In: *Proceedings of the fifth international conference on adaptive computing in design and manufacture (ACDM 2002)*

Full Elite Sets for Multi-objective Optimisation

Richard M. Everson, Jonathan E. Fieldsend, Sameer Singh

Department of Computer Science
University of Exeter, Exeter, EX4 4PT, UK.
{R.M.Everson,J.E.Fieldsend,S.Singh}@exeter.ac.uk

Abstract

Multi-objective evolutionary algorithms frequently use an archive of non-dominated solutions to approximate the Pareto front. We show that the truncation of this archive to a limited number of solutions can lead to oscillating and shrinking estimates of the Pareto front. New data structures to permit efficient query and update of the full archive are proposed, and the superior quality of frontal estimates found using the full archive is illustrated on test problems.

1 Introduction

Genetic algorithms and evolutionary techniques have been successfully used for a number of years for the optimisation of single objectives. However, one often desires the optimisation of more than one objective, for example, product performance and cost of production. In [1], for example, multi-objective optimisation is applied to four performance measures of a gas turbine, and in [2] different loads in trusses are the competing objectives to be minimised. Recently different measures of a neural network's performance for financial prediction have been optimised [3]. In all these cases a single solution will seldom optimise all objectives, and the goal of optimisation methods is to discover solutions that lie on the curve (for two objectives) or surface (more than two objectives) that describes the optimal trade-off possibilities between objectives. This surface is known as the Pareto front and solutions lying on it are known as *non-dominated* solutions. A non-dominated solution lying on the Pareto front cannot improve any objective without degrading at least one of the others, and, given the constraints of the model, no solutions exist beyond the Pareto front.

Although evolutionary methods have been used in multi-objective optimisation for many years, there has been recent renewed practical and theoretical interest in multi-objective genetic algorithms (MOEAs) [1, 4, 5, 6]. In a comparative study Zitzler *et al.* [7] show that their Strength Pareto Evolutionary Algorithm (SPEA) outperforms other algorithms on a variety of standard test problems.

Laumanns *et al.* [8] have presented a unified model for multi-objective evolutionary algorithms (UMMEA), of which SPEA is an example. Algorithms in this class extend the notion of *elitism* to multi-objective algorithms by maintaining an archive or frontal set, F_t, of non-dominated solutions, in addition to the usual GA population, B_t. As the evolutionary algorithm proceeds this archive set comprises the current estimate of the Pareto front. It is also actively used in the search process as part of the cross-over process to generate new individuals, rather than being a mere static store of the best solutions found.

For computational reasons the archive is of fixed size in SPEA: if it were allowed to grow without bound the cost of searching it to discover whether new solutions should be added to it becomes prohibitive. However, fixing the size of the archive can lead to defects in the the speed and stability of the SPEA and algorithms like it, manifest in the form of shrinking, oscillating and retreating estimates of the Pareto front. In this paper we describe some of the problems inherent in representing the Pareto front by a limited number of solutions, and we present a novel data structure, the dominated tree, for storing the non-dominated solutions.

2 Background

The notions of non-dominance and Pareto optimality, which we briefly review, are central to most MOEAs. Without loss of generality we assume that the multi-objective problem seeks to minimise D objectives, $y_i = f_i(\mathbf{x})$, $i = 1, \ldots, D$, where each objective depends on $\mathbf{x}$, a P-dimensional vector of parameters. The multi-objective optimisation problem may be succinctly stated as:

$$\text{Minimise} \quad \mathbf{y} = \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_D(\mathbf{x})) \tag{1}$$

$$\text{subject to} \quad e(\mathbf{x}) = (e_1(\mathbf{x}), \ldots, e_J(\mathbf{x})) \geq 0 \tag{2}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_P)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_D)$ and $e_j(\mathbf{x})$, $j = 1, \ldots, J$ are constraints upon the solution.

When there is more than one objective to be minimised it is clear that there may be solutions for which performance on one objective cannot be improved without sacrificing performance on at least one other. Such solutions are said to be *Pareto optimal* [9] and the set of all Pareto optimal solutions is said to form the Pareto front. The notion of *dominance* may be used to make Pareto optimality more precise. A decision vector $\mathbf{u}$ is said to *strictly dominate* another $\mathbf{v}$ (denoted $\mathbf{u} \prec \mathbf{v}$) iff

$$f_i(\mathbf{u}) \leq f_i(\mathbf{v}) \quad \forall\, i = 1, \ldots, D \tag{3}$$

$$\text{and} \quad f_i(\mathbf{u}) < f_i(\mathbf{v}) \quad \text{for some } i \tag{4}$$

