Local Search, Multiobjective Optimization and the Pareto Archived Evolution Strategy

Joshua Knowles and David Corne
Department of Computer Science, University of Reading, UK
J.D.Knowles@reading.ac.uk, D.W.Corne@reading.ac.uk
www.reading.ac.uk/∼ssr97jdk

October 11, 1999

Abstract

The Pareto Archived Evolution Strategy[1] (PAES) [KC99c, KC99b, KC99a], a local search algorithm for multiobjective optimization tasks, is compared with a modern, proven population-based EA, the Strength Pareto Evolutionary Algorithm (SPEA) of Zitzler and Thiele [ZT98, ZDT99, ZT99]. Comparison is carried out with respect to six test functions designed by Deb, each of which is designed to capture and isolate a specific problem feature that may present difficulties to multiobjective optimizers. Statistical techniques introduced previously, and derived from those of Fonseca and Fleming [FF96], are used to process the results to form confidence measures relating to the percentage of the non-dominated front which is covered by each algorithm. These results indicate that, with no attempt at tuning PAES to any of the problems, it outperforms SPEA conclusively on four of the test functions. Some investigation of the mutation rates used by PAES is then carried out. This shows that using a higher mutation rate improves the performance of PAES on two of the test problems so that it outperforms SPEA on a further function in the test suite, $\mathcal{T}_4$, a highly multimodal function. The use of Gray encoding of this problem is also found to improve performance significantly, while it is found to be detrimental on some of the other problems. We find that PAES is unable to compete with the performance of SPEA on function $\mathcal{T}_5$, a strongly deceptive problem, although with higher mutation rates its performance on this problem is shown to be competitive with a range of other multiobjective EAs including NPGA. The results presented provide evidence that local search may be a powerful technique for optimization in multiobjective spaces. To explain this finding we conjecture that there are fewer local optima in multiobjective problem spaces than in their equivalent objective aggregated spaces, and provide some argument to support this claim.

1 Introduction

The field of Evolutionary Multiobjective Optimization [Coe99b, Coe99a, Vel99] is dominated by population-based algorithms that seek to find a non-dominated set of solutions to a problem, from a single optimization run. The prevailing wisdom is that EAs of this sort, often employing recombination too, are well-suited to the task of finding diverse, high-quality sets of solutions to problems which may possess features which have already been investigated in a single-objective context using evolutionary approaches, such as multimodality and deception. Whether this view is supported by sufficient comparison with other techniques or not, there have certainly been a large number of successful applications of MOEAs to real world problems (see [Coe99b] for a comprehensive list) and a steady stream of variations on the theme [HL92, FF93, HNG94, SD94, BW97, PM98] since the pioneering work of Schaffer [Sch84]. Recently, a large subset of the well known MOEA schemes was compared [ZDT99] using a set of test functions of differing difficulty and encompassing problem features hypothesized to cause difficulties to multiobjective optimizers, provided by Deb according to the recommendations given in [Deb98]. The findings of this study were threefold. First, that the test functions were of a sufficient difficulty to differentiate the performance characteristics of the various algorithms. Second, the algorithms were found to fall into a rank ordering which was almost independent of the test function. And third, the addition of elitism to the MOEAs was found to improve their performance significantly on all of the test functions.

In this paper, we take the raw data sets from the study by Zitzler *et al.*[2] and compare them with the performance of a *local search* optimizer, PAES [KC99c, KC99b, KC99a], on the same test functions. In doing this, we have attempted to recreate the same operating modes (i.e. number of evaluations, parameter settings etc.) for the local optimizer as was set for each of the multiobjective EAs, and where this was not possible two or more data sets were collected under different operating modes to establish if any differences in performance could be explained by these different conditions. The aim of these experiments is to establish whether a local search optimizer can cope with the diverse set of test functions which are used in [ZDT99]. Furthermore, we

[1] a brief decription of PAES, electronic copies of publications relating to it, and all of our results are available from http://www.reading.ac.uk/∼ssr97jdk/multi/PAES.html

[2] available from http://www.tik.ee.ethz.ch/∼zitzler/testdata.html

wish to find out if, like the other MOEAs, PAES can be placed at some position in a hierarchy of performance, independently of the test function, or whether it has any different areas of strength or weakness. Finally, a number of further runs are also described, in which some tuning of the local optimizer is carried out. A number of different mutation rates are tried, and the coding of the test functions is changed from binary to Gray. These experiments are carried out to explore the best parameter settings for PAES and to establish how the coding of problems may interact with the search algorithm.

In the next section the test functions used in [ZDT99] are reviewed. Section 3 describes the techniques used for comparing the results achieved by different algorithms. Section 4 provides details of the experiments that were carried out in [ZDT99], explains how these experiments have been replicated for testing PAES, giving the parameter settings used and noting where there are differences in experimental set up. The results of these experiments are tabulated, graphical results are presented, and some discussion is provided. A summary of our experiments and findings is then given. Finally, a concluding section presents a conjecture relating to our findings and indicates lines of possible future research.

