A. Jaszkiewicz (to appear). On the computational effectiveness of multiple objective metaheuristics. *Proceeding of the Fourth International Conference on the Multi-Objective Programming and Goal Programming MOPGP'00. Theory & Applications*, Ustroń, Poland, May 29-June 1, 2000, Springer-Verlag, Berlin - Heidelberg.

On the computational effectiveness of multiple objective metaheuristics

Andrzej Jaszkiewicz

Institute of Computing Science
Poznań University of Technology
ul. Piotrowo 3a, 60-965 Poznań, Poland
Jaszkiewicz@cs.put.poznan.pl
www-idss.cs.put.poznan.pl/~jaszkiewicz

Abstract

The paper describes a technique for comparison of computational effectiveness of two approaches to generation of approximately Pareto-optimal solutions with the use of metaheuristics. In the on-line generation approach the approximately Pareto-optimal solutions are generated during the interactive process, e.g. by optimization of some scalarizing functions. In the off-line generation approach, the solutions are generated prior to the interactive process with the use of multiple objective metaheuristics. The results of experiment on travelling salesperson instances indicate that in the case of some multiple objective metheuristics the off-line generation approach may be computationally effective alternative to the on-line generation of approximately Pareto-optimal solutions.

Keywords Multiple objective optimization, metaheuristics, scalarizing functions, interactive methods, computational effectiveness

Introduction

In recent years, one could observe growing interest in multiple objective analysis of computationally hard problems, e.g. multiple objective combinatorial optimization (MOCO) problems. In the case of such problems, the use of exact methods that guarantee generation of exact Pareto-optimal solutions may be not possible because of computational requirements of the methods. As single objective metaheuristics proved to be successful on many hard optimization problems, it seems natural to apply them to generation of approximately Pareto-optimal solutions in multiple objective context.

A number of authors proposed multiple objective metaheuristic algorithms that aim at effective generations of samples of approximately Pareto-optimal solutions being approximations of the whole Pareto set. The methods are usually based on classical single objective metaheuristics. For example, the methods of Schaffer [21], Fonseca and Fleming [4], Horn, Nafpliotis and Goldberg [9], Srinivas and Deb [24] are based on genetic algorithms, the methods of Serafini [22], Czyzak and Jaszkiewicz [2], Ulungu et al. [29] are based on simulated annealing, and the methods of Gandibleux et. al. [6] and Hansen [8] are based on tabu search.

Hwang et al. [11] proposed a classification of MOO methods taking into account the moment of collecting the preference information with respect to the exploration process. They classify the MOO methods as either methods with a priori or a posteriori, or progressive (interactive) articulation of preferences. According to this classification the multiple objective metaheuristic algorithms should be treated as techniques used within a posteriori articulation of preferences approach. Note, however, that generation of a set of approximately Pareto-optimal solutions does not necessarily allows easy selection of the best compromise by the DM. The set of approximately Pareto-optimal solutions may contain a large number of solutions. In the case of two objectives, the objective trade-offs may be visualized in a two-dimensional plot allowing the DM to select the best compromise. No such simple visualization is possible in the case of three or more objectives. Thus, the DM analyzing a generated a priori large set of approximately Pareto-optimal solutions may need some further support characteristic to interactive procedures in the search for the best compromise.

Several interactive procedures for analysis of finite sets of alternatives have been already proposed. This class of methods includes: Zionts method [32], Korhonen, Wallenius and Zionts method [16], Köksalan, Karwan and Zionts method [14], Korhonen method [15], Malakooti method [18], Taner and Köksalan method [27], AIM [17], Light Beam Search-Discrete [13] and Interquad [26]. Such methods could be used for interactive analysis of large sets of approximately Pareto-optimal solutions. The methods are usually based on well-known interactive procedures for continuous case. In fact, the DM may not be even aware if the solutions presented to him/her in decision phases were generated a priori or if the solutions are generated on-line in computational phases alternating with phases of decision. Clearly, in both cases we deal with progressive articulation of preferences.

In results, we propose to distinguish two versions of approaches with progressive articulation of preferences, taking into account the way of generation of the (approximately) Pareto-optimal solutions. In the *on-line* approach the solutions are generated during the interactive process, i.e. generation of solutions alternates with articulation of DM's preferences. In contrary, in the *off-line* approach the (approximately) Pareto-optimal solutions are generated prior to interactive analysis.

