(x,y) &= 1.5 - x(1-y), \\ f_2(x,y) &= 2.25 - x(1-y^2), \\ f_3(x,y) &= 2.625 - x(1-y^3) \end{aligned}$$	$-10 \leq x, y \leq 10,$ $$\begin{aligned} 0 &\geq -x^2 - (y-0.5)^2 + 9, \\ 0 &\geq (x-1)^2 + \\ &\quad (y-0.5)^2 - 6.25 \end{aligned}$$

Table B.2 (continued)

Researcher & Major MOP Characteristics	**Definition**	**Constraints**
Jimenez [169]; P_{true} connected and symmetric, PF_{true} convex	Maximize $F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= 5x + 3y, \\ f_2(x,y) &= 2x + 8y \end{aligned}$$	$x, y \geq 0$, $$\begin{aligned} 0 &\geq x + 4y - 100, \\ 0 &\geq 3x + 2y - 150, \\ 0 &\geq 200 - 5x - 3y, \\ 0 &\geq 75 - 2x - 8y \end{aligned}$$
Kita [181]; P_{true} disconnected, PF_{true} disconnected and concave	Maximize $F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= -x^2 + y, \\ f_2(x,y) &= \frac{1}{2}x + y + 1 \end{aligned}$$	$x, y \geq 0$, $$\begin{aligned} 0 &\geq \frac{1}{6}x + y - \frac{13}{2}, \\ 0 &\geq \frac{1}{2}x + y - \frac{15}{2}, \\ 0 &\geq 5x + y - 30 \end{aligned}$$
Obayashi [238]; P_{true} connected and symmetric, PF_{true} convex	Maximize $F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= x, \\ f_2(x,y) &= y \end{aligned}$$	$0 \leq x, y \leq 1$, $$x^2 + y^2 \leq 1$$
Osyczka [245]; P_{true} disconnected, PF_{true} convex	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= x + y^2, \\ f_2(x,y) &= x^2 + y \end{aligned}$$	$2 \leq x \leq 7,\ 5 \leq y \leq 10$, $$\begin{aligned} 0 &\leq 12 - x - y, \\ 0 &\leq x^2 + 10x - \\ & \quad y^2 + 16y - 80 \end{aligned}$$
Osyczka (2) [245]; P_{true} disconnected, PF_{true} disconnected	$F = (f_1(\vec{x}), f_2(\vec{x}))$, where $$\begin{aligned} f_1(\vec{x}) &= -(25(x_1 - 2)^2 + (x_2 - 2)^2 + (x_3 - 1)^2 \\ & \quad + (x_4 - 4)^2 + (x_5 - 1)^2, \\ f_2(\vec{x}) &= x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2 \end{aligned}$$	$0 \leq x_1, x_2, x_6 \leq 10$, $1 \leq x_3, x_5 \leq 5$, $0 \leq x_4 \leq 6$, $$\begin{aligned} 0 &\leq x_1 + x_2 - 2, \\ 0 &\leq 6 - x_1 - x_2, \\ 0 &\leq 2 - x_2 + x_1, \\ 0 &\leq 2 - x_1 + 3x_2, \\ 0 &\leq 4 - (x_3 - 3)^2 - x_4, \\ 0 &\leq (x_5 - 3)^2 + x_6 - 4 \end{aligned}$$
Srinivas[7] [306]; P_{true} disconnected, PF_{true} connected	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= (x-2)^2 + (y-1)^2 + 2, \\ f_2(x,y) &= 9x - (y-1)^2 \end{aligned}$$	$-20 \leq x, y \leq 20$, $$\begin{aligned} 0 &\geq x^2 + y^2 - 225, \\ 0 &\geq x - 3y + 10 \end{aligned}$$

[7]Deb uses this function with no side constraints as an example in another paper [87].

Table B.2 (continued)

Researcher & Major MOP Characteristics	Definition	Constraints
Tamaki [314]; P_{true} connected, a curved surface, PF_{true} a curved surface	Maximize $F = (f_1(x,y,z), f_2(x,y,z), f_3(x,y,z))$, where $$\begin{aligned} f_1(x,y,z) &= x, \\ f_2(x,y,z) &= y, \\ f_3(x,y,z) &= z \end{aligned}$$	$0 \le x, y, z,$ $$x^2 + y^2 + z^2 \le 1$$
Tanaka [316]; P_{true} connected, PF_{true} disconnected and convoluted	Minimize $F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= x, \\ f_2(x,y) &= y \end{aligned}$$	$0 < x, y \le \pi,$ $$\begin{aligned} 0 &\ge -(x^2) - (y^2) + 1 + \\ &\quad 0.1 * cos(16\ arctan \frac{x}{y}) \\ \frac{1}{2} &\ge (x - \frac{1}{2})^2 + (y - \frac{1}{2})^2 \end{aligned}$$
Viennet (4) [334]; P_{true} connected and unsymmetric, PF_{true} a curved surface	$F = (f_1(x,y), f_2(x,y), f_3(x,y))$, where $$\begin{aligned} f_1(x,y) &= \frac{(x-2)^2}{2} + \frac{(y+1)^2}{13} + 3, \\ f_2(x,y) &= \frac{(x+y-3)^2}{175} + \frac{(2y-x)^2}{17} - 13, \\ f_3(x,y) &= \frac{(3x-2y+4)^2}{8} + \frac{(x-y+1)^2}{27} + 15 \end{aligned}$$	$-4 \le x, y \le 4,$ $$\begin{aligned} y &< -4x + 4, \\ x &> -1, \\ y &> x - 2 \end{aligned}$$

Appendix C. P_{true} & PF_{true} for Selected Numeric MOPs

The following figures present both P_{true} and PF_{true} for each listed function in Table B.1. We stress that these figures are deterministically derived; *Pareto representations may change when computational resolution is increased/decreased.*

