# TECHNICAL RESEARCH REPORT

The Measure of Pareto Optima: Applications to Multi-objective Metaheuristics

*by Mark Fleischer*

**CSHCN TR 2002-17**
**(ISR TR 2002-32)**

# The Measure of Pareto Optima
## Applications to Multi-Objective Metaheuristics

M. Fleischer*

Institute for Systems Research
University of Maryland, College Park
mfleisch@isr.umd.edu

**Abstract.** This article describes a set function that maps a set of Pareto optimal points to a scalar. A proof is presented that shows that the maximization of this scalar value constitutes the necessary and sufficient condition for the function's arguments to be maximally diverse Pareto optimal solutions of a discrete, multi-objective, optimization problem. This scalar quantity, a hypervolume based on a Lebesgue measure, is therefore the best metric to assess the quality of multiobjective optimization algorithms. Moreover, it can be used as the objective function in simulated annealing (SA) to induce convergence in probability to the Pareto optima. An efficient algorithm for calculating this scalar and analysis of its complexity is presented.

## 1 Introduction

This article describes a measure theoretic approach for defining a set function that can be utilized for solving *multi*-objective optimization problems (MOPs). Zitzler *et al.* introduced the foundation for this set function in the following passage:

> In the two dimensional case each Pareto optimal solution $\mathbf{x}$ covers an area, a rectangle, defined by the points $(0, 0)$ and $(f_1(\mathbf{x}), f_2(\mathbf{x}))$. The union of all rectangles covered by the Pareto optimal solutions constitutes the space totally covered, its size is used as measure. This concept may be canonically extended to multiple dimensions [1].

This article embellishes this notion of a set-cover measure by 1) extending it to an arbitrary number of dimensions, 2) rigorously proving that the maximization of the associated set function's scalar output is the necessary and sufficient condition for its arguments to be Pareto optimal solutions to a multi-objective

* The author was supported in part by the Center for Satellite and Hybrid Communications Networks in the Institute for Systems Research at the University of Maryland, College Park, and through collaborative participation in the Collaborative Technology Alliance for Communications & Networks sponsored by the U.S. Army Research Laboratory under Cooperative Agreement DAAD19-01-2-0011 and the National Aeronautics and Space Administration under award No. NCC8-235.

optimization problem, and finally, 3) using insights from this proof to develop an efficient algorithm for computing the value of this set function. Analysis of the algorithm's complexity is also provided.

As the reader will no doubt discover, the intuition for this scalar is quite simple, yet its first appearance was surprisingly quite recent (much to the frustration of the author) [1–3]. Although in the two dimensional case (two objective functions) proving the validity of this measure seems almost trivial, for an arbitrary number of objective functions the proof seems less obvious. Presenting a formal proof therefore serves four purposes:

1. it establishes, with mathematical rigor, that the maximum value of the set function is a necessary and sufficient condition for the Pareto optimality of the function's arguments, hence, is *the best*[1] measure for evaluating heuristics that seek to find Pareto optima;
2. it therefore provides a sound mathematical basis for comparisons to other similar measures or approximations;
3. the proof points the way to a simple approach for calculating the measure;
4. it provides a mechanism for generalizing *any* optimization metaheuristic to handle multiple objectives.

With regard to Point 4, this scalar can be used, *e.g.,* as the objective function in simulated annealing (SA). Because it is well-known that SA converges in probability to the global optima ([6]), using this set function as the objective function in SA induces SA to *converge in probability to Pareto optima!*[2]

This article is organized as follows: Section 2 provides background on approaches for solving multi-objective optimization problems and recent results in the literature. This includes a philosophical discussion of the issues surrounding the relative merits of using genetic algorithms (GAs) versus SA. Although some of these issues will be further explored in future work, this discussion provides motivation for what is to follow. Formal definitions of Pareto optimality and other mathematical elements are described in Section 3. Section 4 describes the hypervolume in MOPs, its mathematical characteristics and presents the main results. Section 5 presents an efficient algorithm for computing this scalar and an analysis of its complexity. Finally, Section 6 discusses issues for future research and provides concluding remarks.

## 2 Background

### 2.1 Considerations of GAs vs. SA

Quite a few multi-objective algorithms have been described in recent years often motivated by design optimization problems. Most of these approaches have been

[1] Regarding Points 1 and 2, practicalities may suggest other measures with more utility depending upon the nature of the problem and the algorithm used to solve it (see *e.g.,* [4, 5]).

[2] This is the only mathematically convergent approach to Pareto optima this author is aware of.

based on GAs (see *e.g.,* [7, 8]) although some have been based on SA [9]. While the level of research into multi-objective GAs seems to dominate similar research using an SA approach, SA may provide some untapped potential and have several advantages over GAs for solving MOPs.

One clear advantage of SA is its mathematical convergence properties described earlier. As will become clear later on, using the set function as the objective function in SA forces convergence to points in objective function space that are distinct.[3] This means that SA will converge to solutions with as good a 'spread' as possible. This *diversity* of solutions has been cited in a number of articles as an indicator of good performance of multi-objective optimization algorithms [2, 4, 8].

Notwithstanding the mathematical convergence of SA, GAs offer advantages in problems where some structure exists in the underlying domain. Indeed, the very elements of the GAs, in particular the crossover operator, lend themselves toward propagating those features in a chromosome that tend to be associated with high fitness values [10]. These particular advantages of GAs may however be diminished when compared to an SA-based approach *in the context of MOPs*.

Quite often in MOPs features associated with the underlying structure are masked and confounded by the interplay of several competing objective functions. Pareto optima may correspond to solutions that are not local optima of *any* of the objective functions—they may lie on the sides of hills rather than at their tops or bottoms. In other words, the pre-image of Pareto optima may be scattered throughout the domain space in an apparently haphazard or random manner rendering any structure within it to be of little or no consequence. As such, the *thermodynamic* approach inherent in SA may be more suitable for solving MOPs than GAs.

Before an elegant SA approach for solving MOPs is a realistic possibility, however, an efficient method for calculating this set function value is needed. To date, this has not been done even in the context of GAs although some GA approaches have indirectly utilized this notion of a scalar. Wu *et al.* [4] quantify a *hyperarea difference* metric closely related to the hypervolume based on the Lebesgue measure of the set of dominated points. Fonseca, *et al.* [5] describe the concept of the "attainment surface" which attempts to quantify how different chromosomes contribute to a performance metric linked to this scalar. Other methods involve archiving or updating solutions that are non-dominated [11], hence indirectly maximize this scalar. All of these methods, in various ways, attempt to produce solutions that ultimately maximize the value of this scalar, but avoid dealing with or computing it directly ostensibly for various reasons: either it is not computationally feasible, it is too difficult given some formulations, it can be well approximated, or other indirect methods are simpler (in some sense).

To illustrate why, perhaps, this measure has not been used directly consider the application of the *inclusion-exclusion* formula described in [12] (see also

[3] This depends on how many decision variables relative to Pareto optima are used in SA.

[4]). For sets $F_1, F_2, \ldots, F_n$ (using William's notation here) with $\mu$ the Lebesgue measure:

$$\begin{aligned}\mu\left(\bigcup_{i\leq n} F_i\right) = &\sum_{i\leq n}\mu(F_i) - \sum\sum_{i<j\leq n}\mu(F_i\cap F_j) + \\ &\sum\sum\sum_{i<j<k\leq n}\mu(F_i\cap F_j\cap F_k) \\ &-\cdots + (-1)^{n-1}\mu(F_1\cap F_2\cap\ldots\cap F_n)\end{aligned} \quad (1)$$

*successive partial sums alternating between over- and under-estimates.* [12, p.21].

This rather ugly and unwieldy expression (in terms of computation) is due to the number of combinations of intersection sets the measures of which must be added and subtracted from the sum of the unions. Indeed, Wu *et al.* [13, p. 47] show a closed form solution for their *hyperarea difference* measure, something directly related to the hypervolume described here, that accounts for just three points that was quite "cumbersome" [4, see *e.g.,* Eq.(16) on p. 21]. Accounting for more points with more objective functions further complicates this approach. See also [2–4] for related works on quality metrics. Figure 2 illustrates the potential difficulties in computing this hypervolume that can arise from the topological complications in higher dimensions. It also provides hints for the efficient computation of this scalar inspired by the proof in Section 4 and the area of *computational geometry.*

### 2.2 Computational Geometry

Computation of the hypervolume has a similar flavor to problems in *computational geometry*, a relatively new area of computer science (see *e.g.,* [14]). It turns out that Pareto optimal points constitute a *maximal set* of points (they have identical definitions). In the field of computer science, many algorithms pertaining to maximal sets have been studied. For example, articles have focused on dynamically maintaining a list of maximal points [15, 16]. Unfortunately, the field of computational geometry has focussed on problems that can be visualized or rendered on a computer screen [14, p.2].