On-line generation of (approximately) Pareto-optimal solutions is assumed in most classical interactive procedures proposed for continuous case (see e.g. reviews in [9] and [23]). The methods usually generate Pareto-optimal solutions by optimization of some substitute problems which global optima correspond to Pareto-optimal solutions. For example, a number of methods use, so called, scalarizing functions that are optimized on the original set of feasible solutions. In particular, optimization of weighted Tchebycheff scalarizing functions allows generation of all Pareto-optimal solutions ([25], ch. 14.8; [31]). In the case of hard MOO problems a natural approach consist in optimization of the scalarizing functions with classical single objective metaheuristics.

Note that most of the mentioned above interactive procedures for analysis of finite sets of alternatives also select the solutions presented to the DM applying a scalarizing function. In this case, however, the best solution on a scalarizing function is selected from the set of explicitly known solutions without optimization. In the rest of the paper, we will concentrate on on-line and off-line approaches that use scalarizing functions for selection of the approximately Pareto-optimal solutions presented to the DM.

Off-line generation of approximately Pareto-optimal solutions has several advantages with respect to the on-line approach:

- It allows different types of statistical analysis, e.g. calculation of correlation between objectives, and graphical visualization of the explicitly known set of approximately Pareto-optimal solutions that may increase the DM's knowledge about the problem.
- It assures very fast interaction with the DM, as no optimization is performed during the interactive process.
- It guarantees that all solutions presented to the DM are mutually non-dominated. In contrary, approximate solutions generated by a heuristic used within the on-line approach may dominate each other.

In this paper, we focus, however, on the issue of computational effectiveness of generation of approximately Pareto-optimal solutions. We propose a technique that allows comparing quality of solutions generated by the on-line and off-line approaches. Then, we propose to compare computational requirements of the two approaches needed to achieve the same quality of approximately Pareto-optimal solutions.

The paper is organized in the following way. The next section contains problem statement and basic definitions. The technique for comparison of computational effectiveness of the on-line and off-line approaches to generation of approximately Pareto-optimal solutions is described in details in the third section. In the fourth section, computational experiments on travelling salesperson instances are described. The conclusions are presented in the last section.

Problem statement and basic definitions

The general multiple objective optimization (MOO) problem is formulated as:

$$\max\left\{f_1(\mathbf{x}) = z_1, \ldots, f_J(\mathbf{x}) = z_J\right\} \qquad \text{(P1)}$$

$$\text{s.t.} \qquad \mathbf{x} \in D,$$

where *solution* $\mathbf{x} = \lfloor x_1, \ldots, x_I \rfloor$ is a vector of *decision variables*, D is the set of feasible solutions.

A solution $\mathbf{x} \in D$ is *Pareto-optimal* (*efficient*) if there is no $\mathbf{x}' \in D$ such that $\forall_j f_j(\mathbf{x}') \geq f_j(\mathbf{x})$ and $f_j(\mathbf{x}') > f_j(\mathbf{x})$ for at least one *j*. The set of all Pareto-optimal solutions is called *Pareto set*.

The point $\mathbf{z}^*$ composed of the best attainable objective function values is called the *ideal point*:

$$z_j^* = \max\left\{z_j \mid \mathbf{z} \in Z\right\} \qquad j = 1, \ldots, J.$$

Range equalization factors [25] are defined in the following way:

$$\pi_j = \frac{1}{R_j}, \ j=1, \ldots, J$$

A. Jaszkiewicz (to appear). On the computational effectiveness of multiple objective metaheuristics. *Proceeding of the Fourth International Conference on the Multi-Objective Programming and Goal Programming MOPGP'00. Theory & Applications*, Ustroń, Poland, May 29-June 1, 2000, Springer-Verlag, Berlin - Heidelberg.

---

where R_j is the (approximate) range of objective j in the set N or D. Objective function values multiplied by range equalization factors are called *normalized objective function values*.

Weighted Tchebycheff scalarizing functions are defined in the following way:

$$s_\infty\left(\mathbf{z},\mathbf{z}^{\mathrm{o}},\Lambda\right)=\max_j\left\{\lambda_i\left(z_j^{\mathrm{o}}-z_j\right)\right\}.$$

In the rest of the paper we will assume that $\mathbf{z}^{\mathrm{o}} = \mathbf{z}^*$. Each scalarizing function of this type has at least one global optimum (minimum) belonging to the set of Pareto-optimal solutions. For each Pareto-optimal solutions $\mathbf{x}$ there exists a weighted Tchebycheff scalarizing function s_∞ such that $\mathbf{x}$ is global optimum of s_∞ [25].

