Effects of Repair Procedures on the Performance of EMO Algorithms for Multiobjective 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

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

Abstract - Multiobjective 0/1 knapsack problems have been used for examining the performance of EMO (evolutionary multiobjective optimization) algorithms in the literature. In this paper, we demonstrate that their performance on such a test problem strongly depends on the choice of a repair procedure. We show through computational experiments that much better results are obtained from greedy repair based on a weighted scalar fitness function than the maximum profit/weight ratio, which has been often used for ordering items in many studies. This observation explains several reported results in comparative studies about the superiority of EMO algorithms with a weighted scalar fitness function. It is also shown that the performance of EMO algorithms based on Pareto ranking is significantly improved by the use of the weighted scalar fitness function in repair procedures. We also examine randomized greedy repair where items are ordered based on the profit/weight ratio with respect to a randomly selected knapsack.

1. Introduction

Evolutionary multiobjective optimization (EMO) is a very active research area in the field of evolutionary computation (see, for example, Deb [1] and Coello et al. [2]). Since the study of Zitzler & Thiele [3], multiobjective 0/1 knapsack problems have been frequently used in computational experiments for examining the performance of various EMO algorithms (e.g., Knowles & Corne [4], [5], Zitzler et al. [6], Jaszkiewicz [7], [8], Ishibuchi et al. [9], [10]). When EMO algorithms are applied to knapsack problems, unfeasible solutions are often generated by genetic operations. That is, generated solutions do not always satisfy constraint conditions. Thus repair procedures were used in the above-mentioned studies [3]-[10] for deriving feasible solutions from unfeasible ones.

Zitzler & Thiele [3] used a greedy repair procedure where items were removed in the increasing order of the maximum profit/weight ratio over all knapsacks. This repair procedure was also used in Knowles & Corne [4], [5], Zitzler et al. [6] and Ishibuchi et al. [9]. Knowles & Corne [5] showed in their computational experiments that their memetic Pareto archived evolution strategy (M-PAES) outperformed the multiobjective genetic local search (MOGLS) algorithm of Jaszkiewicz [11] when both algorithms used this repair procedure. On the other hand, Jaszkiewicz [7], [8] used a more sophisticated greedy repair procedure based on a weighted scalar fitness function in his MOGLS algorithm. It was shown in his comparative studies [7], [8] that his MOGLS algorithm with the repair procedure based on the weighted scalar fitness function outperformed the M-PAES based on the maximum profit/weight ratio. In our former study [10], we examined the performance of the MOGLS and the M-PAES using these two greedy repair procedures.

In this paper, we compare the following three greedy repair procedures through computational experiments on multiobjective 0/1 knapsack problems:

1. **Maximum ratio repair**: This is the greedy repair based on the maximum profit/weight ratio.
2. **Random ratio repair**: In this procedure, items are removed in the increasing order of the profit/weight ratio with respect to a randomly chosen knapsack. A different knapsack is randomly chosen for the repair of each unfeasible solution.
3. **Weighted scalar repair**: In this procedure, items are removed in the increasing order of the profit/weight ratio with respect to the weighted scalar fitness function with a random weight vector. A different weight vector is randomly specified for the repair of each unfeasible solution.

In the maximum ratio repair, the same order of items is used for the repair of all unfeasible solutions. On the other hand, a different order of items is used for each unfeasible solution in the other repair procedures. We show in this paper that such a different order of items has a positive effect on the diversity of obtained solutions by EMO algorithms.

We use four well-known powerful EMO algorithms in our computational experiments. They are the strength Pareto evolutionary algorithm (SPEA) of Zitzler & Thiele [3], the elitist non-dominated sorting genetic algorithm (NSGA-II) of Deb et al. [12], the M-PAES of Knowles & Corne [4], and the MOGLS of Jaszkiewicz [11]. The first

three algorithms use Pareto ranking for evaluating each solution while the last one is based on a weighted scalar fitness function. Thus it is not straightforward to incorporate the weighted scalar repair into those EMO algorithms except for the MOGLS of Jaszkiewicz [11]. The SPEA and the NSGA-II are pure EMO algorithms while the M-PAES and the MOGLS are hybrid algorithms often called memetic algorithms.

In this paper, we first describe the three repair procedures in Section 2. Then we demonstrate through computational experiments that the performance of the four EMO algorithms strongly depends on the choice of a repair procedure in Section 3. In Section 4, we further discuss characteristic features of each repair procedure. Finally Section 5 concludes this paper.

2. Multiobjective 0/1 Knapsack Problems

2.1 Problem Formulation

Multiobjective 0/1 knapsack problems with k knapsacks (i.e., k objectives) and n items in Zitzler & Thiele [3] can be written 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 an n-dimensional binary vector (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.