Less stringently, $\mathbf{u}$ *weakly dominates* $\mathbf{v}$ (denoted $\mathbf{u} \preceq \mathbf{v}$) iff

$$f_i(\mathbf{u}) \leq f_i(\mathbf{v}) \quad \forall i = 1, \ldots, D \tag{5}$$

A set of M decision vectors $\{\mathbf{w}_i\}$ is said to be a *non-dominated set* (an estimate of the Pareto front) if no member of the set is dominated by any other member:

$$\mathbf{w}_i \not\prec \mathbf{w}_j \quad \forall i, j = 1, \ldots, M \tag{6}$$

```
t := 0
(F_0, B_0, p_0^e) = initialise()
while terminate(F_t, B_t, p_t^e) = false:
    t := t + 1
    F'_t := update(F_{t-1}, B_{t-1})
    F_t := truncate(F'_t)
    p_t^e := adapt(F_t, B_{t-1}, p_{t-1}^e)
    B_t := vary(sample(evaluate(F_t, B_t, p_t^e)))
end
```

Figure 1. The sequential unified multi-objective evolutionary algorithm [8]. F_t denotes the elite archive, B_t the general population and p_t^e the elitism intensity at generation t.

3 UMMEAs

Laumanns *et al.* [8] have described a general framework for multi-objective evolutionary algorithms incorporating elitism. Fig. 1 summarises the sequential UMMEA framework in terms of various stochastic operators. The general genetic population B_0 is initialised, together with the elite set F_0 which is generally initialised to be the empty set. At each generation, the elite set is augmented to form F'_t by incorporating those solutions in B_{t-1} which are not dominated any member of $F_{t-1} \cup B_{t-1}$; also any elements of the F_{t-1} which are dominated by members of B_{t-1} are deleted from F'_t. For computational reasons, the updated frontal set is then truncated (usually to some fixed size). In the SPEA this truncation is achieved by clustering. In the final stage of the algorithm the fitness of individuals is evaluated and used to control the sampling of individuals from both the frontal set and the search population for crossover and mutation to yield an offspring population B_t. The crossover and mutation operators are abstractly represented by the *vary()* operator. In SPEA [10] and a recent extension to SPEA [11] fitness of $\mathbf{x}$ is evaluated according to the number of individuals in F_t dominating $\mathbf{x}$.

The 'elitism intensity', p_t^e, controls the probability that an individual from the elite archive set will be selected for the binary tournament [8]. A straight-forward scheme is to choose $p_t^e = |B_t|/|F_t|$ so that equal numbers of individuals are chosen from B_t and F_t. In [11] a more sophisticated method for sampling from F_t is described which ensures that all regions of the front are equally well represented in the population entering the binary tournament. This prevents the search from concentrating in regions in which there are already many individuals and thus forces exploration along the entire estimated Pareto front.

3.1 Truncation

The archive of elite solutions forms the best estimate of the Pareto front at any stage. It should therefore 'advance' in the sense that no individual in F_t should be dominated by any member of an earlier frontal set, $F_0, \ldots, F_{t-1}$. Informally, we say that an

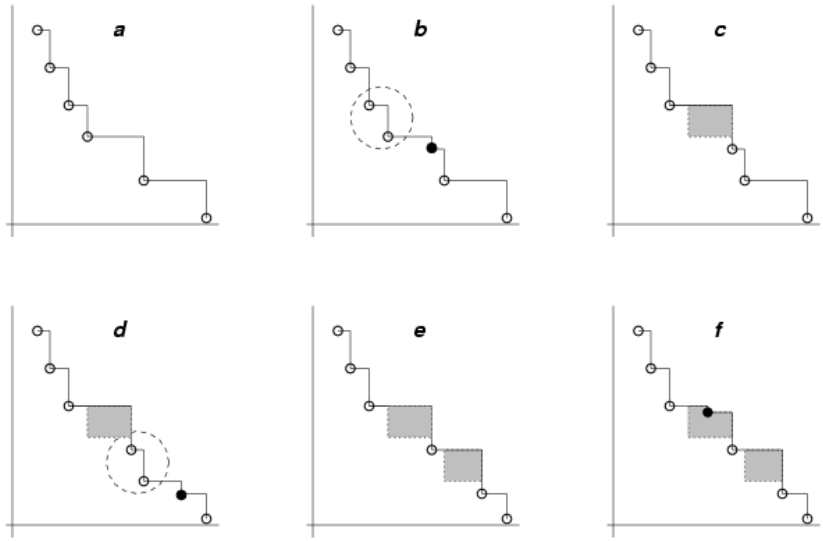

Figure 2. A retreating archive set produced by clustering. Panels illustrate the elite set at successive generations. Elements of F_t are shown as open circles, those entering are shown as filled circles, and elements to be clustered are circled.

individual $\mathbf{x}$ lies *behind* the front if a member of the frontal set dominates $\mathbf{x}$.

