Parallel Problem Solving from Nature – PPSN V, Amsterdam, The Netherlands, September 1998, pages 292–301. © Springer.

Multiobjective Optimization Using Evolutionary Algorithms — A Comparative Case Study

Eckart Zitzler and Lothar Thiele

Swiss Federal Institute of Technology Zurich,
Computer Engineering and Communication Networks Laboratory (TIK),
Gloriastrasse 35, CH-8092 Zurich, Switzerland

Abstract. Since 1985 various evolutionary approaches to multiobjective optimization have been developed, capable of searching for multiple solutions concurrently in a single run. But the few comparative studies of different methods available to date are mostly qualitative and restricted to two approaches. In this paper an extensive, quantitative comparison is presented, applying four multiobjective evolutionary algorithms to an extended 0/1 knapsack problem.

1 Introduction

Many real-world problems involve simultaneous optimization of several incommensurable and often competing objectives. Usually, there is no single optimal solution, but rather a set of alternative solutions. These solutions are optimal in the wider sense that no other solutions in the search space are superior to them when *all* objectives are considered. They are known as *Pareto-optimal* solutions.

Mathematically, the concept of Pareto-optimality can be defined as follows: Let us consider, without loss of generality, a multiobjective maximization problem with m parameters (decision variables) and n objectives:

$$\text{Maximize} \quad \mathbf{y} = f(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_n(\mathbf{x})) \tag{1}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_m) \in X$ and $\mathbf{y} = (y_1, y_2, \ldots, y_n) \in Y$ are tuple. A decision vector $\mathbf{a} \in X$ is said to *dominate* a decision vector $\mathbf{b} \in X$ (also written as $\mathbf{a} \succ \mathbf{b}$) iff

$$\forall i \in \{1, 2, \ldots, n\} : f_i(\mathbf{a}) \geq f_i(\mathbf{b}) \quad \wedge \quad \exists j \in \{1, 2, \ldots, n\} : f_j(\mathbf{a}) > f_j(\mathbf{b}) \tag{2}$$

Additionally, in this study we say $\mathbf{a}$ *covers* $\mathbf{b}$ iff $\mathbf{a} \succ \mathbf{b}$ or $\mathbf{a} = \mathbf{b}$. All decision vectors which are not dominated by any other decision vector are called *nondominated* or *Pareto-optimal*.

Often, there is a special interest in finding or approximating the Pareto-optimal set, mainly to gain deeper insight into the problem and knowledge about alternate solutions, respectively. Evolutionary algorithms (EAs) seem to be especially suited for this task, because they process a set of solutions in parallel, eventually exploiting similarities of solutions by crossover. Some researcher suggest that multiobjective search and optimization might be a problem area where EAs do better than other blind search strategies [1][12].

Since the mid-eighties various multiobjective EAs have been developed, capable of searching for multiple Pareto-optimal solutions concurrently in a single run. But up to now, no extensive, quantitative comparison of different methods has been reported in literature. The few comparisons available to date are mostly qualitative and restricted to two different methods; quite often, the test problems considered are rather simple.

In this study, however, we provide a comparison which a) uses two complementary quantitative measures to evaluate the performance of the EAs, b) bases on a NP-hard test problem (0/1 knapsack problem), which represents an important class of real-world problems, and c) includes four different multiobjective EAs as well as a pure random search algorithm. The comparison focuses on the effectiveness in finding multiple Pareto-optimal solutions, disregarding its number. Nevertheless, in the case the trade-off surface is continuous or contains many points, the distribution of the nondominated solutions achieved is also an important aspect. Although we do not consider the distribution explicitly, it indirectly influences the performance of the EA.

The paper is organized in the following way. The next section gives a brief overview of evolutionary approaches in the field of multiobjective optimization and a more detailed description of the EAs considered in this investigation. Section 3 introduces a multiobjective 0/1 knapsack problem, discusses the test data sets used in the experiments and presents the chromosome coding and decoding for the EAs. Afterwards, the experimental results are summarized in Section 4, and Section 5 comprises conclusions and future perspectives.

2 Multiobjective Evolutionary Algorithms

A comprehensive overview of EAs in multiobjective optimization was published by Fonseca and Fleming [1]. The authors categorized several evolutionary approaches regarding plain aggregating approaches, population-based non-Pareto approaches and Pareto-based approaches; moreover, approaches using niche induction techniques were considered.