Each solution is handled as a binary string of the length n in EMO algorithms. When new solutions are generated by genetic operations, they are often unfeasible. For deriving feasible solutions from unfeasible ones, repair procedures have been used in the literature. In the following subsections, we briefly explain three greedy repair procedures.

2.2 Maximum Ratio Repair

Zitzler & Thiele [3] used a greedy repair procedure where items were removed in the increasing order of the maximum profit/weight ratio q_j over all knapsacks:

$$q_j = \max\{p_{ij} / w_{ij} \mid i = 1,2,...,k\}, \quad j = 1,2,...,n. \quad (4)$$

This maximum ratio repair has been used in many studies on EMO algorithms [3]-[6], [9].

2.3 Random Ratio Repair

In the maximum ratio repair, items are always removed in the same order defined by q_j in (4). This may have a negative effect on the diversity of obtained solutions by EMO algorithms. For increasing the diversity of obtained solutions, we examine the use of a randomized greedy repair procedure where items are removed in the increasing order of the profit/weight ratio (i.e., p_{ij} / w_{ij}) with respect to a randomly selected knapsack j. A different knapsack is randomly selected for the repair of each unfeasible solution in this random ratio repair.

2.4 Weighted Scalar Repair

While Pareto ranking was used for evaluating solutions in many EMO algorithms, the following weighted scalar fitness function was used in some MOGLS algorithms (e.g., Jaszkiewicz [7], [8], [11] and Ishibuchi et al. [13], [14]):

$$f(\mathbf{x}, \boldsymbol{\lambda}) = \sum_{i=1}^{k} \lambda_i f_i(\mathbf{x}), \quad (5)$$

where

$$\forall i \ \lambda_i \ge 0 \ \text{ and } \ \sum_{i=1}^{k} \lambda_i = 1. \quad (6)$$

In the MOGLS of Jaszkiewicz [11], the weighted scalar fitness function in (5) was used in the following manner. When a pair of parent solutions is to be selected, first the weight vector $\boldsymbol{\lambda} = (\lambda_1, ..., \lambda_k)$ 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 a pair of parent solutions is randomly chosen from those K solutions in order to generate offspring by genetic operations from the selected pair. A local search procedure is applied to the generated offspring using the scalar fitness function with the current weight vector. The same weighted scalar fitness function was also used in a greedy repair procedure where items were removed in the increasing order of the following ratio:

$$q_j = \sum_{i=1}^{k} \lambda_i p_{ij} \Big/ \sum_{i=1}^{k} w_{ij}, \quad j = 1,2,...,n. \quad (7)$$

It should be noted that this greedy repair procedure is directly applicable only to EMO algorithms with the weighted scalar fitness function. In other EMO algorithms, we use this greedy repair procedure by randomly updating the weight vector $\boldsymbol{\lambda} = (\lambda_1, ..., \lambda_k)$ for each unfeasible solution. That is, a randomly specified different weight vector is assigned to each unfeasible solution.

3. Computational Experiments

3.1 Test Problems

In our computational experiments, we use the nine multiobjective 0/1 knapsack problems of Zitzler & Thiele [3]. 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 objectives and n is the number of items (see their general form in (1)-(3)).

Table 1: Parameter values in our computational experiments.

Problems	Population Size						*K*	*l_fails*	*l_opt*	*max_evals*
	Main Population				Secondary	Initial				
	SPEA	NSGA-II	M-PAES	MOGLS	SPEA	MOGLS	MOGLS	M-PAES & MOGLS		
2-250	120	150	30	3,000	38	150	20	20	100	75,000
2-500	160	200	40	4,000	50	200	20	20	100	100,000
2-750	200	250	50	5,000	63	250	20	5	20	125,000
3-250	160	200	40	4,000	50	200	20	20	50	100,000
3-500	200	250	50	5,000	63	250	20	20	50	125,000
3-750	240	300	60	6,000	75	300	20	5	20	150,000
4-250	200	250	50	5,000	63	250	20	20	50	125,000
4-500	240	300	60	6,000	75	300	20	20	50	150,000
4-750	280	350	70	7,000	88	350	20	5	20	175,000

3.2 EMO Algorithms

In our computational experiments, we use the following four EMO algorithms:

SPEA of Zitzler & Thiele [3]
NSGA-II of Deb et al. [12]
M-PAES of Knowles & Corne [4]
MOGLS of Jaszkiewicz [11]

The SPEA and the NSGA-II are well-known pure EMO algorithms while the M-PAES and the MOGLS are hybrid EMO algorithms often called memetic algorithms. Only the MOGLS uses the weighted scalar fitness function in (5) for solution evaluation. In the other algorithms, Pareto ranking together with some form of crowding is used for solution evaluation.