2 Test Functions

In [Deb98] a procedure for creating tunable test functions with specific problem features is presented. First, two tasks that any effective MOEA should perform are identified. These are :

1. Guide the search towards the global Pareto-optimal region, and
2. Maintain population diversity in the current non-dominated front.

Next, a number of problem features are identified which can cause difficulty in performing these tasks. Several example test functions are then given to illustrate how test problems incorporating some of these features can be constructed. Six such test functions, constructed by Deb, were used in a comparison of the performance of eight different MOEAs undertaken by Zitzler [ZDT99]. The six functions comprised $\mathcal{T}_1$ with a convex Pareto front, $\mathcal{T}_2$ with a non-convex Pareto front, $\mathcal{T}_3$ having a discontinuous Pareto front, $\mathcal{T}_4$ a multimodal problem with 21^9 Pareto fronts, $\mathcal{T}_5$ a deceptive problem, and $\mathcal{T}_6$ having a non-uniformly distributed search space with solutions non-uniformly distributed along the Pareto front. Each of the functions is a two-objective minimization problem on m parameters. In five of the problems the parameters x_i were coded as a binary string decoded such that $x_i \in [0, 1]$. The remaining function ($\mathcal{T}_5$) also employed a binary chromosome but this time unitation was used to evaluate each of the parameters. The experiments presented in this paper employ identical functions to those presented in [ZDT99] and are coded onto chromosomes using identical numbers of bits to represent each parameter.

3 Statistical Comparison of Multiobjective Optimizers

Proper comparison of the results of two multiobjective optimizers is a complex issue. Several different solutions have been put forward in recent years [SD94, Rud98, VL98, HJ98, SNT+99, SFF99, ZDT99]. However, we use a technique [KC99a] based on a promising method put forward by Fonseca and Fleming [FF96], in which the set of non-dominated solutions generated on an optimization run are taken to define a surface, called the attainment surface, that delineates the objective space into two regions, the dominated region and the non-dominated region. Two optimizers then can be compared with respect to the set of attainment surfaces that they generate over several optimization runs. Specifically, the surfaces can be sampled using lines of intersection, to determine the distribution of these surfaces along the intersection line. A non-parametric statistical test can be used to determine which algorithm has performed best on the sampled part of the objective space, at a given confidence level. By performing many such tests using lines distributed over the whole objective space, an indication of the proportion of the objective space on which one algorithm outperforms another can be computed.

The technique we employ has the advantage that it is completely independent of the actual *numbers* of solutions found, instead only concerning itself with the surface defined by the set of these solutions taken together. In the case of Pareto multiobjective optimization, where we must assume that all non-dominated solutions are to be preferred to all dominated solutions it should be irrelevant how many solutions were found in a particular part of the space; only the attainment surface they define should be relevant. Some other methods [ZDT99] rely on counting the number of nondominated solutions that are *covered* by another nondominated set of solutions. This is not necessarily equivalent to calculating the percentage of the space on which one algorithm covers the other, and can thus give misleading information, favouring particularly algorithms which generate a large number of solutions.

There are disadvantages to the method we employ too, however. First, the non-parametric test cannot determine how much better one algorithm is than another. Second, the number of sampling lines needs to be large to ensure they cover the space sufficiently well. Third, the sampling lines can distort the proportion of the space that they each cover, giving unreliable information. Fourth, it is computationally expensive. On the positive side, it can be used on problems of any number of objectives, it compactly provides statistical decisions as to which algorithm covers more of the space, and can also be used to plot statistically meaningful graphs,

such as the median attainment surface. Improving the technique further is the subject of future work. In this study we use the technique as it is described in [KC99a].

Here, the technique is used to compare pairs of algorithms only, although it can also be used to compare any number of optimizers together. When comparing two algorithms A and B the statistics returned are made up of a pair of numbers $[a, b]$ that represent :

a the percentage of the space on which algorithm A beats algorithm B, and

b the percentage of the space on which algorithm B beats algorithm A

each with respect to a p value of 0.95 using the Mann-Whitney U test [MB94]. The tests carried out in this study all used 500 sampling lines of intersection.

4 Experimental Results

Experimental method

In [ZDT99] eight algorithms were compared on the six Deb test functions. Each algorithm was executed 30 times on each test problem, and off-line a record of all nondominated solutions encountered was kept, and returned as the outcome of each run. For each pair of algorithms A, B and for each of thirty runs, a metric $\mathcal{C}$ was used to compute the proportion of nondominated points in A covered by B and vice-versa. The median and interquartile ranges of the resulting $\mathcal{C}$ values were then plotted using box plots.