Notwithstanding the research on maximal sets, there does not seem to be any literature from the computer science community concerning the hypervolume of space covered by maximal sets even in the basic texts cited earlier. This could be due to that fact that the problem seems too easy or uninteresting, or there is no motivation, or possibly that researchers assume others have already dealt with these problems and issues. Despite looking for some time, no similar results in the computer science literature similar to the ones presented here have been found. This seems to be the state-of-affairs in the optimization community as well except for those references cited herein that refer to Zitzler, *et al.* [1].

## 3 Mathematical Preliminaries and Definitions

Before describing the set function in detail, a mathematical framework is necessary. The following describes the basic elements of this framework and the notion of Pareto optimality.

Let $\mathcal{X} \subset \mathrm{R}^d$ be a *finite* set of $s$ feasible points in $\mathrm{R}^d$ for some MOP with objective functions $f_i,\ i = \{1, \ldots, n\}$ where for each $i$

$$f_i : \mathrm{R}^d \longrightarrow \mathrm{R}^1.$$

Also, let $\mathcal{X}_P \subseteq \mathcal{X}$ be the set of Pareto optima (defined below) in set $\mathcal{X}$ with $p \leq s$ the number of Pareto optima. Thus for MOPs with $n$ objectives, each vector $\mathbf{x} \in \mathcal{X}$ produces $n$ (real) objective function values $\mathbf{f}(\mathbf{x}) = \{f_1(\mathbf{x}), \ldots, f_n(\mathbf{x})\}$ corresponding to a single point $\mathbf{p_x}$ in objective function space. The image of set $\mathcal{X}$ is therefore a finite set of points $\mathbf{p_x} \in \mathrm{R}^n$ and denoted by $S$ with $s' \leq s$ elements. Thus, the vector valued mapping $\mathbf{f}(\mathcal{X}) = S$ is onto and not necessarily one-to-one.

**Definition 1.** [4] *A point* $\mathbf{p_x} = (f_1(\mathbf{x}), \ldots, f_n(\mathbf{x}))$ *is Pareto optimal if for all feasible points* $\mathbf{p} = (f_1, \ldots, f_n) \in S$ *(this means the corresponding points* $\mathbf{x}$ *are feasible), there exists an* $i$ *such that* $f_i(\mathbf{x}) < f_i$[5] *or for all* $i$, $f_i(\mathbf{x}) \leq f_i$ *(this latter case again refers to a single point that dominates all other feasible points).*

**Definition 2.** *A point* $\mathbf{p_x} \in S$ *corresponding to solution* $\mathbf{x}$ *is* **non-dominated with respect to set** $S$ *if and only if for all other points* $\mathbf{p_y} \in S$ *there exists an* $i$ *such that* $f_i(\mathbf{x}) < f_i(\mathbf{y})$ *(assume that if* $S$ *has a single element it is non-dominated with respect to set* $S$*). A set* $S$ *is a* **non-dominated set** *if all points* $\mathbf{p} \in S$ *are non-dominated with respect to set* $S$*.*

## 4 The Hypervolume

The goal of this section is to define the set function that maps a subset (or the entire set) of Pareto optima to a scalar. Let $m$ be the number of arguments $\mathbf{x} \in \mathcal{X}$ of a set function $F$. These $m$ points in $\mathcal{X}$ map to $m$ points $\mathbf{p} \in S$ in objective function space which must then map to a *single* scalar, the hypervolume $\mu$. Equation (2) makes this mapping clear:

$$\left.\begin{array}{lcl} \mathbf{x}_1 \mapsto \{f_1(\mathbf{x}_1), f_2(\mathbf{x}_1), \ldots, f_n(\mathbf{x}_1)\} & = & \mathbf{p}_1 \\ \vdots \;\; \vdots \qquad\qquad \vdots & & \vdots \\ \mathbf{x}_l \mapsto \{f_1(\mathbf{x}_l), f_2(\mathbf{x}_l), \ldots, f_n(\mathbf{x}_l)\} & = & \mathbf{p}_l \\ \vdots \;\; \vdots \qquad\qquad \vdots & & \vdots \\ \mathbf{x}_m \mapsto \{f_1(\mathbf{x}_m), f_2(\mathbf{x}_m), \ldots, f_n(\mathbf{x}_m)\} & = & \mathbf{p}_m \end{array}\right\} \mapsto \mu. \tag{2}$$

[4] This definition is often written in the negative. That is, a solution $\mathbf{x}$ is in the set $P$ of Pareto optimal solutions if there is *no* solution $\mathbf{y}$ such that $f_i(\mathbf{x}) \geq f_i(\mathbf{y})$ and there exists an $i$ where $f_i(\mathbf{x}) > f_i(\mathbf{y})$. Equivalently, $\mathbf{x} \notin P$ if $\exists \mathbf{y} \in \Omega$, where $\forall i, f_i(\mathbf{x}) \geq f_i(\mathbf{y}) \bigwedge \exists i, f_i(\mathbf{x}) > f_i(\mathbf{y})$. Thus, the negation of this statement is used above to define the Pareto optima in positive terms.

[5] Without loss of generality and to make the definition less confusing, minimizing objective functions are assumed here.

Consequently, some function $F : \mathrm{R}^{n\times m} \longrightarrow \mathrm{R}^1$ must be defined. This is made possible by generalizing the concept of optimality using a measure theoretic approach and extending the associated measures of performance to the multi-dimensional case. Zitzler [1] in effect[6] uses the interval length $M - f(\mathbf{x})$ between some upper bound on the objective function values, $M$, and the objective function value as a measure of performance for a given objective function.[7] This interval captures the important feature of a performance measure, the ability to rank solutions according to their desirability. For given $M$, the larger the interval length, the smaller the objective function value, hence, the better the solution.

Generalizing Zitzler's *et al.* [1] notion of interval as a set measure and establish the mappings in (2) requires the following formal definitions. For $n$ objective functions, a solution $\mathbf{x}$ defines the following dominance set and measure.

**Definition 3. Lebesgue Measure of the Deleted Dominated Set:** *Let* $\mathbf{p} = (f_1, f_2, \ldots, f_n)$ *represent a point in objective function space where, without loss of generality,* $i = 1, \ldots, j$ *are indices of minimization functions and* $i = j+1, \ldots, n$ *are the indices of maximization functions. Let* $f_i(\mathbf{x})$ *be a particular value of* $f_i$ *produced by solution* $\mathbf{x}$ *where* $M_i$ *and* $m_i$ *are the upper and lower bounds for minimization and maximization objective functions, respectively. Then the* deleted dominated set $D_{\mathbf{x}} = \{\mathbf{p} : \forall\ i,\ f_i \in ([m_i, f_i(\mathbf{x})] \ \cup\ [f_i(\mathbf{x}), M_i]) \bigwedge \mathbf{p} \neq \mathbf{p}_{\mathbf{x}}\}$[8] *and constitutes a set of points* $\mathbf{p}$ **strictly inferior** *to* $\mathbf{p}_{\mathbf{x}}$. *The Lebesgue measure of this set is* $\mu(D_{\mathbf{x}}) = \left(\prod_{i=1}^{j}[M_i - f_i(\mathbf{x})]\right)\left(\prod_{i=j+1}^{n}[f_i(\mathbf{x}) - m_i]\right)$.

The following lemmas will be useful in proving the main result.

**Lemma 1.** *Given a finite set of points* $S$ *in objective function space, point* $\mathbf{p}_{\mathbf{x}} \in S$ *is dominated if and only if there exists a* $\mathbf{p}_{\mathbf{y}} \in S$ *such that* $\mathbf{p}_{\mathbf{x}} \in D_{\mathbf{y}}$.

*Proof.* This follows directly from application of Definition 3.

**Corollary to Lemma 1:** *Point* $\mathbf{p}_{\mathbf{x}} \in S$ *is non-dominated with respect to* $S$ *if and only if for all* $\mathbf{p}_{\mathbf{y}} \in S$, $\mathbf{p}_{\mathbf{x}} \notin D_{\mathbf{y}}$.

*Proof.* See Appendix A.1.

With several points $\mathbf{x}_1 \ldots \mathbf{x}_m$, the union of the corresponding deleted dominated sets constitutes the set of dominated points defined by a finite number of

---

6 The quote in Section 1 was obviously referring to a maximization problem. Here we extend this notion by defining upper bounds on minimizing objective functions.