The truncation stage of UMMEAs, introduced to control the size of F_t and thus the space required to store it, and, more crucially, the time required to search it, can be deleterious: in particular, the front can fail to advance and instead may retreat or oscillate.

In SPEA, for example, F_t is truncated by clustering members of F_t according to their separation in objective space and then replacing clusters with the member closest to the cluster centroid.

The effect of clustering is illustrated in Fig. 2. Fig. 2a illustrates an archive set with a maximum of $M = 6$ members. In Fig. 2b a new non-dominated member (drawn as a filled circle) has entered the set. Since there are now 7 elements in the frontal set, one must be removed by clustering; the pair of solutions nearest each other form a cluster of two and one of them (chosen at random) is deleted, resulting in the set shown in Fig. 2c. If at a subsequent generation a new element enters the frontal set (Fig. 2d), the clustering process will reduce the frontal set as shown in Fig. 2e to yield a frontal set (Fig. 2f) containing an element that lies behind (is dominated by) elements of the original frontal set (Fig. 2a). Repeated occurrences of this process can lead to the estimated front retreating or, more commonly, oscillating as the front advances in the search stage but retreats during clustering.

This artifact has two principal consequences. First, search time is wasted 'rediscovering' individuals and regions that have been eliminated by clustering. Secondly, numerical simulations show that this oscillation is particularly serious when the estimated front lies close to the true front; the oscillation can prevent convergence to

the true front, leading to poor estimates and difficulties in assessing convergence. Although the artifact has been illustrated here using clustering to limit $|F_t|$ other truncation methods suffer from the same artifacts.

In order to counter this artifact we propose that all non-dominated solutions are retained to form the archive or frontal set F_t. Such a frontal set has the desirable property that it cannot retreat or oscillate. The frontal set therefore always moves towards the true Pareto front. In addition to improving the efficiency of the algorithm, this property also permits sensible convergence criteria to be defined.

4 Dominated trees

As the frontal set contains all the currently non-dominated decision vectors found, it may become very large as the search progresses. Since, at each generation, F_t must be queried to discover whether elements of it are dominated by elements of the search population and vice versa, linear-time query, deletion and insertion can become prohibitive. Data structures to facilitate searching F_t in logarithmic time are the *dominated* and *non-dominated trees.*[1]

Sun & Steuer have also described a modification to quadtrees for maintaining non-dominated sets [12], and the PAES [13] and PEAS [14] algorithms are both based upon recursively dividing objective space into hyper-rectangles that could be supported by quadtrees; however, both algorithms truncate the archive set.

Here it is convenient to regard members of the frontal set and individuals from the search population as points $\mathbf{y}$ in D-dimensional space. Geometrically, finding individuals in F that dominate $\mathbf{y}$ amounts to finding frontal individuals that lie to the 'south-west' or 'left and below' $\mathbf{y}$.

The 'dominates' relation imposes a partial order on individuals. However, since the elements of F are mutually non-dominating, this relation cannot be used directly to construct, for example, a binary tree to enable fast searching.

The dominated tree consists of an ordered list of *composite points* (usually stored as binary tree). Each composite point represents (upto) D elements of F and composite points are defined so that they are ordered by the weakly-dominates relation, $\preceq$. An example of a dominated tree is shown in Fig. 3.

The essential property of dominated trees is that the composite points are ordered:

$$\mathbf{c}_i \preceq \mathbf{c}_j \quad \text{iff } i > j \tag{7}$$

Usually, the stronger condition, $\mathbf{c}_i \prec \mathbf{c}_j \quad$ iff $i > j$, will hold. In addition, if $\mathbf{c}_j \succ \mathbf{c}_i$ then the constituent points of $\mathbf{c}_i$ also dominate $\mathbf{c}_j$. Thus, for example, in Fig. 3 the constituent points of $\mathbf{c}_4, \mathbf{c}_5\mathbf{c}_6$ and $\mathbf{c}_7$ dominate $\mathbf{c}_3$. Note, however, that they do not necessarily dominate the constituent points of $\mathbf{c}_3$, namely $\mathbf{y}_3$ and $\mathbf{y}_6$.