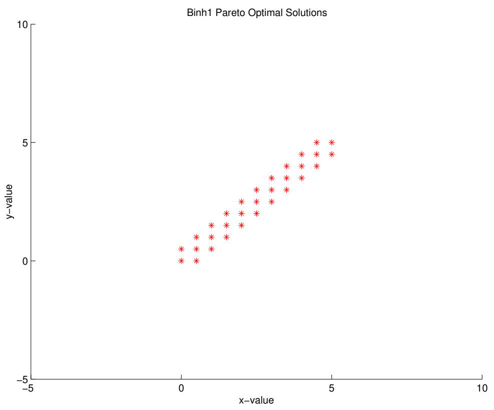


Figure C.1. Binh Pareto Optimal Set

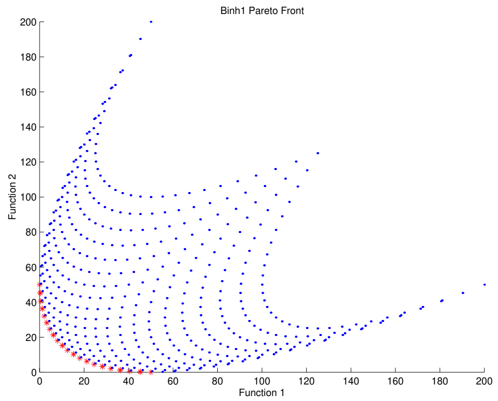


Figure C.2. Binh Pareto Front

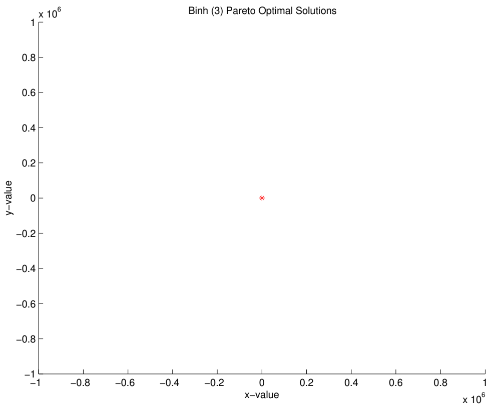


Figure C.3. Binh (3) Pareto Optimal Set

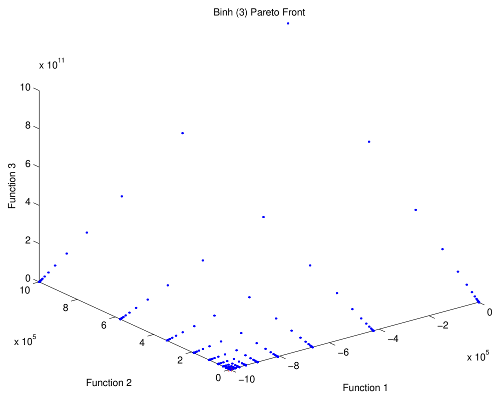


Figure C.4. Binh (3) Pareto Front

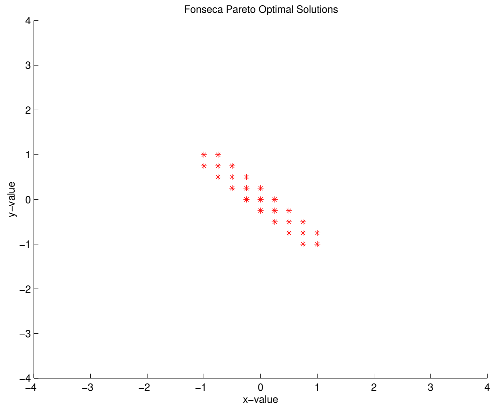


Figure C.5. Fonseca Pareto Optimal Set

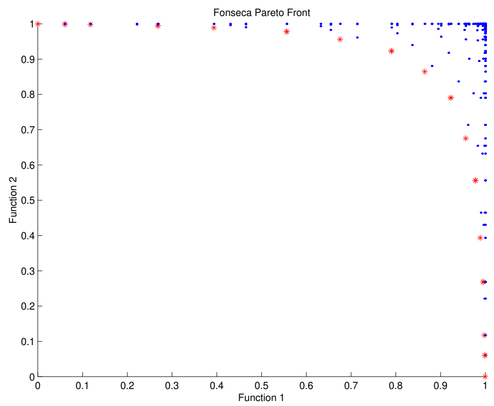


Figure C.6. Fonseca Pareto Front

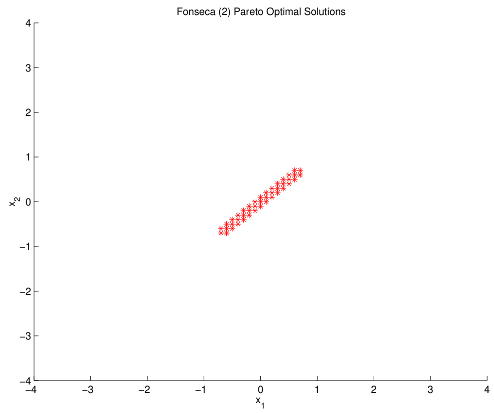


Figure C.7. Fonseca (2) Pareto Optimal Set

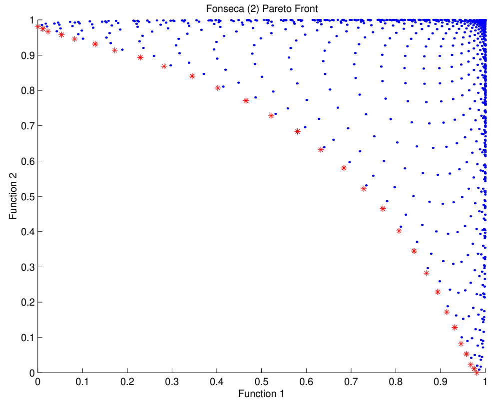


Figure C.8. Fonseca (2) Pareto Front

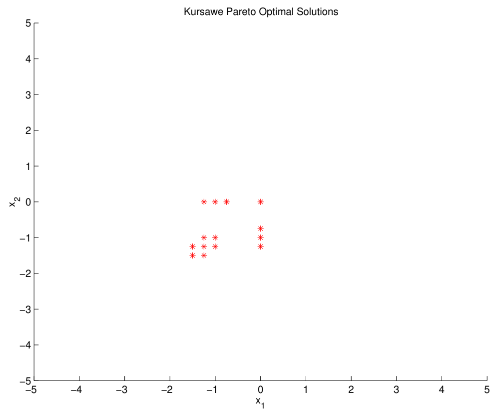


Figure C.9. Kursawe Pareto Optimal Set

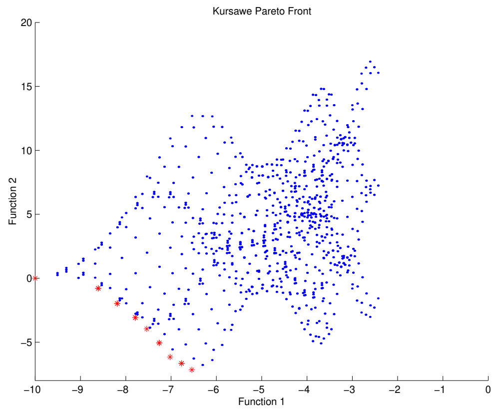


Figure C.10. Kursawe Pareto Front

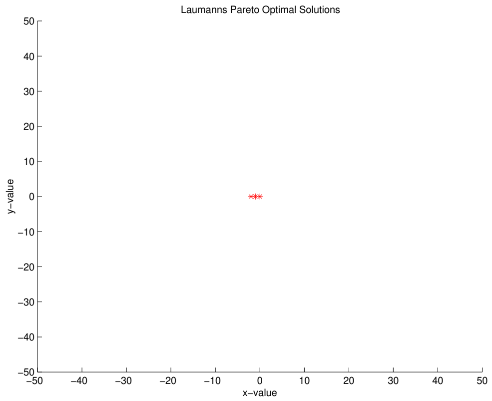


Figure C.11. Laumanns Pareto Optimal Set

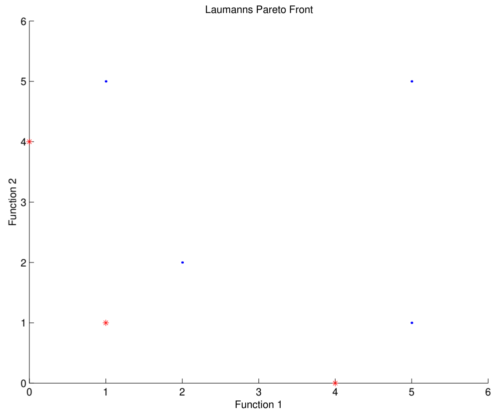


Figure C.12. Laumanns Pareto Front

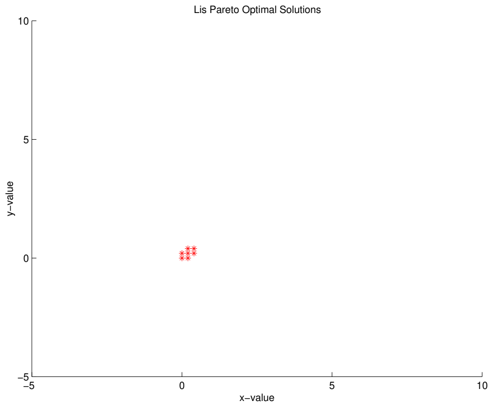


Figure C.13. Lis Pareto Optimal Set

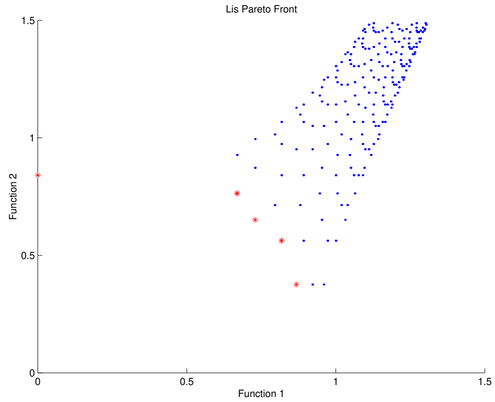


Figure C.14. Lis Pareto Front

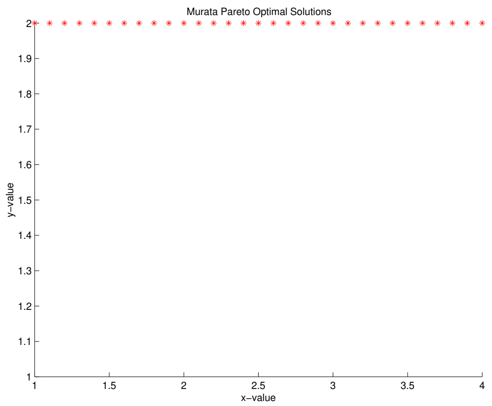


Figure C.15. Murata Pareto Optimal Set

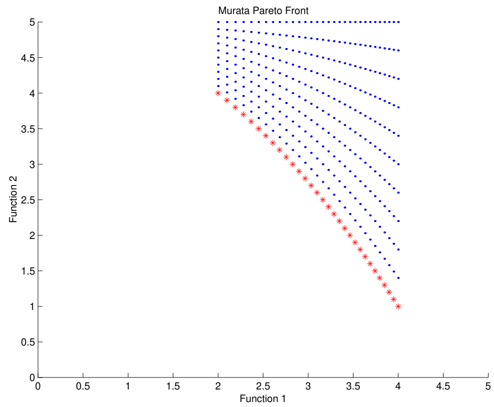


Figure C.16. Murata Pareto Front

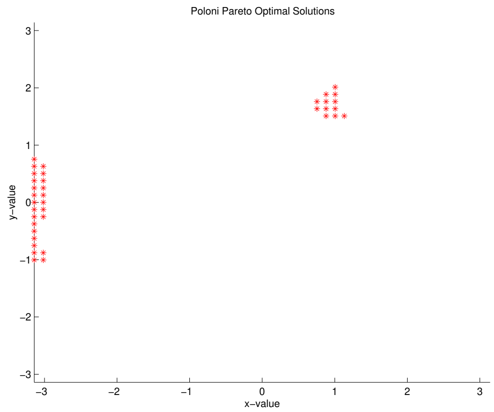


Figure C.17. Poloni Pareto Optimal Set

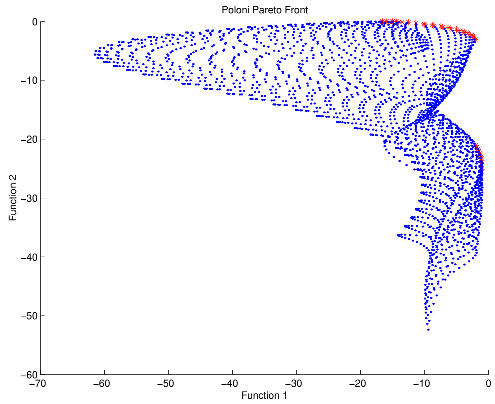


Figure C.18. Poloni Pareto Front

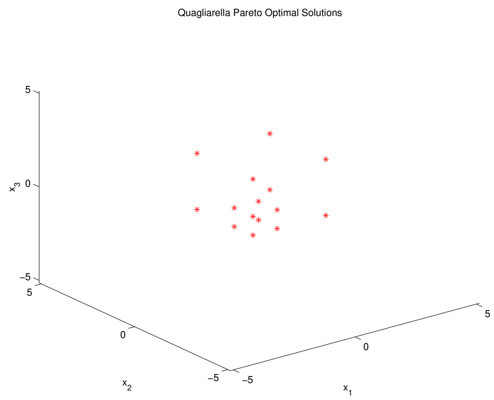


Figure C.19. Quagliarella Pareto Optimal Set (for $n = 3$)

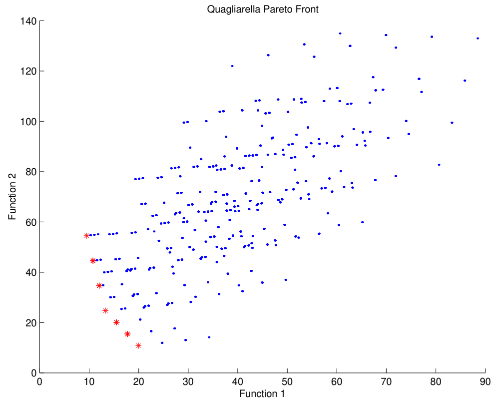


Figure C.20. Quagliarella Pareto Front (for $n = 3$)

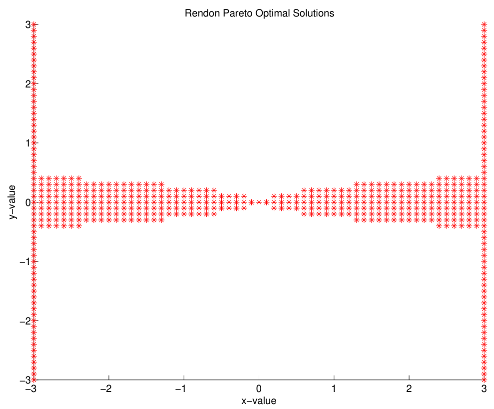


Figure C.21. Rendon Pareto Optimal Set

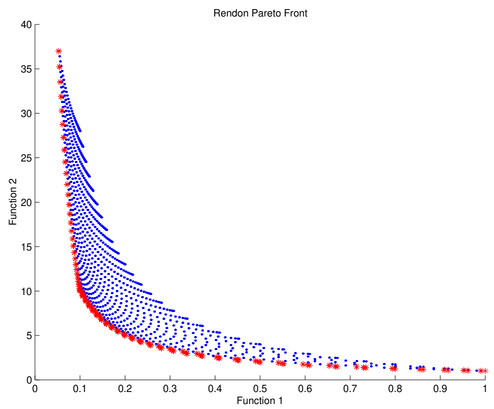


Figure C.22. Rendon Pareto Front

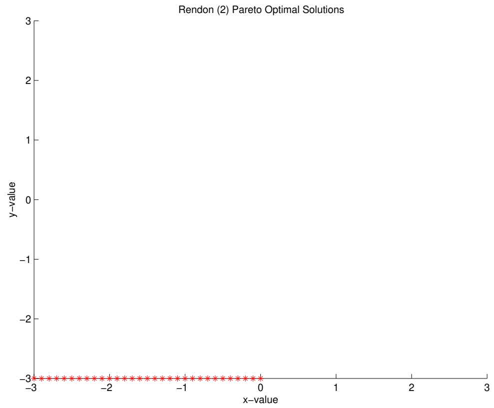


Figure C.23. Rendon (2) Pareto Optimal Set

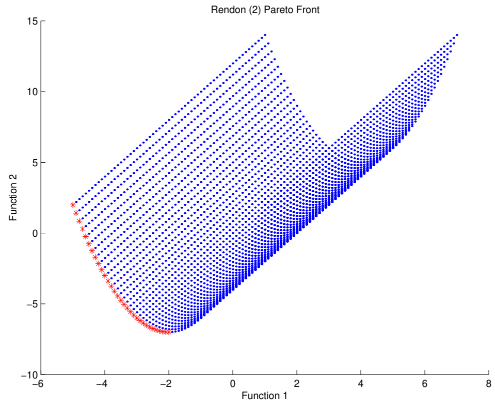


Figure C.24. Rendon (2) Pareto Front

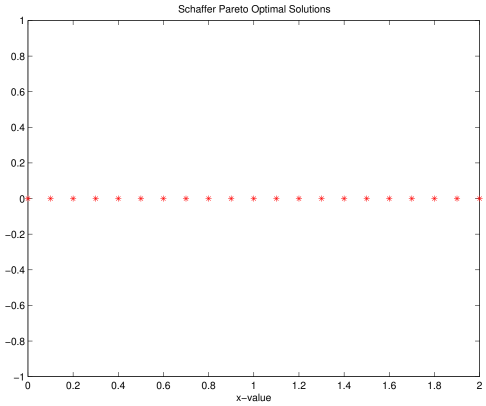


Figure C.25. Schaffer Pareto Optimal Set

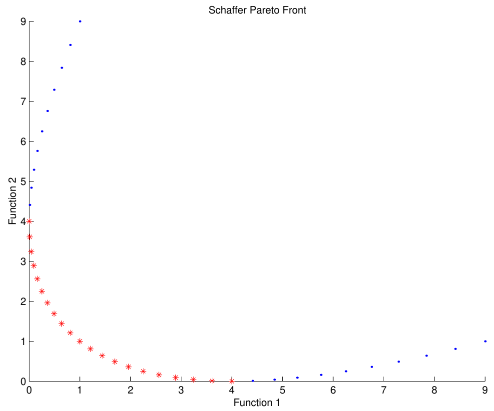


Figure C.26. Schaffer Pareto Front

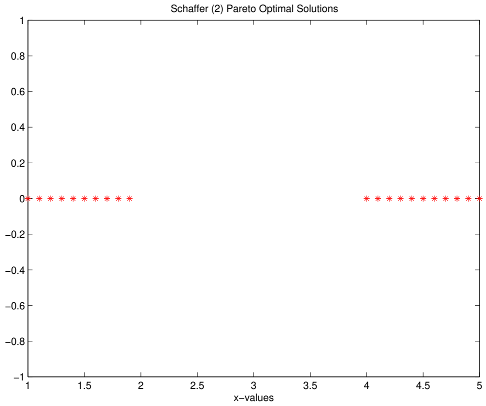


Figure C.27. Schaffer (2) Pareto Optimal Set

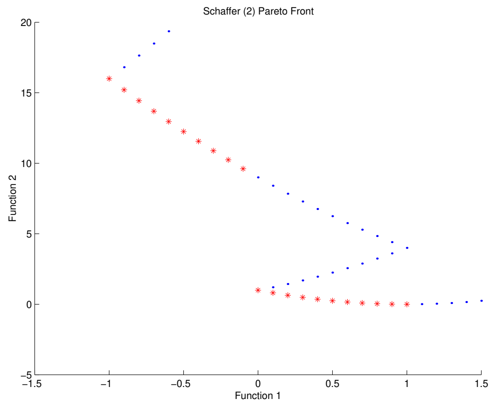


Figure C.28. Schaffer (2) Pareto Front

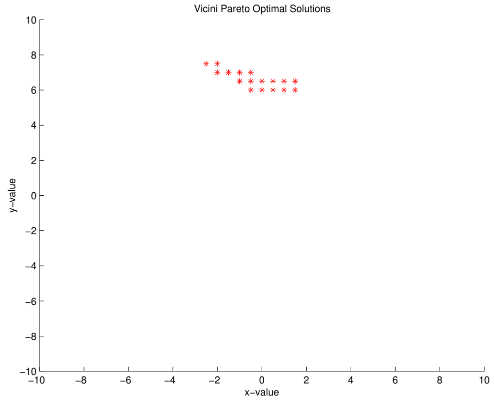


Figure C.29. Vicini Pareto Optimal Set

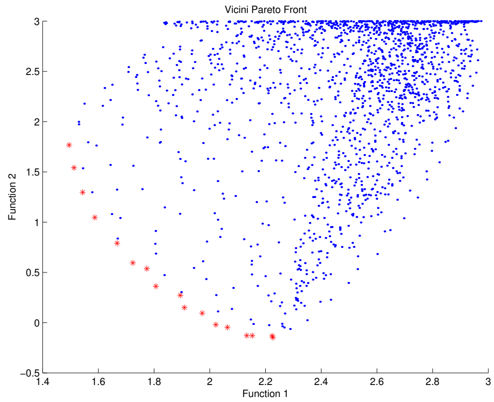


Figure C.30. Vicini Pareto Front

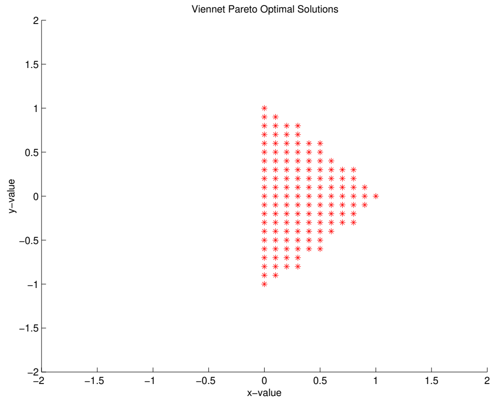


Figure C.31. Viennet Pareto Optimal Set

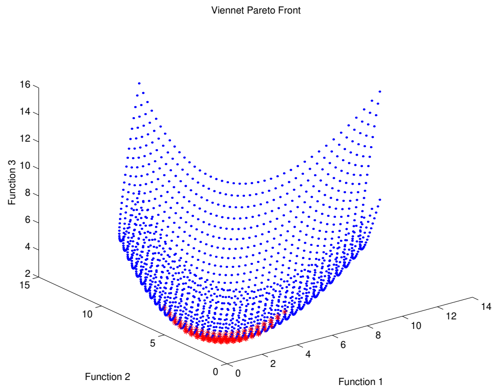


Figure C.32. Viennet Pareto Front

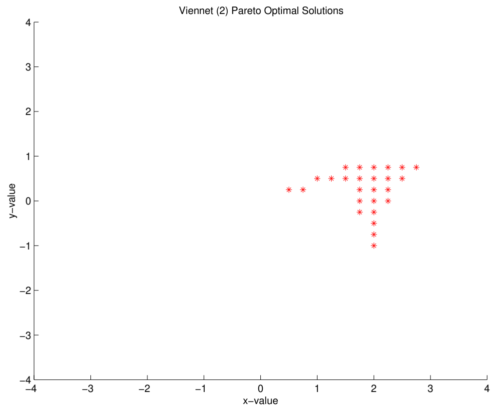


Figure C.33. Viennet (2) Pareto Optimal Set

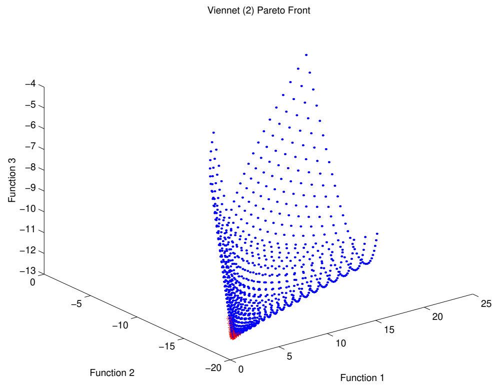


Figure C.34. Viennet (2) Pareto Front

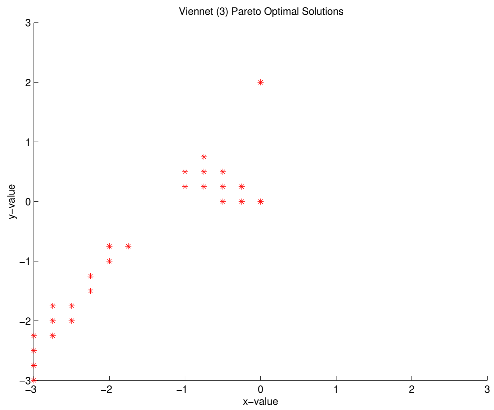


Figure C.35. Viennet (3) Pareto Optimal Set

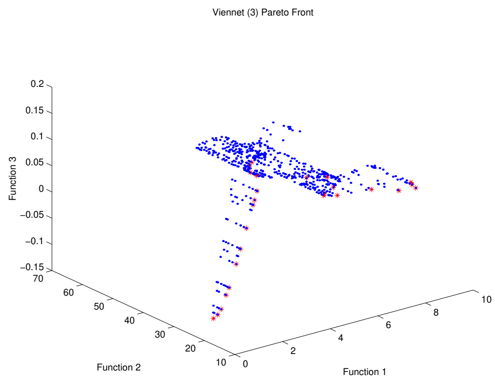


Figure C.36. Viennet (3) Pareto Front

Appendix D. P_{true} & PF_{true} for Selected Numeric (Side-Constrained) MOPs

The following figures present both P_{true} and PF_{true} for each listed function in Table B.2. We stress that these figures are deterministically derived; *Pareto representations may change when computational resolution is increased/decreased.*

