Comparison of Multiobjective Memetic Algorithms on 0/1 Knapsack Problems

Hisao Ishibuchi

Dept. of Industrial Engineering
Osaka Prefecture University
1-1 Gakuen-cho, Sakai, Osaka 599-8531, JAPAN
E-mail: hisaoi@ie.osakafu-u.ac.jp
Phone: +81-72-254-9350

Shiori Kaige

Dept. of Industrial Engineering
Osaka Prefecture University
1-1 Gakuen-cho, Sakai, Osaka 599-8531, JAPAN
E-mail: shiori@ie.osakafu-u.ac.jp
Phone: +81-72-254-9351

Abstract

This paper compares two well-known multiobjective memetic algorithms through computational experiments on 0/1 knapsack problems. The two algorithms are MOGLS (multiple objective genetic local search) of Jaszkiewicz and M-PAES (memetic Pareto archived evolution strategy) of Knowles & Corne. It is shown that the MOGLS with a sophisticated repair algorithm based on the current weight vector in the scalar fitness function has much higher search ability than the M-PAES with a simple repair algorithm. When they use the same simple repair algorithm, the M-PAES performs better overall. It is also shown that the diversity of non-dominated solutions obtained by the M-PAES is small in comparison with the MOGLS. For improving the performance of the M-PAES, we examine the use of the scalar fitness function with a random weight vector in the selection procedure of parent solutions.

1. INTRODUCTION

Memetic algorithms are one of the most successful metaheuristics in combinatorial optimization. Recently memetic algorithms have been applied to multiobjective optimization problems for efficiently finding their Pareto-optimal or near Pareto-optimal solutions (Ishibuchi & Murata (1998), Jaszkiewicz (2002a), Knowles & Corne (2000b)). It was demonstrated in some comparative studies (Jaszkiewicz (2001), Jaszkiewicz (2002b), Knowles & Corne (2000c)) that MOGLS (multiple objective genetic local search) of Jaszkiewicz (2002a) and M-PAES (memetic Pareto archived evolution strategy) of Knowles & Corne (2000b) have high search ability to efficiently find near Pareto-optimal solutions of multiobjective 0/1 knapsack problems. For example, it was clearly shown in Jaszkiewicz (2002b) that the MOGLS outperformed SPEA (strength Pareto evolutionary algorithm), which is a well-known high-performance evolutionary multiobjective optimization algorithm (Zitzler et al. (2000), Zitzler & Thiele (1999)). An interesting observation is that the final conclusions of those comparative studies are somewhat different. While Knowles & Corne (2000c) concluded that their M-PAES was superior to the MOGLS, Jaszkiewicz (2001, 2002b) concluded that his MOGLS outperformed the M-PAES.

In this paper, we try to find why such different conclusions were derived with respect to the relative performance of the two multiobjective memetic algorithms. For this purpose, we compare the MOGLS and the M-PAES with each other in the same manner as Jaszkiewicz (2001) and Knowles & Corne (2000c) through computational experiments on the nine multiobjective 0/1 knapsack problems originally used in the computational experiments of Zitzler & Thiele (1999). Then we examine the effect of two important factors on the relative performance of the two algorithms. They are the implementation of greedy repair and the specification of the population size. The importance of these factors was pointed out by Jaszkiewicz (2001, 2002b). Our experimental results partially support his claim.

As in the above-mentioned comparative studies, our experimental results show that the M-PAES cannot find a variety of non-dominated solutions over a wide range of each objective. On the other hand, the MOGLS can find non-dominated solutions over a much wider range of each objective than the M-PAES. Based on these observations, we try to improve the performance of the M-PAES by incorporating the scalar fitness function with a random weight vector of the MOGLS into the selection procedure of the M-PAES. More specifically, we use tournament selection based on the scalar fitness function with a

random weight vector. Whenever a pair of parents is to be chosen, weight values are randomly updated. It was shown in Ishibuchi et al. (2003) that the performance of a simple MOGLS of Ishibuchi & Murata (1998) was improved by increasing the selection pressure (i.e., using the tournament selection instead of the roulette wheel selection and increasing the tournament size).

