M-PAES: A Memetic Algorithm for Multiobjective Optimization

Joshua D. Knowles and David W. Corne
School of Computer Science, Cybernetics and Electronic Engineering
University of Reading, Reading RG6 6AY, UK
j.d.knowles@reading.ac.uk, d.w.corne@reading.ac.uk
http://www.rdg.ac.uk/∼ssr97jdk

Abstract-

A memetic algorithm for tackling multiobjective optimization problems is presented. The algorithm employs the proven local search strategy used in the Pareto archived evolution strategy (PAES) and combines it with the use of a population and recombination. Verification of the new algorithm is carried out by testing it on a set of multiobjective 0/1 knapsack problems. On each problem instance, comparison is made between the new memetic algorithm, the (1+1)-PAES local searcher, and the strength Pareto evolutionary algorithm (SPEA) of Zitzler and Thiele.

1 Introduction

In recent years, genetic algorithms (GAs) have been applied more and more to multiobjective problems. For a comprehensive overview, see [2]. Undoubtedly, as an extremely general metaheuristic, GAs are well qualified to tackle problems of a great variety. This asset, coupled with the possession of a population, seems to make them particularly attractive for use in multiobjective problems, where a number of solutions approximating the Pareto front are required. Indeed, changing a generic GA into a multiobjective GA (MOGA) is a relatively simple task: A selection scheme that operates with solutions possessing a vector of objective scores is the only extra requirement, although some means of maintaining diversity in the population is also often desirable. The first, pioneering work in the field was Schaffer's vector evaluated GA (VEGA) [25], which alternately optimized each of the different objectives. Later, Goldberg [9], suggested an elegant method of ranking a population of solutions, based on their mutual dominance relations. This was implemented in an algorithm, NSGA, by Srinivas and Deb [27] in 1994. Since then, Pareto methods like these have been very popular — due again to their very general applicability, and their lack of assumptions about the decision maker — and several other methods of assigning fitness based on some form of Pareto ranking have been devised, e.g. [4, 10]. More recently, elitism has been shown to improve the performance of multiobjective GAs (for example see [21]), and a very elegant method of exploiting co-evolution to perform fitness assignment in an elitist GA was put forward by Zitzler and Thiele [30, 31, 32]. The latter has been compared to some of the most popular MOGAs, on a range of problems and test functions, with very positive results. Some theoretical justification for the use of evolutionary algorithms in multiobjective optimization, in the form of convergence proofs, has also been provided [23, 24].

Almost in parallel to the development of MOGAs, there has been a growing research effort in the use of metaheuristics within the field of multiple criteria decision making (MCDM) — a branch of operations research. Algorithms based on both tabu search and simulated annealing have been put forward [3, 7, 8, 22, 26, 28]. Most of these algorithms do not have a population but store the nondominated solutions discovered during a local search process. Rather than using Pareto ranking, weighted metrics are used to aggregate the objectives into a single score to be used in the acceptance function [29]. Some researchers argue that the use of such scalarizing vectors naturally allows the preferences of the decision maker to be used to guide the direction(s) of the search towards the region(s) of interest (see for example [3]). This may be true, but the use of purely random utility functions in the absence of such preference information, as used in many algorithms, seems unsatisfactory.

Whether the algorithms devised and investigated in the MCDM field are more or less effective than MOGAs remains an open question: Very few studies that attempt to directly measure and compare the performance of MOGAs with algorithms based on tabu search [7, 8, 22] or simulated annealing [3, 26, 28], on problems of sufficient variety, have been carried out. But, despite the lack of communication between the two fields, there does seem to be some convergence of the approaches taken by them. For example, our own work on (1+1)-PAES [15, 16, 17] shows that an algorithm that employs only local search moves may be competitive with many of the most respected MOGAs, on a range of problems. PAES goes some way to bridging the gap between local search and population based methods, and is unique amongst local search algorithms in its use of a form of Pareto ranking for selection. However, our results do indicate, perhaps predictably, that while local search seems very well suited to many problems it is outperformed by some population-based methods on functions which are highly multimodal or strongly deceptive [15].

Similarly, work by Czyżak and Jaszkiewicz [3] also takes inspiration from both multiobjective camps, with a novel population-based approach to multiobjective simulated annealing. Their algorithm uses utility functions to obtain a single score from the vector of objective values, but exploits population information to adjust the direction of the utility function to direct progress in a direction approximately perpendicular to the current Pareto front. The technique inherently encourages an even spread of solutions, as well. Comparison with a single point multiobjective simulated annealer demonstrated that the use of a population was beneficial on the multiobjective knapsack problems tackled.