It should be emphasised that the points forming the tree in Fig. 3 do not form a non-dominated set. This is for expository purposes, because non-dominated sets in two dimensions have the peculiar property that listing the points in order of increasing

[1] An example implementation of dominated and non-dominated trees is available from `http://www.dcs.ex.ac.uk/academics/reverson/moea`

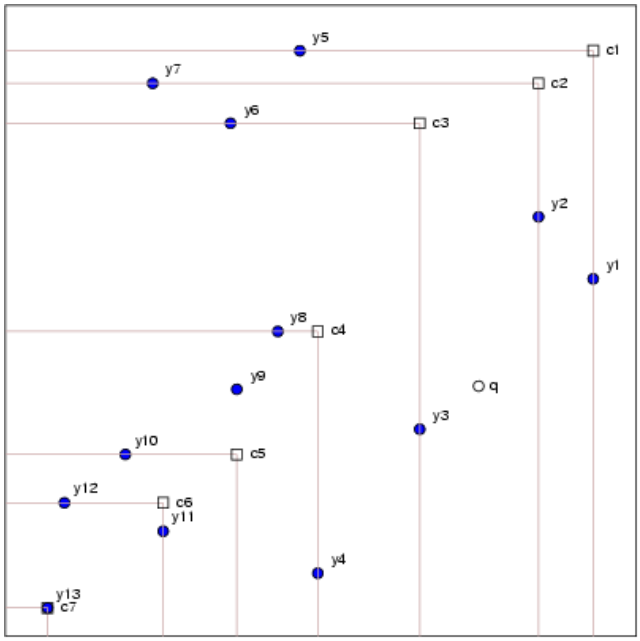

Figure 3. A dominated tree. 13 points $\mathbf{y}_m$ in two dimensions and the composite points $\mathbf{c}_i$ (squares) forming a dominated tree. The open circle, $\mathbf{q}$, marks a query point. Composite nodes listed are: $\mathbf{c}_1 = (\mathbf{y}_1, \mathbf{y}_5) \succ \mathbf{c}_2 = (\mathbf{y}_2, \mathbf{y}_7) \succ \mathbf{c}_3 = (\mathbf{y}_3, \mathbf{y}_6) \succ \mathbf{c}_4 = (\mathbf{y}_4, \mathbf{y}_8) \succ \mathbf{c}_5 = (\mathbf{y}_9, \mathbf{y}_{10}) \succ \mathbf{c}_6 = (\mathbf{y}_{11}, \mathbf{y}_{12}) \succ \mathbf{c}_7 = (\mathbf{y}_{13}, \mathbf{y}_{13})$.

first coordinate (objective), y_1, is equivalent to listing them in order of decreasing second coordinate, y_2. With more than two objectives this is no longer the case and the points illustrated in Fig. 3 are more akin to the general case.

Construction. Construction of a dominated tree from $M = |F|$ points $F = \{\mathbf{y}_m\}_1^M$ proceeds as follows. The first composite point $\mathbf{c}_1$ is constructed by finding the point $\mathbf{y}_m$ with maximum first coordinate; this value forms the first coordinate of the composite point:

$$\mathbf{c}_{1,1} = \max_F \mathbf{y}_{m,1} \tag{8}$$

This point $\mathbf{y}_m$ is now associated with $\mathbf{c}_1$ and deleted from F. Likewise, the second coordinate of $\mathbf{c}_1$ is the maximum second coordinate of the points remaining in F, thus $\mathbf{c}_{1,2} = \max_F \mathbf{y}_{m,2}$. In this manner each coordinate of $\mathbf{c}_1$ is defined in turn. The process is then repeated to construct $\mathbf{c}_2$ and subsequent points until F is empty. Note that in construction of the final composite point, (that is, the composite point that dominates all other composite points) F may be empty before the D coordinates of the composite point have been defined. As illustrated by $\mathbf{c}_7$ in Fig. 3, in this case the $\mathbf{y}_m$ ($\mathbf{y}_{13}$ in Fig. 3) already comprising the composite point serve to define the coordinates in any unfilled dimensions.

Since (except possibly for the dominating composite point) D elements of F are used in the construction of each composite point, the maximum number of composite points is $\lceil M/D \rceil$.

Although we have described the construction of a dominated tree from a static frontal set, dynamic insertion and deletion are straight-forwardly accomplished, together with a rebalancing or 'cleaning' operation [11].