This paper is organized as follows. In Section 2, we briefly explain our computational experiments, which are designed in the same manner as the existing comparative studies (Jaszkiewicz (2001), Knowles & Corne (2000c)). In Section 3, we point out the effect of the implementation of greedy repair and the specification of the population size on the relative performance of the MOGLS and the M-PAES. In Section 4, the use of the scalar fitness function with a random weight vector is examined. Section 5 concludes the paper.

2. COMPUTATIONAL EXPERIMENTS

As test problems, we use the nine multiobjective 0/1 knapsack problems of Zitzler & Thiele (1999). Each test problem has two, three or four objectives and 250, 500 or 750 items. We refer to each test problem as a *k*-*n* problem where *k* is the number of knapsacks (i.e., the number of objectives) and *n* is the number of items. The nine test problems are denoted as 2-250, 2-500, 2-750, 3-250, 3-500, 3-750, 4-250, 4-500 and 4-750. Those test problems are written in a generic form as follows:

$$\text{Maximize } \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}),\ f_2(\mathbf{x}),\ ...,\ f_k(\mathbf{x})), \quad (1)$$

$$\text{subject to } \sum_{j=1}^{n} w_{ij} x_j \le c_i, \quad i = 1, 2, ..., k, \quad (2)$$

where

$$f_i(\mathbf{x}) = \sum_{j=1}^{n} p_{ij} x_j, \quad i = 1, 2, ..., k. \quad (3)$$

In this formulation, $\mathbf{x}$ is a binary vector of the length n (i.e., $(x_1, x_2, ..., x_n) \in \{0, 1\}^n$), p_{ij} is the profit of item j according to knapsack i, w_{ij} is the weight of item j according to knapsack i, and c_i is the capacity of knapsack i. For details of the test problems, see Zitzler & Thiele (1999). The same test problems were used in the comparative studies of Jaszkiewicz (2001, 2002b) and Knowles & Corne (2000c).

The MOGLS of Jaszkiewicz (2002a) uses the scalar fitness function with a random weight vector:

$$f(\mathbf{x}, \Lambda) = \sum_{i=1}^{k} \lambda_i f_i(\mathbf{x}), \quad (4)$$

where

$$\forall i \ \ \lambda_i \ge 0 \ \text{ and } \ \sum_{i=1}^{k} \lambda_i = 1. \quad (5)$$

When a pair of parents is chosen, first the weight vector is randomly specified. Next the best *K* solutions are selected from the current population (*CS*) with respect to the scalar fitness function with the current weight vector. Then two parents are randomly chosen from those *K* solutions. In this manner, mating restriction is implemented in the MOGLS where a pair of similar parents in the objective space is selected for generating an offspring. A local search procedure is applied to the generated offspring using the scalar fitness function with the current weight vector. In the original proposal of the MOGLS by Jaszkiewicz (2002a), local search is iterated until a locally optimal solution is found. On the other hand, no local improvement procedure except for a greedy repair algorithm is used in his recent comparative study (2002b). In this paper, we use two parameters for terminating local search for each solution as in Jaszkiewicz (2001) and Knowles & Corne (2000c). One is the maximum number of local search moves (i.e., *l_opt*) and the other is the maximum number of consecutive fails of local search moves (i.e., *l_fails*). We also use these two parameters in the M-PAES. In both algorithms, a neighboring solution is generated by applying a bit-flip mutation with a probability of 4/*n* to each bit of the current solution as in Jaszkiewicz (2001) and Knowles & Corne (2000c).

The M-PAES was proposed in Knowles & Corne (2000b) by introducing a population and a recombination operation to a multiobjective local search algorithm: (1+1)-PAES (Pareto archived evolution strategy) of Knowles & Corne (2000a). While each solution is evaluated using the scalar fitness function in the MOGLS, Pareto ranking is used in the M-PAES. The concept of crowding is also utilized for evaluating each solution in the M-PAES. Two secondary populations (i.e., a local archive *H* and a global archive *G*) are stored separately from the main population *P*. The local archive *H* is used for evaluating each solution in local search while a pair of parent solutions is randomly chosen from $P \bigcup G$ for generating an offspring.