Aggregation methods combine the objectives into a higher scalar function which is used for fitness calculation; they produce one single solution and require profound domain knowledge which is often not available. Population-based non-Pareto approaches, however, are able to evolve multiple nondominated solutions in parallel; thereby, the population is mostly monitored for nondominated solutions. But in contrast to the Pareto-based approaches they do not make direct use of the concept of Pareto dominance. Pareto-based EAs compare solutions according to the $\succ$ relation in order to determine the reproduction probability of each individual; this kind of fitness assignment was first proposed by Goldberg [3].

Since preservation of diversity is crucial in the field of multimodal optimization, many multiobjective EAs incorporate niching techniques, the mostly implemented of which is *fitness sharing* [2]. Fitness sharing bases on the idea that individuals in a particular niche have to share the resources available, similar to nature. Thus, the fitness value of a certain individual is the more degraded the more individuals are located in its neighborhood. Neighborhood is defined in terms of a distance measure and specified by the so-called niche radius σ_{share}. Sharing can be performed both in genotypic space and phenotypic space.

In this study we consider two population-based non-Pareto EAs and two Pareto-based EAs: the Vector Evaluated Genetic Algorithm (VEGA) [10], an EA incorporating weighted-sum aggregation [4], the Niched Pareto Genetic Algorithm [5][6], and the Nondominated Sorting Genetic Algorithm (NSGA) [11]; all but VEGA use fitness sharing to maintain a population distributed along the Pareto-optimal front. Pure aggregation methods are disregarded here because they are not designed for finding a family of solutions.

2.1 Vector Evaluated Genetic Algorithm

Probably the first who recognized EAs to be applicable in multiobjective optimization was Schaffer [10]. He presented a multimodal EA called Vector Evaluated Genetic Algorithm (VEGA) which carries out selection for each objective separately. In detail, the mating pool is divided in n parts of equal size; part i is filled with individuals that are chosen at random from the current population according to objective i. Afterwards, the mating pool is shuffled and crossover and mutation are performed as usual. Schaffer implemented this method in combination with fitness proportionate selection.

2.2 Aggregation by Variable Objective Weighting

Another approach which is based on plain aggregation was introduced by Hajela and Lin [4]. They use the weighted-sum method for fitness assignment. Thereby, each objective is assigned a weight $w_i \in]0,1[$, such that $\sum w_i = 1$, and the scalar fitness value is calculated by summing up the weighted objective values $w_i \cdot f_i(\mathbf{x})$.[1] To search for multiple solutions in parallel, the weights are not fixed but coded in the genotype. The diversity of the weight combinations is promoted by phenotypic fitness sharing. As a consequence, the EA evolves solutions and weight combinations simultaneously. Finally, the authors emphasize mating restrictions to be necessary in order to "both speed convergence and impart stability to the genetic search" [4, p. 102].

2.3 Niched Pareto Genetic Algorithm

The Niched Pareto Genetic Algorithm proposed by Horn and Nafpliotis [5][6] combines tournament selection and the concept of Pareto dominance. Two competing individuals and a comparison set of other individuals are picked at random from the population; the size of the comparison set is given by the parameter t_{dom}. If one of the competing individuals is dominated by any member of the set, and the other is not, then the latter is chosen as winner of the tournament. If both individuals are dominated or not dominated, the result of the tournament is decided by sharing: The individual which has the least individuals in its

[1] Normally, the objectives values have to be scaled in the case the magnitude of each objective criterion is quite different. In this study, however, scaling was not implemented due to the nature of the test problems used.

niche (defined by σ_{share}) is selected for reproduction. Horn and Nafpliotis used phenotypic sharing on the objective values $f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_n(\mathbf{x})$ in their study.

2.4 Nondominated Sorting Genetic Algorithm

Another multiobjective EA which is based on Pareto ranking is the Nondominated Sorting Genetic Algorithm (NSGA) developed by Srinivas and Deb [11]. The fitness assignment is carried out in several steps. In each step, the nondominated solutions constituting a nondominated front are assigned the same dummy fitness value. These solutions are shared with their dummy fitness values (phenotypic sharing on the parameter values $x_1, x_2, \ldots, x_n$), and ignored in the further classification process. Finally, the dummy fitness is set to a value less than the smallest shared fitness value in the current nondominated front. Then the next front is extracted. This procedure is repeated until all individuals in the population are classified. In the original study [11], this fitness assignment method was combined with a stochastic remainder selection.