The results indicated that on four of the test functions, $\mathcal{T}_1 - \mathcal{T}_3$ and $\mathcal{T}_6$, the Strength Pareto Evolutionary Algorithm (SPEA) [ZT98], generates solution sets which consistently dominate *all* of the other algorithms tested. On test function $\mathcal{T}_4$ SPEA is clearly superior to all other algorithms in terms of the median of the $\mathcal{C}$ value although it no longer consistently beats two of the algorithms, namely NSGA [SD94], and SOEA (a single objective EA run 100 times with a different randomly chosen linear combination of the objectives). However, although SPEA is not consistently better than these two algorithms on this function it does produce a better overall distribution of $\mathcal{C}$ values. On test function $\mathcal{T}_5$, very similar performance can be observed between NSGA and SPEA while SOEA actually does slightly better in terms of the median and the whole distribution of $\mathcal{C}$ values achieved. To summarize these results, one can say that SPEA is the best algorithm overall and is only beaten on one test function, $\mathcal{T}_5$, by one algorithm, SOEA, which was given far more evaluations than any other algorithm (in fact 100 times as many).

Due to the dominance of SPEA alluded to above, PAES is initially tested only against the raw results achieved by SPEA, on each of the test functions. Where PAES does not exhibit clear superiority over SPEA on any test function then further tests against the other algorithms in [ZDT99] are performed, again using the raw data available from the web-site given earlier. This should give an indication of the approximate rank of PAES compared with the other algorithms, without the need for explicitly testing against each of them.

The algorithms in [ZDT99] each employed a population of 100 solutions and were run for 250 generations on each of the test functions. The only exceptions to this were SOEA (using the results from 100 runs) and SPEA which has two populations, an internal population and an external population. Zitzler *et al.* chose to give SPEA an internal population of 80 and external population of 20 out of fairness to the other algorithms. This immediately gives us a problem when we wish to replicate the conditions given to the other algorithms for PAES. Since we would like to be sure of our results we try all of the alternative options and observe the effect of these different conditions. PAES has an internal population of just two since it uses a (1+1) or hillclimbing strategy. However, like SPEA it does have an external population, though it is used only as a comparison set for the purpose of ranking the current and candidate solutions, and is never used as a pool from which selection can occur. All of the algorithms in [ZDT99] have a total population of 100. Therefore, PAES could be given an external population limit of 98 so that it too has a total population of 100. This seems to be the fairest method of comparing these algorithms. However, SPEA was only given an external population of 20. Potentially, this could have hindered its performance (although this set up was chosen presumably to be the best compromise of internal and external population size given a maximum total population of 100) so we also compare PAES to SPEA with an external population of 20.

The number of evaluations carried out in [ZDT99] by each algorithm is 25000 (250×100). The total number of evaluations for all versions of PAES tested is also 25000. All of the algorithms in [ZDT99] were tested with respect to their off-line performance. Zitzler *et al.* note that this changes the relative performance of some algorithms compared to when on-line performance is considered. The performance of VEGA, for example, is improved greatly by considering its off-line performance. Thus, to demonstrate the on-line performance of PAES we include comparison of PAES with SPEA where PAES is only allowed to return the solutions stored in its external archive population at the end of each run, a maximum of 100, in addition to our off-line tests.

A mutation rate of 1% was used with each algorithm, on all problems in [ZDT99]. However, we choose to use a mutation rate (in our initial experiments) of 1 bit per chromosome because in previous experiments with PAES this setting has provided reasonable results. As the EAs in [ZDT99] all employ crossover, and mutation is only a secondary operator, we do not deem it necessary to choose the same mutation rate for PAES since its primary and only method of generating new points is through mutation. Nonetheless, we note that in this

	Algorithm version			
Problem	PAES_on-line	PAES[98]	PAES[20]	PAES[98]Gray
$\mathcal{T}_1$	[47.6, 7.5]	[88.5, 3.6]	[24.9, 3.6]	[0, 97.1]
$\mathcal{T}_2$	[93.5, 0.8]	[86.9, 3.6]	[90.1, 1.1]	[2.3, 48.5]
$\mathcal{T}_3$	[97.6, 1.0]	[95.9, 2.1]	[95.1, 2.6]	[0.3, 77.5]
$\mathcal{T}_4$	[0.0, 100.0]	[0.0, 100.0]	[0.0, 100.0]	[97.7, 0.1]
$\mathcal{T}_5$	[0.0, 100.0]	[0.0, 100.0]	[0.0, 100.0]	—
$\mathcal{T}_6$	[100.0, 0.0]	[100.0, 0.0]	[99.5, 0.0]	[98.1, 0.0]

Table 1: PAES vs SPEA on the six test functions (four different operating modes of PAES were tested)

