(Final Draft) TR-98-03

Multiobjective Evolutionary Algorithm Research: A History and Analysis

David A. Van Veldhuizen and **Gary B. Lamont**
Department of Electrical and Computer Engineering
Graduate School of Engineering
Air Force Institute of Technology
Wright-Patterson AFB, OH 45433-7765
{dvanveld,lamont}@afit.af.mil

October 14, 1998

Contents

1 Introduction **8**

2 MOEA Background **10**
2.1 Multiobjective Technique Classification . . . 10
2.2 Search Space Description . . . 11
2.3 Mathematical Notation . . . 12
2.4 Pareto Optimality and Terminology . . . 12
2.4.1 MOEA Pareto Notation . . . 13
2.5 Presentation Layout . . . 13

3 *A Priori* Techniques **15**
3.1 Lexicographic Techniques . . . 15
3.2 Linear Fitness Combination Techniques . . . 15
3.3 Nonlinear Fitness Combination Techniques . . . 17
3.3.1 Multiplicative Techniques . . . 17
3.3.2 Target Vector Techniques . . . 18
3.3.3 Minimax Techniques . . . 18

4 Progressive Techniques **20**

5 *A Posteriori* Techniques **21**
5.1 Independent Sampling Techniques . . . 21
5.2 Criterion Selection Techniques . . . 22
5.3 Aggregation Selection Techniques . . . 23
5.4 Pareto Sampling Techniques . . . 25
5.4.1 Pareto Rank- and Niche-Based Selection . . . 27
5.4.2 Pareto Deme-Based Selection . . . 28
5.4.3 Pareto Elitist-Based Selection . . . 29

6 Related MOEA Publications **31**
6.1 Technique Comparisons . . . 31
6.2 Multiobjective EA Theory . . . 32

7 MOEA Survey Analysis (Quantitative) **33**
7.1 Published Papers: Numbers and Dates . . . 33
7.2 Technique Discussions . . . 34
7.2.1 *A Priori* Techniques . . . 34
7.2.2 Progressive Techniques . . . 36
7.2.3 *A Posteriori* Techniques . . . 36
7.2.4 MOEA Comparisons . . . 38
7.2.5 MOEA Theory . . . 38
7.3 MOEA Fitness Functions . . . 38
7.4 MOEA Chromosomal Representations . . . 39

7.5 MOEA Problem Domains 40

8 MOEA Survey Analysis (Empirical) 41
8.1 MOEA Characteristics 41
8.2 MOEA Theory 42
8.2.1 Fitness Functions 43
8.2.2 Pareto Ranking 44
8.2.3 Pareto Niching and Fitness Sharing 45
8.2.4 Mating Restriction 46
8.2.5 Solution Stability or Robustness 47
8.3 MOEA Secondary Populations 48
8.4 MOEA Complexity 49
8.5 MOEAs for Beginners: Which are Appropriate? 50

9 Parallel MOEAs 52
9.1 MOEA Task Decomposition 52
9.1.1 Decomposing MOEA Fitness Assignment and Transformation 53
9.2 MOEA Parallelization 55
9.2.1 Parallel Sharing and Niching 55
9.3 MOP and MOEA Domain Integration 56
9.3.1 Integration Mechanics 57
9.3.2 Parallel Scheduling 58
9.3.3 Load Balancing Issues 58
9.3.4 Parallel Performance Issues 59
9.3.5 Parallel Architecture Issues 59
9.3.6 Parallel MOEA Effectiveness 60

10 MOP Test Functions 61
10.1 An MOP Test Suite: Is it Important? 61
10.1.1 General MOEA Test Suite Issues 62
10.1.2 MOEA Test Suite Guidelines 62
10.1.3 Requirements for an MOEA Test Suite 63
10.2 An MOEA Test Function Suite 63
10.2.1 The MOP Domain 67
10.2.2 MOEA Test Suite Functions 68
10.2.3 NP-Complete MOPs 69
10.3 MOEA Experimental Methodology 69
10.3.1 Experimental Database 70
10.3.2 MOEA Comparative Metrics 71

11 Conclusions and Recommendations 73

12 Things to Think About 86

13 NP-Complete MOPs **87**
13.1 Knapsack Problem 87
13.2 Knapsack 2 87
13.3 Traveling Salesman Problem 87
13.4 Vehicle Routing Problem 88

List of Figures

1 EA-Based MOP Solution Techniques 11
2 MOP Evaluation Mapping 12
3 MOEA Citations by Year 33
4 MOEA Citations by Technique 33
5 Generalized EA Task Decomposition 41
6 MOEA Task Decomposition 41
7 Rank Assignment Algorithm 44
8 Parallel Fitness Evaluation Possibilities 54
9 Serial Sharing & Niching MOEA Pseudo-Code 55
10 Parallel MOEA Task Decomposition 57

List of Tables

1 Lexicographic Techniques . . . 15
2 Linear Fitness Combination . . . 16
3 Multiplicative Techniques . . . 18
4 Target Vector Techniques . . . 18
5 Minimax Techniques . . . 19
6 Interactive Techniques . . . 20
7 Independent Sampling Techniques . . . 21
8 Criterion Selection Techniques . . . 22
9 Aggregation Selection Techniques . . . 24
10 Pareto Selection Techniques: Ranking . . . 26
11 Pareto Selection Techniques: Ranking and Niching . . . 27
12 Pareto Selection Techniques: Demes . . . 29
13 Pareto Selection Techniques: Elitist . . . 29
14 Technique Comparisons . . . 31
15 MOEA Theory . . . 32
16 MOEA Fitness Function Types . . . 39
17 MOEA Algorithmic Complexity . . . 50
18 MOP Numeric Test Functions . . . 63
19 MOP Numeric Test Functions (with side constraints) . . . 65
20 Possible Multiobjective NP-Complete Functions . . . 70

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. Thus, the intent of this paper is to organize, present, and analyze contemporary Multiobjective Evolutionary Algorithm (MOEA) research and associated MOPs. Under the umbrella of *a priori*, progressive, and *a posteriori* algorithms, all known MOEA techniques are discussed. Each MOEA proposed in the literature is classified and cataloged based upon this umbrella and more detailed algorithmic characteristics; among others these include objective aggregation, interactive methods, sampling, search, ranking, and niching. The classification, incorporating a consistent MOEA notation, is presented in tabular form for ease of MOEA identification and selection.

A detailed quantitative and qualitative analysis is presented. The tabular data gives a basis for various conclusions about the various algorithmic techniques, fitness functions, gene representations, and problem domains within which MOEAs are applied. On a qualitative level, MOEA "state of the art" is discussed, including MOEA characteristics, theory, additional populations, and complexity. A detailed description of possible MOEA parallelization schemes is the basis for proposed parallel MOEA implementations.

Example MOPs from the current MOEA literature are also presented in tabular form. A proposed classification of these MOPs is based upon problem domain genotype and and phenotype characteristics; these include connectivity, disjointness, concave or convex shape, constraints, and symmetry. A collection of MOEA test suite guidelines is discussed based on our classification. Appropriate MOEA MOP test suites can then be generated based upon known MOP characteristics, quantitative evaluation of a specific MOEA approach, and applicable MOEA theory. An experimental methodology, associated experimental MOEA database, and a collection of test suite metrics are offered as a proposed evaluation framework.

Finally, philosophical issues concerning test suite generation, variations in MOEA structures, completeness of MOEA analysis, and the proposed focus of our future research are addressed. In its totality, this document presents a complete view of the current MOEA start-of-the-art and possible MOEA research trends.

1 Introduction

Multiobjective Evolutionary Algorithms (MOEAs) are an expanding research area in the Evolutionary Computation field. The first MOEA was Schaffer's Vector Evaluated Genetic Algorithm in 1985 [128]; since then, numerous EA-based approaches to solving Multiobjective Optimization Problems (MOPs) have appeared in the literature. For a good introduction to and historical overview of relevant MOEA concepts and efforts, see the articles by Fonseca and Fleming [50, 47] and by Horn [69].

This paper is more than a summary review of existing MOEAs however. Its overall objective is to organize, present, analyze, and extend existing MOEA research. This paper accomplishes this goal in several ways. First, it expands upon previous reviews by adding recent and related MOEA research. It classifies and catalogs all known (to date) MOEA efforts. Proposed approaches are grouped together, and key elements of each effort identified in a condensed summary. These categorized results are listed in tabular form, allowing easy perusal of past research, and quick access to desired information. The classification structure used was first proposed by Horn [69]; we substantiate and extend its use. Our cataloged presentation highlights previously unnoticed MOEA research trends, and distinguishes the various approaches from each other.

Second, the classification structure and cataloged components allow for easy identification of "suitable" MOEA techniques for a given MOP. A high-level discussion accompanies each technique, and the mathematical formulation for fitness assignment and/or selection is also presented. Quantitative conclusions are drawn, e.g., the appropriateness of various techniques for varying MOP domains, possible MOEA fitness function types, and the current state of MOEA theory.

Third, we address many relevant philosophical topics. The cataloging effort highlights several MOEA issues which are treated lightly or even ignored in the literature. For example, we discuss MOEA characteristics, fitness functions, Pareto ranking variants, niching and sharing, mating restriction, secondary populations, solution stability, complexity, and other selected topics.

This paper concludes by presenting all MOEA numeric test functions found in the cited literature. Some of these are not appropriate MOEA test functions because they are not characteristic of typical MOPs. We thus discuss and suggest an appropriate MOEA benchmark function set to test major MOP domain characteristics.

In this report we formalize an algorithmic framework for the important and rapidly expanding research in MOEAs. This review currently classifies more than 100 references to relevant research efforts and practical applications. The listing is *not* absolute; no matter how much effort is spent collecting and evaluating references, a proposed listing is never complete. We also assume many applications might remain unpublished for confidentiality reasons, but conjecture that our reported data is representative of the field's direction(s). Some approaches have been published in several different venues; we cite only primary sources for each distinct effort.

The remainder of this document is organized as follows. Section 2 introduces our classification scheme and discusses necessary mathematical and MOP concepts. Sections 3, 4, and 5 present the three major categories of MOEA research. Section 6 presents publications whose main purpose is either comparing different MOEA techniques or discussing MOEA theory. Basic quantitative and qualitative analysis based on the cataloged research are given in Sections 7 and 8. Building on the analysis, Section 9 presents a discussion on MOEA parallelization. Section 10 discusses historic MOEA test functions and presents a proposed MOEA test suite. Finally, our conclusions of past research and recommendations for future research are offered in Section 11.

2 MOEA Background

Many successful MOEA approaches are predicated upon previous mathematical MOP solution techniques. For example, the Operations Research field proposed several methods well before 1985 [27, 135, 73]. Their Multiple Objective Decision Making (MODM) problems are closely related to design problems; 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 nonquantifiable objectives) [73].

Various MODM techniques are commonly classified from a Decision Maker's (DM's) point of view (i.e., how the DM performs search and decision making). Cohon [26] further distinguishes methods between two types of DM: a single DM/group, or multiple DMs whose decisions conflict. In this paper we consider the DM to be either a single DM or a group, but a group united in its decisions.

2.1 Multiobjective Technique Classification

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 results from both *optimization* (by some method) and *decision* processes. Several OR classification schemes [73, 27] define three variants of this process: the final solution results from a DM's preferences being made known either before, during, or after the optimization process. This is more formally declared as follows [73]:

***A Priori* Preference Articulation.** (*Decide* $\longrightarrow$ *Search*) Decision maker 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*) Decision maker is presented with a set of efficient (defined in Section 2.4) 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 now limit discussion to implemented MOEA techniques. A hierarchy of known approaches is shown in Figure 1 where the techniques are classified by the different ways in which the fitness function and/or selection is treated. See Cohon [27] and Duckstein [38] for other multiobjective techniques which may be suitable for, but have not yet been implemented in MOEAs.

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 1.

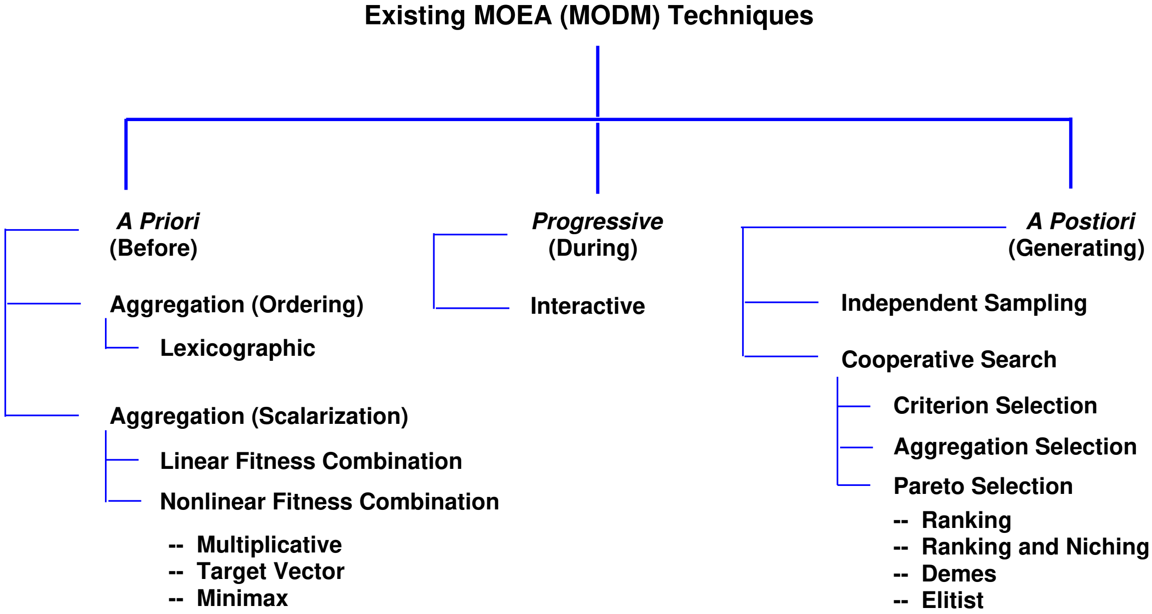

Figure 1: EA-Based MOP Solution Techniques

2.2 Search Space Description

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 decision space. A decision maker then implicitly chooses an acceptable solution by selecting one of these vectors. In mathematical terms, an MOP minimizes[1] 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. Or in general,

$$\begin{aligned} \text{minimize} \quad & F(\vec{x}) = (f_1(\vec{x}), \ldots, f_k(\vec{x})) \\ \text{subject to} \quad & g_i(\vec{x}) \leq 0, i = 1, \ldots, m, \ \vec{x} \in \Omega \, . \end{aligned} \tag{1}$$

An MOP thus consists of n variables, m constraints, and k objectives, of which any or all of the objective functions may be linear or nonlinear [73]. The MOP's evaluation function, $F : \Omega \longrightarrow \Lambda$, maps decision variables $x_1, \ldots, x_n$ to vectors ($\vec{y} = a_1, \ldots, a_k$). This situation is represented in Figure 2 for the case $n = 2$ and $k = 3$.

MOPs are often characterized by distinct measures of performance (the objectives), which may be (in)dependent and/or non-commensurable. The multiple objectives being optimized almost always conflict. These opposing objectives place a partial, rather than total, ordering on the search space. In fact, finding the global optimum of a general MOP is NP-complete [6]. *Optimal* solutions to these problems, where all decision variables satisfy associated constraints and the objective function attains a global minimum, may not even exist.

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

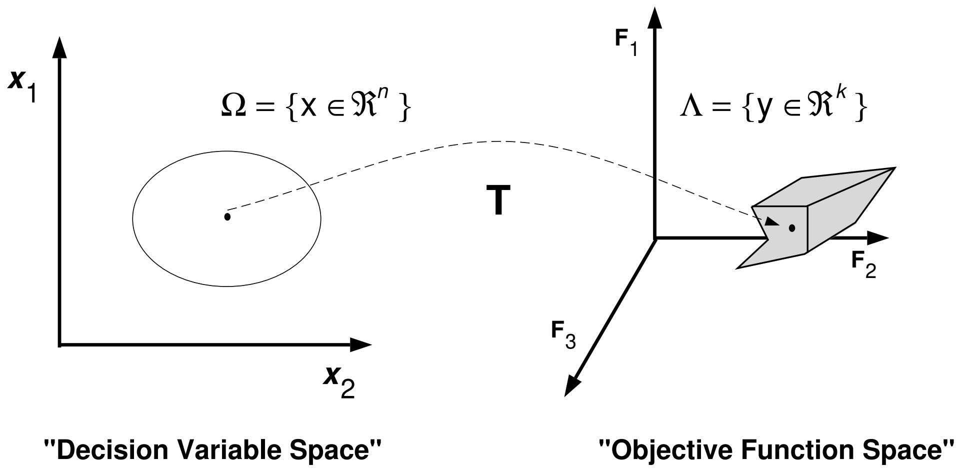

Figure 2: MOP Evaluation Mapping

2.3 Mathematical Notation

We define a formal MOP model to define the mathematical notation used in this paper. The k objectives (where $k \geq 2$) are denoted by $f_1, \ldots, f_k$. All problems are assumed to be minimization problems unless otherwise specified.

$$
\begin{aligned}
\Phi &: \Omega \longrightarrow \Lambda \\
f(a_i) &\triangleq F(f_1(a_i), \ldots, f_k(a_i)) \\
\Psi &: P \longrightarrow P' \\
\mathbb{P} &= \{a_i | \forall a_i, a_i \text{is “desirable''}\}
\end{aligned} \tag{2}
$$

Equation 2 presents the general formats used to mathematically represent the various MOEA techniques. "$\Phi : \Omega \longrightarrow \Lambda$" describes a particular technique's domain (Ω) and range (Λ). "$f(a_i)$" is the solution's scalar fitness value derived via the defined equation. Ψ indicates the particular technique incorporates specialized selection EVOPs, perhaps *not* relying on an overall solution fitness. Section 5 explains that *A Posteriori* MOEA techniques seek the Pareto front. $\mathbb{P}$ is a set of solutions returned to a DM such that every solution in the set is a member of P_{known} .