The main characteristic feature of the SPEA is the use of a secondary population where a prespecified number of non-dominated solutions among examined ones are stored separately from the current population. Those non-dominated solutions can be viewed as elite solutions. While the NSGA-II does not explicitly use such a secondary population, a kind of elitism is realized by its generation update procedure.

The M-PAES was proposed in [4] by introducing the concept of a population and a recombination operation into a multiobjective local search algorithm: (1+1)-PAES [15]. 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 solution evaluation in local search while a pair of parent solutions is randomly chosen from $P \bigcup G$ for generating an offspring.

In the M-PAES and the MOGLS, we use two parameters for terminating local search for each solution as in Knowles & Corne [5] and Jaszkiewicz [7]. 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*). In both algorithms, a neighboring solution is generated by applying the standard bit-flip mutation operation with a probability of $4/n$ to each bit of the current solution where n is the number of items (i.e., n is the string length). This operation is also used as mutation in the pure EMO algorithms (i.e., SPEA and NSGA-II). No mutation is used in the memetic EMO algorithms (i.e., M-PAES and MOGLS). The standard one-point crossover is used in all the four algorithms. The crossover rate is specified as 0.8 for the pure EMO algorithms and 1.0 for the memetic EMO algorithms.

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. Our parameter specifications are almost the same as those in Zitzler & Thiele [3]. For some parameters in the M-PAES and the MOGLS, we use almost the same specifications as Knowles & Corne [4], [5] and Jaszkiewicz [7].

3.3 Experimental Results

For visually demonstrating the effect of the choice of a repair procedure on the performance of the four EMO algorithms, we show a set of solutions obtained by a single run of each algorithm on the two-objective test problems in Figs. 1-12. In each figure, we show three solution sets obtained by each algorithm with the three different repair procedures.

From these figures, we can see that the best results were obtained from the weighted scalar repair procedure. This procedure outperformed the maximum ratio repair in terms of not only the diversity of obtained solutions but also the convergence speed to the Pareto front. The superiority of the weighted scalar repair can be more or less observed in all figures independent of the problem size and the EMO algorithm. This observation supports the experimental results in Jaszkiewicz [7], [8] where the MOGLS with the weighted scalar repair outperformed the M-PAES and the SPEA with the maximum ratio repair. The reported superiority of the MOGLS on multiobjective 0/1 knapsack problems in [7], [8] can be (at least partially) attributed to the superiority of the weighted scalar repair over the maximum ratio repair.

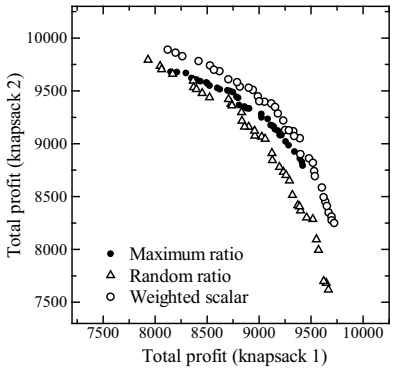

Figure 1: SPEA on 2-250 problem.

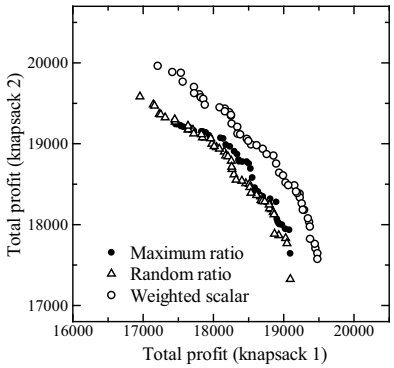

Figure 2: SPEA on 2-500 problem.

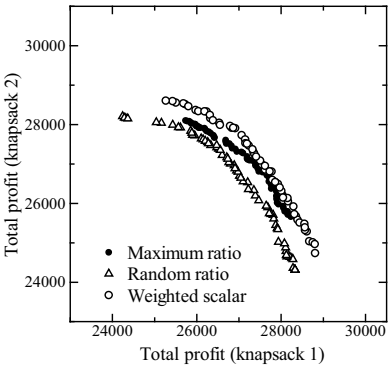

Figure 3: SPEA on 2-750 problem.

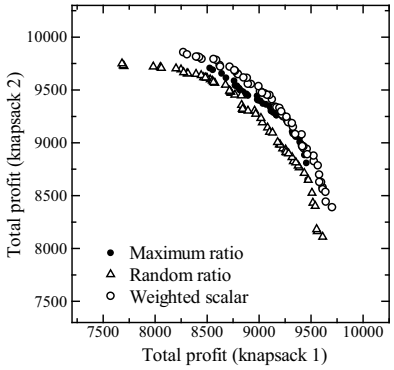

Figure 4: NSGA-II on 2-250 problem.

