ANDRZEJ JASZKIEWICZ

MULTIPLE OBJECTIVE METAHEURISTIC ALGORITHMS FOR COMBINATORIAL OPTIMIZATION

POZNAŃ 2001

Recenzent
prof. dr hab. PIOTR JĘDRZEJOWICZ

Redakcja techniczna
RENATA LUBAWY

Skład i łamanie
ANDRZEJ JASZKIEWICZ

Projekt okładki
PIOTR GOŁĘBNIAK

ISSN 0551-6528
ISBN 83-7143-498-7

WYDAWNICTWO POLITECHNIKI POZNAŃSKIEJ
60-965 Poznań, pl. M. Skłodowskiej-Curie 2, tel. (061) 6653516
e-mail: office_ed@put.poznan.pl

Wydanie I

Druk w Zakładzie Graficznym Politechniki Poznańskiej

Contents

Summary

Multiple objective combinatorial optimization (MOCO) problems arise in many areas, e.g. in production scheduling, project scheduling, staff scheduling, time tabling, production facilities design, vehicle routing, telecommunication routing, investment planning and location. The number of successful real-life applications of MOCO reported in the literature is, however, relatively low in comparison with other areas of multiple objective optimization or single objective combinatorial optimization. Clearly, MOCO problems constitute one of the most difficult fields of optimization.

Following the success of metaheuristics in single objective optimization many researchers proposed the use of metaheuristics in multiple objective case. Since the proposition of the Vector Evaluated Genetic Algorithm in 1985 a significant number of different multiple objective metaheuristics have been proposed. The goal of the methods is to generate a set of approximately Pareto-optimal solutions being an approximation of the whole or a part of the nondominated set. At present, multiple objective metaheuristics constitute one of the most active fields of research in multiple objective optimization. The use of multiple objective metaheuristics is one of the most promising approaches to MOCO.

The goal of this work is to advance the methodology of multiple objective metaheuristics for MOCO problems. The work contains a review of the methodology of multiple objective metaheuristics. The review focuses on both characterization of the main algorithms as well as the issue of evaluation of multiple objective metaheuristics. The quality measures proposed by other authors are usually based on cardinal or geometrical arguments. In this work a new approach that evaluates approximations of the nondominated set on the basis of some assumptions about the decision maker's preferences is presented. Then, two new multiple objective metaheuristics developed by the author are presented. They are based on two well-known single objective methods – simulated annealing and genetic local search. This work describes also a two-phase approach to interactive analysis of MOCO problems. In the first phase, a multiple objective metaheuristic is applied to generating the set of solutions approximating the whole nondominated set. In the second phase, the set of solutions is analyzed with an interactive method for discrete alternative multiple objective choice problems. Two new interactive methods for discrete choice problems proposed by the author are presented. The presented methodology has been extended to handle fuzzy MOCO problems. The proposed methods have been tested in a number of comparative experiments. The results indicate that the methods are competitive with respect to other state of the art multiple objective metaheuristics. Exemplary applications of the presented methodology in nurse scheduling, multiple objective software project scheduling and logistic distribution system design are also presented. Finally, further research directions are outlined.

1. Introduction

1.1. Metaheuristics in multiple objective combinatorial optimization

Optimization is one of the most important fields of computers' applications. Combinatorial optimization deals with discrete problems. It finds applications in many areas, e.g. in production scheduling, project scheduling, staff scheduling, time tabling, production facilities design, vehicle routing, telecommunication routing, investment planning, location and many others [183].

Many real-life optimization problems require taking into account multiple points of view corresponding to multiple objectives. The goal of multiple objective optimization (MOO) is to find the single solution giving the *best compromise* between multiple objectives. As usually there is no single solution that optimizes simultaneously all the objectives, selection of the best compromise solution requires taking into account preferences of the DM. Under very weak and generally accepted assumptions about the DM's preferences the best compromise solution belongs to the set of, so called, *Pareto-optimal solutions* ([154] , ch. 6.6-6.7).

Research activities within the area of multiple objective optimization traditionally concentrated on multiple objective linear programming (MOLP) (see e.g. [154], [181]). In recent years, the demand for new applications and the increasing power of computers resulted in growing interest in computationally hard MOO problems, e.g. non-linear and combinatorial optimization problems. This work focuses on multiple objective combinatorial optimization (MOCO).

The number of real-life applications of MOCO reported in the literature is relatively low [33], [70]. In our opinion, this is not due to the fact that combinatorial problems rarely require multiple objectives. Indeed, different objectives are often used in particular classes of combinatorial problems. For example, in vehicle routing the typically used objectives are total cost, distance, number of vehicles, travel time [2]; and in project scheduling the typical objectives are net present value, project completion time, mean weighted delay, number of delayed tasks, mean weighted flow time, total resource utilization [151]. The objectives, however, are usually used separately or they are combined into a single objective. In our opinion, the relatively low number of real-life applications of MOCO is a consequence of the fact that MOCO problems constitute one of the most difficult classes of MOO problems.

Methods proposed for solving MOLP problems usually assume generation of Pareto-optimal solutions with the use of single objective optimization. Pareto-optimal solutions are obtained by solving some substitute problems whose global optima correspond to Pareto-optimal solutions. Typical example is optimization of a scalarizing function on the feasible set of the original MOO problem [180]. Techniques of this kind often cannot be directly applied to MOCO problems because of their complexity. In the case of

MOCO, computational requirements of algorithms used for generation of Pareto-optimal solutions become of crucial importance. It is well known that many single objective combinatorial optimization problems are NP-complete, i.e. they cannot be solved with polynomial time algorithms unless P=NP. NP-completeness was originally formulated for single objective optimization. In [145] a way of extending this definition for multiple objective case was proposed. It turns out that generation of Pareto-optimal solutions in the case of many MOCO problems requires exponential time algorithms, even if optimization of particular objectives is easy. In particular, testing if a given solution is Pareto-optimal is NP-hard even in the case of the unconstrained 0-1 programming problem. Thus, the use of heuristic methods seems to be the most practical approach to MOCO.

Heuristic algorithms can be classified as either specialized or general purpose, i.e. metaheuristics. Specialized heuristics are often able to generate good solutions in very short time. In the case of MOCO, specialized heuristics can be developed either for the substitute single objective problems (see e.g. [99]), or for the problems of generation of a set of approximately Pareto-optimal solutions (see e.g. [135]). Specialized heuristics have the known disadvantage of being restricted to some specific problems. In multiple objective case, this disadvantage becomes even more important. Specialized heuristics are usually restricted to a given set of objectives. Inclusion or removal of a single objective usually makes the methods not applicable. Furthermore, different kinds of substitute problems used for generation of approximately Pareto-optimal solutions may require different methods. For example, optimization of linear, Tchebycheff and achievement scalarizing functions may require different specialized heuristics. In general, specialized heuristics are restricted not only to specific MOCO problems, but also restricted to specific methods for generation of approximately Pareto-optimal solutions. Most of the specialized heuristics for MOCO problems proposed in the literature are developed for problems with all objectives having the same mathematical definition. Real-life problems however, often involve objectives with different mathematical definitions, e.g. sum, min-max, max-min, quadratic, etc.

Following the success of metaheuristics in single objective optimization (see e.g. [127]) many researchers proposed the use of metaheuristics in MOO. Since the proposition of the Vector Evaluated Genetic Algorithm in 1985 [144] a significant number of different multiple objective metaheuristics have been proposed. We use the term "multiple objective metaheuristic" to characterize methods that generate a set of approximately Pareto-optimal solutions in a single run. The set may be treated as an approximation of the whole or a part of the nondominated set. At present, multiple objective metaheuristics constitute one of the most active fields of research in MOO [33], [70].

Multiple objective metaheuristics are applied to both continuous nonlinear MOO problems and to MOCO problems. Presently, most of the successful real-life applications concern continuous nonlinear MOO problems with relatively low number of variables. Such problems are, for example, common in engineering design [28]. The main source of difficulty of such problems is not their size but their nonlinearity. MOCO problems are apparently more difficult for multiple objective metaheuristics. Although some authors claim that their methods can be applied to any kind of problems, in our opinion MOCO problems require specific approaches. Experiments reported in this work indicate that some well-known algorithms used in continuous case perform poorly on MOCO problems.

In our opinion, the demand for various real-life applications that can be modeled as hard MOCO problems requires new effective and general methods. This work is devoted to the methodology of multiple objective metaheuristics for MOCO problems.

1.2. The goal and the scope of this work

The main goal of this work is to advance the methodology of multiple objective metaheuristics for MOCO problems. The specific goals of this work may be formulated as follows:

- to propose a methodology of evaluation of multiple objective metaheuristics,
- to develop new effective multiple objective algorithms for MOCO problems based on simulated annealing and genetic local search,
- to propose a methodology of interactive analysis of MOCO problems applying multiple objective metaheuristics to generation of approximately Pareto-optimal solutions,
- to develop computer implementations of the proposed methods,
- to perform comparative experiments with the proposed methods,
- to apply the proposed methods to real-life MOCO problems.

Section 2 contains basic definitions related to multiple objective combinatorial optimization.

In section 3, a review of methodology of multiple objective metaheuristics is presented. The review focuses on methods that generate a set of approximately Pareto-optimal solutions in a single run, and on applications of metaheuristics in MOCO. The presentation concentrates on essential elements of the existing algorithms. This section contains also a review of the quality measures proposed for the evaluation of approximations of the nondominated set.

Section 4 concerns the evaluation of sets of solutions approximating the whole or a part of the nondominated set. The quality measures proposed by other authors are usually based on cardinal or geometrical arguments. The approach proposed in this section evaluates approximations of the nondominated set on the basis of some assumptions about the decision maker's (DM's) preferences.

Section 5 describes Pareto simulated annealing algorithm proposed by Czyżak and the author. The method uses a population of so called generating solutions (agents) searching different parts of the objective space. Special mechanisms introduced in this method assure both a tendency for approaching the nondominated set by the population and an inclination for dispersing the solutions over the whole nondominated set. The method shares with single objective simulated annealing the scheme of temperature changes and the dependence of the acceptance probability on the temperature level.

In section 6, multiple objective genetic local search algorithm is presented. In each iteration, the algorithm draws at random a scalarizing function. A pair of solutions selected from a set of previously generated solutions being good on the current scalarizing function is recombined. Local improvement procedure is applied to each offspring.

Section 7 describes the methodology of interactive analysis of MOCO problems with the use of a two-phase approach. In the first phase, a set of solutions approximating the whole nondominated set is generated. In the second phase, the set of solutions is analyzed with an interactive method for discrete alternative multiple objective problems. The advantages of the two-phase approach in comparison with the traditional approach assuming on-line generation of approximately Pareto-optimal solutions are discussed. The issue of computational effectiveness of the two-phase approach is further discussed in section 7.2. A discrete version of Light Beam Search method as well as a new Interactive Trichotomy Segmentation procedure developed for this kind of applications are presented.

Extensions of Pareto simulated annealing and Light Beam Search for fuzzy combinatorial optimization problems are presented in section 8. The extensions were motivated by real-life MOCO problems with uncertain data, e.g. software project scheduling. The extensions required the definition of the dominance in the fuzzy objective space, modification of rules for calculating probability of accepting new solutions, and definition of scalarizing function in the fuzzy objective space.

Section 9 describes comparative experiments on multiple objective knapsack problem and multiple objective traveling salesperson problem.

Section 10 presents exemplary applications of the proposed methods to nurse scheduling, multiple objective software project scheduling and logistic distribution system design.

The last section summarizes the work and discusses further research directions.

In the appendix a C++ library implementing the two algorithms proposed in this work, as well as a number of other well known multiple objective metaheuristics is outlined. The library allows a relatively easy adaptation of the methods to a given problem.

2. Basic definitions related to multiple objective optimization

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

$$\begin{aligned} &\text{maximize}\left\{f_1(\mathbf{x})=z_1,\dots,f_J(\mathbf{x})=z_J\right\} \\ &\text{s.t.} \qquad \mathbf{x}\in D, \end{aligned} \tag{P1}$$

where: *solution* $\mathbf{x}=[x_1,\dots,x_I]$ is a vector of *decision variables*, D is the set of feasible solutions. If the variables are discrete the MOO problem is called *multiple objective combinatorial optimization* (MOCO) problem.

The image of a solution $\mathbf{x}$ in the objective space is a *point* $\mathbf{z}^{\mathbf{x}}=\left[z_1^{\mathbf{x}},\dots,z_J^{\mathbf{x}}\right]=\mathbf{f}(\mathbf{x})$, such that $z_j^{\mathbf{x}}=f_j(\mathbf{x})$, j=1,..,J.

Point $\mathbf{z}^1$ *dominates* $\mathbf{z}^2$, $\mathbf{z}^1\succ\mathbf{z}^2$, if $\forall j\ z_j^1\geq z_j^2$ and $z_j^1>z_j^2$ for at least one j. Solution $\mathbf{x}^1$ dominates $\mathbf{x}^2$ if the image of $\mathbf{x}^1$ dominates the image of $\mathbf{x}^2$.

A solution $\mathbf{x}\in D$ is *Pareto-optimal* (*efficient*) if there is no $\mathbf{x}'\in D$ that dominates $\mathbf{x}$. Point being an image of a Pareto-optimal solution is called *nondominated*. The set of all Pareto-optimal solutions is called the *Pareto-optimal set*. The image N of the Pareto-optimal set in the objective space is called the *nondominated set* or *Pareto front*.

An *approximation of the nondominated set* is a set A of points (and corresponding solutions) such that $\neg\exists\mathbf{z}^1,\mathbf{z}^2\in A$ such that $\mathbf{z}^1\succ\mathbf{z}^2$, i.e. set A is composed of mutually nondominated points.

By $ND(S)$, where S is a set of (feasible or infeasible) points in the objective space, we will denote a *set of points nondominated within the set S*.

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

$$z_j^*=\max\left\{f_j(\mathbf{x})\,|\,\mathbf{x}\in D\right\} \qquad j=1,\dots,J.$$

The point $\mathbf{z}_*$ composed of the worst attainable objective function values in the nondominated set is called the *nadir point*.

An approximation of the ideal point based on set A is dentoed by $\mathbf{z}^{**}(A)$:

$$z_j^{**}(A)=\max\left\{z_j\,|\,\mathbf{z}\in A\right\} \qquad j=1,\dots,J\ .$$

Range equalization factors ([154], ch. 8.4.2) are defined in the following way:

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

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

Weighted linear scalarizing functions are defined in the following way:

$$s_l(\mathbf{z},\Lambda)=\sum_{j=1}^{J}\lambda_j z_j=\sum_{j=1}^{J}\lambda_j f_j(\mathbf{x})$$

where $\Lambda=[\lambda_1,...,\lambda_J]$ is a weight vector such that $\lambda_j\geq 0\ \forall j$. Each weighted linear scalarizing function has at least one global optimum (maximum) belonging to the set of Pareto-optimal solutions [154]. A Pareto-optimal solution that is a global optimum of a weighted linear scalarizing function is called *supported Pareto-optimal solution* [168].

Weighted Tchebycheff scalarizing functions are defined in the following way:

$$s_\infty(\mathbf{z},\mathbf{z}^0,\Lambda)=\max_j\{\lambda_j(z_j^0-z_j)\}=\max_j\{\lambda_j(z_j^0-f_j(\mathbf{x}))\},$$

where $\mathbf{z}^0$ is a reference point, $\Lambda=[\lambda_1,...,\lambda_J]$ is a weight vector such that $\lambda_j\geq 0\ \forall j$. Each weighted Tchebycheff scalarizing function has at least one global optimum (minimum) belonging to the set of Pareto-optimal solutions. For each Pareto-optimal solution $\mathbf{x}$ there exists a weighted Tchebycheff scalarizing function s such that $\mathbf{x}$ is a global optimum (minimum) of s ([154], ch. 14.8).

Minimization of the weighted Tchebycheff scalarizing function corresponds to a min-max problem. The problem can be, however, transformed to the following one:

minimize α (P2)

s.t.

$\alpha\geq\lambda_j(z_j^0-z_j)=\lambda_j(z_j^0-f_j(\mathbf{x})),\qquad j=1,...,J$,

$\mathbf{x}\in D$.

Composite scalarizing functions are defined in the following way:

$$s_c(\mathbf{z},\mathbf{z}^0,\Lambda,\omega)=(1-\omega)s_\infty(\mathbf{z},\mathbf{z}^0,\Lambda)-\omega s_1(\mathbf{z},\Lambda)=(1-\omega)\max_j\{\lambda_j(z_j^0-z_j)\}-\omega\sum_{j=1}^{J}\lambda_j z_j\ ,$$

where $\omega\in<0,1>$.

Achievement scalarizing functions are defined in the following way:

$$s_a(\mathbf{z},\mathbf{z}^0,\Lambda,\rho)=\max_j\{\lambda_j(z_j^0-z_j)\}+\rho\sum_{j=1}^{J}\lambda_j(z_j^0-z_j)=$$

$$=\max_j\{\lambda_j(z_j^0-f_j(\mathbf{x}))\}+\rho\sum_{j=1}^{J}\lambda_j(z_j^0-f_j(\mathbf{x}))=$$

$$=\max_j\{\lambda_j(z_j^0-f_j(\mathbf{x}))\}-\rho\sum_{j=1}^{J}\lambda_j f_j(\mathbf{x})+\text{const}\,,$$

where ρ is a sufficiently small positive number. Note that the definition the achievement scalarizing functions is equivalent to the definition of the composite scalarizing functions with low value of ω.

Weight vectors that meet the following conditions:

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

are called *normalized weight vectors*.

A *utility function u*: $\Re^J \rightarrow \Re$, is a model of the DM's preferences that maps each point in the objective space into a value of utility. It is assumed that the goal of the DM is to maximize the utility.

A utility function u is *compatible with the dominance relation* (or coordinatewise increasing) if and only if $\forall \mathbf{z}^1, \mathbf{z}^2 \in \Re^J \quad \mathbf{z}^1 \succ \mathbf{z}^2 \Rightarrow u(\mathbf{z}^1) \geq u(\mathbf{z}^2)$. Each utility function of this type has at least one global optimum belonging to the set of Pareto-optimal solutions. For each Pareto-optimal solution $\mathbf{x}$ there exists a utility function u compatible with the dominance relation such that $\mathbf{x}$ is a global optimum of u ([154], ch. 6.6-6.7). The set of all utility functions compatible with the dominance relation is denoted by U_C.

Weighted linear utility functions are defined in the following way:

$$u_1(\mathbf{z}, \Lambda) = \sum_{j=1}^{J} \lambda_j z_j = \sum_{j=1}^{J} \lambda_j f_j(\mathbf{x}) = s_l(\mathbf{z}, \Lambda).$$

where $\Lambda = [\lambda_1, ..., \lambda_J]$ is a weight vector such that $\lambda_j \geq 0 \; \forall j$. The set of all weighted linear utility functions is denoted by U_1.

Weighted Tchebycheff utility functions are defined in the following way:

$$u_\infty(\mathbf{z}, \mathbf{z}^0, \Lambda) = -\max_j \{\lambda_j (z_j^0 - z_j)\} = -\max_j \{\lambda_j (z_j^0 - f_j(\mathbf{x}))\} = -s_\infty(\mathbf{z}, \mathbf{z}^0, \Lambda),$$

where $\mathbf{z}^0$ is a reference point, $\Lambda = [\lambda_1, ..., \lambda_J]$ is a weight vector such that $\lambda_j \geq 0 \; \forall j$. The set of all weighted Tchebycheff utility functions is denoted by U_∞.

The most criticized property of the utility function as a preference model is the transitivity of the indifference relation, which ignores the existence of an *indifference* region [115]. Existence of this region is caused by the inability of the DM to perceive small differences on a criterion as well as by imprecision and uncertainty of measurements and/or of a mathematical model.

Furthermore, experience indicates that, usually, there is no precise limit between indifference and strict preference, but there exists an intermediary region where the DM hesitates between both. This corresponds to situation of *weak preference* between two alternatives.

Another realistic situation which cannot be represented by the utility function model is the situation of *incomparability* corresponding to the lack of preference judgment between *a* and *b*.

The four conflicting situations of indifference, strict preference, weak preference and incomparability are useful in establishing a realistic representation of the DM's preferences. This claim has been formulated by Roy [139] as the *axiom of limited comparability* on which the European School of MCDA is largely based. It further states that the DM, or the analyst judging in his/her name, can develop a satisfactory preference model that assigns one or a grouping of two or three of these four basic situations to any pair of actions.

Indeed, the fact that two or three situations are considered possible by the DM for a given pair of alternatives is due to the fact that he/she cannot, does not wish to, or does not know how to decide upon the appropriate situation. A possible grouping of preference statements can be modeled by a consolidated system of preference relations (CSPR). The most popular are CSPR's that use the *outranking relation S* (see Roy, 1985). We say that *a* outranks b if a is considered to be at least as good as b. Thus, stating that $a\ S\ b$, the DM does not have to decide whether this means that a is strictly preferred to, weakly preferred to, or indifferent to b.

Using S for a pair of alternatives (a, b), one is able to represent indifference I, preference (weak Q or strict P) and incomparability R among a and b:

$$a\ S\ b \wedge b\ S\ a \Leftrightarrow a\ I\ b$$
$$a\ S\ b \wedge \neg b\ S\ a \Leftrightarrow a\ Q\ b \vee a\ P\ b$$
$$\neg a\ S\ b \wedge \neg b\ S\ a \Leftrightarrow a\ R\ b$$

Thus, the comprehensive preference model based on the outranking relation S is compatible with the axiom of limited comparability.

3. Review of methodology of multiple objective metaheuristics

3.1. Genetic/evolutionary algorithms

3.1.1. Vector Evaluated Genetic Algorithm

The first multiple objective metaheuristics proposed in the literature were based on genetic algorithms. The first method was Vector Evaluated Genetic Algorithm (VEGA) proposed by Schaffer in 1985 [144]. The method is a simple adaptation of single objective genetic algorithm with modified selection mechanism. At each generation, J subpopulations are generated. The j-th population is obtained by a standard selection of *PopulationSize*/J solutions according to the value of j-th objective. Then, the subpopulations are joined into a single population to which the recombination and mutation operators are applied in a regular way. The outcome of the algorithm is the final population obtained in the last generation.

Several applications of VEGA or its modifications were reported in the literature [18], [133], [159]. It was observed, however, that the solutions generated by VEGA tend to be concentrated in separate regions corresponding to the extreme values of particular objectives while the "middle" regions of the nondominated set are poorly represented [18]. Fonseca and Fleming [36] considered fitness landscapes induced by very large, uniformly distributed populations corresponding to various selection schemes used in multiple objective genetic/evolutionary algorithms. The fitness landscapes corresponding to VEGA selection scheme exhibit peeks around the optima of particular objectives. Some authors proposed modifications of VEGA that aim at avoiding the above drawback. For example, Tamaki et al. [161] propose to preserve solution nondominated in the current population. This may be interpreted as a form of hybridization with Pareto ranking-based multiple objective genetic algorithms described below.

3.1.2. Pareto ranking-based multiple objective genetic/evolutionary algorithms

At present probably most often used multiple objective genetic/evolutionary algorithms are based on Pareto ranking (compare [173], ch. 3.3.2.2 and 3.3.2.3). The family of methods is based on observation that in genetic/evolutionary algorithms the selection does not need to be based on a function but may be based on a ranking. This is especially evident in the case of tournament selection scheme, which requires choosing the best solution from a sample composed of two or more solutions. Such decision can be made on the basis of a ranking of the solutions. Even in the case of roulette wheel selection scheme the selection probabilities can be set based on the positions of the solutions in a ranking. In 1988 Goldberg outlined in a short 10-line sketch the idea of ranking the genetic population based on the dominance relation [46]. Although he did not implement the idea, a number of other authors used it in their own algorithms.

Two main ways of implementing the Goldberg's idea have been proposed in the literature. Fonseca and Flemming [35] assign to each solution in the population rank equal to the number of other solutions that dominate it plus one. Thus, solutions nondominated in the current population are assigned rank 1. Other solutions have rank greater than 1. Srinivas and Deb [152] proposed an approach, which is the most direct implementation of the Golberg's idea. Their algorithm called Nondominated Sorting Genetic Algorithm (NSGA) assigns rank 1 to all the solutions nondominated in the current population. Then, the nondominated solutions are temporarily removed from the population and the next rank is assigned to the solutions nondominated in the remaining part of the population. The process is continued until all the solutions in the population are ranked. Horn et al. [67] proposed a tournament selection approach in which two solutions are compared against a reference set of solutions being a subset of the whole population. If one of them is not dominated by any of the reference solutions while the other one is dominated, the former solution wins the tournament. If both of them are dominated or not dominated, then other rules are applied. In recently proposed Strength Pareto Evolutionary Approach (SPEA) [188] the ranking algorithm takes into account not only members of the current population but also the archive of potentially Pareto-optimal solutions generated so far. Osyczka and Kundu [128] proposed a fitness assignment method that grants rank 1 to solutions nondominated in the current population. The fitness of other solutions depends on their distance to the nondominated solutions. Thus, the method may be interpreted as hybridization with the scalarizing functions-based fitness assignment (see section 3.1.3).

A clear advantage of Pareto ranking is its independence of any monotonic transformation of objective functions. Note, however, that this kind of fitness assignment may promote regions with higher density of solutions. In such regions a given solution may dominate many others, even if the differences on objective values are very low.

Pareto ranking alone does not guarantee that the population will disperse over all regions of the nondominated set. In the above mentioned study, Fonseca and Fleming [36] consider fitness landscapes induced by very large, uniformly distributed populations. In the case of Pareto ranking-based selection schemes, all Pareto-optimal solutions have highest fitness, i.e. define a plateau of the fitness landscape. This situation is similar to optimization of a single function having global optima at a plateau. It is well known that in this case finite populations converge to a single optimum. This phenomenon is called "genetic drift" [47]. In the multiple objective case, genetic drift means that finite populations tend to converge to small regions of the Pareto-optimal set. Fonseca and Fleming [35] and Srinivas and Deb [152] propose the use of fitness sharing to prevent genetic drift. The idea of this technique is to penalize (decrease fitness) of solutions being too close, either in the objective or in the decision space, to some other solutions in the current population. Some particular mechanisms are used in order to preserve the pressure towards the nondominated set. For example, in Fonseca and Fleming algorithm [35] fitness sharing takes into account only solutions with the same rank. They propose also some special approaches to set the parameters of fitness sharing.

In classical single objective GAs all solutions from the current population may be recombined (mated) to produce offspring. This may be, however, very ineffective in the multiple objective case. The role of recombination is to join important features of the parents in a new solution. Features that often appear in good solutions have good chance to survive and spread in the genetic population. In the case of standard binary coding, such features are called schemas and correspond to some patterns of zeros and ones (see the Holland's schema theorem [66]). The question arises, whether in multiple objective case good solutions, i.e. solutions close to the nondominated set have some similarities. In general, there is no reason to expect such similarities. Pareto-optimal set of a multiple objective problem includes, among other solutions, optima of particular objectives. If the objectives are not positively correlated their optima will be, in general, completely different. The question of similarities between approximately Pareto-optimal solutions was experimentally studied by Borges and Hansen [7] for multiple objective traveling salesperson problem. Significant similarities (measured by the number of common arcs) between good solutions of single objective TSP were earlier observed in experiments of Mühlenbein [123] and Boese et al. [6]. Borges and Hansen in their study found significant similarities between approximately Pareto-optimal solutions placed close in the objective space. However, the greater the distance between two solutions in the objective space the less similar these solutions. This suggests that recombination of approximately Pareto-optimal solutions being distant in the objective space is very unlikely to yield good offspring. Fonseca and Fleming [35] propose the use of mating restrictions to avoid not promising recombinations, i.e. they propose to ban mating of distant solutions.

Note that fitness sharing and mating restrictions do in some sense opposite jobs. Fitness sharing penalizes close solutions while mating restrictions ban mating of distant solutions. Thus, parameters of these techniques should be carefully set. Furthermore, many fitness sharing and mating restrictions techniques use some kinds of distance measure in the objective space. Hence, the techniques are dependent on scaling of objectives. In result, the use of the techniques may nullify the mentioned above advantage of independence of any monotonic transformation of objective functions.

Another disadvantage of Pareto ranking is that the selection pressure decreases with the growing number of objectives. Consider, for example a population of N randomly generated solutions (e.g. starting population of a genetic algorithm) of a J-objective problem. Let us assume that the probability that one of two solutions is better on objective $j\in[1,..,J]$ is equal to 0.5, i.e. we assume that the probability of having equal values can be neglected. Thus, the probability that a solution x is dominated by y is equal to 0.5^J. The probability that solution x is dominated by at least one solution in the population is $1-(1-0.5^J)^{(N-1)}$. For N=100, and J=2, 3, 7, 10 the probability is equal to 0.9999999999996, 0.9999982, 0.534, and 0.0922, respectively. This means that for J=7 about half of the solutions, and for J=10 about 90% of the solutions will be assigned rank 1. Of course, genetic populations in further generation are not random; however, in general one may expect "flatter" ranks (i.e. ranks with fewer levels) for larger numbers of objectives.

An interesting approach aiming at avoiding the above drawback was proposed in [142]. A solution is evaluated better than another solution on a given objective j only if the difference on this objective exceeds a specified threshold. Otherwise, the two solutions are considered indifferent on this objective. This corresponds to the use of the indifference threshold [141]. In other words, the ranking of solutions is constructed based on an outranking relation instead of the dominance relation.

Most of the applications of Pareto ranking-based multiple objective genetic/evolutionary algorithms concern nonlinear continuous problems [70]. Such problems arise for example in engineering design. Few applications concern combinatorial problems. Hilliard et al. [65] applied a Pareto ranking method to bi-objective scheduling problem and concluded that it outperforms VEGA. Liepins et al. [113] applied such an algorithm to a set covering problem. Todd and Sen [164] solved a containership layout problem. In a recent experiment Ziltzer and Thiele [188] applied several multiple objective genetic algorithms to the multiple objective 0/1 knapsack problem with multiple knapsack constraints. Among others they used Pareto ranking-based algorithms. The best results were achieved with SPEA method.

3.1.3. Scalarizing functions-based multiple objective genetic/evolutionary algorithms

Some authors proposed selection schemes based on scalarizing functions in order to induce pressure towards the nondominated set. In order to spread the search over all regions of the nondominated set various objective weights should be used. Hajela and Lin [49] proposed to encode the weights in the solutions description. Thus, the weights evolve in results of recombination and mutation operators. Fitness sharing is used in order to achieve diversification of weights. The weights are used in linear scalarizing functions. Murata et al. [124] proposed to draw at random a weight vector for the use in each iteration composed of a single recombination. The two solutions for recombination are selected according to the roulette wheel scheme. The fitness of solutions changes from iteration to iteration and depends on the value of the current weighted linear scalarizing function. Murata et al. [124] applied this method to a flowshop scheduling problem.

3.1.4. The outcome of multiple objective genetic/evolutionary algorithms

Many authors of multiple objective genetic/evolutionary algorithms assume that the outcome of their methods is the set of solutions being Pareto-optimal within the final population [144], [35], [152]. In this case, however, many potentially Pareto-optimal solutions generated in previous generations may be lost. Thus, some authors proposed to use an external set of potentially Pareto-optimal solutions [109], [124], [164], [188]. In some cases, the set of potentially Pareto-optimal solutions plays an active role in the algorithms. For example Murata et al. [124] use this set as a source of elite solutions added to each generation, while in Zitzler's and Thiele's SPEA algorithm [188] all solutions from this set participate in selection.

3.2. Local search methods

3.2.1. Simulated annealing

The first multiple objective version of simulated annealing was proposed by Serafini [146] in 1992. The method uses the standard scheme of simulated annealing with single current solutions. The outcome of the algorithm is the set of potentially Pareto-optimal solutions containing all the solutions not dominated by any other solution generated by the algorithm. Serafini considered a number of acceptance rules, defining acceptance probability of new neighborhood solutions. Single objective simulated annealing accepts with probability equal to one new solutions better or equal to the current solution. If the new solution is worse than the current solution it is accepted with a probability lower than one. In the multiple objective case three situations may appear while comparing the new solution **x'** with the current solution **x** (see Figure 1):

- **x'** may dominate **x,**
- **x'** may be dominated by **x,**
- **x** and **x'** may be mutually nondominated.