In this study, we take a further step towards devising methods that incorporate both local search and population-based search strategies in the multiobjective domain. With PAES, we have a local search engine that is fast and effective at approximating the Pareto front. However, its performance may be improved with the addition of a population, particularly with regard to problems that may exhibit multimodality and/or deception. Our approach is to maintain the fitness assignment methods used in PAES, a Pareto-based method, and incorporate a population and crossover to form a memetic algorithm for multiobjective optimization.

Memetic algorithms (also called genetic local search, hybrid genetic algorithms, and cultural algorithms) derive from many sources. In recent years, the methods have become more homogeneous and some great successes have been had in the optimization of a variety of classical $\mathcal{NP}$-hard optimization problems, most notably the travelling salesperson problem (TSP) [6, 20]. An overview of the technique, outlining its origins and history is given in [19]. Much of the success of memetic algorithms relies on the property of global convexity in the search space [1]. Recently, a further improvement to memetic algorithms was suggested by Jaszkiewicz [13], in which the local search space topology is changed by restricting mutations to loci in the genotype *not* common to both parents.

Proposals for multiobjective memetic algorithms have already been put forward. The first of these, devised by Ishibuchi and Murata [11], assigns fitness using a randomly selected linear utility function. Parents are then chosen using roulette wheel selection, crossover and mutation are performed, and the resulting offspring is improved by a local search, using the same utility function by which the parents were selected. The local search procedure is terminated when k neighbours of the current solution have been examined with no improvement. An elitist strategy is also incorporated in the procedure. The algorithm was tested on some Flowshop Scheduling tasks but comparison was limited to fairly outdated multiobjective algorithms including VEGA [25]. More recently, two proposals for novel algorithms were put forward in a paper by Jaszkiewicz [12]. The first was based on a hybrid with simulated annealing, and the second is similar to Ishibuchi and Murata's but introduces a form of mating restriction, so that only the N best solutions measured using a random utility function are allowed to mate. The two proposed algorithms were tested and compared with Ishibuchi and Murata's on a set of multiobjective travelling salesperson problems. On these problems, global convexity may be exploited [1], and so restricting mating is advantageous. The results of the comparison reflect this fact.

In this paper, a direct comparison is made between the performance of the new memetic algorithm proposed, and two existing approaches: The local search method, (1+1)-PAES; and the strength Pareto evolutionary algorithm (SPEA) [30, 31, 32]. To make the comparison, each algorithm is run 30 times on a suite of 9 multiobjective 0/1 knapsack problems [32]. The remainder of the paper is organised as follows: In Section 2 the M-PAES algorithm is described. The experimental method used for verifying the algorithm, and comparing its performance are discussed in Section 3. The parameter choices made for each algorithm, including two versions of SPEA, and a benchmark single objective EA, are presented in this section. Results are presented in Section 4. Finally, some concluding remarks are made in the last section.

2 M-PAES

The memetic-PAES algorithm (M-PAES) is shown in pseudocode in Figure 1. It is based on the local search multiobjective algorithm, (1+1)-PAES [16], but uses a population of solutions and periodically employs crossover to recombine the distinct local optima found using the PAES procedure. The archiving of solutions in M-PAES is a little more complicated than in (1+1)-PAES. Recall that at the heart of PAES is a procedure for maintaining a finite sized archive of nondominated solutions. The solutions in the archive are representative of the best nondominated solutions found by the algorithm as it searches the space. The solutions in the archive serve a dual purpose in (1+1)-PAES: as a memory of the solutions found during the run for presentation at the end; and as a comparison set to aid in estimating the dominance rank of new candidate solutions. In order that these same jobs are performed in M-PAES, two archives are required. This is because each local search phase needs to be partially independent of the global search being performed by the algorithm as a whole. Thus we have a global archive G that maintains a finite set of nondominated solutions found, and a local archive H that is used as the comparison set in each of the local search phases. At the beginning of a local search phase, H is cleared and filled with solutions from G which do not dominate the candidate solution c. The archive H is then used as in (1+1)-PAES to improve c, i.e. H is maintained and used as a comparison set, while G is continually updated but plays no part in the estimation of the quality of new solutions.