3 The Knapsack Problem

A test problem for a comparative study like this has to be chosen carefully. On the one hand, the problem should be understandable and easy to formulate so that the experiments are repeatable and verifiable. On the other hand, it ideally represents a certain class of real-world problems. Both applies to the knapsack problem: the problem description is simple, yet, the problem itself is difficult to solve (NP-hard). Moreover, due to its practical relevance it has been subject to several investigations in various fields, in particular, in the domain of evolutionary computation (e.g. [8]).

3.1 Formulation as Multiobjective Optimization Problem

Generally, a 0/1 knapsack problem consists of a set of items, weights and profits associated with each item, and an upper bound for the capacity of the knapsack. The task is to find a subset of all items which maximizes the total of the profits in the subset, yet, all selected items fit into the knapsack, i.e. the total weight does not exceed the given capacity [7].

This single-objective problem can be extended straight forward for the multiobjective case by allowing an arbitrary number of knapsacks. Formally, the multiobjective 0/1 knapsack problem considered here is defined in the following way: Given a set of m items and a set of n knapsacks, with

$$\begin{aligned} p_{i,j} &= \text{profit of item } j \text{ according to knapsack } i \\ w_{i,j} &= \text{weight of item } j \text{ according to knapsack } i \\ c_i &= \text{capacity of knapsack } i, \end{aligned}$$

find a vector $\mathbf{x} = (x_1, x_2, \ldots, x_m) \in \{0,1\}^m$, such that

$$\forall i \in \{1, 2, \ldots, n\} : \sum_{j=1}^{m} w_{i,j} \cdot x_j \leq c_i \tag{3}$$

and for which $f(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_n(\mathbf{x}))$ is maximum, where

$$f_i(\mathbf{x}) = \sum_{j=1}^{m} p_{i,j} \cdot x_j \tag{4}$$

and $x_j = 1$ iff item j is selected.

3.2 Test Data

In order to obtain reliable and sound results, we used nine different test problems, where both number of knapsacks and number of items were varied. Two, three, and four objectives were taken under consideration, in combination with 100, 250, and 500 items.

Following suggestions in [7], *uncorrelated* profits and weights were chosen, where $p_{i,j}$ and $w_{i,j}$ are random integers in the interval $[10, 100]$. The knapsack capacities were set to half the total weight regarding the corresponding knapsack: $c_i = 0.5 \sum_{j=1}^{m} w_{i,j}$. As reported in [7], about half of the items are expected to be in the optimal solution (of the single-objective problem), when this type of knapsack capacities is used.[2]

3.3 Implementation

Concerning the chromosome coding as well as the constraint handling, we draw upon results published by Michalewicz and Arabas [8]. They examined EAs with different representation mappings and constraint handling techniques on the (single) 0/1 knapsack problem. Concluding from their experiments, an approach using a vector representation and a greedy repair algorithm to correct infeasible solutions appears to be most appropriate for various kinds of knapsack capacities. We adopted this approach with a slightly modified repair mechanism.

In detail, a binary string $\mathbf{s}$ of length m is used to encode the solution $\mathbf{x} \in \{0,1\}^m$. Since many codings lead to infeasible solutions, a simple repair method r is applied to the genotype $\mathbf{s}$: $\mathbf{x} = r(\mathbf{s})$. The repair algorithm step by step removes items from the solution coded by $\mathbf{s}$ until all capacity constraints are fulfilled. The order in which the items are deleted is determined by the maximum profit/weight ratio per item ; for item j the maximum profit/weight ratio q_j is given by the equation $q_j = \max_{i=1}^{n}\{p_{i,j}/w_{i,j}\}$.[3] The items are considered in increasing order of the q_j.

4 Experiments

4.1 Methodology

In the context of this comparative study several questions arose: What quantitative measures should be used to express the quality of the outcomes so that the EAs can be compared in a meaningful way? What is the outcome of an multiobjective EA regarding a set of runs? How can side effects caused by different

[2] We also examined more restrictive capacities ($c_i = 200$) where the solutions contain only a few items; however, this had no significant influence on the relative performance of the EAs.

[3] This is a straight forward extension to the single-objective approach presented in [8] where $q_j = p_{1,j}/w_{1,j}$.

selection schemes or mating restrictions be precluded, such that the comparison is not falsified? How can the parameters of the EA, particularly the niche radius, be set appropriately? In the following we deal with these problems.