2.4 Pareto Optimality and Terminology

Although "Pareto optimality" and its related concepts and terminology are frequently invoked, they are sometimes used incorrectly in the literature. To ensure understanding and consistency, we define Pareto Dominance and Optimality, and then introduce a consistent notation. Using the MOP presented in Equation 1, key Pareto concepts are mathematically defined as follows [12]:

Definition 1 (Pareto Dominance): *A vector* $\mathbf{u} = (u_1, \ldots, u_k)$ *is said to dominate* $\mathbf{v} = (v_1, \ldots, v_k)$ *if and only if* $\mathbf{u}$ *is partially less than* $\mathbf{v}$*, i.e.,*

$\forall i \in \{1, \ldots, k\}, u_i \leq v_i \wedge \exists i \in \{1, \ldots, k\} : u_i < v_i$. □

Definition 2 (Pareto Optimality): *A solution* $x_u \in \mathcal{U}$ *is said to be Pareto optimal if and only if there is no* $x_v \in \mathcal{U}$ *for which* $v = f(x_v) = (v_1, \ldots, v_k)$ *dominates* $u = f(x_u) = (u_1, \ldots, u_k)$. □

Pareto optimal solutions are also termed *non-inferior*, *admissible*, or *efficient* solutions. Their corresponding vectors are termed *non-dominated* [69]; selecting a vector(s) from this non-dominated vector set implicitly indicates acceptable Pareto optimal solutions (genotypes). These solutions may have no clearly apparent relationship besides their membership in the Pareto optimal set. We stress here that Pareto optimal solutions are classified as such based on their *phenotypical* expression. Their expression (the non-dominated vectors), when plotted in criterion space, is known as the *Pareto front*. Researchers have inconsistently used these terms in the literature, implying that a consistent notation is required in order to eliminate the resulting confusion.

2.4.1 MOEA Pareto Notation

During EA execution, a "local" set of Pareto optimal solutions (with respect to the *current* EA population) is determined at each EA generation and termed $P_{current}$. This solution set can be added to a secondary population termed P_{known} (i.e., $P_{current} \bigcup P_{known}$), and the process repeated until EA termination. Because a solution's classification as Pareto optimal is dependent upon the context within which it is evaluated (i.e., the given set of which it's a member), all corresponding vectors of P_{known} are periodically tested and solutions whose associated vectors are dominated are removed. The result is a final set of Pareto optimal solutions *found by the EA*. Of course, the actual Pareto optimal solution set (termed P_{true}) is not explicitly known for problems of any difficulty. P_{true} is defined by the functions composing an MOP; it is fixed and does not change.[2]

Because of the manner in which Pareto optimality is defined, $P_{current}$ is always a non-empty solution set. We proved this in a previous work [148], as well as showing that the global optimum of an MOP is the Pareto front (PF_{true}) determined by evaluating the Pareto optimal solution set (P_{true}).

$P_{current}$ and P_{known} are sets of EA genotypes. EA fitness is judged in the phenotype domain, which is a Pareto front in the MOP case. We term the associated Pareto front for each of the previous solution sets as $PF_{current}$, PF_{known}, and PF_{true}. Thus, when using an EA to solve MOPs, the implicit assumption is that one of the following holds: $P_{known} = P_{true}$, $P_{known} \subset P_{true}$, or $PF_{known} \in [PF_{true}, PF_{true} + \epsilon]$ over some norm (Euclidean, RMS, etc.).

2.5 Presentation Layout

Figure 1 is presented. This is followed by each technique's mathematical description, such as Fonseca and Fleming present [47]. Finally, a table cataloging relevant research

[2]Horn [69] uses P_{online}, $P_{offline}$, and P_{actual} instead of $P_{current}$, P_{known}, and P_{true}.

efforts 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 were capture essential information about each approach; they are not meant as a complete description. Because of the manner in which the research efforts were classified, the "Approach" or "Description" categories contain information about the specific MOEA type, parallelized implementation, specialized evolutionary operators (EVOPs), etc. Finally, each table's entries are chronologically ordered by year published.

3 *A Priori* Techniques

All techniques presented in this section expect DM input before the EA search process begins. After search, an optimal solutionn is then presented to the DM. Ordering, linear, and nonlinear combination techniques are discussed.

3.1 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 [12] and is mathematically represented by:

$$\begin{aligned} \Phi \quad &: \quad \mathbb{R}^n \longrightarrow \{0, 1, \ldots, \mu - 1\} \\ f(a_i) \quad &\triangleq \quad \sum_{j=1}^{\mu} \chi\left(f(a_j)\ l\!<\ f(a_i)\right), \end{aligned} \tag{3}$$

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

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

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 1 lists the known lexicographic MOEA techniques.

Table 1: Lexicographic Techniques

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [54] (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) [37] (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

3.2 Linear Fitness Combination Techniques

Linear fitness combination is 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 [135]. This technique is mathematically represented by:

$$\begin{aligned} \Phi \quad &: \quad \mathbb{R}^n \longrightarrow \mathbb{R} \\ f(a_i) \quad &\triangleq \quad \sum_{j=1}^{k} w_j f_j(a_i) \ , \end{aligned} \tag{4}$$

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 2 lists the known linear combinatoric MOEA techniques.

Table 2: Linear Fitness Combination

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [137] (1991)	Hybrid GA implementation; Incorporates a schedule builder and evaluator	Laboratory resource scheduling	Not stated	Permutation task ordering
GA-based learning system [79] (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 [80] (1993)	Linear normalized fitness and weighted penalties	3-D structure conformational search	(2) Match penalty; Energy penalty	Binary string; Genes are rotation angles
GA [14] (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
Multi-Niche Crowding (MNC) GA [150, 21] (1995,1997)	Fitness obtained by summing individual rank ***in each objective***; Phenotypic-based crowding; Integrated with flow-transport simulation code	Groundwater pollution containimant monitoring; Also tested on multimodal, dynamic function	(3) Cost; Contanimant removal; Contanimant leakage	Variable length integer string; Genes are geographic nodes
GA [125] (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 [5] (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 [103] (1995)	Standard GA	Pot core transformer design	(2) Device area; Magnetic flux density	Binary string
GA [18] (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 [75] (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

Table 2: continued

Approach	Description	Application	Objectives (#)	Chromosome
Multi-Sexual GA [92] (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 [149] (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 [104] (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
GA [164] (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 [23] (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

3.3 Nonlinear Fitness Combination Techniques

Nonlinear fitness combination is also a scalar aggregative method; several EA-based variants have been implemented. This aggregation incorporates nonlinear terms normally derived in some "trial and error" fashion. Thus, *in general*, many researchers appear to use "more structured" methods.

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 [57], and transforming constraints "into" objectives. According to Cohon [26] this 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 $i = 2, \ldots, k$.

However, it appears no EA-based MOP implementations which primarily use penalty techniques have appeared, at least where the technique results in a *single* solution presented to the decision maker. They have been implemented as part of cooperative population searches, and are thus classified in Sections 5.3 and 5.4. Other implemented nonlinear combination techniques are now addressed in turn.

3.3.1 Multiplicative Techniques

Multiplicative techniques are scalar aggregative methods where individual functions are combined through multiplication. This technique's general form is mathematically represented by:

$$\Phi \quad : \quad \mathbb{R}^n \longrightarrow \mathbb{R}$$

$$f(a_i) \triangleq \prod_{j=1}^{k} f_j(a_i) \ . \tag{5}$$

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

Table 3: Multiplicative Techniques

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
Multi Attribute Utility Analysis (MAUA)-GA [70] (1993)	*Proposes* MAUA to determine fitness function	None	None	None
GA [155] (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

3.3.2 Target Vector Techniques

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

$$\begin{aligned} \Phi \ &: \ \mathbb{R}^n \longrightarrow \mathbb{R} \\ f(a_i) \ &\triangleq \ \| \, [f(a_i) - g]\mathbf{W}^{-1} \, \|_\alpha \ , \end{aligned} \tag{6}$$

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 ($\alpha = 2$) [158]. This technique can be used in fitness proportional, tournament, or rank-based selection. Table 4 lists the known target vector MOEA techniques.

Table 4: Target Vector Techniques

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [158] (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

3.3.3 Minimax 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:

$$\Phi \ : \ \mathbb{R}^n \longrightarrow \mathbb{R}$$

$$f(a_i) \triangleq \max_{j=1,\ldots,k} \frac{f_j(a_i) - g_j}{w_j}, \tag{7}$$

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 $\parallel g_j \parallel$ [47]. This technique can be used in fitness proportional, tournament, or rank-based selection. Table 5 lists the known minimax MOEA techniques.

Table 5: Minimax Techniques

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [159][3] (1993)	Goal attainment; Population monitored for nondominated solutions	Unknown	Unknown	Unknown
GA [24] (1995)	Tchebycheff weighting, Uniformly varies key parameter	Groundwater contanimant monitoring	(2) Undetected plumes; Contaminated area	Fixed-length integer string
GA [25] (1995)	Objectives optimized in turn; Results optimize weighted min-max formulation	Robot arm balancing	(4) Torque at joints 1 and 2; Reaction force at joints 1 and 2	Real values

[3]This effort was cited by Fonseca [50]; attempts to obtain the reference failed.

4 Progressive 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 5, the *a posteriori* techniques provide a *set* of solutions instead of a single one; this situation is often more preferable. Table 6 lists the known progressive MOEA techniques.

Table 6: Interactive Techniques

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
Multiple Objective Genetic Algorithm (MOGA) [48, 52] (1993, 1998)	Fonseca's [50] 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
GA [142] (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 [131] (1996)	Uses Fonseca's MOGA [48]; Compares to weighted-sum approach	Meal production line scheduling	(3) Rejected orders; Batch lateness; Shift/staff balancing	Permutation ordering
Evolutionary Co-Design (EvoC) [72, 61] (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 [51] (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

5 *A Posteriori* Techniques

A Posteriori techniques perform an MOP search process resulting in a set of identified solutions for DM selection. Both independent sampling and cooperative population search techniques are presented. We agree with Horn [69] 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 unnecessary. Thus, we consider a primary goal of these *a posteriori* techniques to be identification of P_{true}.

5.1 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:

$$\begin{aligned} \Phi &: \mathbb{I} \longrightarrow \mathbb{R} \\ \mathbb{P} &= \{a_i | \forall a_i, a_i \in P_{known}\} \ , \end{aligned} \tag{8}$$

where Φ is some fitness function assigning solution fitness for an entire EA "run" (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 results from a single run. Table 7 lists the known independent sampling MOEA techniques.

Table 7: Independent Sampling Techniques

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [54] (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 [100] (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 [122] (1994)	*Proposes* multiple GA runs optimizing one criterion at a time, then varying the constraints	None	None	None
GA [24] (1995)	Tchebycheff weighting, uniformly varies key parameter	Groundwater monitoring	(2) Undetected plumes; Contaminated area	Fixed-length integer string
GA [22] (1995)	Multiple runs uniformly varies weights; Fitness scaling	Firing angles in railway traction substations	(2) Power supply; Uniform load sharing	Binary string

Table 7: (continued)

Approach	Description	Application	Objectives (#)	Chromosome
GAA and GAA2 [146] (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

5.2 Criterion Selection Techniques

Criterion selection techniques are the first discussed to directly use an EA's population capability. 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. These techniques are mathematically represented by:

$$\begin{aligned} \Psi &: P \longrightarrow P' \\ \mathbb{P} &= \{a_i | \forall a_i, a_i \in P_{known}\} , \end{aligned} \qquad (9)$$

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 8 lists the known criterion selection MOEA techniques.

Table 8: Criterion Selection Techniques

Approach	Description	Application	Objectives (#)	Chromosome
Vector Evaluated GA (VEGA) [128] (1985)	$\frac{1}{k}$ of new population selected using each of the k objectives	None	(2) Numeric optimization	Binary string; Contains genes and objective performance information
ES [88] (1990)	Objectives' associated probabilities used as selection criteria; Polyploid individuals	None	(2) Numeric optimization	Both decision and stepsize variables have dominant and recessive chromosomes
ES (μ+λ) [3] (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 [118, 16] (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 [14] (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
GA [54] (1995)	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

Table 8: (continued)

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
Multiobjective GA [151] (1997)	Both nondominated- and roulette wheel (one objective)- based selection	Transonic airfoil design	(2) Mach number; Lift coefficient	Binary string; Genes are airfoil parameters
Multiobjective GA [109] (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 [42] (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 [31] (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 [?] (1998)	"n-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
ES [90] (1998)	"Predator-prey" model; Predators "attack" based on one of k objectives	None	(2) Numeric optimization	Real values

5.3 Aggregation Selection Techniques

Aggregation selection techniques also directly use an EA's population capability. Here, succeeding populations are selected using their fitness as computed by either linear or nonlinear combination techniques, and are thus able to find multiple members of P_{known} within a single EA run. These techniques are mathematically represented by:

$$\begin{aligned} \Psi &: P \longrightarrow P' \\ \mathbb{P} &= \{a_i | \forall a_i, a_i \in P_{known}\} , \end{aligned} \tag{10}$$

where Ψ is some generation transition function selecting solutions based on their performance using some aggregative technique (which is not necessarily identical for each evaluated solution, i.e., $\Phi : \mathbb{I} \longrightarrow \mathbb{R}$ changes during the run), and $\mathbb{P}$ (i.e., P_{known}) is returned to the DM. Table 9 lists the known aggregation selection MOEA techniques.

Table 9: Aggregation Selection Techniques

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [65] (1992)	Weighted sum; Weights are chromosomally encoded; Compares fitness sharing (applied only to weighting variables) and two VEGA [128] 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 [106] (1995)	Randomly assigned weights; Pareto elitist selection	None	(2) Numeric & Scheduling & Rule selection examples	Binary string & Permutation ordering & Tri-Valued string
ES [153] (1996)	Fuzzy controller selects each solution's evaluation function	Railway network scheduling	(2) Cost; Waiting time	Unknown
Multi-Objective Genetic Local Search Algorithm [76, 77] (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) [147] (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 [94] (1997)	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 [19] (1997)	Constraints converted into functions; Both efficient and dominated solutions determine search direction	None	(2) Numeric optimization (Original function; Constraints)	Real values
ES $(\mu+\lambda)$ [165] (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 [29] (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 [78] (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 [55] (1998)	Specialized encoding and selection EVOP; Incorporates adaptive objective evaluation hyperplane [165] and auxiliary bi-objective problem	Topological Network Design	(2) Connection cost; Message delay	Prüfer number encoding; Integer string uniquely encodes a spanning-tree

5.4 Pareto Sampling Techniques

Pareto sampling directly utilizes the EA's population capability. It also appears that some approaches incorporate a secondary population, storing all Pareto optimal solutions yet found during EA execution. When using these methods, the generational population (possibly) holds several solutions of P_{known} and at least one member of $P_{current}$. The secondary population is periodically updated to remove solutions which have become dominated. Pareto sampling techniques explicitly use Pareto concepts in selection so that Pareto solutions are given preference over dominated solutions, but are treated equivalently among themselves.

Two types of Pareto fitness assignment are in wide use. The first technique, proposed by Goldberg [57], is mathematically (recursively) represented by:

$$\begin{aligned} \Phi &: \mathbb{R}^n \longrightarrow \{1, \ldots, \mu\} \\ f(a_i) &\triangleq \begin{cases} 1 & \neg(f(a_j)\ p\!<\ f(a_i))\ \forall j \in \{1, \ldots, \mu\} \\ \phi & \neg(f(a_j)\ p\!<\ f(a_i))\ \forall j \in \{1, \ldots, \mu\} | \{l : \Phi(f(a_l)) < \phi\} \end{cases}, \end{aligned} \tag{11}$$

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

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

and where the symbol $\neg$ denotes logical negation.

The second technique, proposed by Fonseca and Fleming [50] is mathematically represented by:

$$\begin{aligned} \Phi &: \mathbb{R}^n \longrightarrow \{0, 1, \ldots, \mu - 1\} \\ f(a_i) &\triangleq \sum_{j=1}^{\mu} \chi(f(a_j)\ p\!<\ f(a_i))\ , \end{aligned} \tag{12}$$

where χ (condition) $= 1$ if the condition is true, else 0.

Several Pareto-based selection techniques have been implemented, selecting solutions based directly upon their domination status. These techniques are mathematically represented by:

$$\begin{aligned} \Psi &: P \longrightarrow P' \\ \mathbb{P} &= \{a_i | \forall a_i, a_i \in P_{known}\}\ , \end{aligned} \tag{13}$$

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. Table 10 lists the known "pure" Pareto selection MOEA techniques.

Table 10: Pareto Selection Techniques: Ranking

Approach	Description	Application	Objectives (#)	Chromosome
Thermodynamical Genetic Algorithm (TDGA) [82, 105] (1995)	Simulated annealing concepts used in selection; Attempts to balance population diversity and fitness; Goldberg's [57] ranking	None	(2) Numeric optimization	Binary string
Constrained Optimization by Multi-Objective Genetic Algorithms (COMOGA) [136] (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
Genetic Algorithm running on the INternet (GAIN) [134] (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 [117] (1995)	Integrated with FORMOSA-P; Goldberg's [57] ranking	Pressurized water reactor reload core design	(3) Boron contamination, Discharge burnup; Power peaking	Integer matrices; Genes are fuel assembly layouts, loadings, and orientations
Multiobjective GA (MGA) [93] (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
Multi-Criteria GA [86, 87, 114] (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 [130] (1996)	Goldberg's[57] ranking; GA applied to backpropagation neural network	Spinning production process	(2) Yarn strength; Yarn elongation	Binary string; Genes are neural net inputs
Diploid GA [152] (1996)	Separately minimizes each function, Dominated solutions removed from combined populations	None	(3) Numeric optimization	Implies real values
Parallel Multi-objective GA [2] (1997)	Unspecified Pareto ranking scheme	Pairwise object recognition parameter selection	(3) Histogram distance; Variant set; Histogram area	Unknown
GA [63] (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 [127] (1997)	Goldberg's [57] ranking; Integrated with two local search schemes; Progressive penalties	Water pump scheduling	(2) Energy cost; Pump switches	Binary string

Table 10: (continued)

Approach	Description	Application	Objectives (#)	Chromosome
Pareto Converging GA (PCGA) [84] (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
MOGA [4] (1998)	*Proposes* use of Fonseca's [48] MOGA; Uses simulation to determine performance criteria	Fuzzy logic traffic signal controller	(3) CO emissions; NO_x emissions; Mean travel time	Unknown

5.4.1 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 [60] and Deb [34] 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. Two major MOEA sharing techniques are used; Section 8.2.3 discusses them in detail. Table 11 lists the known Pareto ranking- and niching-based MOEA techniques.

Table 11: Pareto Selection Techniques: Ranking and Niching

Approach	Description	Application	Objectives (#)	Chromosome
Multiple Objective Genetic Algorithm (MOGA) [48, 52] (1993, 1998)	Fonseca's [50] 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) [133] (1994)	Assigns and shares dummy fitnesses in each front; Goldberg's [57] ranking	None	(2) Numeric optimization	Binary string
Niched Pareto GA (NPGA) [71] (1994)	Specialized Pareto domination tournaments	Groundwater contanimant monitoring	(2) Plumes detected; Average volume detected	Binary string; Genes are x, y, z coordinates
Pareto GA [101] (1995)	Uses the NSGA [133]	Electromagnetic absorber design	(2) Absorber layer thickness; Electromagnetic reflection	Binary string; Genes are layer's material type and thickness
NSGA [156] (1996)	Uses the NSGA [133]; Population size of 8,000	Microwave absorber design	(2) Thickness; Reflectance	Unknown
Reduced Pareto Set Algorithm (RPSA) [32] (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) [144] (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

Table 11: (continued)

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [96] (1997)	Parallelized; Integrated with CFD and CEM codes; Uses NSGA [133] with tournament selection	Two-dimensional airfoil design	(2) Drag coefficient; Transverse magnetic radar cross section	Real values
Multiobjective GA [113] (1997)	Uses Fonseca's MOGA [48]; Elitism incorporated; Integrated with Navier-Stokes code	Cascade airfoil design	(3) Pressure rise; Flow turning angle; Pressure loss	Real values
Multi-Objective EP (MOEP) [107] (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 [111] (1998)	Uses Fonseca's [48] MOGA; Integrated with Navier-Stokes and Squire-Young codes	Transonic wing design	(2) Lift; Drag	Real values
Multi-Objective Genetic Programming (MOGP) [68] (1998)	Uses Fonseca's [48] 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	Program
Strength Pareto Evolutionary Algorithm (SPEA) [166] (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

5.4.2 Pareto Deme-Based Selection

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

At one extreme of the island model, where 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 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 imposed upon the population as criteria for selection. Table 12 lists the known Pareto deme-based selection MOEA techniques.