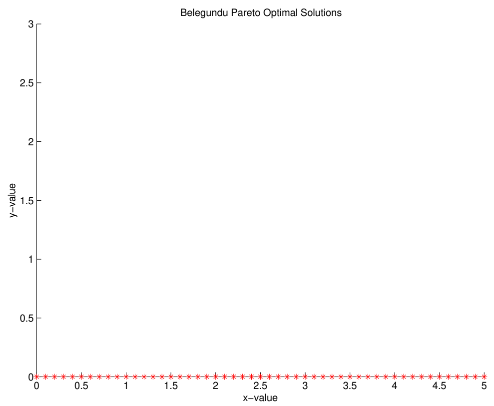


Figure D.1. Belegundu Pareto Optimal Set

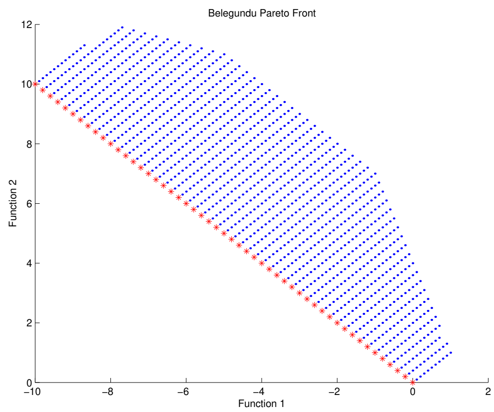


Figure D.2. Belegundu Pareto Front

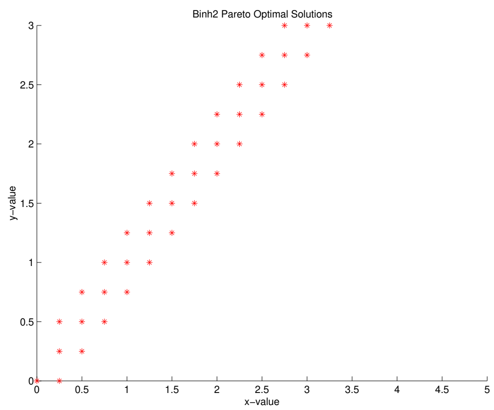


Figure D.3. Binh (2) Pareto Optimal Set

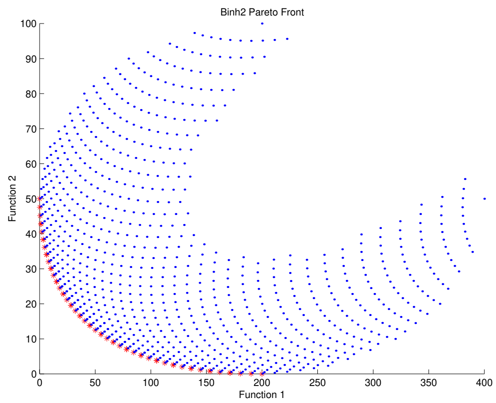


Figure D.4. Binh (2) Pareto Front

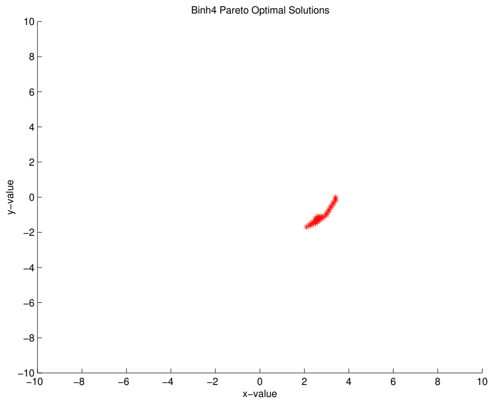


Figure D.5. Binh (4) Pareto Optimal Set

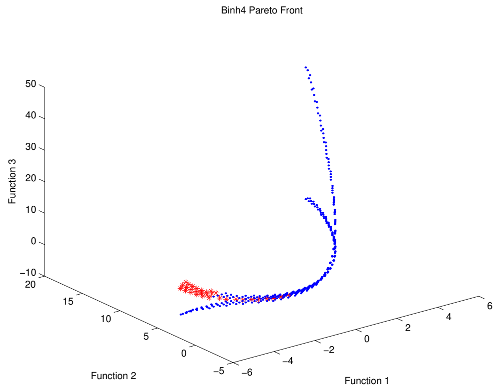


Figure D.6. Binh (4) Pareto Front

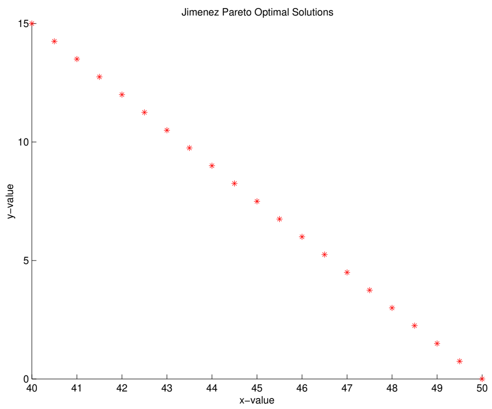


Figure D.7. Jimenez Pareto Optimal Set

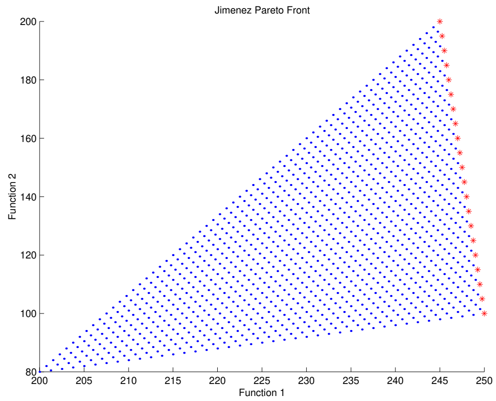


Figure D.8. Jimenez Pareto Front

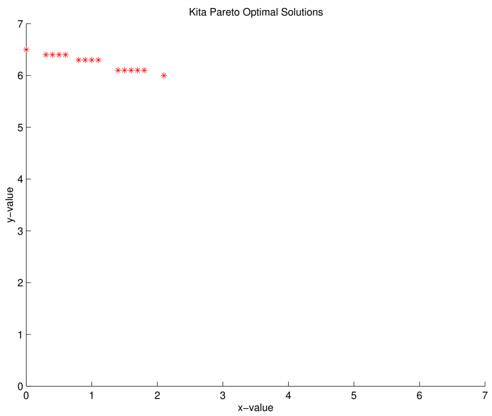


Figure D.9. Kita Pareto Optimal Set

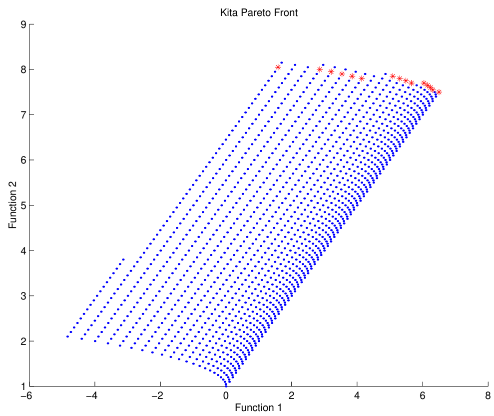


Figure D.10. Kita Pareto Front

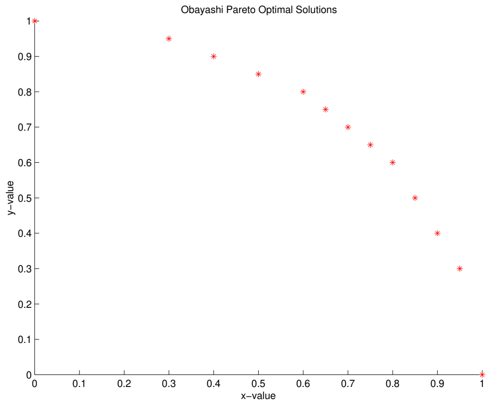


Figure D.11. Obayashi Pareto Optimal Set

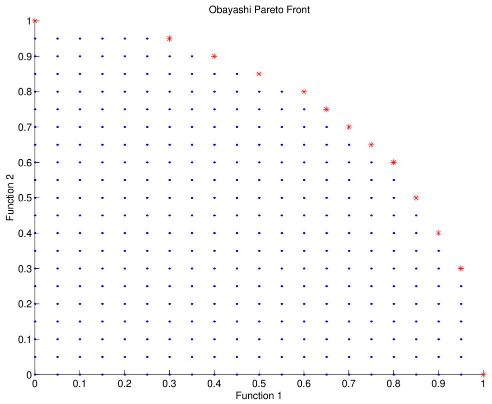


Figure D.12. Obayashi Pareto Front

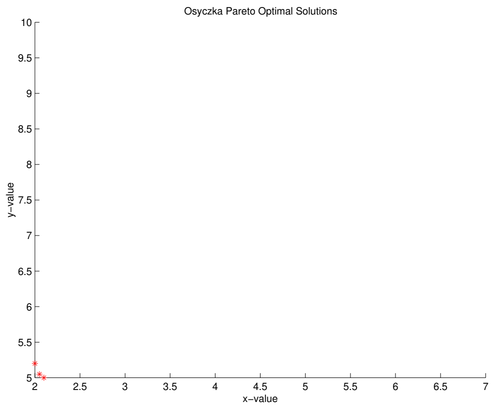


Figure D.13. Osyczka Pareto Optimal Set

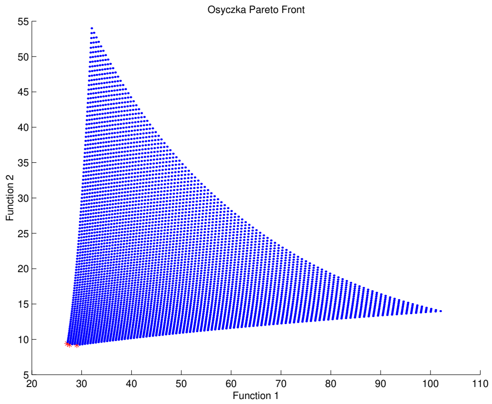


Figure D.14. Osyczka Pareto Front

Figure D.15. Osyczka (2) Pareto Optimal Set not shown ($n = 6$)

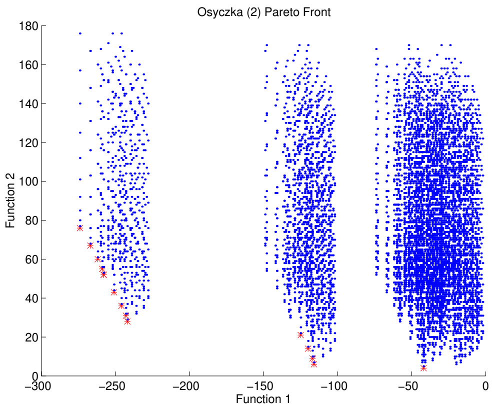


Figure D.16. Osyczka (2) Pareto Front

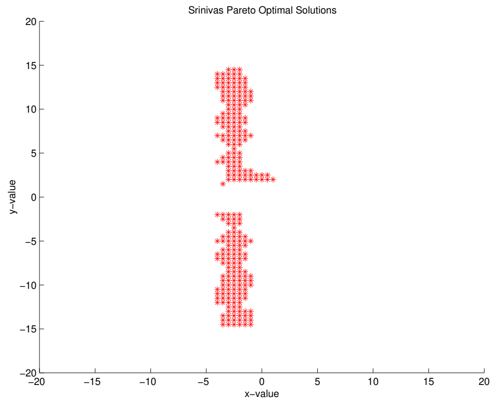


Figure D.17. Srinivas Pareto Optimal Set

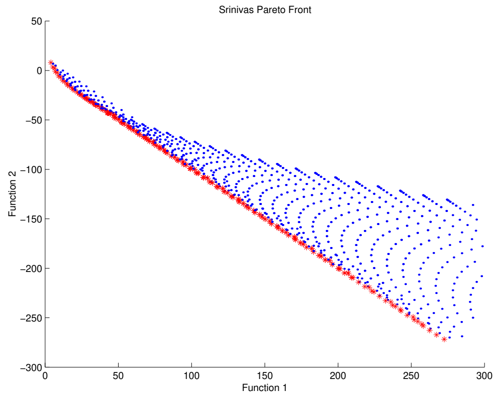


Figure D.18. Srinivas Pareto Front

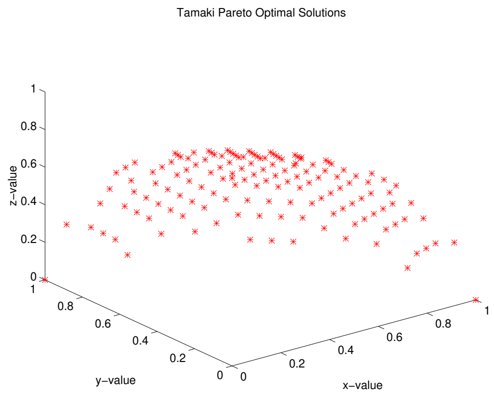


Figure D.19. Tamaki Pareto Optimal Set

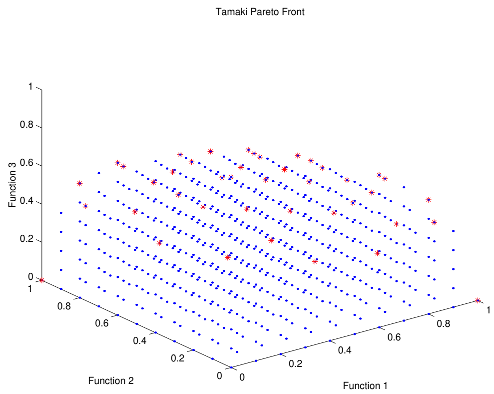


Figure D.20. Tamaki Pareto Front

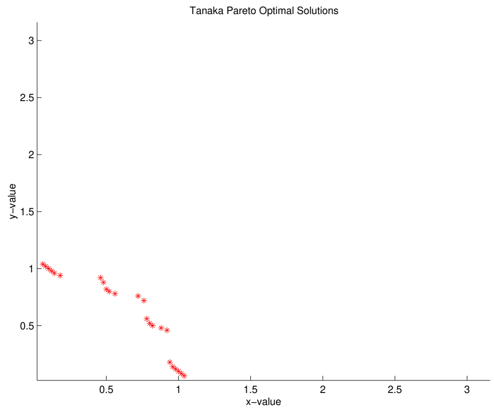


Figure D.21. Tanaka Pareto Optimal Set

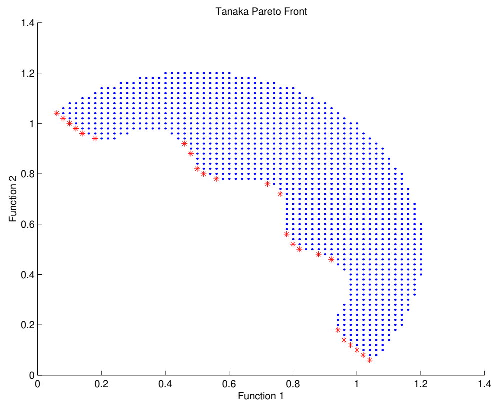


Figure D.22. Tanaka Pareto Front

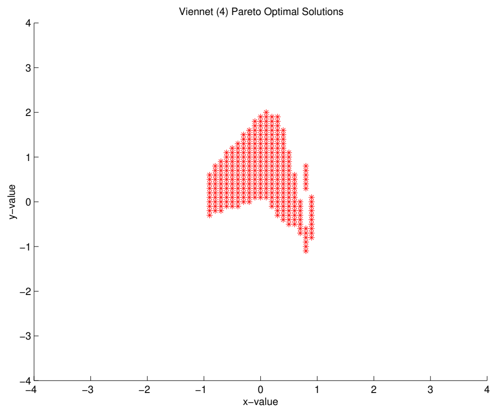


Figure D.23. Viennet (4) Pareto Optimal Set

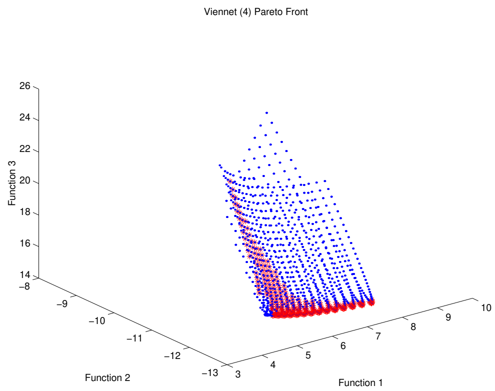


Figure D.24. Viennet (4) Pareto Front

Bibliography

1. Abbod, M. F., et al. "Multi-objective Genetic Optimization for Self-Organizing Fuzzy Logic Control." *Proceedings of UKACC Control'98*. 1575–1580. University of Wales Swansea, UK: IEE, September 1998.