Two complementary measures were used in this study to evaluate the Pareto fronts produced by the various EAs. The first concerns the size of the objective value space which is covered by a set of nondominated solutions. In the two dimensional case each Pareto-optimal solution $\mathbf{x}$ covers an area, a rectangle, defined by the points $(0, 0)$ and $(f_1(\mathbf{x}), f_2(\mathbf{x}))$. The union of all rectangles covered by the Pareto-optimal solutions constitutes the space totally covered, its size is used as measure. This concept may be canonically extended to multiple dimensions. An advantage of this measure is that each EA can be evaluated independent of the other EAs. On the other side, convex regions may be preferred to concave regions, possibly leading to overrating of certain solutions.[4] Therefore, we additionally compared the outcomes of the EAs directly by using the coverage relation (cf. Section 1). Given two sets of nondominated solutions, we computed for each set the fraction of the solutions which are covered by solutions in the other set.

Since in this study the focus is on finding the Pareto-optimal set rather than obtaining a uniform distribution over the trade-off surface, we did not consider the on-line performance of the EAs but the off-line performance. Thus, the Pareto-optimal set regarding all individuals generated over all generations is taken as output of an EA. In addition, to restrict the influence of random effects, the experiments were repeated ten times per test problem, always using a different randomly generated initial population (per experiment all EAs ran on the same initial population). The performance of a particular EA on a given test problem was calculated by averaging its performances over all ten experiments.

Actually, each multiobjective EA should be combined with the selection scheme originally applied. But the influence of the selection scheme on the outcome of an EA cannot be neglected, e.g., fitness proportionate selection, which is used in VEGA, is well-known to have serious disadvantages. In order to guarantee a fair comparison, all EAs considered were implemented with the same selection scheme, binary tournament selection.[5]. Unfortunately, a conventional combination of fitness sharing and tournament selection may lead to chaotic behavior of the EA, as reported by Oei, Goldberg and Chang [9]. Therefore, NSGA as well as Hajela's and Lin's approach were implemented using a slightly modified version of sharing, called *continuously updated sharing*, which was proposed by the same researchers. Thereby, not the current generation but rather the partly filled next generation is used to calculate the niche count. Horn and Nafpliotis

[4] In our opinion, this problem will probably always occur if the optimal Pareto front as well as the density of the search space are unknown.

[5] This selection method turned out to be superior to both stochastic remainder selection (used in [11]) and linear ranking selection on our test problems - that has been confirmed experimentally. Moreover, Srinivas and Deb themselves proposed to apply the combination of tournament selection and sharing to NSGA, which we used in this study.

no. of knap.	no. of items	algorithm				
		Random	Weighted	Niched	VEGA	NSGA
2	100	$1.2237 \cdot 10^7$	$1.3303 \cdot 10^7$	$1.4002 \cdot 10^7$	$1.4325 \cdot 10^7$	$\mathbf{1.4559 \cdot 10^7}$
	250	$6.2888 \cdot 10^7$	$6.7418 \cdot 10^7$	$7.0643 \cdot 10^7$	$7.1992 \cdot 10^7$	$\mathbf{7.3624 \cdot 10^7}$
	500	$2.4466 \cdot 10^8$	$2.5712 \cdot 10^8$	$2.6771 \cdot 10^8$	$2.7681 \cdot 10^8$	$\mathbf{2.7831 \cdot 10^8}$
3	100	$4.0641 \cdot 10^{10}$	$4.7860 \cdot 10^{10}$	$4.7068 \cdot 10^{10}$	$4.7965 \cdot 10^{10}$	$\mathbf{4.8997 \cdot 10^{10}}$
	250	$4.9232 \cdot 10^{10}$	$5.6527 \cdot 10^{11}$	$5.5859 \cdot 10^{11}$	$5.6530 \cdot 10^{11}$	$\mathbf{5.8229 \cdot 10^{11}}$
	500	$3.6504 \cdot 10^{11}$	$4.1189 \cdot 10^{13}$	$4.0743 \cdot 10^{13}$	$4.1127 \cdot 10^{13}$	$\mathbf{4.2111 \cdot 10^{13}}$
4	100	$1.0338 \cdot 10^{14}$	$1.1897 \cdot 10^{14}$	$1.2335 \cdot 10^{14}$	$1.1871 \cdot 10^{14}$	$\mathbf{1.2464 \cdot 10^{14}}$
	250	$3.4600 \cdot 10^{15}$	$4.0347 \cdot 10^{15}$	$4.0497 \cdot 10^{15}$	$4.0464 \cdot 10^{15}$	$\mathbf{4.2122 \cdot 10^{15}}$
	500	$5.0967 \cdot 10^{16}$	$5.9123 \cdot 10^{16}$	$5.8055 \cdot 10^{16}$	$5.8651 \cdot 10^{16}$	$\mathbf{5.9959 \cdot 10^{16}}$