Query. Given a test point $\mathbf{q}$, the properties of dominated trees can be used to discover which points in F dominate $\mathbf{q}$. Although the dominated tree may conveniently be implemented as a binary tree, the query procedure is most easily described in terms of an ordered list of $L = \lceil M/D \rceil$ composite points. First, the list is searched to find the indices h and l of composite points $\mathbf{c}_h$ and $\mathbf{c}_l$ that dominate and are dominated by $\mathbf{q}$ respectively:

$$l = \begin{cases} 0 \text{ if } \mathbf{c}_1 \prec \mathbf{q} \\ \max\{i \,:\, \mathbf{c}_i \prec \mathbf{q}\} \text{ otherwise} \end{cases} \tag{9}$$

and

$$h = \begin{cases} L+1 \text{ if } \mathbf{c}_i \succ \mathbf{q} \\ \min\{i \,:\, \mathbf{c}_i \succ \mathbf{q}\} \text{ otherwise} \end{cases} \tag{10}$$

Also, denote by $\mathbf{c}_H$ the 'least' composite point that strictly dominates $\mathbf{c}_h$:

$$H = \min\{i \,:\, \mathbf{c}_i \prec \mathbf{c}_h\} \tag{11}$$

For the query point illustrated in Fig. 3, $l = 2$, $h = 5$ and $H = 6$. (Note that it is not necessarily true that $H = h + 1$.) Since $\mathbf{c}_h \prec \mathbf{q}$ it is clear that all the constituent points of the composite points $\mathbf{c}_i$ that dominate $\mathbf{c}_h$, $H \leq i \leq L$, also dominate $\mathbf{q}$. (Note that the constituent points of $\mathbf{c}_h$ (and indeed and $\mathbf{c}_i$) that only weakly dominate $\mathbf{c}_h$, need not dominate $\mathbf{q}$; in Fig. 3 $\mathbf{c}_5 \prec \mathbf{q}$, but $\mathbf{y}_9 \not\prec \mathbf{q}$.) Also, since $\mathbf{q} \prec \mathbf{c}_l$ and the constituent points of $\mathbf{c}_l$ have at least one coordinate equal to a coordinate of $\mathbf{c}_l$, it may be concluded that $\mathbf{q}$ is not dominated by any of the constituent points of $\mathbf{c}_1, \ldots \mathbf{c}_l$. Each constituent point $\mathbf{c}_i$ with $l < i < H$ must be checked individually to determine whether it dominates $\mathbf{q}$; in Fig. 3 the points $\mathbf{y}_3, \mathbf{y}_6, \mathbf{y}_4, \mathbf{y}_8, \mathbf{y}_9, \mathbf{y}_{10}$ must be individually checked.

Note that when determining whether $\mathbf{q}$ is to be included in F, $\mathbf{q}$ can be immediately rejected if $h < L$ because it is certainly dominated by at least one of the constituents of $\mathbf{c}_L$.

Since the composite points are arranged as a sorted list, determination of l and h takes $O(\lg(M/D))$ comparisons each. Hence the total number of comparisons required is $O(2\lg(M/D)+K)$, where K is the number of points that have to be checked individually. Clearly, certain configurations of F and $\mathbf{q}$ can result in all elements of F being checked — in linear time. However, such arrangements are seldom encountered in practice and the logarithmic query time permits the use of very large frontal sets.

If it is determined that $\mathbf{q}$ is to be included in F (because it is not dominated by any element of F), those elements of F which are dominated by $\mathbf{q}$ must be identified and deleted from F. Queries about which elements of F are dominated by $\mathbf{q}$ can be answered using the dominated tree, however, it may be inefficient. This is because

although $\mathbf{q} \prec \mathbf{c}_i$ for $i \leq l$, $\mathbf{q}$ need not dominate the constituent points of these $\mathbf{c}_i$ and the constituent points must therefore be checked individually. Thus in Fig. 3, $\mathbf{q} \prec \mathbf{c}_2 \prec \mathbf{c}_1$, but $\mathbf{y}_5$ and $\mathbf{y}_7$ are not dominated by $\mathbf{q}$. The *non-dominated tree* is a data structure which permits this sort of query to be answered efficiently.

Non-dominated are analogous to dominated trees. A non-dominated tree consists of a ordered composite points, $\mathbf{c}_i \succeq \mathbf{c}_j$ iff $i < j$, with the additional property that if $\mathbf{c}_i \succ \mathbf{c}_j$, then the constituent points of $\mathbf{c}_j$ are also dominated by $\mathbf{c}_i$. Construction and querying of non-dominated trees is analogous to dominated trees and they are not discussed further here.