2. Aherne, F. J., et al. "Optimizing Object Recognition Parameters Using a Parallel Multiobjective Genetic Algorithm." *Proceedings of the 2nd International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*. 1–6. IEE, September 1997.

3. Aherne, Frank, et al. "Automatic Parameter Selection for Object Recognition Using a Parallel Multiobjective Genetic Algorithm." *Proceedings of the Seventh International Conference on Computer Analysis of Images and PatternsLecture Notes in Computer Science 1296*. 559–566. Berlin: Springer, 1997.

4. Allen, Arnold O. *Probability, Statistics, and Queuing Theory with Computer Science Applications* (Second Edition). Boston, MA: Academic Press, Inc., 1990.

5. Allenson, Robin. *Genetic Algorithms with Gender for Multi-function Optimization*. Technical Report EPCC-SS92-01, 5 Forrest Hill, Edinburgh EH1 2QL: University of Edinburgh, September 1992.

6. Alligood, Kathleen T., et al. *Chaos: An Introduction to Dynamical Systems*. New York, NY: Springer, 1996.

7. Alotto, P., et al. *Multiobjective Optimization in Magnetostatics: A Proposal for Benchmark Problems*. Technical Report, Graz, Austria: Institut für Grundlagen und Theorie Electrotechnik, Technische Universität Graz, 1996. http://www-igte.tu-graz.ac.at/berl01.htm.

8. Anderson, J. M., et al. "Optimization of a Fuzzy Logic Traffic Signal Controller by a Multiobjective Genetic Algorithm." *Proceedings of the Ninth International Conference on Road Transport Information and Control*. 186–190. London: IEE, April 1998.

9. Anderson, M. B. and W. R. Lawrence. "Launch Conditions and Aerodynamic Data Extraction By An Elitist Pareto Genetic Algorithm." *AIAA Atmospheric Flight Mechanics Conference*. San Diego, California: AIAA Paper 96-3361, July 1996.

10. Anderson, M. B., et al. "Using an Elitist Pareto Genetic Algorithm for Aerodynamic Data Extraction." *4th Aerospace Sciences Meeting and Exhibit*. Reno, Nevada: AIAA Paper 96-0514, January 1996.

11. Anderson, Murray B. "The Potential of Genetic Algorithms for Subsonic Wing Design." *1st AIAA Aircraft Engineering, Technology, and Operations Congress*. Los Angeles, California: AIAA Paper 95-3925, September 1995.

12. Anderson, Murray B. and Glenn A. Gebert. *Using Pareto Genetic Algorithms for Preliminary Subsonic Wing Design*. Technical Report AIAA-96-4023-CP, Washington, D.C.: AIAA, 1996.

13. Anton, Howard. *Calculus with Analytic Geometry* (2nd Edition). New York: John Wiley & Sons, 1984.

14. Arslan, T., et al. "Structural Synthesis of Cell-based VLSI Circuits using a Multi-Objective Genetic Algorithm," *IEE Electronic Letters*, *32*(7):651–652 (March 1996).

15. Atkinson, Rick. *Crusade: The Untold Story of the Persian Gulf War*. New York, NY: Houghton Mifflin Company, 1993.

16. Awadh, B., et al. "A Computer-Aided Process Planning Model Based on Genetic Algorithms," *Computers in Operations Research*, *22*(8):841–856 (1995).

17. Bäck, Thomas. *Evolutionary Algorithms in Theory and Practice*. New York: Oxford University Press, 1996.

18. Bäck, Thomas, editor. *Proceedings of the Seventh International Conference on Genetic Algorithms*, San Francisco, CA: Morgan Kaufmann Publishers, Inc., July 1997.

19. Bäck, Thomas, et al., editors. *Handbook of Evolutionary Computation*, *1*. IOP Publishing Ltd. and Oxford University Press, 1997.

20. Bäck, Thomas and Zbigniew Michalewicz and Hiroaki Kitano, editor. *Proceedings of the Third IEEE Conference on Evolutionary Computation*, Piscataway NJ: IEEE Service Center, 1996.

21. Baita, Flavio, et al. "Genetic Algorithm with Redundancies for the Vehicle Scheduling Problem." In Biethahn and Nissen [33], 341–353.

22. Baluja, S. *Population-Based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning*. Technical Report CMU-CS-94-163, CMU, 1994.

23. Barr, Richard S., et al. "Designing and Reporting on Computational Experiments with Heuristic Methods," *Journal of Heuristics*, *1*:9–32 (1995).

24. Beasley, J. R., "OR Library." Online, 1999. Available: http://mscmga.ms.ic.ac.uk/info.html.

25. Belegundu, A. D., et al. "Multi-Objective Optimization of Laminated Ceramic Composites Using Genetic Algorithms." *Proceedings of the 5th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization*. 1015–1022. Washington, D.C.: AIAA, 1994.

26. Belegundu, A. D. and P. L. N. Murthy. *A New Genetic Algorithm for Multiobjective Optimization*. Technical Report AIAA-96-4180-CP, Washington, D. C.: AIAA, 1996.

27. Ben-Tal, Aharon. "Characterization of Pareto and Lexicographic Optimal Solutions." *Multiple Criteria Decision Making Theory and Application 177*. Lecture Notes in Economics and Mathematical Systems, edited by G. Fandel and T. Gal, 1–11, Berlin: Springer-Verlag, 1980.

28. Benson, Harold P. and Serpil Sayin. "Towards Finding Global Representations of the Efficient Set in Multiple Objective Mathematical Programming," *Naval Research Logistics*, *44*:47–67 (1997).

29. Bentley, P. J. and J. P. Wakefield. *An Analysis of Multiobjective Optimization within Genetic Algorithms*. Technical Report ENGPJB96, UK: University of Huddersfield, 1996.

30. Bentley, P. J. and J. P. Wakefield. "Overview of a Generic Evolutionary Design System." *Proceedings of the 2nd On-Line World Conference on Evolutionary Computation (WEC2)*. 53–56. 1996.

31. Bentley, P. J. and J. P. Wakefield. "Finding Acceptable Solutions in the Pareto-Optimal Range Using Multiobjective Genetic Algorithms." *Soft Computing in Engineering Design and Manufacturing*, edited by P. K. Chawdhry, et al. 231–240. Springer Verlag, 1997.

32. Bhanu, Bir and Sungkee Lee. *Genetic Learning for Adaptive Image Segmentation*. Boston: Kluwer Academic Publishers, 1994.

33. Biethahn, J. and Volker Nissen, editors. *Evolutionary Algorithms in Management Applications*. Berlin: Springer-Verlag, 1995.

34. Bilchev, G. and I. C. Parmee. "Adaptive Search Strategies for Heavily Constrained Design Spaces." *Proceedings of the 22nd International Conference on Computer Aided Design (CAD '95)*. May 1995.

35. Binh, To Thanh. *A Multiobjective Evolutionary Algorithm: The Study Cases*. Technical Report, Institute for Automation and Communication, Barleben, Germany, January 1999.

36. Binh, To Thanh and Ulrich Korn. "An Evolution Strategy for the Multiobjective Optimization." *Proceedings of the Second International Conference on Genetic Algorithms (Mendel '96)*. 23–28. 1996.

37. Binh, To Thanh and Ulrich Korn. "MOBES: A Multiobjective Evolution Strategy for Constrained Optimization Problems." *Proceedings of the Third International Conference on Genetic Algorithms (Mendel '97)*. 176–182. 1997.

38. Binh, To Thanh and Ulrich Korn. "Multicriteria Control System Design Using an Intelligent Evolution Strategy with Dynamical Constraints Boundaries." *Proceedings of the Control of Industrial Systems Conference (CIS '97)2*. 242–247. 1997.

39. Binh, To Thanh and Ulrich Korn. "Multiobjective Evolution Strategy with Linear and Nonlinear Constraints." *Proc. of the* 15^{th} *IMACS World Congress on Scientific Computation, Modeling and Applied Mathematics*. 357–362. 1997.

40. Binh, To Thanh and Ulrich Korn. "A Parallel Multiobjective Evolutionary Algorithm," *Submission paper for Evolutionary Computing 1999* (1999).

41. Borghi, C. A., et al. "Reduction of the Torque Ripple in permanent Magnet Actuators by a Multiobjective Minimization Technique," *IEEE Transactions on Magnetics*, *34*(5):2869–2872 (September 1998).

42. Brassard, Giles and Paul Bratley. *Algorithmics: Theory and Practice* (First Edition). Englewood Cliffs NJ: Prentice Hall, 1988.

43. Broekmeulen, Rob A. C. M. "Facility Management of Distribution Centers for Vegetables and Fruits." In Biethahn and Nissen [33], 199–210.

44. Cadenas, J. M. and F. Jiménez. "A Genetic Algorithm for the Multiobjective Solid Transportation Problem: A Fuzzy Approach." *Proceedings of the 27th International Symposium on Automotive Technology and Automation*. 327–334. 1994.

45. Camponogara, Eduardo and Sarosh N. Talukdar. "A Genetic Algorithm for Constrained and Multiobjective Optimization." *3rd Nordic Workshop on Genetic Algorithms and Their Applications (3NWGA)*, edited by Jarmo T. Alander. 49–62. Vaasa, Finland: University of Vaasa, August 1997.

46. Cantú-Paz, Erick. *A Survey of Parallel Genetic Algorithms*. Technical Report 97003, University of Illinois at Urbana-Champaign, May 1997.

47. Cedeno, Walter and V. Rao Vemuri. "On the Use of Niching For Dynamic Landscapes." In Porto [259], 361–366.

48. Chang, C. S., et al. "Genetic Algorithm Based Bicriterion Optimization for Traction Substations in DC Railway System." In Fogel [102], 11–16.

49. Chang, Ni-Bin and H. W. Chen. "The Use of Fuzzy Interval Genetic Algorithm for Solving Multiobjective Nonlinear Mixed Integer Programming Model." *Proceedings of the First International Conference on Operations and Quantitative Management*. 76–82. 1997.

50. Chang, T. J., et al. *Heuristics for Cardinality Constrained Portfolio Optimization*. Technical Report, London SW7 2AZ, England: The Management School, Imperial College, May 1998.

51. Chellapilla, Kumar. "Combining Mutation Operators in Evolutionary Programming," *IEEE Transactions on Evolutionary Computation*, *2*(3):91–96 (September 1998).

52. Cheng, Franklin Y. and Dan Li. "Multiobjective Optimization Design with Pareto Genetic Algorithm," *Structural Engineering*, *123*(9):1252–1261 (September 1997).

53. Chipperfield, A. J., et al. "Multiobjective Robust Control Using Evolutionary Algorithms." *Proceedings of the IEEE International Conference on Industrial Technology*. 269–273. Piscataway, NJ: IEEE, 1996.

54. Chipperfield, A. J. and P. J. Fleming. "Gas Turbine Engine Controller Design using Multiobjective Genetic Algorithms." *Proceedings of the First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems : Innovations and Applications, GALESIA'95*, edited by A. M. S. Zalzala. 214–219. Halifax Hall, University of Sheffield, UK: IEEE, September 1995.

55. Chipperfield, A. J. and P. J. Fleming. "Multiobjective Gas Turbine Engine Controller Design Using Genetic Algorithms," *IEEE Transactions on Industrial Electronics*, *43*(5):583–587 (October 1996).

56. Chipperfield, A. J. and P. J. Fleming. "Evolutionary Design of Gas Turbine Aero-Engine Controllers." In DiCesare and Jafari [89], 2401–2406.

57. Chow, C. Rick. "An Evolutionary Approach to Search for NCR-Boards." In Fogel [103], 295–300.

58. Cieniawski, Scott E., et al. "Using Genetic Algorithms to Solve a Multiobjective Groundwater Monitoring Problem," *Water Resources Research*, *31*(2):399–409 (February 1995).

59. Coello, Carlos A., et al. "Use of Genetic Algorithms for Multiobjective Optimization of Counterweight Balancing of Robot Arms." *EXPERSYS-95 Expert Systems Applications and Artificial Intelligence*, edited by Jacob J. G. Chen. 243–248. San Francisco, California: Technology Transfer Series, November 1995.

60. Coello, Carlos A. Coello. "Using the Min-Max Method to Solve Multiobjective Optimization Problems with Genetic Algorithms." *IBERAMIA '98, Lecture Notes in Computer Science*. 303–314. Springer-Verlag, October 1998.

61. Coello, Carlos A. Coello. "A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques," *Knowledge and Information Systems. An International Journal* (1999). (Accepted for publication).

62. Coello, Carlos A. Coello. "Treating Constraints as Objectives for Single-Objective Evolutionary Optimization," *Engineering Optimization*, *32* (1999).

63. Coello, Carlos A. Coello and Alan D. Christiansen. "Two New GA-based methods for multiobjective optimization," *Civil Engineering Systems* (1998). (In Press).

64. Coello, Carlos A. Coello and Alan D. Christiansen. "MOSES : A Multiobjective Optimization Tool for E ngineering Design," *Engineering Optimization*, *31*(3) (1999). (Accepted for publication).

65. Coello, Carlos A. Coello, et al. "Multiobjective Design Optimization of Counterweight Balancing of a Robot Arm Using Genetic Algorithms." *Seventh International Conference on Tools with Artificial Intelligence*. 20–23. IEEE, 1995.