Table 1. Results of the experiments concerning the size of the space covered by the nondominated solutions. The numbers set in bold face give the best value achieved per test problem.

introduced this concept in the Niched Pareto GA, too. Moreover, mating was not restricted.[6]

On all test problems the population size was set to 100, the probabilities of crossover and mutation were fixed (0.65 and 0.05[7], respectively), the crossover operator used was one-point crossover. Each single EA run was aborted after 500 generations.[8] The niche radii were determined experimentally, for each EA and test problem separately: First, and based on experiments, we sized the range of meaningful values for σ_{share}. Then, we chose five different niche radii and ran the EA on the test problem for each single niche radius. Afterwards, the niche radius which yielded the best result regarding the size of the space covered was selected. Finally, an analogous procedure was applied in order to find the appropriate values for t_{dom}, a parameter used by the Niched Pareto GA. On each test problem, we tried six different values $(1, 5, 10, 15, 20, 25)$ and chose the one providing the best results concerning the space covered. Thereby, the niche radii were determined in the aforementioned manner for each value separately.

4.2 Results

The results concerning the size of the space covered are shown in Table 1, where the column titled *Random* is related to the outcomes produced by a simple random search algorithm. This probabilistic algorithm, which serves as additional point of reference, randomly generates a certain number of individuals per generation, according to the rate of crossover and mutation (but neither crossover and mutation nor selection are performed). Hence, the number of fitness evaluations

[6] Hajela and Lin found it necessary to use mating restrictions in their evolutionary approach to multiobjective optimization. Therefore, all runs were also carried out with mating restrictions, where we tried several mating radii σ_{mat} and niche radii σ_{share} (following the common practice of setting $\sigma_{\text{mat}} = \sigma_{\text{share}}$). On all test problems no improvement of the results could be observed.

[7] Michalewicz and Arabas [8] used the same values in their study.

[8] It has been experimentally verified that no significant improvement has been achieved when increasing the number of generations. This has also been observed by Michalewicz and Arabas for the single-objective 0/1 knapsack problem.