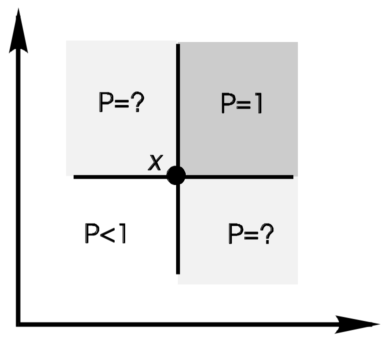

Figure 1. The probability of acceptance in multiple objective acceptance rules in the case of two maximized objectives

In the first situation it is natural to accept **x'** with probability equal to one. In the second case the acceptance probability should be lower than one. The question arises what should be the acceptance probability in the third case. Serafini proposed a number of multiple objective acceptance rules that differently treat the third situation. The rules correspond to some local aggregation functions that could be interpreted as weighted scalarizing functions with reference point at the image of the current solution **x**. Serafini analyzed convergence of the algorithm to the nondominated set in a way similar to that used in single objective case. He was able to show that the acceptance rules guarantee that the algorithm achieves one of the Pareto-optimal solutions if the temperature is decreased sufficiently slowly. Thus the acceptance rules assure a tendency for approaching the nondominated set but they do not guarantee appropriate dispersion over the whole set of nondominated points. In order to achieve this dispersion Serafini proposed to modify the weights randomly from iteration to iteration. The weights were allowed to change by no more than 10%.

Ulungu et al. [170] proposed a method called MOSA. They also use multiple objective acceptance rules. MOSA uses a number of predefined weight vectors. Each of the vectors is associated with an independent annealing process. Each process starts from a random solution or a solution constructed by a specialized heuristic. Then, the solution performs local moves that are accepted with probability defined by an acceptance rule. The outcome of the algorithm is the set of potentially Pareto-optimal solutions containing all the solutions not dominated by any other solution generated by any of the annealing processes. Thus, the processes co-operate in generating the common set of potentially Pareto-optimal solutions. As, however, the update of the set of potentially Pareto-optimal solutions does not influence acceptance probability, each of the processes is completely independent. The independence of the annealing processes allows natural parallelization of the algorithm. The set of the predefined weight vectors is a new parameter of the algorithm. MOSA have been applied to a bi-objective assignment problem [166] and a problem of homogenous grouping of nuclear fuel cans [171].

Suppapitnarm and Parks [158] proposed a method in which the acceptance probability of a new solution depends on whether or not it is added to the set of potentially Pareto-optimal solutions. If it is added to this set, which means that it is not dominated by any known solution, it is accepted to be the current solution with probability equal to one. Otherwise, a multiple objective acceptance rule is used.

3.2.2. Tabu search

Gandibleux et al. [44] proposed a multiple objective version of tabu search. The method assumes optimization of weighted scalarizing functions whose weights are changed periodically. The modification of weight vector degrades weights of objectives that were significantly improved. Two tabu lists are used. One is a regular tabu list that prevents returning to already visited solutions. The second tabu list contains weight vectors. Gandibleux and Fréville [43] suggested that in bi-objective case it is enough to use a predefined set of weight vectors. In this case the method becomes a direct correspondent of MOSA method [170] (see section 3.2.1).

Hansen [50] proposed a multiple objective tabu search based on the idea of Pareto simulated annealing proposed by us (see section 4). The method uses a sample (population) of solutions that search different regions of the nondominated set. A weight vector is associated with each of the solutions. The weight vector is used in a scalarizing function. Furthermore, with each solution a separate tabu list is associated. The main element of the method taken from Pareto simulated annealing is the mechanism assuring proper dispersion of the population of solution. The dispersion of the solutions is achieved by modification of their weight vectors. For each solution the aim of the weight vector modification is to move it away from other solutions close in the objective space. Hansen applied this method to the multiple objective knapsack problem. Viana [175] applied this method to a multiple objective project scheduling problem.

Ben Abdelaziz and Krichen proposed a multiple objective tabu search algorithm designed especially for multiple objective knapsack problem [4]. The method is guided by both weighted linear scalarizing functions and dominance relation. The dominance relation is applied not only to solutions but also to the items. Local search is based on the idea of inserting the nondominated items to the knapsack. It is not clear if the idea of the method may be extended to other problems.

3.2.3. Other local search methods

Recently Borges [8] proposed a new approach called changing horizon efficient set search (CHESS) that can be used with any type of local search method. The idea is to guide the local search by a function that depends on the current set of potentially Pareto-optimal solutions. The function may be interpreted as a generalization of the Tchebycheff scalarizing function over the set of potentially Pareto-optimal solutions. The function is not constant but it changes whenever the set of potentially Pareto-optimal solutions set is changed. Borges proposed an effective technique for calculation of the function based on quad trees data structure [34], [48]. He used a number of local search techniques in his experiment including simulated annealing and tabu search.

Another recent proposition is Pareto archived evolution strategy (PAES) developed by Knowles and Corne [100]. It is a local search method based purely on dominance relation. The acceptance of a new neighborhood solution depends on its comparison with the current solution and with an archive of potentially Pareto-optimal solutions. Thus, the method shares with CHESS the idea of referring new neighborhood solutions to the set of the potentially Pareto-optimal solutions.

3.3. Hybrid genetic algorithms

In recent years, the development of hybrid genetic algorithms is one of the most significant trends in the field of metaheuristics. Methods of this kind hybridize recombination operators with local heuristics, e.g. local search. Other frequently used names are memetic algorithms or genetic local search. It is quite difficult to track the single origin of GLS. To our knowledge, the first description of GLS was published by Ackley [1], but similar algorithms were developed probably completely independently by several authors. Heuristics based on the GLS scheme often prove to be extremely effective in combinatorial optimization. In particular, methods of this type were able to improve best known results on a number of well studied combinatorial optimization problems, e.g. Traveling Salesperson Problem [119], Graph Coloring Problem [41] and Quadratic Assignment Problem [160].

Ishibuchi and Murata [75] were the first authors to propose a multiple objective genetic local search algorithm. The main idea of their method is to randomly generate a weight vector for each iteration. The weight vector is used in a scalarizing function. Originally they suggested the use of linear functions, but the method can be easily generalized to other weighted scalarizing functions. Each iteration of the method consists of a single recombination and a single run of local search applied to the offspring. The method uses a standard genetic population with no mating restrictions. The solutions for recombination are selected according to the roulette wheel scheme, with fitness depending on the current scalarizing function. Generation replacement is used. Furthermore, a portion of elite potentially Pareto-optimal solutions is added to the new generation. Ishibuchi and Murata applied their method to a multiple objective flowshop problem.

Knowles and Corne [101] proposed a method called Memetic-Pareto Archive Evolution Strategy (M-PAES) being extension of PAES (see section 3.2.3). The method uses a population of solutions. Each solution works according the PAES algorithm with the difference that each of them uses a separate local set of potentially Pareto-optimal solutions. Periodically, the method employs recombination to distinct local optima.

Recently Balicki and Kitowski [3] proposed and adaptive multiple objective evolutionary algorithm hybridized with tabu search. The algorithm uses Pareto ranking to assign fitness to solutions. Tabu search algorithm optimizing the position in Pareto ranking is applied to some solutions. The solution improved by tabu search replaces its ancestor. Note that the idea of the algorithm is very general and any other local search algorithm may be used in place of tabu search. The algorithm has been applied to a task assignment problem. Note that Pareto ranking is, in general, not well suited for local optimization. Change of the rank of a given solution may require significant changes of the objective values; so, many local moves will not influence the rank. In the case of solutions having rank 1, no local improvement is possible, even, if the solutions are far from the efficient set. Furthermore, evaluation of local moves depends on other solutions in the population. In the case of some problems, for which evaluation of local moves is very fast, it may significantly increase computational time of local optimization.

3.4. Ant colonies

A number of authors proposed adaptation of ant colonies algorithm to multiple objective case. Mariano and Morals [118] proposed to use a separate colony of ants for each objective. They considered particular problems, where each objective depends only on a part of the solution description. They assume that the objectives are ordered according to their importance. In each iteration, an ant from colony j receives a single solution from colony j–1. Then the ant from j-th colony tries to improve the solution on objective j. The solutions that remain potentially nondominated after passing all the colonies update the pheromone information.

Gambardella and Taillard [42] proposed an algorithm for bi-objective case and applied it to a vehicle routing problem. They also used two separate colonies for each objective. The two colonies share the same best solution that is used for pheromone updates in both colonies. The objectives differ in importance. Whenever the colony corresponding to the more important objective finds a new better solution on this objective both colonies start anew with this best solution.

Iredi et al. [74] developed an algorithm for bi-objective case. It uses a number of colonies, each of them exploring a different region of the nondominated set. The colonies are heterogeneous because each ant uses a different weight vector. The colonies co-operate by exchanging the information about the set of potentially Pareto-optimal solutions. The pheromone information is updated by ants that correspond to solutions nondominated within their colony. The authors studied a number of approaches to pheromone updates and weight vectors setting. The method was applied to a bi-objective scheduling problem.

Shelokar et al. [148] proposed an ant colonies algorithm using the fitness assignment proposed in SPEA method [188]. The fitness of an ant takes into account both members of the current colony and the archive of potentially Pareto-optimal solutions generated so far. The algorithm uses two kinds of ants. Global ants search for good regions in the search space. Local ants try to improve the fitness locally within a given region. The algorithm was tested on relatively small continuous problems. The authors concluded that the use of the fitness assignment proposed in SPEA helps to achieve a good distribution of the solutions over the nondominated set.

3.5. Measures of the quality of approximations of the nondominated set

Since a significant number of multiple objective metaheuristics have been proposed the issue of their comparison becomes of crucial importance. As in the single objective case, multiple objective metaheuristics are evaluated mainly experimentally. The evaluation of metaheuristics should take into account at least two main criteria – computational requirements and the quality of results.

The issue of measuring computational requirements of multiple objective metaheuristics is analogous to the single objective case. The running time of the methods is measured by either the number of functions evaluations or the CPU time. The advantage of the first measure is its independence of the details of implementation and the characteristic of computers used in the experiments. On the other hand, the measure does not take into account that some methods have the advantage of allowing very fast evaluation of solutions. This applies especially to local search methods. In the case of many combinatorial problems, neighborhood solutions may be evaluated in time much shorter than needed by for example recombination operator. Thus, local search methods often may evaluate many more solutions than genetic/evolutionary algorithms in the same amount of time.

The real challenge in evaluation of multiple objective metaheuristic is, however, measurement of the quality of their results. The outcomes of the multiple objective methods are approximations of the nondominated set, i.e. sets of points and corresponding solutions that are mutually nondominated. Note that under the assumption that the DM makes his/her decisions purely on the basis of solutions' images in the objective space, the issue of evaluating the approximations boils down to evaluation of sets of mutually nondominated points. In this section, we analyze the main quality measures used in the literature. Note that in the single objective case an obvious quality measure is the value of the objective function. Thus, the issue analyzed in this section is characteristic to multiple objective (meta-) heuristics.

Experimental comparison of multiple objective metaheuristics is probably the main motivation for development of the quality measures of approximations of the nondominated set. In addition, the quality measures may be used to:

- Adjust parameters of metaheuristic algorithms for a given problem. Such parameter adjusting is usually done in a series of experiments whose results obviously have to be evaluated.
- Define stopping rules of metaheuristic algorithms. Many single objective metaheuristics use adaptive stopping rules based on observation of the objective function value. For example, a method may be stopped if the best objective function value is not improved in a number of iterations. The quality measures may submit some ideas for stopping rules in multiple objective case.
- Develop new ideas for adaptation of the single objective metaheuristics to the multiple objective case.

The quality measures proposed in the literature may be classified as either cardinal or geometrical. The idea of the cardinal measures is to count the number or ratio of points that meet some requirements. To define the requirements equivalence and/or dominance relation is used. Geometrical measures take into account geometrical positions of the nondominated points in the objective space.

3.5.1. Cardinal quality measures

One of the most natural cardinal measures is the ratio of points from a reference set RS found. The reference set should be the actual nondominated set or a very good approximation of it. The measure may be defined in the following way:

$$Q_1(A) = \frac{|A \cap RS|}{|RS|}.$$

The measure has been used in [167] and [170]. The main disadvantage of this metric is that it requires the knowledge of the actual nondominated set. Even if a very good approximation is used instead it may so happen that the evaluated approximation will contain some actual nondominated points not contained in *RS*. Furthermore, multiple objective metaheuristics may be unable to find even a single actual nondominated point for large problems. In this case, the measure will not differentiate the quality of different approximations even if one of them is much better. Thus, the measure is limited to relatively small problems even if the actual nondominated set is known [169].

If the reference set does not contain all actual nondominated points then the points from *A*, which are not dominated by points contained in *RS*, may actually belong to the nondominated set. In this case, it may be more reasonable to use the following measure, which is defined as the ratio of approximation points not dominated by *RS*:

$$Q_2(A) = \frac{\left|\left\{\mathbf{u} \in A \mid \nexists \mathbf{r} \in RS \ \mathbf{r} \succ \mathbf{u}\right\}\right|}{|A|}.$$

This measure was used in [166] in experiments on a bi-objective assignment problem. In this case a set of supported nondominated points (i.e. points corresponding to supported solutions) being a subset of *N* was easy to find. This set was used as the reference set. Note that this measure may grant very high evaluations to the approximations that concentrate in a very small region of the nondominated set. In particular, an approximation composed of a single point not dominated by any point in *RS* will have the highest evaluation equal to 1.

Van Veldhuizen proposed a measure similar to Q_1 – ratio of points in the approximation that are actual nondominated points:

$$Q_3(A) = \frac{|A \cap N|}{|A|}.$$

This measure has similar properties to Q_1. Note also that alike Q_2 it may grant very high evaluations to the approximations that concentrate in a very small region of the nondominated set. In particular, an approximation composed of a single actual nondominated point will have the highest evaluation equal to 1.

Zitzler and Thiele [187], [188] proposed a measure called coverage of two approximations of the nondominated set. The measure allows relative evaluation of two approximations *A* and *B*. The measure is defined as:

$$Q_4(A,B) = \frac{\left|\left\{\mathbf{z}'' \in B\right\} \mid \exists \mathbf{z}' \in A : \mathbf{z}' \succeq \mathbf{z}''\right|}{|B|}.$$

The value $Q_4(A, B) = 1$ means that all points in *B* are dominated by or equal to (covered by) some points in *A*. The value $Q_4(A, B) = 0$ means that no point in *B* is covered by any point in *A*.

3.5.2. Geometric quality measures

A number of quality measures tend to measure a “distance” between the actual nondominated set or a reference set and an approximation of the nondominated set.

Czyżak and Jaszkiewicz [23], [26] proposed the following distance measure based on a reference set *RS*:

$$Q_5(A) = \frac{1}{|RS|} \sum_{\mathbf{r} \in RS} \min_{\mathbf{z} \in A} \left\{ s_\infty(\mathbf{z}, \mathbf{r}, \Lambda) \right\},$$

where $\Lambda = [\lambda_1,....,\lambda_J]$, $\lambda_j = 1/R_j$, $j = 1,...,J$,with R_j being the range of objective j in set *RS*. In other words, the measure is the average distance from each reference point to its closest neighbor in A. This measure was also used in [166], [170] and [175]. Note that since set *RS* is assumed to be the actual nondominated set or a very good approximation of it the scalarizing function $s_\infty(\mathbf{z},\mathbf{r},\Lambda)$ should never take on values below zero. In this case, it may be interpreted as a weighted Tchebycheff metric measuring the distance between the reference and approximate points.

Another measure based on the same idea takes into account the worst case distance from a reference point to its closest neighbor in A:

$$Q_6(A) = \max_{\mathbf{r}\in RS} \min_{\mathbf{z}\in A} \{s_\infty(\mathbf{z},\mathbf{r},\Lambda)\}.$$

The measure was used together with Q_5 in [23], [26], [166], [170] and [175]. Furthermore, the ratio:

$$Q_5(A)/Q_6(A)$$

is a measure of uniformity of the quality of approximation.

Another kind of distance measure was proposed by Van Veldhuizen and Lamont [174]. It measures the average Euclidean distance from a point in N to its closest neighbor in A:

$$Q_7(A) = \frac{1}{|N|} \sum_{\mathbf{r}\in N} \min_{\mathbf{z}\in A} \{d(\mathbf{z},\mathbf{r})\},$$

where $d(.,.)$ denotes the Euclidean distance in the objective space.

Zitzler and Thiele [187], [188] proposed to use as a quality measure the hyperarea of the part of the objective space covered by an approximation. By the space covered they understand the part of the objective space that contains points dominated or equal to some points in the approximation A being evaluated. Obviously, hyperarea of the covered space is infinite, so, the authors assumed implicitly that the lower bound on each maximized objective is equal to 0. This choice is obviously arbitrary and may be invalid for many problems. Van Veldhuizen [173] proposed a modification of this measure for the case when the actual nondominated set is known. He proposed to measure the quality by the ratio of the hyperarea covered by an approximation over the hyperarea covered by the actual nondominated set.

Fonseca and Fleming [37] proposed a geometric technique for comparison of pairs of approximations and pairs of algorithms. The technique uses attainment surface, i.e. the boundary between the space covered by an approximation and the rest of the objective space. Attainment surfaces are sampled in a number of lines starting from an origin at e.g. point with zero coordinates. If a given line intersects the attainment surface of approximation A further from the origin than the attainment surface of approximation B, approximation A is considered better on this line than approximation B. Furthermore, Fonseca and Fleming proposed to run each algorithm several times. Thus, each algorithm generates a sample of attainment surfaces. The intersection points along each line give a univariate distribution, which can be analyzed using standard non-parametric statistical tests. While comparing two samples of intersection points corresponding to attainment surfaces generated by two algorithms the statistical significance of the hypothesis that one of the algorithms generates better approximation along this line may be tested. Then, the percentage of lines on which approximations generated by an algorithm were significantly better than approximations generated by the other algorithm is a measure of relative quality of the two algorithms.

The geometric measures are obviously dependent on the objective functions scaling. If the range of an objective z_j is much greater than the ranges of other objectives, objective z_j may have predominant influence on the quality measures. Thus, normalization of the objective function values to comparable ranges is a necessary step in application of such measures.

3.6. Summary of the review

Multiple objective metaheuristics are grounded on various single objective methods. Although their single objective roots significantly differ some fundamental ideas used in adaptation of metaheuristics to multiple objective case may be observed.

Single objective metaheuristics use the value of the objective function to evaluate the intermediate solutions and guide the search for better solutions. In multiple objective case two main approaches to evaluation of intermediate solutions predominate. The evaluation mechanism uses either dominance relation or scalarizing functions. The first approach is used in Pareto ranking-based multiple objective genetic/evolutionary algorithms (see section 3.1.2), PAES [100], M-PAES [101] and Shelokar's et al. ant colonies algorithm [148]. Scalarizing functions are used in Murata et al. multiple objective genetic algorithm [124], Serafini's simulated annealing [146], MOSA [170], Gandibleux et al. tabu search [44], Hansen's tabu search [50], CHESS [8], Iredi's et al. ant colonies algorithm [74] and Ishibuchi's and Murata's genetic local search [74]. Some methods use both the ideas, e.g. Suppapitnarm's and Parks's [158] simulated annealing or Ben Abdelaziz's and Krichen's tabu search [4]. The use of dominance relation has the advantage of being independent of any monotonic scaling of the objective functions. The dominance relation alone may, however, be insufficient to assure dispersion over all regions of the nondominated set. Scalarizing functions, on the other hand require more parameters, e.g. weight vectors, reference points, and are dependent on functions scaling. The advantage of scalarizing functions is the possibility of forcing particular solutions to explore the desired regions of the nondominated set. In addition, the use of scalarizing functions often allows to take advantage of problem specific heuristics for construction of initial solutions and/or local improvement. Note that the use of scalarizing functions may not be interpreted as an aggregation of multiple objectives to a single function. The scalarizing functions are just tools used in the search for approximately Pareto-optimal solutions. Their use has nothing to do with modeling the decision maker's preferences in the form of an aggregate function.

Other fundamental differences in adaptation of single objective metaheuristics to multiple objective case concern the general algorithmic scheme of the methods. Some multiple objective metaheuristics apply the unmodified schemes of the single objective root algorithms with adjusted selection mechanism. Typical examples of such approach are most of the Pareto ranking-based genetic/evolutionary algorithms (see section 3.1.2). They use the general scheme of a genetic algorithm and differ in the way of assigning the fitness to solutions. Among local search methods this idea is used in Serafini's simulated annealing [146], CHESS [8], Suppapitnarm's and Parks's [158] and PAES [100]. Methods of this kind treat the search for sets of solutions being good approximations of the nondominated set in the same way as the search for good solutions in the single objective case. Single objective methods, however, are usually based on the idea that good solutions have some common properties. For example, genetic/evolutionary algorithms recombine features of good solutions in order to obtain new good solutions, local search methods search for new good solutions in a neighborhood of known good solutions, ant colonies use pheromones to attract new solutions to regions of the search spaced explored by good solutions. In the multiple objective case, in general, there is no reason to expect all approximately Pareto-optimal solutions to have some common properties. In fact, in practice the objectives are often conflicting. Thus, solutions good on one objective may be completely different from the solutions good on other objectives (see section 3.1.2). Thus, the task of multiple objective metaheuristics differs significantly from the task of single objective methods.

Other methods use algorithmic schemes developed especially for the multiple objective case employing the elements of the root methods on the lower levels of their algorithms. For example, a number of local search methods, e.g. MOSA [170], Gandibleux et al. tabu search [44], Hansen's tabu search [50] use several independent or co-operating solutions each of them exploring the search space according to the rules of simulated annealing or tabu search. Murata's et al. [124] and Ishibuchi's and Murata's [74] algorithms in each iteration perform selection of solutions for single recombination, each time using a randomly selected scalarizing function. M-PAES [101] periodically recombines distant local optima generated with the use of PAES [100] algorithm. Iredi et al. [74] use multiple colonies of heterogeneous ants. In general, the goal of the development of the new algorithmic schemes is to achieve better covering of different regions of the nondominated set. For example, the local search methods that use several solutions try to assure that each of them explores a different region of the nondominated set.

As it was mentioned in section 3.5, the quality measures of approximations of the nondominated set may submit some new ideas for adaptation of single objective metaheuristics to the multiple objective case. It seems natural that the intermediate solutions generated by a multiple objective metaheuristic should be evaluated in a way similar to the techniques used to evaluate the outcomes of the method. Many authors, however, evaluate results of their algorithms using techniques much different to the techniques used inside the algorithms. In particular, the authors of methods using dominance based evaluation mechanisms often use geometric quality measures [37], [100], [101], [102], [103], [187], [188] to evaluate outcomes of their algorithms. On the contrary, the methods described in this work use evaluation mechanisms compatible with the quality measures proposed in section 4. In fact, the idea of the multiple objective genetic local search algorithm (see section 6) was directly motivated by our work on evaluation of approximations of the nondominated set.

4. Preference-based evaluation of approximations of the nondominated set

4.1. Outperformance relations

As it was mentioned in section 3.5 the quality measures of approximations of the nondominated set are based on either cardinal or geometric ideas. In this section, we analyze the issue of evaluating the quality of the approximations on the basis of some assumptions about the DM's preferences [52].

As it was stated in section 1.1, the goal of multiple objective optimization is to find the single solution giving the *best compromise* between multiple objectives. Since usually there is no single solution that optimizes simultaneously all the objectives, selection of the best compromise solution requires taking into account preferences of the DM. For example, if the DM's preferences correspond to a utility function, the best compromise is the solution that maximizes the utility. Under very weak and generally accepted assumptions about compatibility of the DM's preferences with the dominance relation the best compromise solution belongs to the set of Pareto-optimal solutions ([154], ch. 6.6-6.7). Note that this is not valid if the DM searches for a sample of best solutions as the second best and other good solutions do not need to be Pareto-optimal under the same assumptions about the DM's preferences.

In practice the preferences of the DM are often not known a priori. In many cases the DM may not have established preferences at the beginning of the solution process. Even if the preferences are well established the DM may be unable or unwilling to deliver the preference information necessary to build a complete model of his/her preferences. In such situations, generation of approximations of the whole or a part of the nondominated set is justified. This is, however, just the first phase of the solution process. In the second phase the DM selects the best compromise solution possibly supported by an interactive procedure, various kinds of visualization and statistical information about the set of solutions. Of course, the solution selected by the DM from an approximation is, in general, worse than the best solution contained in the actual nondominated set. Therefore, the DM may consider approximation A to be better than approximation B if he/she can find a better compromise solution in A than in B.

Note that even if the DM's preferences are unknown, one may be able to make some general assumptions about possible DM's preferences. An approximation could be considered good if it contains a good compromise solution for any possible preferences held by the DM. Therefore, the main idea of our framework is to evaluate approximations of the nondominated set under all possible DM's preferences.

Assume that according to some incomplete knowledge about DMs preferences they can be restricted to utility functions belonging to a set U. We restrict our analysis to utility function preference model because it always allows a comparison of two approximations for a given utility function. Note that any final decision (i.e. the final selection of the best compromise) of the DM can be explained by a utility function.

Let A and B be two approximations. Let $U(A > B) \subseteq U$ denote a subset of utility functions for which approximation A is better that B, i.e. $U(A > B) = \{u \in U \mid u^*(A) > u^*(B)\}$, where $u^*(A) = \max_{\mathbf{z} \in A}\{u(\mathbf{z})\}$ is the maximum value reached by the utility function u on an approximation A. Then, the following relation may be defined.

Definition 1 (Outperformance relation subject to a set of utility functions)

Approximation A *outperforms B subject to a set U of utility functions*, i.e. $A\ O_{/U}\ B$, if $U(A > B) \neq \emptyset$ and $U(B > A) = \emptyset$, i.e. if there exist some utility functions in set U that achieve better values in A than in B, while the opposite is not true.

Obviously, the weaker the assumptions about DM's preferences, the more general the outperformance relation. The weakest, generally accepted assumption about the DM's preferences is that his/her utility function is compatible with the dominance relation [136]. In other words, the DM never prefers a solution that is dominated. This assumption means that the DM can limit the search for the best compromise solution to the set of Pareto-optimal solutions ([154], theorem 6.11). In consequence, when two approximations A and B from which the DM must select a compromise solution are known, the DM can limit the search for the best compromise to the set $ND(A \cup B)$, i.e. the set of points (and corresponding solutions) nondominated within $A \cup B$. This allows us to define the following three outperformance relations based on the assumption about compatibility of the DMs preferences with the dominance relation.

Definition 2 (Weak outperformance)

Approximation A *weakly outperforms* B, $A\ O_W\ B$, if $A \neq B$ and if $ND(A \cup B) = A$, i.e. if for each point $\mathbf{z}^2 \in B$ there exists a point $\mathbf{z}^1 \in A$ that is equal to or dominates $\mathbf{z}^2$ and at least one point $\mathbf{z}^1 \in A$ is not contained in B.

An approximation that weakly outperforms B can be obtained by adding to B a new point that is nondominated with respect to all points in B as illustrated in Figure 2.

Note that Definition 2 is equivalent to outperformance relation subject to the set U_C of utility functions compatible with the dominance relation. It is also equivalent to the outperformance relation subject to the set U_∞ of weighted Tchebycheff utility functions.

Definition 3 (Strong outperformance)

Approximation A *strongly outperforms* B, $A\ O_S\ B$, if $ND(A \cup B) = A$ and $B \setminus ND(A \cup B) \neq \emptyset$, i.e. if for each point $\mathbf{z}^2 \in B$ there exists a point $\mathbf{z}^1 \in A$ that is equal to or dominates $\mathbf{z}^2$ and at least one point $\mathbf{z}^2 \in B$ is dominated by a point $\mathbf{z}^1 \in A$.

An approximation that strongly outperforms B can be obtained by adding to B a new point that dominates at least one point in B as illustrated in Figure 2.

Definition 4 (Complete outperformance)

Approximation A *completely outperforms* B, $A\ O_C\ B$, if $ND(A \cup B) = A$ and $B \cap ND(A \cup B) = \emptyset$, i.e. if each point $\mathbf{z}^2 \in B$ is dominated by a point $\mathbf{z}^1 \in A$.

An approximation that completely outperforms B can be obtained by adding to B a new point(s) that dominate all points in B as illustrated in Figure 2.

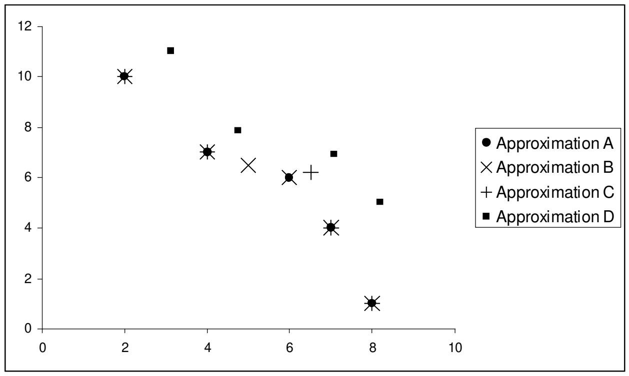

Figure 2. Examples of weak, strong and complete outperformance, for example $B\ O_W\ A$, $C\ O_S\ A$, $D\ O_C\ A$

Note that $O_C \subset O_S \subset O_W$, i.e. complete outperformance is the strongest and weak outperformance is the weakest of the outperformance relations. Each of the relations defines an incomplete ranking in the set of all approximations, i.e. some of the approximations remain incomparable.

If no additional assumptions about the DM's preferences except of the compatibility with the dominance relation are justified, one may compare two or more approximations using the above three dominance based outperformance relations. They allow to assess whether one approximation is better than another is and to grade the level of outperformance. For example, one can run two different multiple objective heuristics on a number of instances and report the percentages of cases when approximations generated by one of the methods completely, strongly and weakly outperformed approximations generated by the other method, as well as the percentage of cases in which the generated approximations were incomparable.

As it was mentioned above, many pairs of approximations will remain, however, incomparable with respect to the weak outperformance (and so with respect to the stronger relations), i.e. none of the approximations will weakly outperform the other one, as is the case in Figure 3. In this case, one may try to introduce stronger assumptions about the DM's preferences, e.g. by defining another set U of possible utility functions, and test if one of the approximations outperforms the other one subject to this set. For example, the two approximations presented in Figure 4 are incomparable with respect to the weak outperformance, but approximation A outperforms B subject to the set U_1 of weighted linear utility functions. Of course, the assumptions about the DM's preferences should be justified in a given context.

Furthermore, the outperformance relations allow for qualitative comparison only. In many cases one will be interested in not only whether an approximation is better than another one but also how much better it is. For example both approximations B and C presented in Figure 5 are completely outperformed by approximation A. Intuitively, however, the difference between A and B is much smaller than the differences between A and C and between B and C.

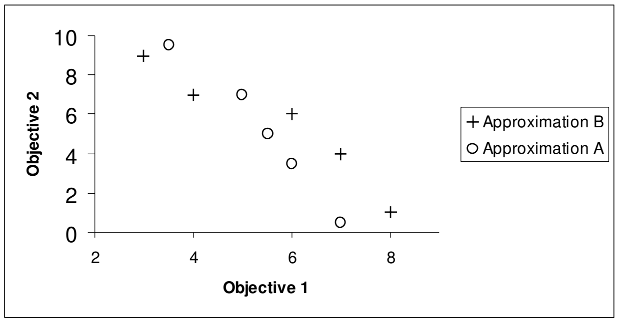

Figure 3. Approximations incomparable according to the weak outperformance

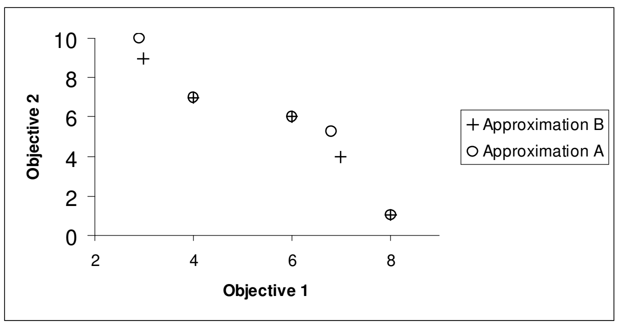

Figure 4. Approximations incomparable according to the weak outperformance but comparable according to outperformance subject to the set U_1 of weighted linear utility functions – $A \not{O}_W B$ **but** $A\, O_{/U_1}\, B$

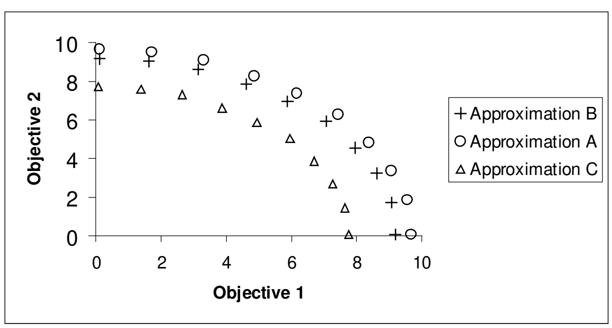

Figure 5. Various levels of differences between approximations

A possible extension of the outperformance relations would be a quality measure that would take into account the amounts or ratios of solutions defining particular relations. For example, the weak outperformance requires existence of just a single solution in *A* that is nondominated in $A \cup B$ and does not belong to B. It seems natural that taking into account the number or ratio of such solutions would contain more information about the "strength" of the relation. In fact this idea is used in the coverage measure Q_4 described in section 3.5.1. Notice that $A\ O_W\ B \Leftrightarrow Q_4(A, B) = 1$ and $Q_4(B, A) < 1$. Furthermore, $A\ O_C\ B \Leftrightarrow Q_4(A, B) = 1$ and $Q_4(B, A) = 0$. Thus, if $Q_4(A, B) = 1$ then $Q_4(B, A)$ may be intuitively interpreted as a measure of the "strength" of the weak outperformance, i.e. the lower $Q_4(B, A)$ the "closer" the relation between the two approximations of the complete outperformance. Note, however, that counting the number of such solutions has no interpretation from the point of view of the DM's preferences. In particular, $Q_4(A, B_1) = 1$ and $Q_4(B_1, A) < Q_4(B_2, A) < 1$ does not imply that B_2 is better than B_1 under assumption about compatibility of the DM's preferences with the dominance relation.