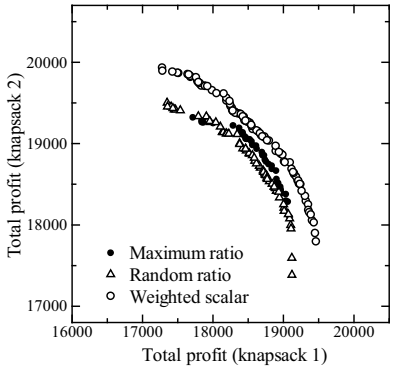

Figure 5: NSGA-II on 2-500 problem.

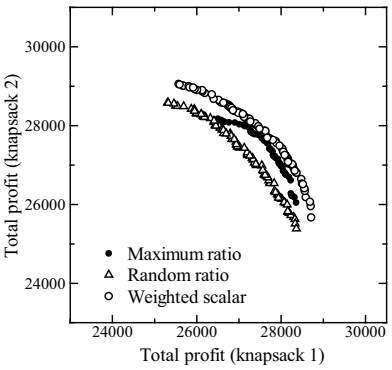

Figure 6: NSGA-II on 2-750 problem.

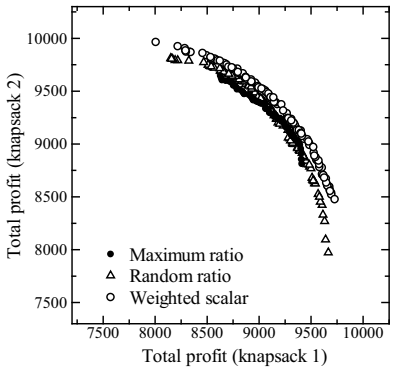

Figure 7: M-PAES on 2-250 problem.

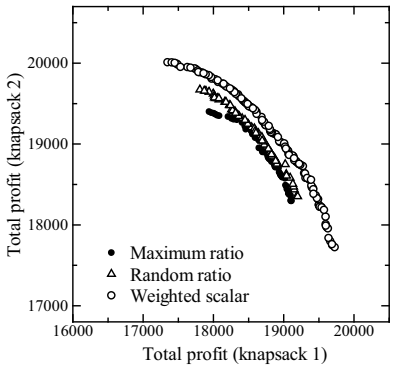

Figure 8: M-PAES on 2-500 problem.

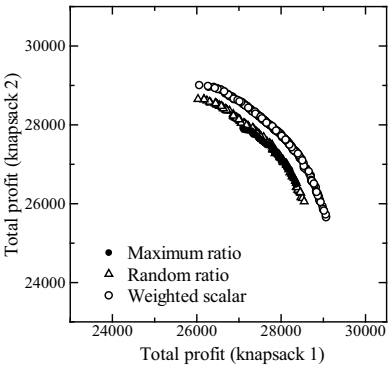

Figure 9: M-PAES on 2-750 problem.

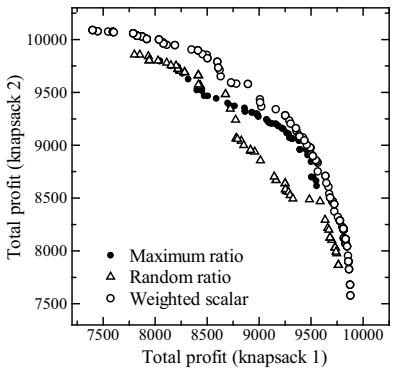

Figure 10: MOGLS on 2-250 problem.

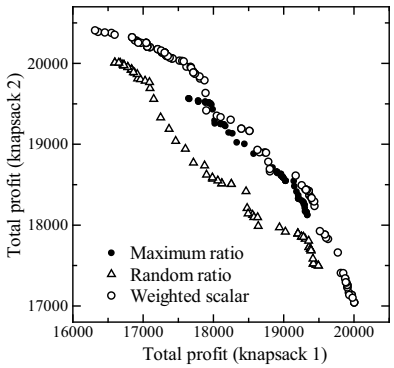

Figure 11: MOGLS on 2-500 problem.

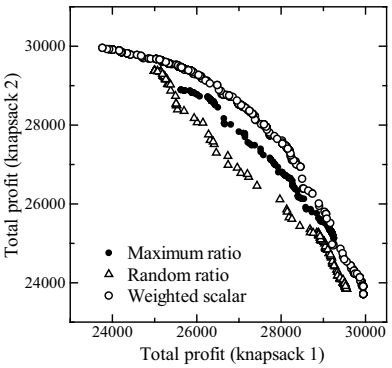

Figure 12: MOGLS on 2-750 problem.