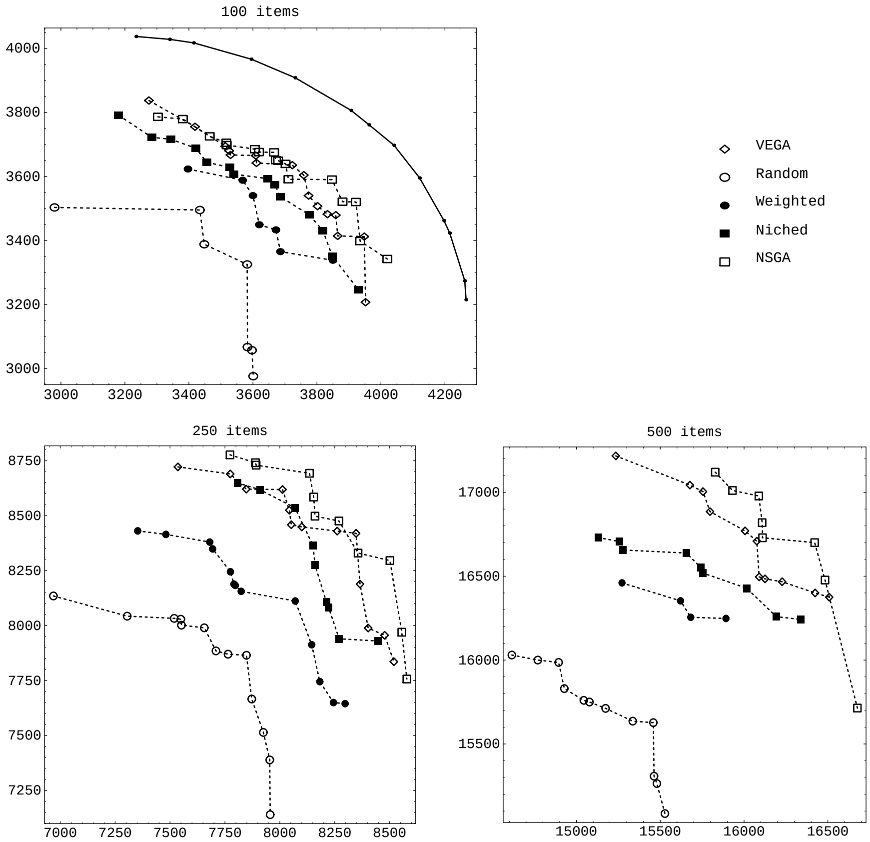

Fig. 1. Nondominated fronts for two objectives; for each method the nondominated solutions regarding all ten runs are plotted; for better visualization the points achieved by a particular method are connected by dashed lines. In the case of 100 items, the solid curve represents optimal nondominated solutions which have been calculated by integer linear programming using weighted-sum scalarization.

is the same as for the EAs. The output of the algorithm is the Pareto-optimal set of all solutions generated.

On all test problems NSGA outperformed the other approaches regarding this quality measure. Furthermore, all EAs achieved higher values than the pure random search strategy. The absolute values must not be overrated, however, for the two-dimensional case the ranking of performance is very well reflected by the Pareto fronts depicted in Figure 1.

The direct comparison of the multiobjective optimization methods is given in Table 2. Again, NSGA covers the greatest fraction of the Pareto sets achieved by the other algorithms. Regarding two objectives VEGA has second best performance in this comparison, similar to the results concerning the absolute size of the space covered. On the remaining test problems, VEGA and the weighted-sum

Coverage ($A \succeq B$)											
algorithm		test problem (no. of knapsacks / no. of items)									
A	B	2/100	2/250	2/500	3/100	3/250	3/500	4/100	4/250	4/500	mean
Random	Weighted	1,1%	0%	0%	0%	0%	0%	0%	0%	0%	**0.1%**
	Niched	0%	0%	0%	0%	0%	0%	0%	0%	0%	**0%**
	VEGA	0%	0%	0%	0%	0%	0%	0%	0%	0%	**0%**
	NSGA	0%	0%	0%	0%	0%	0%	0%	0%	0%	**0%**
Weighted	Random	98%	100%	100%	100%	100%	100%	99.3%	99.9%	100%	**99.7%**
	Niched	2.5%	1.5%	0%	72.7%	72.6%	75.7%	30.8%	49.5%	79.2%	**42.7%**
	VEGA	0%	0%	0%	41.4%	32.9%	30.6%	38%	30%	40.9%	**23.8%**
	NSGA	0%	0%	0%	23.2%	22%	14.1%	24.1%	11.7%	27%	**13.6%**
Niched	Random	100%	100%	100%	100%	99.6%	100%	99.6%	99.9%	100%	**99.9%**
	Weighted	92.5%	95%	100%	12.9%	20.1%	14.6%	40.8%	26.5%	4.5%	**45.2%**
	VEGA	0.9%	10.3%	0%	12.4%	14.2%	7.6%	47.8%	23.3%	8.6%	**13.9%**
	NSGA	0.7%	4.4%	2.2%	7.7%	5.6%	0.8%	27.1%	8.8%	3.6%	**6.8%**
VEGA	Random	100%	100%	100%	100%	100%	100%	99.4%	100%	100%	**99.9%**
	Weighted	100%	98.8%	100%	43.8%	54.6%	47.4%	34.3%	48.3%	34.7%	**62.4%**
	Niched	86.5%	87.9%	92%	73.2%	77.6%	80%	31.7%	59.7%	79.6%	**74.2%**
	NSGA	25.8%	16.9%	20.5%	20.8%	23.8%	16%	22.4%	16.9%	26%	**21%**
NSGA	Random	100%	100%	100%	100%	100%	100%	99.5%	100%	100%	**99.9%**
	Weighted	100%	100%	100%	59.7%	67.2%	72.7%	49.9%	72.8%	49.7%	**74.7%**
	Niched	93.8%	97.5%	98.8%	88%	89.7%	95.2%	51.1%	84.4%	91%	**87.7%**
	VEGA	58%	76.3%	67.4%	62.5%	58.7%	72.7%	60.5%	72.4%	58%	**65.2%**

Table 2. Direct Comparison of the outcomes achieved by the different multiobjective EAs; each cell gives the fraction of nondominated solutions evolved by method B, which are covered by the nondominated points achieved by method A in average; the last column comprises the mean values for each row.

approach show almost equal performance.