Augmented weighted Tchebycheff scalarizing functions are defined in the following way:

$$s_a\left(\mathbf{z},\mathbf{z}^*,\Lambda\right)=\max_j\left\{\lambda_i\left(z_j^*-z_j\right)\right\}+\varepsilon\sum_j\lambda_i\left(z_j^*-z_j\right),$$

where ε is a small number greater than zero.

Weight vectors than meet the following conditions:

$$\forall j\, \lambda_j \geq 0, \sum_{j=1}^{J}\lambda_j = 1,$$

are called normalized weight vectors.

Comparison of the computational effectiveness of on-line and off-line generation of approximately Pareto-optimal solutions

Consider a scalarizing function s used in a given iteration of an interactive procedure. In the case of both on-line and off-line approaches, the solution presented to the DM is not guaranteed to be optimal on s. Denote solution obtained within the on-line approach, i.e. by optimization of s with a single objective metaheuristic, by $\mathbf{x}^s$. Denote the solution obtained within the off-line approach, i.e. by selection from the set of a priori generated approximately Pareto-optimal solutions, by $\mathbf{x}^m$. The values $s(\mathbf{x}^s)$ and $s(\mathbf{x}^m)$ allow comparison of the two solutions. For example, solution $\mathbf{x}^s$ is better than $\mathbf{x}^m$ if $s(\mathbf{x}^s) < s(\mathbf{x}^m)$.

Of course, comparison on a single scalarizing function is meaningless. We propose to compare the two kinds of approaches on a set $S = \{s_1,...,s_L\}$ of randomly selected scalarizing functions. Let $\mathbf{x}^{s1},...,\mathbf{x}^{sL}$ be the best solutions obtained by optimization of $s_1,...,s_L$, respectively, with a single objective metaheuristic. Let $\mathbf{x}^{m1},...,\mathbf{x}^{mL}$ be the best solutions on $s_1,...,s_L$, respectively, selected from a set PE of potentially Pareto-optimal solutions generated by a multiple objective metaheuristic, i.e. $\forall \mathbf{x} \in PE\ s_l\left(\mathbf{x}^{ml}\right) \leq s_l\left(\mathbf{x}\right)$, $l = 1,...,L$. We consider the multiple objective metaheuristic not worse than the single objective metaheuristic if:

$$\sum_{l=1}^{L}\left(s_l\left(\mathbf{x}^{ml}\right)-s_l\left(\mathbf{x}^{sl}\right)\right)\leq 0\ . \qquad (*)$$

A. Jaszkiewicz (to appear). On the computational effectiveness of multiple objective metaheuristics. *Proceeding of the Fourth International Conference on the Multi-Objective Programming and Goal Programming MOPGP'00. Theory & Applications*, Ustroń, Poland, May 29-June 1, 2000, Springer-Verlag, Berlin - Heidelberg.

If $L \rightarrow \infty$ the above condition means that the set *PE* of potentially Pareto-optimal solutions gives the same average quality of approximation over all scalarizing functions that optimization of the functions with the single objective metaheuristic. In practice, we use, of course, finite values of *L*. The parameter plays the role of the size of statistical sample. The greater *L* the more significant the result given by (*).

If condition (*) is fulfilled, we may compare computational requirements of the single and multiple objective metaheuristics. Let CT_s be the average running time of the single objective method spent on optimization of $s_1,...,s_L$. Let CT_m be the running time of the multiple objective method needed to generate *PE*. We define then *effectiveness index EI*:

$$EI = \frac{CT_m}{CT_s} .$$

The lower *EI* the more effective the multiple objective metaheuristic with respect to the single objective method.

We propose to apply effectiveness index to comparison of single and multiple objective metaheuristics based on similar ideas. For example, multiple objective genetic algorithms could be compared to single objective GAs, multiple objective genetic local search could be compared to single objective genetic local search, etc. As most multiple objective metaheuristics are some extensions/modifications of single objective methods, the effectiveness index gives some information about quality of this extension. Of course, in this case, it is natural to expect $EI > 1$.

The effectiveness index may be used to compare different multiple objective metaheuristics based on the same single objective method.

The effectiveness index has some clear interpretation from the point of view of interactive procedures. Assume that the interactive process requires generation of *R* approximately Pareto-optimal solutions of a given problem in order meet the stopping criteria or to reach the solution satisfying the DM. If $R > L$ then the overall computational requirements of the on-line generation of approximately Pareto-optimal solutions are higher than computational requirements of the off-line approach.

Computational experiment

Overview of the experiment

This section describes computational experiment performed on multiple objective travelling sales person instances. The pairs of methods compared are genetic algorithm vs. Pareto ranking based multiple objective genetic algorithm, and genetic local search vs. multiple objective genetic local search.