The PAES local search procedure used by M-PAES to improve solutions in P is almost the same as the basic (1+1)-PAES algorithm. However, it differs in the way that termination of the procedure is determined. Termination may be invoked when either of two conditions are fulfilled: (1) If the maximum number of local search moves l_opt is exceeded. (2) If the maximum number of local search fails l_fails is exceeded. To achieve (2), the variable $\#fails$, initially zero, is incremented every time the mutant is dominated by the current solution. It is reset to zero every time a move occurs i.e. when the mutant is accepted as the new current solution. Hence, $\#fails$ effectively counts the number of potentially detrimental moves between improving moves. If this number exceeds the threshold l_fails, the local search is stopped. The local search procedure $PAES(c, G, H)$ is shown in Figure 2.

In the recombination phase, parents are randomly selected

```
Generate initial population P of n random solutions and evaluate
Place each nondominated member of P in a global archive G
Do
  For(each candidate solution c ∈ P)                         %% local search phase
     Set the current local archive H = ∅
     Fill H with any solutions from G that do not dominate c
     Copy the solution c from P into H
     Perform local search using procedure PAES(c, G, H)
     Replace improved solution c back into population P
  End For
  Set intermediate population empty:  n_i = 0, P' = ∅
  Do                                                         %% recombination phase
    Set # recombination trials r = 0
    Do
       Randomly choose two parents from P ∪ G and recombine to form offspring c
       Compare c with the solutions in G
       Update G with c as necessary
       r++
    While (((c is dominated by G)∨(c is in more crowded grid location than both parents))∧
     (r < recomb_trials_max))
    If (c is dominated by G)
      Discard c and use binary tournament to select a new solution c from G
    Endif
    Place offspring c into intermediate population P', n_i++
  While (n_i < n)
  Update population:  P ← P'
While (stopping criterion is not satisfied)
Return global archive G of unique nondominated solutions
```

Figure 1: The M-PAES Algorithm.

from the union of the post-local search population, and the global archive. The resultant child is accepted only if it is nondominated with respect to the entire global archive, and it resides in a less crowded region (grid location [17]) than at least one of its parents. If it dominates any member of G it is naturally accepted too. However, solutions that are dominated by member(s) of G, or that reside in crowded regions are rejected. In this case two new parents are selected again and recombination is applied once more. The procedure is repeated until either a child is accepted or a threshold number of recombinations $recomb\_trials\_max$ is exceeded. In the latter case, a solution is selected by binary tournament, from the global archive, to join the intermediate population P'. The recombination strategy is, as a whole, extremely elitist, following the general form of the (1+1)-PAES algorithm employed in the local search phase. In the development of M-PAES, early versions did not have the facility of repeatedly rejecting children of recombination. However, we found that this weakened the effectiveness of the elitism inherent in (1+1)-PAES and so the recombination phase was made more stringent in later versions.

3 Experimental Method

The M-PAES algorithm is tested on a suite of multiobjective 0/1 knapsack problems. The problems are taken from a recent paper [32] by Zitzler and Thiele (ZT), in which the general ability of their strength Pareto evolutionary algorithm (SPEA) was demonstrated. In [32], the performance of SPEA on these problems is compared with eight other evolutionary algorithms (EAs). Four of the algorithms, each a well-known multiobjective EA, as well as two versions of SPEA, were run 30 times with different random seeds on each of the problems, for 500 generations using the same population sizes[1]. The nondominated sets generated from each of the runs were used to make a statistical comparison of the algorithms tested.

The findings of the ZT study show that SPEA is superior to each of the other MOEAs on all of the knapsack problems. However, also included in the set of eight algorithms tested in [32], are two single-objective EAs that use weighted-sum aggregation of the objectives. The relative performance of SPEA and these algorithms is not clear-cut. Hence, in this study we select as benchmarks, the data sets from the SPEA runs and those of the more powerful of the two single objective algorithms, SO-5. The other algorithms are not considered.

As addition comparators, we generated our own data sets for the (1+1)-PAES algorithm, and an enhanced setup of SPEA, on the knapsack problems. The parameter settings for each of these five algorithms are described in Section 3.2.