4.2. Quantitative comparison methods

As it was mentioned above the outperformance relations will, in general, leave many pairs of approximations incomparable under justified assumptions about the DM's preferences. The quality measures described in section 3.5 usually try to represent the quality of an approximation or the relative quality of two approximations with a single number and, in general, allow comparison of approximations incomparable according to the outperformance relations. Note that any such comparison of a pair of approximations going beyond the outperformance relation under justified assumptions about the DM's preferences my be discordant with the actual opinion of the DM, i.e. the DM may prefer the best solution selected from an approximation that has been evaluated worse according to the quality measure.

Below we propose some preference-based quality measures that require, however, much stronger assumptions about the DM's preferences.

4.2.1. Comparison based on probability

In section 4.1 we have assumed that the only known property of the DM's preferences is that they are restricted to a set of possible preferences, e.g. a set of utility functions. Assume now that each of the considered utility functions has a given probability of being the one held by the DM. When comparing two approximations A and B, it would then make sense to consider approximation A as being the better one, if there is a high probability for the utility functions in which approximation A presents a better solution than approximation B.

We express the probabilities of utility functions by an intensity function $p(u)$ and introduce an outcome function for the comparison of two approximations using a given utility function:

$$C(A,B,u) = \begin{cases} 1 & if\ u^*(A) > u^*(B) \\ ½ & if\ u^*(A) = u^*(B) \\ 0 & if\ u^*(A) < u^*(B) \end{cases}$$

We then build the measure $R1$ to reflect the probability that approximation A is better than approximation B by integrating over all utility functions:

$$R1(A,B,U,p) = \int_{u \in U} C(A,B,u)\, p(u)\, du$$

According to this measure approximation A is better than B if $R1(A,B,U,p) > ½$ and A is not worse than B if $R1(A,B,U,p) \geq ½$. Notice that $R1(A,B,U,p) = 1 - R1(B,A,U,p)$.

As an example, Figure 6 shows the u* values of two approximations A={[3,10],[4.9,7],[9,7]} and B={[2,9],[5,6],[10,6]} using the Tchebycheff utility function with $\mathbf{z}^0$ = [10,10] and $\Lambda = [t, 1-t]$, $t \in \langle 0,1 \rangle$:

$$u(t) = \left\{ u_\infty\left(\mathbf{z}, \mathbf{z}^0, \Lambda(t)\right) = -\max_j \left\{ t(10 - z_1), (1-t)(10 - z_2) \right\}, t \in <0,1> \right\}$$

Only when $t > 0.8$ will the approximation B contain the best solution. If we presume that the utility functions are distributed as follows by letting t belong to the continuous uniform distribution, $t \in U\langle 0,1 \rangle$, we get $R1(A,B) = 0.8$.

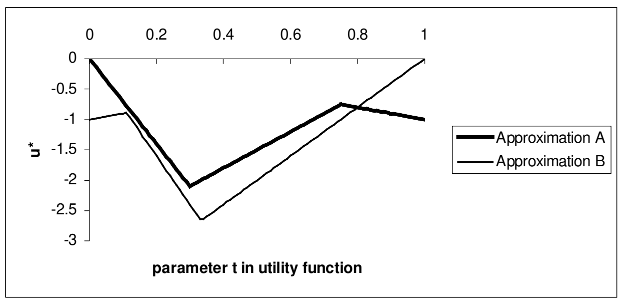

Figure 6. Values of u* on approximations A and B

When using $R1$ to rank more than 2 approximations it is possible to obtain cycles where $R1(A,B,U,p) < ½$, $R1(B,C,U,p) < ½$ and $R1(B,C,U,p) < ½$, indicating that B is better than A, that C is better than B and that A is better than C. Figure 7 outlines such an example.

4.2.2. Comparison based on expected values

Consider the following two approximations A={[1,10],[10,1.8]} and B={[2.2,10],[7,–1]}. Values of u^* are presented in Figure 8, where u is a Tchebycheff utility function defined in the same way as in Section 4.2.1. When t < 0.5125, approximation B contains the best alternative. So, applying measure $R1$ we obtain $R1(B,A) = 0.5125$, so B is evaluated better than A. However, in the region where approximation B gives better utility approximation A gives only slightly worse results, while in the other region the utility given by approximation A is significantly better. So, intuitively we might evaluate A better than B.

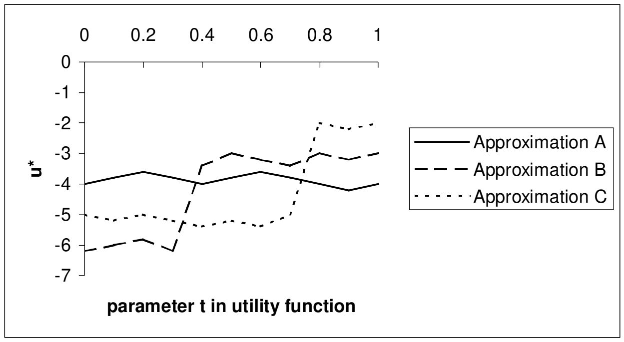

Figure 7. Example of cycles in ranking

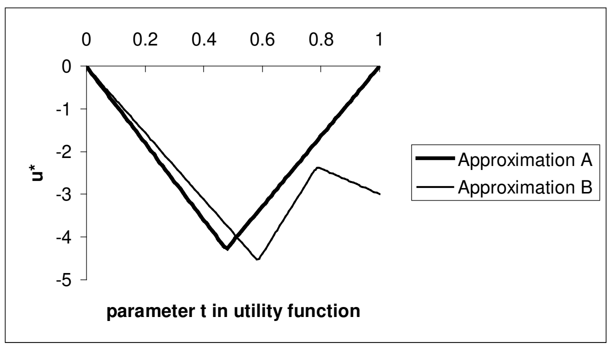

Figure 8. Values of u^* on approximations A and B

We propose to use the following measure that takes into account expected values of the utility:

$$R2(A,B,U,p) = E\left(u^*(A)\right) - E\left(u^*(B)\right) = \int_{u\in U} \left(u^*(A) - u^*(B)\right) p(u)\,du =$$
$$= \int_{u\in U} u^*(A)p(u)\,du - \int_{u\in U} u^*(B)p(u)\,du = R2'(A,U,p) - R2'(B,U,p)$$

Measure $R2$ is a relative quality measure defined as the difference of two values of absolute measure $R2'$. According to this measure approximation A is better than B if $R2(A,B,U,p) > 0$ and A is not worse than B if $R2(A,B,U,p) \geq 0$. Obviously, $R2(A,B,U,p) = -R2(B,A,U,p)$.

In the above example we obtain $E(u^*(A)) = -2.1239$, while $E(u^*(B)) = -2.5924$, and therefore $R2(A,B,U,p) = 0.4684 > 0$. So, approximation A is evaluated better than B.

Measure $R2$ induces complete ranking in the set of all approximations.

Measure $R2$ is based on an assumption that we are allowed to add values of different utility functions. Therefore, it is also dependent on the scaling of the different utility functions. Assume for example that in the above example each Tchebycheff function is multiplied by $(1-t)^2$. In this case we obtain $E(u^*(A)) = -0.6361$, while $E(u^*(B)) = -0.59996$ and then $R2(A,B,U,p) = -0.03614 < 0$. So, approximation B is evaluated better than A.

The same method may be used to construct a quality measure that evaluates a single approximation A with respect to a fixed reference set RS:

$$R2_{RS}(A,U,p) = R2(RS,A,U,p) = E\left(u^*(RS)\right) - E\left(u^*(A)\right) = \\ = \int_{u\in U} u^*(RS)p(u)du - \int_{u\in U} u^*(A)p(u)du = \int_{u\in U} \left(u^*(RS) - u^*(A)\right)p(u)du$$

In this case, the lower the value of the measure is, the higher is the evaluation of A.

In some cases, ratios of best utility values may be more meaningful than their differences. Therefore, we propose also the following ratio measure:

$$R3_{RS}(A,U,p) = R3(A,RS,U,p) = E\left(\frac{u^*(RS) - u^*(A)}{u^*(RS)}\right) = \int_{u\in U} \frac{u^*(RS) - u^*(A)}{u^*(RS)} p(u)du$$

The idea of $R3_{RS}$ follows an approach often used in the single objective optimization where approximate solutions obtained by single objective heuristics are evaluated by the ratio of the deviation from a fixed bound, for instance the optimal value.

In general, also the values of the ratio measures are dependent on the scaling of the objective functions.

4.2.3. Expected value of the weighted Tchebycheff utility/scalarizing function measure

The general measures described in section 4.2.1 and 4.2.2 involve many parameters that have to be defined to obtain concrete measures. In particular, one has to decide for a particular type of utility functions, a particular probability distribution of the utility functions and a particular utility functions scaling. In this subsection, we propose a specific measure obtained by a common sense definition of parameters of measure $R2'$.

If no additional information about the possible preferences of the DM is available, one should choose a set of utility functions that does not neglect any nondominated point. Obviously, this is assured by the set U_c of utility functions compatible with the dominance relation but this set cannot be used in practice. We propose to use the set U_∞ of weighted Tchebycheff utility functions.

If two approximations are incomparable according to outperformance relation subject to the selected set of utility functions U then it is always possible to choose a probability distribution such that one of the approximations will be evaluated higher than the other one by any of the $R1 \div R3$ quality measures. This can be done by defining intensity function $p(u)$ that takes on high values in regions of set U where the preferred approximation gives higher utility than the other one and low values in the other regions. Of course, some of the distributions will be more "artificial" that the others.

An obvious requirement is that the intensity function $p(u)$ should be greater than 0 for all $u\in U$. Otherwise, functions with intensity function value equal to zero could be excluded from U. The probability distributions also should not "favor" any sub-region of U.

Weighted Tchebycheff utility functions are defined by the weight vector Λ. Therefore, the distribution of the utility functions can be defined via the distribution of weights. We propose to use normalized weight vectors, i.e. vectors belonging to the following set:

$$\Psi = \left\{ \Lambda \in \Re^J \mid \sum_{j=1}^{J} \lambda_j = 1 \text{ and } \lambda_j \geq 0, j = 1,.., J \right\},$$

We then propose to use a uniform distribution of weights, i.e. a distribution for which:

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

where $V(\Psi)$ and $V(\Psi')$ are Euclidean hyper-volumes of Ψ and Ψ', respectively. In other words, the probability of a weight vector belonging to Ψ' should be proportional to the hyper-volume of Ψ'. The normalized weight vectors should be applied to normalized objective values, i.e. the original objective values multiplied by range equalization factors (see section 2). The normalization defines also scaling of the utility functions.

An additional parameter of the weighted Tchebycheff utility functions is the reference point. We propose to use the ideal point or an approximation of this point.

In result we obtain the following measure:

$$R^u(A) = E\left(u_\infty^*\left(\mathbf{z}^0, A, \Lambda\right)\right) = \int_{\Lambda \in \Psi} u_\infty^*\left(\mathbf{z}^0, A, \Lambda\right) p(\Lambda) d\Lambda,$$

where $u_\infty^*\left(\mathbf{z}^0, A, \Lambda\right) = \max_{\mathbf{z} \in A}\left\{u_\infty\left(\mathbf{z}, \mathbf{z}^0, \Lambda\right)\right\}$ is the best value achieved by function $u_\infty\left(\mathbf{z}, \mathbf{z}^0, \Lambda\right)$ on approximation A.

Because the definition of the weighted Tchebycheff utility functions is almost the same as the definition of the weighted Tchebycheff scalarizing functions the measure may be interpreted as the expected value of the weighted Tchebycheff scalarizing function:

$$R(A) = E\left(s_\infty^*\left(\mathbf{z}^0, A, \Lambda\right)\right) = \int_{\Lambda \in \Psi} s_\infty^*\left(\mathbf{z}^0, A, \Lambda\right) p(\Lambda) d = -R^u(A),$$

where $s_\infty^*\left(\mathbf{z}^0, A, \Lambda\right) = \min_{\mathbf{z} \in A}\left\{s_\infty\left(\mathbf{z}, \mathbf{z}^0, \Lambda\right)\right\}$ is the best value achieved by function $s_\infty\left(\mathbf{z}, \mathbf{z}^0, \Lambda\right)$ on approximation A.

Since we do not know any procedure for exact calculation of the measure R, we propose to estimate the expected value through a numerical integration, i.e. to calculate the average value of the weighted Tchebycheff scalarizing function over a large sample of systematically generated normalized weight vectors. We propose to use all normalized weight vectors in which each individual weight takes on one of the following values: $\left\{ l/k, l = 0,...,k \right\}$, where k is a sampling parameter. The set of such weight vectors is denoted by Ψ_s and defined mathematically as:

$$\Psi_s = \left\{ \Lambda = [\lambda_1,...,\lambda_J] \in \Psi \mid \lambda_j \in \left\{0, 1/k, 2/k, ..., k-1/k, 1\right\}\right\}.$$

With a combinatorial argument, we notice that Ψ_s contains $\binom{k+J-1}{J-1}$ weight vectors. For example, for k=3 and J=3, Ψ_s={[0,0,1], [0,1/3,2/3], [0,2/3,1/3], [0,1,0], [1/3,0,2/3], [1/3,1/3,1/3], [1/3,2/3,0], [2/3,0,1/3], [2/3,1/3,0], [1,0,0]}. The estimation of measure R is calculated as:

$$\hat{R}(A)=\frac{\sum_{\Lambda\in\Psi_s} s_\infty^*\left(\mathbf{z}^0, A, \Lambda\right)}{\left|\Psi_s\right|}$$

Figure 9 illustrates that measures R and $\hat{R}$ are correlated (but not mathematically equivalent) to the volume between the reference point and the attainment surface of approximation A, which in turn is negatively correlated to the volume measure described in section 3.5.2.

All the previously proposed quality measures used to evaluate approximations of the nondominated set do not allow to assess the gap between the approximations and the actual nondominated set unless this latter set is known. The necessity of knowing the actual nondominated set usually limits the assessment of the gap to very small problems.

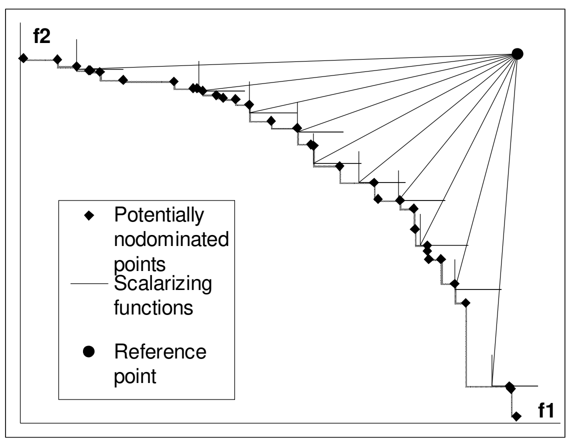

Figure 9. The idea of the expected value of weighted Tchebycheff scalarizing function measure

In the single objective case, the gap between an approximate solution value and the optimum value is often estimated by the gap between the approximate solution value and a lower bound (assuming minimization) on the objective. A good lower bound on the weighted Tchebycheff scalarizing function can often be calculated by optimization of the function on a relaxed problem, i.e. by solving problem P2 applying constraints of the relaxed problem. Knowing lower bounds on all the weighted Tchebycheff utility functions used in the numerical integration one may calculate the lower bound on measure $\hat{R}$. Let $LB\left(s_\infty\left(\mathbf{z},\mathbf{z}^0,\Lambda\right)\right)$ be a lower bound on scalarizing function $s_\infty\left(\mathbf{z},\mathbf{z}^0,\Lambda\right)$. The lower bound on $\hat{R}$ – $LB(\hat{R})$, is then equal to:

$$LB\left(\hat{R}(A)\right) = \frac{\sum_{\Lambda \in \Psi_s} LB\left(s_\infty^*\left(\mathbf{z}^0, A, \Lambda\right)\right)}{\left|\Psi_s\right|}.$$

4.3. Summary

The outperformance relations presented in section 4.1 leave many pairs of approximations of the nondominated set incomparable under justified assumptions about the DM's preferences. It is natural consequence of the fact that many pairs of points are incomparable according to the dominance relation. On the other hand, quantitative preference-based quality measures described in section 4.2 require information about the DM's preferences that may be difficult to get in practice. The possibility of interpretation of the parameters of the measures from the point of view of assumptions about the DM's preferences allows, however, for a common sense definition of the parameters. Such approach resulted in the definition of the expected value of the weighted Tchebycheff utility/scalarizing function quality measure.

More detailed discussion of the additional reference information that could be helpful in setting the parameters of the quantitative comparison measures may be found in [52]. This work contains also an analysis of compatibility with the outperformance relations of both some quality measures proposed in the literature and the quantitative quality measures described in section 4.2.

5. Pareto simulated annealing

Pareto simulated annealing (PSA) proposed in [20], [21] and developed further in [22], [23], [24], [26] is a multiple objective metaheuristic that uses the following concepts of its root method – single objective simulated annealing (SA) [110]:

- the concept of the neighborhood,
- probabilistic acceptance of new neighborhood solutions,
- dependence of the acceptance probability on a parameter called the temperature,
- the scheme of the temperature changes.

The characteristic features of the method are:

- The use of scalarizing functions based probabilities for acceptance of new neighborhood solutions.
- The use of a sample (population) of generating solutions each of them exploring the search space according to SA rules. The solutions may be treated as agents working almost independently but exchanging information about their positions. With each of the generating solutions a separate weight vector is associated.
- A rule for updating the generating solutions' weight vectors that aims at assuring dispersion of the solutions over all regions of the nondominated set.

5.1. Multiple objective acceptance rules

Each generating solution $\mathbf{x}$ may be modified by accepting a randomly generated solution from the neighborhood of $\mathbf{x}$. The new solution is accepted with some probability. PSA uses the concept of multiple objective acceptance rules proposed by Serafini [147] (see section 3.2.1). In the multiple objective case one of the following three exclusive situations may occur while comparing a new solution $\mathbf{y}$ with the current one $\mathbf{x}$ (see Figure 1):

- $\mathbf{y}$ dominates or is equal to $\mathbf{x}$,
- $\mathbf{y}$ is dominated by $\mathbf{x}$,
- $\mathbf{y}$ is nondominated with respect to $\mathbf{x}$.

In the first situation the new solution may be considered as not worse than the current one and accepted with probability equal to one. In the second situation the new solution may be considered as worse than the current one and accepted with probability less than one. Various acceptance rules treat the third situation in various ways.

Rule *C* may be interpreted as a local aggregation of all objectives with the weighted Tchebycheff scalarizing function with the reference point at $\mathbf{f}(\mathbf{x})$. The probability of accepting the new solution $\mathbf{y}$ is defined by the following expression:

$$P(\mathbf{x},\mathbf{y},T,\Lambda) = \min\left\{1, \exp\left(\max_j \left\{\lambda_j \left(f_j(\mathbf{x}) - f_j(\mathbf{y})\right)/T\right\}\right)\right\},$$

where T is the temperature, Λ is the weight vector. A graphical illustration of the rule is given in Figure 10.

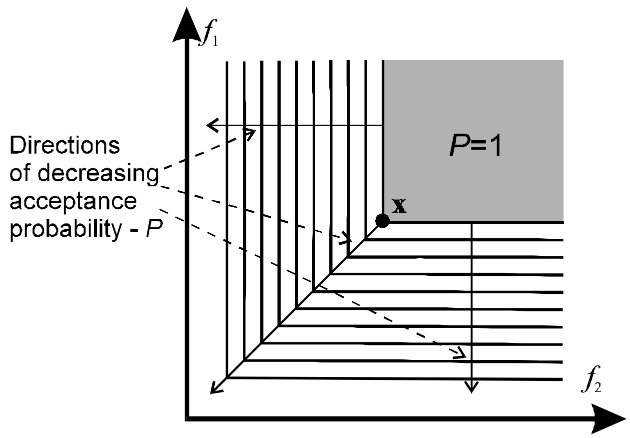

Figure 10. Graphical illustration of rule *C* – the case of two maximized objectives

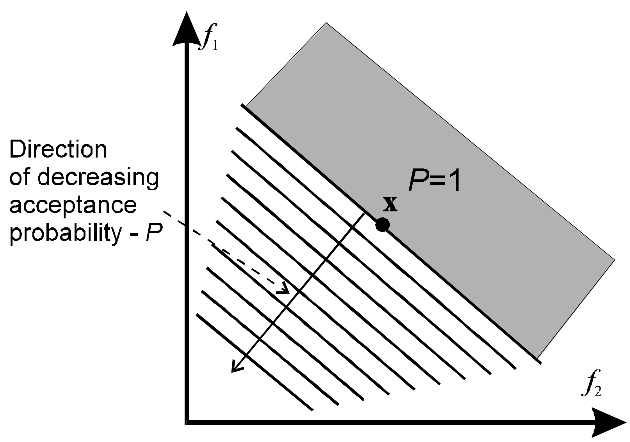

Figure 11. Graphical illustration of rule SL – the case of two maximized objectives

Rule *SL* may be interpreted as a local aggregation of all objectives with a weighted linear scalarizing function. It is defined by the following expression:

$$P(\mathbf{x},\mathbf{y},T,\Lambda) = \min\left\{1, \exp\left(\sum_{j=1}^{J} \lambda_j \left(f_j(\mathbf{x}) - f_j(\mathbf{y})/T\right)\right)\right\}.$$

A graphical illustration of the rule is given in Figure 11.

It is also possible to use composite acceptance rules based on the definition of the composite scalarizing functions.

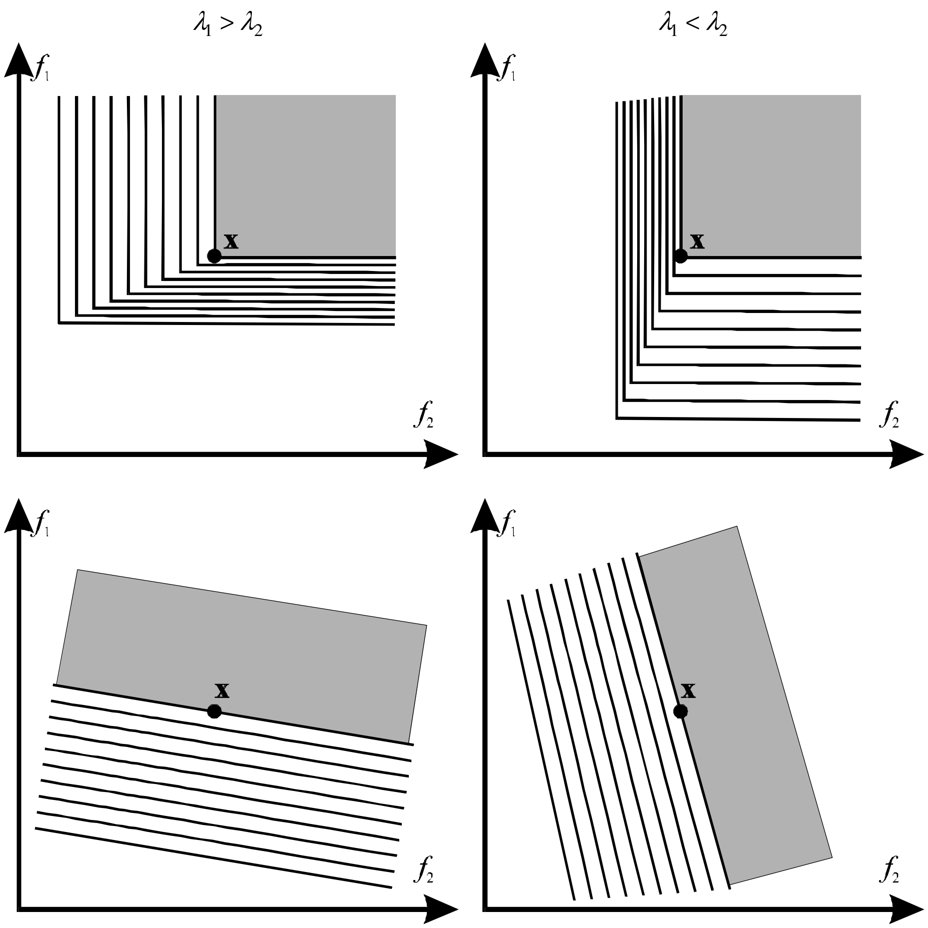

Figure 12. The role of weights in the multiple objective rules for acceptance probability

5.2. Management of the population of generating solutions

As it was mentioned in section 3.2.1 Serafini [146] has proven that the use of the multiple objective acceptance rules ensures a tendency for approaching the nondominated set, however, it may be insufficient to ensure dispersion of the generating solution over all regions of this set. The weights used in the acceptance rules allow to influence the direction of search in the objective space for particular generating solutions. Note that the higher the weight associated with a given objective the higher the influence of this objective on the probability of acceptance of new solutions, and the higher the pressure towards improvement of this objective. So, controlling the weight vectors the method may “push” generating solutions into desired directions in the decision space. This fact is graphically illustrated in Figure 12.

PSA controls the weight vectors associated with particular generating solutions in order to achieve a form of repulsion between the solutions. The weight vector associated with a given generating solution **x** is modified in order to increase the probability of moving **x** away from its closest neighbor **x**' in the generating sample. This is obtained by increasing the weights of the objectives on which **x** is better than **x**' and decreasing the weights of the objectives on which **x** is worse than **x**'. The Euclidean distance between solutions in the space of normalized objectives is used. An additional requirement is that the closest neighbor has to be nondominated with respect to **x**. If there is no generating solution that meets this requirement each weight is either increased or decreased with probability equal to 0.5. Note that the repulsion mechanism never repulses the generating solutions away from the nondominated set.

During the run of the method some of the generating solutions may get stacked in regions of the search space far from the nondominated set. If a generating solution is dominated by at least one of the other generating solutions for a number of generations, it is considered not promising and is replaced by a solution from the set of potentially Pareto-optimal solutions having maximum distance to the closest generating solution. In other words the idea is to move the generating solution to a poorly explored region of the nondominated set.

5.3. Normalization of objectives

In real-life applications the ranges of objectives usually differ a lot, often by many levels of magnitude. On the other hand the idea of PSA is to work with normalized weight vectors. So, the weight vectors should be applied to normalized objective values (see section 2). The range equalization factors are calculated from the ranges of particular objectives in the set of potentially Pareto-optimal solutions.

5.4. Updating the set of potentially Pareto-optimal solutions

The set of potentially Pareto-optimal solutions *PP* is empty at the beginning of the method. It is updated whenever a new solution is generated. Updating the set of potentially Pareto-optimal solutions with solution **x** consists of:

- adding **f**(**x**) to *PP* if no point in *PP* dominates **f**(**x**),
- removing from *PP* all points dominated by **f**(**x**).

The process of updating set *PP* may be very time consuming. Thus, the following techniques could be used in order to reduce the computational requirements:

- New solution **y** obtained from **x** should be used to update *PP* only if it is not dominated by **x**.
- New potentially Pareto-optimal solutions could be added to *PP* only if they differ enough from all solutions contained already in this set. We propose to use a threshold defining the minimum Euclidean distance in the space of normalized objectives between solutions in *PP*. A new potentially Pareto-optimal solution is neglected if it is closer to at least one solution in *PP* than the threshold.

- We have observed experimentally that the solutions added to *PP* in early iterations have a good chance to be removed from this set in further iterations. In other words the time spent on updating *PP* in early iterations is likely to be lost. Thus, it is possible to neglect updating *PP* in a number of starting iterations.
- A data structure called quad trees [34], [48] could be used to accelerate the process of updating *PP*. According to our experiments this approach reduces the running time in case of four and more objectives.

5.5. Using partial preference information

Generating the approximation of the whole nondominated set makes sense if the DM's preferences are completely unknown, or more precisely, if the only justified assumption about DM's preferences is their compatibility with the dominance relation. In many cases, however, some partial preference information is known. Such information does not allow to determine the single best solution (best compromise) but may allow focusing on the interesting region of the nondominated set. Examples of such partial preference information are:

- objective 1 is more important than objective 2,
- solutions having value on objective 1 below a given threshold are not interesting to the DM,
- the DM is interested in solutions close to a given point in the objective space.

The most natural way of taking into account partial preference information in PSA is possible if the preferences may be expressed in a form of constraints in the weight space. In this case, the algorithm should not allow for modifications of weight vectors that would violate the constraints. Of course, in addition the starting random weight vectors should also be within the feasible range. The simplest way to achieve this is to repeat drawing the random weight vectors until a feasible weight vector is drawn. Note that this approach should take into account the interpretation of weights in the scalarizing function used in a given acceptance rule. This interpretation is different in the case of *C* and *SL* rules.

If the partial preference information is expressed in a form of constraints in the objective space (e.g. second of the above examples) PSA can handle it as additional constraints. A number of techniques for handling constraints in simulated annealing may be used in this case. For example, a penalty function term may be added to the scalarizing function value in the acceptance rule. This would decrease the probability of accepting solutions that violate the constraints. It is also possible to reject all moves that violate the constraints. The latter approach may, however, result in a disconnected search space. Furthermore, it may be difficult to construct starting solutions meeting the additional constraints.

5.6. Summary of the algorithm

The basic version of PSA algorithm is summarized in Figure 13.

Parameters: number of generating solutions, starting temperature T_0, repulsion coefficient $\alpha > 0$ and close to one (e.g. 0.01), the rules of temperature decreasing, stopping criterion

Randomly generate a starting sample of generating solutions $S \subset D$ and associate a random weight vector with each of the solutions
for each $\mathbf{x} \in S$ **do**
  Update set *PP* of potentially Pareto-optimal solutions with **x**
$T := T_0$
repeat
  for each $\mathbf{x} \in S$ **do**
    Construct $\mathbf{y} \in V(\mathbf{x})$
    if **y** is not dominated by **x** **then**
      Update set *PP* with **y**
    Select solution $\mathbf{x}' \in S$ closest to **x** and nondominated with respect to **x**
    if there is no such solution **x**' **then**
      for each objective f_j
        $\lambda_j^{\mathbf{x}} = \lambda_j^{\mathbf{x}} + \alpha$ or $\lambda_j^{\mathbf{x}} = \lambda_j^{\mathbf{x}} - \alpha$ with probability equal to 0.5
    else
      for each objective f_j
        $$\lambda_j^{\mathbf{x}} = \begin{cases} \lambda_j^{\mathbf{x}} + \alpha & \text{,if } f_j(\mathbf{x}) \geq f_j(\mathbf{x}') \\ \lambda_j^{\mathbf{x}} - \alpha & \text{,if } f_j(\mathbf{x}) < f_j(\mathbf{x}') \end{cases}$$
    normalize the weights such that $\sum_j \lambda_j^{\mathbf{x}} = 1$
    $\mathbf{x} := \mathbf{y}$ (accept **y**) with probability $P\left(\mathbf{x}, \mathbf{y}, T, \Lambda^{\mathbf{x}}\right)$
  if the conditions for changing the temperature are fulfilled **then**
    decrease T
until the stopping criterion is met

Figure 13. Basic version of PSA algorithm

6. Multiple objective genetic local search

This section describes a multiple objective genetic local search (MOGLS) algorithm proposed and further developed by the author in [81], [84], [85] and [86]. The name of the method suggests that it hybridize recombination operators with local search. Indeed, this was the concept of the original proposition of the method. The idea of the method is, however, more general and allows hybridization with other local heuristics (compare section 9.3). In this work we refer, however, to this method with the name used in the original proposition.

6.1. Motivations

As it was mentioned in section 3.3 hybrid genetic algorithms (HGAs) (genetic local search, memetic algorithms) have proved in recent years to be a very effective class of methods for combinatorial optimization. The methods tend to achieve synergy of recombination operators and local heuristics. In some cases very simple local heuristics are used while other implementations use extensions of local search. For example, Radcliffe and Surry [131] consider an algorithm in which a single iteration of local search is applied to each offspring while Taillard [160] applies tabu search to each offspring.