We can also see from Figs. 1-12 that more diverse solutions were obtained from the random ratio repair than the maximum ratio repair. This observation suggests that a different order of items for each unfeasible solution has a positive effect on the diversity of obtained solutions. While the weighted scalar repair improved both the diversity of obtained solutions and the convergence speed to the Pareto front, the random ratio repair degraded the convergence speed in many cases in Figs. 1-12.

Let us further examine the effect of the choice of a repair procedure on the performance of EMO algorithms. In Tables 2-3, we show the average number of obtained solutions by each algorithm over ten runs. From Table 3, we can see that much more solutions were obtained by the memetic EMO algorithms with the weighted scalar repair (i.e., Weight in Table 3) than those with the other repair procedures. On the other hand, the number of obtained solutions does not strongly depend on the choice of a repair procedure in the case of the pure EMO algorithms in Table 2. This is because the population size has a dominant effect on the number of solutions obtained by the pure EMO algorithms.

Table 2: Average number of solutions by the pure EMO algorithms. The larger values mean the better results.

Problem	SPEA			NSGA-II		
	Max	Rand	Weight	Max	Rand	Weight
2-250	37	38	38	38	53	55
2-500	46	49	49	46	57	87
2-750	56	61	62	55	68	103
3-250	50	50	50	148	179	176
3-500	63	63	62	199	234	232
3-750	75	75	75	248	287	281
4-250	63	63	63	213	247	248
4-500	75	75	75	284	311	315
4-750	88	88	88	335	376	385

Table 3: Average number of solutions by the memetic EMO algorithms. The larger values mean the better results.

Problem	M-PAES			MOGLS		
	Max	Rand	Weight	Max	Rand	Weight
2-250	55	75	106	82	79	173
2-500	48	83	144	71	67	123
2-750	59	113	178	364	300	639
3-250	429	570	886	370	333	638
3-500	454	605	1183	341	297	578
3-750	342	585	1178	1191	905	2072
4-250	1231	1484	2415	741	652	1256
4-500	1287	1865	2702	678	672	1270
4-750	1583	2011	3229	2981	2017	4455

We also examine the diversity using the width of the range of obtained solutions. The width of the range of a solution set S is measured for each objective $f_i(\mathbf{x})$ as

$$width_i(S) = \max\{f_i(\mathbf{x}) \mid \mathbf{x} \subset S\} - \min\{f_i(\mathbf{x}) \mid \mathbf{x} \subset S\}. \quad (8)$$

Then the sum of the widths over the k objectives is calculated as

$$width(S) = \sum_{i=1}^{k} width_i(S). \quad (9)$$

In Tables 4-5, we show the average sum of the widths of the obtained solution set by each algorithm over ten runs. From these tables, we can see that wider solution sets were obtained from the random ratio repair and the weighted scalar repair than the maximum ratio repair. This observation suggests that a different order of items for each unfeasible solution has a positive effect on the diversity of obtained solutions. We can also see that the widest solution sets were obtained by the MOGLS with the weighted scalar repair (i.e., Weight in Table 5).

Table 4: Average sum of widths by the pure EMO algorithms. The larger values mean the better results.

Problem	SPEA			NSGA-II		
	Max	Rand	Weight	Max	Rand	Weight
2-250	2373	3397	3256	1972	3298	3131
2-500	2965	4386	4503	2539	3617	4324
2-750	4457	7333	7491	3729	5694	6296
3-250	3290	4171	3753	3394	4644	4818
3-500	4743	6607	5781	4388	6343	6088
3-750	5768	8320	7288	5015	8012	7466
4-250	3965	5255	4550	3829	5998	5409
4-500	6096	7468	6383	5910	9255	7804
4-750	7849	9615	8619	6642	10703	10008

Table 5: Average sum of widths by the memetic EMO algorithms. The larger values mean the better results.

Problem	M-PAES			MOGLS		
	Max	Rand	Weight	Max	Rand	Weight
2-250	1828	3183	3394	2814	3882	4749
2-500	2105	3312	4846	3952	5682	7224
2-750	2503	5130	6040	7207	9976	12642
3-250	3529	4786	4935	4349	6005	7530
3-500	4260	7144	7085	6600	10132	13101
3-750	4364	6570	7202	8339	13856	18884
4-250	4425	6818	6230	4820	7914	9493
4-500	6384	9690	8911	8392	14025	17872
4-750	6443	10604	9569	10467	18225	25580

We have already examined the effect of the choice of a repair procedure on the diversity of obtained solutions by each EMO algorithm through computational experiments in Tables 2-5. Now let us examine its effect on the convergence speed to the Pareto front. For this purpose, we use the generational distance measure.