7 Stated this way is subtly different than stating that this measure is the size of the set cover of dominated solutions (which it of course is—see [3]). It is this subtle difference that indicates we can use this measure as the objective function in an optimization algorithm provided an efficient algorithm exists to calculate its value.

8 For convenient notation, we assume that if $m_i < f_i(\mathbf{x})$, *i.e.,* where $m_i$ is a lower bound for a maximization function $f_i$, then the interval $[f_i(\mathbf{x}), m_i] = \emptyset$ and similarly for the case where $M_i$ is an upper bound for a minimizing function $f_i$. This notation effectively deals with both minimization and maximization objectives.

Pareto optima. The measure of this set therefore is a measure of performance for MOPs. The following definition makes this clear.[9]

**Definition 4.** *Let $S_m = \{\mathbf{p}_1, \ldots, \mathbf{p}_m\} \subseteq S$, a set of $m$ feasible points in objective function space. Then the dominance set $D_{S_m}$ is the union of the dominance sets of each element of set $S_m$. That is, $D_{S_m} = \bigcup_{i=1}^{m} D_{\mathbf{p}_i}$ and the measure of this set is $\mu(D_{S_m}) = \mu\left(\bigcup_{i=1}^{m} D_{\mathbf{p}_i}\right)$.*

**Lemma 2. Dominance Calculus:** *From the previous definitions the following statements are true for any point sets $A$ and $B$,*

$$D_A \bigcup D_B \equiv D_{A\cup B}$$
$$\text{If } A \cap B \neq \emptyset, \text{ then } D_A \bigcap D_B \equiv D_{A\cap B}.$$

*Proof.* See A.2.

The following definitions and lemmas show important relationships among points in a set $S$ and bounds on objective function values and will be used to prove the main result in Theorem 1. For notational simplicity and without loss of generality, we shall assume all objective functions are to be minimized. The following definitions are needed:

**Definition 5.**
$\mathbf{F}_i = \{m_i, f_i(\mathbf{x}_1), f_i(\mathbf{x}_2), \ldots, f_i(\mathbf{x}_m), M_i\}$*, the set of the $i^{th}$ objective function values among elements in set $S$ and their lower and upper bounds.*
$u_i(\mathbf{p_x})$*, the least upper bound of $f_i(\mathbf{x})$ in set $\mathbf{F}_i$.*
$l_i(\mathbf{p_x})$*, the greatest lower bound of $f_i(\mathbf{x})$ in set $\mathbf{F}_i$.*
$D'_{\mathbf{x}} = \{(f_1, f_2, \ldots, f_m) : \forall i, f_i(\mathbf{x}) < f_i < u_i(\mathbf{p_x})\}$*, the set of points in a minimization problem exclusive to set $D_{\mathbf{x}}$. See Lemma 3 below.*

For example, given $\mathbf{p}_1 = (1,3,2)$ $\mathbf{p}_2 = (4,1,6)$ $\mathbf{p}_3 = (4,5,1)$ with $m_i = 0$ and $M_i = 7$ for all $i$, then $\mathbf{F}_2 = \{0,3,1,5,7\}$, $u_2(\mathbf{p}_1) = 5$, $u_2(\mathbf{p}_2) = 3$ and

$$D'(\mathbf{p}_2) = \{(f_1, f_2, f_3) : 4 < f_1 < 7, \quad 1 < f_2 < 3, \quad 6 < f_3 < 7\}. \tag{3}$$

Figure 1 illustrates set $D'_{\mathbf{p_x}}$ for the two-dimensional case and the relationships of the definitions above. These will help to clarify elements of the proof. Notice that the shaded area indicated by hash marks associated with $\mathbf{p_x}$ shows a set of points exclusive to $D_{\mathbf{p_x}}$ that add to the measure of set $S_m$.

The following lemma is a key element in proving the main result and provides the basic idea behind the algorithm described in Section 5.

[9] Note that $\mathbf{p_x} \equiv (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x}))$ and will often be denoted using the simpler notation $(f_1, f_2, \ldots, f_m)$ and $\mathbf{p}$ where it is sufficiently clear that we mean $(f_1(\mathbf{x})\ldots)$ and $\mathbf{p_x}$. Also, depending on the context, the dominance set associated with some point $\mathbf{p_x}$ or $\mathbf{p}_i$ will be denoted as $D_{\mathbf{p_x}} \equiv D_{\mathbf{x}}$ and $D_{\mathbf{p}_i} \equiv D_i$, respectively.

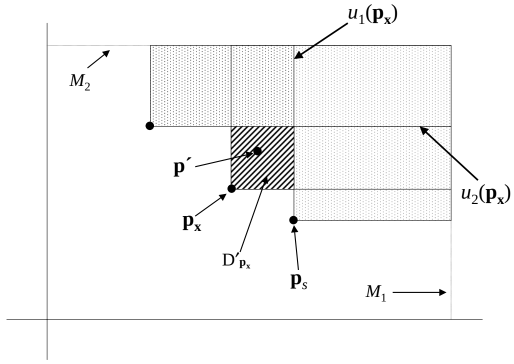



**Fig. 1.** Relationships of Points, $D'_{\mathbf{x}}$ and Upper bounds

**Lemma 3.** *For every non-dominated point* $\mathbf{p_x} \in S$ *there exists a set of points* $D'_{\mathbf{x}} \subset D_{\mathbf{x}}$ *such that for all* $\mathbf{p'} \in D'_{\mathbf{x}}$, $\mathbf{p'}$ *is non-dominated with respect to set* $S \setminus \mathbf{p_x}$ *and such that for all* $\mathbf{p_y} \in S$, $\ D'_{\mathbf{x}} \bigcap D_{\mathbf{y}} = \emptyset$.

*Proof.* See A.3.

**Corollary to Lemma 3:** For all sets $D'_{\mathbf{x}} \subset D_S$, $\ \mu(D'_{\mathbf{x}}) > 0$.

*Proof.* See A.3.

**Lemma 4.** *Given points* $\mathbf{p_x}, \mathbf{p_y} \in S$, $\ \mathbf{p_x}$ *dominates* $\mathbf{p_y}$ *if and only if* $D_{\mathbf{y}} \subset D_{\mathbf{x}}$.

*Proof.* See Appendix A.4.

**Corollary to Lemma 4:** If $\mathbf{p_x}$ dominates $\mathbf{p_y}$ then $\mu\left(D_{\mathbf{x}} \cup D_{\mathbf{y}}\right) = \mu\left(D_{\mathbf{x}}\right) > \mu\left(D_{\mathbf{y}}\right)$.

*Proof.* See Appendix A.4.

Theorem 1 shows that the measure of set $S_m$ achieves its maximum value if and only if points $\mathbf{p} \in S_m$ are Pareto optimal.

**Theorem 1.** *Given set* $S_m$ *of* $m$ *points in objective function space in an MOP with* $p$ *Pareto optimal solutions, let* $M_i$ *be the given bounds for* $f_i$ *(for the sake of clarity and without loss of generality, we assume each objective function is to be minimized and the* $M_i$ *are therefore upper bounds on* $f_i$*). Let* $F(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m) \equiv \mu(D_{S_m})$ *be a set function mapping a subset of points* $S_m \subseteq S$ *to the Lebesgue measure of the dominance set. Then the following are true:*

**Case 1** $(m < p)$**:** *If* $F$ *is at its maximum value then all* $m$ *points in* $S_m$ *are Pareto optimal and for all* $\mathbf{p}_k, \mathbf{p}_l \in S_m$, $\ k \neq l \Rightarrow \mathbf{p}_k \neq \mathbf{p}_l$.

**Case 2** $(m \geq p)$**:** $F$ *is at its maximum value* **if and only if** *there is a subset* $S' \subseteq S_m$ *of size* $p$ *such that all* $\mathbf{p} \in S'$ *are Pareto optimal and for all* $\mathbf{p}_k, \mathbf{p}_l \in S'$, $\ k \neq l \Rightarrow \mathbf{p}_k \neq \mathbf{p}_l$.

*Proof.* See B.1.

## 5 Calculating the Hypervolume

As noted earlier, calculating the hypervolume based on the inclusion-exclusion formula can be quite messy. The basic idea behind the approach describe here stems directly from Lemma 3, its corollaries, and Theorem 1: the algorithm successively lops off hybercubes containing points in sets $D'_{\mathbf{p_x}}$ and adds its volume to a partial sum. This 'lopping off' procedure continues until there is nothing left with positive measure.