5 Numerical example

We briefly illustrate the benefits of retaining all non-dominated solutions by comparing the performance of SPEA [10] and E-SPEA [11], a modification of SPEA which retains all non-dominated solutions. Although standard benchmark functions for MOEAs exist [7], these are peculiar in that the first objective, y_1 depends solely on the single parameter x_1. We therefore prefer to use the more severe test functions which are defined in terms of the following five base functions:

$$B_1(\mathbf{x}) = \sum_{i=1}^{P} |x_i - \frac{1}{3} \exp i/m^2|^{\frac{1}{2}}$$

$$B_2(\mathbf{x}) = \sum_{i=1}^{P} [x_i - \frac{1}{2} (\cos (10\pi (i/m)) + 1)]^2$$

$$B_3(\mathbf{x}) = \sum_{i=1}^{P} \left|x_i - \sin^2 (i-1) \cos^2 (i-1)\right|^{\frac{1}{2}}$$

$$B_4(\mathbf{x}) = \sum_{i=1}^{P} |x_i - \frac{1}{4} (\cos (i-1) \cos (2 (i-1)) + 2) |^{\frac{1}{2}}$$

$$B_5(\mathbf{x}) = \sum_{i=1}^{P} [x_i - \frac{1}{2} (\sin (1000\pi (i/m)) + 1)]^2$$

where m = 30 and $x_i \in [0, 1]$. We then define the following two-dimensional test functions: $\mathbf{F}_1(\mathbf{x}) = (B_1(\mathbf{x}), B_2(\mathbf{x}))$, $\mathbf{F}_2(\mathbf{x}) = (B_3(\mathbf{x}), B_4(\mathbf{x}))$, $\mathbf{F}_3(\mathbf{x}) = (B_3(\mathbf{x}), B_5(\mathbf{x}))$, the three-dimensional objective function, $\mathbf{F}_4(\mathbf{x}) = (B_1(\mathbf{x}), B_4(\mathbf{x}), B_5(\mathbf{x}))$ and the four-dimensional function $\mathbf{F}_5(\mathbf{x}) = (B_1(\mathbf{x}), B_3(\mathbf{x}), B_4(\mathbf{x}), B_5(\mathbf{x}))$.

A detailed discussion of measures for comparing non-dominated sets is given in [11]. Here we compare two non-dominated sets, A and B, by $\mathcal{V}(A, B) \in [0, 1]$ which measures the fraction of the volume of the minimum hypercube containing both fronts that is strictly dominated by members of A but is not dominated by members of B.

$\mathcal{V}(A, B)$ is defined as follows. For any set of D-dimensional vectors Y, let H_Y denote the smallest axis-parallel hypercube containing Y:

$$H_Y = \left\{\mathbf{z} \in \mathbb{R}^D \ : \ a_i \leq z_i \leq b_i \text{ for some } \ \mathbf{a}, \mathbf{b} \in Y, \ i = 1, \ldots, D\right\} \qquad (12)$$

	$\mathcal{V}(S,E)$	$\mathcal{V}(S,E)$
$\mathbf{F}_1$	0.01	**4.91**
$\mathbf{F}_2$	0.02	**3.92**
$\mathbf{F}_3$	0.08	**8.65**
$\mathbf{F}_4$	0.61	**8.00**
$\mathbf{F}_5$	0.12	**6.13**

Table 1. Comparison of SPEA and E-SPEA on five test functions. $\mathcal{V}(S,E)$ measures the percentage of objective space dominated by the SPEA front but not dominated by the E-SPEA front, while $\mathcal{V}(S,E)$ measures the fraction dominated by the E-SPEA front but not by the SPEA front. Figures in bold are significant at the 2% level (Wilcoxon signed ranks test).

Now denote by $h_Y(\mathbf{y}) : H_Y \mapsto [0,1]^D$ the normalising scaling and translation that maps H_Y onto the unit hypercube. This transformation serves to remove the effects of scaling the objectives. Finally, let

$$D_Y(A) = \{\mathbf{z} \in [1,0]^D \ : \ \mathbf{z} \prec h_Y(\mathbf{a}), \ \mathbf{a} \in A\} \tag{13}$$

be the set of points in the unit cube which are dominated by the normalised elements of A. Then $\mathcal{V}(A,B)$ is defined as

$$\mathcal{V}(A,B) = \lambda\left(D_{A\bigcup B}(A) \setminus D_{A\bigcup B}(B)\right) \tag{14}$$

where $\lambda(A)$ denotes the Lebesgue measure of A.