Some parameter values in our computational experiments are summarized in Table 1. In this table, *max_evals* is the total number of evaluated solutions, which is used as the stopping condition of each algorithm in this paper. Our parameter specifications in the M-PAES and the MOGLS are almost the same as those in Knowles & Corne (2000b, 2000c) and Jaszkiewicz (2001), respectively.

As the performance measure, we use the coverage measure $C(\cdot, \cdot)$ of Zitzler & Thiele (1999) for comparing the two multiobjective memetic algorithms. This measure mainly evaluates the relative convergence speed to the Pareto front of the two algorithms. So we also visually examine the diversity of obtained solutions. For various performance measures and their characteristic features in multiobjective optimization, see Knowles & Corne (2002).

Table 1: Parameter values in our computational experiments.

Problems	Initial population size		K	$\|CS\|$	*l_fails*	*l_opt*	*max_evals*
	M-PAES	MOGLS	MOGLS	MOGLS	M-PAES & MOGLS		
2-250	30	150	20	3,000	20	100	75,000
2-500	40	200	20	4,000	20	100	100,000
2-750	50	250	20	5,000	5	20	125,000
3-250	40	200	20	4,000	20	50	100,000
3-500	50	250	20	5,000	20	50	125,000
3-750	60	300	20	6,000	5	20	150,000
4-250	50	250	20	5,000	20	50	125,000
4-500	60	300	20	6,000	20	50	150,000
4-750	70	350	20	7,000	5	20	175,000

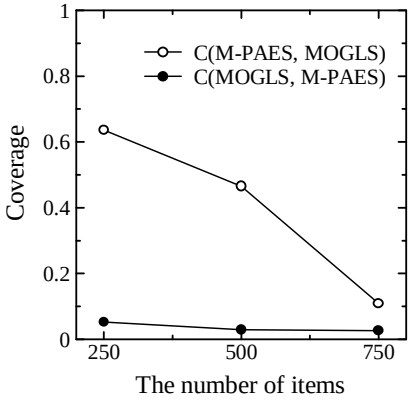

Figure 1: Results on 2-objective problems.

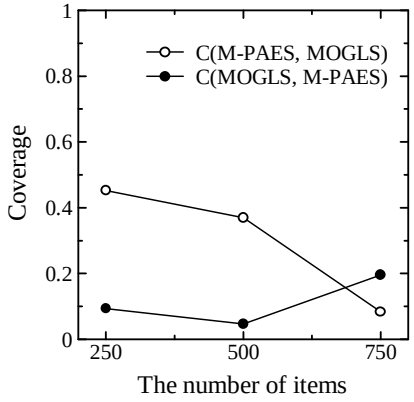

Figure 2: Results on 3-objective problems.

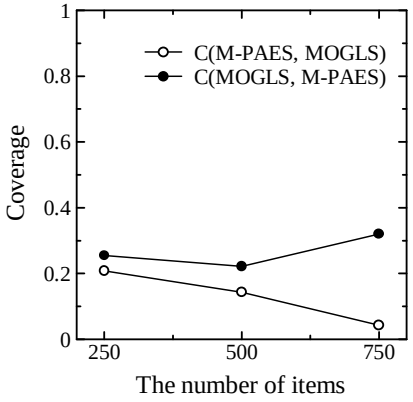

Figure 3: Results on 4-objective problems.

3. EXPERIMENTAL RESULTS

3.1 EFFECT OF GREEDY REPAIR

Zitzler & Thiele (1999) used a simple greedy repair algorithm where the items were removed in the increasing order of the maximum profit/weight ratio over all knapsacks. The same greedy repair algorithm was used in Knowles & Corne (2000c). We first used this greedy repair algorithm in the MOGLS and the M-PAES in our computational experiments. Experimental results are summarized in Figs. 1-3 where the average value of the coverage measure over 30 runs is calculated for each test problem. A solution set obtained by a single run of each algorithm is depicted in Fig. 4 for the 2-500 problem. As shown in those figures, the M-PAES outperformed the MOGLS for the two-objective and three-objective problems in terms of the coverage measure. The same conclusion was derived in Knowles & Corne (2000c).