3.1 Multiobjective 0/1 Knapsack Problems

The knapsack problems have been described fully in [32]. There are nine problems altogether, of differing combinations of size (number of items), and number of objectives (knapsacks). At the time of writing, the problems are available from an Internet web-site[2]. Provided at the same site are the raw results obtained in the ZT study, for all the algorithms

[1] In the case of SPEA, an internal and an external population exist. The sizes of these were chosen to provide a fair comparison with the other MOEAs in the study.

[2] http:/www.tik.ee.ethz.ch/∼zitzler

```
While((#fails < l_fails)∧(#moves < l_opt))
    Mutate c to produce m and evaluate m
    If (c dominates m) discard m, #fails++
    Else if (m dominates c)
        Replace c with m, add m to H, #fails = 0
    Else if (m is dominated by any member of H) discard m
    Else apply test(c,m,H) to determine which becomes the new
      current solution and whether to add m to the archive
    Archive m in G as necessary
    #moves++
End while
```

Figure 2: The $PAES(c, G, H)$ procedure.

```
If the archive is not full
    Add m to the archive
    If (m is in a less crowded region of the archive than c)
        Accept m as the new current solution
    Else maintain c as the current solution
Else
    If (m is in a less crowded region of the archive than x for
      some member x on the archive)
        Add m to the archive, and remove a member of the archive from
          the most crowded region
        If (m is in a less crowded region of the archive than c)
            Accept m as the new current solution
        Else maintain c as the current solution
    Else
        If (m is in a less crowded region of the archive than c)
            Accept m as the new current solution
        Else maintain c as the current solution
```

Figure 3: Pseudocode for $test(c, m, archive)$.

tested. We make use of some of this data as baseline results for comparison purposes. In particular, the data sets from the two algorithms, SO-5 and SPEA are used here.

We employ the same chromosome encoding and constraint handling techniques as described in [32], and no additional heuristics for use with the knapsack problems are employed. This allows for a direct comparison between our results and those published by Zitzler and Thiele.

3.2 Parameter Choices

Our philosophy in comparing the performance of M-PAES with the other algorithms considered, (1+1)-PAES and SPEA, is that each algorithm be run with settings, found through some experimentation, to provide near-best performance for that algorithm. The aim is to demonstrate the utility of the three algorithms on the knapsack problems considered, and we do not claim that the parameter choices can be rigorously justified, although some explanation is given. Indeed, the main objective is to show that the proposed memetic algorithm can produce competitive results on a well-known $\mathcal{NP}$-hard problem [7].

Details of the parameter choices made for each of the algorithms tested are given below. The data from the setup of SPEA used in [32] is referred to as SPEA(ZT). Our own setup of SPEA is labelled SPEA(KC).

SO-5

The SO-5 data comes from one of two single-objective EAs used in [32]. Unlike the other algorithms considered, these single-objective EAs were run 100 times per test problem, each run optimizing toward a different randomly chosen linear combination of the objectives. The resultant non-dominated solutions among all those generated in the runs form the tradeoff front achieved by the algorithm. The two algorithms both employed equal population sizes to their multiobjective rivals, and differed only in that one (SO-1) was run for 100 generations, and the other (SO-5) was run for 500 generations in every single of the 100 runs used to form the non-dominated front. Thus, in the case of SO-5, one hundred times as many function evaluations as in the other MOEAs in the ZT study were performed in order to generate the (single) set of nondominated solutions. Zitzler and Thiele did not perform the whole process thirty times to give thirty different data sets, but instead just used the same set repeatedly in the statistical analysis carried out. We follow this approach, using ZT's data sets. Hence, where statistical information is given in relation to the SO-5 algorithm, it should be noted that, in fact only one data set for this algorithm is being used, in contrast to all the other algorithms in this study for which 30 runs were performed.

As with the other algorithms in the ZT study, one-point crossover was used. The mutation probability and crossover rate were fixed at 0.01 and 0.8 respectively, as for SPEA(ZT).