Comparing the Niched Pareto GA with the weighted-sum based approach, it can be observed that the former clearly outperformed the latter in the two-objective case, while the latter performed better in the three-objective case. Considering the problems with four objectives, neither can be said to be superior.

The bad performance of the Niched Pareto GA in the three-objective case may be explained by suboptimal parameter settings for t_{dom}. However, for these test problems t_{dom} was set to 10, which corresponds to guidelines given in [5] (10% of the population size); all other five t_{dom}-settings lead to worse results. As stated by Horn and Nafpliotis, the value of t_{dom} is critical to the convergence of the Niched Pareto GA, and to our experience, it seems to be rather difficult to find the optimal t_{dom}.

5 Conclusions and Future Work

In this study we compared four multiobjective EA on a multiobjective 0/1 knapsack problem with nine different problem settings. The quality of the Pareto-optimal sets achieved was measured quantitatively by the size of the space covered. Additionally, the approaches were compared directly by evaluating the outcomes regarding the concept of Pareto dominance.

All EAs clearly outperformed a pure random search strategy which randomly generates new points in the search space without exploiting similarities between solutions. Among the EAs the Nondominated Sorting Genetic Algorithm

(NSGA) proposed by Srinivas and Deb [11] achieved the best evaluations on all test problems, followed by Schaffer's VEGA [10] when all nine test problems are considered. Concerning the other two approaches, the results are ambiguous: In the two-objective case, the Niched Pareto Genetic Algorithm presented by Horn and Nafpliotis [5] outperformed the weighted-sum based approach proposed by Hajela and Lin [4]. On the other side, the latter EA achieved better evaluations for three objectives. In the case of four objectives neither of them was superior.

Regarding future perspectives, the issue of distributing the population over the tradeoff surface might be subject to further examinations. In many applications, where the tradeoff surface is continuous or containing a huge number of solutions, it is essential that the EA is capable of "selecting" representative solutions. Furthermore, the influence of mating restrictions might be investigated, although restricted mating is not very widespread in the field of multiobjective EA. Finally, as stated in [1] a theory of evolutionary multiobjective optimization is much needed, examining different fitness assignment methods in combination with different selections schemes.

References

1. Carlos M. Fonseca and Peter J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1):1–16, 1995.
2. D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Genetic Algorithms and their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, pages 41–49, Hillsdale, NJ, 1987. Lawrence Erlbaum.
3. David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, Massachusetts, 1989.
4. P. Hajela and C.-Y. Lin. Genetic search strategies in multicriterion optimal design. *Structural Optimization*, 4:99–107, 1992.
5. Jeffrey Horn and Nicholas Nafpliotis. Multiobjective optimization using the niched pareto genetic algorithm. IlliGAL Report 93005, Illinois Genetic Algorithms Laboratory, University of Illinois, Urbana, Champaign, July 1993.
6. Jeffrey Horn, Nicholas Nafpliotis, and David E. Goldberg. A niched pareto genetic algorithm for multiobjective optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Computation*, volume 1, pages 82–87, Piscataway, NJ, 1994. IEEE Service Center.
7. Silvano Martello and Paolo Toth. *Knapsack Problems: Algorithms and Computer Implementations*. Wiley, Chichester, 1990.
8. Zbigniew Michalewicz and Jaroslaw Arabas. Genetic algorithms for the 0/1 knapsack problem. In *Methodologies for Intelligent Systems (ISMIS'94)*, pages 134–143, Berlin, 1994. Springer.
9. Christopher K. Oei, David E. Goldberg, and Shau-Jin Chang. Tournament selection, niching, and the preservation of diversity. IlliGAL Report 91011, University of Illinois at Urbana-Champaign, Urbana, IL 61801, December 1991.
10. J. David Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In John J. Grefenstette, editor, *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, pages 93–100, 1985.
11. N. Srinivas and Kalyanmoy Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1994.

12. Manuel Valenzuela-Rendón and Eduardo Uresti-Charre. A non-generational genetic algorithm for multiobjective optimization. In *Proceedings of the Seventh International Conference on Genetic Algorithms*, pages 658–665, San Francisco, California, 1997. Morgan Kaufmann.

This article was processed using the LaTeX macro package with LLNCS style