Let S be a solution set obtained by an EMO algorithm. The proximity of S to the Pareto front is evaluated by the generational distance defined as follows [1], [2], [16]:

$$\mathrm{GD}(S) = \frac{1}{|S|} \sum_{\mathbf{x} \subset S} \min\{d_{\mathbf{xy}} \mid \mathbf{y} \subset S^*\}, \quad (10)$$

where S^* is a set of reference solutions and $d_{\mathbf{xy}}$ is the distance between a solution $\mathbf{x}$ and a reference solution $\mathbf{y}$ in the k-dimensional objective space:

$$d_{\mathbf{xy}} = \sqrt{(f_1(\mathbf{x}) - f_1(\mathbf{y}))^2 + \cdots + (f_k(\mathbf{x}) - f_k(\mathbf{y}))^2}. \quad (11)$$

In our computational experiments, the reference solution set S^* was generated for each test problem in the following manner. We applied the four EMO algorithms with the three repair procedures to each test problem ten times (i.e., 120 times to each test problem in total). Among the obtained 120 solution sets for each test problem, we picked up only non-dominated solutions to construct the reference solution set S^*.

Experimental results are shown in Tables 6-7. From these tables, we can see that the best results with respect to the convergence speed to the Pareto front were obtained from the weighted scalar repair. We can also see that the random ratio repair degraded the convergence speed.

For examining both the diversity of obtained solutions and the convergence speed to the Pareto front, we calculate the $\mathrm{D1_R}$ measure defined as follows [16]:

$$\mathrm{D1_R}(S) = \frac{1}{|S^*|} \sum_{\mathbf{y} \subset S^*} \min\{d_{\mathbf{xy}} \mid \mathbf{x} \subset S\}. \quad (12)$$

It should be noted that $\mathrm{D1_R}(S)$ in (12) is the average distance from each reference solution $\mathbf{y}$ to its nearest obtained solution in S while $\mathrm{GD}(S)$ in (10) is the average distance from each obtained solution $\mathbf{x}$ to its nearest reference solution in S^*. The generational distance $\mathrm{GD}(S)$ measures the proximity of the obtained solution set S to the reference solution set S^*. On the other hand, $\mathrm{D1_R}$ evaluates how well the obtained solution set S approximates the reference solution set S^*.

Tables 8-9 show the average value of the $\mathrm{D1_R}$ measure obtained by each EMO algorithm for each test problem. From these tables, we can see that the best results in terms of the $\mathrm{D1_R}$ measure were obtained from the EMO algorithms with the weighted scalar repair. It should be noted that better results were obtained from the random ratio repair than the maximum ratio repair in some cases in Tables 8-9 while better results were obtained from the maximum ratio repair in Tables 6-7.

Table 6: Average generational distance by the pure EMO algorithms. The smaller values mean the better results.

Problem	SPEA			NSGA-II		
	Max	Rand	Weight	Max	Rand	Weight
2-250	208	309	79	110	202	34
2-500	581	696	264	316	426	104
2-750	946	1287	656	503	798	224
3-250	531	727	406	207	295	81
3-500	1037	1911	997	368	662	164
3-750	2050	2952	1646	815	1258	295
4-250	688	1219	595	177	452	122
4-500	1730	2573	1358	606	953	198
4-750	2827	4016	2346	922	1799	406

Table 7: Average generational distance by the memetic EMO algorithms. The smaller values mean the better results.

Problem	M-PAES			MOGLS		
	Max	Rand	Weight	Max	Rand	Weight
2-250	91	105	16	140	232	24
2-500	228	257	26	379	499	77
2-750	319	418	38	309	394	79
3-250	159	189	44	216	272	76
3-500	275	562	64	400	717	172
3-750	717	829	84	466	735	209
4-250	148	338	88	135	353	130
4-500	605	786	137	478	782	267
4-750	780	1363	144	476	1054	342

Table 8: Average value of the $\mathrm{D1_R}$ measure by the pure EMO algorithms. The smaller values mean the better results.

Problem	SPEA			NSGA-II		
	Max	Rand	Weight	Max	Rand	Weight
2-250	291	319	131	275	224	106
2-500	702	710	323	580	529	212
2-750	1209	1338	747	974	965	463
3-250	606	735	495	356	340	190
3-500	1252	2022	1131	690	784	396
3-750	2431	3359	2062	1472	1603	946
4-250	818	1231	679	419	490	282
4-500	2000	2873	1604	943	1121	609
4-750	3442	4570	2996	1879	2372	1371

Table 9: Average value of the $\mathrm{D1_R}$ measure by the memetic EMO algorithms. The smaller values mean the better results.