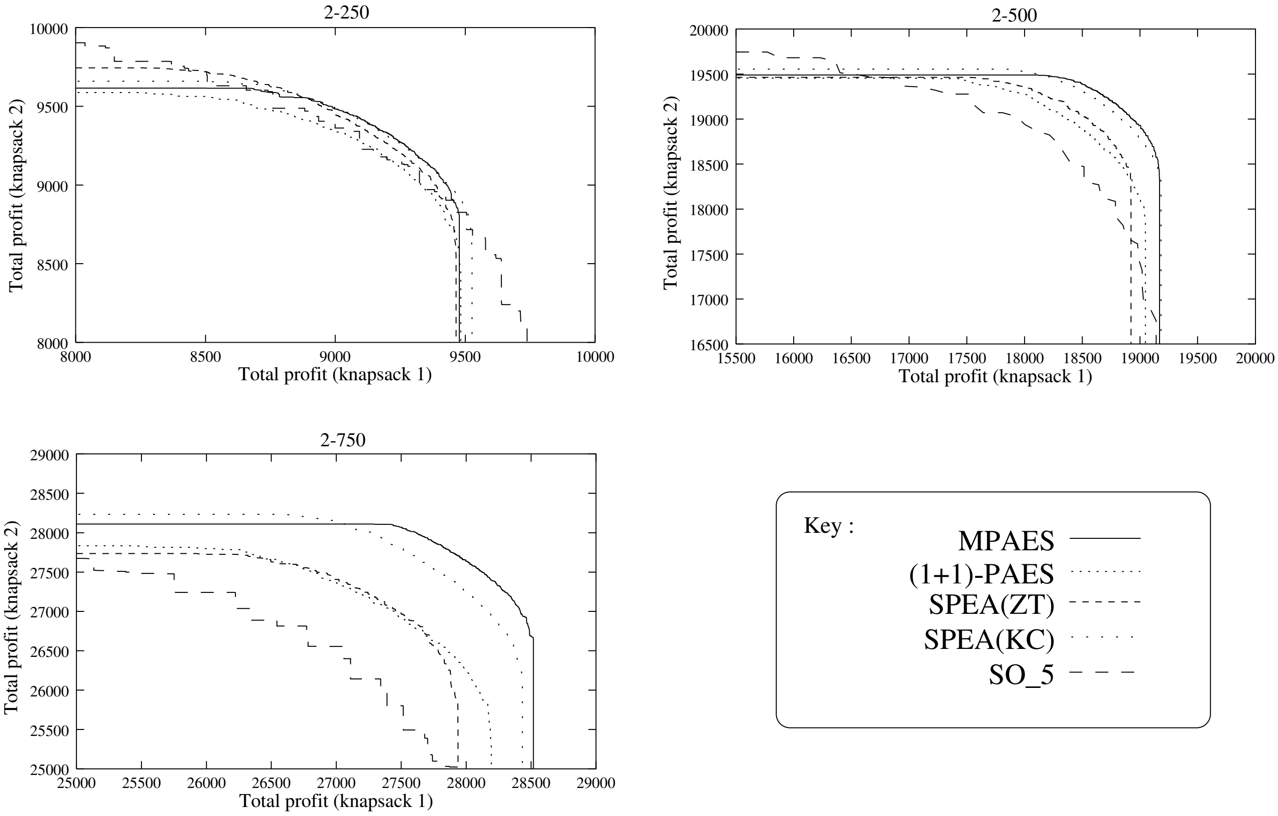

Figure 4: Median surfaces calculated from 30 runs of each algorithm, on the two-objective knapsack problems.

SPEA(ZT)

The setup of SPEA is described fully in [32]. The study was designed to show that SPEA could clearly outperform the other MOEAs tested, even with very conservative choices of parameters. Thus the authors kept the external population quite small - 1/5 of the population size of the other MOEAs in the study.

It is important to note that the data sets for SPEA(ZT) record the off-line performance of the algorithm. That is, all of the nondominated solutions returned in a run were recorded. With the exception of SO-5 the other algorithms in this study are judged using the on-line performance. That is, only solutions stored in the external population (or archive) at the end of the run are recorded.

SPEA(KC)

In [32], the authors chose to run the algorithms for a fixed number of generations and increase population size with the size and number of objectives of the knapsack problem being tackled. To make direct comparison possible, we choose to use the same number of function evaluations as ZT, but do not deem it necessary to employ equal population sizes. In fact, the total number of evaluations max_evals used by each of the algorithms in this study is the same for a given knapsack problem. Figure 6 lists the value of max_evals for each knapsack problem.

SPEA requires two population sizes, N and N' to be set. ZT selected to use $N = 4/5$ and $N' = 1/4$ of the size of population used by the other GAs in their study. Experiments performed by us show that the performance of SPEA on these knapsack problems is improved significantly when population sizes of $N = 1/5$ and $N' = 4/5$ are used, for the same total number of function evaluations.