Table 12: Pareto Selection Techniques: Demes

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA with Redundancies [9] (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 [89] (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 [120] (1995)	Local geographic selection via Pareto tournaments; Hybridized; Parallelized	Aerodynamic shape parameterization	(2) Pressure distributions	Integer string and real-valued vector
GA [1] (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

5.4.3 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; the remainder of the population is filled via some other method. This is simply a "twist" on the traditional elitist selection technique. These approaches primarily use a solution's "elite" status as the selection criteria. Table 13 lists the known Pareto elitist-based selection MOEA techniques.

Table 13: Pareto Selection Techniques: Elitist

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
Pareto Optimal Genetic Algorithm [95] (1993)	Pareto optimal solutions selected from efficient set formed by parents and offspring	None	(2) Numeric optimization	Binary string
GENMO [10, 11] (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 [162] (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 [56] (1995)	Design rule-set evolves; Optimizes the inverse problem to obtain attainable criteria set; Next generation formed as per Louis [95]	Beam section topology	(2) Surface area; Moment of inertia	Binary string; Genes are sets of executable rules producing a design

Table 13: (continued)

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [139][4] (1995)	Retains all (or subset of) $P_{current}$	Unknown	Unknown	Unknown
PAReto optimal and Amalgamated induction for DEcision trees (PARADE) [163] (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 [123] (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 [138][5] (1996)	Discards all dominated solutions; Prohibits solution duplication; Population size varies	Unknown	Unknown	Unknown
Multiobjective GA [116] (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

[4]Cited by Tamaki [140]; in Japanese.
[5]Cited by Tamaki [140]; in Japanese.

6 Related MOEA Publications

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.

6.1 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. We include these efforts for completeness, and as a reference for the discussion on MOEA test functions in Section 10.1. Table 14 lists the known efforts comparing different MOEAs.

Table 14: Technique Comparisons

Approach	**Description**	**Application**	**Objectives (#)**	**Chromosome**
GA [91, 67] (1990, 1989)	Compares VEGA [128] and Goldberg's ranking [57]; Specialized crossover	Set covering problem–Scheduling algorithm parameter search	(2) Cost; Violated constraints–(4) Fitness function weights	Binary string
GA [65] (1992)	Weighted sum with chromosomally encoded weights; Compares fitness sharing (applied only to weighting variables) and two VEGA [128] 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
GA [141] (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 [122] (1994)	2 variants: VEGA; Goldberg's [57] ranking	Groundwater pollution contanimant monitoring	(2) System cost; System reliability	Binary string; Genes are operating mode and pumping rate of n wells
Modified Combinatorial ES [83] (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 [24] (1995)	Compares VEGA, Tchebycheff weighting, Pareto ranking, and VEGA-Pareto GA variants	Groundwater monitoring	(2) Undetected plumes; Contaminated area	Fixed-length integer string
Multiple Objective GA [131] (1996)	Compares Fonseca's MOGA [48] to separate weighted-sum runs	Meal production line scheduling	(3) Rejected orders; Batch lateness; Shift/staff balancing	Permutation ordering
Multiobjective GA [28] (1996)	Compares linear combination, "two-branch-" and Pareto domination-tournament	Rotor system design	(2) Rotor system power; Rotor system weight	Binary string

Table 14: (continued)

Approach	Description	Application	Objectives (#)	Chromosome
GA [140] (1996)	4 variants: Parallel selection; Pareto ranking; Tournament selection with sharing; Pareto reservation	None	(2) Numeric optimization	Binary string
MOGAC [35] (1997)	Fonseca's [50] ranking; Implements inverse elitism and dynamic parameter adaptation	Hardware/software co-design	(2) Cost; Power consumption	Unknown
GA [36] (1998)	Compares weighted sum, Goldberg's ranking [57], and Fonseca's MOGA [48]; Specialized EVOPs	"Cell" configuration (constrained facility layout)	(2) Yearly processing; Overall Cost	Integer string; Genes represent the number of "reactors" in the corresponding cell
GA [13] (1998)	Compares six multiobjective ranking methods	None	(2) Numeric optimization	Binary string
MOGA [112, 110] (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
MOGA [167] (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

6.2 Multiobjective EA Theory

Many of the preceding cited efforts at least pay "lip service" to different facets of underlying MOEA theory, but make no significant contribution. These efforts often simply cite relevant issues raised by others. However, some (e.g., Fonseca [52] and Horn [70]) go into significant theoretical detail. Their work provides basic MOEA models and theories, which are addressed in Section 8.2. Other recent papers are focused on MOEA theory, using application examples to illustrate key concepts. We group these related papers here for easy reference. Table 15 lists the known efforts discussing MOEA theory.

Table 15: MOEA Theory

Researcher(s)	Paper Focus
Fonseca and Fleming [49] (1995)	Selection, sharing, and mating parameter values
Fonseca and Fleming [50] (1995)	General Pareto concepts
Fonseca and Fleming [52] (1998)	Goal incorporation; MOEA parameters and values
Horn and Nafpliotis [70] (1995)	Sharing and niching values
Fonseca and Fleming [47] (1997)	MOEA mathematical formulations
Horn [69] (1997)	MOEA-Pareto observations
Rudolph [124] (1998)	MOEA convergence
Van Veldhuizen and Lamont [148] (1998)	MOEA convergence and Pareto terminology

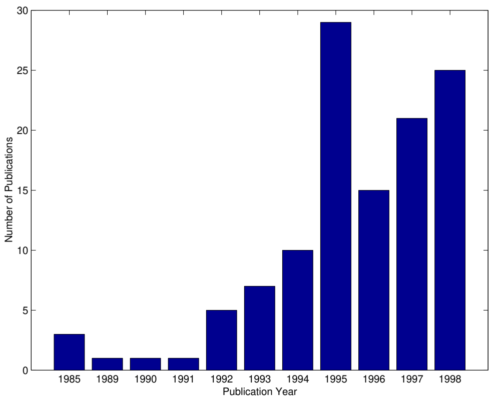

Figure 3: MOEA Citations by Year

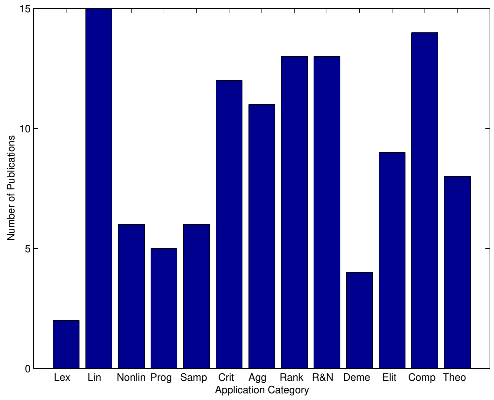

Figure 4: MOEA Citations by Technique

7 MOEA Survey Analysis (Quantitative)

Details of past MOEA research is presented in this section which is concerned primarily with analyzing raw data; Section 8 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, and the fitness functions used in MOEA research.

7.1 Published Papers: Numbers and Dates

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

Two graphs quantifying the cataloged information are presented; Figure 3 shows the number of citations by year and Figure 4 shows the number by technique. We immediately see that the initial transformations of EAs into the multiobjective domain did not spark any real interest for several years. We also note here that although several researchers credit Schaffer with "inventing" the first MOEA, Fourman too deserves credit for the two MOEAs he implemented that same year.

Not until the mid 1990's is a noticeable increase in MOEA research apparent. It is more than an increase, however, as almost three times as many MOEA approaches were published in the last four years (1995-1998) as in the first ten (1985-1994). Thus, the number of publications indicates an active community interest in MOEAs, at least as far as new approaches and applications are concerned. However, MOEA theory has lagged far behind applications (in numbers of papers published). This is even more clear when noting only two of the papers in Table 15 *concentrate* on MOEA theoretical concerns. The others do discuss MOEA theory but only as regarding various parameters of their respective approaches. This quantitative lack of theory is not necessarily bad, but indicates supporting work is necessary to (possibly) increase the effectiveness and efficiency of existing approaches. Section 8.2 discusses this in detail.

Comparing citations by technique highlights the popularity of *a posteriori* techniques. 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" problems, it seems DMs would want 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 that decision without the additional knowledge gained through initial (or interactive) search.

7.2 Technique Discussions

Real estate agents claim that three major factors set the price one can reasonably expect when buying or selling a home. These factors are: 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. This is no different than is the case with single-objective EAs, but bears mentioning.

Several MOEA techniques are currently available. Selection of the *appropriate* technique is dependent upon careful examination of the problem domain. Careful integration of both problem and algorithm domains ensures derived solutions are the best *available*. Identifying MOEA techniques and components which have and have not historically "worked" should improve present and future MOEAs. This section presents general observations about the reported efforts classified in each identified MOEA technique. General comments about each of the three major techniques are given followed by detailed discussions of the classified efforts.

7.2.1 *A Priori* Techniques

By definition, these techniques require objective importance to be defined first, before search occurs. It seems that in scientific and engineering problems of any difficulty this would not be easy to do. The ramifications of "bad" objective prioritization choices are easy to see: the DM's "cost", no matter how defined, could be quite great. Additionally, although we know that EAs are *not* guaranteed to return the global optimum, choosing some arbitrary prioritization may well miss more "acceptable" solutions. No matter what optimization algorithm is used, this is an inescapable consequence of this category of techniques, which we now examine in detail.

Lexicographic techniques have not found favor with MOEA researchers, as only two implementations are reported. This is most likely due to the fact that this technique explores objective space unequally, in the sense that priority is given to solutions in one of the objectives over (an)others, e.g., one objective is optimized at all costs. As evidenced by Figure 4, the greatest body of MOEA research centers upon approaches exploring "equally" in all objective dimensions.

The lexicographic method is most suitable *only* when the importance of each objective (in comparison to the others) is surely known. On one hand, any reported solutions are Pareto optimal (by definition). One the other hand, though, when is such an "all costs" goal is necessary or even appropriate? If one objective is to be optimized at all costs, it seems more appropriate to instead use a single objective EA. That approach does not incur the additional overhead of an MOEA for what may well be marginal returns.

The linear fitness combination technique is a popular approach despite its identified shortfalls. This is probably due to its simplicity. Table 2 reflects its application to many real-world problems, although often incorporated with "variations on a theme." The basic weighted sum MOEA is both easy to understand and implement; the 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 time is short, this may be a suitable method to discover an acceptable MOP solution. However, this method has an identified disadvantage.

Fonseca and Fleming [47] explain that for any positive set of weights, Φ's global optimum is always a nondominated solution. However, if PF_{true} is (partially) concave, optima in that portion of the front can *not* be found via this method. Additionally, they state that the weighted sum approach is the most popular MOEA technique. Figure 4 clearly indicates that over twice as many Pareto-based approaches implemented; thus, that statement is no longer true.

The nonlinear combination techniques seem unpopular with researchers. Of the two cited multiplicative efforts, only one reports actually *implementing* the technique. This is most likely due to the overhead involved in determining appropriate probability of acceptance or utility functions, and also the various conditions the functions must meet [81]. Other MOEA techniques may be as effective and efficient without the overhead required by these nonlinear implementations.

Only one goal programming approach has been incorporated into an MOEA. If a DM is certain of the desired levels in each objective beforehand, no major problems should arise when using this method. However, just as in all *a priori* techniques, specifying exact goals before search begins may unnecessarily limit the search space and miss desirable solutions. On the positive side, overhead is minimal in implementing this technique because the desired goal levels are incorporated into the fitness function. These comments also hold true for the cited minimax techniques. However, using the minimax or goal programming approach to minimize Φ does *not* guarantee the resulting solutions are members of P_{true} [47].

It appears that in general these *a priori* MOEA techniques may be undesirable for use except in isolated situations. If a DM takes the time and trouble to search for a solution, it stands to reason the DM wishes to obtain an optimal solution. Because these techniques limit the search space they appear to have a good chance of not finding optimal solutions. Additionally, as is shown in Sections 8.5 and 9.3, implementing "more" effective and efficient MOEAs might not be as difficult as imagined.

7.2.2 Progressive Techniques

The lack of cited interactive search efforts is surprising. It seems that no matter what particular solution technique is implemented, close interaction between the DM and the searchers can only increase the efficiency of discovered solutions. It is understandable that time is at a premium for some DMs. At least to some level, though, more interaction between DM and searcher(s) implies better results. Although either *a priori* or *a posteriori* techniques may be used interactively, the latter are more desirable because these techniques 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 a greater number of choices vice a single one is more advantageous in most situations.

One particular technique (Fonseca's MOGA [48]) was used in three of the four reported efforts from the literature. It is unclear why other techniques have not been used, as incorporating DM preferences within and through an interactive search-decision making process may benefit all involved. Do researchers and/or practitioners feel they don't have the time for this? 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.

7.2.3 *A Posteriori* Techniques

As indicated in Section 5, these techniques are explicitly seeking P_{true}. A search process is executed and 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 methods are reported but we question their effectiveness. All cited methods use a linear combination technique, where the weights assigned to each objective are varied between a number of separate EA runs. This is effectively identical to a number of single objective EA runs, recording each solution. This technique may have limited utility if only two objectives are being considered, as it may give a *rough* idea of P_{true}. For example, if each objective's weight varies from 0 to 1 by 0.05 increments, only 21 EA runs are necessary to explore the possible weight combinations. However, even at this weight resolution, the required number of runs combinatorially increases with the number of objectives. Thus, its overall usefulness seems quite limited, especially as the arbitrary weight combinations may well prevent discovery of solutions in P_{true} .

Schaffer's VEGA [128] 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 the capability of an MOEA to return a number of possible solutions *within a single run* is seen. Several criterion techniques are faulted for ignoring solutions performing "acceptably" in *all* dimensions for those performing "well" in one [69]. However, even with this known shortfall, several criterion techniques were recently implemented.

Crossley et al. [31, 109] realize that this approach reduces the diversity of any given generation. They implement elitist selection to ensure survival of PF_{known} endpoints during MOEA execution, claiming this ensures survival of solutions which produce designs along PF_{true} . We assume their results were "good enough" for their particular purpose, but question the ability of solutions corresponding to endpoints of PF_{true} to quickly evolve into solutions corresponding to PF_{true} 's interior.

Aggregative selection MOEAs incorporate a variety of specific techniques to optimize an MOP. Table 9 shows weighted sums, constraint and objective combinations, and hybrid search approaches used. However, rather than using the same weight combination for the objectives statically throughout the run, the weights are varied between generation and/or each function evaluation. Sometimes the weights are assigned randomly, sometimes they are functions of the particular solution, and in other cases were encoded in the chromosome where EVOPs acted upon them.

The major advantage of any of these techniques is the *set* of solutions returned by the search process. Thus, P_{known} and PF_{known} may be reasonable approximations to P_{true} and PF_{true} . But these methods are not without their disadvantages. When using the weighted sum technique we know certain members of PF_{known} 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 [165]). Thus, a fitness assignment or selection technique able to 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 the basic technique is suitable for a number of different engineering problem domains. Additionally, rather than the usual two objective functions, several approaches used three, four, or six. Pareto methodology handles this increased number of functions easily; one must assume the additional information given to researchers via these additional functions is worthwhile.

Taken as a whole the Pareto-based techniques are the most popular MOEA type, with 20 more papers published than efforts using linear combination (the next most popular technique). Also, judging merely by the number of published efforts, more interest is evident in either "pure" Pareto ranking, or ranking combined with sharing and niching. The additional overhead of deme-based selection may be a contributing reason for the technique's relative unpopularity. Deme-based selection also adds parameters, but sharing and niching do also; one must assume that factor then has no bearing. Finally, although various elitist methods are proposed, they may not retain diverse enough populations to find and retain a P_{known} truly representative of PF_{true} . Pareto elitist approaches keep only Pareto optimal solutions in the population and discard all others. Once a population contains almost all Pareto optimal solutions the remaining solutions may not provide enough diversity for effective exploration.

7.2.4 MOEA Comparisons

To date, most MOEA researchers' *modus operandi* is an algorithm's comparison (usually the researcher's own new and improved variant) against an older MOEA (typically VEGA, even though it has identified shortfalls [47]), and analyzing results for some MOP (typically Schaffer's F2 [128], or some other numeric example). Results are often "clearly" shown in graphical form, indicating which algorithm's PF_{known} is a better representation of PF_{true} .

Only recently has any researcher proposed experimental methodologies for *general* MOEA comparative analysis [167]; we present an extensive discussion on this subject in Section 10. Most cited efforts simply compared results of two or more MOEAs applied to their particular application. Although this is all well and good, a "sample size of one" is in general undesirable.

7.2.5 MOEA Theory

Seven published papers exist whose major focus is the underlying theoretical analysis of what "true" MOEAs should and should not operate. Three researchers produced most of these papers. Other researchers often cite these works (primarily those by Fonseca and Fleming), but our detailed classifications show their efforts are often modifications of similar techniques, or perhaps the same technique applied to different problems, and add little or nothing to MOEA theoretical underpinnings.

Five papers focus mainly on theoretical analyses of MOEA parameters and behavior, whereas two attempt to further define the nature and limitations of Pareto optimality, and the subsequent effects upon MOEA search. Rudolph especially focuses on the limits of MOEA search [124], while Van Veldhuizen and Lamont attempt to further define the Pareto optimal problem domain [148] (see also Section 10.2.1).

7.3 MOEA Fitness Functions

The cataloged research efforts provide various types of fitness functions used by MOEAs. Table 7.3 lists generic fitness function types, their identifying characteristics, and examples of each. The listed types are not limited to MOEA appli-

Table 16: MOEA Fitness Function Types

Category	Characteristic	Examples
Electromagnetic	Energy transfer or reflection	[100, 103, 149]
Economic	Production growth	[63, 131]
Entropy	Information content and (dis)order	[51, 84, 123]
Environmental	Environmental benefit or damage	[3, 24, 146]
Financial	Direct monetary (or other) cost	[5, 72, 150]
Geometrical	Structural relationships	[37, 54, 79]
Physical (Energy)	Energy emission or transfer	[80, 116, 158]
Physical (Force)	Exerted force or pressure	[30, 113, 151]
Resources	Resource levels or usage	[9, 35, 131]
Temporal	Timing relationships	[48, 76, 131]

cations, nor are they the only ones possible. However, MOEAs offer the exciting possibility of simultaneously employing different fitness functions to capture desirable characteristics of the current problem domain. This is true regardless of MOEA technique.