Problem	M-PAES			MOGLS		
	Max	Rand	Weight	Max	Rand	Weight
2-250	280	151	74	226	300	37
2-500	596	437	117	488	637	134
2-750	1067	729	371	641	855	141
3-250	301	235	110	310	363	104
3-500	646	670	242	611	1074	310
3-750	1497	1340	863	1129	1390	324
4-250	340	349	167	314	444	147
4-500	898	956	426	784	1033	358
4-750	1779	1963	1197	1291	1650	485

4. Discussions on Experimental Results

Our experimental results in Section 3 demonstrated that the best results were obtained from the weighted scalar repair procedure in terms of both the diversity of solutions and the convergence speed to the Pareto front. It was also shown that the choice of a repair procedure is more important than the choice of an EMO algorithm (e.g., see Figs. 1-12). In this section, we further examine each repair procedure.

For examining the effect of each repair procedure separately from the EMO algorithms, we perform the following computational experiment. First we randomly generate an n-dimensional binary vector $\mathbf{x} \mid (x_1, ..., x_n)$ by assigning 0 with the probability 0.4 and 1 with the probability 0.6 to each x_j. Then we generate a feasible solution using one of the three repair procedures if the randomly generated binary vector is unfeasible. When the randomly generated binary vector is feasible, we return to the first step in order to generate another binary vector. This two-step procedure is iterated for generating a prespecified number of feasible solutions from unfeasible binary vectors.

In Fig. 13, we show the trajectories of the repair from 10 unfeasible binary vectors by the maximum ratio repair for the 2-500 test problem. In this figure, unfeasible and feasible solutions are depicted by open circles and closed circles, respectively. It should be noted that the same order of items is always used for all the 10 unfeasible binary vectors in this repair procedure. Thus the directions of the trajectories are similar to each other in Fig. 13.

Fig. 14 shows the trajectories of the repair from the same 10 unfeasible binary vectors in the case of the random ratio repair. In this repair procedure, two orders of items are randomly chosen for each unfeasible binary vector because the 2-500 test problem involves two knapsacks. From the comparison between Fig. 13 and Fig. 14, we can see that the trajectories in Fig. 14 have more diverse directions.

Fig. 15 shows the results by the weighted scalar repair for the same 10 unfeasible binary vectors. In this repair procedure, a different order of items is used for each unfeasible binary vector. We observe various directions of the trajectories in Fig. 15.

In the same manner as in Figs. 13-15, we generate a set of 10000 feasible solutions using each repair procedure. Then we pick up only non-dominated solutions by comparing solutions with each other within each solution set. The non-dominated solutions obtained by each repair procedure (i.e., obtained from each solution set with 10000 repaired feasible solutions) are shown in Fig. 16. We can see that Fig. 16 is somewhat similar to Figs. 1-12. For example, the best results were obtained from the weighted scalar repair in Fig. 16. This observation suggests that the choice of a repair procedure has a dominant effect on the performance of EMO algorithms.

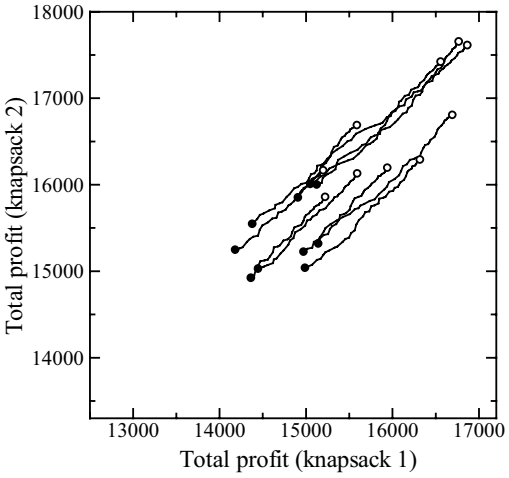

Figure 13: Maximum ratio repair.

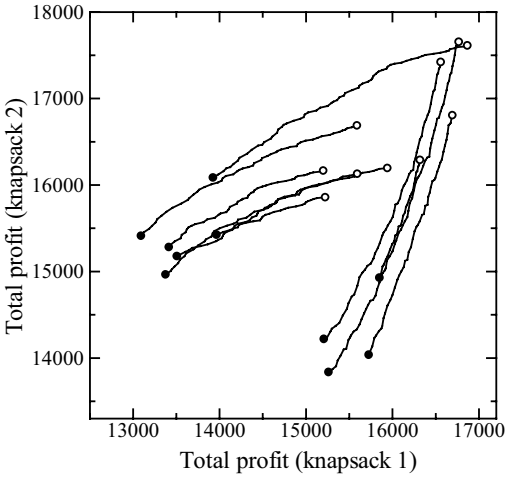

Figure 14: Random ratio repair.