66. Coello, Carlos A. Coello, et al. "Using a New GA-Based Multiobjective Optimization Technique for the Design of Robot Arms," *Robotica*, *16*:401–414 (1998).

67. Coello, Carlos Artemio Coello. *An Empirical Study of Evolutionary Techniques for Multiobjective Optimization in Engineering Design*. PhD dissertation, Department of Computer Science, Tulane University, New Orleans, LA, April 1996.

68. Coello, Carlos Coello, "List of References on Evolutionary Multiobjective Optimization." Online, 1999. Available: http://www.lania.mx/EMOO

69. Cohon, Jared L. *Multiobjective Programming and Planning*. New York: Academic Press, 1978.

70. Cohon, Jared L. and David H. Marks. "A Review and Evaluation of Multiobjective Programming Techniques," *Water Resources Research*, *11*(2):208–220 (1975).

71. Corno, F., et al. "A new Evolutionary Algorithm Inspired by the Selfish Gene Theory." In Fogel [103].

72. Corno, F., et al. "The Selfish Gene Algorithm: a new Evolutionary Optimization Strategy." *Proceedings of the 13th Annual ACM Symposium on Applied Computing*. 349–355. February 1998.

73. Crossley, William A. "Genetic Algorithm Approaches for Multiobjective Design of Rotor Systems." *Proceedings of the 6th AIAA/NASA ISSMO Symposium on Multidisciplinary Analysis and Optimization*. 384–394. AIAA, 1996.

74. Crossley, William A. *A Genetic Algorithm with the Kreisselmeier-Steinhauser Function for Multiobjective Constrained Optimization of Rotor Systems*. Technical Report AIAA-97-0080, 1801 Alexander Bell Drive, Suite 500, Reston, VA 22091: AIAA, January 1997.

75. Crossley, William A. "A Genetic Algorithm with the Kreisselmeier-Steinhauser Function for Multiobjective Constrained Optimization of Rotor Systems." *35th Aerospace Sciences Meeting and Exhibit*. Number AIAA-97-0080 in AIAA. January 1997.

76. Crossley, William A., et al. *Using the Two-Branch Tournament Genetic Algorithm for Multiobjective Design*. Technical Report AIAA-98-1914, AIAA, 1998.

77. Cunha, A. Gaspar, et al. "Use of Genetic Algorithms in Multicriteria Optimization to Solve Industrial Problems." In Bäck [18], 682–688.

78. Cvetković, Dragan, et al. "Multi-objective Optimization and Preliminary Airframe Design." *The Integration of Evolutionary and Adaptive Computing Technologies with Product/System Design and Realization*, edited by Ian Parmee. 255–267. Plymouth, United Kingdom: Springer-Verlag, April 1998.

79. Cvetković, Dragan and Ian C. Parmee. "Evolutionary Design and Multi–objective Optimization." *6th European Congress on Intelligent Techniques and Soft Computing EUFIT'98*. 397–401. September 1998.

80. De Jong, Kenneth A. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD dissertation, The University of Michigan, Ann Arbor MI, 1975.

81. de Neufville, Richard. *Applied Systems Analysis: Engineering Planning and Technology Management*. New York, NY: McGraw-Hill, 1990.

82. Deb, Kalyanmoy. *Binary and Floating-Point Function Optimization Using Messy Genetic Algorithms*. PhD dissertation, University of Alabama, Tuscaloosa, AL, 1991.

83. Deb, Kalyanmoy. *Multiobjective Genetic Algorithms: Problem Difficulties and Construction of Test Problems*. Technical Report TR CI-49/98, University of Dortmund, Germany: Department of Computer Science/XI, 1998.

84. Deb, Kalyanmoy. "Evolutionary Algorithms for Multi-Criterion Optimization in Engineering Design." *Proceedings of Evolutionary Algorithms in Engineering and Computer Science (EUROGEN-99)*. 1999.

85. Deb, Kalyanmoy. *Non-linear Goal Programming Using Multi-Objective Genetic Algorithms*. Technical Report TR CI-60/98, University of Dortmund, Germany: Department of Computer Science/XI, 1999.

86. Deb, Kalyanmoy and David E. Goldberg. "An Investigation of Niche and Species Formation in Genetic Function Optimization." *Proceedings of the Third International Conference on Genetic Algorithms*, edited by J. David Schaffer. 42–50. San Mateo, California: Morgan-Kaufmann Publishers, Inc., June 1989.

87. Deb, Kalyanmoy and Amarendra Kumar. "Real-coded Genetic Algorithms with Simulated Binary Crossover: Studies on Multimodal and Multiobjective Problems," *Complex Systems*, *9*:431–454 (1995).

88. Deerman, Karl R. *Protein Structure Prediction Using Parallel Linkage Investigating Genetic Algorithms*. MS Thesis, AFIT/GCS/ENG/99M-03, Air Force Institute of Technology, Wright-Patterson AFB, 1999.

89. DiCesare, Frank and Mohsen Jafari, editors. *Proceedings of the 1998 IEEE International Conference on Systems, Man, and Cybernetics*, IEEE, 1998.

90. Dick, Robert P. and Niraj K. Jha. "MOGAC: A Multiobjective Genetic Algorithm for the Co-Synthesis of Hardware-Software Embedded Systems." *Proceedings of the IEEE/ACM Conference on Computer Aided Design*. 522–529. Los Alamitos, CA: IEEE Computer Society Press, 1997.

91. Dimopoulos, C. and A.M.S. Zalzala. "Optimization of Cell Configuration and Comparisons Using Evolutionary Computation Approaches." In Fogel [103], 148–153.

92. Dozier, Gerry, et al. "Multiobjective Evolutionary Path Planning via Fuzzy Tournament Selection." In Fogel [103], 684–689.

93. Duckstein, L. "Multiobjective Optimization in Structural Design: The Model Choice Problem." *New Directions in Optimum Structural Design* edited by E. Atrek, et al., chapter 22, 459–481, John Wiley & Sons Ltd., 1984.

94. Duda, Richard O. and Peter E. Hart. *Pattern Classification and Scene Analysis*. New York, NY: John Wiley & Sons, 1973.

95. Dymek, Andrew. *Examination of Hypercube Implementations of Genetic Algorithms*. MS Thesis, AFIT/GCS/ENG/92M-02, EN, WPAFB, December 1992.

96. Eiben, A. E., et al., editors. *Parallel Problem Solving from Nature - PPSN V*, Berlin: Springer, 1998.

97. El-Rewini, Hesham, et al. *Task Scheduling in Parallel and Distributed Systems*. Prentice Hall, 1994.

98. Ely, T. A., et al. "Satellite Constellation Design for Zonal Coverage Using Genetic Algorithms." *8th AAS/AIAA Space Flight Mechanics Meeting*. February 1998.

99. Esbensen, Henrik and Ernest Kuh. "EXPLORER: An Interactive Floorplanner for Design Space Exploration." *Proceedings of the European Design Automation Conference*. 356–361. 1996.

100. Eshelman, Larry J., editor. *Proceedings of the Sixth International Conference on Genetic Algorithms*, San Mateo CA: Morgan Kaufmann Publishers, Inc., 1995.

101. Flynn, Robert and Porter D. Sherman. “Multicriteria Optimization of Aircraft Panels: Determining Viable Genetic Algorithm Configurations,” *International Journal of Intelligent Systems*, *10*:987–999 (1995).

102. Fogel, David, editor. *Proceedings of the Second IEEE Conference on Evolutionary Computation*, Piscataway NJ: IEEE Service Center, 1995.

103. Fogel, David, editor. *Proceedings of the 1998 (5th) IEEE International Conference on Evolutionary Computation*, Piscataway, NJ: IEEE, May 1998.

104. Fogel, David B. *Handbook of Evolutionary Computation*, chapter Introduction, A.1.1:1–A.1.1:2. Volume 1 of Bäck et al. [19], 1997.

105. Fogel, David B. and Adam Ghozeil. “A Note on Representations and Variation Operators,” *IEEE Transactions on Evolutionary Computation*, *1*(2):159–161 (July 1997).

106. Fogel, L. J., et al. *Artificial Intelligence Through Simulated Evolution*. New York: Wiley Publishing, 1966.

107. Fonseca, Carlos M. and P. J. Fleming. *Handbook of Evolutionary Computation*, chapter Multiobjective Optimization, C4.5:1 – C4.5:9. Volume 1 of Bäck et al. [19], 1997.

108. Fonseca, Carlos M. and Peter J. Fleming. “Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion, and Generalization.” In Forrest [116], 416–423.

109. Fonseca, Carlos M. and Peter J. Fleming. “Multiobjective Genetic Algorithms Made Easy: Selection, Sharing, and Mating Restriction.” *Proceedings of the 1st International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*. 45–52. September: IEE, 1995.

110. Fonseca, Carlos M. and Peter J. Fleming. *Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms I: A Unified Formulation*. Technical Report 564, Sheffield, UK: University of Sheffield, January 1995.

111. Fonseca, Carlos M. and Peter J. Fleming. “An Overview of Evolutionary Algorithms in Multiobjective Optimization,” *Evolutionary Computation*, *3*(1):1–16 (Spring 1995).

112. Fonseca, Carlos M. and Peter J. Fleming. “Non-Linear System Identification with Multiobjective Genetic Algorithms.” *Proceedings of the 13th Triennial World Congress of the International Federation of Automatic Control*. 187–192. Pergamon, 1997.

113. Fonseca, Carlos M. and Peter J. Fleming. “Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms – Part II: Application Example,” *IEEE Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans*, *28*(1):38–47 (January 1998).

114. Fonseca, Carlos M. and Peter J. Fleming. “Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms – Part I: A Unified Formula-

tion," *IEEE Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans*, *28*(1):26–37 (January 1998).

115. Fonseca, Carlos Manuel Mira. *Multiobjective Genetic Algorithms with Application to Control Engineering Problems*. PhD dissertation, The University of Sheffield, Sheffield, UK, 1995.

116. Forrest, Stephanie, editor. *Proceedings of the Fifth International Conference on Genetic Algorithms*, San Mateo CA: Morgan Kaufmann Publishers, Inc., July 1993.

117. Fourman, Michael P. "Compaction of Symbolic Layout Using Genetic Algorithms." In Grefenstette [134], 141–153.

118. Fujita, K., et al. "Multi-Objective Optimal Design of Automotive Engine Using Genetic Algorithm." *Proceedings of DETC'98 – ASME Design Engineering Technical Conferences*. 11–21. 1998.

119. Fuller, J. F. C. *A Military History of the Western World, Volume I: From the Earliest Times to the Battle of Lepanto*. New York, NY: Da Capo Press, 1954.

120. Garey, M.R. and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, 1979.

121. Gates, Jr., George H. *Predicting Protein Structure Using Parallel Genetic Algorithms*. MS Thesis, AFIT/GCS/ENG/94D-03, EN, WPAFB, December 1994.

122. Gen, M., et al. "Solving Multiobjective Solid Transportation Problem by Genetic Algorithm," *Journal of Japan Industrial Management Association*, *46*(5):446–454 (1995). (in Japanese).

123. Gen, Mitsuo, et al. "A Spanning Tree-Based Genetic Algorithm for Bicriteria Topological Network Design." In Fogel [103], 15–20.

124. Gero, J. S. and S. Louis. "Improving Pareto Optimal Designs Using Genetic Algorithms," *Microcomputers in Civil Engineering*, *10*(4):241–249 (1995).

125. Gero, John S., et al. "Evolutionary Learning of Novel Grammars for Design Improvement," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, *8*:83–94 (1994).

126. Goldberg, David E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison Wesley, 1989.

127. Goldberg, David E. *From Genetic and Evolutionary Optimization to the Design of Conceptual Machines*. Technical Report 98008, University of Illinois at Urbana-Champaign, May 1998.

128. Goldberg, David E., et al. "Rapid, Accurate Optimization of Difficult Problems Using Fast Messy Genetic Algorithms." *Proceedings of the Fifth International Conference on Genetic Algorithms*, edited by Stephanie Forrest. 56–64. San Mateo, CA: Morgan Kaufmann Publishers, July 1993.

129. Goldberg, David E., et al. "Messy Genetic Algorithms Revisited: Studies in Mixed Size and Scale," *Complex Systems*, *4*:415–444 (1990).

130. Goldberg, David E., et al. "Messy Genetic Algorithms: Motivation, Analysis, and First Results," *Complex Systems*, *3*:493–530 (1989).

131. Goldberg, David E. and Jon Richardson. "Genetic Algorithms with Sharing for Multimodal Function Optimization." *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, edited by John J. Grefenstette. 41–49. Hillsdale, New Jersey: Lawrence Erlbaum Associates, Publishers, July 1987.

132. Gómez-Skarmeta, Antonio F., et al. "Pareto Optimality in Fuzzy Modeling." *6th European Congress on Intelligent Techniques and Soft Computing EUFIT'98*. 694–700. September 1998.

133. Greenwood, Garrison W., et al. "Fitness Functions for Multiple Objective Optimization Problems: Combining Preferences with Pareto Rankings." *Foundations of Genetic Algorithms 4*, edited by Richard K. Belew and Michael D. Vose. 305–323. San Mateo, CA: Morgan Kaufmann, 1997.

134. Grefenstette, John J., editor. *Proceedings of the First International Conference on Genetic Algorithms and Their Applications*, Hillsdale, New Jersey: Lawrence Erlbaum Associates, Publishers, July 1985.

135. Grefenstette, John J. *A User's Guide to Genesis 5.0*. Technical Report, Nashville, TN: Vanderbuilt University, 1990.

136. Grignon, Pierre M., et al. *Bi-Objective Optimization of Components Packing Using a Genetic Algorithm*. Technical Report AIAA-96-4022-CP, Washington, D.C.: AIAA, 1996.

137. Groppetti, R. and R. Muscia. "On a Genetic Multiobjective Approach for the Integration and Optimization of Assembly Product Design and Process Planning." *Integrated Design and Manufacturing in Mechanical Engineering* edited by P. Chedmail, et al., 61–70, The Netherlands: Kluwer Academic Publishers, 1997.

138. Guillen, Michael. *Bridges to Infinity*. Los Angeles: Jeremy P. Tarcher, Inc., 1983.

139. Hajela, P. and J. Lee. "Constrained Genetic Search via Schema Adaptation: An Immune Network Solution," *Structural Optimization*, *12*:11–15 (1996).

140. Hajela, P. and C. Lin. "Genetic Search Strategies in Multicriterion Optimal Design," *Structural Optimization*, *4*:99–107 (1992).

141. Hajela, P., et al. "GA Based Simulation of Immune Networks Applications in Structural Optimization," *Engineering Optimization*, *29*:131–149 (1997).

142. Halhal, D., et al. "Multi-objective Improvement of Water Distribution Systems Using a Structured Messy Genetic Algorithm Approach," *Journal of Water Resources Planning and Management ASCE*, *123*:137–146 (1997).

143. Hammack, Lonnie P. *Parallel Data Mining with the Message Passing Interface Standard on Clusters of Personal Computers*. MS Thesis, AFIT/GCS/ENG/99M-06, Air Force Institute of Technology, Wright-Patterson AFB, March 1999.

144. Harik, Georges. *Linkage Learning via Probabilistic Modeling in the ECGA*. Technical Report IlliGAL TR 99010, Illinois Genetic Algorithms Laboratory, 1999.

145. Harik, Georges R. *Learning Gene Linkage to Efficiently Solve Problems of Bounded Difficulty Using Genetic Algorithms*. PhD dissertation, University of Michigan, 1997.