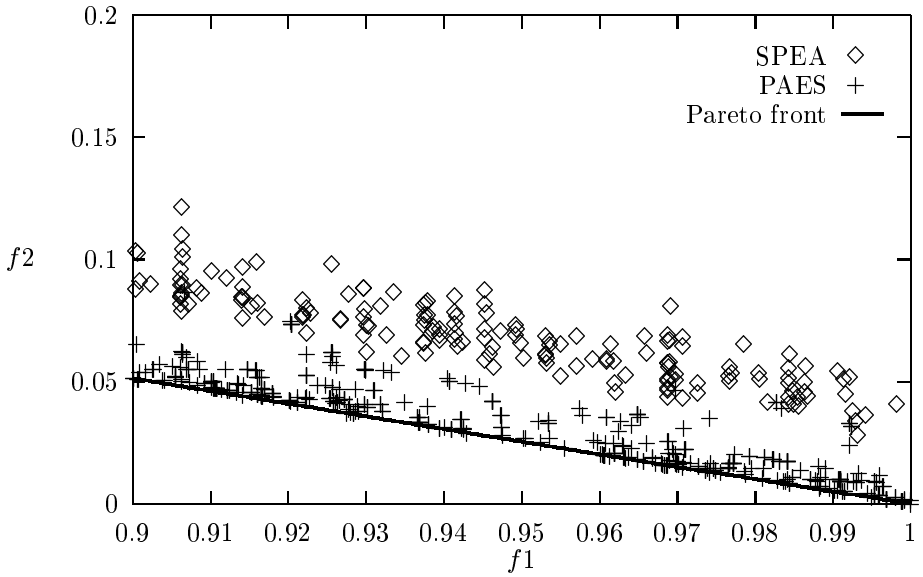

Figure 1: Solutions found from 30 runs of SPEA and PAES to test function $\mathcal{T}_1$ with $0.9 \leq f1 \leq 1.0$

study we are unable to 'tune' the parameters of the MOEAs that we are using for comparison, and accept that this may affect the strength of our conclusions.

PAES also uses a form of crowding for which one parameter must be set. In PAES, the crowding is computed by dividing the objective space recursively into hypercubes. A count of the solutions residing in each hypercube is then performed. The number of recursive divisions, l of the space determines the 'niche size'. In previous experiments with PAES for a two-objective problem, $l = 5$ giving 1024 hypercubes has given good performance, and we use that setting here. Although using this crowding method is not equivalent to the niching used by the algorithms in [ZDT99], it is an integral part of the PAES algorithm and in any case we have shown in previous work [KC99a] that the crowding technique used by PAES is less computationally expensive than niching.

Initial results

The results for these initial experiments are shown in Table 1. PAES_on-line refers to runs in which the *on-line* performance of PAES was examined[3]. PAES[98] refers to PAES with an archive population of 98, and similarly for PAES[20]. PAES[98]Gray uses Gray coding of the parameters in place of binary. The results indicate that PAES outperforms SPEA on functions $\mathcal{T}_1 - \mathcal{T}_3$ and $\mathcal{T}_6$ regardless of its operating mode (with the exception of the Gray coded version). Using Gray codes adversely affects the performance of PAES on functions $\mathcal{T}_1 - \mathcal{T}_3$ but causes PAES to outperform SPEA on function $\mathcal{T}_4$. Otherwise, PAES is outperformed by SPEA on problems $\mathcal{T}_4$ and $\mathcal{T}_5$, the multimodal and deceptive problems respectively. Further investigation of these two problems is given below.

As the results shown above do not indicate the degree to which PAES outperforms SPEA on functions $\mathcal{T}_1 - \mathcal{T}_3$ and $\mathcal{T}_6$, two examples plots are shown. In Figure 1 all of the solutions returned by PAES[98] and SPEA over 30 runs are plotted for function $\mathcal{T}_1$ over a small region of the front (for clarity only). In Figure 2 the median and best attainment surfaces of PAES[98] and SPEA are shown for function $\mathcal{T}_6$. The Pareto optimal lines are also shown. In both plots the extent of PAES's superiority is clear to see.

[3]The on-line version of PAES has an archive population of 100.