Despite this awkward definition, $\mathcal{V}(A,B)$ and $\mathcal{V}(B,A)$ are easily calculated by Monte Carlo sampling of $H_{A\bigcup B}$ and counting the fraction of samples that are dominated exclusively by A or B. $\mathcal{V}(A,B)$ is insensitive to spatial distribution of elements in the sets A and B, as well as rescaling of the objective axes. Note, however, that $\mathcal{V}(A,B) \neq \mathcal{V}(B,A)$ since A may dominate parts of B while B dominates parts of A. Also note that if W is a non-dominating set, and $A \subseteq W$ and $B \subseteq W$, $\mathcal{V}(A,B)$ and $\mathcal{V}(B,A)$ may both be positive.

5.1 Results

Table 1 shows a comparison of the fronts found by an implementation of SPEA [10] and E-SPEA, an extended version of ESPEA which retains all non-dominated solutions [11].

The two EAs were each executed 30 times on each test problem, and the resultant non-dominated solutions saved at the end of each run. In the case of E-SPEA these were simply those individuals residing in F at the end of run, whereas an off-line store of the non-dominated solutions discovered by SPEA was kept. For both algorithms the $|B_t| = 80$. In SPEA F_t was limited to 20 individuals (the same number as used in SPEA studies [7]), all of whom were entered into the binary tournament selection for breeding; in E-SPEA 20 individuals for breeding were selected quasi-randomly, ensuring a spatially uniform distribution across the front (see [11] for details). Both algorithms used single point crossover with a crossover rate of 0.8 and a mutation rate

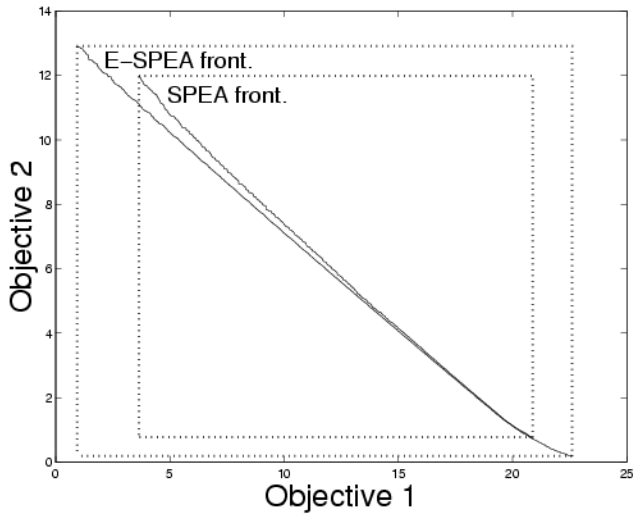

Figure 4. Empirical fronts. The two estimated Pareto fronts generated by SPEA and E-SPEA for test function $\mathbf{F}_1$. The non-dominated individual from 30 separate runs of 2500 generations are plotted.

of 0.01. In each of the 30 different runs E-SPEA and SPEA were initialised from identical decision vector populations, and $x_i^0 \sim U(0,1)$.

Denoting the aggregate frontal set after 2500 generations from SPEA and E-SPEA by S and E respectively, table 1 shows $\mathcal{V}(S,E)$ and $\mathcal{V}(E,S)$. These results indicate that the E-SPEA frontal set lies substantially 'in front' of that found by SPEA, although the fact that $\mathcal{V}(S,E) > 0$ indicates that there are regions were the SPEA front is 'in front'. Fig. 5.1 shows the two estimated Pareto fronts for test function $\mathbf{F}_1$. It is clear that not only is the E-SPEA front substantially 'in front' of the SPEA front, it is also of wider extent: another artifact of truncation is that the front tends to shrink as extremal solutions are discarded by clustering and other forms of truncation.

It is worth noting that although the frontal set may become large, its rate of increase is at most linear, because at most $|B_t|$ individuals may be added at any generation. Indeed Fig. 5 shows that the rate is far smaller than this maximum. Nonetheless Fig. 5 indicates that frontal sets of several hundreds are quickly encountered, necessitating efficient structures for manipulating them.