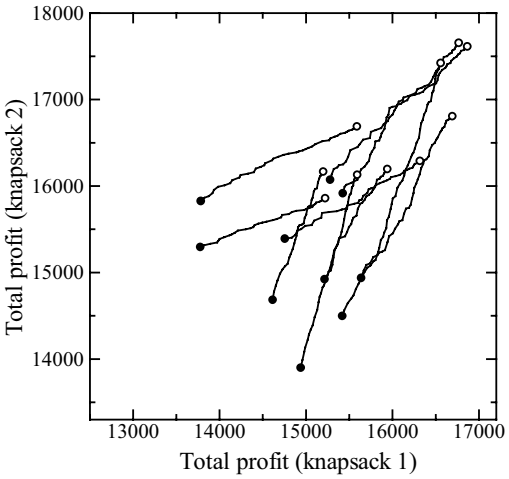

Figure 15: Weighted scalar repair.

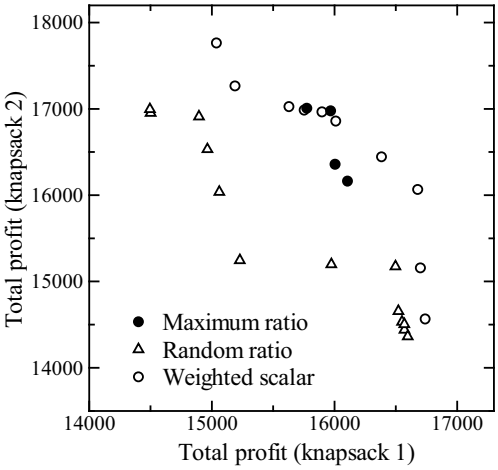

Figure 16: Non-dominated solutions obtained by each procedure.

5. Concluding Remarks

In this paper, we demonstrated that the performance of EMO algorithms on multiobjective 0/1 knapsack problems strongly depended on the choice of a repair procedure. While the repair has been often performed based on the maximum profit/weight ratio in the literature, much better results were obtained from a greedy repair procedure based on a weighted scalar fitness function in our computational experiments. We also showed that the use of this repair procedure significantly improved the performance of EMO algorithms even when they did not use any scalar fitness function for solution evaluation.

Acknowledgement

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).

Bibliography

[1] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, John Wiley & Sons, Chichester, 2001.

[2] C. A. Coello Coello, D. A. van Veldhuizen, and G. B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Kluwer Academic Publishers, Boston, 2002.

[3] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. on Evolutionary Computation*, vol. 3, no. 4, pp. 257-271, November 1999.

[4] J. D. Knowles and D. W. Corne, "M-PAES: A memetic algorithm for multiobjective optimization," *Proc. of 2000 Congress on Evolutionary Computation*, pp. 325-332, July 16-19, 2000.

[5] J. D. Knowles and D. W. Corne, "A comparison of diverse approaches to memetic multiobjective combinatorial optimization," *Proc. of 2000 Genetic and Evolutionary Computation Conference Workshop Program (WOMA I),* pp. 103-108, July 8, 2000.

[6] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," *TIK-Report* 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology, May 2001.

[7] A. Jaszkiewicz, "Comparison of local search-based metaheuristics on the multiple objective knapsack problem," *Foundations of Computing and Decision Sciences*, vol. 26, no. 1, pp. 99-120, 2001.

[8] A. Jaszkiewicz, "On the performance of multiple-objective genetic local search on the 0/1 knapsack problem - A comparative experiment," *IEEE Trans. on Evolutionary Computation*, vol. 6, no. 4, pp. 402-412, August 2002.

[9] H. Ishibuchi and Y. Shibata, "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*, pp. 433-447, April 8-11, 2003.

[10] H. Ishibuchi and S. Kaige, "Comparison of multiobjective memetic algorithms on 0/1 knapsack problems," *Proc. of 2003 Genetic and Evolutionary Computation Conference Workshop Program (WOMA IV)*, pp. 222-227, July 12, 2003.

[11] A. Jaszkiewicz, "Genetic local search for multi-objective combinatorial optimization," *European Journal of Operational Research*, vol. 137, no. 1, pp. 50-71, February 2002.

[12] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. on Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, April 2002.

[13] H. Ishibuchi and T. Murata, "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*, vol. 28, no. 3, pp. 392-403, August 1998.

[14] H. Ishibuchi, T. Yoshida, and T. Murata, "Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling," *IEEE Trans. on Evolutionary Computation*, vol. 7, no. 2, pp. 204-223, April 2003.

[15] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using Pareto archived evolution strategy," *Evolutionary Computation*, vol. 8, no. 2, pp. 149-172, Summer 2000.

[16] J. D. Knowles and D. W. Corne, "On metrics for comparing non-dominated sets," *Proc. of 2002 Congress on Evolutionary Computation*, pp. 711-716, Honolulu, May 12-17, 2002.