In the comparative studies of Jaszkiewicz (2001, 2002b), a more sophisticated greedy repair algorithm was used in the MOGLS where the items were removed in the increasing order of the following ratio:

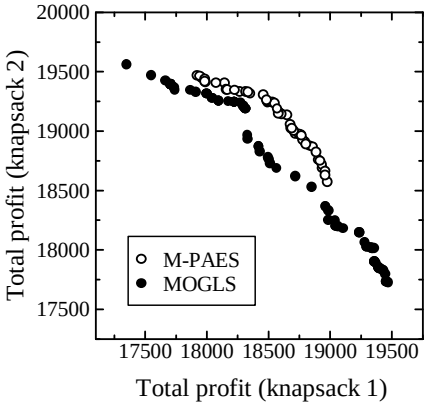

Figure 4: Results on the 2-500 problem.

$$\sum_{i=1}^{k} \lambda_i p_{ij} \Big/ \sum_{i=1}^{k} w_{ij}, \quad j = 1,2,...,n. \tag{6}$$

That is, the current weight vector $(\lambda_1, \lambda_2, ..., \lambda_k)$ was taken into account. Note that this greedy repair algorithm can be used only for the MOGLS with the scalar fitness function. We executed computational experiments using

the MOGLS with this greedy repair algorithm. Then we compared the MOGLS with the M-PAES where only the MOGLS used the sophisticated greedy repair algorithm. Experimental results are shown in Fig. 5 and Fig. 6. From these figures, we can see that the MOGLS outperformed the M-PAES. The same conclusion was derived in Jaszkiewicz (2001, 2002b). It should be noted that the MOGLS and the M-PAES did not use the same greedy repair algorithm in Fig. 5 and Fig. 6.

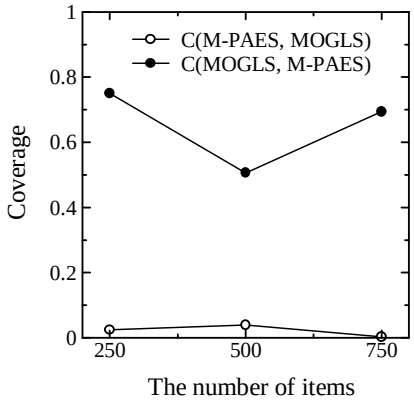

Figure 5: Results on 2-objective problems. Different greedy repair algorithms were used in the MOGLS and the M-PAES.

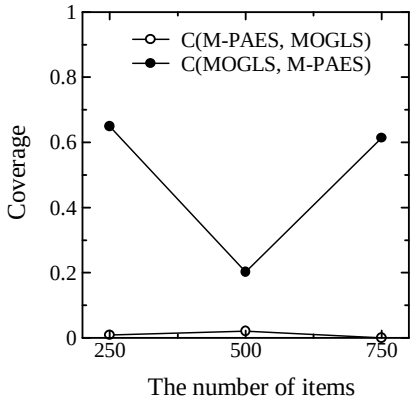

Figure 6: Results on 3-objective problems. Different greedy repair algorithms were used in the MOGLS and the M-PAES.

3.2 EFFECT OF POPULATION SIZE

In Knowles & Corne (2000c), the population size (i.e., the size of *CS*) in the MOGLS was specified as 100. Jaszkiewicz (2001, 2002b) claimed that the poor performance of the MOGLS in Knowles & Corne (2000c) was due to this parameter specification and the use of the simple greedy repair algorithm of Zitzler & Thiele (1999). We examine the performance of the MOGLS with the sophisticated greedy repair algorithm using various specifications of $|CS|$. Since $|CS|$ is determined by the size of the initial population S and the value of K as $|CS| = K \times |S|$, we varied the size of S as $|S| = 5$, 10, 20, 50, 100, 200, 500 and used the constant value of K ($K = 20$). Experimental results are summarized in Fig. 7 and Fig. 8 where the M-PAES was executed under the conditions of Table 1 as in the previous subsection. From those figures, we can see that the performance of the MOGLS was not so sensitive to the size of *CS* if it was not too large. It should be noted that the MOGLS with the sophisticated greedy repair algorithm was compared with the M-PAES with the simple greedy repair algorithm.