The algorithm works by storing the original set of points $S$ in a list $L$. The lopped off hypercube is 'removed' from $L$ by the computation of new points created by its 'removal' using the `SpawnData` procedure. The 'spawned' points that are nondominated with respect to the remaining points in $L$ are added to $L$. The size of $L$ therefore grows and shrinks as this process continues inevitably halting when the last vector's volume is added to the partial sum.

This procedure avoids the necessity of dealing with intersection sets and works for an arbitrary number of objective functions and points. In the following pseudocode, two data structures, *List* and *SpawnData*, hold the original and spawned vectors respectively, and *Size* equals the number of vectors in *List*.

**The LebMeasure Algorithm**

```
Initialize:   LebMeasure = 0.0;
              newSize = Size;
while(newSize > 1) do: {
        lopOffVol := 1.0;
        get first vector p_1 in List
        for(i = 0; i < n; i++) do {
              b_i : = getBoundValue(f_i(x_1))  = {u_i(p_1), l_i(p_1)}
              spawnVector(p_1, i, b_i); //Add spawned vectors to SpawnData
              lopOffVol *=  |f_i(x_1) − b_i|;
        }
        LebMeasure += lopOffVol;
        delete p_1 from  List
        newSize = ndFilter(List, SpawnData);
              //Check if and List vectors dominate SpawnData vectors
        clear SpawnData
} end of while loop.
lastVol = 1.0;
for(i = 0; i < numMops; i++){
        lastVol ×=  |f_i(x_1) − {m_i, M_i}|;
}
return(LebMeasure);
```

**getBoundValue($f_i(\mathbf{x}_1)$):** This routine compares $f_i(\mathbf{x}_1)$ to the corresponding elements in vectors 2 through *Size* and returns either $u_i(\mathbf{p}_1)$ or $l_i(\mathbf{p}_1)$ depending on whether $f_i$ is to be minimized or maximized, respectively and assigns this value to $b_i$ (note the notation in the pseudocode). Its time complexity is therefore $(L-1)n$ where $L$ is the number of original vectors in *List*. Thus, in the first while loop, the time complexity is $(m-1)n$.

**spawnVector($\mathbf{p}_1, i, b_i$):** This routine creates the following $n$ vectors (is executed $n$ times) based on the removal of $\mathbf{p}_1$ from *List* (based on maximizing objectives):[10]

$$\begin{array}{l} \mathbf{p}_{11} = \{l_1(\mathbf{p}_1),\ f_2, \ldots, f_n\} = \{b_1,\ f_2, \ldots, f_n\} \\ \mathbf{p}_{12} = \{f_1,\ l_2(\mathbf{p}_1), \ldots, f_n\} = \{f_1,\ b_2, \ldots, f_n\} \\ \quad\vdots \qquad\qquad \vdots \qquad\qquad \vdots \qquad\qquad \vdots \\ \mathbf{p}_{1n} = \{f_1, f_2, \ldots,\ l_n(\mathbf{p}_1)\} = \{f_1, f_2, \ldots,\ b_n\} \end{array} \tag{4}$$

The notation $\mathbf{p}_{1j}$ refers to the $j^{\text{th}}$ spawned vector from $\mathbf{p}_1$ in *List*. These $n$ vectors comprise the *SpawnData* data structure. Note that all vectors in *SpawnData* are non-dominated with respect to *SpawnData*.

**ndFilter:** This routine compares the vectors in *List* to those in *SpawnData* and deletes any vectors in *SpawnData* that are either dominated by a vector in *List* *or* has an $m_i$ or $M_i$ as one of its elements. The *SpawnData* is then inserted into *List* and replaces $\mathbf{p}_1$.

The following example illustrates this algorithm's operation on *List* which produces the following *SpawnData* data structure after the first `while` loop is completed:[11]

$$\text{List} = \begin{bmatrix} 2,\ 2,\ 2 \\ 1,\ 3,\ 1 \\ 1,\ 1,\ 3 \\ 3,\ 1,\ 1 \end{bmatrix} \ \text{SpawnData} = \begin{bmatrix} 1,\ 2,\ 2 \\ 2,\ 1,\ 2 \\ 2,\ 2,\ 1 \end{bmatrix}.$$

The routine initializes *Size* $= 4$. The first `for` loop selects the vector $\{2, 2, 2\}$, computes bounds for each vector element (1 in this case), creates *SpawnData*, and calculates the incremental hypervolume based on the dimensions of the lopped off hypercube, $1, 1, 1$. These are multiplied together to create the partial sum of *lopOffVol* $= 1$ which is added to *LebMeasure*. Vector $\{2, 2, 2\}$ is then deleted from List.

The routine **ndFilter** deletes any vectors in *SpawnData* dominated by the *remaining* vectors in *List* or that have elements equal to $m_i$ or $M_i$ (in that case, the edge of a hypercube would be of zero length). The resulting *SpawnData* is then added to *List* and *newSize* $= 6$. *List* and its *SpawnData* are now

$$\mathit{List} = \begin{bmatrix} 1,\ 2,\ 2 \\ 2,\ 1,\ 2 \\ 2,\ 2,\ 1 \\ 1,\ 3,\ 1 \\ 1,\ 1,\ 3 \\ 3,\ 1,\ 1 \end{bmatrix} \ \mathit{SpawnData} = \begin{bmatrix} 0,\ 2,\ 2 \\ 1,\ 1,\ 2 \\ 1,\ 2,\ 1 \end{bmatrix}.$$

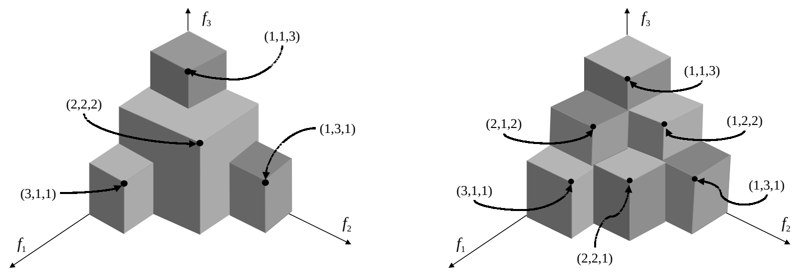
Now vector $\{1, 2, 2\}$ will be deleted from *List*. Although *List* is larger, eventually vectors in *SpawnData* will all contain $m_i$ and/or be dominated by vectors in *List*.

[10] To simplify the expression, we will use $f_i$ for objective function values $f_i(\mathbf{x}_1)$.

[11] Note in this example that all objective functions are to be maximized.

For example, all of the *SpawnData* vectors above will be deleted since the first vector has an element at the lower bound (0), and the second and third vectors are dominated by the remaining vectors in *List*. *List* thus shrinks from 6 vectors to 5 and eventually to 1 breaking the while loop and ending the computation with the last vector's hypervolume being added to *LebMeasure*.

The following figures depict the operation of this algorithm. Initially, *List*




a. Initial *List*. b. *List* After First Iteration.

**Fig. 2.** Evolution of the Hypervolume.

corresponds to Figure 2a. After the first while loop, *List* contains 6 vectors corresponding to Figure 2b where a hypercube with dimensions $1 \times 1 \times 1$ has been "lopped off". Its volume of 1 is added to the partial sum of *LebMeasure*. Continuing with this procedure until it ends yields *LebMeasure* = 11. The reader can verify the result by starting the procedure with any of the four points (*i.e.,* the order of vectors in *List* makes no difference to the value of *LebMeasure*).

### 5.1 Complexity of Computing LebMeasure

Analyzing the long-run behavior of LebMeasure first requires an assessment of the size of a problem instance. Assuming all vectors in *List* are non-dominated, the size of the problem instance involves $m \times n$ values. Including the bounds for each objective, the $(m+1)n$ numbers are sufficient to compute the hypervolume, hence constitutes the size of the problem instance.

The time complexity of the algorithm can be determined by first observing, as in the example, that the size of *List* may grow and shrink at various stages of the algorithm. Thus, the key to analyzing the complexity of LebMeasure is in determining the manner and extent to which spawned vectors are added to and deleted from *List*. Understanding the spawning procedure and how it works is therefore critical towards understanding the complexity of LebMeasure. Again, perusal of the pseudocode and study of Figures 2a,b should help.

Before proceeding further, an important distinction is made among the vectors in *List*—those that are original members of *List*, and those that are spawned from these original members of *List*. Those vectors having only one subscript, *e.g.,* $\mathbf{p}_1$ are the original members of *List* while those with more than one subscript, *e.g.,* $\mathbf{p}_{11}$ are spawned descendants from the original members.

One important property of the spawning procedure is that the total number of spawned vectors in *List* can never exceed $n$, the number of objective functions. This property can help in analyzing the worst-case scenario and is shown in the following arguments. To gain insight into the developing patterns, the first few iterations of the while loop are closely examined.