Our experiments also indicate that changing the crossover type from one-point to uniform improves the performance of SPEA on the knapsack problems. Thus, SPEA(KC) employs uniform crossover. A fixed per-bit mutation rate $p_m = 0.01$ is used, as in [32]. No experiments in which p_m was varied were undertaken by us. Since no other parameters need to be set for SPEA, we believe that SPEA(KC) is close to the best setup of SPEA possible for the problems tackled.

(1+1)-PAES

With (1+1)-PAES, very few parameters must be set. The archive size was set equal to the external population size N' of SPEA, so that the same number of solutions is returned by each algorithm. Similarly, the number of evaluations is set in accordance with the total number performed by SPEA.

The mutation rate p_m was set to $4/L$ (where L is the number of bits in the chromosome) for all problems. This setting follows our principle of using the best setting for the particular algorithm.

Knapsack problem	Algorithm			
	(1+1)-PAES	SPEA(ZT)	SPEA(KC)	S0-5
2-250	[87.2, 0]	[61.6, 24.5]	[28.1, 21.8]	[64.3, 27.5]
2-500	[100, 0]	[100, 0]	[80.9, 7.2]	[92.8, 3.6]
2-750	[100, 0]	[100, 0]	[95.5, 0]	[100, 0]
3-250	[100, 0]	[69.1, 18.7]	[53.6, 26.6]	[44.4, 51.8]
3-500	[100, 0]	[93.7, 0]	[67.9, 20.8]	[74.5, 21.7]
3-750	[100, 0]	[100, 0]	[84.2, 3.9]	[94.2, 3.6]
4-250	[100, 0]	[46.1, 31.6]	[32.9, 43.4]	[10.0, 85.9]
4-500	[100, 0]	[92.5, 3.7]	[63.3, 21.7]	[35.5, 56.6]
4-750	[100, 0]	[100, 0]	[82.9, 7.4]	[71.7, 25.4]

Figure 5: The results of testing M-PAES against the algorithms shown, using our statistical techniques [17]. The knapsack problems have 2, 3 or 4 objectives and 250, 500 or 750 items, as indicated.

The number of bisections of the objective space l used in the adaptive grid algorithm for maintaining diversity [17] was set according to the number of objectives of the problem. The values $l = 5$, $l = 4$, $l = 3$ were used for the 2, 3, and 4 objective multiple knapsack problems, respectively.

M-PAES

The total number of evaluations, number of bisections of objective space, l, and per-bit mutation rate used in M-PAES are as for (1+1)-PAES. The population size N, was set equal to the internal population of SPEA(KC). The two archives were sized equally, to match the external population of our setup of SPEA. Thus, the same number of solutions are returned by SPEA(KC), M-PAES and (1+1)-PAES.

In M-PAES, three more parameters must be set. These are the number of crossover trials, the maximum number of local moves l_opt, and the maximum number of consecutive failing local moves l_fails. Choices that give good general performance were found to be $l\_opt = 50$, $l\_fails = 20$, and $cr\_trials = 25$. However, it was found that increasing the number of crossover trials for the 3 and 4-objective problems increased performance further. A list of the best parameter selections found is given in Table 6. The results presented in Figure 4 and Table 5 are for these settings.

Knapsack problem	Parameter			
	l_fails	l_opt	cr_trials	max_evals
2-250	20	100	25	75000
2-500	20	100	25	100000
2-750	20	100	25	125000
3-250	20	50	100	100000
3-500	5	20	125	125000
3-750	20	50	150	150000
4-250	20	50	125	125000
4-500	20	50	150	150000
4-750	5	20	150	175000

Figure 6: Parameter settings used in the M-PAES algorithm for the various multiple objective knapsack problems. The same total number of function evaluations max_evals, shown for each problem, was used in M-PAES, SPEA(KC) and (1+1)-PAES.

4 Results

4.1 Performance Metrics

As in previous research, we measure the performance of the algorithms tested using a statistical comparative assessment technique adapted from [5]. We refer the reader to [14, 16] for a complete description of our implementation of the technique and a discussion of its advantages and disadvantages.