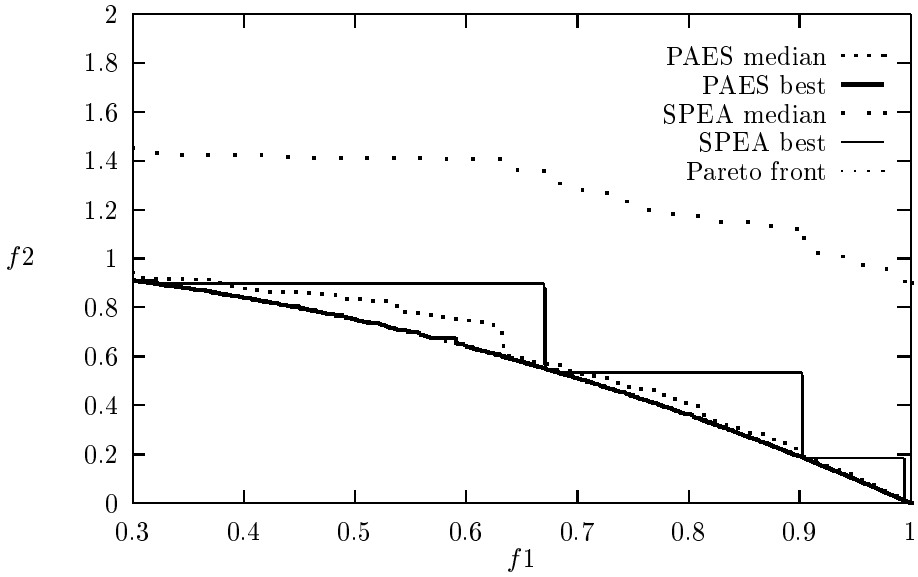

Figure 2: Median and best attainment surfaces achieved by PAES and SPEA to test function $\mathcal{T}_6$

Mutation rate tuning

The results presented above were generated with no tuning of the mutation rate of PAES. On the multimodal test function $\mathcal{T}_4$, where SPEA was found to be superior to PAES using a mutation rate of 1 bit per chromosome, increasing the mutation rate was found to improve the performance of PAES. A mutation rate of 8 bits per chromosome or about 3% was found to give the following statistics [96.8, 0] when comparing PAES with SPEA, in favour of PAES. Improvements in performance were also found on $\mathcal{T}_5$ with increased mutation rates. These are discussed below. On the other test functions using a mutation rate of more than about 1% was found to be extremely detrimental to the performance of PAES, however. This leads us to the conclusion that PAES is relatively sensitive to the mutation rate which is chosen and this can be problem dependent. We return to this point in the conclusion. On function $\mathcal{T}_4$ we also performed a number of longer runs. These show that PAES continues to find better solutions even after 500000 evaluations. A plot showing this is given in Figure 3, where 5 runs of PAES for 25000 evaluations are compared with 5 runs of PAES for 500000 evaluations. The mutation rate was 8 bits per chromosome or 2.67% in both sets of runs.

The deceptive function $\mathcal{T}_5$

Test function $\mathcal{T}_5$ is a deceptive problem made up of 10 deceptive subproblems of 5 bits each, and a 30-bit non-deceptive problem. The 5-bit subproblems are all *fully deceptive* [Whi91] i.e. all schema of order four or less lead towards a sub-optimum which is the genotypic complement of the optimum. The performance of PAES on this problem compared to SPEA at a mutation rate of 1 bit chromosome (=1.25%) is shown in Table 1, and indicates that the distribution achieved by SPEA is better than PAES in 100% of the space. However, the degree to which SPEA is better than PAES is not known. To investigate this, PAES was compared with the other algorithms tested in [ZDT99]. It was found to generate poorer distributions of solutions than SOEA, NSGA, HLGA [HL92] and VEGA but was slightly better than NPGA [HNG94] with a statistic of [39.1, 27.4]. We can conclude that with the initial mutation rate tried, PAES is significantly outperformed by SPEA on this test function, and does not compare favourably with other population-based EAs either. However, increasing the mutation rate to 4 bits per chromosome (=5.0%) increases the performance of PAES on this problem. At this mutation rate comparisons with VEGA and HLGA favour PAES - [100, 0.0] and [90.7, 0.0] respectively - while it is still beaten by NSGA and SPEA. With long runs of up to 5 million evaluations, the solutions found by PAES continue to improve, indicating that with this mutation rate at least, PAES does not converge prematurely. However, even with runs of this length PAES does not approach the performance of SPEA or NSGA using just 25000 evaluations; a statistic of [0.0, 37.9] resulted from comparing the solutions generated by 30 runs of PAES, for 5 million evaluations each, to the original data from SPEA running for just 25000 evaluations.

5 Summary

In this study, a local search optimizer, PAES, has been compared with a modern, proven, population-based EA, SPEA, on a suite of six difficult test functions. With no tuning of the PAES algorithm to any of the test functions, and in several different operating modes, it was found that PAES generated solutions that were