We use augmented Tchebycheff scalarizing functions with $\varepsilon = 0.1$. Parameter *L* - the number of scalarizing functions on which the pairs of methods are compared, is equal to 50. The scalarizing functions are defined by *L* normalized weight vector randomly generated with the algorithm presented in Figure 1. The algorithm assures that weight vectors are drawn with uniform probability distribution $p(\Lambda)$, i.e. a distribution for which:

$$\forall \Psi' \subseteq \Psi \quad \int_{\Lambda \in \Psi'} p(\Lambda) d\Lambda \Big/ \int_{\Lambda \in \Psi} p(\Lambda) d\Lambda = V(\Psi') / V(\Psi)$$

where Ψ and Ψ' denote the set of all normalized weights and a subset of it, respectively; $V(\Psi)$ and $V(\Psi')$ are Euclidean hyper-volumes of Ψ and Ψ', respectively. In other words, the probability of drawing a weight vector belonging to Ψ' is proportional to the hyper-volume of Ψ'.

$$\lambda_1 = 1 - \sqrt[J-1]{rand()}$$

...

$$\lambda_j = \left(1 - \sum_{l=1}^{j-1} \lambda_l\right)\left(1 - \sqrt[J-1-j]{rand()}\right)$$

...

$$\lambda_J = 1 - \sum_{l=1}^{J-1} \lambda_l$$

Figure 1. Algorithm of generation of random weight vectors.

For each instance the following experiment was performed. In the first phase 50 random scalarizing functions were optimized with a single objective method. Then, the multiple objective method was started. After specified number of iterations of the multiple objective method the quality of the current set of potentially Pareto-optimal solutions was compared to the quality of solutions generated by the single objective method with condition (*). When condition (*) was met the multiple objective method was stopped and the effectiveness index was calculated. Another stopping criterion of the multiple objective method was running time greater than 500 times the average running time of the single objective method. In the second case, we can state that the effectiveness index is ≥ 500.

Methods used in the experiment

Single objective genetic local search

Genetic local search (GLS) is a method that hybridizes recombination operators with local search. The version of GLS algorithm we use assumes complete elitism, i.e. the current population is always composed of a sample of best known solutions. The details of this algorithm are in Figure 2.

```
Parameters: $K$ – size of the current population, stopping criterion
Initialization:
Current population $P := \emptyset$
repeat $K$ times
    Construct randomly a new feasible solution $\mathbf{x}$
    Optimize locally the objective function starting from solution $\mathbf{x}$ obtaining $\mathbf{x}'$
    Add $\mathbf{x}'$ to $P$.
Main loop:
repeat
    Draw at random with uniform probability two solutions $\mathbf{x}_1$ and $\mathbf{x}_2$ from $P$.
    Recombine $\mathbf{x}_1$ and $\mathbf{x}_2$ obtaining $\mathbf{x}_3$
    Optimize locally the objective function starting from solution $\mathbf{x}_3$ obtaining $\mathbf{x}_3'$
    if $\mathbf{x}_3'$ is better than the worst solution in $P$ and different in the decision space to all solutions in $P$ then
            Add $\mathbf{x}_3'$ to $P$ and delete from $P$ the worst solution
until the stopping criterion is met
```

Figure 2. Algorithm of the basic single objective genetic local search

In the experiments described in this paper, the optimization was stopped if in 20 successive iterations current population was not changed. This value was selected experimentally. It was observed that population that was not changed in 20 iterations gives little chance for further improvements.

Size of the current population K is the main parameter controlling the calculation time. Generally, the larger K the larger CPU time and the better quality of results.

In this algorithm, mutation operator is not explicitly used. The recombination operator may introduce, however, some elements of randomness. In other cases explicit mutation operators may be necessary.