Here, the reader need only understand that the technique allows us to present results in two ways. First, as a pair of numbers $[a, b]$ indicating respectively the percentage of the space where algorithm A outperforms algorithm B, and the percentage of the space where algorithm B outperforms algorithm A. The percentages returned are the product of a Mann-Whitney U test [18] at a given confidence level (we choose 95%). Second, for two-objective problems we can plot surfaces representing the median, best, or worst surface that the algorithm returns. Here, we plot just the median surface of the five algorithms tested, for the two-objective problems only.

4.2 Analysis

The median surfaces plotted for the two-objective problems are shown in Figure 4. A number of observations can be made from these plots. First, our setup of (1+1)-PAES gives very similar levels of performance to the setup of SPEA used by Zitzler and Thiele on the three problems. This observation is in keeping with previous research where (1+1)-PAES was compared with SPEA [15]. Second, in all cases SPEA(KC) outperforms SPEA(ZT). Third, M-PAES is very competitive with SPEA(KC) and clearly outperforms both (1+1)-PAES and SPEA(ZT). In the largest of the three problems, M-PAES generates a median surface that is not dominated in any part by the median surface of any of the other algorithms. On the smaller problems, M-PAES fails to generate solutions as far towards the extremes of the objective space as either SO-5 or SPEA(KC), but has generated a median surface that dominates these algorithms in the region where the two objectives trade off most rapidly with each other.

The statistical results for all the problems are summarised in Table 5. Comparisons between M-PAES and each of the algorithms are presented only. Thus, the statistic [100, 0] in the upper left entry in the table means that M-PAES gives a better distribution of surfaces over 100% of the combined nondominated front than (1+1)-PAES on the 250 item, 2 knapsack problem. Again, several observations from these results can be made. First, the first three rows of the table verify that M-PAES performs well on the two-objective problems, as suggested by the plots in Figure 4. Its relative performance increases as the number of items increases. This is true, not only on the 2-objective problem but on all the problems presented. However, as the number of objectives increases, the performance of SPEA(KC) and SO-5 increase relative to M-PAES, so that M-PAES is outperformed by SO-5 on the two smaller four-objective knapsack problems. SPEA(KC) only outperforms M-PAES on one problem, the smallest of the 4-objective knapsack problems, and only by a small margin, according to our statistical analysis.

5 Conclusion

A memetic algorithm for multiobjective optimization, M-PAES, was described. It uses the local search method of (1+1)-PAES, and combines it with the use of a population and crossover. The utility of M-PAES was verified on a set of nine multiobjective 0/1 knapsack problems. On these problems, the performance of M-PAES and a selection of other algorithms was compared. The results indicate that M-PAES performs better than (1+1)-PAES on all problems. Compared with the strength Pareto evolutionary algorithm (SPEA), the performance of M-PAES is similar for a setup of each algorithm empirically derived to give near-best performance. Indeed, M-PAES appears to be superior on some problem instances, although comparison between these very different algorithms is difficult. Nonetheless, the findings indicate that further investigation into the use of memetic algorithms in this problem domain is warranted.

In future work, comparison between M-PAES, local-search methods from the MCDM field, and other recently proposed memetic algorithms should be performed. We would also like to investigate how mating restrictions and the changing of the local search topology, as suggested in [12] and [13] respectively, could be used to improve the performance of M-PAES still further.

Acknowledgments

The first author would like to express his gratitude to BT Labs Plc. for the continuing sponsorship of his Ph.D.

Bibliography

[1] P. C. Borges and M. Pilegaard Hansen. A basis for future successes in multiobjective combinatorial optmization. Technical Report IMM-REP-1998-8, Institute of Mathematical Modelling, Technical University of Denmark, 2800 Lyngby, Denmark, 1998.

[2] C. 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.

[3] P. Czyżak and A. Jaszkiewicz. Pareto simulated annealing - a metaheuristic technique for multiple-objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, 7(1):34–47, January 1998.

[4] C. M. Fonseca and P. J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In S. 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.

[5] C. M. Fonseca and P. J. Fleming. Nonlinear System Identification with Multiobjective Genetic Algorithms. In *Proceedings of the 13th World Congress of the International Federation of Automatic Control*, pages 187–192, San Francisco, California, 1996. Pergamon Press.

[6] B. Freisleben and P. Merz. A Genetic Local Search Algorithm for Solving Symmetric and Asymmetric Traveling Salesman Problems. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, pages 616–621. IEEE Press, 1996.