146. Harik, Georges R., et al. "The Compact Genetic Algorithm." In Fogel [103], 523–528.

147. Harris, Stephen P. and Emmanuel C. Ifeachor. "Nonlinear FIR Filter Design by Genetic Algorithm." *1st Online Conference on Soft Computing*. August 1996.

148. Higashishara, Tomoyouki and Masayasu Atsumi. "Evolutionary Acquisition of Sensory—Action Network of Mobile Robot using Multiobjective Genetic Algorithm," *IPSJ SIG-ICS*, *98-ICS-111*:1–6 (1998).

149. Hilliard, Michael R., et al. "The Computer as a Partner in Algorithmic Design: Automated Discovery of Parameters for a Multi-Objective Scheduling Heuristic." *Impacts of Recent Computer Advances on Operations Research* edited by Ramesh Sharda, et al., 321–331, Elsevier Science Publishing Co., Inc., 1989.

150. Hillier, Frederick S. and Gerald J. Lieberman. *Introduction to Operations Research*. San Francisco: Holden-Day, Inc., 1967.

151. Hinchliffe, Mark, et al. "Chemical Process Systems Modeling Using Multi-objective Genetic Programming." *Proceedings of the Third Annual Genetic Programming Conference*, edited by John R. Koza, et al. 134–139. San Francisco, CA: Morgan Kaufmann Publishers, 1998.

152. Horn, Jeffrey. *Handbook of Evolutionary Computation*, chapter Multicriterion Decision Making, F1.9:1 – F1.9:15. Volume 1 of Bäck et al. [19], 1997.

153. Horn, Jeffrey. *The Nature of Niching: Genetic Algorithms and the Evolution of Optimal, Cooperative Populations*. PhD dissertation, University of Illinois at Urbana Champaign, Urbana, Illinois, 1997.

154. Horn, Jeffrey and Nicholas Nafpliotis. *Multiobjective Optimization Using the Niched Pareto Genetic Algorithm*. Technical Report 930005, University of Illinois at Urbana-Champaign, 117 Transportation Building, 104 South Matthews Avenue, Urbana, IL 61801-2996: Illinois Genetic Algorithms Laboratory (IlliGAL), July 1993.

155. Horn, Jeffrey, et al. "A Niched Pareto Genetic Algorithm for Multiobjective Optimization." *Proceedings of the First IEEE Conference on Evolutionary Computation*, edited by Zbigniew Michalewicz. 82–87. Piscataway NJ: IEEE Service Center, June 1994.

156. Hu, Xiabo (Sharon), et al. "An Evolutionary Approach to Hardware/Software Partitioning." In Voigt et al. [337], 900–909.

157. Husbands, Philip. "Genetic Algorithms in Optimization and Adaptation." *Advances in Parallel Algorithms* edited by Lydia Kronsjö and Dean Shumsherudin, chapter 8, 227–276, New York: Halsted Press, 1992.

158. Hwang, Ching-Lai and Abu Syed Md. Masud. *Multiple Objective Decision Making - Methods and Applications*. Springer Verlag, 1979.

159. Iba, Hitoshi, et al. "Genetic Programming Using a Minimum Description Length Principle." *Advances in Genetic Programming* edited by Angeline and Kinnear, 265–284, MIT PRess, 1994.

160. Ignizio, James P. "Integrating Cost, Effectiveness, and Stability," *Acquisition Review Quarterly*, 51–60 (Winter 1998).

161. Ikonen, Ilkka, et al. "A Genetic Algorithm for Packing Three-Dimensional Non-Convex Objects Having Cavities and Holes." In Bäck [18], 591–598.

162. IlliGAL, "IlliGAL Home Page." Online, 1999. Available: http://gal4.ge.uiuc.edu/illigal.home.html.

163. Ishibuchi, Hisao and Tadahiko Murata. "Multi-Objective Genetic Local Search Algorithm." In Bäck, Thomas and Zbigniew Michalewicz and Hiroaki Kitano [20], 119–124.

164. Ishibuchi, Hisao and Tadahiko Murata. "Multi-Objective Genetic Local Search Algorithm and Its Application to Flowshop Scheduling," *IEEE Transactions on Systems, Man, and Cybernetics*, *28*(3):392–403 (August 1998).

165. Ishibuchi, Hisao and Tadahiko Murata. "Multi-Objective Genetic Local Search for Minimizing the Number of Fuzzy Rules for Pattern Classification Problems." In Fogel [103], 1100–1105.

166. Jackson, Richard H. F., et al. "Guidelines for Reporting Results of Computational Experiments. Report of the Ad Hoc Committee," *Mathematical Programming*, *49*:413–425 (1991).

167. Jakob, Wilfried, et al. "Application of Genetic Algorithms to Task Planning and Learning." *Parallel Problem Solving from Nature, 2*, edited by R. Männer and B. Manderick. 291–300. Elsevier Science Publishers B. V., 1992.

168. Jiménez, Fernando and José L. Verdegay. "Interval Multiobjective Solid Transportation Problem via Genetic Algorithms." *Proceedings of the Sixth International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU '96)*. 787–792. 1996.

169. Jiménez, Fernando and José L. Verdegay. "Constrained Multiobjective Optimization by Evolutionary Algorithms." *Proceedings of the International ICSC Symposium on Engineering of Intelligent Systems (EIS'98)*. 1998.

170. Jones, Brian R., et al. "Aerodynamic and Aeroacoustic Optimization of Airfoils Via a Parallel Genetic Algorithm." *Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis & Optimization*. 1998. AIAA-98-4811.

171. Jones, Gareth, et al. "Searching Databases of Two-Dimensional and Three-Dimensional Chemical Structures Using Genetic Algorithms." In Forrest [116], 597–602.

172. Kargupta, Hillol. "The Gene Expression Messy Genetic Algorithm." In Bäck, Thomas and Zbigniew Michalewicz and Hiroaki Kitano [20], 631–636.

173. Kato, Kosuke and Masatoshi Sakawa. "Interactive Decision Making for Multiobjective Block Angular 0-1 Programming Problems with Fuzzy Parameters Through Genetic Algorithms with Decomposition Procedures." *Proceedings of the Sixth IEEE Conference on Fuzzy Systems*. 1645–1650. 1997.

174. Kato, Kosuke and Masatoshi Sakawa. "An Interactive Fuzzy Satisficing Method for Large Scale Multiobjective 0-1 Programming Problems with Fuzzy Parameters Through Genetic Algorithms," *European Journal of Operational Research*, *107*:590–598 (1998).

175. Kato, Kosuke, et al. "Interactive Decision Making for Multiobjective Block Angular 0-1 Programming Problems with Fuzzy Parameters Through Genetic Algorithms," *Japanese Journal of Fuzzy Theory and Systems*, *9*(1):49–59 (1997).

176. Kato, Kosuke, et al. "An Interactive Fuzzy Criteria Method for Multiobjective 0-1 Programming Problems with Block Angular Structure Using Genetic Algorithms," *Electronics and Communications in Japan*, *81*(8):10–17 (1998).

177. Keeney, Ralph L. and Howard Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. New York, NY: John Wiley & Sons, 1976.

178. Khatib, Wael and Peter J. Fleming. "An Introduction to Evolutionary Computing for Multidisciplinary Optimization." *Genetic Algorithms in Engineering Systems: Innovations and Applications*. 7–12. 1997.

179. Kim, Min-Kyu, et al. "Multiobjective Optimal Design of Three-Phase Induction Motor Using Improved Evolution Strategy," *IEEE Transactions on Magnetics*, *34*(5):2980–2983 (September 1998).

180. Kinnear, Kenneth E., et al. *Handbook of Evolutionary Computation*, chapter Derivative Methods, B1.5:1 – B1.5:15. Volume 1 of Bäck et al. [19], 1997.

181. Kita, Hajime, et al. "Multi-Objective Optimization by Means of the Thermodynamical Genetic Algorithm." In Voigt et al. [337], 504–512.

182. Krause, Matthias and Volker Nissen. "On Using Penalty Functions and Multicriteria Optimization Techniques in Facility Layout." In Biethahn and Nissen [33], 153–166.

183. Kumar, Rajeev and Peter Rockett. "Assessing the Convergence of Rank-Based Multiobjective Genetic Algorithms." *Proceedings of the 2nd International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*. 19–23. IEE, September 1997.

184. Kumar, Rajeev and Peter Rockett. "Multiobjective Genetic Algorithm Partitioning for Hierarchical Learning of High-Dimensional Pattern Spaces: A Learning-Follows-Decomposition Strategy," *IEEE Transactions on Neural Networks*, *9*(5):822–830 (1998).

185. Kumar, Vipin, et al. *Introduction to Parallel Computing: Design and Analysis of Algorithms*. Redwood City, CA: The Benjamin/Cummings Publishing Company, Inc., 1994.

186. Kundu, S. "A Multicriteria Genetic Algorithm to Solve Optimization Problems in Structural Engineering Design." *Information Processing in Civil and Structural Engineering Design* edited by B. Kumar, 225–233, Civil-Comp Press, 1996.

187. Kundu, Sourav and Seichi Kawata. "AI in Control System Design Using a New Paradigm for Design Representation." *Artificial Intelligence in Design*, edited by J. S. Gero and F. Sudweeks. 135–150. The Netherlands: Kluwer Academic Publishers, 1996.

188. Kundu, Sourav, et al. "A Multicriteria Approach to Control System Design with Genetic Algorithms." *Proceedings of the 13th Triennial World Congress of the International Federation of Automatic Control*. 315–320. Pergamon, 1996.

189. Kursawe, Frank. "A Variant of Evolution Strategies for Vector Optimization." *Parallel Problem Solving from Nature, 1496*. Lecture Notes in Computer Science, edited by H.-P. Schwefel and R. Männer. 193–197. Berlin: Springer-Verlag, October 1990.

190. Lamont, Gary B., editor. *Compendium of Parallel Programs for the Intel iPSC Computers*. Wright-Patterson AFB, OH 45433: Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology, 1993.

191. Langdon, William B. "Evolving Data Structures with Genetic Programming." In Eshelman [100], 295–302.

192. Langdon, William B. *Pareto, Population Partitioning, Price and Genetic Programming*. Research Note RN/95/29, Gower Street, London WC1E 6BT, UK: University College, London, April 1995.

193. Langdon, William B. "Data Structures and Genetic Programming." *Advances in Genetic Programming 2* edited by Peter J. Angeline and Kenneth E. Kinnear, Jr., chapter 20, 395–414, Cambridge, MA, USA: MIT Press, 1996.

194. Langdon, William B. "Using Data Structures within Genetic Programming." *Genetic Programming 1996: Proceedings of the First Annual Conference*, edited by John R. Koza, et al. 141–148. Stanford University, CA, USA: MIT Press, 28–31 July 1996.

195. Laumanns, Marco, et al. "A Spatial Predator-Prey Approach to Multi-Objective Optimization." In Eiben et al. [96].

196. Lee, Jongsoo and Prabhat Hajela. "Parallel Genetic Algorithm Implementation in Multidisciplinary Rotor Blade Design," *Journal of Aircraft*, *33*(5):962–969 (September-October 1996).

197. Li, Yinzhen, et al. "Evolutionary Computation for Multicriteria Solid Transportation Problem with Fuzzy Numbers." In Bäck, Thomas and Zbigniew Michalewicz and Hiroaki Kitano [20], 596–601.

198. Li, Yinzhen, et al. "Improved Genetic Algorithm for Solving Multiobjective Solid Transportation Problem with Fuzzy Numbers," *Computers in Industrial Engineering*, *33*(3-4):589–592 (1997).

199. Liang, Simon J. and John M. Lewis. "Job Shop Scheduling Using Multiple Criteria." *Proceedings of the Joint Hungarian-British Mechatronic Conference*. 77–82. Computational Mechanics, September 1994.

200. Liepins, G. E., et al. *Operations Research and Artificial Intelligence: The Integration of Problem-Solving Strategies*, chapter Genetic Algorithms Applications to Set Covering and Traveling Salesman Problems, 29–57. Boston, MA: Kluwer Academic Publishers, 1990.

201. Lis, Joanna and A. E. Eiben. "A Multi-Sexual Genetic Algorithm for Multiobjective Optimization." In Porto [259], 59–64.

202. Liu, Tung Kuan, et al. "Multiobjective Control Systems Design by Genetic Algorithms." *Proceedings of the 34th Society of Instrument and Control Engineers Annual Conference*. 1521–1526. 1995.

203. Liu, Xiaojian, et al. "Genetic Approach to Optimal Topology/Controller Design of Adaptive Structures," *International Journal for Numerical Methods in Engineering*, *41*:815–830 (1998).

204. Loughlin, D. H. *Genetic Algorithm-Based Optimization in the Development of Tropospheric Ozone Control Strategies: Least Cost, Multiobjective, Alternative Generation, and Chance-Constrained Applications*. PhD dissertation, North Carolina State University, Raleigh, NC, 1998.

205. Loughlin, Daniel H. and S. Ranjithan. "The Neighborhood Constraint Method: A Genetic Algorithm-Based Multiobjective Optimization Technique." In Bäck [18], 666–673.

206. Louis, Sushil J. *Genetic Algorithms as a Computational Tool for Design*. PhD dissertation, Department of Computer Science, Indiana University, August 1993.

207. Louis, Sushil J. and Gregory J. E. Rawlins. "Pareto Optimality, GA-Easiness, and Deception." In Forrest [116], 118–123.

208. Mahfouf, M., et al. "Multi-Objective Genetic Optimization of the Performance Index of Self-Organizing Fuzzy Logic Control Algorithm Using a Fuzzy Ranking Approach." *Proceedings of the Sixth European Congress on Intelligent Techniques and Soft Computing*, edited by H. J. Zimmerman. 1799–1808. Aachen: Verlag Mainz, 1998.

209. Mäkinen, R., et al. "A Genetic Algorithm for Multiobjective Design Optimization in Aerodynamics and Electromagnetics." *Computational Fluid Dynamics '98, Proceedings of the ECCOMAS 98 Conference2*, edited by K. D. Papailiou et al. 418–422. Athens, Greece: Wiley, September 1998.

210. Mäkinen, R. A. E., et al. "Parallel Genetic Solution for Multiobjective MDO." *Parallel Computational Fluid Dynamics: Algorithms and Results Using Advanced Computers* edited by P. Schiano, et al., 352–359, Elsevier Science, 1997.

211. Manna, Zohar. *Mathematical Theory of Computation*. New York, NY: McGraw Hill, 1974.

212. Marmelstein, Robert E., et al. "A Distributed Architecture for an Adaptive Computer Virus Immune System." *Proceedings of the 1998 IEEE International Conference on Systems, Man, and Cybernetics*. 3838–3843. 1998.

213. McClave, James T., et al. *Statistics for Business and Economics* (7th Edition). Upper Saddle River, NJ: Prentice Hall, 1998.

214. Merkle, Laurence D. *Analysis of Linkage-Friendly Genetic Algorithms*. PhD Thesis, AFIT/DS/ENG/96-11, Air Force Institute of Technology, Wright-Patterson AFB, 1996.

215. Merkle, Laurence D., et al. "Application of the Parallel Fast Messy Genetic Algorithm to the Protein Folding Problem." *Proceedings of the Intel Supercomputer Users' Group 1994 Annual North America Users Conference*. 189–195. Beaverton Oregon: Intel Supercomputer Systems Division, 1994.

216. Merkle, Laurence D. and Gary B. Lamont. "A Random Function Based Framework for Evolutionary Algorithms." *Proceedings of the 7th International Conference on Genetic Algorithms*. 105–112. Morgan Kauffman, 1997.