6 Conclusion

We have argued that truncation of the archive of non-dominated solutions in MOEAs is detrimental to their performance. Truncation is detrimental to both the quality of the approximation to the true Pareto front and to the rate of convergence. The theoretical argument is supported by empirical results on realistic test functions. Truncation is usually introduced for computational expediency, to avoid the expense of searching

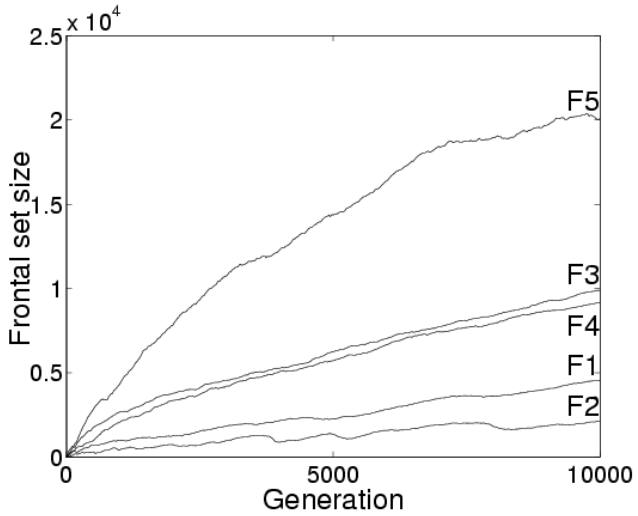

Figure 5. Size of frontal set. Number of non-dominated individuals located by E-SPEA versus generation.

through a large archive of non-dominated solutions at each generation. To relieve the burden of a linear search we have introduced the dominated and non-dominated tree data structures which permit logarithmic time search.

Retaining all non-dominated solutions confers a desirable property on the frontal set, namely that it can only 'advance': solutions in the archive cannot be dominated by any solution from a previous generation. This property, in turn, permits the introduction of robust stopping criteria that have so far been absent from the MOEA literature [11].

References

[1] C.M. Fonseca and P.J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California, 1993. Morgan Kaufmann. URL `citeseer.nj.nec.com/fonseca93genetic.html`.

[2] P. Hajela and C-Y. Lin. Genetic search strategies in multicriterion optimal design. *Structural Optimization*, 4:99–107, 1992.

[3] J.E. Fieldsend and S. Singh. Pareto Multi-Objective Non-Linear Regression Modelling to Aid CAPM Analogous Forecasting. In *Proc. IEEE Intl. Conf. on Computational Intelligence for Financial Engineering*, 2002.

[4] J. Horn, N. Nafpliotis, and D.E. Goldberg. A Niched Pareto Genetic Algorithm

for Multiobjective Optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, volume 1, pages 82–87, Piscataway, New Jersey, 1994. IEEE Service Center. URL `citeseer.nj.nec.com/horn94niched.html`.

[5] J.D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proc. of the First Int. Conf. on Genetic Algorithms*, pages 99–100, 1985.

[6] N. Srinivas and K. Deb. Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation*, 2(3):221–248, 1995. URL `citeseer.nj.nec.com/srinivas94multiobjective.html`.

[7] E. Zitzler, K. Deb, and L. Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195, 2000. URL `citeseer.nj.nec.com/zitzler99multiobjective.html`.

[8] M. Laumanns, E. Zitzler, and L. Thiele. A Unified Model for Multi-Objective Evolutionary Algorithms with Elitism. In *Proc. of the 2000 Congress on Evol. Comp.*, pages 46–53. IEEE, 2000.

[9] D. Van Veldhuizen and G. Lamont. Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art. *Evolutionary Computation*, 8(2):125–147, 2000. URL `citeseer.nj.nec.com/vanveldhuizen00multiobjective.html`.

[10] E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology Zurich (ETH), 1999. Diss ETH No. 13398.

[11] J.E. Fieldsend, R.M. Everson, and S. Singh. Extensions to the Strength Pareto Evolutionary Algorithm. *IEEE Trans. Evol. Comp.*, 2001. URL `www.dcs.ex.ac.uk/people/reverson`. *(submitted)*.

[12] M. Sun and R.E. Steuer. InterQuad: An interactive quad tree based procedure for solving the discrete multiple criteria problem. *European Journal of Operational Research*, 89:462–472, 1996.

[13] Joshua D. Knowles and David Corne. Approximating the nondominated front using the pareto archived evolution strategy. *Evolutionary Computation*, 8(2):149–172, 2000. URL `citeseer.nj.nec.com/knowles00approximating.html`.

[14] D. W. Corne, J. D. Knowles, and M. J. Oates. The pareto envelope-based selection algorithm for multiobjective optimization. In Hans-Paul Schwefel Marc Schoenauer, Kalyanmoy Deb, Günter Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, editor, *Parallel Problem Solving from Nature - PPSN VI 6th International Conference*, Paris, France, 16-20 2000. Springer Verlag. URL `citeseer.nj.nec.com/corne00pareto.html`.