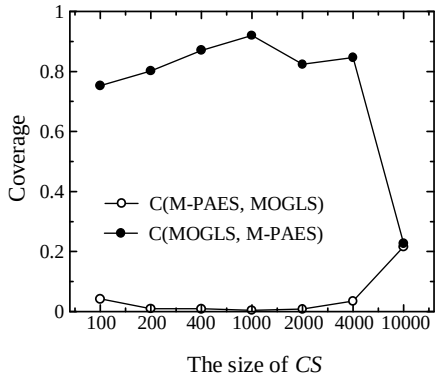

Figure 7: Results on the 2-250 problem. Different greedy repair algorithms were used in the MOGLS and the M-PAES.

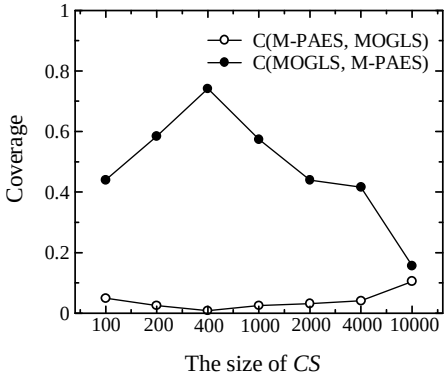

Figure 8: Results on the 2-500 problem. Different greedy repair algorithms were used in the MOGLS and the M-PAES.

We also performed computational experiments using the same simple greedy repair algorithm in the MOGLS and the M-PAES. Experimental results are summarized in Fig. 9 and Fig. 10. From these figures, we can see that the M-PAES outperformed the MOGLS on the two-objective test problems in terms of the convergence speed to the Pareto front when they were compared under the same greedy repair algorithm. As shown in Figs. 7-10, the effect of greedy repair is much more significant than the specification of the population size.

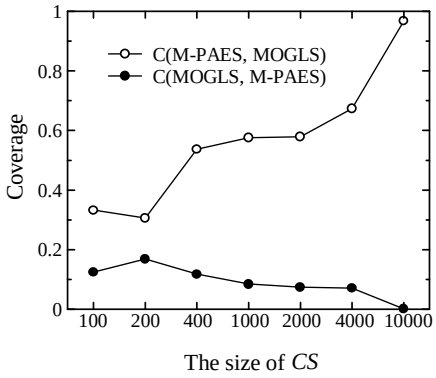

Figure 9: Results on the 2-250 problem. The same greedy repair algorithm was used in the MOGLS and the M-PAES.

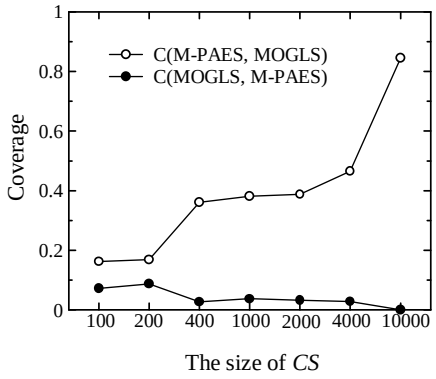

Figure 10: Results on the 2-500 problem. The same greedy repair algorithm was used in the MOGLS and the M-PAES.

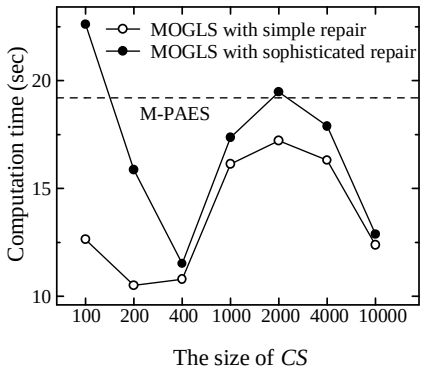

Figure 11: Average CPU time for the 2-500 problem.