Without loss of generality, assume each objective is to be maximized. In the first iteration of the while loop, and for the worst-case analysis, assume that all the spawned vectors of $\mathbf{p}_1$ in *SpawnData* are non-dominated with respect to the vectors in *List*. In this case, all of these vectors are added to *List* and $\mathbf{p}_1$ is removed. These spawned vectors are indicated in (4). The length of *List* therefore increases from $m$ to at most $m+n-1$ and evolves thusly:

$$\begin{array}{c} \mathbf{p}_1, \mathbf{p}_2, \ \ldots, \ \mathbf{p}_m \\ \downarrow \\ \mathbf{p}_{11}, \ \ldots, \ \mathbf{p}_{1n}, \ \mathbf{p}_2, \ \ldots, \ \mathbf{p}_m \end{array}$$

requiring the following computational effort:

**Table 1.** Complexity of first while loop.

| Subroutine | Complexity |
|---|---|
| BoundVal | $(m-1)n$ |
| Spawn | $n$ |
| ndFiler | $(m-1)n$ |
| Total | $2(m-1)n+n$ |

Now the spawned vector $\mathbf{p}_{11}$ (see (4)) will itself spawn the following $n$ vectors:

$$\begin{array}{ll} \mathbf{p}_{111} = \{ \ l_1(\mathbf{p}_{11}), & f_2, \quad \ldots, \quad f_n \quad \} \\ \mathbf{p}_{112} = \{ \ l_1(\mathbf{p}_1), \ l_2(\mathbf{p}_1), & \ldots, \quad f_n \quad \} \\ \quad \vdots & \quad \vdots \\ \mathbf{p}_{11n} = \{ \ l_1(\mathbf{p}_1), & f_2, \quad \ldots, l_n(\mathbf{p}_1) \ \} \end{array} . \tag{5}$$

Note that the vector $\mathbf{p}_{111}$, contains $l_1(\mathbf{p}_{11})$ the only new greatest lower bound which, by definition, is less than $l_1(\mathbf{p}_1)$. The other vectors in (5) are the same as those in (4) except that the $f_1$ in (4) has been replaced by the value $l_1(\mathbf{p}_1)$. This means that the vectors in (4) dominate those in (5)—*i.e.,* for all $i$, $\mathbf{p}_{1i}$ dominates $\mathbf{p}_{11i}$. But because $\mathbf{p}_{11}$ has been removed from *List*, no point currently in *List* *necessarily* dominates $\mathbf{p}_{111}$. Consequently, the only possible point in the $\mathbf{p}_{11i}$ generation (*i.e.,* in (5)) that *may* be non-dominated is $\mathbf{p}_{111}$ which for purposes of a worst-case analysis is added to *List* replacing $\mathbf{p}_{11}$. The size of *List* therefore remains at $m+n-1$. The *List* data structure evolves thusly:

$$\begin{array}{c} \mathbf{p}_{11}, \mathbf{p}_{12}, \ldots, \mathbf{p}_{1n}, \mathbf{p}_2, \ \ldots, \ \mathbf{p}_m \\ \downarrow \\ \mathbf{p}_{111}, \mathbf{p}_{12}, \ldots, \ \mathbf{p}_{1n}, \ \mathbf{p}_2, \ \ldots, \ \mathbf{p}_m \end{array}$$

requiring the following computational effort:

**Table 2.** Complexity of while loop 2 to $(m-1)$.

| Subroutine | Complexity |
|---|---|
| BoundVal | $(m-1)$ |
| Spawn | $n$ |
| ndFiler | $(m-1)n$ |
| Total | $(m-1)+(m-1)n+n$ |

Now, $\mathbf{p}_{111}$ is at the top of *List* and may spawn another set of vectors, but again, the same property as described above holds and may yield a maximum of only one non-dominated $\mathbf{p}_{1111}$ and so on. Thus, for all these iterations, the size of *List* remains at $m+n-1$.

Eventually, some descendant of $\mathbf{p}_{11}$, spawns vectors of the form $\mathbf{p}_{1\cdots 1i}$ which all become dominated by vectors in *List* or contain elements at the lower bound of $m_i$ at which point no vectors in *SpawnData* get added to *List* for the next iteration and the last of all the descendants of $\mathbf{p}_{11}$ along with $\mathbf{p}_{11}$ are removed from *List*.

The question arises as to how many successive generations of $\mathbf{p}_{11}$ are possible in the worst-case. Obviously, it cannot be greater than $|List|-1$. Consequently, an upper bound on the number of generations that the first *spawned* vector in *List* may spawn is $m-1$. Thus, the complexity of while loop 3 to $(m-1)$ is the same as given in Table 2.

At this point in the algorithm, the next while loop starts with only $m+n-2$ vectors $\mathbf{p}_{12},\ \ldots,\ \mathbf{p}_{1n},\ \mathbf{p}_2,\ \ldots,\ \mathbf{p}_m$. Now the spawned vector $\mathbf{p}_{12}$ is at the top of *List* and spawns its descendants:

$$\begin{aligned}
\mathbf{p}_{121} &= \{\ l_1(\mathbf{p}_1),\ l_2(\mathbf{p}_1),\ \ldots,\ \ f_n\ \ \} \\
\mathbf{p}_{122} &= \{\ \ f_1,\ \ l_2(\mathbf{p}_{12}),\ \ldots,\ \ f_n\ \ \} \\
&\ \ \vdots \\
\mathbf{p}_{12n} &= \{\ \ f_1,\ \ l_2(\mathbf{p}_1),\ \ldots,\ l_n(\mathbf{p}_1)\ \}
\end{aligned} \tag{6}$$

and $\mathbf{p}_{12}$ is itself removed leaving $m+n-3$ vectors in *List* to which are added the non-dominated vectors in *SpawnData*, some subset of (6). Again, the same pattern is present. That is, for all $k$, $\mathbf{p}_{1k}$ dominates vectors $\mathbf{p}_{12k}$. But now there are only $k-2$ vectors from the first spawned set, $\mathbf{p}_{13}\ldots\mathbf{p}_{1k}$ left in *List* to dominate the $\mathbf{p}_{12i}$ generation. Consequently, it is possible that *two* vectors, $\mathbf{p}_{121}$ and $\mathbf{p}_{122}$, may be non-dominated with respect to vectors in *List*. The size of *List* therefore increases from $m+n-2$ back to at most $m+n-3+2=m+n-1$. *List* becomes:

$$\underbrace{\mathbf{p}_{121},\mathbf{p}_{122},\mathbf{p}_{13},\ldots,\ \mathbf{p}_{1n}}_{n\ \text{vectors}},\ \underbrace{\mathbf{p}_2,\ \ldots,\ \mathbf{p}_m}_{m-1\ \text{vectors}}$$

and its length is again $m+n-1$. The following general observation is made:

*Each time one of the first spawned vectors (*e.g.*, vectors with 2 subscripts such as $\mathbf{p}_{1i}$) is removed from List, the successive generations of the remaining* *vectors of the form* $\mathbf{p}_{1i}$*, add vectors to List in sufficient numbers so that the length of List remains the same.*

The decrease by 1 observed earlier happens at various points, but for purposes of a worst-case analysis can be ignored (it also simplifies the analysis). Thus, each of the first $n$ spawned vectors, have the following number of basic computations where the length of *List* is $m$:

$$[\underbrace{2(m-1)n+n}_{\text{first iteration}}+\underbrace{(m-1)((m-1)+(m-1)n+n)}_{\text{iterations 2 to } m-1}]n$$

which is of order $m^2n^2$. After these calculations, the process begins with the original vector $\mathbf{p}_2$ at the top of *List* with the size decremented. Accounting for this decrease in the length of *List* we have

$$\begin{aligned} T(m,n) \approx \sum_{i=0}^{m-1}(m-i)^2n^2 &= n^2\sum_{i=0}^{m-1}(m-i)^2 \\ &= n^2\left(\frac{m(m+1)(2m+1)}{6}\right) \end{aligned}$$

from the Sum of Squares Formula [17, p.199]. Consequently, the time complexity of LebMeasure is $T(m,n) \in O(m^3n^2)$.

## 6 Future Research and Conclusion

This article described a set function $F(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m) = \mu(D_{S_m})$, a hypervolume on a point set $S_m$, that maps the arguments to a scalar and achieves its maximum value only when these arguments are distinct Pareto optima. Mapping Pareto optima to a scalar unifies the concepts of single and multi-objective optimization. A polynomial algorithm for calculating this hypervolume was also described.