From the genetic algorithms perspective, HGAs may be interpreted as standard genetic/evolutionary algorithms working on a reduced set of solutions, e.g. containing local optima only. From this point of view, local heuristic is just a part of the recombination operator. The efficiency of HGAs may be explained by the fact that they work on smaller search spaces than standard genetic/evolutionary algorithms. So, the best results should be achieved on problems where local optima constitute a relatively small part of the search space and the local optima can be achieved in an efficient way.

HGAs may be also interpreted as modifications of multiple start local heuristics with random starting solutions. In HGAs, starting solutions are constructed in an smart way that combines the properties of other good solutions. If the recombination operator is well designed, starting solutions obtained by recombination should constitute better starting points for local improvement than random solutions. The efficiency of HGAs in comparison with multiple start local heuristics can be explained by the fact that local heuristics when started from good starting solutions usually yield better solutions and often require less time.

The multiple objective genetic local search (MOGLS) algorithm described in this section is motivated by the good performance of the single objective algorithm presented in figure Figure 14 in a number of experiments and real-life applications [82], [87], [90].

In most of our experiments and applications the single objective optimization was stopped if the current population was not changed in K successive iterations. This usually indicates that the current population converged to a set of close solutions that cannot yield any further improvement.

Parameters: K – size of the current population, stopping criterion
Generation of the starting population:
  Current population P:=∅
  repeat K times
    Construct a new feasible solution **x** by a randomized algorithm
    Apply local heuristic to **x** obtaining **x'**
    Add **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 $\mathbf{x}_3$
    Apply local heuristic to $\mathbf{x}_3$ obtaining $\mathbf{x}_3$'
    if $\mathbf{x}_3$' is better than the worst solution in P and different from all the solutions in P **then**
      Add $\mathbf{x}_3$' to P and delete the worst solution from P
  until the stopping criterion is met

Figure 14. Basic single objective genetic local search algorithm

The above algorithm is not an orthodox version of a genetic/evolutionary algorithm. It assumes acceptance of every improvement, i.e. the algorithm puts more emphasis on the intensification mechanisms than on the diversification with respect to the more traditional algorithms using roulette wheel or tournament selection. Furthermore, the above algorithm does not explicitly introduce genetic generations; instead, it implements any new solution to the current population. The focus on intensification is motivated by the fact that good solutions of many combinatorial problems are contained in very small regions of the search space. For example, Boese et al. [6] estimate that all local optima of a traveling salesperson problem (TSP) instance with 100 cities are contained in a region that corresponds to only $1/10^{59}$ of the search space. For an instance with 500 cities this fraction is estimated to be $1/10^{468}$. Thus, a too strong diversification mechanism may easily guide the algorithm away from these regions.

Table 1. Comparison of three single objective algorithms

Multiple start local search			Simulated annealing			Genetic local search		
Number of local searches	Thousands of iterations	Average excess	Moves on temp. plateau	Thousands of iterations	Average excess	Population Size	Thousands of iterations	Average excess
10	470	0.0458	10000	440	0.0388	4	753	0.0206
20	940	0.0401	20000	880	0.0310	12	2027	0.0061
40	1903	0.0339	40000	1760	0.0295	20	3293	0.0022
80	3805	0.0298	80000	3520	0.0251	28	4804	0.0019
160	7608	0.0248	160000	7040	0.0168	36	5801	0.0007
320	15241	0.0192	320000	14080	0.0159	44	7365	0.0006
640	30436	0.0156	640000	28160	0.0116	52	7890	0.0006
1280	60882	0.0136	1280000	56320	0.0084	68	10740	0.0005

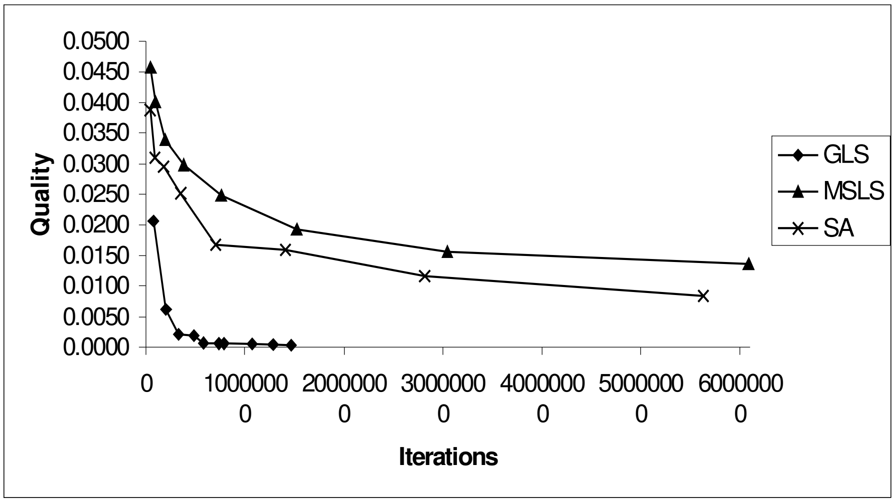

Figure 15. Comparison of three single objective algorithms

In order to illustrate the effectiveness of the above algorithm an experiment on the single objective traveling salesperson problem (TSP) is briefly presented. The recombination operator is described in section 9.1.4. The local heuristic used in this case was standard local search with 2-arcs exchange neighborhood. Greedy local search was used in *generation of the starting population* phase, while steepest local search was used in *main loop*. Such a selection of local search versions was made experimentally. The algorithm was compared with multiple start local search (MSLS) and simulated annealing (SA). In MSLS starting solutions were constructed randomly and greedy algorithm was used. The starting temperature of SA was set equal to 100 and the final temperature was equal to 1. An intensive experiment was performed in order to find good temperature settings for SA. After each temperature plateau the temperature was multiplied by 0.9. The number of moves on a temperature plateau was constant in each run of the procedure.

The three methods were compared on five problems coming from TSPLIB library [72], [132] – kroA100, kroB100,..., kroE100. The results presented in Table 1 and Figure 15 are averages of 25 runs of the algorithms (5 runs for each problem). The comparison of the algorithms takes into account two criteria – number of iterations (local moves) and the quality of results. Since for each of the 5 problems the global optimum is known, the quality is measured by relative excess over the optimum value, i.e. $\left(f - f_{opt}\right)/ f_{opt}$, where f is the obtained value of the objective function and f_{opt} is the optimum value. Note that the goal of the experiment was not to develop a very effective procedure for the single objective TSP but to illustrate that GLS can achieve synergy between relatively simple neighborhood operator and a recombination operator. More advanced systems for TSP may be found for example in [87], [119] and [163].

6.2. Random selection of scalarizing functions

The goal of multiple objective metaheuristics is to generate good approximations of the nondominated set. Of course, the best possible approximation is the whole nondominated set itself. As it was mentioned in section 2, all weighted linear and all weighted Tchebycheff scalarizing functions achieve optima at solutions corresponding to nondominated points. Thus, finding the whole nondominated set (with corresponding solutions) is equivalent to finding the optima of all weighted Tchebycheff and all weighted linear scalarizing functions. Hence, we reformulate the goal of multiple objective metaheuristics as simultaneous optimization of all weighted Tchebycheff or all weighted linear scalarizing functions. Obviously, it is also possible to use composite scalarizing functions. The term "optimization" in the previous sentence is understood as a tendency of the algorithm to improve values of all the scalarizing functions. Note that it is enough to consider scalarizing functions with normalized weight vectors.

Simultaneous optimization of all weighted Tchebycheff, all weighted linear or all composite scalarizing functions still seems to be an unfeasible task. Since, however, single objective metaheuristics generate new solutions in an iterative way, they tend to generate solutions dispersed in the objective space even if applied to optimization of a single scalarizing function (see Figure 16). Furthermore, because of its stochastic and heuristic elements, a metaheuristic which optimizes a given scalarizing function may generate good solutions on scalarizing functions with weight vectors close the weight vector of the function being optimized.

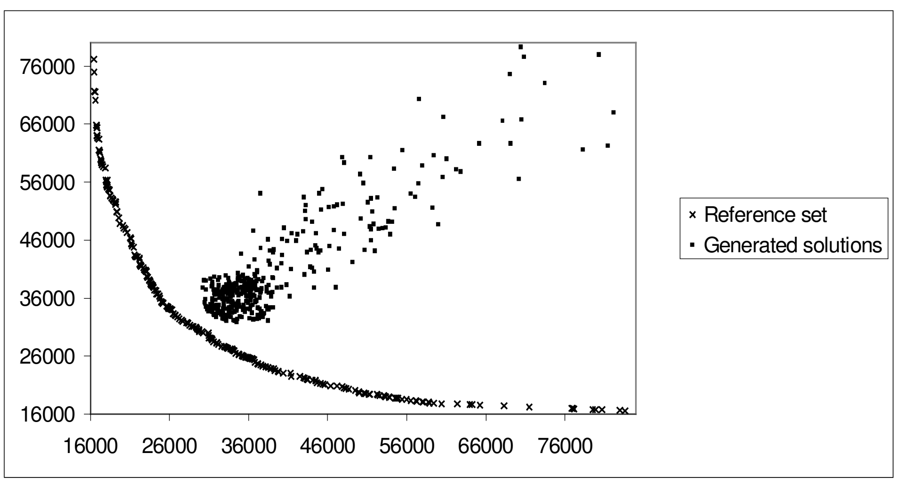

Figure 16. Solution generated by a GA optimizing a single weighted linear utility function for a bi-objective TSP with 50 cities. The reference set is a good approximation of the nondominated set generated by another method

To illustrate the latter phenomenon the following experiment was performed on a bi-objective TSP instance with 100 cities. GLS was used to optimize several weighted linear scalarizing functions with distinct weight vectors. The weight of the first objective λ_1 was set equal to 0.5, 0.505, 0.51,...,0.65. The weight of the second objective was equal to $1-\lambda_1$. Thus, 31 different objective functions were optimized. Each of the functions was optimized 1000 times. The scalarizing function with weight vector [0.5,0.5] was treated as the basic function. We estimated the probability that optimizing a given function with weight vector $[\lambda_1,1-\lambda_1]$ we find a better solution of the basic function than that found by optimization of the basic function (we excluded the cases when the same values were found). If the same algorithm optimized exactly the same basic function the probability would be equal to 0.5. The results are presented in Figure 17. One may notice that when a function with weight vector close to [0.5,0.5] is optimized the probability is very close to 0.5. The probability goes to zero when the difference between weight vectors increases. The results suggest that it is virtually irrelevant to optimize weighted linear scalarizing function with weight vector [0.5,0.5] or [0.515, 0.485].

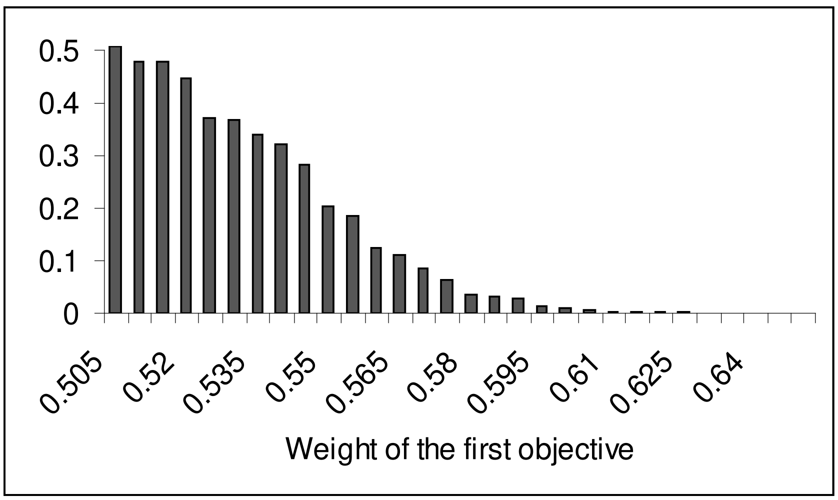

Figure 17. The estimated probability that optimizing linear weighted sum with given weight of the first objective GLS finds better solution of linear weighted sum with weights [0.5, 0.5] than GLS optimizing the second function

MOGLS implements the idea of simultaneous optimization of all weighted Tchebycheff, all weighted linear or all composite scalarizing functions with normalized weight vectors by random choice of the scalarizing function optimized in each iteration. In other words, in each iteration, MOGLS tries to improve the value of a randomly selected scalarizing function. A single iteration of MOGLS consists of a single recombination of a pair of solutions and application of a local heuristic that takes into account the value of the current scalarizing function. In order to draw at random the scalarizing function, a normalized weight vector is drawn at random by the algorithm presented in Figure 18. The algorithm assures that weight vectors are drawn with uniform probability distribution $p(\Lambda)$ (see section 4.2.3). The idea of attempting an improvement of a randomly selected scalarizing function in each iteration is in fact directly motivated by the expected value of weighted Tchebycheff scalarizing function quality measure (see section 4.2.3).

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

$$\ldots$$

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

$$\ldots$$

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

Figure 18. Algorithm for generating random normalized weight vectors. Function rand() returns a random value from the range <0,1> with uniform probability

Weighted Tchebycheff scalarizing functions seem to have the advantage of being able to generate all Pareto-optimal solutions even in the case of non-convex and discrete problems (see section 2). Note, however, that this property applies to global optima of the weighted Tchebycheff scalarizing functions. In the case of large size problems, multiple objective metaheuristics may be unable to find global optima of the functions. In some cases we have observed that the use of weighted linear functions gives better results than the use of weighted Tchebycheff functions (see sections 9.1, 9.3, 10.3). This is caused by the fact that local search may work better on linear functions (cf. [51]). Furthermore, sub-optimal solutions of linear scalarizing functions may yield higher probability of finding Pareto-optimal solutions (see section 9.1). Nevertheless, the use of weighted Tchebycheff functions is a more general approach and we propose to consider it as the default choice.

6.3. Selection of solutions for recombination

The main reason for the good performance of the single objective GLS is that the method combines features of two good solutions. The offspring produced in this way are good starting points for heuristic improvement. Analogously, MOGLS should combine features of solutions that are already good on the current scalarizing function. The general scheme of the algorithm is summarized in Figure 19.

The root algorithm (see Figure 14) recombines two solutions from a population containing K different solutions being the best on the objective function among all the prior generated solutions. We adapt this idea to the multiple objective case in the following way. In each iteration, our MOGLS algorithm constructs a temporary elite population composed of K different solutions being the best on the current scalarizing function among all known solutions. Then, two different solutions are drawn for recombination from the temporary elite population with uniform probability.

The idea of recombining solutions being already good on the current scalarizing function is motivated by "global convexity" exhibited by many combinatorial problems. This term introduced in [6] for single objective case means that good solutions of a problem are similar in the decision space. In multiple objective case it means that good solutions on a given scalarizing function being close in objective space are similar. Global convexity of multiple objective traveling salesperson problem was experimentally confirmed by Borges and Hansen [7] (compare section 3.1.2). Note that global convexity depends on the definition of similarity in the decision space. In the case of real-life problems it may not be obvious what solutions features are related to the solutions quality (compare [91]). Furthermore, the recombination operator should effectively merge the important features of the parent solutions in the offspring.

Generate the initial set of solutions
repeat
    Draw at random a weight vector Λ defining the current scalarizing function
    Select from the set of previously generated solutions two solutions being good on the current scalarizing function
    Recombine the two solutions
    Apply a local heuristic to the offspring
until the stopping criterion is met

Figure 19. General scheme of the multiple objective genetic local search algorithm

6.4. Generating the initial set of solutions

The initial set of solutions is constructed by applying iteratively the local heuristic to random starting solutions. The local heuristic optimizes the scalarizing functions with randomly generated weight vectors. The number of the initial solutions *S* is an additional parameter of the method. Its influence on the performance of the algorithm is yet to be tested. Below we propose an approach that allows to stop generating the initial solutions when the average quality of *K* best solutions in the set of initial solutions over all scalarizing functions is the same as the average quality of solutions generated by the local heuristic used for optimization of these functions. In other words, the method assures that on average the quality of *K* best solutions on a scalarizing function with a given weight vector will be the same as the quality of the staring population generated by the root single objective algorithm applied to optimization of this function.

Let $\mathbf{x}$ be an initial solution being obtained by local optimization of a scalarizing function s_x. Note that $\mathbf{x}$ does not need to be the best solution on s_x in the set *CS* of solutions containing all initial solutions. Let $B(K, CS, \mathbf{x}, s_x) \subseteq CS$ be the set of *K* best solutions of function s_x different from $\mathbf{x}$. Let $\bar{s}_x(B(K, CS, \mathbf{x}, s_x))$ be the average value of s_x in $B(K, CS, \mathbf{x}, s_x)$:

$$\bar{s}_x(B(K, CS, \mathbf{x}, s_x)) = \frac{\sum_{\mathbf{y} \in B(K, CS, \mathbf{x}, s_x)} s_x(\mathbf{y})}{K}.$$

We propose to stop generating the initial solutions when the following condition is met:

$$\frac{1}{|CS|} \sum_{x \in S} \left(\bar{s}_x(B(K, CS, \mathbf{x}, s_x)) - s_x(\mathbf{x}) \right) \geq 0. \qquad (1)$$

Of course, the above condition could only be tested when $|CS| \geq K + 1$.

Table 2 presents exemplary sizes of the initial sets of solutions obtained with the above approach in the case of multiple objective TSP (see section 9.1.1). The results are averages over 50 results – 5 runs for 10 instances of each size. In addition, standard deviations are presented. Note that the variance of the numbers of initial solutions is relatively low. The results have been obtained with the use of linear function (see section 9.1.5).

Table 2. Estimated numbers of starting solutions for multiple objective symmetric TSP problems. Standard deviations are given in brackets

K	Number of initial solutions			
	Bi-objective instances with 50 cities	Three-objective instances with 50 cities	Bi-objective instances with 100 cities	Three-objective instances with 100 cities
4	33.6 (3.13)	115.52 (8.07)	43.52 (2.87)	198.96 (11.84)
8	57.28 (3.74)	203.84 (14.20)	75.04 (5.08)	349.76 (18.57)
16	108.48 (6.70)	381.12 (20.87)	142.4 (6.66)	662.4 (31.17)
32	202.88 (15.31)	733.44 (35.90)	268.8 (15.83)	1290.88 (41.70)

6.5. Management of the current set of solutions

Our original idea was to store all the generated solutions in the current set of solutions *CS* from which the temporary elite populations are selected. Storing and handling all solutions would, however, be very time and memory consuming. Thus, 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 elite population and different from all solutions in the temporary elite population. If the size of the queue is bigger than $K \times S$ then the last solution from the queue is removed. The size $K \times S$ was established experimentally. We have observed that such a size of *CS* results in no significant deterioration of results with respect to the version of our algorithm that stores all generated solutions. Note that set *CS* should not be interpreted as a genetic population. It is just the source of solutions placed in the temporary elite populations in particular iterations.

6.6. Updating the reference point

In the case of weighted Tchebycheff and composite scalarizing functions, the reference point is an important parameter playing a role equivalent to the weight vectors, i.e. by changing the reference point with constant weight vector having all elements greater than 0 one can locate the optimum of the function at any nondominated point. In MOGLS an approximation of the ideal point composed of the best known values of each objective is used as the reference point. The reference point changes (improves) during the run of the method. Such a choice of the reference point has, however, a disadvantage if one of the weights, say λ_j, is close to one. It this case, it is likely that the local heuristic would be able to improve the best known value on objective z_j. In this case $\max_j \left\{ \lambda_i \left(z_j^0 - z_j \right) \right\}$ will be defined by the elements corresponding to the objectives with weights close to 0. Thus, objective z_j will stop influencing the value of the scalarizing function. In order to avoid this phenomenon the reference point is temporarily augmented by adding to each of its elements 10% of the range of each objective in the current set of potentially Pareto-optimal solutions.

The first approximation of the reference point is obtained by applying the local heuristic to optimization of each objective individually.

6.7. Normalization of objectives

Analogously to PSA the normalized weight vectors are applied to normalized objective function values. The normalization of objectives is performed in exactly the same way as in PSA (see section 5.2).

6.8. Updating the set of potentially Pareto-optimal solutions

The process of updating the set of potentially Pareto-optimal solutions is analogous to that used in PSA. We propose, however, to update this set only with the final solutions obtained with each run of the local heuristic. If the local heuristic is a standard local search another option would be to update this set with all the solutions accepted during the local search process or even with all the neighborhood solutions evaluated during local search. The two latter approaches would, however, increase the time spent on updating the set of potentially Pareto-optimal solutions. Our preliminary experiments indicated that the proposed approach does not reduce significantly the quality of the algorithm with respect to the versions that more often update the set of potentially Pareto-optimal solutions.

6.9. Using partial preference information

Analogously to PSA (see section 5.5) MOGLS may naturally handle partial preference information expressed in the form of constraints in the weight space. In this case the algorithm for generating random weight vectors should be modified to generate only vectors meeting these constraints. Also analogously to PSA it may handle preference information expressed in the form of constraints in the objective space by such techniques as penalty functions.

Another interesting possibility in the case of MOGLS is to modify the probability distribution of random weight vectors while keeping all of the weight vectors feasible. Assume for example, that the DM specifies the interesting region of the nondominated set in a vague form, e.g. as close to a given point or a set of points. In this case, the probability of drawing weight vectors yielding solutions in the interesting region could be increased. Of course, it may be difficult to know a priori which weight vectors correspond to the interesting region of the nondominated set. This may be performed during the run of the method by gathering the knowledge about the weight vectors, which were used in iterations that produced solutions in the interesting region.

6.10. Summary of the algorithm

The algorithm of multiple objective genetic local search is presented in Figure 20.

Parameters: K – size of the temporary elite population, S – number of initial solutions, stopping criterion
Initialization:
  The set of potentially Pareto-optimal solutions PP:=∅
  The current set of solutions CS:=∅
Generation of the first approximation of the ideal point:
  for each objective f_j
    Construct a new feasible solution **x** by a randomized algorithm
    Apply local heuristic optimizing objective f_j to solution **x** obtaining **x**'
    Add **x**' to the current set of solutions CS
    Update set PP with **x**'
Generation of the initial set of solutions:
  repeat
    Draw at random a weight vector Λ
    Construct randomly a new feasible solution **x**
    Apply local heuristic optimizing $s(\mathbf{z},...,\Lambda)$ to solution **x** obtaining **x**'
    Add **x**' to the current set of solutions CS
    Update set PP with **x**'
  until S initial solutions are generated or condition (1) is fulfilled
Main loop:
  repeat
    Draw at random a weight vector Λ
    From CS select K different solutions being the best on scalarizing function $s(\mathbf{z},...,\Lambda)$ forming temporary elite population TEP
    Draw at random with uniform probability two solutions $\mathbf{x}_1$, $\mathbf{x}_2 \in TEP$
    Recombine $\mathbf{x}_1$ and $\mathbf{x}_2$ obtaining $\mathbf{x}_3$
    Mutate $\mathbf{x}_3$
    Apply local heuristic optimizing $s(\mathbf{z},...,\Lambda)$ to solution $\mathbf{x}_3$ obtaining $\mathbf{x}_3$'
    if $\mathbf{x}_3$' is better on $s(\mathbf{z},...,\Lambda)$ than the worst solution in TEP and different in the decision space to all solutions in TEP **then**
      Add $\mathbf{x}_3$' to the current set of solutions CS
    Update set PP with $\mathbf{x}_3$'
  until the stopping criterion is met

Figure 20. The algorithm of multiple objective genetic local search – MOGLS. Function $s(\mathbf{z},...,\Lambda)$ stands for either $s_\infty\left(\mathbf{z},\mathbf{z}^{**}(PP),\Lambda\right)$, $s_c\left(\mathbf{z},\mathbf{z}^{**}(PP),\Lambda,\omega\right)$ or $s_l(\mathbf{z},\Lambda)$

7. Two-phase approach to interactive analysis of multiple objective combinatorial optimization problems

7.1. The two-phase approach

The goal of multiple objective optimization is to find the single solution giving the *best compromise* between multiple objectives (see section 1.1). Thus, generating a sample of solutions approximating the whole or a part of the nondominated set, in general, does not solve the original problem but is just a step in the search for the best compromise. Of course, in some cases the DM may be able to select the best compromise from the approximation of the nondominated set without any further support. This is especially easy in the bi-objective case, where graphical visualization in a two-dimensional plot is possible. In general, the DM analyzing a generated a priori large set of approximately Pareto-optimal solutions may, however, need some further support characteristic to interactive procedures in the search for the best compromise.

We have proposed a two-phase approach for solving multiple objective combinatorial optimization problems with the use of metaheuristics [80]. The approach is composed of the following two phases:

Phase 1. Generation of an approximation of the whole or a part of the nondominated set with a multiple objective metaheuristic.

Phase 2. Interactive analysis of the generated approximation.

Several interactive procedures for analysis of finite sets of alternatives that could be used in the second phase have been already proposed. This class of methods includes: Zionts method [185], Korhonen, Wallenius and Zionts method [107], Köksalan, Karwan and Zionts method [108], Korhonen method [105], Malakooti method [117], Taner and Köksalan method [162], AIM [114], Light Beam Search-Discrete [93] (see section 7.3), Interactive Trichotomy Segmentation [89] (see section 7.4) and Interquad [157]. Such methods could be used for interactive analysis of large sets of approximately Pareto-optimal solutions.

Hwang et al. [73] and Słowiński [150] proposed a classification of MOO methods which takes 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, a posteriori, or progressive (interactive) articulation of preferences. According to this classification the two-phase approach should be classified as an a posteriori articulation of preferences approach.

Note, however, that the above mentioned interactive procedures for the analysis of finite sets of alternatives are usually based on well-known interactive procedures for multiple objective mathematical programming that generate (approximately) Pareto-optimal solutions in computational phases interlacing with decision phases. In fact, the DM may not be even aware whether the solutions presented to him/her in decision phases were generated a priori or they are generated on-line in computational phases interlacing with phases of decision. Clearly, in both cases we deal with progressive articulation of preferences.

Consequently, we propose to distinguish two versions of interactive approach, taking into account the way of generating 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. On the contrary, in the *off-line approach* corresponding to the two-phase approach the (approximately) Pareto-optimal solutions are generated prior to the interactive analysis.

In the case of MOCO, both the above approaches may use metaheuristics. In the on-line approach it is possible to use a single objective metaheuristic applied to a substitute problem whose optimal solution corresponds to a Pareto-optimal solution of the original MOCO problem. The substitute problem may for example be optimization of a scalarizing function [179] or optimization of one of the objectives with constrained values of the remaining objectives (ε-constraints method [154], ch. 8.5). In the off-line approach multiple objective metaheuristics could be used.

The off-line generation of approximately Pareto-optimal solutions has several advantages compared with the on-line approach:

- It allows different types of statistical analysis, e.g. calculation of correlation between objectives, as well as 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, because no optimization is performed during the interactive process.
- It guarantees that all solutions presented to the DM are mutually nondominated. On the contrary, approximate solutions generated by a heuristic used within the on-line approach may dominate each other.

The issue of computational effectiveness of the off-line approach compared with the on-line approach is considered in section 7.2.

Note that the two versions of the interactive analysis of MOCO problems are not the only possible interactive approaches utilizing metaheuristics. Nabrzyski [125] proposed an interactive tabu search method in which the DM expresses his/her preferences periodically during the run of the method.

7.2. Computational effectiveness of the off-line generation of approximately Pareto-optimal solutions

In section 7.1 we have defined two different approaches to generation of approximately Pareto-optimal solutions with the use of metaheuristics – the on-line and the off-line approach. The latter approach is used within the two-phase approach to MOCO problems described in the same section. The off-line approach may seem to be a computationally ineffective, brute force technique because it generates many solutions that would not be interesting to the DM. On the contrary, the on-line approach uses partial preference information to generate approximately Pareto-optimal solutions interesting to the DM. In this section we describe a technique proposed in [83] that allows to compare the computational requirements of the two approaches.

As it was mentioned in section 7.1, the classical interactive methods assuming the on-line generation of (approximately) Pareto-optimal solutions may use different kinds of substitute problems to generate the solutions. Many of the methods use optimization of scalarizing functions, e.g. the Interval Criterion Weights method [153], the Interactive Weighted Tchebycheff method [155] and the Visual Interactive Approach [106]. In particular, optimization of weighted Tchebycheff scalarizing functions allows to generate all the Pareto-optimal solutions (see section 2). On the other hand, the interactive procedures for the analysis of finite sets of alternatives that could be applied after the off-line generation of approximately Pareto-optimal solutions also usually use scalarizing functions. In this case, however, the best solution on a scalarizing function is selected from the set of explicitly known solutions without optimization. Thus, in this section, we concentrate on the on-line and the off-line approaches that use scalarizing functions for selection of the approximately Pareto-optimal solutions presented to the DM.

The idea of the proposed evaluation technique is to use the average values of the scalarizing functions to compare the quality of solutions generated on-line and off-line. Then, we consider the question of computational requirements needed to achieve the same quality with the two approaches.

Let us 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. Let us denote solution obtained within the on-line approach, i.e. by optimization of s with a single objective metaheuristic, by $\mathbf{x}^s$. Let us 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 to compare the two solutions. For example, solution $\mathbf{x}^s$ is better than $\mathbf{x}^m$ if $s(\mathbf{x}^s) < s(\mathbf{x}^m)$.

Obviously, the comparison on a single scalarizing function is irrelevant. We propose to compare the two kinds of approaches on a set $S = \{s_1,...,s_L\}$ of randomly or systematically generated scalarizing functions (see e.g. the systematic approach described in section 4.2.3). 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 PP of potentially Pareto-optimal solutions generated by a multiple objective metaheuristic, i.e. $\forall \mathbf{x} \in PP\ s_l\left(\mathbf{x}^{ml}\right) \le s_l(\mathbf{x}),\ l = 1,...,L$ (we assume minimization of the scalarizing function). We consider the multiple objective metaheuristic not worse than the single objective metaheuristic if:

$$\sum_{l=1}^{L} \left(s_l\left(\mathbf{x}^{ml}\right) - s_l\left(\mathbf{x}^{sl}\right) \right) \le 0 . \tag{2}$$

If $L \rightarrow \infty$ the above condition means that the set PP of potentially Pareto-optimal solutions gives the same average quality of approximation over all scalarizing functions as 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 (2). Note that if the set of scalarizing functions is generated systematically the above condition may be interpreted as application of the expected value of the weighted Tchebycheff utility/scalarizing function measure $\hat{R}$ (see section 4.2.3) to both the solutions generated on-line and off-line.

If condition (2) is fulfilled, one 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 set *PP*. We define then the *effectiveness index EI*:

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

The lower *EI* the more effective the multiple objective metaheuristic in relation to the single objective method.

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

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

The effectiveness index has some clear interpretation from the interactive procedures perspective. Assume that the interactive process requires generation of *W* approximately Pareto-optimal solutions of a given problem in order to meet the stopping criteria or to reach the solution satisfying the DM. If $W > EI$ 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.

Note that the computational effectiveness of the on-line approach may be significantly improved if samples of solutions are generated in the computational phases instead of single solutions (see e.g. [171]). This could be achieved by using a multiple objective metaheuristic to generate a set of solutions coming from the interesting region of the nondominated set. Both PSA and MOGLS could be used in this way (see sections 5.5 and 6.9).

7.3. Interactive Light Beam Search method for discrete alternative set multiple criteria problems

7.3.1. Background

The Light Beam Search (LBS) procedure was proposed several years ago by Jaszkiewicz and Słowiński [92] to deal with multiple-objective analysis of linear and non-linear problems. Since that time it went through several modifications imposed by applications to real-life problems. In particular, a version of the method for discrete alternative multiple criteria choice problems with large sets of explicitly known alternatives called Light Beam Search-Discrete (LBS-discrete) has been proposed [93]. Although the method is a general tool for interactive analysis of discrete alternative problems, the main goal of its development was its use within the two-phase approach described in section 7.1. In that case, the set of alternatives contains potentially Pareto-optimal solutions generated by a metaheuristic method.

A review of the development of interactive procedures shows an increasing interest in techniques, which present to the DM a sample of nondominated points instead of a single point in each decision phase. This is motivated by the willingness to present richer information taking into account some elements of arbitrariness inherent to the definition of preferential information guiding the search for a new proposal in the computation phase. The class of interactive procedures characterized by iterative presentation of samples of nondominated points includes such known representatives for both continuous and discrete problems as: the Interval Criterion Weights method [153], the Choo-Atkins method [17], the Zionts-Wallenius method [186], the Interactive Weighted Tchebycheff method [155], the Interactive Weighted-Sum method [154], the Visual Interactive Approach [106], the Jacquet-Lagrèze, Meziani and Słowiński's method [76], the Search Beam method [5], the Steuer, Silverman and Whishman's method [156], AIM [114] and Interquad [157]. In each decision phase, the DM is expected to supply a preferential information based on multiple-criteria comparison of the presented points; it often boils down to specification of the best point in the sample. Thus, in all these interactive procedures, the DM is expected to solve, in each decision phase, an MCDM subproblem with an explicitly given set of alternatives. The interactive procedures mentioned above are based on an assumption, usually not declared, that pairwise comparison of an explicitly given set of alternatives can be done by the DM without any further support.