Multiple objective genetic local search

One of the multiple objective metaheuristics used in the experiment is multiple objective genetic local search (MOGLS) proposed in [12] on the basis of the single objective algorithm presented in Figure 2. In each iteration, the method draws at random a scalarizing function s for optimization. Then, two of previously generated solutions being good on s are recombined and local search is applied to their offspring. We use augmented weighted Tchebycheff scalarizing functions with $\varepsilon = 0.1$ in this experiment.

```
    Parameters: $K$ – size of the temporary population, stopping criterion
    Initialization:
    The set of potentially Pareto-optimal solutions $PE:=\emptyset$
    The current set of solutions $CS:=\emptyset$
    repeat until $CS$ meets stopping condition for generation of initial solutions
        Draw at random a scalarizing function $s$
        Construct randomly a new feasible solution $\mathbf{x}$
        Optimize locally the scalarizing function $s$ starting from solution $\mathbf{x}$ obtaining $\mathbf{x}'$
        Add $\mathbf{x}'$ to the current set of solutions $CS$
        Update set $PE$ with $\mathbf{x}'$
    Main loop:
    repeat
        Draw at random a scalarizing function $s$
        From $CS$ select $K$ different solutions being the best on scalarizing function $s$ forming temporary population $TP$
        Draw at random with uniform probability two solutions $\mathbf{x}_1$ and $\mathbf{x}_2$ from $TP$.
        Recombine $\mathbf{x}_1$ and $\mathbf{x}_2$ obtaining $\mathbf{x}_3$
        Optimize locally the scalarizing function $s$ starting from solution $\mathbf{x}_3$ obtaining $\mathbf{x}_3'$
        if $\mathbf{x}_3'$ is better than the worst solution in $TP$ and different in the decision space to all solutions in $TP$ then
                Add $\mathbf{x}_3'$ to the current set of solutions $CS$
                Add $\mathbf{x}_3'$ to $TP$ and delete from $TP$ the worst solution
        Update set $PE$ with $\mathbf{x}_3'$
until the stopping criterion is met
```

Figure 3. Algorithm of the multiple objective genetic local search

The generation of initial solutions is stopped when the average quality of *K* best solutions in *CS* over all scalarizing functions is the same as average quality of local optima of this function. Consider **x** being an initial solution obtained by local optimization of function scalarizing s_x. Note that **x** need not be the best solutions on s_x in current set of solutions *CS*. Let $B(K,CS,\mathbf{x},s_x)\subseteq CS$ be the set of *K* best solutions of function s_x different than **x**. Let $\bar{s}(B(K,CS,\mathbf{x},s_x))$ be the average value of s_x in $B(K,CS,\mathbf{x},s_x)$. We stop generation of the initial solutions when the following condition is met:

$$\sum_{x\in S}\left(\bar{s}\left(B\left(K,CS,\mathbf{x},s_x\right)\right)-s_x(\mathbf{x})\right)\geq 0$$

Of course, the above condition could only be tested when the number of solutions in *CS* is greater or equal to $K+1$.

Set *CS* is organized as a queue of size $K\times S$, where *S* is the number of initial solutions. In each iteration, the newly generated solution is added to the beginning of the queue if it is better than the worst solution in the temporary population and different to all solutions in the temporary population. If the size of the queue is greater than $K \times S$ the last solution from the queue is removed.

The stopping criterion of the main loop is defined by maximum number of iterations.

Updating the set of potentially Pareto-optimal solutions *PE* with solution **x** consists of:

- adding **x** to *PE* if no solution in *PE* dominates **x**,
- removing from *PE* all solutions dominated by **x**.

Note that the set of potentially Pareto-optimal solutions is updated with local optima only. In general, other solutions generated during the local search may also be potentially Pareto-optimal. This approach allows, however, for significant reduction of running time. Furthermore, a data structure called quad tree allows for very effective implementation of this step [3], [7].

The random scalarizing functions are defined by random weight vectors constructed with the algorithm presented in Figure 1. The weights are applied to normalized objective function values.

Genetic algorithm

In our experiment, we used genetic algorithm presented in Figure 4.

```
Parameters: $K$ – size of the genetic population, maximum number of generations, mutation probability
Initialization:
Generate $K$ random solutions forming the initial population
Main loop:
repeat
    Select $K$ pairs of solutions with roulette wheel selection
    Recombine each pair of solutions
    Mutate each of the offspring with mutation probability
    Replace previous population with the new one
until the maximum number of generations is reached
```

Figure 4. Genetic algorihm.

Pareto ranking based multiple objective genetic algorithm

In the experiment we used a Pareto ranking based multiple objective genetic algorithm (Pareto MOGA) proposed in [4] without mating restrictions. In addition, we maintain the set of potentially Pareto-optimal solutions updated with each newly constructed solution. The details are given in Figure 5.

```
Parameters: K – size of the genetic population, maximum number of generations, mutation probability
Initialization:
The set of potentially Pareto-optimal solutions PE:=∅
Generate K random solutions forming the initial population
Update set PE with each solution from the current population
Main loop:
repeat
    Assign to each solution from the current population the fitness on the basis of Pareto ranking
    Reduce fitness of close solutions by fitness sharing
    Select K pairs of solutions with roulette wheel selection
    Recombine each pair of solutions
    Mutate each of the offspring with mutation probability
    Replace previous population with the new one
    Update set PE with each solution from the current population
until the maximum number of generations is reached
```

Figure 5. Pareto-ranking based multiple objective genetic algorithm.