Because this scalar provides necessary and sufficient conditions for the arguments to be Pareto optima, it is also the best measure for evaluating different multi-objective evolutionary algorithms. The many different GAs for example can all be evaluated according to the magnitude of this scalar quantity. By using a scalar quantity to evaluate performance, the average rate of convergence can also be assessed, hence, the performance of evolutionary algorithms quantified using appropriate statistics to estimate the average hypervolume over many independent trials.

Finally, this result shows how *any* multi-objective optimization problem can be put into standard math programming form with a single scalar objective function. As such, many global optimization metaheuristics can be recast to solve multi-objective problems. Simulated annealing, for example, and its parallel variants can be fashioned to converge in probability to Pareto optima. Future research will therefore describe the relative merits of using parallel versions of SA [18], with the hypervolume as the objective function, and various GA implementations.

## Disclaimer

The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory, the National Aeronautics and Space Administration, or the U.S. Government.

## References


1. Zitzler, E., Thiele, L.: Multiobjective optimization using evolutionary algorithms–a comparative case study. In A.E. Eiben, T. Bäck, M.S.H.S., ed.: Proceedings of the Fifth International Conference on Parallel Problem Solving from Nature—PPSN V, Berlin, Germany (1998)
2. M. Laumanns, G.R., Schwefel, H.: Approximating the pareto set: Diversity issues and performance assessment. (1999)
3. Zitzler, E.: Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications. Ph.d. dissertation, Swiss Federal Institute fo Technology (ETH), Zurich, Switzerland (1999) The relevant part pertaining to our Lebesgue measure is discussed in Ch. 3.
4. Wu, J., Azarm, S.: Metrics for quality assessment of a multiobjective design optimization solution set. Transactions of the ASME **123** (2001) 18–25
5. Fonseca, C., Fleming, P.: On the performance assessment and comparison of stochastic multiobjective optimizers. In G. Goos, J.H., van Leeuwen, J., eds.: Proceedings of the Fourth International Conference on Parallel Problem Solving from Nature—PPSN IV, Berlin, Germany (1998) 584–593
6. Mitra, D., Romeo, F., Sangiovanni-Vincentelli, A.: Convergence and finite-time behavior of simulated annealing. Advances in Applied Probability **18** (1986) 747–771
7. Coello, C.: Bibliograph on evolutionary multiobjective optimization. http://www.lania.mx/∼coello/EMOO/EMOObib.html (2001)
8. Veldhuizen, D.V., Lamont, G.B.: Multiobjective evolutionary algorithms: Analyzing the state-of-the-art. Evolutionary Computation **8** (2000) 125–147
9. Czyzak, P., Jaskiewicz, A.: Pareto simulated annealing—a metaheuristic technique for multiple-objective combinatorial optimization. Journal of Multi-criteria Decision Analysis **7** (1998) 34–47
10. Reeves, C.: Modern Heuristic Techniques for Combinatorial Problems. John Wiley & Sons, Inc., New York, NY (1993)
11. Knowles, J.D., Corne, D.W.: Approximating the nondominated front using the pareto archived evolution strategy. Evolutionary Computation **8** (2000) 149–172
12. Williams, D.: Probability with Martingales. Cambridge University Press, Cambridge, England (1991)
13. Wu, J.: Quality Assisted Multiobjective and Multi-disciplinary Genetic Algorithms. Department of mechanical engineering, University of Maryland, College Park, College Park, Maryland (2001) S. Azarm, Ph.D. advisor.
14. Mount, D.: Lecture notes for cmsc 754: Computational geometry. On the web at http://www.cs.umd.edu/ mount/754/ (2000) Department of Computer Science, University of Maryland, College Park.
15. Kapoor, S.: Dynamic maintenance of maxima of 2-d point sets. SIAM Journal on Computing **29** (2000) 1858–1877

16. Frederickson, G., Rodger, S.: A new approach to the dynamic maintenance of maximal points in a plane. Discrete and Computational Geometry **5** (1990) 365–374
17. Hildebrand, F.B.: Introduction to Numerical Analysis. 2nd edn. Dover Publications, Inc., Mineola, NY (1987)
18. Fleischer, M.A.: 28: Generalized Cybernetic Optimization: Solving Continuous Variable Problems. In: Metaheuristics: Advances and Trends in Local Search Paradigms for Optimization. Kluwer Academic Publishers (1999) 403–418

# A Proofs of Lemmas

This appendix contains restatements of the lemmas used in the text followed by their proofs.

## A.1 Proof of Corollary to Lemma 1

*Proof.* From Definition 2, a point $\mathbf{p_y}$ corresponding to solution $\mathbf{y}$ is dominated by solution $\mathbf{x}$ if and only if for all $i, f_i(\mathbf{x}) \leq f_i(\mathbf{y})$ and there is an $i$ such that $f_i(\mathbf{x}) < f_i(\mathbf{y})$. The solution $\mathbf{y}$ therefore satisfies all the criteria for inclusion in set $D_{\mathbf{x}}$. Consequently, the statement from Lemma 1 that $\mathbf{p_x} \in S$ is dominated if and only if there exists a $\mathbf{p_y} \in S$ such that $\mathbf{p_x} \in D_{\mathbf{y}}$ is true and taking the inverse of this statement yields the required result.

## A.2 Proof of Lemma 2

*Proof.* From Definition 4, $D_A = \bigcup_{\mathbf{p}_i \in A} D_{\mathbf{p}_i} \equiv \{\mathbf{p} : (\exists \mathbf{p}_i)(\mathbf{p}_i \in A \wedge \mathbf{p} \in D_{\mathbf{p}_i})\}$ and similarly for $D_B$. Therefore,

$$\begin{aligned} D_A \cup D_B &= \left( \bigcup_{\mathbf{p}_i \in A} D_{\mathbf{p}_i} \right) \cup \left( \bigcup_{\mathbf{p}_i \in B} D_{\mathbf{p}_i} \right) \\ &= \bigcup_{\mathbf{p}_i \in A \vee B} D_{\mathbf{p}_i} \equiv \bigcup_{\mathbf{p}_i \in A \cup B} D_{\mathbf{p}_i} = D_{A \cup B} \end{aligned}$$

For non-empty $A \cap B,\ D_A \cap D_B \equiv \{\mathbf{p} : \mathbf{p}_i \in A \wedge \mathbf{p}_i \in B \Rightarrow \mathbf{p} \in D_{\mathbf{p}_i}\}$ hence by the distributive property we have

$$\begin{aligned} D_A \cap D_B &= \left( \bigcup_{\mathbf{p}_i \in A} D_{\mathbf{p}_i} \right) \cap \left( \bigcup_{\mathbf{p}_i \in B} D_{\mathbf{p}_i} \right) \\ &= \bigcup_{\mathbf{p}_i \in A \wedge B} D_{\mathbf{p}_i} \equiv \bigcup_{\mathbf{p}_i \in A \cap B} D_{\mathbf{p}_i} \\ &= D_{A \cap B} \end{aligned}$$

The requirement that $A \cap B \neq \emptyset$ stems from the fact that for all sets $D_A$ and $D_B$, $D_A \cap D_B \neq \emptyset$. Consequently, for the notational virtues, we require that the point sets $A$ and $B$ have elements in common.

### A.3 Proof of Lemma 3

*Proof.* This proof relies on the fact non-dominated points $\mathbf{p_x}$ have the property that there is an objective function $f_i(\mathbf{x})$ such that it is strictly less than the same objective function evaluated for some other point in a set $S$. Notice that the definition of $D'_{\mathbf{x}}$ in Definition 5 above relies on the least upper bound among all the other points in set $S$. This inequality is therefore maintained for all points in $D'_{\mathbf{x}}$. This may help the reader see that each point $\mathbf{p} \in D'_{\mathbf{x}}$ is non-dominated with respect to the *other* points in set $S$. The formal proof of the statement is most easily appreciated by using contradiction.