In Fig. 11, we show the average CPU time of the M-PAES and the two versions of the MOGLS for the 2-500 problem (i.e., average CPU time in the computational experiments in Fig. 8 and Fig. 10). It should be noted that the M-PAES was executed under the conditions of Table 1 while the various values of the size of the current population (i.e., $|CS|$) were examined for the MOGLS. From Fig. 11, we can see that the effect of $|CS|$ on the average CPU time was not monotonic. We can also see that the average CPU time did not strongly depend on the implementation of greedy repair. In Fig. 11., the M-PAES needed longer CPU time than the MOGLS in many cases.

4. MODIFICATION OF M-PAES

As shown in Fig. 4, the M-PAES cannot find a variety of non-dominated solutions over a wide range of each objective. Moreover, as shown in Figs. 1-3, the relative performance of the M-PAES in comparison with the MOGLS degrades as the problem size increases. Even when they used the same simple greedy repair algorithm, the MOGLS outperformed the M-PAES on the large test problems in Fig. 2 and Fig. 3 (i.e., 3-750 and 4-750).

We try to improve the performance of the M-PAES by using the scalar fitness function with a random weight vector in its selection procedure. In the original M-PAES, a pair of parent solutions is randomly chosen from $P \bigcup G$ for generating an offspring. We modify this selection procedure as follows: A pair of parent solutions is chosen by tournament selection from $P \bigcup G$ based on the scalar fitness function with a random weight vector. When another pair of parent solutions is to be chosen, the weight vector is randomly updated.

In Fig. 12, we compare our modified M-PAES with the original M-PAES by depicting a single solution set obtained by a single run of each algorithm for the 2-750 test problem. It should be noted that our modified M-PAES is exactly the same as the original M-PAES when the tournament size is one. By increasing the tournament size, the selection pressure is increased and more similar parents are likely to be selected (see Ishibuchi et al. (2003)). Fig. 12 suggests a possibility that the use of the scalar fitness function can improve the search ability of the M-PAES on multiobjective 0/1 knapsack problems.

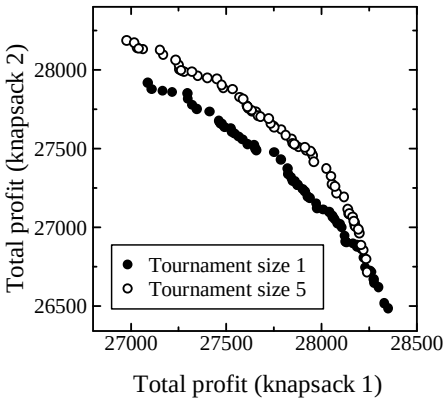

Figure 12: Results by our modified M-PAES on the 2-750 problem. The simple greedy repair algorithm was used.

5. CONCLUDING REMARKS

We compared the MOGLS of Jaszkiewicz (2002a) and the M-PAES of Knowles & Corne (2000b) with each other through computational experiments on multiobjective 0/1 knapsack problems. We showed that the MOGLS with the sophisticated repair algorithm based on the current weight vector in the scalar fitness function clearly outperformed the M-PAES with the simple repair algorithm. When they used the same simple repair algorithm, the M-PAES performed better on two-objective and three-objective test problems while the MOGLS performed better on four-objective test problems. We also tried to improve the performance of the M-PAES by the use of the scalar fitness function in its selection procedure. This modification improved the search ability of the M-PAES on multiobjective 0/1 knapsack problems. While we could not include experimental results due to the page limitation, we observed the improvement in the search ability of the M-PAES on multiobjective 0/1 knapsack problems by the introduction of mating restriction schemes (Ishibuchi & Shibata (2003a, 2003b)). In our computational experiments, we always used the simple repair algorithm in the M-PAES as in the existing comparative studies. We can, however, use the sophisticated repair algorithm in the M-PAES as in the MOGLS. As expected, its use significantly improved the performance of the M-PAES on multiobjective 0/1 knapsack problems (see Fig. 13).