Multiple objective symmetric travelling salesperson problem

Single objective TSP is often used to test single objective metaheuristics. It is defined by a set of cities and cost (distance) of travel between each pair of cities. In symmetric TSP the cost does not depend on direction of travel between two cities. The goal is to find the lowest cost hamiltonian cycle.

In *J*-objective TSP, *J* different cost factors are defined between each pair of cities. In practical applications, the cost factors may for example corresponds to cost, length, travel time or tourist attractiveness. In our case, *J*-objective symmetric TSPs instances are constructed from *J* different single objective TSP instances. Thus, *j*-th cost factor, $j=1,...,J$, between a pair of cities comes from *j*-th single objective problem. Individual optima of particular objectives are equal to optima of corresponding single objective problems. In our case, the single objective problems are completely independent, so, also objectives are independent and therefore non-correlated. The same approach was used by [1] and [12].

Also following [1] and [12] we use multiple objective problem instances based on TSPLIB library [20]. For example, problem instance kroABC100 denotes a three-objective problem with cost factors corresponding to the first objective taken from kroA100, cost factors corresponding to the second objective taken from kroB100, and cost factors corresponding to the second objective taken from kroC100. In this way 10 different three-objective problem instance were created. We used also 10 three-objective instances and 5 four-objective instances with 50 leading cities taken from kroA100-kroE100 instances.

The recombination operator used by all the methods is the distance-preserving crossover introduced in [5]. An offspring is constructed in the following steps:

Step 1. Put in the offspring all arcs common to both parents

A. Jaszkiewicz (to appear). On the computational effectiveness of multiple objective metaheuristics. *Proceeding of the Fourth International Conference on the Multi-Objective Programming and Goal Programming MOPGP'00. Theory & Applications*, Ustroń, Poland, May 29-June 1, 2000, Springer-Verlag, Berlin - Heidelberg.

Step 2. Complete the hamiltonian cycle with randomly selected arcs.

Table 1. Results of GLS – MOGLS comparison

(Temporary) population size		Average running time of GLS (number of functions' evaluations)	Running time of MOGLS (number of functions' evaluations)	Effectiveness index	Number of potentially Pareto-optimal solutions generated by MOGLS
3 objectives, 50 cities					
10	Average	614 309	9 676 310	**15.69**	1003
	Standard dev.	32 877	3 043 922	**4.51**	235
20	Average	1 186 259	18 164 128	**15.12**	1573
	Standard dev.	57 406	13 115 967	**10.38**	406
30	Average	1 676 203	19 408 022	**11.64**	1790
	Standard dev.	106 760	2 314 632	**1.70**	219
3 objectives, 100 cities					
10	Average	4 912 063	187 879 890	**38.63**	2726
	Standard dev.	238 474	91 281 659	**19.42**	921
20	Average	10 111 330	357 766 821	**35.84**	4837
	Standard dev.	514 896	190 908 770	**20.29**	1886
30	Average	14 837 900	557 799 727	**37.89**	6547
	Standard dev.	790 362	165 688 406	**12.26**	1800
4 objectives, 50 cities					
10	Average	680 775	57 336 646	**84.55**	8785
	Standard dev.	25 605	11 756 380	**19.16**	2038
20	Average	1 183 152	96 166 142	**81.05**	14267
	Standard dev.	42 735	19 870 387	**15.16**	3199
30	Average	1 750 500	116 369 545	**66.44**	16602
	Standard dev.	78 749	12 193 400	**5.89**	1804

The local search used is based on standard 2-arcs exchange neighborhood. While constructing the initial population, greedy local search is used. After recombination, steepest local search is used. This combination was found to give the best results. As local search consumes most of the CPU time, we measure the running time in the number of functions' evaluations (number of evaluated solutions). In the case of 2-arcs exchange operator, neighbor solutions can be evaluated in very short time. In results about 1 000 000 functions' evaluations can be performed on 350 MHz Pentium PC in one second.

The mutation operator exchanges to randomly selected arcs.

Results

Table 1 contains results of the experiments with genetic local search and multiple objective genetic local search. The entries in the table contain averages and standard deviations of ten experiments on three objective instances and five experiments on four objective instances (one experiment for each instance). As could be expected the effectiveness index grows with the problem size and the number of objectives. In both cases, the growth of the effectiveness index is correlated to the growth of the size of the set of potentially Pareto-optimal solutions.