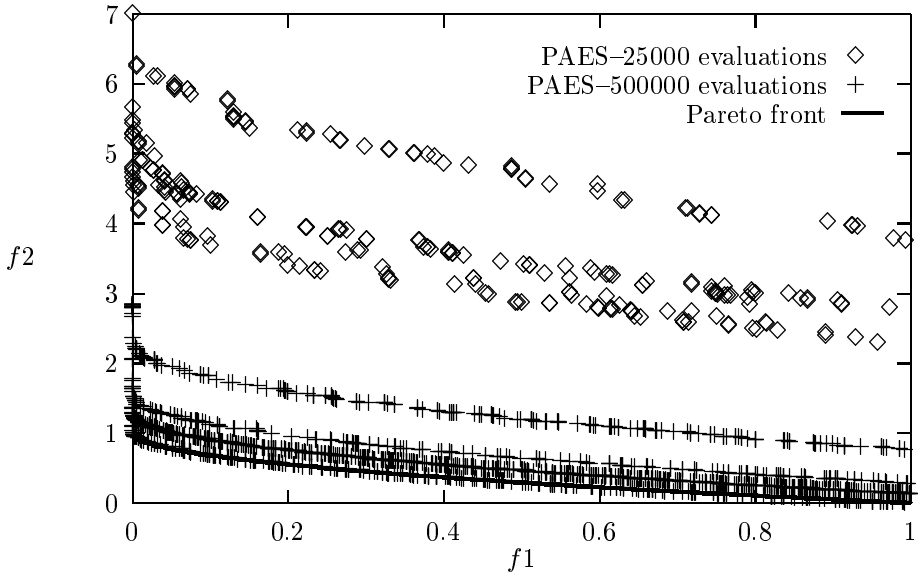

Figure 3: Solutions found to test function $\mathcal{T}_4$ from 5 runs of PAES with 25000 evaluations per run and 500000 evaluations per run

statistically significantly better than those generated by SPEA on four of the test functions. These functions are $\mathcal{T}_1$ and $\mathcal{T}_2$ with convex and non-convex Pareto fronts respectively, $\mathcal{T}_3$ with a discontinuous Pareto-optimal front, and $\mathcal{T}_6$ with both non-uniform density of the search-space and Pareto optimal solutions non-uniformly distributed along the global Pareto front; all 900 bit problems.

On the two remaining functions $\mathcal{T}_4$ and $\mathcal{T}_5$, SPEA was found to generate solutions significantly better than those generated by PAES. These two test problems are difficult examples of multimodality and deception, respectively. The function $\mathcal{T}_4$ has 21^9 local optima and consists of 300 bits, and $\mathcal{T}_5$ has 10 fully deceptive 5-bit subproblems. Although, PAES was not competitive with SPEA on these two functions it was still better than NPGA and FFGA [FF93]

Furthermore, with tuning of the mutation rate of PAES, on $\mathcal{T}_4$ the performance of SPEA was bettered. With longer runs at this mutation rate PAES was shown to evolve solutions to the global Pareto optimum. This result showed that on long runs, PAES has comparable performance to an elitist version of NSGA tested in [ZDT99]. On test function $\mathcal{T}_5$ the performance of PAES was again improved with an increase in the mutation rate. However, the performance of SPEA on $\mathcal{T}_5$ was not bettered even with runs of up to 5 million evaluations. Nonetheless, PAES does outperform all the other multiobjective EAs in [ZDT99] except for NSGA with the increased mutation rate, running for the same 25000 evaluations.

Experiments in which the mutation rate was changed showed PAES is sensitive to this parameter, and the best setting of it depends on problem type. Unfortunately, in this study it was not possible to experiment with the parameters of the other algorithms PAES was compared against. Therefore, we can't be sure that PAES would outperform tuned versions of SPEA and the other EAs on problems like $\mathcal{T}_4$, where it was initially worse before tuning of the mutation rate. However, on $\mathcal{T}_4$, we were able to replicate very similar levels of performance to experiments in [ZDT99] with long runs of NSGA using elitism and very large population sizes. This indicates that even in this highly multimodal function, PAES is probably competitive with most MOEAS, given the right mutation rate.

The use of Gray coding on function $\mathcal{T}_4$ also caused a great improvement in the performance of PAES. Whitley [Whi99] has shown using NFL arguments that Gray coding induces fewer local optima than Binary over the set of all functions with fewer than $2^L - 1$ optima. For a hillclimbing algorithm this may help substantially when the number of local optima is large, as in $\mathcal{T}_4$. However, when Gray was applied to the other problems in the test suite the performance of PAES was substantially reduced. In fact using Gray codes had much the same effect as increasing mutation rates. At present, the authors do not have a satisfying explanation of this phenomenon, but simply note that when tackling multimodal functions, employing Gray coding and increasing mutation rates may be fruitful adjustments to experiment with.

6 Conclusions and Future Work

The results summarised above lead us to the conclusion that tackling multiobjective problems, even those with specific features known to cause problems in proceeding to the Pareto front, or maintaining a diversity of solutions along that front, should not be the exclusive preserve of population-based evolutionary algorithms. In fact a simple hillclimbing or (1+1) selection strategy is sufficient to tackle even the difficult problems used in