[7] X. Gandibleux and A. Fréville. Tabu Search Based Procedure for solving the 0/1 Multiobjective knapsack Problem : the two objective case. *Journal of Heuristics*, Accepted 1998. (to appear).

[8] X. Gandibleux, N. Mezdaoui, and A. Fréville. A tabu Search Procedure to Solve Multiobjective Combinatorial Optimization Problems. In R. Caballero and R. Steuer, editors, *Proceedings volume of MOPGP'96*, pages 291–300, Berlin, 1996. Springer-Verlag.

[9] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1989.

[10] J. Horn and N. Nafpliotis. Multiobjective Optimization using the Niched Pareto Genetic Algorithm. Technical Report IlliGAl Report 93005, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, 1993.

[11] H. Ishibuchi and T. Murata. Multi-Objective Genetic Local Search Algorithm. In T. Fukuda and T. Furuhashi, editors, *Proceedings of the 1996 International Conference on Evolutionary Computation*, pages 119–124, Nagoya, Japan, 1996. IEEE.

[12] A. Jaszkiewicz. Genetic local search for multiple objective combinatorial optimization. Technical Report RA-014/98, Institute of Computing Science, Poznan University of Technology, December 1998.

[13] A. Jaszkiewicz. Improving performance of genetic local search by changing local search space topology. *Foundations of Computing and Decision Sciences*, 24:77–84, 1999.

[14] J. D. Knowles and D. W. Corne. Assessing the Performance of the Pareto Archived Evolution Strategy. In A. S. Wu, editor, *Proceedings of the 1999 Genetic and Evolutionary Computation Conference. Workshop Program*, pages 123–124, San Francisco, CA, July 1999. Morgan Kaufmann.

[15] J. D. Knowles and D. W. Corne. Local Search, Multiobjective Optimization and the Pareto Archived Evolution Strategy. In B. McKay, editor, *Proceedings of The Third Australia-Japan Joint Workshop on Intelligent and Evolutionary Systems*, pages 209–216, Ashikaga, Japan, November 1999. Ashikaga Institute of Technology.

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

[17] J. D. Knowles and D. W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2):149–172, 2000.

[18] W. Mendenhall and R. J. Beaver. *Introduction to Probability and Statistics - 9th edition*. Duxbury Press, International Thomson Publishing, Pacific Grove, CA, 1994.

[19] P. Moscato. Memetic algorithms: A short introduction. In D. Corne, F. Glover, and M. Dorigo, editors, *New Ideas in Optimization*, pages 219–234. McGraw-Hill, 1999.

[20] P. Moscato and F. Tinetti. Blending heuristics with a population-based approach: A memetic algorithm for the traveling salesman problem. Report 92-12, Universidad Nacional de La Plata, C.C. 75, 1900 La Plata, Argentina, 1992.

[21] G. 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.

[22] M. Pilegaard Hansen. Tabu Search in Multiobjective Optimisation : MOTS. In T. Stewart and R. van den Honert, editors, *Proceedings of 13th International Conference on Multiple Criteria Decision Making*, Berlin, 1996. Springer-Verlag.

[23] G. Rudolph. Evolutionary Search for Minimal Elements in Partially Ordered Finite Sets. In V. Porto, N. Saravanan, D. Waagen, and A. Eiben, editors, *Evolutionary Programming VII, Proceedings of the 7th Annual Conference on Evolutionary Programming*, pages 345–353, Berlin, 1998. Springer-Verlag.

[24] G. 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.

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

[26] P. Serafini. Simulated Annealing for Multi Objective Optimization Problems. *Multiple Criteria Decision Making - Expand and Enrich the Domains of Thinking and Application*, pages 283–292, 1994.

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

[28] E. L. Ulungu, J. Teghem, P. H. Fortemps, and D. Tuyttens. MOSA Method: A Tool for Solving Multiobjective Combinatorial Optimization Problems. *Journal of Multi-Criteria Decision Analysis*, 8(4):221–236, 1999.

[29] A. P. Wierzbicki. The use of reference objectives in multiobjective optimization. In G. Fandel and T. Gal, editors, *MCDM Theory and Application, Proceedings Hagen/Konigswinter 1979*, Lecture Notes in Economics and Mathematical Systems 177, pages 468–486, Berlin, 1980. Springer-Verlag.

[30] E. Zitzler, K. Deb, and L. 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.

[31] E. Zitzler and L. 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.

[32] E. Zitzler and L. Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, November 1999.