Practical experience shows, however, that evaluation of an even small explicitly given set of alternatives in the presence of multiple criteria can be difficult to the DM. Numerous psychological studies indicate the highly limited capability of the DMs in processing multi-criteria information [97]. Taking into account the results of the studies, Larichev et al. [111] have concluded that the selection of the best solution from a finite sample is one of the most psychologically difficult operations.

LBS is based on the assumption that some elements of the methodology proposed for MCDM problems with an explicitly given set of alternatives could be used to help the DM in comparison of the solutions presented in the decision phase. The well-known representatives of this methodology are Multiple Attribute Utility Theory [98] and the family of Outranking methods [140], [141]. It seems that the preference models built and exploited in these methods could be used to support the DM in making the local multiple criteria comparisons.

As it was mentioned is section 2 the outranking relation preference model allows for more realistic preference modelling taking into account four conflicting situations of indifference, strict preference, weak preference and incomparability. We should emphasize, however, that in practical decision aiding, the DM is **not** asked to judge explicitly the truth of relation S for each pair of alternatives. Instead, the outranking relation is constructed on the set of alternatives, using the available preference information, and then exploited in order to give answers pertinent to the decision problematic (see [141]).

In order to ease the comparison of points in the decision phase of an interactive procedure, the sample of points presented for the examination should meet some requirements. Specifically, the points in the sample should not be incomparable for the DM and also they should not be all indifferent for him/her. A "forced" comparison of points which do not meet these requirements can yield serious incoherence in the preferential information supplied by the DM. Furthermore, in the case of indifferent or incomparable points in the sample, the DM may stop the interactive procedure being unable to find a better proposal among the presented points even if the current point is far from the best compromise.

The above observations led to propose an interactive method distinguished by the following features:

- the search over the nondominated set is organized through sampling of a neighborhood of the current point which moves in result of either a change of the DM's aspiration point or a shift of the current point within its neighborhood,
- the outranking relation is used as a local preference model in a neighborhood of the current point; its construction is based on relatively weak preferential information,
- the neighborhood of the current point is composed of nondominated points that outrank this point, so the neighborhood includes points that are comparable with the current point; the points from outside the neighborhood are either incomparable with or are outranked by the current point.

These features submit some analogy with projection of a focused beam of light from a spotlight at the aspiration point onto the nondominated set. For this reason the procedure is called the Light Beam Search or, shortly, LBS (see Figure 21).

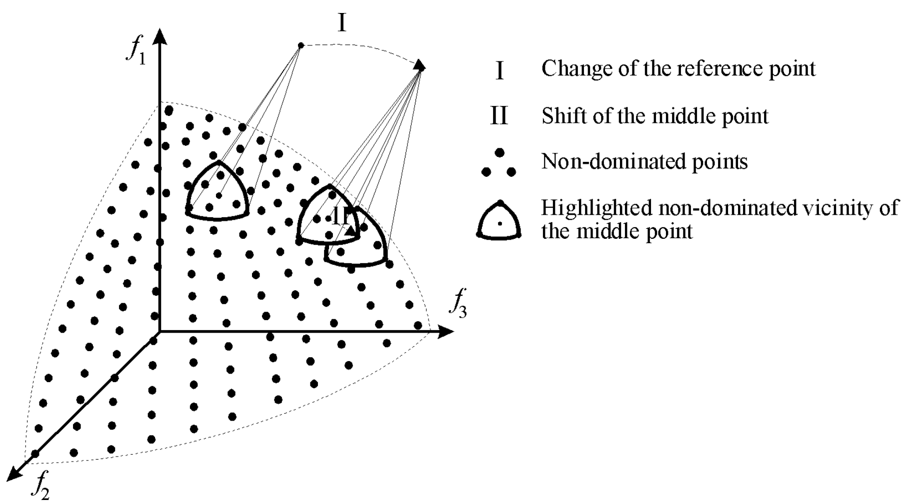

Figure 21. The Light Beam Search over a nondominated set

The outranking relation has also been used as a local preference model in the Aspiration-Level Interactive Model [114]. The method does not use, however, any preferential information to define of the neighborhood.

7.3.2. Description of the method

In each computation phase of LBS-discrete a finite sample of nondominated points and corresponding solutions is selected and presented to the DM. The sample is composed of the current point, called the *middle point*, obtained in a previous iteration, and a number of nondominated points from its neighborhood. In order to define the neighborhood, the outranking relation S is used as a local preference model. Precisely, for the current middle point, the neighborhood is defined as a set of nondominated points that are **not worse** than the middle point, i.e. outrank the middle point. The neighborhood is also called the *outranking neighborhood* of the middle point.

```
Present to the DM the ideal and nadir point in the analyzed set of points
Ask the DM to specify the starting aspiration point
Compute the starting middle point
Ask the DM to specify local preferential information used to build the outranking relation
repeat
    Present the middle point to the DM
    Find the neighbors of the middle point and present them to the DM
    if the DM wants to store the middle point then
        Add it to the set of stored points
    case
        The DM wants to define a new aspiration point:
            Ask the DM to specify the point
            Project the aspiration point onto the nondominated set
        The DM wants a point from the neighborhood to be the new
        middle point:
            Ask the DM to indicate the new middle point
        The DM wants to return to one of the stored points:
            Use an indicated stored point as a new middle point
        The DM wants to update the preferential information:
            Ask the DM to specify the new preferential information
    end
until the DM feels satisfied with a point found during the interactive process or
concludes that no acceptable compromise point exists in the set of available
points
```

Figure 22. The algorithm of Light Beam Search-Discrete

In general, in the outranking neighborhood, apart from the points, which outrank the middle point, there are also points outranked by the middle point. According to the local preference model, these points are *indifferent* to the middle point while the remaining points of the neighborhood are *preferred* to the middle point.

The algorithm of LBS-discrete is presented in Figure 22:

The starting middle point $\mathbf{z}^C$ is obtained by projecting the aspiration point $\mathbf{z}^r$ onto the nondominated set. By projection onto the nondominated set we mean optimization of the achievement scalarizing function with the given aspiration point used as reference point on the set of available points. The problem is solved not by optimization but by browsing the set of available points for the one with the minimum value of the scalarizing function.

Then, the local preferential information given by the DM is used to define the outranking neighborhood around point $\mathbf{z}^C$. This information consists of indifference thresholds (q_j) and, optionally, preference (p_j) and veto (v_j) thresholds on particular objectives (j=1,..,J). In general, all the thresholds are functions of z_j, however, as the outranking relation is used as the local preference model, a constant value of each threshold is sufficient for a given middle point. Moreover, $p_j \geq q_j \geq 0$.

The indifference and preference thresholds allow to distinguish between the three following preference relations with respect to z_j, for any ordered pair ($\mathbf{z}^a$, $\mathbf{z}^b$) of points (maximization of objectives is assumed):

$\mathbf{z}^a \, I_j \, \mathbf{z}^b$, i.e. $\mathbf{z}^a$ and $\mathbf{z}^b$ are *indifferent* $\Leftrightarrow$ $-q_j \le z_j^a - z_j^b \le q_j$,

$\mathbf{z}^a \, Q_j \, \mathbf{z}^b$, i.e. $\mathbf{z}^a$ is *weakly preferred* to $\mathbf{z}^b$ $\Leftrightarrow$ $q_j < z_j^a - z_j^b \le p_j$,

$\mathbf{z}^a \, P_j \, \mathbf{z}^b$, i.e. $\mathbf{z}^a$ is *strictly preferred* to $\mathbf{z}^b$ $\Leftrightarrow$ $p_j < z_j^a - z_j^b$.

The veto threshold v_j allows to take into account the possible difficulties of comparing the relative value of two alternatives when one is significantly better than the other on a subset of objectives, but much worse on at least one other objective. By definition, the assumption that $\mathbf{z}^a \, S \, \mathbf{z}^b$ has to be denied if

$$z_j^b \ge z_j^a + v_j$$

for at least one $j \in \{1,...,J\}$. This situation is called *veto preference* of $\mathbf{z}^b$ over $\mathbf{z}^a$.

Let us notice that the preferential information does not include the weights of objectives which might be too difficult to define; it is assumed, however, that one objective is not more important than all the other together.

In order to compute the neighborhood of the middle point $\mathbf{z}^C$ one has to first specify how to test whether a point $\mathbf{z}$ outranks the middle point $\mathbf{z}^C$. To this end, the following numbers are calculated:

$m_v(\mathbf{z}^C, \mathbf{z}) = \text{card}\ \{j\text{:}\ z_j^c - z_j > v_j\ , j = 1,..,J\}$,

$m_p(\mathbf{z}^C, \mathbf{z}) = \text{card}\ \{j\text{:}\ z_j^c - z_j > p_j\ , j = 1,..,J\}$,

$m_q(\mathbf{z}^C, \mathbf{z}) = \text{card}\ \{j\text{:}\ z_j^c - z_j > q_j\ , j = 1,..,J\}$,

$m_>(\mathbf{z}^C, \mathbf{z}) = \text{card}\ \{j\text{:}\ z_j^c - z_j > 0\ , j = 1,..,J\}$,

$m_>(\mathbf{z}, \mathbf{z}^C) = \text{card}\ \{j\text{:}\ z_j - z_j^c > 0\ , j = 1,..,J\}$.

The following definition of the outranking relation is proposed:

$$\mathbf{z}\, S\, \mathbf{z}^c \Leftrightarrow \begin{cases} \text{either} \quad \begin{cases} m_v\left(\mathbf{z}^c, \mathbf{z}\right) = 0 \text{ and} \\ m_p\left(\mathbf{z}^c, \mathbf{z}\right) = 1 \text{ and} \\ m_>\left(\mathbf{z}^c, \mathbf{z}\right) = 1 \end{cases} \quad \text{or} \quad \begin{cases} m_v\left(\mathbf{z}^c, \mathbf{z}\right) = 0 \text{ and} \\ m_p\left(\mathbf{z}^c, \mathbf{z}\right) = 0 \text{ and} \\ m_q\left(\mathbf{z}^c, \mathbf{z}\right) < m_>\left(\mathbf{z}, \mathbf{z}^c\right) \end{cases} \end{cases}$$

According to this definition, in order to accept $\mathbf{z}\, S\, \mathbf{z}^c$, we exclude veto preference of $\mathbf{z}^c$ over $\mathbf{z}$ on any objective, however, if there is one strict preference of $\mathbf{z}^c$ over $\mathbf{z}$ then all other objectives should be in favor of $\mathbf{z}$. Furthermore, if there is no strict preference of $\mathbf{z}^c$ over $\mathbf{z}$ then weak preference of $\mathbf{z}^c$ over $\mathbf{z}$ on some objectives may be compensated by the existence of a greater number of objectives in favor of $\mathbf{z}$.

We allow that the preference and veto thresholds are not specified by the DM on some or all objectives. If the preference threshold is not specified for objective j, then indifference threshold q_j defines the limit between indifference I_j and strict preference P_j, i.e. we assume that $p_j = q_j$. If the veto threshold is not specified for objective j, then veto preference is identified with strict preference, i.e. we assume that $v_j = p_j$. In conclusion, if neither p_j nor v_j is specified, we assume that $v_j = p_j = q_j$.

The outranking neighborhood is obtained by testing the conditions of the outranking relation for all available points. The outranking neighborhood can be empty if no available point outranks the middle point.

In the decision phase, the middle point and its neighbors are presented to the DM. Each point of the neighborhood presented to the DM is qualified as preferred or indifferent to the middle point.

The procedure stops when one of the presented points is satisfactory to the DM on all objectives. Otherwise, he/she can continue the procedure by either changing the aspiration point or by shifting the middle point to another point from its neighborhood.

Before continuing the procedure, the DM can store the current middle point. He/she is also allowed to restore any of the stored points at any iteration if he/she decides that the stored point gives a better compromise than the current point.

Finally, the DM is allowed to modify the preferential information given for each objective, i.e. the indifference, preference and veto thresholds.

7.3.3. Comparison with related methods

In LBS, the DM has two possibilities, or degrees of freedom, to control the search over the nondominated set. The first degree permits specification of his/her aspiration levels defining the aspiration point. The aspiration point is then projected onto the nondominated set. Similar degree of freedom is given in the Interactive Goal Programming [39], the Naive method [13], [15], the Reference Point method [179] and the Satisficing Trade-Off method [125]. These procedures belong to the class of learning-oriented interactive procedures [172]. Although they guarantee free search over the entire nondominated set, they do not give any substantial support to the DM in the decision phases.

The second degree of freedom permits the selection of a new middle point from a finite sample of nondominated points. A similar possibility is given to the DM in: the Interval Criterion Weights method [153], the Choo-Atkins method [17], the Zionts-Wallenius method [186], the Interactive Weighted Tchebycheff method [155], the Interactive Weighted-Sum method [154], the Jacquet-Lagrèze, Meziani and Słowiński's method [76] and the Steuer, Silverman and Whishman's method [156]. LBS differs from these methods in the way in which the sample of points is constructed. While in the methods mentioned above, the sample is constructed in a way that ensures a "convergence" of the procedure, in LBS the sample is constructed in a way that ensures a feasibility of comparison of the points from the sample.

By combining the two degrees of freedom in a single interactive procedure, the DM is given some advantages of methods using only one of the degrees. The specification of the aspiration point gives the DM substantial control over the procedure. The DM can freely scan the entire nondominated set, go back and change previous decisions, or to make "big jumps" to other regions of the nondominated set. The presentation of the outranking neighbors of the middle point supports the DM in learning about the problem and finding a new solution better fitting his/her preferences. Moreover, because one can assume that the DM would not stop the interactive process if a point from the outranking neighborhood is found better than the middle point, LBS shares the property of "intuitive convergence" (cf. [154]) with such methods like the Interval Criterion Weights, the Interactive Weighted Tchebycheff and the Interactive Weighted-Sum.

The idea of combining the two degrees of freedom in a single interactive procedure is also supported by the results of experimental comparisons of some existing interactive procedures [9], [12], [14], [15], [32] and [176]. In the experiments made by Buchanan and Daellenbach [15], by Buchanan [12] and Buchanan and Corner [14], the Naive method and the Interactive Weighted Tchebycheff method were tested among others. As it was mentioned above, each of them organizes the interactive process giving the DM one of the two degrees of freedom used in LBS. In these experiments, participants solved some multiple objective linear programming problems with each interactive procedure. The Naive method and the Interactive Weighted Tchebycheff method were clearly preferred by the participants over the methods using utility functions to ensure a formal convergence. The Interactive Weighted Tchebycheff method was found easy to understand, easy to use and giving high confidence. Some participants, however, found it difficult to assimilate all the information contained in the presented solutions over multiple objectives. Moreover, as Buchanan [12] reports, “Initially they found it difficult to choose among six very different solutions and, as the procedure converged, they found it even more difficult to choose among six very similar solutions. There was an expressed desire among a number of participants to place bounds on some objectives, principally to limit the size of the solution space”. The Naive method was also evaluated high, some participants, however, disliked the procedure for the lack of support in the search for the best compromise. The Naive method moved quickly to the locality of the final solution, but the refining process of moving closer to the best compromise was somewhat haphazard. Buchanan [12] says that the participants “sometimes appeared to be anchored by solutions they found in the first two or three iterations [...]. The logic appeared to be 'well, I got what I guessed; I suppose I should stop' ”. In conclusion, a need was expressed for a combined procedure in which a neighborhood of the current solution would be sampled instead of the entire nondominated set.

Let us remark that in LBS, the difficulty appearing in comparison of very different or very similar solutions observed in the Interactive Weighted Tchebycheff method is eliminated due to comparability condition imposed on the neighborhood region. Moreover, the risk of anchoring observed in the Naive method is reduced in LBS due to presentation of a number of neighborhood points in each iteration.

Although combination of the two degrees of freedom gives LBS some advantages over other interactive procedures, we do not claim that it should be used to solve all kinds of problems. The willingness of a hypothetical DM to work with LBS could be based on the following features of the problem:

- The existence of incomparability in the nondominated set. The incomparability of some points occurs, for example, if some of the objectives are in strong conflict and the DM refuses to accept unlimited substitution between objectives. Strong mutual conflicts between objectives are typical for situations in which different criteria represent aspirations of different groups of interest. So, although LBS does not directly support group decision making, it can be a useful tool for a single DM operating under the pressure of different lobbies.
- The existence of indifference in a close neighborhood. It can be caused by a limited ability of the DM to perceive small differences as well as by imprecision and uncertainty of data and/or of the mathematical model of the decision problem.
- The number of objectives between three and nine. A larger number of objectives is beneficial for the construction of the outranking relation which requires that none ob-

jective is more important than all the others together. However, if the number of objectives exceeds nine, it is commonly acknowledged that the comparison of nondominated points by the DM gets too difficult.

- Incomplete knowledge about the problem pushing the DM to learn more about it during the interactive process. LBS supports the learning process by presenting samples of points rather than a single point in each iteration.

Taking into account the importance of support given to the DM, LBS requires relatively weak preferential information from him/her.

7.4. Interactive Trichotomy Segmentation

7.4.1. Introduction

This section describes a method for multiple criteria choice problems with an explicitly given but relatively large set of alternatives proposed by Jaszkiewicz and Ferhat [89]. Analogously to LBS-discrete the main goal of its development was its use within the two-phase approach described in the section 7.1. The method can be placed between methods with a priori specification of the DM's preferences and interactive procedures. The method includes some interaction with the DM, but is based on an assumption that at the beginning of the solution process he/she already has relatively well defined preferences. The DM's preferences are modeled with an outranking relation in the neighborhood of an aspiration point in the nondominated set. This preference model is used in order to define a trichotomy of the nondominated set in terms of accepting good alternatives, rejecting non-interesting ones and defining alternatives that can neither be accepted, nor rejected based on the available preference information. The method stops when the class of good alternatives is sufficiently small and the DM is able to select one of them as the best compromise. Otherwise, the DM can modify the preferential information by changing either the aspiration point, or the thresholds used to build the outranking relation. In other words, the DM defines the three categories dynamically.

As it was mentioned in section 7.3.3, specification of DM’s aspiration levels defining the aspiration point is used in many interactive methods. The aspiration point is then used as the reference point in a scalarizing function, which optimized on the feasible set, gives a single nondominated point. The scalarizing functions include, however, some other parameters, e.g., the weight vector, that are set by the method. By changing the parameters one can obtain different alternatives for the same aspiration point. So, the choice of a single alternative presented to the DM in a given iteration is rather arbitrary. In order to find the best compromise point the DM may be forced to change the aspiration point even if his/her preferences are stable. In such case, the aspiration point starts to play a technical role. It becomes a tool that allows the DM to control the search for the best compromise. The methods, seems to be most appropriate when the DM is interested in learning about the problem by free scanning of the nondominated set (cf. [172]).

In order to avoid the above drawback LBS method described in the previous section presents to the DM a sample of points coming from the outranking neighborhood of the point obtained by optimizing the scalarizing function. The Trichotomy Segmentation method applies the outranking relation directly to the aspiration point. The aspiration point is used in order to characterize a subset of alternatives being not worse than the aspiration point – category C^+. The remaining alternatives are classified as either worse or incomparable to the aspiration point – categories C^- and $C^?$, respectively. The outranking relation, based on additional preference information supplied by the DM, is used in order to compare nondominated points to the aspiration point $\mathbf{z}^r$. The outranking relation allows to take into account meaninglessness of small differences on criteria for the DM as well as difficulties in comparing points having significantly different scores on particular objectives. So, the procedure allows the DM to concentrate on the most interesting region of the nondominated set but does not select a single point arbitrarily.

A trichotomic segmentation approach has already been used by Moscorola and Roy [122] and Yu [184] (see also [141]). Their methods, however, concern the multiple criteria sorting problems and, furthermore, are based on a priori specification of DM's preferences, while the proposed method assumes interaction.

7.4.2. Description of the method

The algorithm of interactive trichotomy segmentation is presented in Figure 23.

The method starts by presenting to the DM the ideal z^* and the nadir z_* points. In the next step, the starting aspiration point $\mathbf{z}^r$ is specified by the DM. Then, the DM is asked to give the preferential information for each criterion (i.e., the indifference and, optionally, the preference and veto thresholds). At this stage, the DM should decide whether he/she wants to specify the preference and/or veto thresholds, however, he/she is allowed to change these settings at each step of the procedure. The definition of the outranking relation is the same as in the Light Beam Search.

In the computation phase of the procedure, the three categories C^+, C^- and $C^?$ are calculated. Each available point is compared to the aspiration point $\mathbf{z}^r$, the ones which outrank $\mathbf{z}^r$ are classified into the acceptance category C^+, corresponding to the sub-region containing points which are not worse than the aspiration point, i.e., outrank the point. The ones, which are outranked by $\mathbf{z}^r$ and do not outrank $\mathbf{z}^r$ are classified into the rejection category C^-, and the ones which are incomparable to $\mathbf{z}^r$ are classified into the incomparability category $C^?$.

In the decision phase, if the acceptance category C^+ is not empty, then it is presented to the DM. If he/she thinks that the number of points belonging to this category is too large, it is possible to present him/her the so-called *characteristic points* of the acceptance category and the *locally ideal* point $\bar{\mathbf{z}}^*$. The characteristic points are points $\mathbf{z}^j, j = 1,...,J$ that maximize particular criteria within this category. The locally ideal point $\bar{\mathbf{z}}^*$ is defined by the characteristic points: $\bar{z}_j^* = z_j^j,\ j = 1,...,J$.

Find the ideal z^*, and the nadir z_* points; present them to the DM.
Ask the DM to specify an aspiration point $\mathbf{z}^r$.
Repeat
Get the (updated) preferential information (the indifference and optionally the preference and/or veto thresholds).
Calculate the three categories (C^+, C^- and $C^?$).
If {$C^+ \neq \emptyset$} **Then**
Present to the DM points belonging to this category.
If {$C^? \neq \emptyset$} **Then**
Present (optionally) to the DM points incomparable to the aspiration point which are contained in category $C^?$ with possibility of presenting only a given number of representative points.
If {$C^- \neq \emptyset$} **Then**
Present (optionally) to the DM points belonging to this category.
If the DM wants to store the aspiration point **Then**
Add it to the set of "stored points".
If the DM cannot select the final solution from C^+ **Then** he/she can either:

- attempt to reduce the number of points belonging to C^+ by defining an optimistic aspiration point, i.e., specifying a new point dominating the current one,
- attempt to increase the number of points belonging to C^+ by defining a pessimistic aspiration point, i.e., specifying a new point dominated by the current one,
- define a new aspiration point nondominated with respect to the current one in order to scan a new nondominated region,
- update the preferential information used to build the outranking relation,
- return to one of the stored points and consider it as the aspiration point.

Until the DM feels satisfied with a point found during the interactive process or concludes that no acceptable compromise point exists in the set of available points

Figure 23. The algorithm of Interactive Trichotomy Segmentation

The DM can also see the points belonging to the incomparable category $C^?$. Presentation of incomparable points can help the DM in defining a new aspiration point nondominated with respect to the previous one. If the number of points in $C^?$ is too large, the DM may specify the number of representative points that he/she would like to see. Selecting a given number of the most representative points consists in solving the maximal dispersion problem (cf. [182]). Because finding the optimal solution of this problem is prohibitively time consuming, it is advised to use one of the heuristic filtering techniques [154], ch. 11, [165]).

Optionally, the DM can see the rejection category C^-.

If the acceptance category C^+ is not empty but contains too many points, the DM has the possibility to interactively define an optimistic aspiration point by increasing successively or simultaneously the scores of the aspiration point $\mathbf{z}^r$. Otherwise, if the category C^+ is empty or contains too few points, the DM has, as above, the possibility to interactively define a pessimistic aspiration point by decreasing successively or simultaneously the scores of the aspiration point $\mathbf{z}^r$. Moreover, he/she can scan another sub-region of the nondominated set by defining another aspiration point nondominated with respect to the current one.

The procedure stops if one of the points presented is satisfactory for the DM on all criteria.

Before continuing the scanning, the DM can store the aspiration point. He/she is allowed to go back to one of the stored points at any time.

Finally, the DM is able to modify the preferential information given for each criterion, i.e., the values of indifference, preference and veto thresholds.

The willingness of a hypothetical DM to work with the Interactive Trichotomy Segmentation could be defined analogously to LBS. The difference is that the user of Interactive Trichotomy Segmentation is assumed to have a relatively good idea of the desired values of objectives and is assumed to be less interested in learning about the problem. Such situation is typical to problems solved regularly with similar, but not completely equal, data. An exemplary application of this kind is described in section 10.1.

7.5. Summary

In this section, a methodology for interactive analysis of multiple objective combinatorial optimization problems has been presented. The methodology assumes off-line generation of approximately Pareto-optimal solutions prior to the interactive analysis. In result we obtain a two-phase approach to interactive analysis of MOCO problems. The advantages of the approach has been discussed in section 7.1. In section 7.2, we have presented an approach that allows to evaluate computational effectiveness of the off-line generation of the approximately Pareto-optimal solutions. Results of the experiment reported in section 9.2 illustrate that off-line generation may be computationally effective approach.

In sections 7.3 and 7.4 two interactive methods that may be used in the second phase of the two-phase approach have been presented. Both the methods use an outranking relation to model local preferences of the DM. In the Light Beam Search method the outranking relation is used to compare nondominated points to the middle point obtained by projection of the aspiration point on the nondominated set. In the Interactive Trichotomy Segmentation method the outranking relation is used to compare nondominated points directly to the aspiration point. The use of the Light Beam Search method is suggested when the DM is interested in free search over the nondominated set. The method allows the DM to control the search process by specifying the aspiration point. On the other hand, in each iteration it suggest the DM some alternative points form the neighborhood of the current point. Thus, the method should be used if the DM has incomplete knowledge about the problem pushing him/her to learn more about it during the interactive process.

The Interactive Trichotomy Segmentation method assumes that the DM has a relatively good idea of the desired values of objectives. In this case, the DM may be unwilling to use the aspiration point as a technical tool allowing him/her to control the free search over the nondominated set. The DM may be rather interested in a precise definition of the that points he/she considers not worse than the reference point.

8. Extensions of Pareto simulated annealing and Light Beam Search for fuzzy combinatorial optimization problems

Real-life multiple objective combinatorial optimization problems often include many parameters whose values are not known precisely. The lack of historical data as well as the nature of the source of the imprecision, which is often related to human factors, in many cases do not allow probabilistic modeling of the imprecision. In such case, possibilistic approach can be used, in which uncertain parameters are modeled with the use of fuzzy numbers.

The direct motivation for the research presented in this section is our rich experience in modeling and solving project scheduling problems under uncertainty concerning time parameters of activities [55]. Such problems appear for example in software project scheduling. The uncertainty of the time parameters is caused by the very high innovation rate in the software industry as well as the fact that most software production tasks are human-intensive. In practice, it appears that software project managers are often able to estimate approximate duration of particular activities. They base the estimations on their experience as well as their good knowledge about the new techniques and the skills of particular engineers. They can express intervals of possible duration together with the most possible values and decreasing possibilities of other values within the intervals. Such information about activity duration can be naturally expressed by a fuzzy number. This results in formulation of multiple objective fuzzy project scheduling problems. The problems belong to the class of fuzzy multiple objective combinatorial optimization problems whose deterministic equivalents are NP-hard. Due to this complexity, it is advisable to use a metaheuristic procedure for solving (approximately) these problems.

Before developing a metaheuristic approach for the fuzzy MOCO problems, we formulated the following postulates it should fulfil [61]:

- it should not aggregate a priori the fuzzy objective functions into a single objective but it should generate a representative approximation of the nondominated set for further exploration using an interactive procedure,
- it should consider nondominated points in the original fuzzy objective space and not nondominated points in the objective space of an auxiliary deterministic problem (no defuzzification).

The first postulate is satisfied by the use of the two-phase approach described in section 7.1. In order to take into account the second postulate fuzzy versions of Pareto simulated annealing and Light Beam Search have been developed [57], [58], [60], [61]. Before describing the two methods we introduce some basic concepts of the possibilistic approach.

8.1. Basic concepts and definitions

8.1.1. Uncertainty modeling

Set A in a base set X can be described by a membership function $\mu_A : X \rightarrow \{0,1\}$ with $\mu_A(x)=1$ if $x\in A$ and $\mu_A(x)=0$ if $x\notin A$. If it is uncertain, whether or not element x belongs to set A, the above model can be extended such that the membership function maps into the interval <0,1>. A high value of the membership function implies high possibility while a low value a poor possibility. This leads to the definition of a fuzzy set [30].

Let X be a base set and μ_A a function from X into the unit interval <0,1>. Then the set

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) \mid x\in X\}$$

is called a fuzzy set in X and $\mu_{\tilde{A}}$ is called the membership function of $\tilde{A}$.

Another useful definition is that of a level set or a level cut. Let $\tilde{A}$ be a fuzzy set in X and $\alpha\in(0,1>$. The α-level set or α-level cut of $\tilde{A}$ is the set

$$\tilde{A}_\alpha = \{ x \in X \mid \mu_{\tilde{A}}(x) \geq \alpha \}$$

Lower and upper bounds of α-cut $\tilde{A}_\alpha$ are equal to $\inf_{x\in X} \tilde{A}_\alpha$ and $\sup_{x\in X} \tilde{A}_\alpha$, respectively.

Flat fuzzy number is a special kind of a fuzzy set. Precisely, a flat fuzzy number is defined as a normalized convex fuzzy subset of real line R**,** i.e. it is defined according to (1), where $X=\Re$.

The precise form of a fuzzy number can be rarely described by an expert. A practical way of getting suitable membership functions of fuzzy data has been proposed by Rommelfanger [134]. The author proposes that the expert expresses his/her optimistic and pessimistic information about parameter uncertainty on some prominent membership levels α, e.g.:

α=1:	$\mu(x) = 1$	means that value x certainly belongs to the set of possible values,
α=λ:	$\mu(x) > \lambda$	means that the expert estimates that value x with $\mu(x) \geq \lambda$ has a good chance of belonging to the set of possible values,
α=ε:	$\mu(x) < \varepsilon$	means that value x with $\mu(x) < \varepsilon$ has only a very little chance of belonging to the set of possible values, i.e. the expert is willing to neglect the corresponding values of x with $\mu(x) < \varepsilon$.

Thus a flat fuzzy number in six-point convention is represented by six real numbers. Formally, a flat fuzzy number $\tilde{M}$ is represented by the following list of symbols:

$$\tilde{M} = (\underline{m}^{\varepsilon}, \underline{m}^{\lambda}, \underline{m}, \overline{m}, \overline{m}^{\lambda}, \overline{m}^{\varepsilon})$$

and its graphical representation is piecewise-linear.

We assume that λ = 0.6, ε = 0.1. An exemplary flat fuzzy number in the six-point convention is presented in Figure 24.

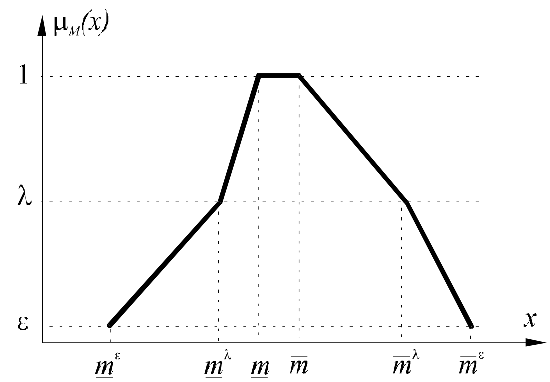

Figure 24. A flat fuzzy number $\tilde{M}$ in six-point representation

It is worth pointing out that the well-known trapezoidal and triangular fuzzy numbers can be modeled in this representation as well.