Assume set $D'_{\mathbf{x}}$ is defined as above by the non-dominated point $\mathbf{p_x} \in S$ and that one of its elements $\mathbf{p}' = (f'_1, \ldots, f'_m) \in D'_{\mathbf{x}}$ *is* dominated by a point $\mathbf{p}_s = (f_1(\mathbf{x}_s), \ldots, f_m(\mathbf{x}_s)) \in S$. From Definition 2 of non-dominance, for all $i$,

$$\begin{array}{ll} f_i(\mathbf{x}_s) \leq f'_i & \text{and there exists an } i \text{ such that} \\ f_i(\mathbf{x}_s) < f'_i. & \end{array} \tag{7}$$

Now recall that $\mathbf{p_x}$ is a non-dominated point with respect to set $S$. From Definition 2 for all points $\mathbf{p_y} \in S$ there is an $i$ such that $f_i(\mathbf{x}) < f_i(\mathbf{y})$. Let $i^*$ be such an $i$ for point $\mathbf{p}_s$. Therefore, $f_{i^*}(\mathbf{x}) < f_{i^*}(\mathbf{x}_s)$. Thus, $f_{i^*}(\mathbf{x}_s)$ is an upper bound of $f_{i^*}(\mathbf{x})$. From the definition of set $D'_{\mathbf{x}}$, the least upper bound of vector element $f'_i$ in vectors in set $D'_{\mathbf{x}}$ is $u_i(\mathbf{p_x})$. Therefore, $f_{i^*}(\mathbf{x}) \leq f'_{i^*} < u_i(\mathbf{p_x}) \leq f_{i^*}(\mathbf{x}_s)$ contradicting (7) which applies to *all* $i$. Consequently, there is no point in set $S \setminus \mathbf{p_x}$ that dominates point $\mathbf{p}' \in D'_{\mathbf{x}}$. Thus, for all $\mathbf{p} \in S \setminus \mathbf{p_x},\ \mathbf{p}' \notin D_{\mathbf{p}}$ and therefore for all $\mathbf{p} \in S \setminus \mathbf{p_x},\ D'_{\mathbf{x}} \cap D_{\mathbf{p}} = \emptyset$.

**Proof of the Corollary to Lemma 3**

*Proof.* From the definition of $D'_{\mathbf{x}}$ and the fact that we are concerned with discrete optimization problems, each interval in the multi-interval that defines $D'_{\mathbf{x}}$ has a length $u_i(\mathbf{p_x}) - f_i(\mathbf{x}) > 0$, hence, $\mu(D'_{\mathbf{x}}) > 0$.

### A.4 Proof of Lemma 4

**Lemma 4:** Given points $\mathbf{p_x}, \mathbf{p_y} \in S$, $\mathbf{p_x}$ dominates $\mathbf{p_y}$ if and only if $D_{\mathbf{y}} \subset D_{\mathbf{x}}$.

*Proof.* Recall Definition 2 which describes conditions whereby a point $\mathbf{p_x}$ is non-dominated if and only if for all $\mathbf{p_y} \in S$ there exists an $i$ such that $f_i(\mathbf{x}) < f_i(\mathbf{y})$. Its negation therefore implies that a point $\mathbf{p_y}$ *is* dominated if and only if there exists some feasible point $\mathbf{p_x}$ such that for all $i$, $f_i(\mathbf{x}) \leq f_i(\mathbf{y})$ and there exists an $i$ such that $f_i(\mathbf{x}) < f_i(\mathbf{y})$. From Definition 3, the elements of set $D_{\mathbf{x}}$ are such that for all $i$ the following inequalities hold for each $f_i :\ f_i(\mathbf{x}) \leq f_i \leq M_i$. But elements of set $D_{\mathbf{y}}$ are such that $f_i(\mathbf{y}) \leq f_i \leq M_i$. Since for all $i$, $f_i(\mathbf{x}) \leq f_i(\mathbf{y})$, each element of $D_{\mathbf{y}}$ also satisfies the criteria for inclusion in set $D_{\mathbf{x}}$. Thus, $\mathbf{p} \in D_{\mathbf{y}} \Rightarrow \mathbf{p} \in D_{\mathbf{x}}$, hence $D_{\mathbf{y}} \subset D_{\mathbf{x}}$.

To prove that $D_{\mathbf{y}} \subset D_{\mathbf{x}} \Rightarrow \mathbf{p_x}$ dominates $\mathbf{p_y}$, note that when all $\mathbf{p} \in D_{\mathbf{y}} \Rightarrow \mathbf{p} \in D_{\mathbf{x}}$ from Definition 3 all such points are dominated by $\mathbf{p_x}$. It remains to show that $\mathbf{p_y}$ is also dominated by $\mathbf{p_x}$ (note from Definition 3 points $\mathbf{p_x} \notin D_\mathbf{x}$ and $\mathbf{p_y} \notin D_\mathbf{y}$). From the definitions of $D_\mathbf{y}$ and $D_\mathbf{x}$ each multi-interval associated with $D_\mathbf{y}$ is $[f_i(\mathbf{y}), M_i]$ and for $D_\mathbf{x}$ is $[f_i(\mathbf{x}), M_i]$. Since $D_\mathbf{y} \subset D_\mathbf{x}$ it must be that each multi-interval associated with $\mathbf{y}$ is a subset of the corresponding multi-interval for $\mathbf{x}$, hence for all $i$, $f_i(\mathbf{x}) \leq f_i(\mathbf{y})$. Since this is a discrete problem domain and that $\mathbf{p_x} \neq \mathbf{p_y}$ there must be some $\delta > 0$ where for some $i$, $f_i(\mathbf{x}) + \delta \leq f_i(\mathbf{y})$, hence for some $i$, $f_i(\mathbf{x}) < f_i(\mathbf{y})$. Therefore, from Definition 2, $\mathbf{p_x}$ dominates $\mathbf{p_y}$.

**Proof of the Corollary to Lemma 4**

*Proof.* It is a basic result of measure theory that for any two measurable sets $A$ and $B$, $\mu(A \cup B) = \mu(A) + \mu(B) - \mu(A \cap B)$ (lattice property). Therefore it follows that

$$\mu\left(D_\mathbf{x} \cup D_\mathbf{y}\right) = \mu\left(D_\mathbf{x}\right) + \mu\left(D_\mathbf{y}\right) - \mu(D_\mathbf{x} \cap D_\mathbf{y}). \tag{8}$$

But if $\mathbf{p_x}$ dominates $\mathbf{p_y}$ then from Lemma 4, $D_\mathbf{y} \subset D_\mathbf{x}$ and it follows that $D_\mathbf{x} \cap D_\mathbf{y} = D_\mathbf{y}$. Consequently, $\mu\left(D_\mathbf{x} \cap D_\mathbf{y}\right) = \mu\left(D_\mathbf{y}\right)$ and using this in (8) yields the equality.

To show that $\mu\left(D_\mathbf{x}\right) > \mu\left(D_\mathbf{y}\right)$ it is sufficient to show that at least one of the factors in $\mu(D_\mathbf{x})$ is greater than the corresponding factor in $\mu(D_\mathbf{y})$. Since $\mathbf{p_x}$ dominates $\mathbf{p_y}$ then for all $i$, $f_i(\mathbf{x}) \leq f_i(\mathbf{y})$ and there is at least one $i$ where $f_i(\mathbf{x}) < f_i(\mathbf{y})$. Consequently, for all $i$, $M_i - f_i(\mathbf{x}) \geq M_i - f_i(\mathbf{y})$ and at least for one $i$, $M_i - f_i(\mathbf{x}) > M_i - f_i(\mathbf{y})$. Therefore,

$$\prod_{i=1}^{k} \left(M_i - f_i(\mathbf{x})\right) > \prod_{i=1}^{k} \left(M_i - f_i(\mathbf{y})\right),$$

hence $\mu\left(D_\mathbf{x}\right) > \mu\left(D_\mathbf{y}\right)$.

# B Proofs of Theorems

## B.1 Proof of Theorem 1

*Proof.* Note that Case 2 provides a stronger result as both the necessary and sufficient conditions for Pareto optimality are proved. Therefore, for the sake of simplicity, clarity, and brevity, we formally prove only Case 2 as Case 1 becomes sufficiently obvious from the details of proving Case 2. First we prove the implication (somewhat more informally stated) that

**Case 2 ($m \geq p$):** If $F$ is at its maximum value then there exists a subset of size $p$ of its arguments that are Pareto optimal *and* different from one another. Using contraposition, we prove the following equivalent statement:

*If for all possible subsets $S' \subseteq S$ of size $p$ where either not all points $\mathbf{p} \in S'$ are Pareto optimal or there exists $\mathbf{p}_k, \mathbf{p}_l \in S'$ where $k \neq l$ and $\mathbf{p}_k = \mathbf{p}_l$, then $F$ is not maximized.*

Assume all subsets of $S$ of size $p$ have less than $p$ *distinct* Pareto optimal points. Then all subsets can have at most $p-1$ distinct Pareto optimal points. Since there are $p$ Pareto optimal points in the problem instance, then there exists at least one Pareto optimal point $\mathbf{p_x}$ that is not currently in set $S$. Further, there are at least $m-p+1 \geq 0$ elements in $S$ that are not Pareto optimal. Also, bear in mind that some or even all of the points in set $S$ may be non-dominated. Thus, there are a number of different ways in which subsets do not have $p$ Pareto optimal points that must all be carefully considered. Finally, recall from Definition 1 that the $p$ Pareto optimal points dominate *all other* feasible points. Consider the following cases:

**Case 2a:** First, suppose that $S$ has $p$ Pareto optimal points, but that at least one of them has a multiplicity greater than one (*i.e.,* at most $p-1$ distinct Pareto optima). In this case, removing one such Pareto optimal point $\mathbf{p'}$ does not change the measure of set $S$. Consequently, $D_{S\setminus\mathbf{p'}} = D_S$ and therefore, adding point $\mathbf{p_x}$ to $S \setminus \mathbf{p'}$ yields the same number of arguments for function $F$ and

$$D_{S\setminus\mathbf{p'}\cup\mathbf{p}_x} = D_{S\cup\mathbf{p_x}}.$$

But from Lemma 2

$$D_{S\cup\mathbf{p_x}} = D_S \cup D_{\mathbf{p_x}} = D_S \cup (D_{\mathbf{p_x}} \setminus D_S)$$

which are mutually exclusive, hence

$$\mu(D_{S\cup\mathbf{p_x}}) = \mu(D_S) + \mu(D_{\mathbf{p_x}} \setminus D_S)$$

From the corollary of Lemma 3, $\mu(D_{\mathbf{p_x}} \setminus D_S) > 0$ and therefore $\mu(D_S)+\mu(D_{\mathbf{p_x}} \setminus D_S) > \mu(D_S)$ and $F$ with its original arguments is not maximized.

**Case 2b:** Now consider the case where there are less than $p$ Pareto optimal points. In this case, there are $m-p+1 > 0$ non-Pareto optimal points in $S$. Choose one such non-Pareto optimal point $\mathbf{p_y} \in S$. Since there exists $p$ Pareto optimal points that dominate all other feasible points, $\mathbf{p_y}$ is either dominated by a Pareto optimal point already in $S$ or it is dominated by the Pareto optimal point $\mathbf{p_x}$. In the former instance, we have the same situation as in Case 2a, *i.e.,* removing $\mathbf{p_y}$ from $S$ does not change its measure and adding $\mathbf{p_x}$ increases the measure and again, $F$ with its original arguments is not maximized.

In the latter case where $\mathbf{p_y}$ is dominated by $\mathbf{p_x}$, define set $\hat{S} = S \setminus \mathbf{p_y}$ and set $S^* = \hat{S} \cup \mathbf{p_x}$ (*i.e.,* by substituting $\mathbf{p_y}$ with $\mathbf{p_x}$ in $S$). Thus, sets $S$ and $S^*$ have the same number of elements $m$, hence $F$ has the same number of arguments. It is therefore sufficient to prove that $F$ is not maximized by showing that $\mu(D_{S^*}) > \mu(D_S)$ or, equivalently, that

$$\mu(D_{\hat{S}} \cup D_{\mathbf{x}}) > \mu(D_{\hat{S}} \cup D_{\mathbf{y}}) \tag{9}$$

Partition set $D_{\mathbf{x}} \setminus D_{\hat{S}}$ into two mutually exclusive subsets:

$$(D_{\mathbf{x}} \setminus D_{\hat{S}}) = ((D_{\mathbf{x}} \setminus D_{\hat{S}}) \setminus D_{\mathbf{y}}) \cup ((D_{\mathbf{x}} \setminus D_{\hat{S}}) \cap D_{\mathbf{y}}). \tag{10}$$

Since $\mathbf{p_x}$ dominates $\mathbf{p_y}$, from Lemma 4, $D_\mathbf{y} \subset D_\mathbf{x}$, hence the set $D_\mathbf{x}$ exclusive of points in $D_{\hat{S}}$ but with points in $D_\mathbf{y}$, is equivalent to the set of points in $D_\mathbf{y}$ exclusive of points in $D_{\hat{S}}$, *i.e.,*

$$(D_\mathbf{x} \setminus D_{\hat{S}}) \cap D_\mathbf{y}) = (D_\mathbf{y} \setminus D_{\hat{S}}) \tag{11}$$

Substituting (11) into (10) then

$$(D_\mathbf{x} \setminus D_{\hat{S}}) = ((D_\mathbf{x} \setminus D_{\hat{S}}) \setminus D_\mathbf{y}) \cup (D_\mathbf{y} \setminus D_{\hat{S}}). \tag{12}$$

Because this is a union of two mutually exclusive sets then,

$$\mu(D_\mathbf{x} \setminus D_{\hat{S}}) = \mu((D_\mathbf{x} \setminus D_{\hat{S}}) \setminus D_\mathbf{y}) + \mu(D_\mathbf{y} \setminus D_{\hat{S}}). \tag{13}$$

From the Corollary to Lemma 3, the measure of the set $D_\mathbf{x}$ exclusive of all other points in $D_{S'}$ is strictly greater than zero. Consequently, in (13) the term $\mu((D_\mathbf{x} \setminus D_{\hat{S}}) \setminus D_\mathbf{y}) > 0$ and (13) reduces to the inequality

$$\mu(D_\mathbf{x} \setminus D_{\hat{S}}) > \mu(D_\mathbf{y} \setminus D_{\hat{S}}). \tag{14}$$

Note however that $D_\mathbf{x}$ and $D_\mathbf{y}$ can be partitioned into two mutually exclusive sets. Thus,

$$\begin{aligned} D_\mathbf{x} &= (D_\mathbf{x} \setminus D_{\hat{S}}) \cup (D_\mathbf{x} \cap D_{\hat{S}}) \\ D_\mathbf{y} &= (D_\mathbf{y} \setminus D_{\hat{S}}) \cup (D_\mathbf{y} \cap D_{\hat{S}}) \end{aligned}$$

hence their measures are such that

$$\begin{aligned} \mu(D_\mathbf{x}) &= \mu(D_\mathbf{x} \setminus D_{\hat{S}}) + \mu(D_\mathbf{x} \cap D_{\hat{S}}) \\ \mu(D_\mathbf{y}) &= \mu(D_\mathbf{y} \setminus D_{\hat{S}}) + \mu(D_\mathbf{y} \cap D_{\hat{S}}). \end{aligned}$$

Consequently,

$$\mu(D_\mathbf{x} \setminus D_{\hat{S}}) = \mu(D_\mathbf{x}) - \mu(D_\mathbf{x} \cap D_{\hat{S}}) \tag{15}$$

$$\mu(D_\mathbf{y} \setminus D_{\hat{S}}) = \mu(D_\mathbf{y}) - \mu(D_\mathbf{y} \cap D_{\hat{S}}) \tag{16}$$

Substituting (15) and (16) into (14) we obtain

$$\mu(D_\mathbf{x}) - \mu(D_\mathbf{x} \cap D_{\hat{S}}) > \mu(D_\mathbf{y}) - \mu(D_\mathbf{y} \cap D_{\hat{S}}) \tag{17}$$

and adding $\mu(D_{\hat{S}})$ to both sides of (17) yields

$$\mu(D_{\hat{S}}) + \mu(D_\mathbf{x}) - \mu(D_\mathbf{x} \cap D_{\hat{S}}) > \mu(D_{\hat{S}}) + \mu(D_\mathbf{y}) - \mu(D_\mathbf{y} \cap D_{\hat{S}}).$$

From the lattice property of measurable sets we obtain (9), *i.e.,* $\mu(D_{\hat{S}} \cup D_\mathbf{x}) > \mu(D_{\hat{S}} \cup D_\mathbf{y})$. Therefore the function $F$ with its original arguments is not at its maximum value.

To prove the inverse statement associated with Case 2, *i.e.,* that if all points $\mathbf{p} \in S$ are Pareto optimal and $k \neq l \Rightarrow \mathbf{p}_k \neq \mathbf{p}_l$ then $F$ attains its maximum value can be proved by contradiction. Assume $S$ has a subset $S'$ of size $p$ that are all distinct Pareto optimal points and $F$ is *not* at its maximum value. In this case, the measure $\mu(D_S) = \mu(D_{S'})$. To increase the value of $F$, the measure of some subset $D_{\mathbf{p}_i} \subset D_{S'}$ must be increased. Suppose we increase it by selecting a feasible point $\mathbf{p}^*$ which contains hence enlarges set $D_{\mathbf{p}_i}$. That is, $D_{\mathbf{p}^*} \supset D_{\mathbf{p}_i}$. From Lemma 4 then $D_{\mathbf{p}^*}$ dominates $D_{\mathbf{p}_i}$ leading to the contradiction that $\mathbf{p}_i$ is Pareto optimal.