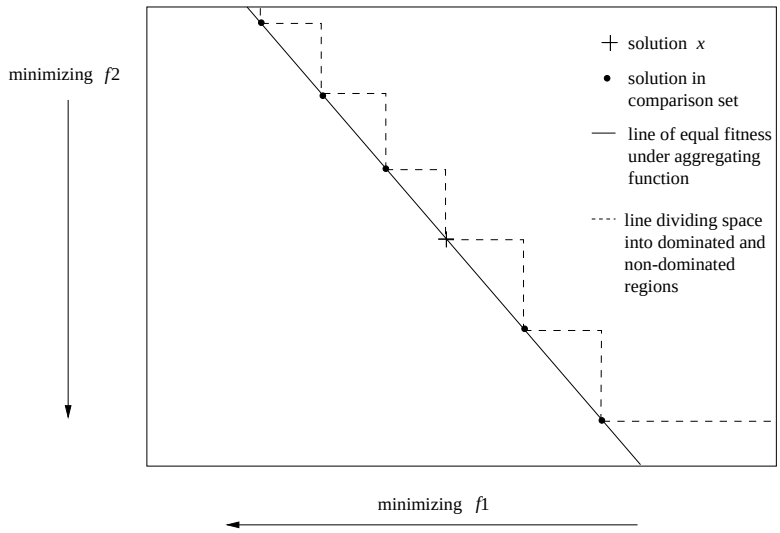

Figure 4: Illustrating how solutions of lower fitness using a linear aggregation of objectives may be of equal fitness when judged using a multiobjective approach with a comparison set of solutions

this paper. Nonetheless, our conclusion cannot be as straight forward as this. It appears from our experiments that certain of the problems in the test suite did cause PAES more difficulty than some EAs, particularly SPEA. On others PAES was substantially better. From this study it appears that PAES cannot be placed into the hierarchy put forward by Zitzler *et al.*, then. Rather, PAES seems to have difficulty with deceptive problems and perhaps multimodality. Other than this, it dominates all the algorithms in the Zitzler *et al.* study.

The empirical findings of this paper, together with previous work where PAES has been used to tackle real-world problems [KC99a] lead us to conclude that local search may often prove a powerful heuristic to be used in multiobjective spaces. We postulate that this is because assessing the quality of solutions using the notion of Pareto dominance leads to fewer local optima than when fitness is assigned using an aggregated objective function. To see why this might be true, consider Figure 4. The diagram shows a solution x in a two-objective space. A line l of equal fitness solutions under a linear objective-aggregating function is shown passing through x. In addition, a set CS of solutions which are non-dominated with respect to x and representing a comparison set against which candidate solutions are compared is also shown. The CS defines a surface $\mathcal{S}$ (shown with a dotted line) which divides the space into two regions, the dominated region and the non-dominated region. The dominated region can be considered here as comprising solutions which are less fit than x. It can be seen that there is a greater 'volume' of solutions which are non-dominated with respect to x than there is a volume of fitter or equally fit solutions under the objective aggregating function. If any of the neighbours of x fell into the region between l and S and none of the neighbours of x were fitter, then x would be a local optimum under the aggregating function but not when judged using the multiobjective comparison set. This merely illustrates that using multiobjective comparison of solutions *can* lead to situations where there are fewer local optima. However, it is simple to construct a counter-example in which there is less volume for solutions of equal or greater fitness under a multiobjective comparison set than under a linear aggregating function. To show statistically that, overall, fewer local optima are induced when multiobjective selection is employed than using objective aggregation is the subject of future work.

There are several other avenues for future research. It seems that PAES is sensitive to the mutation rate chosen and that the best setting is dependent on the type of problem. Some form of adaptive mutation rate may be examined to counteract this behaviour. The problems PAES exhibited with the deceptive function may indicate that large populations and crossover do have an important role to play in these kind of problems. This suggests that a memetic version of PAES that employs local search much of the time but which occasionally employs crossover, may be a direction worth investigating. In addition to these further developments of PAES, ways of improving the means of assessing and comparing the performance of multiobjective optimizers should be examined. To this end we are building on the techniques used in this paper to make steps towards being able to 1. Establish *how much* better one algorithm is than another, and 2. Indicate the proportion of runs that an optimizer generates an *entire* attainment surface which completely covers a given attainment surface e.g. the median attainment surface.

Acknowledgments

Joshua Knowles thanks BT Labs Plc. for sponsorship of his Ph.D. He would also like to express his gratitude to Eckart Zitzler who provided valuable information regarding details of his experimental methods and advised on how these could be best replicated.

References

[BW97] P. J. Bentley and J. P. Wakefield. Finding Acceptable Solutions in the Pareto-Optimal Range using Multiobjective Genetic Algorithms. In P. K. Chawdhry, R. Roy, and R. K. Pant, editors, *Soft Computing in Engineering Design and Manufacturing*, Part 5, pages 231–240, London, June 1997. Springer Verlag London Limited. (Presented at the 2nd On-line World Conference on Soft Computing in Design and Manufacturing (WSC2)).

[Coe99a] Carlos A. Coello. An Updated Survey of Evolutionary Multiobjective Optimization Techniques : State of the Art and Future Trends. In *1999 Congress on Evolutionary Computation*, pages 3–13, Washington, D.C., July 1999. IEEE Service Center.