217. Michalewicz, Z. and G. Nazhiyath. "Genocop III: A Co-Evolutionary Algorithm for Numerical Optimization Problems with Nonlinear Constraints." In Fogel [102], 647–651.

218. Michalewicz, Zbigniew. *Genetic Algorithms + Data Structures = Evolution Programs* (2nd Edition). New York: Springer-Verlag, 1994.

219. Michalewicz, Zbigniew. "Genetic Algorithms, Numerical Optimization, and Constraints." In Eshelman [100], 151–158.

220. Michielssen, Eric, et al. "Design of Lightweight, Broad-Band Microwave Absorbers Using Genetic Algorithms," *IEEE Transactions on Microwave Theory and Techniques*, *41*(6/7):1024–1031 (1993).

221. Michielssen, Eric and Daniel S. Weile. "Electromagnetic System Design Using Genetic Algorithms." In Winter et al. [345], 345–369.

222. Miller, George A. "The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information," *The Psychological Review*, *63*(2):81–97 (1956).

223. Mitchell, Melanie. *An Introduction to Genetic Algorithms*. Cambridge, MA: The MIT Press, 1996.

224. Mitra, K., et al. "Multiobjective Dynamic Optimization of an Industrial Nylon 6 Semibatch Reactor Using Genetic Algorithm," *Journal of Applied Polymer Science*, *69*(1):69–87 (1998).

225. Mohammed, O. A. and G. F. Üler. "Genetic Algorithms for the Optimal Design of Electromagnetic Devices." *Conference on the Annual Review of Progress in Applied Computational Electromagnetics 11*. 386–393. 1995.

226. Montana, David, et al. "Genetic Algorithms for Complex, Real-Time Scheduling." In DiCesare and Jafari [89], 2213–2218.

227. Moon, Chiung, et al. "Evolutionary Algorithm for Flexible Process Sequencing with Multiple Objectives." In Fogel [103], 27–32.

228. Mori, Naoki, et al. "Thermodynamical Selection Rule for the Genetic Algorithm." In Fogel [102], 188–192.

229. Mullei, Silla and Peter Beling. "Hybrid Evolutionary Algorithms for a Multiobjective Financial Problem." In DiCesare and Jafari [89], 3925–3930.

230. Murata, Tadahiko and Hisao Ishibuchi. "MOGA: Multi-Objective Genetic Algorithms." In Fogel [102], 289–294.

231. Murata, Tadahiko, et al. "Multi-Objective Genetic Algorithm and Its Application to Flowshop Scheduling," *Computers and Industrial Engineering Journal*, *30*(4):957–968 (September 1996).

232. Nam, Dongkyung, et al. "Parameter Optimization of a Voltage Reference Circuit Using EP." In Fogel [103], 301–305.

233. Neapolitan, Richard and Kumarss Naimipour. *Foundations of Algorithms*. Lexington, MA: D. C. Heath and Company, 1996.

234. Neter, John, et al. *Applied Linear Statistical Models: Regression, Analysis of Variance, and Experimental Designs* (3rd Edition). Boston, MA: Irwin, 1990.

235. Nievergelt, Jürg, et al. "All the Needles in a Haystack: Can Exhaustive Search Overcome Combinatorial Chaos?." *Computer Science Today: Lecture Notes in Computer Science 1000*, edited by J. van Leeuwen. 254–274. Berlin: Springer, 1995.

236. Nissen, Volker. *Handbook of Evolutionary Computation*, chapter Management Applications and Other Classical Optimization Problems, F1.2:1 – F1.2:50. Volume 1 of Bäck et al. [19], 1997.

237. Norris, Stephen R. and William A. Crossley. *Pareto-Optimal Controller Gains Generated by a Genetic Algorithm*. Technical Report AIAA-98-1010, Washington, D. C.: AIAA, 1998.

238. Obayashi, Shigeru. "Pareto Genetic Algorithm for Aerodynamic Design Using the Navier-Stokes Equations." In Quagliarella et al. [261], 245–266.

239. Obayashi, Shigeru. "Multidisciplinary Design Optimization of Aircraft Wing Planform Based on Evolutionary Algorithms." In DiCesare and Jafari [89].

240. Obayashi, Shigeru, et al. "Transonic Wing Design by Inverse Optimization Using MOGA." *Proceedings of the Sixth Annual Conference of the Computational Fluid Dynamics Society of Canada2*. 41–46. 1998.

241. Obayashi, Shigeru, et al. "Niching and Elitist Models for MOGAs." In Eiben et al. [96].

242. Obayashi, Shigeru, et al. "Cascade Airfoil Design by Multiobjective Genetic Algorithms." *Proceedings of the 2nd International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*. 24–29. IEE, September 1997.

243. Osman, Keith A., et al. "Improving the Efficiency of Vehicle Water-Pump Designs Using Genetic Algorithms." *Smart Engineering Systems: Proceedings of the Artificial Neural Networks in Engineering Conference (ANNIE '98) 8*, edited by C. Dagli, et al. New York: ASME Press, 1998.

244. Osyczka, A. and S. Kundu. "A Genetic Algorithm Based Multicriteria Optimization Method." *Proceedings of the First World Congress of Structural and Multidisciplinary Optimization*. 909–914. Pergamon, 1995.

245. Osyczka, A. and S. Kundu. "A New Method to Solve Generalized Multicriteria Optimization Problems Using the Simple Genetic Algorithm," *Structural Optimization*, *10*:94–99 (1995).

246. Oyama, A., et al. "Coding by Taguchi Method for Evolutionary Algorithms Applied to Aerodynamic Optimization." *Proceedings of the Fourth ECCOMAS Computational Fluid Dynamics Conference*. 196–203. Athens, Greece: John Wiley & Sons, September 1998.

247. Pacheco, Peter. *Parallel Programming with MPI*. Morgan Kauffman, 1996.

248. Parks, G. T. "Multiobjective Pressurized Water Reactor Reload Core Design by Nondominated Genetic Algorithm Search," *Nuclear Science and Engineering*, *124*(1):178–187 (1996).

249. Parks, G. T. and I. Miller. "Selective Breeding in a Multiobjective Genetic Algorithm." In Eiben et al. [96].

250. Parks, Geoffrey T. "Multiobjective PWR Reload Core Optimization Using a Genetic Algorithms." *Proceedings of the International Conference on Mathematics and Computations, Reactor Physics, and Environmental Analyses*. 615–624. 1995.

251. Parmee, I. C. and G. Purchase. "The Development of a Directed Genetic Search Technique for Heavily Constrained Design Spaces." *Proceedings of Adaptive Computing in Engineering Design and Control*. September 1994.

252. Pearl, Judea. *Heuristics*. Reading MA: Addison–Wesley Publishing Company, 1989.

253. Périaux, Jacques, et al. "RCS Multi-Objective Optimization of Scattered Waves by Active Control Elements Using GAs." *Proceedings of the 4th International Conference on Control, Automation, Robotics, and Vision*. 1996.

254. Périaux, Jacques, et al. "GA Multiple Objective Optimization Strategies for Electromagnetic Backscattering." In Quagliarella et al. [261], 225–243.

255. Pohlheim, Hartmut. *Genetic and Evolutionary Algorithm Toolbox for use with MATLAB*. Technical Report, Technical University Ilmenau, 1998.

256. Poloni, Carlo. "Genetic Algorithms in Engineering and Computer Science." In Winter et al. [345], chapter Hybrid GA for Multi-Objective Aerodynamic Shape Optimization, 397–416.

257. Poloni, Carlo, et al. "Multiobjective Optimization by GAs: Application to System and Component Design." *Computational Methods in Applied Sciences '96: Invited Lectures and Special Technological Sessions of the Third ECCOMAS Computational Fluid Dynamics Conference and the Second ECCOMAS Conference on Numerical Methods in Engineering*. 258–264. Chichester: Wiley, 1996.

258. Poloni, Carlo and Valentino Pediroda. "GA Coupled with Computationally Expensive Simulations: Tools to Improve Efficiency." In Quagliarella et al. [261], 267–288.

259. Porto, William, editor. *Proceedings of the 1997 (4th) IEEE International Conference on Evolutionary Computation*, Piscataway, NJ: IEEE, April 1997.

260. Qiu, Min. "Prioritizing and Scheduling Road Projects by Genetic Algorithm," *Mathematics and Computers in Simulation*, *43*:569–574 (1997).

261. Quagliarella, D., et al., editors. *Genetic Algorithms and Evolution Strategy in Engineering and Computer Science: Recent Advances and Industrial Applications*. Chichester: John Wiley & Sons, 1997.

262. Quagliarella, D. and A. Vicini. "Subpopulation Policies for a Parallel Multiobjective Genetic Algorithm with Applications to Wing Design." In DiCesare and Jafari [89], 3142–3147.

263. Quagliarella, Domenico and Alessandro Vicini. "Coupling Genetic Algorithms and Gradient Based Optimization Techniques." In Quagliarella et al. [261], 289–309.

264. Radcliffe, Nicholas J. *Handbook of Evolutionary Computation*, chapter Schema Processing, B.2.5:1 – B.2.5:10. Volume 1 of Bäck et al. [19], 1997.

265. Rao, S. S. "Genetic Algorithmic Approach for Multiobjective Optimization of Structures." *ASME Annual Winter Meeting, Structures and Controls OptimizationAD-Vol. 38*. 29–38. November 1993.

266. Rawlins, Gregory J. E., editor. *Foundations of Genetic Algorithms*. San Mateo, CA: Morgan Kaufmann, 1991.

267. Reardon, Brian J. *Fuzzy Logic vs. Niched Pareto Multiobjective Genetic Algorithm Optimization: Part I. Shaffer's F2 Problem*. Technical Report LA-UR-97-3675, Los Alamos, New Mexico: Los Alamos National Laboratory, September 1997.

268. Reardon, Brian J. *Fuzzy Logic vs. Niched Pareto Multiobjective Genetic Algorithm Optimization: Part II. A Simplified Born-Mayer Problem*. Technical Report LA-UR-97-3676, Los Alamos, New Mexico: Los Alamos National Laboratory, September 1997.

269. Reardon, Brian J. *Optimization Of Densification Modeling Parameters of Beryllium Powder using a Fuzzy Logic Based Multiobjective Genetic Algorithm*. Technical Report LA-UR-98-1036, Los Alamos, New Mexico: Los Alamos National Laboratory, March 1998.

270. Reardon, Brian J. *Optimization of Micromechanical Densification Modeling Parameters For Copper Powder using a Fuzzy Logic Based Multiobjective Genetic Algorithm*.

Technical Report LA-UR-98-0419, Los Alamos, New Mexico: Los Alamos National Laboratory, January 1998.

271. Reinelt, Gerhard, "Traveling Salesman Problem Library." Online, 1995. Available: http://softlib.rice.edu/softlib

272. Ritzel, Brian J., et al. "Using Genetic Algorithms to Solve a Multiple Objective Groundwater Pollution Containment Problem," *Water Resources Research*, *30*(5):1589–1603 (May 1994).

273. Rodríguez-Vázquez, Katya, et al. "Multiobjective Genetic Programming : A Nonlinear System Identification Application." *Late Breaking Papers at the Genetic Programming 1997 Conference*, edited by John R. Koza. 207–212. Stanford University, California: Stanford Bookstore, July 1997.

274. Rowe, Jon, et al. "Parallel GAs for Multiobjective Functions." *2nd Nordic Workshop on Genetic Algorithms and Their Applications (2NWGA)*, edited by Jarmo T. Alander. 61–70. Vaasa, Finland: University of Vaasa, 1996.

275. Rudolph, Günter. "Evolutionary Search for Minimal Elements in Partially Ordered Finite Sets." *Proceedings of the Seventh Annual Conference on Evolutionary Programming*. Berlin: Springer, 1998.

276. Rudolph, Günter. "On a Multi-Objective Evolutionary Algorithm and Its Convergence to the Pareto Set." *Proceedings of the 1998 IEEE Conference on Evolutionary Computation*. 1998.

277. Russell, Stuart and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Upper Saddle River, NJ: Prentice Hall, 1995.

278. Ryan, Conor. "Pygmies and Civil Servants." *Advances in Genetic Programming* edited by Jr. Kenneth E. Kinnear, The MIT Press, 1994.

279. Ryan, Conor. "Racial Harmony and Function Optimization in Genetic Algorithms - The Races Genetic Algorithm." *Evolutionary Programming IV: Proceedings of the Fourth Annual Conference on Evolutionary Programming*, edited by J. R. McDonnell, et al. 109–125. Cambridge, MA: MIT Press, 1995.

280. Sakawa, Masatoshi, et al. "Fuzzy Multiobjective Combinatorial Optimization Through Revised Genetic Algorithms," *Japanese Journal of Fuzzy Theory and Systems*, *6*(1):77–88 (1994).

281. Sakawa, Masatoshi and Toshihiro Shibano. "An Interactive Fuzzy Satisficing Method for Multiobjective 0-1 Programming Problems with Fuzzy Numbers Through Genetic Algorithms with Double Strings," *European Journal of Operational Research*, *107*:564–574 (1998).

282. Sakawa, Masatoshi, et al. "An Interactive Fuzzy Method for Multiobjective 0-1 Programming Problems with Fuzzy Number Criteria Using Genetic Algorithms," *Electronics and Communications in Japan*, *81*(8):64–72 (1998).

283. Saludjian, L., et al. “Genetic Algorithm and Taylor Development of the Finite Element Solution for Shape Optimization of Electromagnetic Devices,” *IEEE Transactions on Magnetics*, *34*(5):2841–2844 (September 1998).

284. Sammon, J. W. “A Nonlinear Mapping for Data Structure Analysis,” *IEEE Transactions on Computers*, *C-18*(5):401–408 (1969).

285. Sandgren, Eric. “Multicriteria Design Optimization by Goal Programming.” *Advances in Design Optimization* edited by Hojjat Adeli, 225–265, London: Chapman & Hall, 1994.

286. Santos, Amâncio and António Dourado. “Constrained GA Applied to Production and Energy Management of a Pulp and Paper Mill.” *Proceedings of the 1999 ACM Symposium on Applied Computing*, edited by Janice Carroll, et al. 324–332. 1999.

287. Savic, Dragan A., et al. “Multiobjective Genetic Algorithms for Pump Scheduling in Water Supply.” *AISB International Workshop on Evolutionary Computing, Lecture Notes in Computer Science1305*. 227–236. Berlin: Springer, April 1997.

288. Schaffer, J. D. *Some Experiments in Machine Learning Using Vector Evaluated Genetic Algorithms*. PhD dissertation, Vanderbilt University, Nashville, TN, 1984.

289. Schaffer, J. David. “Multiple Objective Optimization with Vector Evaluated Genetic Algorithms.” In Grefenstette [134], 93–100.

290. Schaffer, J. David and John J. Grefenstette. “Multiobjective Learning via Genetic Algorithms.” *Proceedings of the 9th International Joint Conference on Artificial Intelligence*. 593–595. 1985.

291. Schnecke, Volker and Oliver Vornberger. “Hybrid Genetic Algorithms for Constrained Placement Problems,” *IEEE Transactions on Evolutionary Computation*, *1*(4):266–277 (November 1997).

292. Schott, J. R. *Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization*. MS thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 1995.

293. Schroder, P., et al. “Multi-Objective Optimization of Distributed Active Magnetic Bearing Controllers.” *Genetic Algorithms in Engineering Systems: Innovations and Applications*. 13 – 18. September 1997.