Function types appear limited only by the practitioner's imagination. Several fitness function types are identified; 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 cited authors that their particular MOEA implementations were successful imply their associated fitness functions were appropriate for the given problems.

Finally, the presented efforts also clearly show the non-commensurability and independence of many fitness function types. These are the factors responsible for the partial ordering of the search space.

7.4 MOEA Chromosomal Representations

Theorems exist [46] showing no intrinsic advantage exists in any given genetic representation. For any particular encoding and its associated cardinality, *equivalent* evolutionary algorithms (in an input/output sense) can be generated *for each individual problem instance*. Although particular gene representations and operators may be more desirable than others in certain situations, the theorems show no choice of representation or evolutionary operators operating on one or two parents, offers any capability which can't be duplicated by another representation and/or operator.

Thus, the genetic representation is another MOEA component limited only by the implementor's imagination. The cited efforts show the most common representation is a binary string having 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, several other representations are more intricate, and therefore of more interest here.

Some MOEAs employ arrays as genome constructs. For example, Baita uses a matrix representation to store recessive information [9]. (As a side note, only two published MOEAs use dominant and recessive genetic information [9, 88]). Parks and Chow use matrices as they are more natural representations of their respective problem domains' decision variables [117, 23].

The Prüfer encoding used by Gen [55] uniquely encodes a graph's spanning tree and allows easy repair of any illegal chromosome. In the only two known multiobjective Genetic Programming implementations [89, 68], a program/program tree representation was used. But whatever representation used, we again see any claims of successful MOEA implementations imply the associated genetic encodings are appropriate for the given problems.

7.5 MOEA Problem Domains

By definition, MOEAs operate on MOPs. A more theoretical discussion of this domain is given in Section 10.2.1. Here, we discuss the domain in more general terms. When implementing an MOEA, it is (implicitly) assumed that the problem domain (fitness landscape) has been examined, and a decision made that an MOEA is the most appropriate search tool for the given MOP. In general, it is accepted that EAs are useful search algorithms when the problem domain is multidimensional (many decision variables), and/or the search space is very large. Most cited efforts appear to exhibit these characteristics.

Overall, almost 85% of the cited efforts were applied to non-pedagogical problems. This indicates EA practitioners are today developing and implementing MOEAs as real-world search tools. These implementations span several disparate scientific and engineering research areas, giving credibility to the MOEA's claim as an effective and efficient general purpose search tool.

8 MOEA Survey Analysis (Empirical)

This section presents issues of a more philosophical nature raised by the preceding discussion. Although not quantitatively derived, our observations are based on the preceding cataloged efforts, and our conclusions substantiated with other relevant citations from the literature.

8.1 MOEA Characteristics

What "makes" an MOEA different from an EA? How "hard" is it to use an MOEA? When should an MOEA be used? We present our views on these issues in this section.

The major MOEA defining characteristic, of course, is the multiple objectives being simultaneously optimized. Otherwise, a task decomposition clearly shows little structural difference between the MOEA and its single objective counterparts. Figures 5 and 6 show a general EA's and MOEA's task decomposition, respectively.

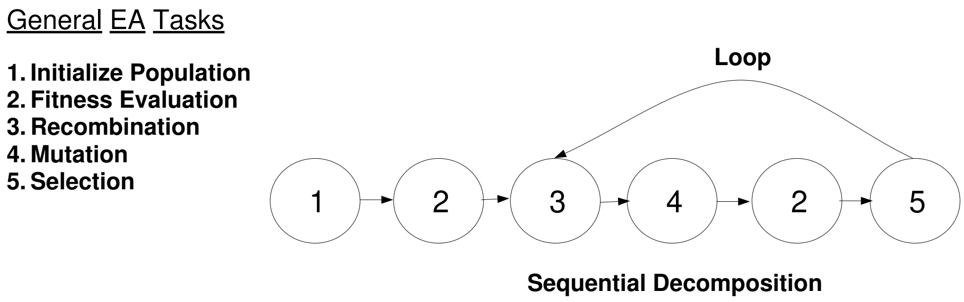

Figure 5: Generalized EA Task Decomposition

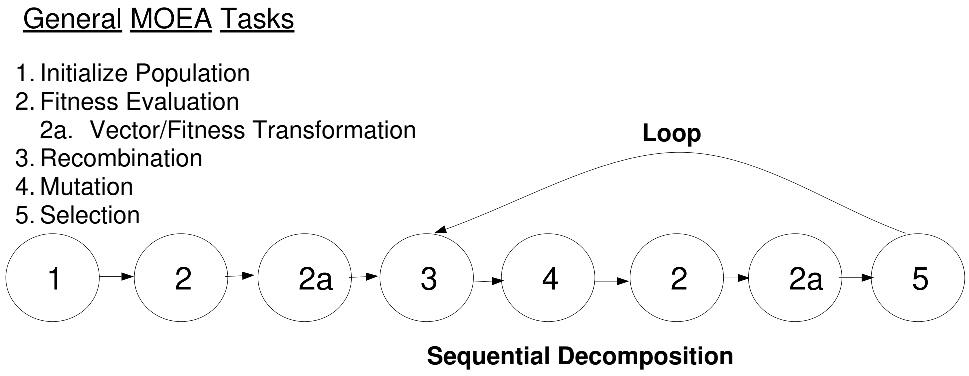

Figure 6: MOEA Task Decomposition

The major differences are noted as follows. By definition, Task 2 in the MOEA case computes k (where $k \geq 2$) fitness functions. Also, because EAs 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 8.4), the remainder of the MOEA is structurally identical to its single-objective counterpart.

A conference reviewer once called a particular MOEA implementation "straightforward;" it was also evident the reviewer did not completely understand relevant MOP concepts. Conversations with other MOEA researchers echo that sentiment. Much time and effort is expended defining and defending MOP concepts 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. The remainder of this section provides a basis for this statement.

This report has detailed 12 different MOEA solution techniques (Sections 3 through 5). Although the cited efforts explain *how* various MOEAs are implemented, they do not always explain *why*. This may well be due to the current lack of MOEA theory.

8.2 MOEA Theory

We agree with other MOEA researchers [69, 50] that MOEA theory is lagging behind the applications being continually implemented. We showed in Sections 3 through 5 that although a steady stream of MOEA publications has ensued since the middle 1990's, this increased number of conference papers and journal articles does not indicate a corresponding *depth* of associated theory, clearly indicated by Table 15. This report makes clear the fact 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 approaches.

Why is there such a lack of underlying MOEA theory? Although some mathematical foundation of MOEA theory exists, the current situation seems akin to Goldberg's [58] recent comparisons of engineer and algorithmist. Here, he likens algorithms to "conceptual machines", and implies that computer scientists are hesitant to move forward without exact models exactly describing the situation. He claims the design engineer often accepts less accurate models in order to build his design. MOEA researchers certainly seem to have taken this approach! Or, is it possible the lack of theory is due to the fact that many current practitioners do not have the requisite mathematical knowledge; that constructing appropriate mathematical models is beyond their current expertise?

Basic EA theory is well-established. A *Handbook of Evolutionary Computation* [8] exists with entire chapters devoted to the theoretical foundations of evolutionary computation established during the past 25 years; sample topics include selection, representation, crossover, mutation, fitness landscapes, etc. Several foundational textbooks are also available, such as those by Goldberg [57], Michalewicz [98], and by Bäck [6]. Although much of this theory is (may be?)

valid as regarding MOEAs, some is not. Thus, this section discusses the current state of MOEA theory.

8.2.1 Fitness Functions

The overwhelming majority of published MOEAs use only two fitness functions. Some use three, and a few use six, but the maximum of any cited effort is seven different functions; both Wienke et al. [158] and Fonseca and Fleming [51] used this number of functions.

However, Wienke et al. (in effect) used seven copies of the same objective function. Their objectives were to meet certain atomic emission intensity goals for seven different elements. However, although the elements are different the fitness functions are conceptually identical. This does not make the MOP "easier", but perhaps makes the MOP easier to understand.

On the other hand, Fonseca and Fleming's MOP's objectives appear both incommensurable and independent. Both P_{known} and PF_{known} are hard to visualize, especially in their effects upon each other. For example, considering the polynomial model built 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.

How many fitness functions are enough? How many objectives are required to adequately capture an MOP's essential characteristics? The cataloged efforts imply most MOPs (whether numeric or real-world) require only two or three objectives. There is some practical limit as the time to compute several complex fitness functions quickly becomes unmanageable. As far as Pareto optimality is concerned, a theoretical limit also exists.

As more objectives are added to the MOP more and more solutions meet the definition of Pareto optimality. Thus, as Fonseca and Fleming indicate for *pure* Pareto MOEAs [50], the size of $P_{current}$, $PF_{current}$, P_{known} , and PF_{known} grows, and Pareto selective pressure decreases. However, both they and Horn [69] err when they imply that the size of PF_{true} grows with additional objectives. We prove in Section 10.2.1 that the Pareto front of an MOP with two objectives ($k = 2$) is at most a curve, and is at most a hypersurface when $k \geq 3$. And, as Cantor proved [64] the infinity of points on a line, surface, cube, or so on are the same ($\aleph_1$). Thus, the size of PF_{true} does *not* grow with the number of objectives. However, since MOEAs deal with a discrete Pareto front representation, the number of possible solutions and certainly the number of points on the front may increase as more objectives are added. However, this number of computationally representable points is dependent more on the front's shape than on the number of objectives.

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 [102] seminal paper proposing the human's one-dimensional span of judgment and immediate memory to be 7 ± 2. He notes that adding objective dimensions increases this capacity but at a decreasing rate. This argues a "more the merrier" viewpoint for the number of objectives, but visualizing and understanding objective inter-relationships becomes more difficult as the number of objectives grow. Thus, techniques were developed to map high-dimensional information to two- or three dimensions for better understanding (e.g., Sammon mapping [126] and trade-off graphs such as those used by Fonseca and Fleming [52]).

Past MOEA implementations imply two or three objectives are enough for most problems. Thus, at the very least, MOEA application to a given MOP should begin with two or three objectives to gain problem domain understanding. One may be able to ascertain how the different objectives affect the others and also some idea of the fitness landscape. Other fitness functions may then be added as deemed necessary to capture other relevant problem characteristics. Section 7.3 identifies several categories of fitness functions for this purpose.

8.2.2 Pareto Ranking

As noted in Section 5.4, one of two Pareto fitness assignments methods is normally used in MOEAs. In general, both assign preferred solutions the same rank and other solutions a higher (less desirable) rank. With the scheme proposed by Goldberg [57], where a solution $\mathbf{x}$ at generation t has a corresponding objective vector $\mathbf{x_u}$, the solution's rank is defined by the algorithm in Figure 7.

```
curr_rank = 1
For i = 1 : n do
    If x_u is nondominated
        rank(x, t) = curr_rank
od
For i = 1 : n do
    If rank(x, t) = curr_rank
        Store x in temporary population
od
curr_rank = curr_rank + 1
```

Figure 7: Rank Assignment Algorithm

The second technique, proposed by Fonseca [50], operates somewhat differently. As before, a solution $\mathbf{x}$ at generation t has a corresponding objective vector $\mathbf{x_u}$. We also let $r_u^{(t)}$ signify the number of vectors associated with the current population dominating $\mathbf{x_u}$; $\mathbf{x}$'s rank is then defined by:

$$\text{rank}(x, t) = r_u^{(t)} \; . \tag{14}$$

This ensures all nondominated solutions receive rank zero.

It appears neither ranking method has any intrinsic value over the other, although it was mentioned in the literature that Fonseca's method, which effectively assigns a cost value to each solution, might be easier to mathematically analyze [47]. Horn [69] also notes Fonseca's ranking can determine more ranks (is finer-grained) than Goldberg's. There is currently no clear evidence as to the benefit(s) of one ranking scheme over the other. No experiments have been reported in the literature directly comparing the two schemes. Thus, at least for now, it appears the choice of rank assignment scheme is left to the preference of individual researchers.

8.2.3 Pareto Niching and Fitness Sharing

Several MOEA Pareto niching and fitness sharing variants have been proposed. Their goal is the same as in the traditional single-objective optimization sense: finding and maintaining *all* optima so that a "close" representation of PF_{true} results; the MOP case uses sharing to attempt and find a uniform distribution *representing* PF_{true} .

Fonseca and Fleming [52] perform sharing in the normal manner *except* that it occurs only between solutions with *identical* Pareto rank. They also measure 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 result is less than σ_{share} the solution's associated niche count is adjusted.

The Nondominated Sorting Genetic Algorithm (NSGA) [133] of Srinivas and Deb implements a slightly different variant. They measure distance (over some norm) in genotypic space, i.e., the distance between two *solutions* is compared to σ_{share}.

Horn slightly modifies the traditional niching definition but is still aspiring to more closely approximate PF_{true} . His and Nafpliotis' Niched Pareto Genetic Algorithm (NPGA) [70] 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, in the case both solutions are nondominated or both are dominated, fitness sharing occurs. Each of the two solution's niche counts is computed not by summing the computed sharing values but by simply counting the number of solutions whose associated vectors are within σ_{share} in phenotype space. The solution with a smaller niche count (i.e., fewer neighbors) is then selected. Horn et al. terms this *equivalence class sharing* [71] because solutions within the various classes can be considered "equally" fit.

Other variants also exist. For example, one uses the NSGA's rank assignment scheme (i.e., Goldberg's [57] Pareto ranking) and fitness sharing within the phenotypic domain [101], and another uses a method combining both genotypic and phenotypic distances [123]. Fitness sharing may also be applied to solutions *regardless* of their rank.

Any of these methods requires setting explicit values for the key sharing parameter σ_{share} (i.e., the niche radius), which can affect MOEA efficiency and effectiveness. Fitness sharing's performance is sensitive to the setting of both σ_{share} and the population size N. Assigning this parameter's values is difficult, as it usually requires some *a priori* knowledge about the shape and separation of a given problem's niches. However, both Fonseca and Fleming [52] and Horn [70] give some bounding guidelines of appropriate σ_{share} values for MOPs. Horn also indicated that the NPGA's tournament size parameter (t_{dom}) affects algorithm effectiveness and suggested appropriate values for it.

Both methods of setting σ_{share} involve the number of individuals in the population, scaling the known attribute values, and determining the minimum/maximum value in each objective dimension. For now, we assume the population size is static throughout MOEA execution. Then, attribute value scaling attempts to reduce niching bias resulting from vastly different attribute ranges. This leaves determination of the min and max attribute values in each dimension.

Simply computing attribute values for the minimum and maximum of associated decision variables does not work; decision variable minimums/maximums may not correspond to attribute minimum/maximums. Thus, the min and max values of either the current generation or secondary population may be used. Fonseca and Fleming [52] indicate using the values present at each generation yield good results. We note that the stochastic nature of MOEAs may not preserve these values' associated solutions between generations. It seems more intuitive to choose the values from the secondary population which by definition contains the min and max values found *so far*. This action continually "pushes" the ends of PF_{known} .

However, we once again currently see no clear evidence as to the benefit(s) of one Pareto niching and sharing variant over another. No experiments are reported in the literature comparing any of these variants, and it thus appears the choice is left to the preference of individual researchers.

8.2.4 Mating Restriction

The idea of restricted mating is not a new one. Goldberg [57] first mentions using restricted mating to prevent or minimize "low-performance offspring (lethals)." In other words, it is a method for biasing how solutions are paired for recombination in the hopes of increasing algorithm effectiveness and efficiency. Goldberg presented an example using genotypic structure as mating criteria. Parallel GAs often implement restricted mating in a geographic sense; here solutions may mate only with neighbors within some restricted topology [20]. Several MOEAs employ restricted mating in this sense.

For example, Baita et al. [9], and Loughlin and Ranjithan [94] place solutions on a grid and restrict the possible area within which a solution may mate. Lis and Eiben [92] allow mating only between solutions of different "sexes." Jakob et al. [79] restrict mating to within a solution's particular deme. A unique form of mating restriction is implemented by Hajela and Lin [65]. In their linear fitness combination (weighted-sum) MOP formulation, they apply restricted mating based on a solution's associated weighting variables, preventing crossover between designs with radically different weight combinations.

When considering general MOEAs however, phenotypic-based restricted mating is of more interest to us. More to the point, several MOEA researchers state in their published reports [48, 49, 167]:

> "Following the common practice of setting $\sigma_{mate} = \sigma_{share}$, ..."

This may indeed be a common practice, but how did it get that way? It is an intuitive choice but little rationale has been offered supporting it. In MOEAs, restricted mating is meant to reduce the formation of unfit (e.g., dominated) offspring as the population distributes itself along PF_{known} . Thus, setting $\sigma_{mate} = \sigma_{share}$ seems reasonable, but is the extra algorithmic overhead of restricted mating worthwhile? We currently have only empirical explanations offered for the implementation (or lack of) restricted mating. Obviously, some researchers believe it is necessary in their particular application or they wouldn't have bothered with it, but others indicate it is of no value!

Zitzler and Thiele [167] indicate that for several different values of σ_{mate} , no improvements were noted for their test problems when compared against the cases with no mating restriction. Shaw and Fleming [131] report the same qualitative results for their application when mating restriction was or was not incorporated. Horn et al. [71] offer empirical evidence directly contradicting the basis for mating restriction; they note that recombining solutions whose associated vectors are on different parts of PF_{known} *can* produce offspring whose vectors are on PF_{known} between their parents! They also claim that for a particular MOP, PF_{known} is maintained by a constant (re)generation of points on PF_{known} through recombination of two different parents. They believe most recombinations of solutions in P_{known} also yield solutions in P_{known} .

Yet again we have no quantitative evidence as to the benefits of this MOEA theory component. The empirical evidence presented in the literature can be interpreted as an argument for *or* against this recombination form and leaves the MOEA field in an unsatisfactory predicament. If mating restriction is indeed beneficial it should be implemented. However, this adds algorithmic complexity to the MOEA. Section 8.4 shows MOEAs may be $\mathcal{O}(n^2)$ algorithms. Any additional algorithmic complexity should be *known* to be worthwhile. The issue of mating restriction would clearly benefit from experiments directly comparing its inclusion/exclusion from MOEAs.

8.2.5 Solution Stability or Robustness

Whether single or multiobjective, EAs seek the global optima of some problem. At least for MOPs, it has been noted [74] that P_{true} may not, and often *is not*, the most desirable solution set. because it is "unstable" (due to engineering tolerances, nonlinear response, etc.). It is 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).