[Coe99b] Carlos A. Coello Coello. A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. *Knowledge and Information Systems. An International Journal*, 1(3):269–308, August 1999.

[Deb98] Kalyanmoy Deb. Multi-Objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems. Technical Report CI-49/98, Dortmund: Department of Computer Science/LS11, University of Dortmund, Germany, 1998.

[FF93] Carlos M. Fonseca and Peter J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California, 1993. University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers.

[FF96] Carlos M. Fonseca and Peter J. Fleming. On the Performance Assessment and Comparison of Stochastic Multiobjective Optimizers. In Hans-Michael Voigt, Werner Ebeling, Ingo Rechenberg, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature—PPSN IV*, Lecture Notes in Computer Science, pages 584–593. Springer-Verlag, Berlin, Germany, September 1996.

[HJ98] Michael P. Hansen and Andrzej Jaskiewicz. Evaluating the Quality of Approximations to the Non-dominated Set. Technical Report IMM-REP-1998-7, Institue of Mathematical Modelling, Technical University of Denmark, March 1998.

[HL92] P. Hajela and C. Y. Lin. Genetic search strategies in multicriterion optimal design. *Structural Optimization*, 4:99–107, 1992.

[HNG94] Jeffrey Horn, Nicholas Nafpliotis, and David 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, June 1994. IEEE Service Center.

[KC99a] Joshua D. Knowles and David W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Submission paper for Evolutionary Computation*, 1999. (Technical Report Number RUCS/99/TR/005/A).

[KC99b] Joshua D. Knowles and David W. Corne. Assessing the Performance of the Pareto Archived Evolution Strategy. In Annie S. Wu, editor, *Proceedings of the 1999 Genetic and Evolutionary Computation Conference. Workshop Program*, pages 123–124, Orlando, Florida, July 1999.

[KC99c] Joshua D. Knowles and David W. Corne. The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Multiobjective Optimisation. In *1999 Congress on Evolutionary Computation*, pages 98–105, Washington, D.C., July 1999. IEEE Service Center.

[MB94] W. Mendenhall and R. J. Beaver. *Introduction to Probability and Statistics - 9th ed.* Duxbury Press, International Thomson Publishing, 1994.

[PM98] Geoffrey T. Parks and I. Miller. Selective Breeding in a Multiobjective Genetic Algorithm. In A. E. Eiben, M. Schoenauer, and H.-P. Schwefel, editors, *Parallel Problem Solving From Nature — PPSN V*, pages 250–259, Amsterdam, Holland, 1998. Springer-Verlag.

[Rud98] Günter Rudolph. On a Multi-Objective Evolutionary Algorithm and Its Convergence to the Pareto Set. In *Proceedings of the 5th IEEE Conference on Evolutionary Computation*, pages 511–516, Piscataway, New Jersey, 1998. IEEE Press.

[Sch84] J. David Schaffer. *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms.* PhD thesis, Vanderbilt University, 1984.

[SD94] N. Srinivas and Kalyanmoy Deb. Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation*, 2(3):221–248, fall 1994.

[SFF99] K. J. Shaw, C. M. Fonseca, and P. J. Fleming. A Simple Demonstration of a Quantitative Technique for Comparing Multiobjective Genetic Algorithm Performance. In Annie S. Wu, editor, *Proceedings of the 1999 Genetic and Evolutionary Computation Conference. Workshop Program*, pages 119–120, Orlando, Florida, July 1999.

[SNT+99] K. J. Shaw, A. L. Nortcliffe, M. Thompson, J. Love, C. M. Fonseca, and P. J. Fleming. Assessing the Performance of Multiobjective Genetic Algorithms for Optimization of a Batch Process Scheduling Problem. In *1999 Congress on Evolutionary Computation*, pages 37–45, Washington, D.C., July 1999. IEEE Service Center.

[Vel99] David A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations.* PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, May 1999.

[VL98] David A. Van Veldhuizen and Gary B. Lamont. Evolutionary Computation and Convergence to a Pareto Front. In John R. Koza, editor, *Late Breaking Papers at the Genetic Programming 1998 Conference*, pages 221–228, Stanford University, California, July 1998. Stanford University Bookstore.

[Whi91] Darrell Whitley. Fundamental Principles of Deception in Genetic Search. Technical Report CS-91-101, Department of Computer Science, Colorado State University, Fort Collins, February 1991.

[Whi99] Darrell Whitley. A Free Lunch Proof for Gray versus Binary Encodings. In Annie S. Wu, editor, *Proceedings of the 1999 Genetic and Evolutionary Computation Conference*, pages 726–733, Orlando, FLorida, July 1999.

[ZDT99] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. Technical Report 70, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland, February 1999.

[ZT98] Eckart Zitzler and Lothar Thiele. An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach. Technical Report 43, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, May 1998.

[ZT99] Eckart Zitzler and Lothar Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 1999. (accepted for final publication).