294. Schwab, M., et al. *Multi-Objective Genetic Algorithm for Pump Scheduling in Water Supply Systems*. Technical Report 96/02, Exeter, United Kingdom: Centre For Systems And Control Engineering, School of Engineering, University of Exeter, 1996.

295. Schwefel, Hans-Paul. *Evolution and Optimum Seeking*. New York: John Wiley & Sons, Inc., 1995.

296. Sette, Stefan, et al. “Optimizing a Production Process by a Neural Network/Genetic Algorithm Approach,” *Engineering Applications in Artificial Intelligence*, *9*(6):681–689 (1996).

297. Shaw, K. J. and P. J. Fleming. "Initial Study of Multi-Objective Genetic Algorithms for Scheduling the Production of Chilled Ready Meals." *Proceedings of the Second International Mendel Conference on Genetic Algorithms*. June 1996.

298. Shaw, K. J. and P. J. Fleming. "An Initial Study of Practical Multi-Objective Production Scheduling Using Genetic Algorithms." *Proceedings of the International Conference on Control'96*. September 1996.

299. Shaw, K. J. and P. J. Fleming. "Including Real-Life Preferences in Genetic Algorithms to Improve Optimisation of Production Schedules." *Proceedings of the GALESIA'97*. Glasgow, Scotland: IEE, September 1997.

300. Shaw, K. J. and P. J. Fleming. "Use of Rules and Preferences for Schedule Builders in Genetic Algorithms Production Scheduling." *Proceedings of the AISB'97 Workshop on Evolutionary Computation. Lecture Notes in Computer Science No. 1305*, edited by Corne and Shapiro. Manchester University: Springer-Verlag, 1997.

301. Sherman, Porter. *Ranking Techniques in Multicriteria Genetic Algorithm-Based Optimization*. PhD dissertation, Department of Computer and Information Science, Polytechnic University, Brooklyn, New York, 1995.

302. Shibano, Toshihiro and Masatoshi Sakawa. "Interactive Decision Making for Fuzzy Multiobjective 0-1 Programs Through Genetic Algorithms with Double Strings." *Proceedings of the Sixth IEEE Conference on Fuzzy Systems*. 1639–1644. 1997.

303. Shoaf, Jaqueline S. and James A. Foster. "A Genetic Algorithm Solution to the Efficient Set Problem: A Technique for Portfolio Selection Based on the Markowitz Model." *Proceedings of the Decision Sciences Institute Annual Meeting*. 571–573. 1996.

304. Siegel, Eric V. and Alexander D. Chaffee. "Genetically Optimizing the Speed of Programs Evolved to Play Tetris." *Advances in Genetic Programming 2* edited by Angeline and Kinnear, 279–298, MIT PRess, 1996.

305. Sridhar, Jagabandhu and Chandrasekharan Rajendran. "Scheduling in Flowshop and Cellular Manufacturing Systems with Multiple Objectives – A Genetic Algorithmic Approach," *Production Planning & Control*, *7*(4):374–382 (1996).

306. Srinivas, N. and Kalyanmoy Deb. "Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms," *Evolutionary Computation*, *2*(3):221–248 (1994).

307. Stanley, Timothy J. and Trevor Mudge. "A Parallel Genetic Algorithm for Multiobjective Microprocessor Design." In Eshelman [100], 597–604.

308. Steuer, Ralph E. *Multiple Criteria Optimization: Theory, Computation, and Application*. New York: John Wiley & Sons, 1986.

309. Surry, Patrick D., et al. "A Multi-Objective Approach to Constrained Optimization of Gas Supply Networks: The COMOGA Method." *Evolutionary Computing: AISB Workshop* edited by T. C. Fogarty, 166–180, Springer-Verlag, 1995.

310. Syswerda, Gilbert and Jeff Palmucci. "The Application of Genetic Algorithms to Resource Scheduling." *Proceedings of the Fourth International Conference on Genetic Algorithms*, edited by Richard K. Belew and Lashon B. Booker. 502–508. San Francisco, CA: Morgan Kaufmann Publishers, Inc., 1991.

311. Takada, Y., et al. "An Approach to Portfolio Selection Problems Using Multi-Objective Genetic Algorithms." *Proceedings of the 23rd Symposium on Intelligent Systems*. 103–108. 1996.

312. Takahashi, S., et al., "Inverse Optimization of Transonic Wing Shape for Mid-Size Regional Aircraft." AIAA Paper 98-0601, January 1998.

313. Tamaki, H., et al. "Generation of a Set of Pareto-Optimal Solutions by Genetic Algorithms," *Transactions of the Society of Instrument and Control Engineers*, *31*(8):1185–1192 (1995).

314. Tamaki, Hisashi, et al. "Multi-Objective Optimization by Genetic Algorithms: A Review." In Bäck, Thomas and Zbigniew Michalewicz and Hiroaki Kitano [20], 517–522.

315. Tamaki, Hisashi, et al. "Multi-Criteria Optimization by Genetic Algorithms: A Case of Scheduling in Hot Rolling Process." *Proceedings of the 3rd Conference of the Association of Asian-Pacific Operational Research Societies*. 374–381. Singapore: World Scientific, 1994.

316. Tanaka, Masahiro, et al. "GA-Based Decision Support System for Multicriteria Optimization." *Proceedings of the International Conference on Systems, Man, and Cybernetics2*. 1556–1561. 1995.

317. Tang, K. S., et al. "Structured Genetic Algorithm for Robust H^∞ Control Systems Design," *IEEE Transactions on Industrial Electronics*, *43*(5):575–582 (October 1996).

318. Thomas, Mark W. *A Pareto Frontier for Full Stern Submarines via Genetic Algorithm*. PhD dissertation, Ocean Engineering Department, Massachusetts Institute of Technology, Cambridge, MA, June 1998.

319. Todd, David S. and Pratyush Sen. "A Multiple Criteria Genetic Algorithm for Containership Loading." In Bäck [18], 674–681.

320. Todd, David S. and Pratyush Sen. "Multiple Criteria Scheduling using Genetic Algorithms in a Shipyard Environment." *Proceedings of the 9th International Conference on Computer Applications in Shipbuilding*. October 1997.

321. Trebi-Ollennu, A. and B. A. White. "Multiobjective Fuzzy Genetic Algorithm Optimization Approach to Nonlinear Control System Design." *Proceedings of the UKACC International Conference on CONTROL '96*. 460–466. 1996.

322. Tsoi, Effie, et al. "Hybrid GA/SA Algorithms for Evaluating Trade-off Between Economic Cost and Environmental Impact in Generation Dispatch." In Fogel [102], 132–137.

323. Tzeng, Gwo-Hshiung and Jen-Swei Kuo. "Fuzzy Multiobjective Double Sampling Plans with Genetic Algorithms Based on Bayesian Model." *Proceedings of the International Joint Conference of CFSA/IFIS/SOFT95 on Fuzzy Theory and Applications*, edited by Weiling Chiang and Jonathan Lee. 59–64. Singapore: World Scientific, 1995.

324. Valenzuela-Rendón, Manuel and Eduardo Uresti-Charre. "A Non-Generational Genetic Algorithm for Multiobjective Optimization." In Bäck [18], 658–665.

325. Van Veldhuizen, David A. and Gary B. Lamont. "Evolutionary Computation and Convergence to a Pareto Front." *Late Breaking Papers at the Genetic Programming 1998 Conference*, edited by John R. Koza. 221–228. Stanford, CA: Stanford University Bookstore, July 1998.

326. Van Veldhuizen, David A. and Gary B. Lamont. *Multiobjective Evolutionary Algorithm Research: A History and Analysis*. Technical Report TR-98-03, Air Force Institute of Technology, 1998.

327. Van Veldhuizen, David A. and Gary B. Lamont. "Multiobjective Evolutionary Algorithm Test Suites." *Proceedings of the 1999 ACM Symposium on Applied Computing*, edited by Janice Carroll, et al. 351–357. 1999.

328. Van Veldhuizen, David A., et al. "Finding Improved Wire-Antenna Geometries with Genetic Algorithms." In Fogel [103], 102–107.

329. Vedarajan, Ganesh, et al. "Investment Portfolio Optimization using Genetic Algorithms." *Late Breaking Papers at the Genetic Programming 1997 Conference*, edited by John R. Koza. 255–263. Stanford University, California: Stanford Bookstore, July 1997.

330. Vemuri, V. Rao and Walter Cedeno. "A New Genetic Algorithm for Multi Objective Optimization in Water Resource Management." In Fogel [102], 495–500.

331. Vicini, A. and D. Quagliarella. *Multipoint Transonic Airfoil Design by Means of a Multiobjective Genetic Algorithm*. Technical Report AIAA-97-0082, Washington, D. C.: AIAA, 1997.

332. Vicini, A. and D. Quagliarella. *Airfoil and Wing Design Through Hybrid Optimization Strategies*. Technical Report AIAA-98-2729, Washington, D.C.: AIAA, 1998.

333. Vicini, Alessandro and Domenico Quagliarella. "Inverse and Direct Airfoil Design Using a Multiobjective Genetic Algorithm," *AIAA*, *35*(9):1499–1505 (September 1997).

334. Viennet, R., et al. "Multicriteria Optimization Using a Genetic Algorithm for Determining a Pareto Set," *International Journal of Systems Science*, *27*(2):255–260 (1996).

335. Voget, S. "Multiobjective Optimization with Genetic Algorithms and Fuzzy Control." *Proceedings of the European Congress on Intelligent Techniques and Soft Computing (EUFIT)*. 391–394. September 1996.

336. Voget, S. and M. Kolonko. "Multidimensional Optimization with a Fuzzy Genetic Algorithm," *Journal of Heuristics*, *4*(3):221–244 (September 1998).

337. Voigt, Hans-Michael, et al., editors. *Parallel Problem Solving from Nature - PPSN IV*, 1996.

338. Wallace, David R., et al. "Design Search Under Probabilistic Specifications Using Genetic Algorithms," *Computer-Aided Design*, *28*(5):405–421 (1996).

339. "WWWebster Dictionary." Online, 1999. Available: http://www.m-w.com.

340. Weile, D. S., et al. "Multiobjective Synthesis of Electromagnetic Devices Using Non-dominated Sorting Genetic Algorithms." *Proceedings of the 1996 IEEE International Symposium on Antennas and Propagation*. 592–595. 1996.

341. Whitley, Darrell, et al. "Evaluating Evolutionary Algorithms," *Artificial Intelligence*, *85*:245–276 (1996).

342. Whitley, L. Darrell. "Fundamental Principles of Deception in Genetic Search." In Rawlins [266], 221–241.

343. Wienke, Dietrich, et al. "Multicriteria Target Vector Optimization of Analytical Procedures Using a Genetic Algorithm," *Analytica Chimica Acta*, *265*:211–225 (1992).

344. Wilson, P. B. and M. D. Macleod. "Low Implementation Cost IIR Digital Filter Design Using Genetic Algorithms." *IEE/IEEE Workshop on Natural Algorithms in Signal Processing1*. 4/1 – 4/8. Chelmsford, UK: IEE, 1993.

345. Winter, G., et al., editors. *Genetic Algorithms in Engineering and Computer Science*. Chichester: Wiley & Sons, 1995.

346. Wolpert, David H. and William G. Macready. "No Free Lunch Theorems for Optimization," *IEEE Transactions on Evolutionary Computation*, *1*(1):67–82 (April 1997).

347. Wright, Alden H. "Genetic Algorithms for Real Parameter Optimization." In Rawlins [266], 205–220.

348. Yang, Xiaofeng and Mitsuo Gen. "Evolution Program for Bicriteria Transportation Problem." *Proceedings of the 16th Annual Conference on Computers and Industrial Engineering*. 481–484. Pergamon, 1994.

349. Yao, X. and Y. Liu. "Fast Evolutionary Programming." *Proceedings of the Fifth Annual Conference on Evolutionary Programming (EP '96)*. 451–460. MIT Press, 1996.

350. Yao, X. and Y. Liu. "Fast Evolution Strategies." *Proceedings of the Sixth Annual Conference on Evolutionary Programming (EP '97)*. 151–161. Springer-Verlag, 1997.

351. Yoshida, Koji, et al. "Generating Pareto Optimal Decision Trees by GAs." *Methodologies for the Conception, Design, and Application of Intelligent Systems: Proceedings of IIZUKA '96*. 854–858. 1996.

352. Yoshimura, Kazuyuki and Ryohei Nakano. "Genetic Algorithm for Information Operator Scheduling." In Fogel [103], 277–282.

353. Zebulum, R. S., et al. "Synthesis of CMOS operational amplifiers through Genetic Algorithms." *Proceedings of the Brazilian Symposium on Integrated Circuits, SBCCI'98*. 125–128. Rio de Janeiro, Brazil: IEEE, September 1998.

354. Zebulum, R.S., et al. "Synthesis of CMOS Operational Amplifiers Through Genetic Algorithms." *Proceedings of the XIII International Conference in Microelectronics and Packaging*. 12–14. August 1998.

355. Zhang, Byoung-Tak and Heinz Mühlenbein. "Adaptive Fitness Functions for Dynamic Growing/Pruning of Program Trees." *Advances in Genetic Programming 2* edited by Angeline and Kinnear, 241–256, MIT Press, 1996.

356. Zhou, Gengui and Mitsuo Gen. "Evolutionary Computation on Multicriteria Production Process Planning Problem." In Porto [259], 419–424.

357. Zhou, Gengui and Mitsuo Gen. "Genetic Algorithm Approach on Multi-Criteria Minimum Spanning Tree Problem," *European Journal of Operations Research*, *114*(1) (April 1999).

358. Zitzler, Eckart and Lothar Thiele. *An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach*. Technical Report TIK 43, Computer Engineering and Communication Networks Lab, Swiss Federal Institute of Technology, Gloriastrasse 35, CH-8092, Zurich, Switzerland: Swiss Federal Institute of Technology (ETH), May 1998.

359. Zitzler, Eckart and Lothar Thiele. "Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Case Study." In Eiben et al. [96].

360. Zuse, Konrad, "Mathematical Programming Testdata." Online, 1997. Available: ftp://ftp.zib.de/Packages/index.html.

Vita

Captain David Van Veldhuizen enlisted in the US Air Force and entered active duty in 1981. During the next six years he served as a Material Facilities Specialist and Computer Operations Shift Supervisor, rising in rank from Airman Basic to Staff Sergeant. While on active duty he received an Associate's Degree in Management Information Systems from the Community College of the Air Force in April 1987. He separated from active duty the next month, enlisting as a cadet at Detachment 470, Reserve Officer Training Corps (ROTC) at the University of Nebraska-Omaha, Omaha, NE.

He was awarded a Bachelor of General Studies Degree by the University of Nebraska-Omaha in August 1988. After his ROTC requirements were fulfilled he was commissioned a 2nd Lieutenant, US Air Force Reserve in May 1989. David received a Master's Degree in Applied Mathematics from the University of Nebraska in December 1989. He then served as an Air Force Scientific Analyst and High Power Microwave Project Leader during the next six years, whereupon he was assigned to the Air Force Institute of Technology (AFIT).

David's military assignment's include Laughlin AFB, TX; Offutt AFB, NE (twice); Kirtland AFB, NM; and Wright-Patterson AFB, OH. Temporary duty assignments and travel have taken him to 36 states, Washington, D.C., Mexico, and Korea – but Europe still eludes him. His AFIT follow-on assignment is with the Air Force Research Laboratory, Brooks AFB, TX.

Permanent address: 1307 North Lincoln
Aberdeen, South Dakota 57401