8.3 MOEA Secondary Populations

We agree with Horn [69] and others that any practical MOEA implementation includes a secondary population of all nondominated solutions found so far during an MOEA run. This is due to the stochastic nature of EAs which does not guarantee that a desirable solution, once found, remains in the population until EA termination. However, what exactly should be done with this population? Is it simply a repository, continually added to and then culled of dominated solutions? Or is it integrated as an active component of the MOEA search process? Although several researchers indicate their use of secondary populations, only a few explain its exact use in their particular MOEA. As there is no consensus for its "best" use, we discuss the various cited implementations of MOEA secondary populations.

A straightforward implementation of a secondary population stores $P_{current}$ at the end of each generation; these solutions are termed P_{known} . This set must be periodically culled of dominated solutions as a solution's designation as Pareto optimal is *always* dependent upon the set within which it is evaluated. How often the set is updated is 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 n grows comparison time may become significant. When using this method no feedback of solutions from P_{known} into the main population is performed.

Conversely, other published algorithms actively involve P_{known} in MOEA operation. For example, at the end of each generation Zitzler and Thiele's [166] Strength Pareto Evolutionary Algorithm (SPEA) stores $P_{current}$ in a secondary population (P_{known}) and then culls dominated solutions. Solutions from both the main population and P_{known} then participate in binary tournaments selecting the next generation. If the number of solutions in P_{known} exceeds a given maximum, the population is reduced by clustering; the goal is generating a representative subset while maintaining the characteristics of the original set. SPEA also uses P_{known} in computing fitness of solutions in the main population.

Todd and Sen [144] insert nondominated solutions from P_{known} into the mating population to maintain diversity, as do Ishibuchi and Murata [76, 78, 77] and Cieniawski et al. [24]. However, these implementations never reduce the size of P_{known} except to remove dominated solutions. Parks and Miller [116] and Parks [117] also implement an *archive* of Pareto optimal solutions. However, solutions in $P_{current}$ are not always added and archiving occurs only if a solution is sufficiently "dissimilar" from those already present (in terms of their reactivity distributions). This also is a form of clustering. If a new solution is added any archive members which are now dominated are removed. They also select next generation members from both P_{known} and the current generation, choosing some fraction from each.

Some researchers use secondary populations *not* composed of Pareto optimal solutions. For example, Bhanu and Lee [14] 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. [152] use a GA to optimize each of the k MOP functions independently; these "additional" populations are later combined and nondominated solutions removed.

We state 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 much information is better than too little. However, once P_{known} exists it intuitively seems that it might also be useful in adding diversity to the current generation and in expanding the "ends" of the known front. How to effectively and efficiently use P_{known} is still unknown. This MOEA issue would also clearly benefit from experiments directly comparing its various implementations.

8.4 MOEA Complexity

It is well known that the time complexity of fitness function evaluation (of real-world problems) dominates EA execution time. Thus, when discussing the algorithmic complexity of various MOEAs, we are concerned mainly about the number of fitness evaluations. We do consider some solution comparisons and calculations, as this is additional overhead missing from simple GA (SGA) implementations. However, EVOP complexity is ignored for our current purposes.

MOEA complexity is generally greater than that of SGAs. In an SGA, after fitness evaluation, the 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, adding additional algorithmic complexity. As a reference we present the complexity of the various MOEA techniques in Table 8.4; SGA complexity is included for comparison. The "worst-case" was used to generate these figures.

The notation in the table is as follows. Population size is denoted by n and the number of generations by G. The number of fitness functions is designated by k and m represents the number of solutions per processor (the Pareto demes case). 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 the computational cost of selection inconsequential. All listed techniques have the identical basic cost of $T_f G(nk)$ fitness computations per generation, and then diverge based on their implementations. Finally, a linear combination technique was assumed for the independent sampling technique, and randomly assigned weights (in the fitness functions) for the aggregation technique. Table 8.4 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 Pareto optimality.

Table 17: MOEA Algorithmic Complexity

MOEA Technique	**Computational Complexity**
SGA	$T_f Gn$
Lexicographic	$T_f Gnk + Gn^2k - Gnk$
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	$T_f Gnk + Gn^2k - Gnk$
Pareto Niche and Share	$T_f Gnk + Gn^2k - Gnk + n^2$
Pareto Demes	$T_f Gnk + G\frac{m^2k}{n^2} - G\frac{mk}{n} + \frac{m}{n}T_{comm}$
Pareto Elitist	$T_f Gnk + Gn^2k - Gnk$

MOEA storage requirements are problem dependent. Like other EAs these requirements are mandated by the data structures used in specific implementations. Although required storage increases linearly with the number of fitness functions used, a dramatic increase occurs when a secondary population is brought into play.

8.5 MOEAs for Beginners: Which are Appropriate?

We cannot specify an "all-around" MOEA technique, nor does the NFL theorem [161] allow for one. As far as the implementations we cite it appears that the *a priori* techniques are *not* appropriate. This conclusion is mainly based on the number of published reports. This technique's lack of popularity may be attributed to the lack of information a DM has before search occurs. Without the knowledge additional search brings, a DM is somewhat "shooting in the dark." In cases where stakes are high, the DM is probably better off using an interactive decision making process, or making decisions after a more unconstrained search occurs. However, this does not imply the interactive or *a posteriori* techniques are always appropriate, either.

The tables in Sections 3-5 present numerous approaches. Based on that information, those wishing to implement an MOEA may well be asking "Where do I begin?" Based on the preceding presentation, we suggest three MOEAs as initial candidates for the following reasons.

We showed in a previous work [148] that the global optimum of an MOP is the Pareto front determined by evaluating each member of the Pareto optimal solution set. Noted in Section 5 is the fact that many approaches are explicitly seeking P_{known} . And as this section implies, algorithms incorporating as much known MOEA theory as possible would appear to be rational choices.

Thus, the three suggested MOEAs are Fonseca and Fleming's MOGA [52], the NPGA [70], and the NSGA [132]. These algorithms are fairly recent and stand out because of their incorporation of known MOEA theory. Although each author (rightly so) points out deficiencies in their own and the other two algorithms, any algorithmic approach is bound to have some shortfalls (c.f., the NFL theorem [161]). The common theme of these algorithms is their understanding and addressing of many known relevant theoretical issues, and their empirical success in both numerical MOPs and real-world applications.

The citations give ample information to implement the algorithms; many existing EA implementations can be easily modified to incorporate MOEA specific routines. For example, we have used the messy GA [59], GENOCOP III [97], and the Genetic and Evolutionary Algorithm Toolbox (GEAT) for use with MATLAB [119] to create various MOEAs. A methodology which may quantitatively show these algorithm's strengths is detailed in Section 10.

9 Parallel MOEAs

We have noted several parallel MOEA implementations [79, 9, 120, 123, 96, 2]. These implementations execute either several MOEAs on different processors or spread the MOEA population among processors in a normal demic manner. None discuss what other parallel MOEA possibilities exist. This section presents detailed analysis of these possibilities.

We first assume the Evolutionary Operators (EVOPs) used by an EA are computationally inconsequential when compared to communication costs. This assumption implies EVOPs should operate on entire populations, prohibiting excessive communication cost due to sending/receiving fractions of a population[6].

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 5 executes for varying time periods. Additionally, one of the tasks (Task 1) executes only once. It is easy to see the proposed mapping's inefficiency. One processor completes its task and then sits idle. The other processors are also unable to operate in parallel which results in a much greater idle than calculation time.

The four steps in the loop *must* occur sequentially. Mutation cannot operate until recombination finishes. Selection cannot (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 processes 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 high communication overhead associated with this implementation. We thus draw the conclusion that this implementation is not useful. "Pipelining" the algorithm tasks is also not effective, because it is a special case of the exact task to processor mapping.

Another possible implementation is a "Single Program – Multiple Data" (SPMD) type. Because MOEAs are a stochastic search algorithm, they are not guaranteed to return *the* global optimum. Thus, one could execute several MOEAs simultaneously on different processors and compare/contrast the reported results. As many practitioners execute a number of EAs sequentially to achieve this same result the parallel implementation's speedup is apparent. However, we are also interested in parallelizing MOEA tasks. This background enables us to discuss these possibilities.

9.1 MOEA Task Decomposition

Although several parallel implementations are possible, we only discuss relevant issues affecting selected implementations' effectiveness. We first note the major differences between single- and multiple-objective EAs. By definition, Task 2 in

[6]We realize some parallel architectures may be optimized for "small" message passing, but we are here considering the more general case.

the MOEA case computes k (where $k \geq 2$) fitness functions. Also, because EAs expect a *single* fitness value, additional processing is sometimes required to transform MOEA solutions' fitness *vectors* into a scalar. We have identified several variants of MOEA fitness assignment and selection techniques (e.g., ordering, scalarization, independent sampling, and cooperative search). These techniques vary in their algorithmic impact, while the remainder of the MOEA is essentially the same as its single-objective counterpart.

We note here that affecting the ability to effectively and efficiently parallelize either the EA or the MOEA is the fact that the majority of each algorithm is inherently sequential. Thus, these algorithms seem to fit the definition of *P-Complete* [145], implying only approximations to the global optimum can be determined in polynomial time.

9.1.1 Decomposing MOEA Fitness Assignment and Transformation

As in single-objective EAs, the steps within the major loop must occur sequentially. Thus, an exact task to processor mapping is undesirable as is pipelining. Simultaneously executing several MOEA implementations on different processors is perfectly acceptable, but we are also interested in parallelizing components of a single MOEA run.

As previously noted, MOEAs are suited for search in large, high-dimensional solution spaces such as those found in real-world scientific and engineering problems. And in fact, many MOEAs are applied to these problems in tasks such as airfoil design, nuclear reactor core loading optimization, and electromagnetic antenna design. Two aspects of these real-world MOPs lend themselves to a parallel approach: fitness evaluation and transformation.

MOEAs have two or more functions to be evaluated *per solution*, permitting a parallel approach 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 are shown in Figure 8; each option is discussed in turn.

Execution time for each fitness function may be radically different. Blindly assigning the entire population and each of the k functions to a different processor may not then be prudent, as one calculation could take many times longer than the others (see Figure 8a). One could "load balance" these fitness computations but the effort expended to do so may not be worthwhile. It is also possible to assign fractions of the population to different processors, as identical numbers of individuals are evaluated via identical fitness functions (see Figure 8b). As long as communication time is not a significant fraction of the subpopulation's calculation time, this is an effective parallelization method for the fitness *evaluation* task. Finally, in the case of an extremely expensive fitness computation, each individual's fitness computation(s) could be split among processors (see Figure 8c). This is likely only in domains such as electromagnetic component design where such parallel codes already exist. One final possibility is partitioning the solu-

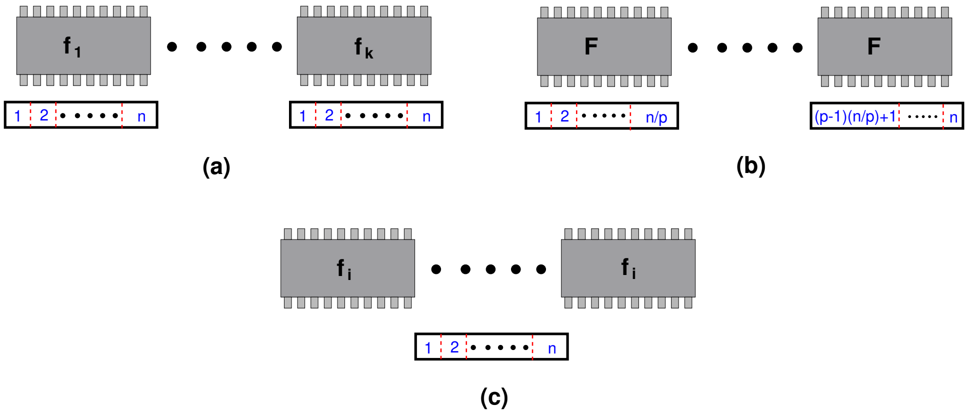

Figure 8: Parallel Fitness Evaluation Possibilities

tion space and searching each portion independently of the others (i.e., multiple MOEA runs searching distinct solution domains). Because Pareto dominance is a transitive relation, results from the independent searches can be combined and compared again, resulting in PF_{known} . Within each portion, the fitness function calculations may also be parallelized using any of the preceding methods.

It is commonly accepted among EA researchers that the most computationally expensive algorithm component is the fitness function evaluation. Considering most EAs execute from a few thousand to tens of thousands of evaluations per run, parallelizing the objective function calculations can significantly decrease total MOEA execution time.

Fitness transformation also offers parallelization possibilities. For example, using Pareto optimality as an MOP solution criterion limits how an MOEA can be parallelized. Many MOEA schemes use Pareto concepts to assign solution fitness, e.g., *Pareto ranking*. This method assigns each solution a fitness based on the number of evaluated vectors dominating its own; one can identify many MOEA "flavors" based on the technique used for fitness assignment and/or selection. As algorithmic complexity is an additional indicator of which MOEA techniques may or may not be good candidates for parallelization, this component must be considered in any proposed implementation (see Table 8.4 for the complexity of all known techniques employed in published MOEAs).

Although several fitness assignment and selection techniques are listed we consider only the Pareto schemes. The other techniques' ability to find members of PF_{true} is limited; each has identified flaws. Whereas most EAs are $\mathcal{O}(n)$, Table 8.4 shows MOEAs using Pareto schemes are $\mathcal{O}(n^2)$ because each solution must be compared against every other. With this fitness assignment scheme, individual fitness (within a subpopulation) can *not* be transformed in parallel as each individual's resultant ranking depends on *every* other solution in the current generation. Parallelizing the Pareto fitness assignment/transformation tasks thus appears worthwhile.

9.2 MOEA Parallelization

Table 8.4 identifies the four major MOEA Pareto variants; we consider only the "Niching and Sharing" variant here. This technique incorporates most existing MOEA theory and thus appears to be the best choice. Figure 9 is the pseudocode for a serial sharing and niching MOEA; each step is annotated with the appropriate task from Figure 6. The Pareto ranking, and sharing and niching actions, constitute the MOEA's fitness transformation.

```
t = 0 (Generation Number)
Randomly Generate Initial Population (Task 1)
Evaluate Each Individual's Fitness Vector (Task 2)
Pareto Rank Each Individual's Fitness Vector (Task 2a)
Assign Fitness via Sharing and Niching(Task 2a)
While t < Maximum Number of Generations do
    Recombine Some Individuals (Task 3)
    Mutate Some Individuals (Task 4)
    Evaluate Each Individual's Fitness Vector (Task 2)
    Pareto Rank Each Individual's Fitness Vector (Task 2a)
    Assign Fitness via Sharing and Niching (Task 2a)
    Select Individuals for Next Generation (Task 5)
    t = t + 1
od
```

Figure 9: Serial Sharing & Niching MOEA Pseudo-Code

The fitness evaluation and transformation tasks (Tasks 2 and 2a in Figure 6) are the most likely candidates for parallelization. We thus assign all other tasks *and* overall algorithm control to a single processor (a master processor). For the moment, we ignore parallelizing the fitness function evaluation itself as that is application problem dependent. However, we may compute the sharing and niching values independently.

9.2.1 Parallel Sharing and Niching

Sharing and niching are processes whereby an individual's fitness is degraded by a quantity proportional to the number of individuals within some specified distance (over some norm) of it. In MOEAs, this technique's intent is to uniformly spread the population over, and expand the ends, of PF_{known} . In general, niching tends to distribute individuals around the best optima and sharing models individual competition for finite resources within a closed environment [52]. Sharing and niching can be thought of as a fitness modification. The process occurs in the following manner. First, sharing values are derived via the following equation:

$$sh(d_{i,j}) = \begin{cases} 1 - (\frac{d_{i,j}}{\sigma_{sh}})^{\alpha_{sh}} & \text{if } d_{i,j} < \sigma_{sh} \\ 0 & \text{otherwise} \end{cases}$$

The parameters α_{sh} and σ_{sh} are user-defined. They are normally chosen to reflect some fitness landscape knowledge or other certain objectives; our concern here is not with these values specifically but how they are used. Distance is computed via some relevant measure; we use Euclidean distance of vectors in the objective domain. Next, for each individual i, the niche count m_i is calculated by:

$$m_i = \sum_{j=1}^{N} sh(d_{i,j}), \tag{15}$$

where N is the population size. Finally, the shared fitness of each individual i is given by:

$$f_{sh,i} \triangleq \frac{f_i}{m_i} \tag{16}$$

We consider Fonseca and Fleming's MOEA [52] as an example, where each solution is given a rank equal to the number of vectors dominating its associated vector. An initial fitness is then assigned by linearly interpolating between the best and worst individual. Niching, sharing, and shared fitness values are then computed.

We see that the Pareto ranking and fitness assignment are *independent* of the sharing and niching value computations (if sharing is not restricted to individuals with the same rank). After fitness function evaluations are completed, the Pareto ranking and fitness assignment may occur using a copy of the fitness vectors. Another copy of these vectors may be used to compute the sharing and niching values. However, Fonseca's MOEA and other variants compute niche counts only between individuals with the same rank, a process requiring the current Pareto rank of each solution. However, even with this restriction the sharing value matrix can be computed independently. Shared fitness is computed after the separate ranking and sharing/niching calculations are finished. As Pareto ranking, sharing, and niching are each $\mathcal{O}(n^2)$ algorithms (described in Section 9.3.2), performing as much of calculation in parallel as possible speeds up overall execution. Of course, an optimal mapping of tasks to processors is dependent on the *specific* problem. We are now able to consider integrating the problem and algorithm domains.

9.3 MOP and MOEA Domain Integration

Here we use numeric MOPs as the problem domain. This is not wasted effort as the fitness function for any real-world problem must be mathematically represented for MOEA use. Thus, identifying numeric MOP characteristics lays the groundwork for application to more complex scientific and engineering problems.

9.3.1 Integration Mechanics

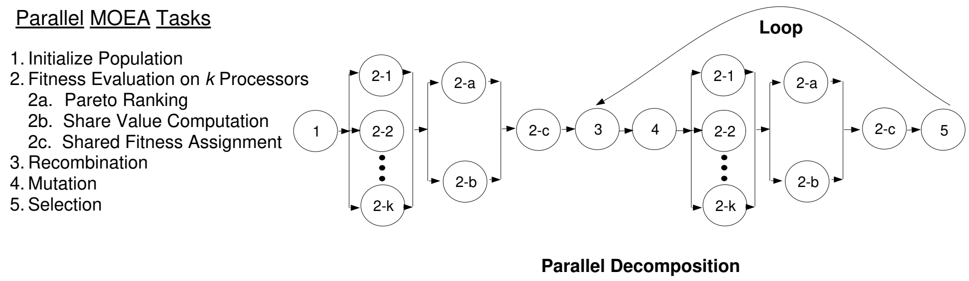

Figure 10: Parallel MOEA Task Decomposition

Figure 10 shows the proposed 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. This parallelization can be easily implemented via the Message Passing Interface (MPI) [115], as necessary MPI routines are readily incorporated into MOEA implementations and are portable across a wide variety of computer architectures with either homogeneous or heterogeneous processors.