8.1.2. Fuzzy arithmetic

Let $\tilde{A}$ and $\tilde{B}$ be fuzzy numbers of the universe X and Y. Let * denote any basic arithmetic operation (+, −, ×, /). Then any operation $\tilde{A} * \tilde{B}$ can be described by Zadeh's extension principle:

$$\mu_{\tilde{A}*\tilde{B}}(z) = \max_{x*y=z} \left\{ \min\,[(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(y)] \right\}$$

In the case of arithmetic operations on fuzzy numbers in piecewise-linear representation the above equation corresponds to

$$\tilde{C}_{\alpha} = (\tilde{A} * \tilde{B})_{\alpha} = \tilde{A}_{\alpha} * \tilde{B}_{\alpha}, \text{ for any } \alpha \in (0, 1].$$

The above equation shows that the α-cut on a general arithmetic operation on two fuzzy numbers is equivalent to the arithmetic operation on the respective α-cuts of the two fuzzy numbers. Both $(\tilde{A} * \tilde{B})_{\alpha}$ and $\tilde{A}_{\alpha} * \tilde{B}_{\alpha}$ are interval quantities so the classical interval analysis [137] can be used for performing operations on them. Thus one can define the following arithmetic operations for fuzzy numbers in the six-point representation:

$$\tilde{A} \oplus \tilde{B} = \left(\underline{a}^{\varepsilon} + \underline{b}^{\varepsilon},\ \underline{a}^{\lambda} + \underline{b}^{\lambda},\ \underline{a} + \underline{b},\ \overline{a} + \overline{b},\ \overline{a}^{\lambda} + \overline{b}^{\lambda},\ \overline{a}^{\varepsilon} + \overline{b}^{\varepsilon}\right)$$

$$\tilde{A} \ominus \tilde{B} = \left(\underline{a}^{\varepsilon} - \overline{b}^{\varepsilon},\ \underline{a}^{\lambda} - \overline{b}^{\lambda},\ \underline{a} - \overline{b},\ \overline{a} - \underline{b},\ \overline{a}^{\lambda} - \underline{b}^{\lambda},\ \overline{a}^{\varepsilon} - \underline{b}^{\varepsilon}\right)$$

$$\tilde{A} \times \tilde{B} = \left(\underline{a}^{\varepsilon} \times \underline{b}^{\varepsilon},\ \underline{a}^{\lambda} \times \underline{b}^{\lambda},\ \underline{a} \times \underline{b},\ \overline{a} \times \overline{b},\ \overline{a}^{\lambda} \times \overline{b}^{\lambda},\ \overline{a}^{\varepsilon} \times \overline{b}^{\varepsilon}\right)$$

$$\tilde{A} / \tilde{B} = \left(\underline{a}^{\varepsilon} / \overline{b}^{\varepsilon},\ \underline{a}^{\lambda} / \overline{b}^{\lambda},\ \underline{a} / \overline{b},\ \overline{a} / \underline{b},\ \overline{a}^{\lambda} / \underline{b}^{\lambda},\ \overline{a}^{\varepsilon} / \underline{b}^{\varepsilon}\right)$$

and moreover:

$\max(\tilde{A}, \tilde{B}) = (\max(\underline{a}^{\varepsilon}, \underline{b}^{\varepsilon}),\ \max(\underline{a}^{\lambda}, \underline{b}^{\lambda}),\ \max(\underline{b}, \underline{a}),\ \max(\overline{b}, \overline{a}),\ \max(\overline{b}^{\lambda}, \overline{a}^{\lambda}),\ \max(\overline{b}^{\varepsilon}, \overline{a}^{\varepsilon}))$

$\min(\tilde{A},\ \tilde{B}) = (\min(\underline{a}^{\varepsilon}, \underline{b}^{\varepsilon}),\ \min(\underline{a}^{\lambda}, \underline{b}^{\lambda}),\ \min(\underline{b}, \underline{a}),\ \min(\bar{b}, \bar{a}),\ \min(\bar{b}^{\lambda}, \bar{a}^{\lambda}),$
$\min(\bar{b}^{\varepsilon}, \bar{a}^{\varepsilon}))$

8.1.3. Comparison of fuzzy numbers

The definitions of dominance and combinatorial search in a fuzzy objective space require comparing fuzzy scores on objective functions. The way of comparing fuzzy scores is related to the semantics of fuzzy numbers. Indeed, the fuzzy numbers can be seen as imprecise probability distributions [29], [143]. In this perspective, the comparison of two fuzzy numbers can be substituted by the comparison of their mean values defined consistently with the well-known definition of expectation in probability theory. The idea exploited by Dubois and Prade [31] relies on the mathematical fact that, with respect to a fuzzy number, the possibility measure corresponds to an upper probability distribution, while the necessity measure, to a lower probability distribution of the corresponding random variable in the sense of Dempster [29]. Then it is reasonable to define the mean value of a fuzzy number as a closed interval whose bounds are expectations of upper and lower probability distributions. The comparison of two fuzzy numbers boils then down to the comparison of arithmetic means of these bounds, which is computationally equivalent to another intuitive comparison principle based on the area compensation determined by the membership functions of two fuzzy numbers being compared [16], [38], [104], [138].

The main advantage of the area compensation method is that it answers not only which one of two fuzzy numbers is greater but also what is the degree to which one fuzzy number is greater than the other one. Working on fuzzy scheduling problems [59] (see also section 10.2.1) we have used this comparison method under the name of weak comparison rule (WCR) in order to distinguish it from a more restrictive comparison method used to observe precedence constraint and called strong comparison rule (SCR).

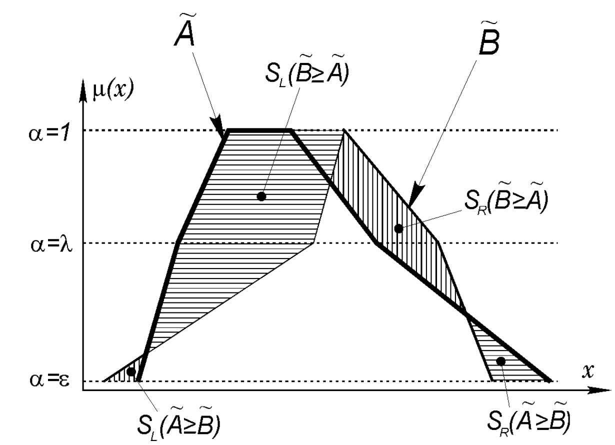

Figure 25. The comparison of two fuzzy numbers based on area compensation

Let $\tilde{A}$, $\tilde{B}$ be two fuzzy numbers and $S_L(\tilde{A} \geq \tilde{B})$, $S_R(\tilde{A} \geq \tilde{B})$, $S_L(\tilde{A} \leq \tilde{B})$, $S_R(\tilde{A} \leq \tilde{B})$ the areas determined by their membership functions (Figure 25):

$$S_L(\tilde{A} \geq \tilde{B}) = \int_{U(A,B)} (\inf_{x \in \mathbf{R}} A_\alpha - \inf_{x \in \mathbf{R}} B_\alpha) d\alpha$$

$$S_R(\tilde{A} \geq \tilde{B}) = \int_{V(A,B)} (\sup_{x \in \mathbf{R}} A_\alpha - \sup_{x \in \mathbf{R}} B_\alpha) d\alpha$$

where

$$U(\tilde{A}, \tilde{B}) = \{\alpha | \inf_{x \in \mathbf{R}} A_\alpha \geq \inf_{x \in \mathbf{R}} B_\alpha, \varepsilon \leq \alpha \leq 1\}$$

$$V(\tilde{A}, \tilde{B}) = \{\alpha | \sup_{x \in \mathbf{R}} A_\alpha \geq \sup_{x \in \mathbf{R}} B_\alpha, \varepsilon \leq \alpha \leq 1\}$$

$S_L(\tilde{A} \leq \tilde{B})$ and $S_R(\tilde{A} \leq \tilde{B})$ are defined analogously.

$S_L(\tilde{A} \geq \tilde{B})$ and $S_R(\tilde{A} \geq \tilde{B})$ are the areas that are related to the possibility of $A^* \geq B^*$, where A^* and B^* are potential realizations of $\tilde{A}$ and $\tilde{B}$. The degree $C(\tilde{A} \geq \tilde{B})$ to which $\tilde{A} \geq \tilde{B}$ is calculated as a sum of areas in which $A^* \geq B^*$ minus a sum of areas in which $A^* \leq B^*$.

Definition 5 (Weak Comparison Rule – WCR)

According to the WCR, the degree C to which a fuzzy number $\tilde{A}$ is greater than or equal to $\tilde{B}$ is defined as:

$$C(\tilde{A} \geq \tilde{B}) = \frac{1}{2}\left\{S_L(\tilde{A} \geq \tilde{B}) + S_R(\tilde{A} \geq \tilde{B}) - S_L(\tilde{B} \geq \tilde{A}) - S_R(\tilde{B} \geq \tilde{A})\right\}$$

Using WCR, one can define three relations between $\tilde{A}$ and $\tilde{B}$, corresponding to weak inequality, strong inequality and equivalence between fuzzy numbers, respectively:

$\tilde{A} \geq \tilde{B}$ iff $C(\tilde{A} \geq \tilde{B}) \geq 0$,

$\tilde{A} > \tilde{B}$ iff $C(\tilde{A} \geq \tilde{B}) > 0$,

$\tilde{A} \sim \tilde{B}$ iff $C(\tilde{A} \geq \tilde{B}) = C(\tilde{B} \geq \tilde{A}) = 0$.

It is easy to note that $C(\tilde{A} \geq \tilde{B}) = -C(\tilde{B} \geq \tilde{A})$.

The proposed WCR is related to the comparison of "mean values of fuzzy numbers", as defined in [31] within the framework of the Dempster-Shafer theory. The mean value of a fuzzy number $\tilde{A}$ is the interval $[E(\tilde{A})_*, E(\tilde{A})^*]$, where $E(\tilde{A})_*$ and $E(\tilde{A})^*$ are the mean values related to the cumulative possibility distribution (left spread), and to the cumulative necessity distribution (right spread), respectively. The natural way of defuzzifying such an interval is to calculate the arithmetic mean of its limits. The defuzzification function $\Im(\tilde{A})$ is defined as follows [16]:

$$\Im(\tilde{A}) = \frac{E_*(\tilde{A}) + E^*(\tilde{A})}{2}$$

The following properties of the function $\Im(\tilde{A})$ and of the WCR will be useful.

Property 1 [53]

For a fuzzy number in the six-point representation, the function $\Im(\tilde{A})$ can be calculated as:

$$\Im(\tilde{A}) = \frac{1}{4(1-\varepsilon)}\left\{(\lambda - \varepsilon)(\underline{a}^\varepsilon + \underline{a}^\lambda + \bar{a}^\lambda + \bar{a}^\varepsilon) + (1 - \lambda)(\underline{a}^\lambda + \underline{a} + \bar{a} + \bar{a}^\lambda)\right\}$$

Property 2

The degree $C(\tilde{A} \geq \tilde{B})$ to which $\tilde{A}$ is greater than or equal to $\tilde{B}$, calculated according to the WCR, can be calculated using $\Im(\tilde{A})$ and $\Im(\tilde{B})$ as:

$$C(\tilde{A} \geq \tilde{B}) = \Im(\tilde{A}) - \Im(\tilde{B}) = \frac{E_*(\tilde{A}) + E^*(\tilde{A})}{2} - \frac{E_*(\tilde{B}) + E^*(\tilde{B})}{2}$$

From properties 1 and 2 immediately follows the formula for $C(\tilde{A} \geq \tilde{B})$ where $\tilde{A}$ and $\tilde{B}$ are six-point fuzzy numbers.

8.1.4. Fuzzy multiple objective combinatorial optimization and fuzzy dominance

The general fuzzy multi-objective combinatorial optimization (fuzzy MOCO) problem is formulated as:

$$\text{maximize}\left\{\tilde{f}_1(\mathbf{x}),...,\tilde{f}_J(\mathbf{x})\right\}$$
$$\text{s.t. } \mathbf{x} \in D,$$

where: *solution* $\mathbf{x} = [x_1,...,x_I]$ is a vector of discrete and crisp *decision variables*, D is a finite set of feasible solutions, $\tilde{f}_1,...,\tilde{f}_J$ are fuzzy objective functions which, for a given $\mathbf{x}$ take values called *fuzzy scores*.

Thus, the image of a solution $\mathbf{x}$ in the objective space is a vector $\tilde{\mathbf{f}}(\mathbf{x}) = [\tilde{f}_1(\mathbf{x}),...,\tilde{f}_J(\mathbf{x})]$ composed of J fuzzy numbers. We will say that $\tilde{\mathbf{f}}(\mathbf{x})$ is a *fuzzy point* in the objective space. In order to define a dominance relation in the objective space we will use the WCR for the fuzzy scores. The dominance relation defined in this way will be called *WCR-dominance*.

Fuzzy point $\tilde{\mathbf{f}}(\mathbf{x})$ *WCR-dominates* fuzzy point $\tilde{\mathbf{f}}(\mathbf{y})$ iff $\tilde{f}_j(\mathbf{y}) >= \tilde{f}_j(\mathbf{x}) \forall j$ and $\tilde{f}_j(\mathbf{y}) > \tilde{f}_j(\mathbf{x})$ for at least one j, i.e. iff $C\left(\tilde{f}_j(\mathbf{y}) \geq \tilde{f}_j(\mathbf{x})\right) \geq 0 \ \forall j$ and $C\left(\tilde{f}_j(\mathbf{y}) \geq \tilde{f}_j(\mathbf{x})\right) > 0$ for at least one j.

A fuzzy point $\tilde{\mathbf{f}}(\mathbf{x})$ is *WCR-nondominated* if there is no other fuzzy point $\tilde{\mathbf{f}}(\mathbf{y})$ ($\mathbf{y} \in D$) that WCR-dominates $\tilde{\mathbf{f}}(\mathbf{x})$. A solution $\mathbf{x}$ is *WCR-Pareto-optimal* (*WCR-efficient*) if its image is WCR-nondominated.

The fuzzy *ideal point* is a fuzzy point $\tilde{\mathbf{f}}^*$ in the objective space composed of the best attainable fuzzy values of objectives, i.e.:

$$\forall j = 1,...,J \ \forall \mathbf{x} \in D \ \tilde{\mathbf{f}}_j^* \geq \tilde{\mathbf{f}}_j(\mathbf{x}).$$

The fuzzy *nadir point* is a fuzzy point $\tilde{\mathbf{f}}_*$ in the objective space composed of the worst attainable fuzzy values of objectives in the set of WCR-nondominated points, i.e.

$$\forall j = 1,...,J \ \forall \mathbf{x} \in N \ \tilde{\mathbf{f}}_{*j} \leq \tilde{\mathbf{f}}_j(\mathbf{x}).$$

In the definitions of the fuzzy ideal and fuzzy nadir points the fuzzy scores are compared using WCR.

8.2. Fuzzy Pareto simulated annealing

Adaptation of PSA to the fuzzy case requires:

- the use of WCR-dominance in updating the set of potentially WCR-Pareto-optimal solutions,
- calculating the probability of accepting the new solutions based on fuzzy scores.

As it was described in section 5.1 two rules are suggested in PSA. In the fuzzy version of the method, rule *SL* takes the following form:

$$P(\mathbf{x},\mathbf{y},T,\Lambda)=\min\left\{1,\exp\left(\sum_{j=1}^{J}\lambda_j\left(C\left(\tilde{f}_j(\mathbf{y})\geq\tilde{f}_j(\mathbf{x})\right)\right)/T\right)\right\},$$

while rule *C* takes the following form:

$$P(\mathbf{x},\mathbf{y},T,\Lambda)=\min\left\{1,\exp\left(\max_j\left\{\lambda_j\left(C\left(\tilde{f}_j(\mathbf{y})\geq\tilde{f}_j(\mathbf{x})\right)\right)/T\right\}\right)\right\}.$$

In other words the distance between two fuzzy scores $\tilde{f}_j(\mathbf{x})$ and $\tilde{f}_j(\mathbf{y})$ is calculated according to the WCR. Note that the two rules assure that **y** is accepted with probability equal to 1 if it WCR-dominates **x**.

8.3. Fuzzy Light Beam Search-Discrete

The fuzzy version of the Light Beam Search-Discrete works on a set of points that are described by fuzzy values of the objectives. Adaptation of LBS-Discrete to the fuzzy case requires:

- the use of the fuzzy ideal and fuzzy nadir points,
- selection of the middle point on the basis of the achievement scalarizing function defined on fuzzy objective values,
- the definition of the outranking relation on fuzzy points.

The definitions of the fuzzy ideal and nadir points have been given in section 8.1.4.

The achievement scalarizing function in the fuzzy case is calculated in the following way:

$$s_a\left(\tilde{\mathbf{f}}(\mathbf{x}),\mathbf{z}^0,\Lambda,\rho\right)=\max_j\left\{\lambda_j C\left(z_j^0\geq\tilde{f}_j(\mathbf{x})\right)\right\}+\rho\sum_{j=1}^{J}\lambda_j C\left(z_j^0\geq\tilde{f}_j(\mathbf{x})\right).$$

In other words the distance between the crisp aspiration level z_j^0 and the fuzzy score $\tilde{f}_j(\mathbf{x})$ is calculated according to the WCR.

The definition of the outranking relation proposed in section 7.3.2 requires distinguishing between indifference, weak preference, strict preference and veto preference relations with respect to objective z_j:

$\tilde{\mathbf{f}}(\mathbf{x})\, I_j\, \tilde{\mathbf{f}}(\mathbf{y})$ i.e. $\tilde{\mathbf{f}}(\mathbf{x})$ and $\tilde{\mathbf{f}}(\mathbf{y})$ are indifferent $\Leftrightarrow -q_j\leq C\left(\tilde{f}_j(\mathbf{x})\geq\tilde{f}_j(\mathbf{x})\right)\leq q_j$,

$\tilde{\mathbf{f}}(\mathbf{x})\, Q_j\, \tilde{\mathbf{f}}(\mathbf{y})$ i.e. $\tilde{\mathbf{f}}(\mathbf{x})$ is weakly preferred to $\tilde{\mathbf{f}}(\mathbf{y}) \Leftrightarrow q_j\leq C\left(\tilde{f}_j(\mathbf{x})\geq\tilde{f}_j(\mathbf{y})\right)\leq p_j$,

$\tilde{\mathbf{f}}(\mathbf{x})\, P_j\, \tilde{\mathbf{f}}(\mathbf{y})$ i.e. $\tilde{\mathbf{f}}(\mathbf{x})$ is strictly preferred to $\tilde{\mathbf{f}}(\mathbf{y}) \Leftrightarrow\ p_j \le C\left(\tilde{f}_j(\mathbf{x}) \ge \tilde{f}_j(\mathbf{y})\right)$,

$\tilde{\mathbf{f}}(\mathbf{x})\, V_j\, \tilde{\mathbf{f}}(\mathbf{y})$ i.e. $\tilde{\mathbf{f}}(\mathbf{x})$ is veto preferred to $\tilde{\mathbf{f}}(\mathbf{y}) \Leftrightarrow\ v_j \le C\left(\tilde{f}_j(\mathbf{x}) \ge \tilde{f}_j(\mathbf{y})\right)$.

8.4. Summary

The approach to extension of the Pareto simulated annealing and the Light Beam Search for fuzzy combinatorial optimization problems presented in this section is quite general and can be used with many other methods. In particular multiple objective local search described in section 6 and the Interactive Trichotomy Segmentation described in section 7.4 may be adapted to fuzzy context in similar way.

The main advantage of our approach is that it considers nondominated points in the original fuzzy objective space and not nondominated points in the objective space of an auxiliary deterministic problem (no defuzzification). In result the best compromise chosen by the DM corresponds to a single solution. On the contrary, the classical scenario analysis may result in different solutions corresponding to particular scenarios. For example, in the case of project scheduling the schedules corresponding to the optimistic, pessimistic and the most possible scenario may be completely different [62]. It is caused by the fact that each of the scenarios corresponds to separate deterministic problem that, in general, has different set of Pareto-optimal solutions.

9. Computational experiments

Although the authors of multiple objective metaheuristics usually claim that their algorithms can solve large-scale real-life problems, most of the computational experiments reported in the literature are performed on toy problems only. For example among thirty test problems described in [173] only six involve more than two variables. Furthermore, descriptions of few experiments give enough details to allow other researchers to compare their new results. In this section we describe a number of comparative experiments with PSA and MOGLS.

9.1. The performance of multiple objective genetic local search on traveling salesperson problem

The experiment described in this section is an updated version of the experiments reported in [81] and [85]. Our MOGLS algorithm was compared with Ishibuchi's and Murata's MOGLS (IMMOGLS) [75] (see also section 3.3), Fonseca's and Fleming's Pareto ranking-based multiple objective genetic algorithm [35] (Pareto GA) and introduced in [81] new MOGLS algorithm based on the idea of MOSA method [170] (MOSA-like MOGLS).

9.1.1. Multiple objective symmetric traveling salesperson problem

Single objective TSP is often used to test single objective metaheuristics. It is defined by a set of cities and a cost (distance) of travel between each pair of cities. In symmetric TSP the cost does not depend on the 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 our case, J-objective symmetric TSP instances are constructed from J different single objective TSP instances having the same number of cities. Thus, j-th cost factor, $j=1,...,J$, between a pair of cities comes from j-th single objective instance. Individual optima of particular objectives are equal to optima of corresponding single objective instances. In our case, the single objective instances are completely independent, so, also objectives are independent and therefore non-correlated. The same approach was used by Borges and Hansen [7].

Also following Borges and Hansen [7] we use multiple objective TSP instances based on the TSPLIB library [72], [132]. For example, problem instance kroAB100 denotes a bi-objective instance with cost factors corresponding to the first objective taken from kroA100, and cost factors corresponding to the second objective taken from kroB100. kroABC100 denotes a three objective instance with cost factors taken from kroA100, kroB100 and kroC100 instances. In this way 10 different bi-objective instances and 10 three-objective instances were created. We used also instances with 50 leading cities taken from kroA100-kroE100 instances. In the experiment described in section 9.2 we also used 5 four-objective instances.

9.1.2. Methods used in the experiment

Pareto ranking-based Fonseca's and Fleming's multiple objective genetic algorithm (Pareto GA)

The algorithm proposed by Fonseca and Flemming [35] (see also section 3.1.2) assigns to each solution in the population ranking equal to the number of other solutions that dominate it plus one. Thus, solutions nondominated in the current population are assigned rank 1. Other solutions have ranks greater than 1. In order to avoid genetic drift to a small region of the nondominated set the algorithm uses fitness sharing in the objective space.

Ishibuchi's and Murata's Multiple Objective Genetic Local Search (IMMOGLS)

Ishibuchi and Murata [75] (see also section 3.3) were the first authors to propose a multiple objective genetic local search algorithm (see also section 3.3). They use the same idea of randomly selecting a weight vector for each iteration. Originally they suggested the use of linear functions, but the method can be easily generalized to other weighted scalarizing functions. Their method uses a standard genetic population with no mating restrictions. The solutions for recombination are selected according to the roulette wheel scheme, with fitness depending on the current scalarizing function. Generation replacement is used. Furthermore, a portion of elite potentially Pareto-optimal solutions is added to a new generation.

In our experiment we used our own implementation of the algorithm which shared the majority of the code with the implementation of our MOGLS. We used the same way of constructing the initial set of solutions, the same way of drawing random weight vectors, the same recombination, mutation and greedy repair operators.

MOSA-like multiple objective genetic local search

Ulungu et al. [169], [170] (see also section 3.2.1) proposed a method based on simulated annealing called MOSA. In MOSA a number of weighted linear functions are selected. Each of them is the optimized sequentially or in parallel by an independent simulated annealing process. The outcome of this algorithm are not only the best solutions obtained for each of the optimized functions, but all potentially Pareto-optimal solutions found during the optimization. The idea of the method is very general and can easily be used with any other metaheuristic, e.g. GLS. Such an algorithm will be called MOSA-like MOGLS. The fact that the algorithm uses a number of separate populations may be interpreted as a form of strict mating restrictions.

9.1.3. Quality measure used in the experiment

As the quality measure in the experiment we used the estimation of the expected value of weighted Tchebycheff utility function on approximation A over the set of normalized weight vectors Λ – $\hat{R}$ (see section 4.2.3). The utility functions were scaled such that each of them was achieving value 1 at the ideal point and value 0 at the approximate nadir point.

In the case of 100 cities instances the ideal point is known, because each individual objective corresponds to a single objective problem coming from the TSPLib library. 50-cities instances are relatively easy to solve optimally. Thus the exact ideal points were used as reference points in the quality measure. The objectives were normalized with normalization factors based on the estimated ranges of objectives in the nondominated set. Since the objectives are independent the elements of the nadir points were estimated on the basis of the expected objectives values for random solutions. In the case of instances with 50 cities they were all set equal to 80000, while in the case of instances with 100 cities they were all set equal to 180000.

9.1.4. Adaptation of the algorithms to the traveling salesperson problem

The recombination operator we used is the distance preserving crossover introduced by Freisleben and Merz [40]. An offspring is constructed in the following steps:

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

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

No mutation operator was used; however, the second step of the recombination operator introduces some randomness.

Local optimization was performed in two phases. In the first phase local search did not take into account arcs that were common to both parents. In the second phase all arcs were considered. In many cases no improvement was found in the second phase. This approach gives only slightly worse results than a single local search considering all arcs, but is significantly faster [82].

The local search used was based on standard 2-arcs exchange neighborhood (see Figure 26). While constructing the initial population greedy local search was used. After recombination steepest local search was used. This combination was found to give the best results. The greedy local search tests the neighborhood moves in random order and performs the first improving move found. The steepest local search tests all neighborhood moves and performs the best improving move. Both versions stop when no improving move is found in the whole neighborhood.

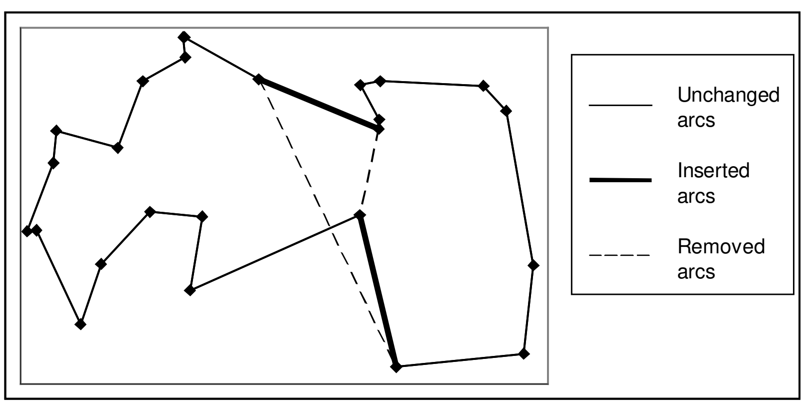

Figure 26. Illustration of the 2-arcs exchange move in 2D Euclidean TSP

Note that the local search is relatively simple and leaves space for many improvements. Merz and Freisleben [119] describe a state of the art GLS for single objective TSP which uses a number of techniques that significantly increase the efficiency of local search.

9.1.5. The selection of the type of utility functions

As it was mentioned in section 6.2, in the case of discrete problems, weighted Tchebycheff scalarizing functions seem to have an advantage over weighted linear scalarizing functions. Each Pareto-optimal solution is a global optimum of a weighted Tchebycheff scalarizing function, while weighted linear scalarizing functions achieve global optima on a subset of Pareto-optimal solutions only, i.e. supported Pareto-optimal solutions (see section 2).

Our implementation of MOGLS for TSP uses, however, weighted linear functions. Several reasons for that are discussed below.

First of all, weighted Tchebycheff functions are more difficult to optimize than weighted linear functions for multiple objective TSP. Hansen [51] noticed that in the case of multiple objective TSP both local search and tabu search give better final values of weighted Tchebycheff functions, when the search within the neighborhood of the current solution is guided by weighted linear functions than when it is guided by weighted Tchebycheff functions. Our observations confirm this phenomenon. Figure 27, for example, presents local optima of 1000 randomly selected weighted Tchebycheff and local optima of 1000 randomly selected weighted linear functions obtained for kroAB100 bi-objective TSP instance with 100 cities. We also present a reference set, i.e. the best approximation we have obtained. Notice that local optima of weighted linear functions are much closer to the reference set.

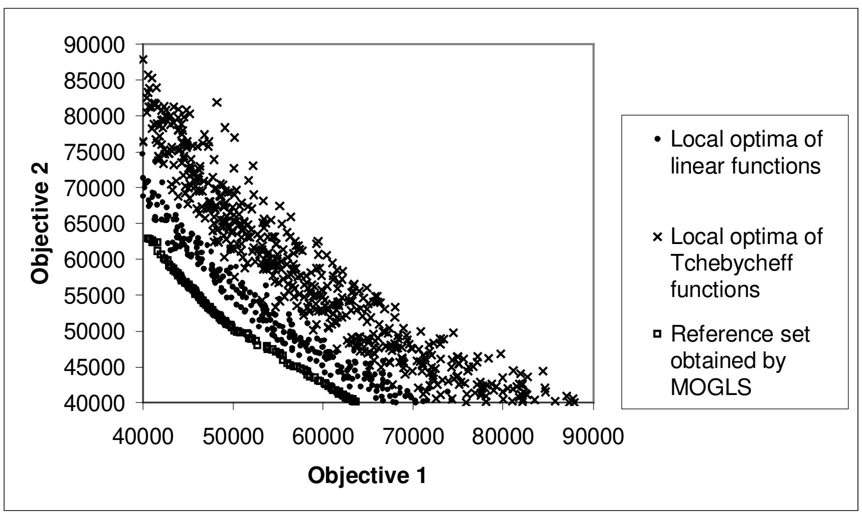

Figure 27. Local optima of weighted linear and weighted Tchebycheff functions with random weights

Because of the special structure of TSP local search is also faster in the case of weighted linear functions, because it requires less arithmetical operations.

As it was mentioned above global optima of weighted Tchebycheff functions contain all Pareto-optimal solutions. Since we work with a heuristic algorithm and since the set of potentially Pareto-optimal solutions is updated with every newly generated local optimum, it is more interesting to analyze what is the chance of finding Pareto-optimal solutions among good local optima of a function. In order to test it we used a set of 1000 local optima of randomly selected weighted linear functions (similar results were obtained when local optima of Tchebycheff functions were used). Among the 1000 solutions 130 were potentially Pareto-optimal. Then, 1000 other randomly selected weighted linear functions and weighted Tchebycheff functions were used. For each of the functions samples composed of 1, 2,...,50 best solutions were selected from the set of 1000 solutions. For each sample, the number of potentially Pareto-optimal solutions contained in the sample was counted. The average results are presented in Figure 28. One can notice that on average *n* best solutions of a weighted linear function contains more potentially Pareto-optimal solutions. For example, among 50 best solutions of a weighted linear function on average more than 19 were potentially Pareto-optimal, while among 50 best solutions of a weighted Tchebycheff function on average less than 11 were potentially Pareto-optimal. Intuitively, it can be explained by comparison of Figure 29 and Figure 30. Good solutions of weighted linear functions are dispersed over the nondominated set while good solutions of weighted Tchebycheff functions are more concentrated and placed inside the feasible set.

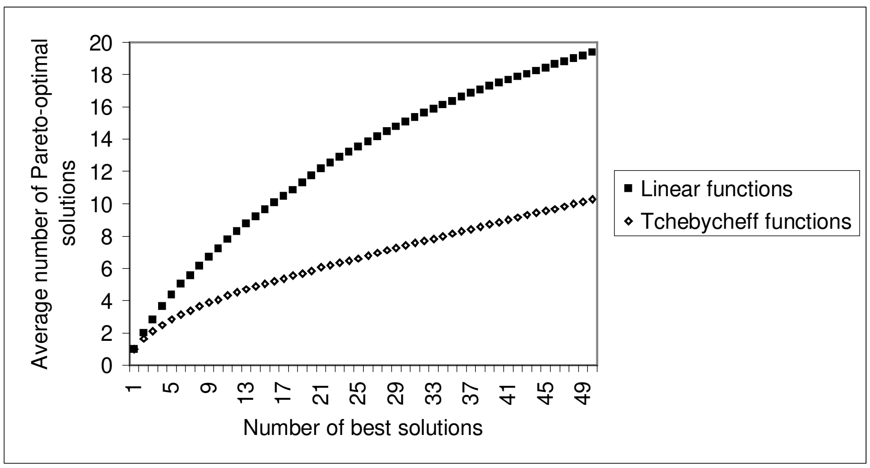

Figure 28. Average numbers of Pareto-optimal solutions among *n* best solutions of linear and weighted Tchebycheff functions

Summarizing, we have decided to use weighted linear functions in the implementation of MOGLS for multiple objective TSP because such functions are easier to optimize and give higher chance for finding new potentially Pareto-optimal solutions than weighted Tchebycheff functions. Note, however, that we do not claim that the two kinds of functions have the same properties on other problems. The global shape of the nondominated set in the case of multiple objective TSP is relatively smooth (see e.g. Figure 32). Weighted Tchebycheff functions may be better for problems with more irregular nondominated sets.