In our opinion, the results prove that generation of approximately Pareto-optimal solutions with the MOGLS is competitive from the computational effectiveness point of view to generation of the solutions with single objective GLS. Note that GLS is one of the best methods for single objective TSP [19].

Significantly different results were obtained in the case of comparison of GA and Pareto MOGA. In none of the experiments the set of potentially Pareto-optimal solutions generated by Pareto MOGA fulfilled the condition (*) in running time lower or equal to 500 times the average running time of GA. Thus, we conclude that the effectiveness index of Pareto MOGA in comparison to GA on the TSP instances used in the experiment is greater than 500.

Conclusions

We have introduced a measure call effectiveness index that relates computational requirements of single and multiple objective metaheuristic necessary to generate approximately Pareto-optimal solutions of the same average quality.

The results of experiments on the TSP instances indicate that off-line generation of approximately efficient solutions may be computationally competitive approach. In particular multiple objective genetic local search is able to generate high quality approximations to the whole Pareto-optimal set in running time 11.64 - 84.55 longer than the average time needed to generate single approximately Pareto-optimal solution with the same average quality using the single objective genetic local search. Note that GLS is known to be very effective method for single objective TSP. The results indicate also that the relative effectiveness of MOGLS decreases with the growth of the number of objectives.

In the experiments with genetic algorithm and Pareto ranking based multiple objective genetic algorithm the effectiveness index was found to be greater than 500. We conclude that from the computational point of view Pareto MOGA is not competitive tool for generation of approximately Pareto-optimal solution in comparison to GA on TSP instances.

Note that off-line generation of approximately Pareto-optimal solutions has a number of additional advantages over on-line approach (see introduction section).

Thus, computational effectiveness is not the only reason for the use of this approach.

The two approaches for generation of approximately Pareto-optimal solutions, are in fact extreme possibilities. In the off-line approach we assume that a single run of a metaheuristic generates a sample of approximately Pareto-optimal solutions approximating the whole Pareto set. In the online, approach only a single Pareto-optimal solution by each run of a metaheuristic. It is also possible to generate in a single run of a metaheuristic a sample of approximately Pareto-optimal solutions approximating a promising subregion of the whole Pareto set. In fact, many multiple objective metaheuristics can be easily used in this way. For example in MOGLS algorithm, it is possible to draw at random scalarizing functions from a subset of all scalarizing functions. In this case, multiple objective metaheuristics may be computationally effective even in the case of many objectives. An approach of this kind is proposed e.g. in [4].

Bibliography

[1] Borges P.C., Hansen P.H. (1998), A basis for future successes in multiobjective combinatorial optimization. *Technical Report, Department of Mathematical Modelling, Technical University of Denmark*, IMM-REP-1998-8.

[2] Czyzak P., Jaszkiewicz A. (1998), Pareto simulated annealing - a metaheuristic technique for multiple-objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, 7, 34-47.

[3] Finkel R.A. and Bentley J.L. (1974), Quad Trees, A data structure for retrieval on composite keys. *Acta Informatica*, **4**, 1-9.

[4] Fonseca C.M., Fleming P.J. (1993), Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In S. Forrest (Ed.), *Genetic Algorithms: Proceedings of 5th International Conference*, San Mateo, CA, Morgan Kaufmann, 416-423.

[5] Freisleben B., Merz P. (1996), A genetic local search algorithm for travelling salesman problem. In H.-M. Voigt, W. Ebeling, I. Rechenberg, H.-P. Schwefel (eds.), *Proceedings of the 4th Conference on Parallel Problem Solving fram Nature- PPSN IV*, 890-900.

[6] Gandibleux, X., Mezdaoui N., Fréville A. (1996). A tabu search procedure to solve multiobjective combinatorial optimization problems, In R. Caballero, R. Steuer (Eds.), *Proceedings volume of MOPGP '96*, , Springer-Verlag.

[7] Habenicht W. (1982), Quad Trees, A datastructure for discrete vector optimization problems. *Lecture Notes in Economics and Mathematical Systems*, **209**, 136-145.

[8] Hansen M. (1997), Tabu search for multiobjective optimization: MOTS, presented at the 13th MCDM conference, Cape Town, South Africa, January 6-10.

[9] Horn. J., Nafpliotis N. (1994). A niched Pareto genetic algorithm for multiobjective optimization. *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, vol. 1, IEEE, New York, 82-87.

[10] Hwang C.-L. and Masud A.S.M. (1979). *Mutiple Objective Decision Making - Methods and Applications*, Springer, Berlin.