The authors would like to thank the financial support from Japan Society for the Promotion of Science (JSPS) through Grand-in-Aid for Scientific Research (B): KAKENHI (14380194).

References

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T.: A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II, *IEEE Trans. on Evolutionary Computation* 6 (2002) 182-197.

Ishibuchi, H., and Murata, T.: A Multi-Objective Genetic Local Search Algorithm and Its Application to Flowshop Scheduling, *IEEE Trans. on Systems, Man, and Cybernetics - Part C: Applications and Reviews* 28 (1998) 392-403.

Ishibuchi, H., and Shibata, Y.: An Empirical Study on the Effect of Mating Restriction on the Search Ability of EMO Algorithms, *Proc. of Second International Conference on Evolutionary Multi-Criterion Optimization* (2003a) (in press).

Ishibuchi, H., and Shibata, Y.: A Similarity-Based Mating Scheme for Evolutionary Multiobjective Optimization, *Proc. of 2003 Genetic and Evolutionary Computation Conference* (2003b) (in press).

Ishibuchi, H., Yoshida, T., and Murata, T.: Balance between Genetic Search and Local Search in Memetic Algorithms for Multiobjective Permutation Flowshop Scheduling, *IEEE Trans. on Evolutionary Computation* (2003) (in press).

Jaszkiewicz, A.: Comparison of Local Search-based Metaheuristics on the Multiple Objective Knapsack Problem, *Foundations of Computing and Decision Sciences* 26 (2001) 99-120.

Jaszkiewicz, A.: Genetic Local Search for Multi-Objective Combinatorial Optimization, *European Journal of Operational Research* 137 (2002a) 50-71.

Jaszkiewicz, A.: On the Performance of Multiple-Objective Genetic Local Search on the 0/1 Knapsack Problem – A Comparative Experiment, *IEEE Trans. on Evolutionary Computation* 6 (2002b) 402-412.

Knowles, J. D., and Corne, D. W.: Approximating the Nondominated Front using Pareto Archived Evolution Strategy, *Evolutionary Computation* 8 (2000a) 149-172.

Knowles, J. D., and Corne, D. W.: M-PAES: A Memetic Algorithm for Multiobjective Optimization, *Proc. of 2000 Congress on Evolutionary Computation* (2000b) 325-332.

Knowles, J. D., and Corne, D. W.: A Comparison of Diverse Approaches to Memetic Multiobjective Combinatorial Optimization, *Proc. of 2000 Genetic and Evolutionary Computation Conference Workshop Program* (2000c) 103-108.

Knowles, J. D., and Corne, D. W.: On Metrics for Comparing Non-Dominated Sets, *Proc. of 2002 Congress on Evolutionary Computation* (2002) 711-716.

Zitzler, E., Deb, K., and Thiele, L.: Comparison of Multiobjective Evolutionary Algorithms: Empirical Results, *Evolutionary Computation* 8 (2000) 173-195.

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

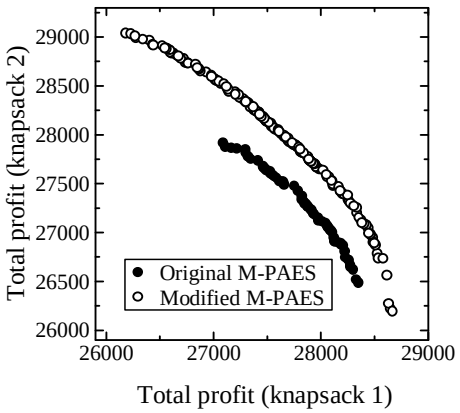

Figure 13: Results on the 2-750 problem. The original M-PAES used the simple greedy repair algorithm while our modified M-PAES with the tournament size 5 in Section 4 used the sophisticated greedy repair algorithm for comparison.