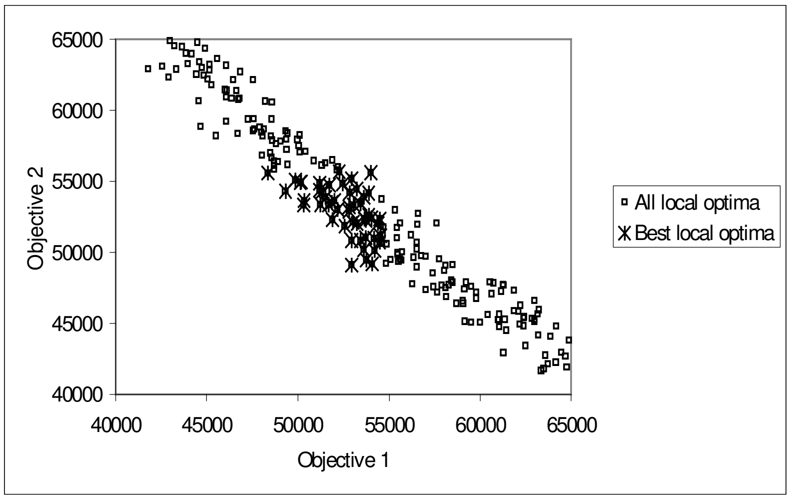

Figure 29. 50 best local optima of a weighted Tchebycheff function

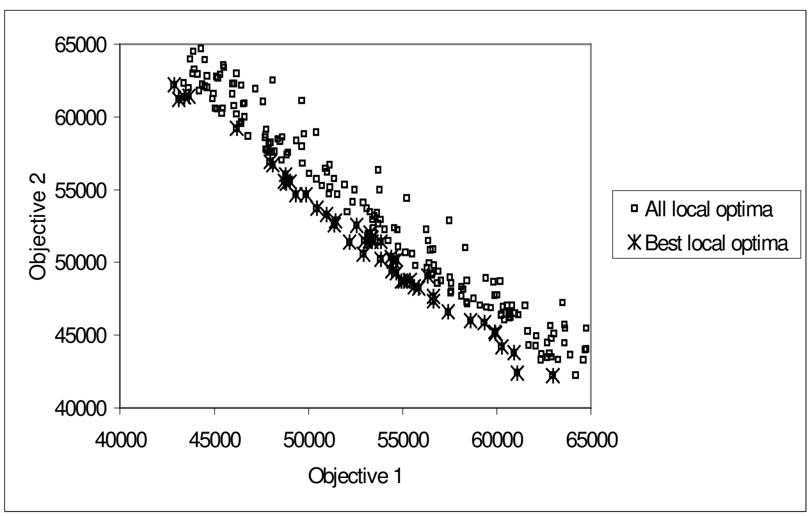

Figure 30. 50 best local optima of a weighted linear function

9.1.6. Computational experiment

The number of initial solutions for our MOGLS was set automatically (see section 6.4) resulting in values reported in Table 2. The CPU time spent by the algorithm was controlled by changing the number of recombinations.

In the case of Ishibuchi's and Murata's algorithm, the size of the population was set in the same way as the number of initial solutions in our MOGLS. Thus both algorithms were starting by generating the same number of random local optima. In both algorithms the random scalarizing functions were generated in the same way. Also the number of recombinations was the same for the two algorithms. The elite size in IMMOGLS was set equal to 10% of the population size. This size was chosen experimentally using the best choice principle. We have noticed however, that this parameter has relatively small influence on the performance of the method.

In the case of MOSA-like MOGLS a predefined numbers of weight vectors were used. They were generated systematically by the approach described in section 4.2.3. The CPU time was controlled by changing the number of weight vectors. Optimization of each of the scalarizing function was continued till the population converged and no further improvement could be achieved as described in section 6.1.

Table 3. Comparison of the algorithms on bi-objective instances with 50 cities

Pareto GA					IMMOGLS			
Number of generations	Number of recombinations	CPU time [s]	Thousands of functions evaluations	Average quality	Number of recombinations	CPU time [s]	Thousands of functions evaluations	Average quality
1000	100000	13.7	100	0.77301	1080	12.3	13938	0.873895
2000	200000	27.7	200	0.77550	2160	23.5	26667	0.874639
3000	300000	41.2	300	0.77575	3240	34.8	39503	0.874909
4000	400000	54.9	400	0.77696	4320	45.8	52103	0.875076
5000	500000	68.7	500	0.77715	5400	57	64871	0.875194
MOSA-like MOGLS					MOGLS			
Number of utility functions	Number of recombinations	CPU time [s]	Thousands of functions evaluations	Average quality	Number of recombinations	CPU time [s]	Thousands of functions evaluations	Average quality
10	1071	3.6	4014	0.873336	1080	3.9	3815	0.874539
20	2194	7.4	7992	0.874595	2160	6.4	6010	0.874903
30	3154	11	11795	0.874894	3240	8.9	8144	0.875108
40	4402	14.9	16154	0.875098	4320	11.5	10266	0.875217
50	5349	18.4	19842	0.875134	5400	14	12367	0.875297

Pareto ranking-based GA (Pareto GA) used fitness sharing and no mating restrictions. The recombination operator was the same as in the case of MOGLS algorithms. Fonseca and Flemming [35] do not give guidelines for setting the population size. Following [173], ch. 6.3.3.6, we used population of size 100. In the case of MOGLS, no mutation operator was used. In the case of Pareto GA we have noticed that the lack of mutation operator results in very fast convergence of the population. Thus a mutation operator was applied with probability 0.1. The mutation operator exchanges two randomly selected arcs.

Results of the experiments are presented in Table 3, Table 4, Table 5 and in Figure 31. For each problem size 10 different instances described in section 6.1 were used. A single run of each method was performed on each instance.

Pareto GA is clearly outperformed by all MOGLS algorithms.

It can be observed that Ishibuchi's and Murata's MOGLS is by far least effective of all the MOGLS algorithms. Only on the smallest bi-objective instances with 50 cities it gives results comparable to other algorithms but requires more functions evaluations. On the instances with 100 cities Ishibuchi's and Murata's MOGLS does not achieve the quality given by the shortest run of our MOGLS even in a 14 times longer time. Note also, that on average Ishibuchi's and Murata's MOGLS performs more functions evaluations per recombination than the other algorithms, i.e. on average local search in this algorithm is significantly longer. In our opinion, this happens because, in general, solutions recombined in Ishibuchi's and Murata's MOGLS are worse on the current scalarizing function and less similar than in the other algorithms. In result, offspring are far from local optima and local search needs more functions evaluations to reach local optimum.

Table 4. Comparison of the algorithms on bi-objective instances with 100 cities

Pareto GA					IMMOGLS			
Number of generations	Number of recombinations	CPU time [s]	Thousands of functions evaluations	Average quality	Number of recombinations	CPU time [s]	Thousands of functions evaluations	Average quality
1000	100000	32.9	100	0.75524	1420	133.1	159739	0.901577
2000	200000	65.8	200	0.75639	2840	256	309435	0.902141
3000	300000	99.2	300	0.75837	4260	377.4	457891	0.902456
4000	400000	132.8	400	0.75861	5680	499.9	607413	0.902632
5000	500000	167.1	500	0.75900	7100	621.7	756456	0.902802
MOSA-like MOGLS					MOGLS			
Number of utility functions	Number of recombinations	CPU time [s]	Thousands of functions evaluations	Average quality	Number of recombinations	CPU time [s]	Thousands of functions evaluations	Average quality
10	1630	29.4	30350	0.901347	1420	30.1	31478	0.903309
20	3341	60.4	61497	0.903267	2840	44.9	46414	0.903768
30	5051	90.6	94029	0.903619	4260	59.6	60406	0.903972
40	7167	124.3	129379	0.903828	5680	74.2	73863	0.904082
50	8766	154.4	160442	0.903889	7100	88.3	86862	0.904167

Table 5. Comparison of the algorithms on three-objective instances with 100 cities

Pareto GA					IMMOGLS			
Number of generations	Number of recombinations	CPU time [s]	Thousands of functions evaluations	Average quality	Number of recombinations	CPU time [s]	Thousands of functions evaluations	Average quality
5000	500000	220.6	500	0.70643	6620	996.6	1029637	0.859863
10000	1000000	487.7	1000	0.71364	13240	1938.8	2015579	0.860726
15000	1500000	767.2	1500	0.71638	19860	2883	2999410	0.86117
20000	2000000	1035.3	2000	0.71684	26480	3830.1	3981994	0.861476
25000	2500000	1300.7	2500	0.71792	33100	4777.3	4966231	0.861703
MOSA-like MOGLS					MOGLS			
Number of utility functions	Number of recombinations	CPU time [s]	Thousands of functions evaluations	Average quality	Number of recombinations	CPU time [s]	Thousands of functions evaluations	Average quality
55	10435	211.4	194225	0.860187	6620	223.2	179230	0.862988
91	17279	369.7	323748	0.862565	13240	411	268712	0.863973
136	26102	589.5	488423	0.863735	19860	619.8	347328	0.864439
190	36520	868.5	682463	0.864336	26480	830.8	421326	0.86472
253	48698	1224.1	911775	0.864751	33100	1044.4	492552	0.864923

The difference between MOSA-like MOGLS and our algorithm is lower. Figure 10 illustrates, however, that our algorithm outperforms MOSA-like MOGLS, i.e. gives better quality in shorter time. Because the difference between the quality of results of the two algorithms is low we study the statistical significance of the differences. Because the average values of quality reported in Table 3, Table 4 and Table 5 are averages of results obtained for different instances their variations cannot be used directly. Thus we analyze variations of differences between results of the two algorithms for the same instances. More precisely, we test the statistical hypothesis that the differences of the quality of results of our MOGLS and MOSA-like MOGLS corresponding to the parameters settings described in the same rows in Table 3, Table 4 and Table 5 are greater than 0. For example we compare the results of MOSA-like MOGLS with 10 utility functions with the results of our MOGLS with 1080 recombinations on bi-objective instances with 50 cities. The results of the analysis are reported in Table 6. The differences are significant at level 0.01 except for the longest runs on bi-objective instances. We expect that this is caused by the fact that on the smallest instances results of both the algorithms converge very close to the nondominated set and thus no significant differences between the algorithms are possible.

The exact nondominated sets for the instances used in the experiment are not known. However, Borges and Hansen [7] generated a large set of supported Pareto-optimal solution for kroABC100 instance. From this set one can extract supported solutions for kroAB100, kroAC100 and kroBC100 instances. Unfortunately, there is no warranty that the sets contain all supported solutions. We use the supported solutions to graphically present the quality of the obtained approximations. Figure 32 and Figure 33 present supported solutions of kroAB100 instance and one of the approximations obtained with temporary elite populations of size 16 after 7100 recombinations. The approximation contains solutions very close or identical to the supported ones.

The approximations of the nondominated sets generated in this experiment, as well as the C++ code used to calculate the quality measure are available on the Internet at www-idss.cs.put.poznan.pl./~jaszkiewicz/motsp/.

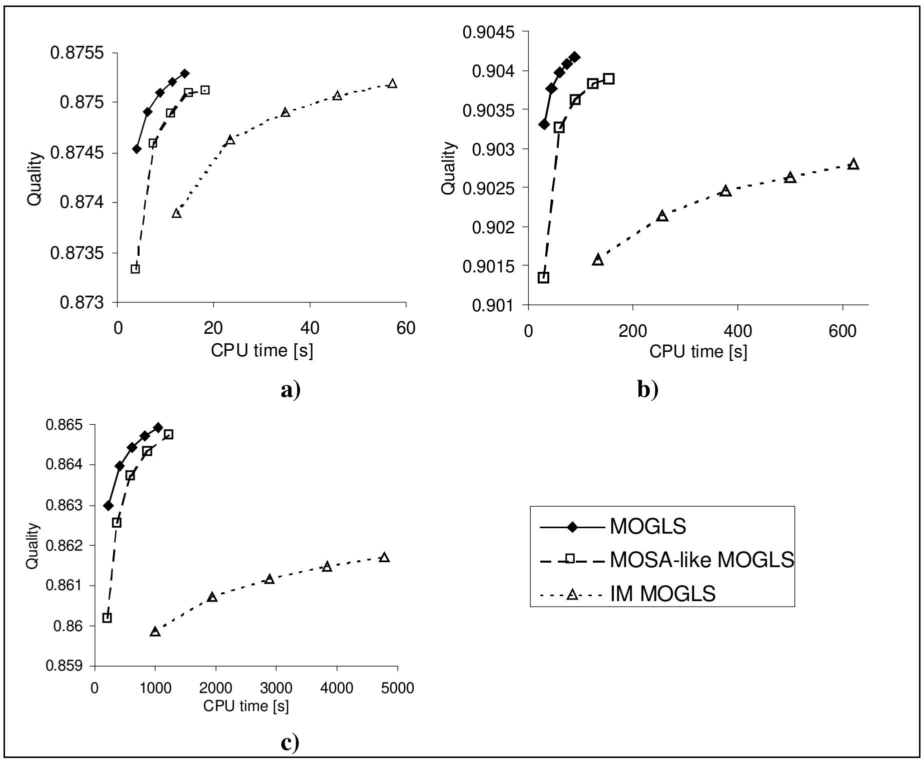

Figure 31. Results on a) bi-objective instances with 50 cities b) bi-objective instances with 100 cities c) three-objective instances with 100 cities

Table 6. Statistical significance of differences between MOSA-like MOGLS and MOGLS

Number of utility functions in MOSA-like MOGLS	Number of recombinations in MOGLS	Average difference of results over 10 instances	Standard deviation of the difference of results	Statistically significant at level 0.01
Bi-objective instances with 50 cities				
10	1188	0.001515	0.000341	yes
20	2268	0.000414	0.000188	yes
30	3348	0.000179	0.000144	yes
40	4428	8.35E-05	9.32E-05	no
50	5508	4.73E-05	8.12E-05	no
Bi-objective instances with 100 cities				
10	1420	0.00234	0.000349	yes
20	2840	0.000425	0.000227	yes
30	4260	0.000209	0.000155	yes
40	5680	0.000202	0.000143	yes
50	7100	0.000118	0.000152	no
Three-objective instances with 100 cities				
55	6620	0.004387	0.000293	yes
91	13240	0.001993	0.000197	yes
136	19860	0.000943	0.000144	yes
190	26480	0.000486	0.000116	yes
253	33100	0.000275	0.000116	yes

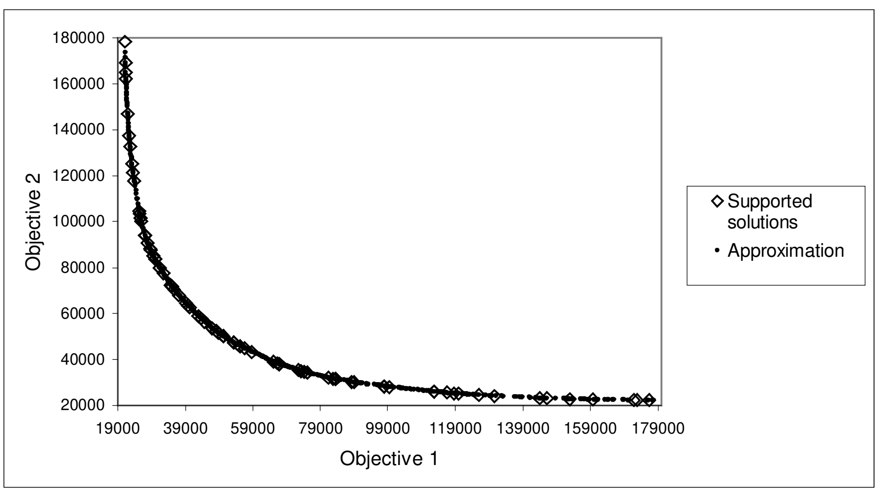

Figure 32. Supported solutions of kroAB100 instance and one of the approximations obtained with population of size 16 after 7100 recombinations

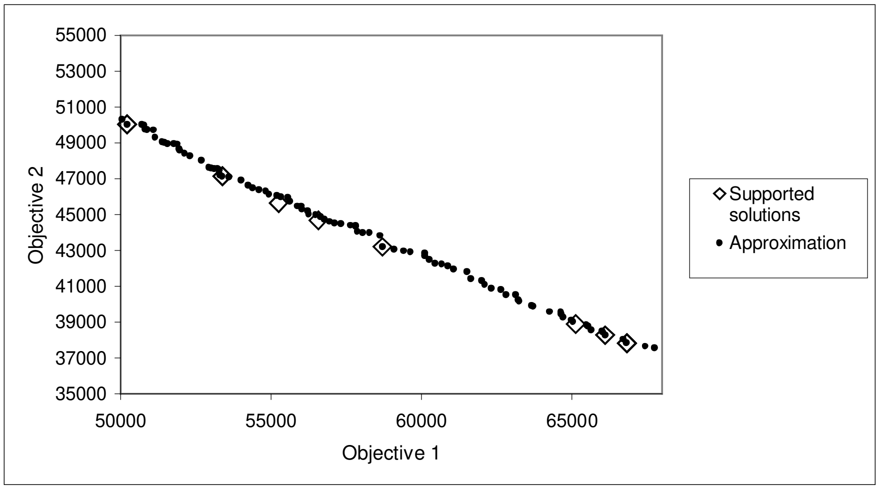

Figure 33. Supported solutions of kroAB100 instance and one of the approximations obtained with temporary elite populations of size 16 after 7100 recombinations – zoomed view

9.2. The computational effectiveness of the off-line generation of approximately Pareto-optimal solutions in the case of traveling salesperson problem

In the experiment described in this section the approach proposed in section 7.2 is applied in the case of multiple objective traveling salesperson problem [83]. The problem is described in section 9.1.1 Two multiple objective metaheuristics are evaluated – multiple objective genetic local search and Pareto ranking-based Fonseca's and Fleming's multiple objective genetic algorithms (Pareto GA) (see section 9.1.2). The adaptation of the two methods to the multiple objective TSP has been described in section 9.1.4. The goal of the experiment was not to evaluate the relative performance of MOGLS and Pareto GA which has been done in section 9.1, but to evaluate each of the two algorithms with respect to its root single objective method. Thus, MOGLS has been compared with the single objective GLS algorithm described in section 6.1, while Pareto GA has been compared with a standard single objective GA. Note that GLS is one of the best methods for single objective TSP [119].

We used the achievement scalarizing functions with $\rho = 0.1$. Parameter L – the number of scalarizing functions, on which the pairs of methods were compared, was equal to 50. The weight vectors defining the functions were generated randomly with the algorithm described in section 6.2.

For each instance the following experiment was performed. In the first phase the 50 scalarizing functions were optimized with a single objective method. Then, the corresponding multiple objective method was started. Periodically 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 (2) (see section 7.2). If condition (2) was met, the multiple objective method was stopped and the effectiveness index was calculated. In other words, the multiple objective method was stopped if the set of potentially Pareto-optimal solutions was giving the same average quality of approximation over the scalarizing functions as optimization of the functions with the single objective metaheuristic. Another stopping criterion of the multiple objective method was the 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.

Table 7. Results of GLS – MOGLS comparison. Standard deviations are given in brackets

(Temporary) population size	Average running time of GLS (thousands of functions evaluations)	Running time of MOGLS (thousands of functions evaluations)	Effectiveness index (ratio of running time: MOGLS to GLS)	Number of potentially Pareto-optimal solutions generated by MOGLS
3 objectives, 50 cities				
10	614 (33)	9 676 (3 044)	**15.69 (4.51)**	1003 (235)
20	1 186 (57)	18 164 (13 116)	**15.12 (10.38)**	1573 (406)
30	1 676 (107)	19 408 (2 315)	**11.64(1.70)**	1790 (219)
3 objectives, 100 cities				
10	4 912 (238)	187 880 (91 282)	**38.63 (19.42)**	2726 (921)
20	10 111 (515)	357 767 (190 909)	**35.84 (20.29)**	4837 (1886)
30	14 838 (790)	557 800 (165 688)	**37.89 (12.26)**	6547 (1800)
4 objectives, 50 cities				
10	680 (26)	57 337 (11 756)	**84.55 (19.16)**	8785 (2038)
20	1 183 (43)	96 166 (19 870)	**81.05 (15.16)**	14267 (3199)
30	1 751 (79)	116 370 (12 193)	**66.44 (5.89)**	16602 (1804)

Three different sizes of (temporary elite) population were used (see Table 7). In each case the same size was used in single objective GLS and MOGLS. Single objective GLS used the adaptive stopping rule described in section 6.1. Single objective GA used population of size 50 and was allowed to perform 200 generations. Pareto GA used population of size 100.

The results of the evaluation of MOGLS vs. GLS are presented in Table 7. The entries in the table contain averages and standard deviations for ten experiments on three objective instances and five experiments on four objective instances (one experiment for each instance). As 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 with the growth of the size of the set of potentially Pareto-optimal solutions.

Significantly different results were obtained in the case of comparison of GA and Pareto GA. In none of the runs the set of potentially Pareto-optimal solutions generated by Pareto GA fulfilled the condition (2) in running time lower or equal to 500 times the average running time of GA. Thus, we conclude that the effectiveness index of Pareto in comparison to GA on the TSP instances used in the experiment is greater than 500. For one instance of each size we have also allowed Pareto GA to perform 5000 in a single run. Neither in this case the method was able to fulfill the condition (2).

In our opinion, the results prove that the off-line generation of approximately Pareto-optimal solutions with the use of MOGLS is competitive from the computational effectiveness point of view to generation of the solutions with single objective GLS. The values of the effectiveness index are within the range 11.64-84.55. It is quite likely, that a DM could need presentation of a higher number of solutions while searching for the best compromise with an interactive procedure, especially in the case of methods that like LBS present samples of solutions in each iteration. In this case the overall computational requirements of MOGLS could be lower than those of single objective GLS. Note, however, that the computational effectiveness is not the only reason for using the two-phase approach with off-line generation of approximately Pareto-optimal solutions (see section 7.1). For example, in many situations the DM's time saved by the two-phase approach is much more valuable than the CPU time.

9.3. The performance of multiple objective genetic local search and Pareto simulated annealing on the 0/1 multiple knapsack problem

Zitzler and Thiele [188] performed a comparative study of five different multiple objective evolutionary algorithms on the multiple objective knapsack problem (MOKP). They compared four well known multiple objective evolutionary algorithms – Vector Evaluated Genetic Algorithm [144], Haleja's and Lin's Genetic Algorithm [49], Niched Pareto Genetic Algorithm [67] and Nondominated Sorting Genetic Algorithm [152]. On the basis of the experiment the authors proposed a new method called the Strength Pareto Evolutionary Algorithm (SPEA) that outperformed the other methods.

Recently Knowles and Corne continued the above experiment using the same set of test instances [101], [102]. They compared SPEA with M-PAES a multiple objective memetic algorithm that hybridizes local search with recombination operators. Their results indicate that M-PAES performs better than SPEA.

In this study, we compare seven algorithms: SPEA – the best performer in the Zitzler's and Thiele's experiment [188], M-PAES – the best performer in Knowles's and Corne's experiments [101], [102], Ishibuchi's and Murata's multiple objective genetic local search (IMMOGLS) [75] the multiple objective genetic local search algorithm proposed by us (MOGLS) (see section 6), Serafini's multiple objective simulated annealing (SMOSA) [146], multiple objective simulated annealing proposed by Ulungu et al. (MOSA) [170] and Pareto simulated annealing (PSA) proposed by us (see section 5). We use the same set of MOKP instances and similar performance measures to those used by Zitzler and Thiele [188]. We adapt our MOGLS and Ishibuchi's and Murata's MOGLS to MOKP in the same way as Zitzler and Thiele [188], i.e. we use the same recombination, mutation and repair operators. The only difference is that we order the elements removed by the greedy repair procedure on the basis of a weighted scalarizing function value. Analogously, we adapt the simulated annealing algorithms using the same neighborhood as Knowles and Corne [101], [102]. Again, the only difference is that we order the elements removed by the greedy repair procedure on the basis of a weighted scalarizing function value.

PSA and Serafini's multiple objective simulated annealing have already been tested on the multiple objective knapsack problem [26]. In that experiment, however, a different version of the problem with a single knapsack was used.

Note that the goal of the work is not to develop the most effective method for MOKP, but to perform a fair comparison of MOGLS and PSA algorithms with other well known multiple objective metaheuristics. More effective algorithms may be obtained by exploiting special features of MOKP in adaptation of the methods. A very effective tabu search procedure for bi-objective single knapsack problem has been proposed recently by Gandibleux and Freville [43].

The experiment described in this section is an extended and updated version of the experiment reported in [84].

9.3.1. Methods used in the experiment

Strength Pareto Evolutionary Algorithm (SPEA)

The Strength Pareto Evolutionary Algorithm (SPEA) is a relatively new multiple objective metaheuristic proposed by Zitzler and Thiele [188] (see also section 3.1.2). According to the authors, the method combines the best features of other multiple objective genetic algorithms with some new techniques. The method was the best performer in experiment run by the authors [188].

The main features of SPEA may be summarized as follows:

- It assigns the fitness to the solutions on the basis of the dominance relation.
- It stores the potentially Pareto-optimal solutions found so far in a secondary population (set of potentially Pareto-optimal solutions).
- It performs clustering of the potentially Pareto-optimal solutions in order to reduce the number of solutions stored.
- All potentially Pareto-optimal solutions stored in the secondary population participate in selection.
- It uses a new Pareto-based niching method to preserve diversity of the population.

In the experiment described in [188] Zitzler and Thiele used the tournament selection with replacement.

Memetic-Pareto Archive Evolution Strategy (M-PAES)

Memetic-Pareto Archive Evolution Strategy (M-PAES) was proposed by Knowles and Corne [101] (see also section 3.3). The method hybridizes local search and recombination operators. Unlike in our algorithm the local search is not guided by any function. The acceptance of a new neighborhood solution depends on its comparison with the current solution and with a local archive of solutions. The comparison is made only on the basis of dominance relation. Recombination operator is applied periodically to distinct local optima.

Recently Knowles and Corne [102] performed a comparative experiment on the same set of MOKP instances as used by Zitzler and Thiele [188]. They implemented their own version of our MOGLS algorithm and concluded that it was outperformed by M-PAES. We make the following remarks with respect to that experiment:

- Knowles and Corne set the size of *CS* to 100. This assured that the number of solutions stored by MOGLS in this set was equal to the number of solutions stored by M-PAES in the external nondominated set. However, this is a serious mistake from the point of view of our method because the size of *CS* is in fact not a parameter but should be calculated as $K \times S$. Since the authors use K=20 and S=100, the size of *CS* should be equal to 2000. Set *CS* of size 100 does not guarantee that the temporary elite populations really contain very good solutions on the current scalarizing function.

- The authors adapted our method to MOKP in a different way than that used in this paper. In particular they used a neighborhood operator and iterative improvement not used in our adaptation of MOGLS.

Our version of MOGLS gives much better results than the version implemented by Knowles and Corne. It can be caused by the error in setting the size of *CS*, as well as, by the use of different operators. It is yet to be determined which of the elements is the main source of the difference in the quality of the results.

The numbers of functions evaluations performed by particular algorithms in the experiment of Knowles and Corne were the same as in the experiment of Zitzler and Thiele [188].

Ishibuchi's and Murata's Multiple Objective Genetic Local Search (IMMOGLS)

The method is described in section 3.3.

Serafini's multiple objective simulated annealing (SMOSA)

The method is described in section 3.2.1.

Multiple objective simulated annealing proposed by Ulungu et al. (MOSA)

The method is described in section 3.2.1. The main parameter of the algorithm is the set of the predefined weight vectors. Since the authors do not give detailed guidelines for setting the set we used the following approach. At first 500 random normalized weight vectors were generated by the algorithm presented in Figure 18. Then a well-dispersed sample of weight vectors was filtered by the approach proposed by Törn [165]. The size of the sample was selected experimentally for each of the instances.

9.3.2. Experiment design

As it was mentioned above the experiment described in this paper is a continuation of the experiments performed by Zitzler and Thiele [188] and Knowles and Corne [101], [102]. We use the same problem and the same instances. We compare the approximations of the nondominated sets generated by MOGLS, IMMOGLS, SMOSA, MOSA and PSA with the results of SPEA and M-PAES. We use a similar way of evaluating the approximations of the nondominated sets. The problem instances and the approximations of the nondominated sets generated by SPEA are available on the Internet [71]. The results of M-PAES are available upon request by email from j.d.knowles@reading.ac.uk. The results of our experiment are also available at: www-idss.cs.put.poznan.pl./~jaszkiewicz/mokp/.

The multiple objective 0/1 knapsack problem

In this study we use the *multiple objective 0/1 knapsack problem* in the same version as used by Zitzler and Thiele in their experiment [188]. The J-objective objective 0/1 knapsack is defined by a set of I items. With each of the items J profits and J weights are associated. Furthermore, J constraints corresponding to the knapsack capacities are imposed. Formally the problem is defined in the following way: Given a set of I items and a set of J knapsacks, with:

p_{ji} – profit of item i according to knapsack j,

w_{ji} – weight of item i according to knapsack j,

c_j – capacity of knapsack j,

find a vector $\mathbf{x} = \{x_1,...,x_I\} \in \{0, 1\}^I$, that maximizes

$$f_j(\mathbf{x}) = \sum_{i=1}^{I} p_{ji} x_i, \quad j = 1,...,J \ ,$$

s. t.

$$\sum_{i=1}^{I} w_{ji} x_i \le c_j, \qquad j = 1,...,J \ .$$

We use the same set of 9 instances of the problem that was used by Zitzler and Thiele [188]. The instances have 2, 3 and 4 objectives, and 250, 500 and 750 items. The instances were generated randomly with uncorrelated profits and weights. The capacities of knapsacks are equal to half the total weight of the items. In result, about 50% of items are expected to be selected in the Pareto-optimal solutions.

Allowing variables x_i, i=1,..,I, to take continuous values in <0,1> one obtains the *relaxed multiple objective 0/1 knapsack problem*. Note that all feasible solutions of the multiple objective 0/1 knapsack problem are also feasible solutions of the corresponding relaxed problem.

Adaptation of the algorithms to the multiple objective 0/1 knapsack problem

Like Zitzler and Thiele [188] we base the adaptation of MOGLS to the multiple objective 0/1 knapsack problem on the work of Michalewicz and Arabas [121] for single objective case. The solution is naturally encoded by a binary string $\mathbf{x} \in \{0, 1\}^I$. Standard one-point crossover operator is used. Since the crossover operator may produce many unfeasible codings Michalewicz and Arabas [121] proposed a *greedy repair procedure* for single objective knapsack problem. The procedure removes items from an unfeasible solution step by step. The items are removed in the increasing order of the profit/weight ratio. In other words, the method removes items that locally decrease the weight in the knapsack at the lowest cost of the objective function. Note that the method of Michalewicz and Arabas is in fact a hybrid GA as the repair procedure is guided by the value of the objective function – profit.

Since SPEA uses dominance based fitness assignment instead of an objective function, Zitzler and Thiele [188] decided to sort the removed items in the increasing order of the maximum profit/weigh ratio over all knapsacks. There is no warranty, however, that this approach decreases the weights at the lowest cost of the solution fitness. The same approach was used by Knowles and Corne in their experiments [101], [102]. In our MOGLS as well as in Ishibuchi's and Murata's MOGLS and the simulated annealing algorithms an obvious choice is to follow the original proposition of Michalewicz and Arabas replacing the item’s profit by the weighted sum of item’s profits. The weighted sum of item’s profits is equal to the change of the weighted linear scalarizing function obtained by removing the item. Thus, in the case of the methods we sort the items according to the following ratio:

$$\frac{\sum_{j=1}^{J} \lambda_j p_{ji}}{\sum_{j=1}^{J} w_{ji}},$$

where λ_j's, j=1,..,J, are elements of the weight vector used in the current iteration. In other words, the procedure removes items that locally decrease the weights in the knapsacks at the lowest cost of the current scalarizing function.

In each iteration of our MOGLS and Ishibuchi's and Murata's MOGLS, the following operations were performed. At first two solutions were selected and recombined by the standard one-point crossover. Then, mutation operator was applied to the offspring. The mutation probability was set equal to 0.01, i.e. the same as in experiment of Zitzler and Thiele [188]. If the resulting solution was unfeasible the greedy repair procedure was invoked.

The initial solutions were constructed randomly by placing each item in the knapsacks with probability equal to 0.5. If the resulting solution was unfeasible the greedy repair procedure was invoked.

Note that the greedy repair procedure is the only heuristic that takes into account the value of the current scalarizing function. We did not use iterative improvement in this experiment. The main reason for that was preservation of compatibility with the experiment of Zitzler and Thiele [188]. Note also that on average the use of greedy repair procedure resulted in only 0.77 of item removals per recombination in the whole experiment. This indicates that the solutions constructed by recombination and mutation were feasible or were violating the constraints only slightly.

In the case of simulated annealing algorithms we used the neighborhood proposed by Knowles and Corne [101], [102]. A neighborhood solution is constructed by a bit mutation operator with probability 4/I. Thus, on average 4 items are added/removed from the knapsacks. Since the mutation operator may result in an unfeasible solution the greedy repair procedure described above was used in order to recover feasibility.

Quality measures

One of the quality measures used in our experiment is the coverage of two approximations of the nondominated set – Q_4 (see section 3.5.1). The measure was used by Zitzler and Thiele in their experiment [188]. As the second quality measure we use the estimation of the expected value of weighted Tchebycheff scalarizing function on approximation A over the set of normalized weight vectors – $\hat{R}$ (see section 4.2.3). The parameter k of measure $\hat{R}$ was set to 200 for bi-objective instances, to 50 for three-objective instances and to 25 for four objective instances. The resulting weight vectors may be found at www-idss.cs.put.poznan.pl./~jaszkiewicz/mokp/.