We assume (for the moment) that any of the k fitness functions used are of nearly equal computational complexity, i.e., there is no significant difference in the times required to compute them. Thus, the parallelization scheme indicated in Figure 8a is preferred. Knowing *a priori* the number of fitness functions, k processors may be assigned at the MOEA run's beginning; the overhead of sending the entire population and a function to be computed should be minimal. The same is true of each processor's result (an n-valued vector containing the kth fitness for each of the n solutions). Figure 8b's scheme likely requires a dynamic processor request and sub-population determination before data send; the parallelization possibility shown in Figure 8c is unnecessary here as we are considering only numeric fitness functions.

An argument may be made that the value of k may be higher than the number of available processors. Our past research indicates that max $|k|$ of any published MOEA is seven. A lack of available processors should not then be a problem; functions may also be "doubled up" on processors if necessary, minimizing processor idle time.

As previously noted, the MOEA being implemented performs sharing only between solutions with the same Pareto rank; however, computing the sharing values themselves (Section 9.2.1) does not require Pareto rank information. Each of these tasks (computing Pareto rank and share values) may then be assigned to different processors. After each task completes each solution's niche count (Equation 15) and shared fitness (Equation 16) can be computed and MOEA execution continued.

9.3.2 Parallel Scheduling

As we assumed each of the k fitness functions were of equal computational complexity, no load balancing is required; processor idle time should be inconsequential. This may not be the case for the parallelized Pareto ranking and share value computation tasks.

The implemented Pareto ranking requires $n - 1$ comparisons per solution for a total of $n^2 - n$ comparisons. Some number of assignments statements are (possibly) executed, bounded by zero and $n^2 - n$. Thus, the total possible computational complexity of the Pareto ranking algorithm is between $n^2 - n$ and $2(n^2 - n)$.

The implemented share value computation builds a matrix whose elements each require two subtractions, two absolute value assignments, one summation, one comparison, and one assignment. Some additional calculations are (possibly) executed, bounded by zero and n^2 divisions *and* subtractions. Thus, building the total possible computational complexity of the share value matrix computation is between $7n^2$ and $9n^2$.

Assuming all mathematical, comparison, and assignment statements execute in (nearly) identical time, maximum and minimum possible load imbalances are $8n^2$ and $5n^2$, respectively. This assumption is not entirely without basis as long as the necessary operations execute in about the same time. The difference in execution times between the Pareto ranking and sharing algorithms may then not be significant unless n is large. Traditionally, MOEAs use a value of n between a few tens and hundreds. Additionally, tuning the share value calculation routine may result in additional computational savings. The "bottom line?" As both Pareto ranking and sharing are $\mathcal{O}(n^2)$ algorithms, processor idle time should not be unreasonable.

9.3.3 Load Balancing Issues

For a general parallel MOEA, implementing many of the available static or dynamic processor scheduling and load balancing techniques appears to be "more trouble than its worth [41, 85]." The overhead involved may well be more than the MOEA computation saved. The proposed general MOEA is *not* a complex algorithm. Represented as a Directed Acyclic Graph (DAG), the MOEA tasks show more precedence relationships than asynchrony. Put another way, the parallel MOEA has a larger grain size; algorithmic decomposability is rapidly reaching its limit.

Because of this relationship where much computational information is available at compile time, some static scheduling method (automated or manual) is probably best. That is not to say that an automated scheduler has no benefit. As parallel MOEAs are applied to real-world scientific and engineering problems (where the fitness calculation time, T_f is significant) these scheduling heuristics become more important.

9.3.4 Parallel Performance Issues

Several performance metrics exist to determine parallel algorithm efficiency and effectiveness. We define primary metrics; for a more complete discussion see Kumar et al.'s textbook [85].

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.

No MOEA parallel performance results have yet been published. Additionally, any calculated results may well vary due to dissimilar MOEA implementations. For pedagogical purposes, however, we use the above Pareto ranking and sharing algorithms' worst case of $2(n^2 - n)$ and $9n^2$, respectively. We also assume any of the n operations and startup times, communication times, etc., are equal. Speedup is

$$S = \frac{9n^2 + 2n^2 - 2n}{9n^2} = \frac{11n^2 - 2n}{9n^2} \sim \frac{11}{9} \sim 1.22 \ , \tag{17}$$

efficiency is

$$E = \frac{1.22}{2} = 0.61 \ , \tag{18}$$

and cost is

$$C = 9n^2 * 2 = 18n^2 \ . \tag{19}$$

Although referring only to the fitness transformation tasks, these results indicate at least some degree of MOEA parallelization is beneficial. More realistic results are obtained by identifying the time required to perform each operation type, start up and communication times, etc., and then computing the above metrics in regard to the *entire* MOEA.

9.3.5 Parallel Architecture Issues

The parallel implementation discussed in this section tacitly considers a distributed architecture of homogeneous processors. A parallel MOEA implemented on a shared memory architecture allowing global data access would look somewhat different, and *may* be more effective than the current implementation.

For example, the fitness assignment scheme shown in Figure 8b becomes more desirable. Shared memory removes the explicit communication bottleneck, allowing "automatic" load balancing as each processor essentially executes in a SPMD manner. Also, some niching and shared fitness computations can begin before all share values are computed. As shown in Section 9.3.2, as Pareto ranking completes before *all* shared values are computed, shared memory allows easy access to the completed data while the remainder is computed.

The effectiveness and efficiency of any implemented algorithm depends on its underlying architecture. We focused more on a distributed architecture because that is the most likely computational domain. However, by identifying possible task and data decompositions, much of the work required to implement a parallel MOEA on various parallel architectures is already completed.

9.3.6 Parallel MOEA Effectiveness

Section 9.3.2 focuses more on MOEA efficiency, i.e., how well it performs computationally. The other major issue to consider is MOEA effectiveness, i.e., how "good" its reported solutions are. As we noted in Section 2.4.1 an MOEA's global optimum is PF_{true} . Thus, an MOEA implementation's effectiveness may be judged by how close to, and how much of PF_{true} the reported solutions cover.

One MOEA execution is not enough to determine effectiveness. Many MOEA runs using *several different examples* are necessary to obtain a more likely indication of MOEA effectiveness. We have elsewhere proposed suitable metrics for judging MOEA effectiveness [148] and have also proposed a standardized MOEA benchmark test function suite (see Section 10).

In broad terms, any parallel implementation offers the *possibility* of evaluating more candidate solutions than a serial version, possibly providing a "better" view of the fitness landscape.

10 MOP Test Functions

We have noted that multiobjective Evolutionary Algorithms (MOEAs) are now a well-established field within Evolutionary Computation. They were "born" in 1985 when Schaffer [128] and Fourman [54] implemented the first MOEAs dealing with Multiobjective Optimization Problems (MOPs). This document has shown that since then, over 100 published papers propose various MOEA implementations and applications, and to a much lesser extent, underlying MOEA theory. Many of these efforts use numeric MOPs as examples to show algorithmic performance. Nowhere in the literature, however, is there a comprehensive collocated discussion of relevant MOP landscape issues; nor is there any explanation of why numeric MOPs may be appropriate MOEA test functions. Extensive experimentation and result analyses from testing various MOEA parameters, components, and approaches are also missing.

Thus, we provide various numeric MOPs for use as part of a standardized MOEA test suite, and propose a methodology whereby various MOEAs can be directly compared. Substantiating these contributions is a detailed discussion of MOP landscapes and general test suite issues, and identification of the salient characteristics of all known MOEA example numeric functions. A theorem is presented defining one MOP characteristic. Possible multiobjective NP-Complete problems are also suggested.

In this section we introduce necessary test suite concepts, discuss the debate the merits of past numeric examples, and propose appropriate numeric examples given the MOP domain. We also offer a methodology for quantitatively comparing MOEA performance.

10.1 An MOP Test Suite: Is it Important?

Test function suites have both supporters and detractors. Any algorithm successfully passing all submitted test functions has no guarantee of continual effectiveness and efficiency (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 were possibly deadly. Pattern recognition work has also recognized the problem of "testing on the training data," where an algorithm is tuned for one or a few problem instantiations [39]. These analogies hold when integrating problem and algorithm domains: new and unforeseen situations may easily arise. An MOEA test suite can be a valuable tool if these issues are properly considered.

Any proposed MOP test suite must contain "MOEA challenging" functions. In order to identify appropriate functions for inclusion, relevant MOP characteristics must be identified. We have shown in a previous work [148] that the global optimum of an MOP is the Pareto front determined by evaluating each member of the Pareto optimal solution set. Pareto concepts are crucial components of appropriate MOEA test functions, and are also now defined.

10.1.1 General MOEA Test Suite Issues

The "No Free Lunch" (NFL) theorem [161] implies that if problem domain knowledge is not incorporated into the algorithm domain, no formal assurances of an implemented algorithm's general effectiveness exist. Previously, EA test suites containing various functions were proposed for testing an EA's capability to "handle" various problem domain characteristics. These suites incorporate relevant search space features which should be addressed by a particular EA instantiation. For example, De Jong [33] suggests five single-objective optimization test functions (**F1 - F5**), and Michalewicz [99] suggests five single-objective *constrained* optimization test functions(**G1 - G5**). Whitley et al. [157] cite five other proposed test sets. Goldberg et al. [59] suggest *deceptive* problems.

De Jong's test bed includes functions with the following characteristics [57]: 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 [99]: type of objective function, number of decision variables and constraints, types of constraints, number of active constraints at the function's optimum, and the ratio between the feasible and complete search space size. Particular EA instantiations are then subjected to test beds like these, e.g., De Jong [33], Michalewicz [99], and Schwefel [129].

Note that the NFL theorem also implies that 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, we must define traits common to all (most) MOPs for test suite consideration.

10.1.2 MOEA Test Suite Guidelines

We have already mentioned that test suites must contain characteristic problems from the tested algorithm's problem domain. Some of these problems should represent real world situations, and also range in difficulty from "easy" to "hard." In addition, we consider the following guidelines suggested by Whitley et al. [157].

- 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.

In Section 10.2.2 we relate proposed functions to these specific guidelines. But first, we discuss the general need for an MOEA test suite.

10.1.3 Requirements for an MOEA Test Suite

To date, most MOEA researchers' *modus operandi* is an algorithm's comparison (usually the researcher's own new and improved variant) against an older MOEA (typically VEGA, even though it has identified shortfalls [47]), and analyzing results for some MOP (typically Schaffer's F2 [128], or some other numeric example). Results are often "clearly" shown in graphical form, indicating which algorithm's PF_{known} is a better representation of PF_{true} .

These empirical comparisons do not contribute much to a common basis for MOEA comparisons. The literature's history of visually comparing MOEA performance on non-standard numeric problems does little to determine a given MOEA's efficiency and effectiveness. A standard suite of numeric functions exhibiting *relevant* MOP problem domain characteristics provides a common comparative basis.

As indicated, the MOEA community has created a limited *de facto* test suite. However, its functions are used only because they appear to exercise MOEA capabilities, or perhaps because other researchers used them as examples for their algorithms. Thus, a documented MOP test suite would be an asset to MOEA research.

10.2 An MOEA Test Function Suite

As indicated, a *de facto* test suite exists. As a reference for this discussion, all known MOPs used as MOEA numeric test examples are presented in Tables 18 and 19. All are minimization problems unless otherwise specified.

Table 18: MOP Numeric Test Functions

Researcher & Major MOP Characteristics	**Definition**	**Constraints**
Fonseca [50]; 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) [49]; P_{true} connected, PF_{true} concave, Analytical solution	$F = (f_1(\vec{x}), f_2(\vec{x}))$, where $$\begin{aligned} f_1(\vec{x}) &= 1 - \exp(-\sum_{i=1}^{n}(x_i - \frac{1}{\sqrt{n}})^2), \\ f_2(\vec{x}) &= 1 - \exp(-\sum_{i+1}^{n}(x_i + \frac{1}{\sqrt{n}})^2) \end{aligned}$$	$-2 \le x_i < 2$

Table 18: (continued)

Researcher & Major MOP Characteristics	Definition	Constraints
Kursawe[7] [88]; P_{true} disjoint, PF_{true} continuous	$F = (f_1(\vec{x}), f_2(\vec{x}))$, where $f_1(\vec{x}) = \sum_{i=1}^{n-1}(-10e^{(-0.2)\sqrt{x_i^2+x_{i+1}^2}})$, $f_2(\vec{x}) = \sum_{i=1}^{n}(\|x_i\|^{0.8} + 5sin(x_i)^3)$	None
Laumanns [90]; P_{true} disjoint, PF_{true} convex	$F = (f_1(x,y), f_2(x,y))$, where $f_1(x,y) = x^2 + y^2$, $f_2(x,y) = (x+2)^2 + y^2$	$-50 \leq x, y \leq 50$
Lis [92]; P_{true} disjoint, PF_{true} discontinuous and concave	$F = (f_1(x,y), f_2(x,y))$, where $f_1(x,y) = \sqrt[8]{x^2+y^2}$, $f_2(x,y) = \sqrt[4]{(x-0.5)^2 + (y-0.5)^2}$	$-5 \leq x, y \leq 10$
Murata[8] [106]; P_{true} connected, PF_{true} concave	$F = (f_1(x,y), f_2(x,y))$, where $f_1(x,y) = 2\sqrt{x}$, $f_2(x,y) = x(1-y) + 5$	$1 \leq x \leq 4,\ 1 \leq y \leq 2$
Rendon [147]; P_{true} connected, PF_{true} convex	$F = (f_1(x,y), f_2(x,y))$, where $f_1(x,y) = \frac{1}{x^2+y^2+1}$, $f_2(x,y) = x^2 + 3y^2 + 1$	$-3 \leq x, y \leq 3$
Rendon (2) [147]; P_{true} connected, PF_{true} convex	$F = (f_1(x,y), f_2(x,y))$, where $f_1(x,y) = x + y + 1$, $f_2(x,y) = x^2 + 2y - 1$	$-3 \leq x, y \leq 3$
Schaffer[9] [128]; P_{true} connected, PF_{true} convex, Analytical solution	$F = (f_1(x), f_2(x))$, where $f_1(x) = x^2$, $f_2(x) = (x-2)^2$	None

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

[8]Tamaki [141] gives an almost identical function.

[9]Norris [109] gives an almost identical function; his modification was intended to ease analysis.

Table 18: (continued)

Researcher & Major MOP Characteristics	Definition	Constraints
Schaffer (2) [133, 13]; P_{true} disjoint, PF_{true} discontinuous	$F = (f_1(x), f_2(x))$, where $$\begin{aligned} f_1(x) &= -x, \text{ if } x \leq 1, \\ &= -2 + x, \text{ if } 1 < x \leq 3, \\ &= 4 - x, \text{ if } 3 < x \leq 4, \\ &= -4 + x, \text{ if } x > 4, \\ f_2(x) &= (x-5)^2 \end{aligned}$$	$-5 \leq x, y \leq 10$
Viennet [152]; P_{true} connected and symmetric, PF_{true} curved surface	$F = (f_1(x,y), f_2(x,y), f_3(x,y))$, where $$\begin{aligned} f_1(x,y) &= x^2 + (y-1)^2, \\ f_2(x,y) &= x^2 + (y+1)^2 + 1, \\ f_3(x,y) &= (x-1)^2 + y^2 + 2 \end{aligned}$$	$-2 \leq x, y \leq 2$
Viennet (2) [152]; P_{true} connected, PF_{true} discontinuous	$F = (f_1(x,y), f_2(x,y), f_3(x,y))$, 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) [152]; P_{true} disjoint and unsymmetric, PF_{true} continuous	$F = (f_1(x,y), f_2(x,y), f_3(x,y))$, 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 19: MOP Numeric Test Functions (with side constraints)

Researcher & Major MOP Characteristics	Definition	Constraints
Belegundu [10]; P_{true} connected, PF_{true} continuous	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= -2x + y, \\ f_2(x,y) &= 2x + y \end{aligned}$$	$0 \leq x \leq 5,\ 0 \leq y \leq 3,$ $$\begin{aligned} 0 &\geq -x + y - 1, \\ 0 &\geq x + y - 7 \end{aligned}$$

Table 19: (continued)

Researcher & Major MOP Characteristics	Definition	Constraints
Kita [82]; P_{true} disjoint, PF_{true} discontinuous and concave	Maximize $F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= -x^2 + y, \\ f_2(x,y) &= \frac{1}{2}x + y + 1 \end{aligned}$$	$x, y \geq 0$, $$\begin{aligned} 0 &\geq \frac{1}{6}x + y - \frac{13}{2}, \\ 0 &\geq \frac{1}{2}x + y - \frac{15}{2}, \\ 0 &\geq 5x + y - 30 \end{aligned}$$
Osyczka [114]; P_{true} disjoint, 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) [114]; P_{true} disjoint, PF_{true} discontinuous	$F = (f_1(\vec{x}), f_2(\vec{x}))$, 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 [133]; P_{true} disjoint, PF_{true} continuous	$F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= (x-2)^2 + (y-1)^2 + 2, \\ f_2(x,y) &= 9x - (y-1)^2 \end{aligned}$$	$-20 \leq x, y \leq 20$, $$\begin{aligned} 0 &\geq x^2 + y^2 - 225, \\ 0 &\geq x - 3y + 10 \end{aligned}$$
Tamaki [140]; 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 $$\begin{aligned} f_1(x,y,z) &= x, \\ f_2(x,y,z) &= y, \\ f_3(x,y,z) &= z \end{aligned}$$	$0 \leq x, y, z$, $$x^2 + y^2 + z^2 \leq 1$$
Tanaka [142]; P_{true} connected, PF_{true} convoluted	Minimize $F = (f_1(x,y), f_2(x,y))$, where $$\begin{aligned} f_1(x,y) &= x, \\ f_2(x,y) &= y \end{aligned}$$	$0 < x, y \leq \pi$, $$\begin{aligned} 0 &\geq -(x^2) - (y^2) + 1 + \\ & \quad 0.1cos(16arctan\frac{x}{y}) \\ \frac{1}{2} &\geq (x - \frac{1}{2})^2 + (y - \frac{1}{2})^2 \end{aligned}$$

Table 19: (continued)

Researcher & Major MOP Characteristics	Definition	Constraints
Viennet (4) [152]; P_{true} connected and unsymmetric, PF_{true} a curved surface	$F = (f_1(x,y), f_2(x,y), f_3(x,y))$, where $\begin{aligned} f_1(x,y) &= \frac{(x-2)^2}{2} + \frac{(y+1)^2}{13} + 3, \\ f_2(x,y) &= \frac{(x+y-3)^2}{175} + \frac{(2y-x)^2}{17} - 13, \\ f_3(x,y) &= \frac{(3x-2y+4)^2}{8} + \frac{(x-y+1)^2}{27} + 15 \end{aligned}$	$-4 \leq x, y \leq 4$, $\begin{aligned} y &< -4x+4, \\ x &> -1, \\ y &> x-2 \end{aligned}$

When implementing an MOEA, it is (implicitly) assumed that the problem domain (fitness landscape) has been examined, and a decision made that an MOEA is the most appropriate search tool for the given MOP. We also assume the MOEA returns P_{known} and PF_{known} , i.e., a *set* of solutions and their evaluated function values. It is not clear that all existing test functions are appropriate for MOEA search; thus, identification of appropriate functions is required to objectively determine MOEA efficiency and effectiveness.