[11] Hwang C.-L., Paidy S.R., Yoon K. and Masud A.S.M. (1980), Mathematical programming with multiple objectives: A tutorial. *Comput. Oper. Res.*, **7**, 5-31.

[12] Jaszkiewicz A. (1998). Genetic local search for multiple objective combinatorial optimization. Research report, Institute of Computing Science, Poznań University of Technology, **RA-014/98**, pp.23.

[13] Jaszkiewicz A., Słowiński R. (1997). The LBS-Discrete Interactive Procedure for Multiple-Criteria Analysis of Decision Problems. In: J. Climaco (red.) *Multicriteria Analysis. Proceedings of the XIth International Conference on MCDM*, 1-6, August 1994, Coimbra, Portugal, Springer-Verlag, Berlin - Heidelberg, 320-330.

[14] Köksalan M., Karwan M.H. and Zionts S. (1988). An Approach for Solving Discrete Alternative Multiple Criteria Problems Involving Ordinal Criteria. *Naval Research Logistics*, **35**, 6, 625-642.

[15] Korhonen P. (1988). A Visual Reference Direction Approach to Solving Discrete Multiple Criteria Problems. *EJOR*, **34**, 2, 152-159.

[16] Korhonen P. Wallenius J. and Zionts S. (1984). Solving the Discrte Multiple Criteria Problem Using Convex Cones. *Management Science*, **30**, 11, 1336-1345.

[17] Lotfi V., Stewart T.J. and Zionts S. (1992). An aspiration-level interactive model for multiple criteria decision making. *Comput. Ops. Res.*, **19**, 677-681.

[18] Malakooti B. (1989). Theories and an Exact Interactive Paired-COmparison Approach for Discrete Multiple Criteria Problems. IEEE Transactions on Systems, Man, and Cybernetics, **19**, 2, 365-378.

[19] Merz P., Freisleben B., Genetic Local Search for the TSP: New Results, In *Proceedings of the 1997 IEEE International Conference on Evolutionary Computation*, IEEE Press, 159-164, 1997.

[20] Reinelt G. (1991). TSPLIB – a traveling salesman problem library. *ORSA Journal of Computing*, **3**, 4, 376-384.

[21] Schaffer J.D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. In: J.J. Grefenstette (ed.), *Genetic Algorithms and Their Applications: Proceedings of the Third International Conference on Genetic Algorithms*, Lawrence Erlbaum, Hillsdale, NJ, 93-100.

[22] Serafini P. (1994). Simulated annealing for multiple objective optimization problems. In: Tzeng G.H., Wang H.F., Wen V.P., Yu P.L. (eds), *Multiple Criteria Decision Making. Expand and Enrich the Domains of Thinking and Application*, Springer Verlag, 283-292.

[23] Shin W.S. and Ravindran A. (1991). Interactive multiple objective optimization: survey I - continuous case, *Comput. Oper. Res.*, **18**, 97-114.

[24] Srinivas N., Deb K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2, 2, 221-248.

[25] Steuer R.E. (1986). *Multiple Criteria Optimization - Theory, Computation and Application*, Wiley, New York.

[26] Sun M., Steuer R. E. (1996). InterQuad: An Interactive Quad Tree Based Procedure for Solving the Discrete Alternative Multiple Criteria Problem. *European Journal of Operational Research*, **89**, No. 3, 462-472.

[27] Taner O.V. and Köksalan M.M. (1991). Experiments and an Improved Method for Solving the Discrete Alternative Multiple-Criteria Problem. *Journal of the Operational Research Society*, **42**, 5, 383-392.

[28] Ulungu E.L. and Teghem J. (1994). Multiobjective Combinatorial Optimization Problems: A Survey. *Journal of Multi-Criteria Decision Analysis*, **3**, 83-101.

[29] Ulungu E.L., Teghem J., Fortemps Ph., Tuyttens (1999). MOSA method: a toll for solving multiobjective combinatorial optimization problems. *Journal of Multi-Criteria Decision Analysis*, 8, 221-236.

[30] Wierzbicki A.P. (1980), The use of reference objective in Multiobjective Optimization. In: Fandel G. and Gal T. (eds.) *Multiple Criteria Decision Making, Theory and Application*, Springer-Verlag, Berlin, 468-486.

[31] Wierzbicki A.P. (1986), On the completeness and constructiveness of parametric characterization to vector optimization problems. *OR Spektrum*, **8**, 73-87.

[32] Zionts S. (1981). A Multiple Criteria Method for Choosing among Discrete Alternatives. *EJOR*, **7**, 1, 143-147