In general, it is 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 of the numerical examples used by MOEA researchers do *not* explicitly meet this criteria. Our research identifies 21 different numerical MOPs (both constrained and unconstrained) used in published MOEA efforts. All but two use at most two decision variables and the majority use only two objective functions. This implies that unless the search space is very large, MOEA performance claims and comparisons based on these functions may not be meaningful. The algorithm may be operating in a problem domain not particularly well-suited to its capabilities. We must then determine relevant MOP problem domain characteristics.

10.2.1 The MOP Domain

What is the nature of the Pareto optimal set (P_{true})? Few MOEA efforts report any description of an example MOP's underlying decision variable space, i.e., the space where P_{true} resides. Since an MOP is composed of two or more single-objective optimization problems, the solution space is restricted by the limitations of those combined functions. Within that space, P_{true} may be connected, disconnected, a hyperarea, separate points, etc. However, in MOEA search the Pareto front is of more interest because solutions are often implicitly chosen via selecting a point from PF_{known} .

What is the nature of the Pareto front (PF_{true})? Other researchers have alluded that the structure of *any* Pareto front (independent of dimensionality) has theoretical limitations. For example, any Pareto surface must be monotonic (i.e., all first-order partial derivatives never change sign), and that the Pareto surface has asymptotic bounds in terms of area [52, 70]. We have recently realized that an additional structural limitation exists, and developed a theorem describing it.

Theorem 1: The Pareto front of an MOP with two objectives ($k = 2$) is at most a curve, and is at most a surface when $k \geq 3$. □

Proof: By definition, all vectors of the Pareto front are *nondominated.* Given a minimization MOP with two objectives ($k = 2$), assume PF_{true} is a polygon. Now assume an imaginary line parallel to either of the objective axes passing through at least two points (represented by vectors) of the polygon. Since performance in each objective is to be minimized, one of these points is clearly "better" than the other and dominates it. But PF_{true} is composed only of *non*dominated vectors. Thus, the original assumption is incorrect and the Pareto front of an MOP with two objectives is at most a curve.

Given a minimization MOP with three or more objectives ($k \geq 3$), assume PF_{true} is a volume. Now assume an imaginary line parallel to any of the objective axes passing through at least two points (represented by vectors) of the volume. Since performance in each objective is to be minimized, one of these points is clearly "better" than the other and dominates it. But PF_{true} is composed only of *non*dominated vectors. Thus, the original assumption is incorrect and the Pareto front of an MOP with three or more objectives is at most a surface. □

Horn states that in a k-objective MOP, the Pareto front *is* a $k-1$ dimensional surface [70]. We have just shown this is incorrect; PF_{true} is *at most* a surface *only when* $k \geq 3$. Although asymptotic bounds are useful, researchers must also understand the front's possible shape within those bounds.

This theorem also implies that any MOEA test suite should contain MOPs with both types of Pareto fronts: k-dimensional curves and $(k-1)$-dimensional surfaces. This is necessary to fully test an MOEA's search capability.

10.2.2 MOEA Test Suite Functions

We have identified several MOP characteristics which must be dealt with by effective and efficient MOEAs: large search spaces, high-dimensionality (i.e., many decision variables), multiple objectives, a global optimum composed of a shape of bounded complexity, and a *set* of solutions. We now propose initial problems for an MOP test suite, drawn from the published literature, which incorporate some of these characteristics.

Schaffer's first (unconstrained) two-objective function (see Table 18) 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. Secondly, as we have noted elsewhere [148], this MOP allows us to determine an analytical expression for the Pareto front. Third, as noted by Rudolph [124], this MOP's PF_{true} is given in closed form. The representation of solutions composing P_{true} , at any resolution, is thus easily determined without the necessity of exhaustively enumerating the search space. However, its one decision variable implies a large search space should be used when testing an MOEA (e.g., $-50 \leq x \leq 50$). We rename this function **MOP1**.

As part of the test suite, we also propose Fonseca's second function (see Table 18). This function has the advantage of arbitrarily adding decision variables without changing PF_{true} 's structure. Additionally, this MOP's P_{true} is given in closed form [49]. We rename this function **MOP2**.

Finally, we propose Viennet's third function (see Table 18). This function has three objective functions, and its Pareto front appears to follow a fairly convoluted path through objective space. This front is a k-dimensional curve. This MOP should challenge an MOEA's ability to find and maintain the entire front. We rename this function **MOP3**.

Appendices **??** and **??** present a complete set of figures showing P_{true} and PF_{true} for each function listed in Tables 18 and 19, respectively. These figures are deterministically derived by computing all variable combinations possible at a given computational resolution. As the underlying resolution is varied the figures may slightly change. These figures highlight major structural characteristics of both P_{true} and PF_{true} .

The proposed MOP test function suite problem's characteristics address some of the issues mentioned in Section 10.1.2. MOP1 is arguably an "easy" MOP. MOP2 is scalable, in that any number of decision variables may be used to increase the search space. MOP3's functions are nonlinear and nonsymmetric. Other relevant MOP characteristics (as reflected in Tables 18 and 19) should be addressed by further MOPs selected for test suite inclusion.

10.2.3 NP-Complete MOPs

We also consider the use of combinatorial optimization problems in the test suite. EAs often employ specialized representations and operators when solving these real-world problems. This may prevent general comparison between MOEAs, but the problems' inherent difficulty should present desired algorithmic challenges. Table 10.2.3 suggests possible NP-Complete MOPs for test set inclusion.

10.3 MOEA Experimental Methodology

Having investigated the MOP domain, and proposed appropriate functions composing an MOP test suite, we are now in a position to perform useful MOEA experiments. Although test suite functions provide a common basis for MOEA comparison, these comparisons are still empirical unless PF_{true} is known. We earlier intimated this is the case for most MOPs of any complexity. However, there is a way to determine PF_{true} at a given computational resolution! This section describes such a process we are currently using to construct an experimental database for selected MOPs. It also suggests appropriate metrics and statistical analysis which might be used when comparing MOEAs.

Table 20: Possible Multiobjective NP-Complete Functions

NP-**Complete Problem**	**Examples**
0/1 Knapsack - Bin Packing	Max profit; Min weight (Multiple Knapsacks [167])
Coloring	Min # colors, # of each color
Layout	Min space, overlap, costs
Maximum Independent Set (Clique)	Max set size; Min geometry
Scheduling	Min time, missed deadlines, waiting time, resources used
Set/Vertex Covering	Min total cost, over-covering
Traveling Salesperson	Min energy, time, and/or distance; Max expansion
Vehicle Routing	Min time, energy, and/or geometry
NP-Complete Problem Combinations	Vehicle scheduling and routing

10.3.1 Experimental Database

When the real (continuous) world is modeled (e.g., via objective functions) upon a computer (a discrete machine), there is a fidelity loss between the real world and implemented model. However, at a standardized resolution and representation, MOEA results can be compared against both each other *and* PF_{true} . Thus, whether or not a given MOP's true Pareto front is actually continuous or discrete is then not a major concern, as the front is always composed of discrete points at a specified computational resolution.

We suggest both a large search space and a binary encoding for any MOEA comparisons. The large search space challenges an MOEA's ability to find the global optimum; the binary encoding allows for both a standard chromosomal representation and a method for deterministically searching an entire space. At a given resolution, the entire solution space is enumerated, thus obtaining P_{true} and PF_{true} (at that resolution).

For example, assume a 30-bit binary encoding. This search space contains 2^{30} possible solutions, over one billion distinct possibilities! Published results indicate most EAs execute between thousands and tens of thousands of fitness evaluations during search, thus, the effectiveness of any MOEA approach should be readily apparent in how well its results (P_{known} and PF_{known}) compare to P_{true} and PF_{true} in a search space of this large size. We have constructed an experimental database allowing this comparison for selected examples.

This effort is in part due in part to a paper suggesting that exhaustive search may be the only viable approach to solving irregular or chaotic problems [108]. The authors propose harnessing ever-expanding computational capability to solve problems by exhaustive deterministic enumeration. We constructed such a program executing on the IBM SP-2.

Our program's "C" implementation uses the Message Passing Interface (MPI) to distribute function evaluation among many processors. For a given MOP, each processor initially evaluates some subset of solutions and stores the resultant vectors. These vectors are compared on the basis of Pareto optimality; nondominated solutions and their corresponding vectors are written to disk. Noting that Pareto optimality places a partial ordering on the entire search space, combining the separate solutions/vectors from different processors and again comparing vectors results in P_{true} and PF_{true} , *for that particular binary encoded resolution.* Program timing and processor loading are also recorded to determine problem scaling. To date, we have successfully extracted an MOP's P_{true} and PF_{true} from a search space of size 2^{26}.

Using this database, solutions offered by various MOEAs can be compared not only against each other, but against the *true* Pareto optimal set and front. At least for selected MOPs, relativity is removed and a true quantitative comparison is possible. This methodology allows absolute, rather than relative observations.

10.3.2 MOEA Comparative Metrics

We propose quantitative comparisons between MOEA implementations via a rigorous, carefully designed series of experiments. However, we must first define the metrics upon which we base MOEA performance.

Our deterministic enumerative search provides P_{true} and PF_{true} (at a given level of resolution). After executing an MOEA on some MOP we are able to compare the reported front (PF_{known}) against the true front and determine error measures. For example, an MOEA reports a finite number of solutions. These solutions will or will not be members of PF_{true} . If they are not the MOEA has erred. Summing each error gives one measure of MOEA effectiveness. For example, an error of "0" means that every vector reported by the MOEA in PF_{known} is actually in PF_{true} . This technique is mathematically represented by:

$$\sum_{i=1}^{n} e_i, \tag{20}$$

where n is the number of vectors in PF_{known} , $e_i = 0$ if vector i is a member of PF_{true} , and 1 otherwise.

The *generational distance* used by Van Veldhuizen and Lamont in earlier experiments [148] may be an effective metric gauging MOEA performance. Generational distance is a value representing how "far" PF_{known} is from PF_{true} (an error measure).

Zitzler and Thiele propose two MOEA comparative metrics [167]. First is that of *coverage*. Coverage occurs when one solution's associated objective vector (phenotype) dominates another's (mathematically represented by $a \prec b$), or the two solutions are equal ($a = b$). Coverage defines the size of objective value space covered by PF_{known} . For example, a point on PF_{known} in the two-dimensional (minimization) case defines a rectangle bounded by the points $(0, 0)$ and $(f_1(x), f_2(x))$. The union of all such rectangles defined by each vector in PF_{known} is used as the comparative measure. Secondly, for any two P_{known} sets, they compute the fraction of solutions in one set covered by solutions in the other.

Finally, another possible metric is one measuring the "spread" of vectors throughout PF_{known} . Many MOEAs perform sharing and niching [52, 70, 133], attempting to spread their population ($PF_{current}$) evenly along the front. Because PF_{true} 's "beginning" and "end" are known (at some resolution), a suitably defined metric judges how well PF_{known} conforms. Srinivas and Deb [133] define such a measure which expresses how well an MOEA has distributed individuals over the nondominated region.

They define this performance measure as:

$$\iota = \sqrt{\sum_{i=1}^{q+1} (\frac{n_i - \overline{n}_i}{\sigma_i})^2} \ , \tag{21}$$

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, 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.

11 Conclusions and Recommendations

This report organizes, presents, analyzes, and extends existing MOEA research. All known MOEA implementations and techniques are discussed under the umbrella of *a priori*, progressive, and *a posteriori* techniques. This classification and tabular presentation forms a consistent foundation upon which to evaluate the contemporary MOEA "state of the art." A short discussion and mathematical formulation of known MOEA techniques, along with key points of these known MOEA implementations are presented. Known publications which either focus on MOEA comparisons or different aspects of MOEA theory are also offered. These descriptions are the basis for our conclusions and recommendations.

Contemporary MOEA research emphasizes Pareto-based approaches. This interest is understandable, as these particular approaches can be used in more desirable MOP search techniques (progressive and *a posteriori*) with their general ability to find acceptable representations of P_{true} $/PF_{true}$. MOEA utility is demonstrated by its application to several problem domains, and incorporation of various fitness function types and chromosomal representations.

Current MOEA theory, however, leaves much to be desired. The efficiency and effectiveness of various algorithmic possibilities (e.g., number and types of fitness functions; Pareto ranking, niching, and fitness sharing schemes; mating restriction schemes, etc.) is not yet quantifiably determined. Furthermore, this performance evidence is sometimes contradictory. The discussion of various MOEA parallelization possibilities does however indicate possible efficiency gains. Although appropriate MOEAs are suggested based on the known evidence, a methodology with which to quantitatively test MOEAs is required.

We suggest such an experimental methodology along with potential functions for inclusion in an MOP test function suite. This methodology allows for the desired quantitative performance comparisons of various MOEA techniques and theoretical components.

Finally, our future efforts include adding new MOEA citations as they become available. We are also extending the MOEA test suite and experimental methodology laid forth in this paper, as well as constructing additional "MOEA challenging" functions. We are also investigating alternative EA-based MOP solution approaches such as Ant Colony Optimization [17] and Immune EAs [66].

References

[1] 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.

[2] 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.

[3] 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.

[4] 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. Londond: IEE, April 1998.

[5] Awadh, B., et al. "A Computer-Aided Process Planning Model Based on Genetic Algorithms," *Computers in Operations Research*, *22*(8):841–856 (1995).

[6] Bäck, Thomas. *Evolutionary Algorithms in Theory and Practice*. New York: Oxford University Press, 1996.

[7] Bäck, Thomas, editor. *Proceedings of the Seventh International Conference on Genetic Algorithms*, San Francisco, CA: Morgan Kaufmann Publishers, Inc., July 1997.

[8] Bäck, Thomas, et al., editors. *Handbook of Evolutionary Computation*, *1*. IOP Publishing Ltd. and Oxford University Press, 1997.

[9] Baita, Flavio, et al. *Evolutionary Algorithms in Management Applications*, chapter Genetic Algorithm with Redundancies for the Vehicle Scheduling Problem, 341–353. In Biethahn and Nissen [15], 1995.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] Bhanu, Bir and Sungkee Lee. *Genetic Learning for Adaptive Image Segmentation*. Boston: Kluwer Academic Publishers, 1994.

[15] Biethahn, J. and Volker Nissen, editors. *Evolutionary Algorithms in Management Applications*. Berlin: Springer-Verlag, 1995.

[16] 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.

[17] Bilchev, G. and I. C. Parmee. "The Ant Colony Metaphor for Searching Continuous Design Spaces." *Proceedings of the 2nd AISB Workshop on Evolutionary Computing*. 25–39. Berlin: Springer, 1995.

[18] Broekmeulen, Rob A. C. M. *Evolutionary Algorithms in Management Applications*, chapter Facility Management of Distribution Centres for Vegetables and Fruits, 199–210. In Biethahn and Nissen [15], 1995.

[19] 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, Aug 1997.

[20] Cantú-Paz, Erick. *A Survey of Parallel Genetic Algorithms*. Technical Report 97003, University of Illinois at Urbana-Champaign, May 1997.

[21] Cedeno, Walter and V. Rao Vemuri. "On the Use of Niching For Dynamic Landscapes." In Porto [121], 361–366.

[22] Chang, C. S., et al. "Genetic Algorithm Based Bicriterion Optimization for Traction Sustations in DC Railway System." In Fogel [44], 11–16.

[23] Chow, C. Rick. "An Evolutionary Approach to Search for NCR-Boards." In Fogel [45], 295–300.

[24] Cieniawski, Scott E., et al. "Using Genetic Algorithms to Solve a Multiobjective Groundwater Monitoring Problem," *Water Resources Research*, *31*(2):399–409 (February 1995).

[25] 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.

[26] Cohon, Jared L. *Multiobjective Programming and Planning*. New York: Academic Press, 1978.

[27] Cohon, Jared L. and David H. Marks. "A Review and Evaluation of Multiobjective Programming Techniques," *Water Resources Research*, *11*(2):208–220 (1975).

[28] 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.

[29] 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.

[30] 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, Jan 1997.

[31] Crossley, William A., et al. *Using the Two-Branch Tournament Genetic Algorithm for Multiobjective Design*. Technical Report AIAA-98-1914, AIAA, 1998.

[32] Cunha, A. Gaspar, et al. "Use of Genetic Algorithms in Multicriteria Optimization to Solve Industrial Problems." In Bäck [7], 682–688.

[33] 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.

[34] Deb, Kalyanmoy. *Binary and Floating-Point Function Optimization Using Messy Genetic Algorithms*. PhD dissertation, University of Alabama, Tuscaloosa, AL, 1991.

[35] 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.

[36] Dimopoulos, C. and A.M.S. Zalzala. "Optimization of Cell Configuration and Comparisons Using Evolutionary Computation Approaches." In Fogel [45], 148–153.

[37] Dozier, Gerry, et al. "Multiobjective Evolutionary Path Planning via Fuzzy Tournament Selection." In Fogel [45], 684–689.

[38] Duckstein, L. *Multiobjective Optimization in Structural Design: The Model Choice Problem*, chapter 22, 459–481. John Wiley & Sons Ltd., 1984.

[39] Duda, Richard O. and Peter E. Hart. *Pattern Classification and Scene Analysis*. New York, NY: John Wiley & Sons, 1973.

[40] Eiben, A. E., et al., editors. *Parallel Problem Solving from Nature - PPSN V*, Berlin: Springer, 1998.

[41] El-Rewini, Hesham, et al. *Task Scheduling in Parallel and Distributed Systems*. Prentice Hall, 1994.

[42] Ely, T. A., et al. *Satellite Constellation Design for Zonal Coverage Using Genetic Algorithms*. Technical Report AAS-98-128, San Diego, CA: American Astronautical Society, February 1998.

[43] Eshelman, Larry J., editor. *Proceedings of the Sixth International Conference on Genetic Algorithms*, San Mateo CA: Morgan Kaufmann Publishers, Inc., 1995.

[44] Fogel, David, editor. *Proceedings of the Second IEEE Conference on Evolutionary Computation*, Piscataway NJ: IEEE Service Center, 1995.

[45] Fogel, David, editor. *Proceedings of the 1998 (5th) IEEE International Conference on Evolutionary Computation*, Piscataway, NJ: IEEE, May 1998.

[46] Fogel, David B. and Adam Ghozeil. "A Note on Representations and Variation Operators," *IEEE Transactions on Evolutionary Computation*, *1*(2):159–161 (July 1997).

[47] 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. [8], 1997.

[48] Fonseca, Carlos M. and Peter J. Fleming. "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion, and Generalization." In Forrest [53], 416–423.

[49] 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*. Number 414. 45–52. September: IEE, 1995.

[50] Fonseca, Carlos M. and Peter J. Fleming. "An Overview of Evolutionary Algorithms in Multiobjective Optimization," *Evolutionary Computation*, *3*(1):1–16 (Spring 1995).

[51] 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.

[52] Fonseca, Carlos M. and Peter J. Fleming. "Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms – Part I: A Unified Formulation, and Part II: Application Example," *IEEE Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans*, *28*(1):26–47 (January 1998).

[53] Forrest, Stephanie, editor. *Proceedings of the Fifth International Conference on Genetic Algorithms*, San Mateo CA: Morgan Kaufmann Publishers, Inc., July 1993.

[54] Fourman, Michael P. "Compaction of Symbolic Layout Using Genetic Algorithms." In Grefenstette [62], 141–153.

[55] Gen, Mitsuo, et al. "A Spanning Tree-Based Genetic Algorithm for Bicriteria Topological Network Design." In Fogel [45], 15–20.

[56] Gero, J. S. and S. Louis. "Improving Pareto Optimal Designs Using Genetic Algorithms," *Microcomputers in Civil Engineering*, *10*(4):241–249 (1995).

[57] Goldberg, David E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison Wesley, 1989.

[58] 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.

[59] Goldberg, David E., et al. "Messy Genetic Algorithms: Motivation, Analysis, and First Results," *Complex Systems*, *3*:493–530 (1989).

[60] 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.

[61] Greenwood, Garrison W., et al. "Fitness Functions for Multipleobjective 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.

[62] 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.

[63] Groppetti, R. and R. Muscia. *Integrated Design and Manufacturing in Mechanical Engineering*, chapter On a Genetic Multiobjective Approach for the Integration and Optimization of Assembly Product Design and Process Planning, 61–70. The Netherlands: Kluwer Academic Publishers, 1997.

[64] Guillen, Michael. *Bridges to Infinity*. Los Angeles: Jeremy P. Tarcher, Inc., 1983.

[65] Hajela, P. and C. Lin. "Genetic Search Strategies in Multicriterion Optimal Design," *Structural Optimization*, *4*:99–107 (1992).

[66] Hajela, P., et al. "GA Based Simulation of Immune Networks Applications in Structural Optimization," *Engineering Optimization*, *29*:131–149 (1997).

[67] 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.

[68] Hinchliffe, Mark, et al. "Chemical Process Systems Modelling 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.

[69] Horn, Jeffrey. *Handbook of Evolutionary Computation*, chapter Multicriterion Decision Making, F1.9:1 – F1.9:15. Volume 1 of Bäck et al. [8], 1997.

[70] 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.

[71] Horn, Jeffrey, et al. "A Niched Pareto Genetic Algorithm for Multiobjective Optpimization." *Proceedings of the First IEEE Conference on Evolutionary Computation*, edited by Zbigniew Michalewicz. 82–87. Piscataway NJ: IEEE Service Center, June 1994.

[72] Hu, Xiabo (Sharon), et al. "An Evolutionary Approach to Hardware/Software Partitioning." In Voigt et al. [154], 900–909.

[73] Hwang, Ching-Lai and Abu Syed Md. Masud. *Multiple Objective Decision Making - Methods and Applications*. Springer Verlag, 1979.

[74] Ignizio, James P. "Integrating Cost, Effectiveness, and Stability," *Acquisition Review Quarterly*, 51–60 (Winter 1998).

[75] Ikonen, Ilkka, et al. "A Genetic Algorithm for Packing Three-Dimensional Non-Convex Objects Having Cavities and Holes." In Bäck [7], 591–598.

[76] Ishibuchi, Hisao and Tadahiko Murata. "Multi-Objective Genetic Local Search Algorithm." In Thomas Bäck and Zbigniew Michalewicz and Hiroaki Kitano [143], 119–124.

[77] 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).

[78] Ishibuchi, Hisao and Tadahiko Murata. "Multi-Objective Genetic Local Search for Minimizing the Number of Fuzzy Rules for Pattern Classification Problems." In Fogel [45], 1100–1105.

[79] Jakob, Wilfried, et al. "Application of Genetic Algorithms to Task Planning and Learning." *Parallel Problem Solving from Nature, 2*. 291–300. Elsevier Science Publishers B. V., 1992.

[80] 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.

[81] Jones, Gareth, et al. "Searching Databases of Two-Dimensional and Three-Dimensional Chemical Structures Using Genetic Algorithms." In Forrest [53], 597–602.

[82] Keeney, Ralph L. and Howard Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. New York, NY: John Wiley & Sons, 1976.

[83] Kita, Hajime, et al. "Multi-Objective Optimization by Means of the Thermodynamical Genetic Algorithm." In Voigt et al. [154], 504–512.

[84] Krause, Matthias and Volker Nissen. *Evolutionary Algorithms in Management Applications*, chapter On Using Penalty Functions and Multicriteria Optimization Techniques in Facility Layout, 153–166. In Biethahn and Nissen [15], 1995.

[85] 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.

[86] Kumar, Vipin, et al. *Introduction to Parallel Computing: Design and Analysis of Algorithms*. Redwood City, CA: The Benjamin/Cummings Publishing Company, Inc., 1994.

[87] 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.

[88] 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.

[89] 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.

[90] Langdon, William B. "Evolving Data Structures with Genetic Programming." In Eshelman [43], 295–302.

[91] Laumanns, Marco, et al. "A Spatial Predator-Prey Approach to Multi-Objective Optimization." In Eiben et al. [40].

[92] 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.

[93] Lis, Joanna and A. E. Eiben. "A Multi-Sexual Genetic Algorithm for Multiobjective Optimization." In Porto [121], 59–64.

[94] 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.

[95] Loughlin, Daniel H. and S. Ranjithan. "The Neighborhood Constraint Method: A Genetic Algorithm-Based Multiobjective Optimization Technique." In Bäck [7], 666–673.

[96] Louis, Sushil J. and Gregory J. E. Rawlins. "Pareto Optimality, GA-Easiness, and Deception." In Forrest [53], 118–123.

[97] Mäkinen, R. A. E., et al. *Parallel Computational Fluid Dynamics: Algorithms and Results Using Advanced Computers*, chapter Parallel Genetic Solution for Multiobjective MDO, 352–359. Elsevier Science, 1997.

[98] Michalewicz, Z. and G. Nazhiyath. "Genocop III: A Co-Evolutionary Algorithm for Numerical Optimization Problems with Nonlinear Constraints." In Fogel [44], 647–651.

[99] Michalewicz, Zbigniew. *Genetic Algorithms + Data Structures = Evolution Programs* (2nd Edition). New York: Springer-Verlag, 1994.

[100] Michalewicz, Zbigniew. "Genetic Algorithms, Numerical Optimization, and Constraints." In Eshelman [43], 151–158.

[101] 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).

[102] Michielssen, Eric and Daniel S. Weile. *Genetic Algorithms in Engineering and Computer Science*, chapter Electromagnetic System Design Using Genetic Algorithms, 345–369. In Winter et al. [160], 1995.

[103] 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).

[104] 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 Electromagnetics11*. 386–393. 1995.

[105] Moon, Chiung, et al. "Evolutionary Algorithm for Flexible Process Sequencing with Multiple Objectives." In Fogel [45], 27–32.

[106] Mori, Naoki, et al. "Thermodynamical Selection Rule for the Genetic Algorithm." In Fogel [44], 188–192.

[107] Murata, Tadahiko and Hisao Ishibuchi. "MOGA: Multi-Objective Genetic Algorithms." In Fogel [44], 289–294.

[108] Nam, Dongkyung, et al. "Parameter Optimization of a Voltage Reference Circuit Using EP." In Fogel [45], 301–305.

[109] 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.

[110] 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.

[111] Obayashi, Shigeru. "Multidisciplinary Design Optimization of Aircraft Wing Planform Based on Evolutionary Algorithms." *1998 IEEE International Conference on Systems, Man, and Cybernetics*. 1998.

[112] 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 Candada2*. 41–46. 1998.

[113] Obayashi, Shigeru, et al. "Niching and Elitist Models for MOGAs." In Eiben et al. [40].

[114] 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.

[115] 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).

[116] Pacheco, Peter. *Parallel Programming with MPI*. Morgan Kauffman, 1996.

[117] Parks, G. T. and I. Miller. "Selective Breeding in a Multiobjective Genetic Algorithm." In Eiben et al. [40].

[118] 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.

[119] 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.

[120] Pohlheim, Hartmut. *Genetic and Evolutionary Algorithm Toolbox for use with MATLAB*. Technical Report, Technical University Ilmenau, 1998.

[121] Poloni, Carlo. "Genetic Algorithms in Engineering and Computer Science." In Winter et al. [160], 397–416.

[122] Porto, William, editor. *Proceedings of the 1997 (4th) IEEE International Conference on Evolutionary Computation*, Piscataway, NJ: IEEE, April 1997.

[123] 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).

[124] 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.

[125] 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.

[126] 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. ?? – ?? Cambridge, MA: MIT Press, 1995.

[127] Sammon, J. W. "A Nonlinear Mapping for Data Structure Analysis," *IEEE Transactions on Computers*, *C-18*(5):401–408 (1969).

[128] 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.

[129] Schaffer, J. David. "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms." In Grefenstette [62], 93–100.

[130] Schwefel, Hans-Paul. *Evolution and Optimum Seeking*. New York: John Wiley & Sons, Inc., 1995.

[131] 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).

[132] 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.

[133] Srinivas, M. and Lalit M. Patnaik. "Genetic Algorithms: A Survey," *COMPUTER*, *27*(6):17–26 (June 1994).

[134] Srinivas, N. and Kalyanmoy Deb. "Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms," *Evolutionary Computation*, *2*(3):221–248 (1994).

[135] Stanley, Timothy J. and Trevor Mudge. "A Parallel Genetic Algorithm for Multiobjective Microprocessor Design." In Eshelman [43], 597–604.

[136] Steuer, Ralph E. *Multiple Criteria Optimization: Theory, Computation, and Application*. New York: John Wiley & Sons, 1986.

[137] Surry, Patrick D., et al. "A Multi-Objective Approach to Constrained Optimisation of Gas Supply Networks: The COMOGA Method." *Evolutionary Computing: AISB Workshop* edited by T. C. Fogarty, 166–180, Springer-Verlag, 1995.

[138] Syswerda, Gilbert and Jeff Palmucci. "The Application of Genetic Algorithms to Resource Scheduling." *Proceedings of the Fourth International Conference on Genetic Algorithms*. 502–508. San Francisco, CA: Morgan Kaufmann Publishers, Inc., 1991.

[139] 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.

[140] Tamaki, H., et al. "Generation of a Set of Pareto-Optimal Solutions by Genetic Algorithms." *Transactions of the Society of Instrument and Control Engineers31*. 1185–1192. 1995.

[141] Tamaki, Hisashi, et al. "Multi-Objective Optimization by Genetic Algorithms: A Review." In Thomas Bäck and Zbigniew Michalewicz and Hiroaki Kitano [143], 517–522.

[142] 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.

[143] 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.

[144] Thomas Bäck and Zbigniew Michalewicz and Hiroaki Kitano, editor. *Proceedings of the Third IEEE Conference on Evolutionary Computation*, Piscataway NJ: IEEE Service Center, 1996.

[145] Todd, David S. and Pratyush Sen. "A Multiple Criteria Genetic Algorithm for Containership Loading." In Bäck [7], 674–681.

[146] Torán, Jacobo. *Lectures on Parallel Computation*, chapter P-Completeness, 177–196. Cambridge, Great Britain: Cambridge University Press, 1993.

[147] Tsoi, Effie, et al. "Hybrid GA/SA Algorithms for Evaluating Trade-off Between Economic Cost and Environmental Impact in Generation Dispatch." In Fogel [44], 132–137.

[148] Valenzuela-Rendón, Manuel and Eduardo Uresti-Charre. "A Non-Generational Genetic Algorithm for Multiobjective Optimization." In Bäck [7], 658–665.

[149] 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.

[150] Van Veldhuizen, David A., et al. "Finding Improved Wire-Antenna Geometries with Genetic Algorithms." In Fogel [45], 102–107.

[151] Vemuri, V. Rao and Walter Cedeno. "A New Genetic Algorithm for Multi Objective Optimization in Water Resource Management." In Fogel [44], 495–500.

[152] 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.

[153] 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).

[154] 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.

[155] Voigt, Hans-Michael, et al., editors. *Parallel Problem Solving from Nature - PPSN IV*, 1996.

[156] Wallace, David R., et al. "Design Search Under Probabilistic Specifications Using Genetic Algorithms," *Computer-Aided Design*, *28*(5):405–421 (1996).

[157] Weile, D. S., et al. "Multiobjective Synthesis of Electromagnetic Devices Using Nondominated Sorting Genetic Algorithms." *Proceedings of the 1996 IEEE International Symposium on Antennas and Propagation*. 592–595. 1996.

[158] Whitley, Darrell, et al. "Evaluating Evolutionary Algorithms," *Artificial Intelligence*, *85*:245–276 (1996).

[159] Wienke, Dietrich, et al. "Multicriteria Target Vector Optimization of Analytical Procedures Using a Genetic Algorithm," *Analytica Chimica Acta*, *265*:211–225 (1992).

[160] 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.

[161] Winter, G., et al., editors. *Genetic Algorithms in Engineering and Computer Science*. Chichester: Wiley & Sons, 1995.

[162] Wolpert, David H. and William G. Macready. "No Free Lunch Theorems for Optimization," *IEEE Transactions on Evolutionary Computation*, *1*(1):67–82 (April 1997).

[163] Yang, Xiaofeng and Mitsuo Gen. "Evolution Program for Bicriteria Transportation Problem." *Proceedings of the 16th Annual Conference on Computers and Industrial Engineering27*. 481–484. Pergamon, 1994.

[164] 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.

[165] Yoshimura, Kazuyuki and Ryohei Nakano. “Genetic Algorithm for Information Operator Scheduling.” In Fogel [45], 277–282.

[166] Zhou, Gengui and Mitsuo Gen. “Evolutionary Computation on Multicriteria Production Process Planning Problem.” In Porto [121], 419–424.

[167] 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.

[168] Zitzler, Eckart and Lothar Thiele. “Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Case Study.” In Eiben et al. [40].

12 Things to Think About

- EA approach competitiveness as opposed to other paradigms? (performance, e and e)
- Note difference between actual application vs numeric test example? (characteristics)
- What kind of random search can be used to compare against EAs on any proposed MOP test suite? (NFL theorem)
- Only progressive talks about involving DM; any technique could theoretically be used in DM involvement. Talk about real-world implications of this?
- Selection is discussed in C.2.1 of the HEC. Do I want to include a discussion of that before techniques, then reference it? (for what each technique is “good” for)
- What about talking about larger population size when more objectives are being considered; sharing must consider a larger space each time; thus, does size increase exponentially?
- What about comparing different Pareto techniques?
- For an MOP with a global minimum, how about f1 = x2, f2 = y2, and f3 = z3? That should work
- General NP-complete formulation?
- Page numbers for Ryan in bibliography
- check the slash-pwhatever before a period?

- talk about solution stability and robustness per ignizio
- do more objectives require more members in the population?
- refigure the deme complexity
- Allude to Horn's unitation problem
- Get current ref from or lit, like a summary paper, from Deckro

13 NP-Complete MOPs

13.1 Knapsack Problem

Given: A finite set of items $U = \{u_1, \ldots, u_m\}$; for each $u \in U$ there is an associated weight $w(u) \in Z^+$, profit $p(u) \in Z^+$, and maximum weight (W) the knapsack can hold. For some subset $U' \subseteq U$:
$F = (f_1(U'), f_2(U'))$, where

$$\begin{aligned} f_1(U') &= \max \sum_{u \in U'} p(u) \ , \\ f_2(U') &= \min \sum_{u \in U'} w(u) \end{aligned} \tag{22}$$

subject to $\sum_{u \in U'} w(u) \leq W$.

13.2 Knapsack 2

Note: This problem is presented in Zitzler and Thiele [167].

Given: A finite set of m items and k knapsacks, where p_{ij} is the profit of item j in knapsack i, w_{ij} is the weight of item j in knapsack i, and c_i is the capacity of knapsack i. Find a vector $\mathbf{x} = (x_1, \ldots, x_m) \in \{0,1\}^m$, such that $\forall i \in \{1, \ldots, k\} : \sum_{j=1}^{m} w_{ij} x_j \leq c_i$ and for which $f(\mathbf{x}) = (f_1(\mathbf{x}, \ldots, f_k(\mathbf{x}))$ is maximum, where

$$f_i(\mathbf{x}) = \sum_{j=1}^{m} p_{ij} x_j \ , \tag{23}$$

and where $x_j = 1$ if and only if item j is selected.

13.3 Traveling Salesman Problem

Given: A finite set of cities $C = \{c_1, \ldots, c_m\}$; a distance $d_1(c_i, c_j)\ in\ \mathbb{Z}^+$ for each pair of cities $c_i, c_j \in C$; a distance $d_2(c_i, c_j) = \sqrt{(c_{i1} - c_{i2})^2 + (c_{j1} - c_{j2})^2}$ for each pair of cities $c_i, c_j \in C$; and a permutation $< c_{\pi(i)}, \ldots, c_{\pi(m)} >$, $\pi : [1 \ldots m \mapsto$

$1 \ldots m]$.
$F = (f_1(C), f_2(C))$, where

$$\begin{aligned} f_1(C) &= \min \sum_{i=1}^{m-1} d_1(c_{\pi(i)}, c_{\pi(i+1)}) + d_1(c_{\pi(m)}, c_{\pi(1)}) \ , \\ f_2(C) &= \min \sum_{i=1}^{m-1} d_2(c_{\pi(i)}, c_{\pi(i+1)}) + d_2(c_{\pi(m)}, c_{\pi(1)}) \ . \end{aligned} \tag{24}$$

13.4 Vehicle Routing Problem

Given: A finite set of cities $C = \{c_1, \ldots, c_m\}$ and edges $E = \{e_1, \ldots, e_n\}$, where each edge represents a possible path between some pair of cities $c_i, c_j \in C$; each edge e_i has an associated time (e_{it}) and cost (e_{ic}) associated with it. Each pair of cities $c_i, c_j \in C$ may be connected via multiple edges.
$F = (f_1(C, E), f_2(C, E))$, where

$$\begin{aligned} f_1(C, E) &= \min \sum_{i=1}^{m}) e_{it} \ , \\ f_2(C, E) &= \min \sum_{i=1}^{m} e_{ic} \ . \end{aligned} \tag{25}$$