The upper approximations of the ideal points of particular instances were used as the reference points $\mathbf{z}^0$. For each instance, the components of the upper approximation of the ideal points were found by individual optimization of each objective in the relaxed multiple objective 0/1 knapsack problem. Note that all feasible solutions of the multiple objective 0/1 knapsack problem are also feasible solutions of the relaxed problem. Thus, the optimum value of each objective in the relaxed problem is not worse than the optimum value of this objective in the original discrete problem. Individual optimization of objectives in the case of the relaxed problem requires solving a relatively easy linear programming problem. In this experiment, Lingo solver was applied [69]. The values of the upper approximations of the ideal points are given in Table 8.

Table 8. The values of the upper approximations of the ideal points

Instance		Upper approximation of the ideal point
Number of objectives	Number of items	
2	250	[9898.86, 10107.3]
2	500	[20096.5, 20494.6]
2	750	[30130.1, 30037.2]
3	250	[9981.99, 10085.5, 9774.19]
3	500	[19705.8, 20082.7, 20437.2]
3	750	[31197.7, 29772.1, 30519.4]
4	250	[9801.73, 9918.88, 10119, 9136.37]
4	500	[19825.8, 19568, 19805.8, 19592.3]
4	750	[29872.5, 29342.1, 30468.6, 30025.2]

As suggested in section 4.2.3, the lower bound on measure $\hat{R}$ allows estimation of the gap between an approximation and the actual nondominated set. In order to calculate the lower bound on measure $\hat{R}$ in the case of the multiple objective 0/1 knapsack problem one can solve problem (P2) for each weighted Tchebycheff scalarizing function used in the numerical integration applying the constraints of the relaxed problem. This requires solving the following linear programming problem:

minimize α

s.t.

$$\alpha \geq \lambda_i\left(z_j^0 - z_j\right) = \lambda_i\left(z_j^0 - f_j(\mathbf{x})\right) = \lambda_i\left(z_j^0 - \sum_{i=1}^{I} p_{ji} x_i\right), \quad j = 1,\ldots,J$$

$$\sum_{i=1}^{I} w_{ji} x_i \leq c_j, \qquad j = 1,\ldots,J$$

$$0 \leq x_i \leq 1, \qquad i = 1,\ldots,I\ .$$

Since all feasible solutions of the multiple objective 0/1 knapsack problem are also feasible solutions of the corresponding relaxed problem, the optimum value of each weighted Tchebycheff scalarizing function on the relaxed problem is not greater (not worse) than the optimum value of this function on the original discrete problem. Thus, the average of the optimum value of the weighted Tchebycheff scalarizing function on the relaxed problem is not greater (not worse) than the average on the original problem for any set of weight vectors. In this experiment, Lingo solver [69] was applied. Optima of weighted Tchebycheff scalarizing functions on the relaxed problem correspond to points in the objective space. The points, in general, do not correspond to feasible solutions of the original problem. The points are, however, used in Figure 40.

In addition we apply the outperformance relations (see section 4.1) to evaluation of the approximations.

The C++ codes used to calculate the two performance measures used in this experiment are available at www-idss.cs.put.poznan.pl./~jaszkiewicz/mokp/.

Parameter settings

The number of initial solutions S in both MOGLS and IMMOGLS was set in the same way as the population size of SPEA in the experiment of Zitzler and Thiele [188] (see Table 9). In the case of IMMOGLS it also played the role of population size. The size of the temporary elite population used by MOGLS was set equal to 20, i.e. for each randomly selected scalarizing function s two recombined solutions were selected from 20 solutions being the best on s. In both MOGLS and IMMOGLS the calculations were stopped after $500\times S$ iterations. In other words, the number of recombinations corresponded to 500 generations of SPEA, i.e. the number of generations used by Zitzler and Thiele [188]. The size of the elite in IMMOGLS was equal to 10% of the population size.

Table 9. Numbers of initial solutions for different instances

Instance		Number of initial solutions
Number of objectives	Number of items	
2	250	150
2	500	200
2	750	250
3	250	200
3	500	250
3	750	300
4	250	250
4	500	300
4	750	350

The size of the generating sample in the case of PSA was set experimentally to the values presented in Table 9. Other parameters of the methods were set as follows:

- starting temperature 0.05,
- stopping condition temperature ≤ 0.0005,
- temperature changing factor 0.9,
- repulsion coefficient 0.001.

The number of the weight vectors used by MOSA was also set experimentally to the values presented in 0. In all the simulated annealing algorithms the temperature was decreased after performing a given number of moves on a temperature plateau. The number of moves on a temperature plateau was set in order to assure the same number of functions evaluations as in the case of the evolutionary algorithms. Note that in the case of MOSA and PSA the number of functions evaluations is the sum of the numbers of moves performed by all the solutions. Thus, each solution in MOSA and PSA was allowed to perform fewer moves than the single solution in the case of SMOSA.

Table 10. Numbers of generating solutions and predefined weight vectors in PSA and MOSA

Instance		Number of generating solutions (PSA)	Number of predefined weight vectors (MOSA)
Number of objectives	Number of items		
2	250	10	20
2	500	12	24
2	750	15	30
3	250	20	40
3	500	25	50
3	750	30	60
4	250	30	60
4	500	40	80
4	750	50	100

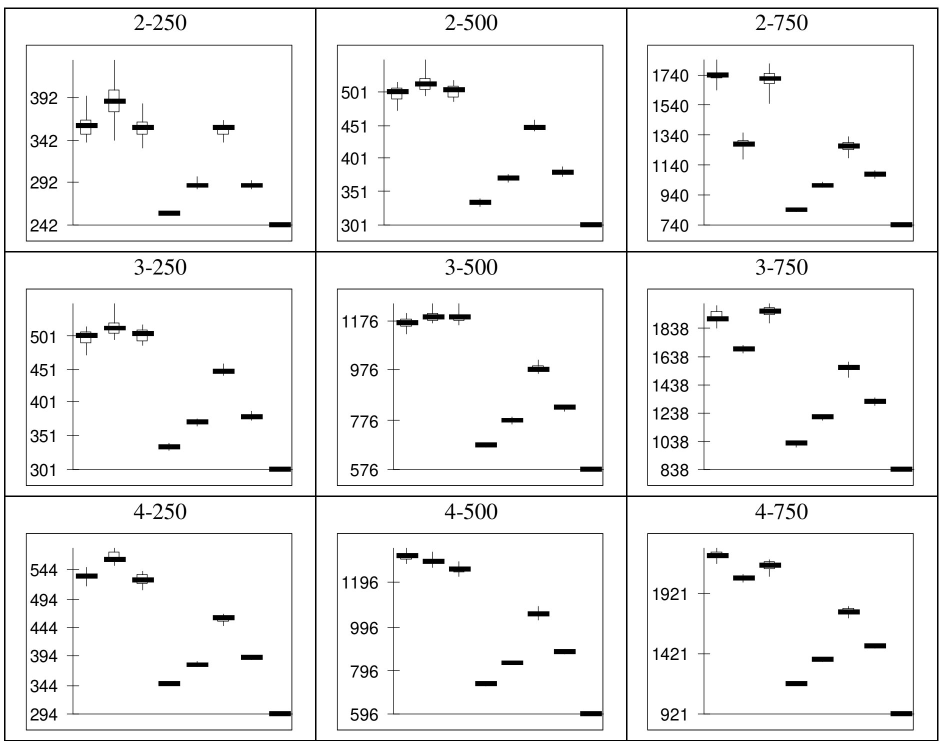

Figure 34. Results of the comparison using measure $\hat{R}$. Each chart corresponds to a different instance. Each chart contains eight box plots representing the distribution of $\hat{R}$ for (from left to right) IMMOGLS, SPEA, M-PAES, MOGLS, PSA, SMOSA, MOSA and the lower bound. For each chart the scale is from the lower bound at the bottom to the highest observed value at the top. Note that the box plots are in many cases practically invisible because of the very low dispersion of the results

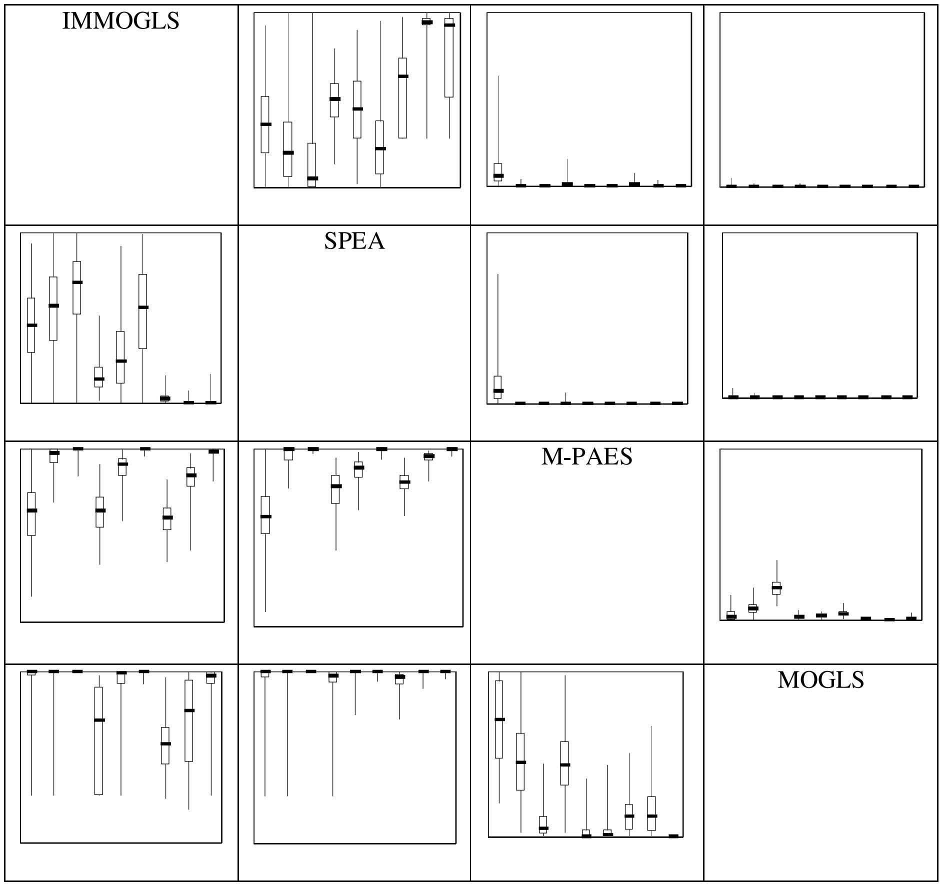

Figure 35. Results of the comparison of evolutionary algorithms using measure Q_4. Each chart contains nine box plots representing the distribution of Q_4 values for a certain ordered pair of algorithms; the three box plots to the left relate to 2 knapsacks and (from left to right) 250, 500, and 750 items; correspondingly the three middle box plots relate to 3 knapsacks and the three to the right to 4 knapsacks. The scale is 0 at the bottom and 1 at the top for each chart. Chart in row of algorithm *A1* and column of algorithm *A2* presents values of coverage of approximations generated by *A2* by approximations generated by *A1*

9.3.3. Results

The results of the evaluation using measure $\hat{R}$ are presented in Figure 34. The results indicate that our MOGLS algorithm generates the best approximations. Furthermore the gap between the quality of approximations generated by MOGLS and the lower bound is much lower than the gap between results of MOGLS and results of the other methods except for PSA. PSA is the second performer in the experiment. MOSA gives comparable results only on the smallest instance with 2 objectives and 250 items. SMOSA is the worst method among simulated annealing algorithms. According to measure $\hat{R}$ all simulated annealing algorithms outperform the other evolutionary algorithms (i.e. all evolutionary algorithms but MOGLS).

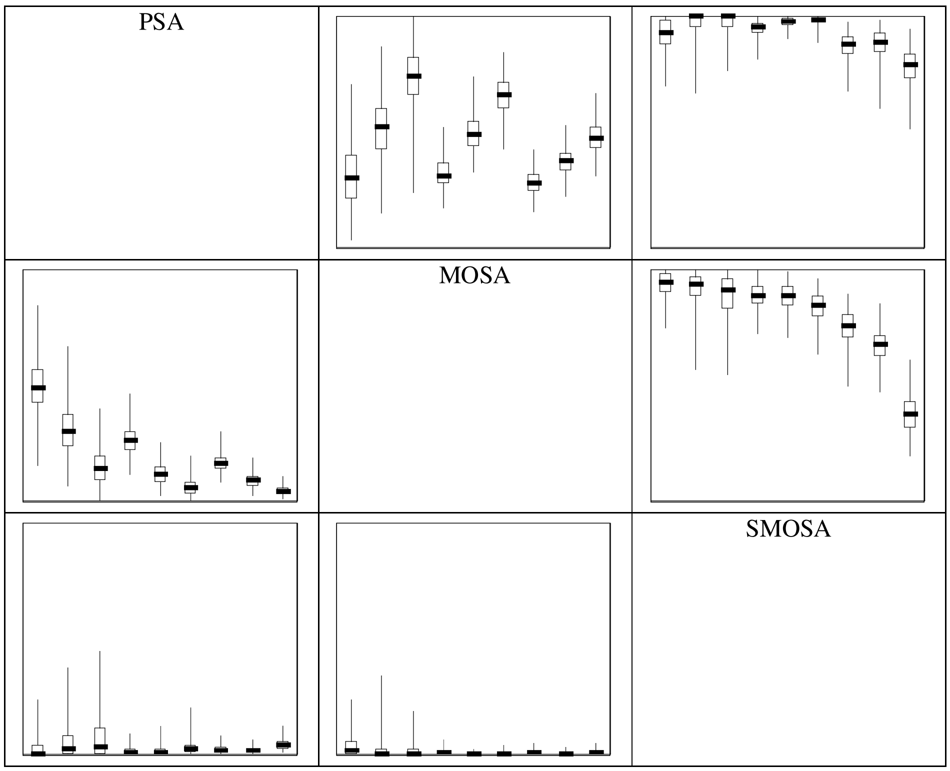

Figure 36. Results of the comparison of simulated annealing algorithms using measure Q_4. For the description of the figure layout see Figure 35

The results of the evaluation of evolutionary algorithms with the coverage measure Q_4 are presented in Figure 35. A simple version of box plots is used. The thick line represents the median value, the upper and lower ends of the box are the upper and lower quartiles, and the ends of the vertical line are the minimum and maximum value. According to measure Q_4 MOGLS and M-PAES outperform SPEA and IMMOGLS. The comparison of the former methods is less obvious. The points generated by M-PAES cover a small percentage of the points generated by MOGLS. On the other hand the points generated by MOGLS also cover a rather small portion of the points generated by M-PAES, especially for the instances with four objectives. More detailed analysis of the generated approximations (see e.g. Figure 40, Table 11 and Table 12) indicates that M-PAES generates very good solutions in a small middle region of the nondominated set but generates no solutions in other regions of the set. MOGLS covers all regions of the nondominated set. Since the points generated by M-PAES in the middle region are very good they cover some points generated by MOGLS located in this region. The covered points are, however, a small portion of all the points generated by MOGLS. The points generated by MOGLS outside the middle region of the nondominated set, in general, do not cover points generated by M-PAES just because the latter method generates no points in those regions. In our opinion, the evaluation using measure $\hat{R}$ is much more concordant with intuitive evaluation of the approximations.

The results of the evaluation of simulated annealing algorithms with the coverage measure Q_4 are presented in Figure 36. This comparison allows to draw similar conclusions as in the case of comparison using measure $\hat{R}$. The results allow to evaluate SMOSA as the worst method and PSA as the best. The relative performance of PSA with respect to MOSA improves with growing dimensionality of the instances.

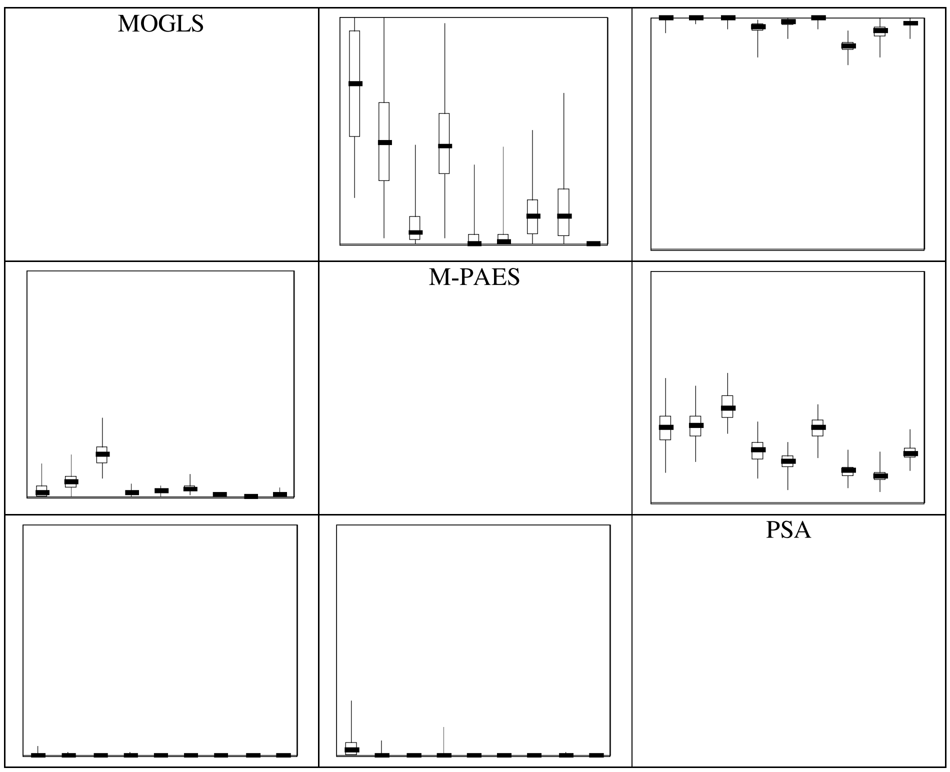

Figure 37. Results of the comparison of MOGLS, M-PAES and PSA algorithms using measure Q_4. For the description of the figure layout see Figure 35

In Figure 37, three best methods in particular experiments are compared using measure Q_4. M-PAES was the best method in the former experiments of Zitzler and Thiele [188] and Knowles and Corne [101], [102]. MOGLS is the best method among evolutionary algorithms and PSA among simulated annealing algorithms. The results indicate better performance of MOGLS than PSA. The comparison of PSA and M-MPAES is similar to the comparison of MOGLS and M-PAES with the same measure. M-PAES generates better solutions than PSA in the middle region but the solutions generated by PSA represent all regions of the nondominated set.

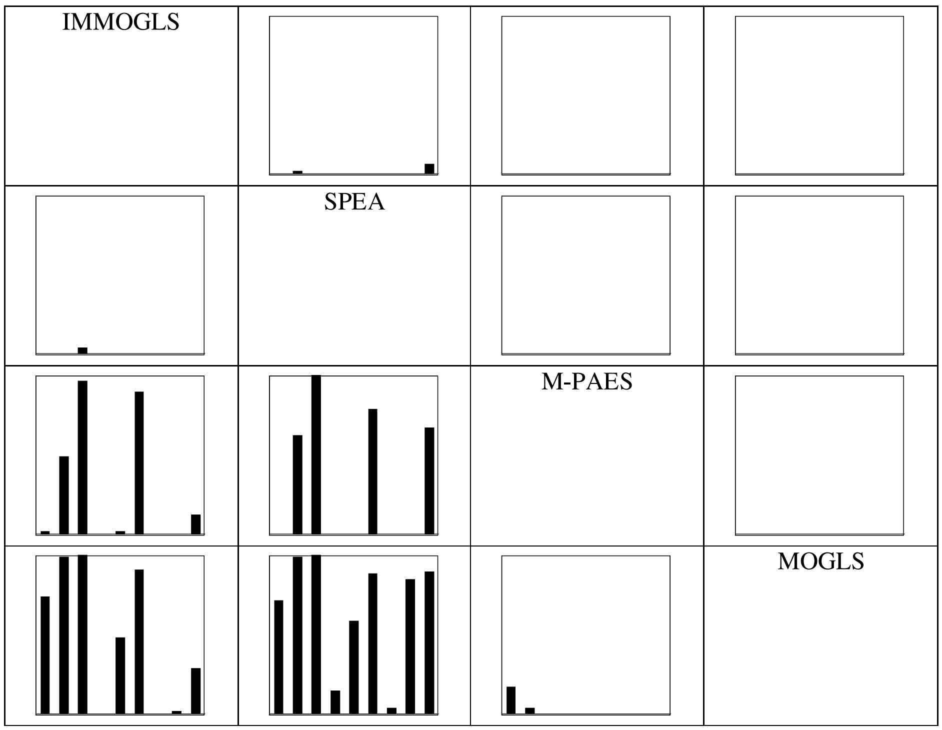

Figure 38. Results of the comparison of evolutionary algorithms using the outperformance relations. Each chart contains nine box plots representing the percentages of occurrence of the outperformance relations among 400 pairs of approximations generated by certain ordered pair of algorithms; the three box plots to the left relate to 2 knapsacks and (from left to right) 250, 500, and 750 items; correspondingly the three middle box plots relate to 3 knapsacks and the three to the right to 4 knapsacks. The scale is 0% at the bottom and 100% at the top for each chart. Chart in row of algorithm *A1* and column of algorithm *A2* presents values of coverage of approximations generated by *A2* by approximations generated by *A1*

Figure 38 and Figure 39 present the results of evaluation with the outperformance relations. An interesting observation is that in all cases (64800 comparisons of pairs of approximations of the nondominated set) either all the three outperformance relations occurred together or none of them occurred. Thus, it is enough to present a single value for each instance and each pair of algorithms. These results suggest that a situation when an approximation weakly or strongly outperforms another approximation but does not completely outperform the other approximation is very unlikely in practice. Some experiments on other problems not reported in this work resulted in the same observation. In general, evaluation with the outperformance relations allows to draw similar conclusions as in the case of coverage measure. This is a natural consequence of the connections between the outperformance relations and the coverage measure (see section 4.1).

In general, all the evolutionary algorithms but MOGLS generated points only in the middle region of the nondominated set (see Table 11). The same applies to other methods tested in the experiment of Zitzler and Thiele [188]. This indicates that the mechanisms of the well-known multiple objective evolutionary algorithms are not effective enough to assure dispersion of the generated points over the whole nondominated set of MOKP. In our opinion, the main reason for this fact is the genetic drift suggested by Fonseca and Fleming [36]. They argue that recombination operator may cause drift of a finite genetic population to a small region of the nondominated set. Note that our MOGLS algorithm does not recombine solutions coming from distant regions of the nondominated set because they are unlikely to be placed in a single temporary elite population. This may be interpreted as a form of mating restrictions suggested in [36].

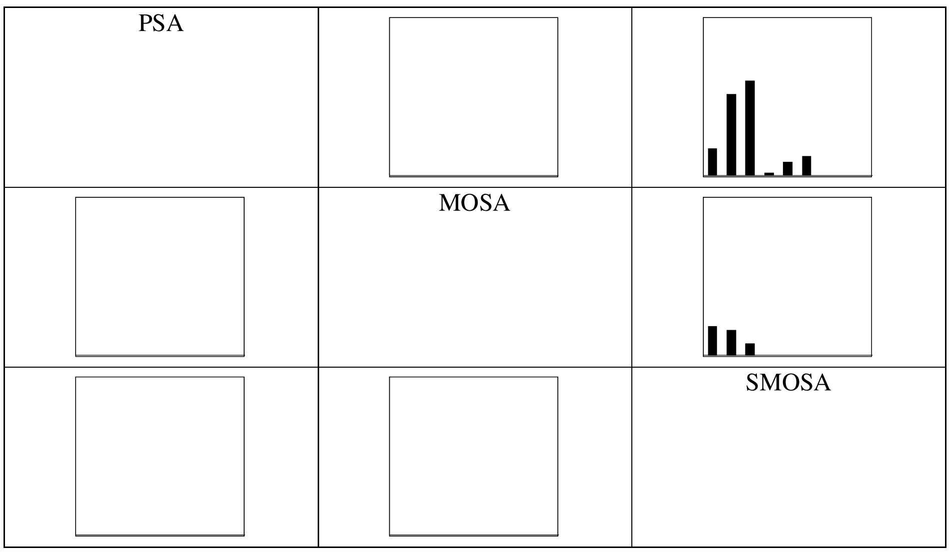

Figure 39. Results of the comparison of simulated annealing algorithms with the outperformance relations. For the description of the figure layout see Figure 38

MOGLS and the simulated annealing algorithms used a slightly different version of the greedy repair procedure than SPEA and M-PAES because it used a different way of ordering the items. The same procedure was, however, used in IMMOGLS whose performance was one of the worst. Thus, this single element cannot explain the good performance of the other methods.

9.3.4. Conclusions

Taking into account the former experiments of Zitzler and Thiele [188] and Knowles and Corne [101], [102] twelve multiple objective metaheuristics have been tested on the same set of instances of the multiple objective 0/1 knapsack problem. All the methods have been adapted to the problem in a compatible way, i.e. they used (if necessary in a given method) the same recombination operator, neighborhood and repair procedure. All the methods were allowed to perform the same number of functions evaluations.

The two methods described in this work: multiple objective genetic local search (MOGLS) and Pareto simulated annealing (PSA), were the best performers in the experiments. The former method gave the best results. In our opinion, the good performance of MOGLS is due to the global convexity of the multiple objective 0/1 knapsack problem. Although, no experimental tests of the global convexity of the problem have been performed it is intuitively clear that good solutions of the problem being close in the objective space should have many common items. Thus, recombination operator joining good properties of two already good solutions may effectively produce new good solutions.

The results of the experiment indicate that evaluation of the intermediate solutions on the basis of dominance relation only does not introduce selection pressure ensuring good dispersion over all regions of the nondominated set. In the case of M-PAES it allowed to generate very good solutions but the solutions were concentrated in a small region of the nondominated set.

Table 11. Average ranges of objectives in approximations generated by evolutionary algorithms [minimum, maximum]

Instance	IMMOGLS	SPEA	M-PAES	MOGLS
2-250	[8520.15, 9537.85]	[8407.63, 9460.87]	[8742.5, 9473.05]	[7332.55, 9883.9]
	[8614.15, 9692.9]	[8809.47, 9747.23]	[8866.95, 9593]	[7747.65, 10093.2]
2-500	[17684.7, 19047.5]	[17697.5, 18900.4]	[18198.5, 19174.7]	[16148.6, 20029.2]
	[18114.7, 19459.3]	[18455.2, 19460.7]	[18541.8, 19514]	[16766.3, 20444.2]
2-750	[25902.6, 27868.5]	[26374.9, 27924]	[27100.3, 28661.8]	[23728.9, 29938.8]
	[25738.1, 27904.3]	[26152.2, 27720.1]	[26460.3, 28255.9]	[23458.2, 29883.1]
3-250	[7937.45, 9203.7]	[7891.7, 9314.17]	[8443, 9317.7]	[7104.15, 9913.7]
	[8287.45, 9415.75]	[7995.83, 9307.5]	[8463.2, 9297.4]	[7514.9, 10020.8]
	[7984.8, 9035.9]	[7864.23, 9070.1]	[8100.25, 8874.65]	[7365.75, 9715.45]
3-500	[16240.5, 17958.1]	[16074.8, 17915.7]	[17021, 17808.9]	[15205.4, 19469.4]
	[16703.4, 18305.6]	[16482, 18404]	[17643, 18348.7]	[15449.7, 19810.4]
	[16635.4, 18420.5]	[16476.5, 18512.9]	[17556.5, 18357.5]	[15424.4, 20227.8]
3-750	[25893.5, 28042.7]	[25891, 28161.5]	[27135.6, 28442.8]	[24032.8, 30797.1]
	[24803, 26914.9]	[24686.6, 26860.6]	[25871.1, 27114.4]	[23248.5, 29401.1]
	[25248.7, 27319.3]	[25138.6, 27423.8]	[26417.2, 27826.8]	[23568.6, 30139.9]
4-250	[7912.9, 8980.25]	[7762.07, 9009.33]	[8232.65, 8867.5]	[7472.65, 9691.2]
	[7722.5, 8919.35]	[7542.77, 9005.93]	[8090.85, 8903.6]	[7151.15, 9825.9]
	[8096.3, 9216.6]	[7810.33, 9151.4]	[8311.7, 8951.65]	[7583.85, 10002.1]
	[7173.6, 8260.6]	[6865.8, 8230.83]	[7522.25, 8217.4]	[6629.25, 9011.85]
4-500	[15483.7, 17337.4]	[15069, 17362.7]	[16127.7, 17180.5]	[14602.3, 19449.3]
	[16001.7, 17603.2]	[15420.9, 17416.4]	[16506.2, 17361]	[15056.6, 19260.5]
	[15953.1, 17734.5]	[15242.7, 17450.8]	[16715.7, 17712.2]	[15109, 19499.4]
	[15354.7, 17138.5]	[14833.4, 17230]	[16030.5, 17104.7]	[14173.9, 19255.6]
4-750	[23848.1, 26151.2]	[23092.2, 25782.6]	[25140.8, 26054.2]	[22685.2, 29242]
	[23111.5, 25498.7]	[22567, 25306]	[24683.5, 25487.7]	[21893, 28588.4]
	[24332.5, 26628]	[23746.9, 26539.1]	[26176.7, 27048.4]	[23212.2, 29872.7]
	[23532.5, 25942.7]	[23161.1, 25972.8]	[25167.6, 26152.5]	[22279.8, 29352.6]

Table 12. Average ranges of objectives in approximations generated by simulated annealing algorithms [minimum, maximum]

Instance	PSA	SMOSA	MOSA
2-250	[7106.1, 9859.6]	[7431.65, 9803.4]	[7304.85, 9849.8]
	[7604.1, 10063]	[7843.7, 9997.4]	[7692.9, 10057.4]
2-500	[15594.2, 19968.6]	[16126.5, 19753.4]	[15795.5, 19888.9]
	[16138.6, 20348.8]	[16603.3, 20133.2]	[16329.1, 20267.1]
2-750	[22840.8, 29751]	[23894.3, 29305.8]	[22912.2, 29541.7]
	[22455, 29722]	[23347.7, 29202.7]	[22747.3, 29503.1]
3-250	[6876, 9883.7]	[7063.15, 9709.45]	[6947.7, 9875.55]
	[7197.8, 10010.9]	[7437, 9779.05]	[7234.45, 9991.7]
	[7028.15, 9690.55]	[7273.6, 9515.55]	[7092.25, 9660.15]
3-500	[14290.7, 19427.9]	[14800, 18874.4]	[14342.4, 19277]
	[14633.8, 19707.4]	[15254, 19164]	[14641, 19604]
	[14600.3, 20183.4]	[15048.9, 19452.5]	[14686.9, 20020.2]
3-750	[22812.9, 30574.7]	[23775.9, 29498.1]	[22873.5, 30197.2]
	[22068.7, 29230.6]	[22778.2, 28143.2]	[22030.7, 28825.7]
	[22351.5, 29822.2]	[23157.8, 28707.2]	[22444.6, 29574]
4-250	[7037.05, 9640.8]	[7282.85, 9378.2]	[7150.15, 9628.45]
	[6756.55, 9797.3]	[6990.7, 9470.9]	[6878.75, 9772.45]
	[7198.5, 10010.5]	[7360.15, 9689.15]	[7254.05, 9985.35]
	[6315.55, 9017.7]	[6478.4, 8715.1]	[6399.55, 8960.05]
4-500	[13695.9, 19377.8]	[14243.9, 18391]	[13860.1, 19191.4]
	[14194.8, 19109]	[14735, 18326.9]	[14318.8, 18974.2]
	[14102, 19394]	[14532.5, 18531.9]	[14221.6, 19244.2]
	[13427.3, 19199.2]	[14005.9, 18105]	[13598.3, 18981.6]
4-750	[21277.3, 29021.7]	[22254.2, 27374.7]	[21317.1, 28513.7]
	[20553.9, 28348.4]	[21455.3, 26753.7]	[20532.1, 27857.1]
	[21673.4, 29430.1]	[22604.4, 27911.2]	[21818.4, 29099]
	[20849.1, 29041.2]	[21854.6, 27291.3]	[21082.5, 28670.7]

On the other hand, the use of weighted scalarizing functions with varying weight vectors allows to obtain a good dispersion over all regions of the nondominated set. Note, however, that this single element does not guarantee a good dispersion. IMMOGLS using weighted scalarizing functions with random weight vectors produced approximations concentrated in a small region of the nondominated set.

Note that MOGLS has much higher memory requirements than the other algorithms. For example, in the case of the largest instance with four objectives and 750 items the method stored 20×350=7000 solutions in set *CS*. Thus, the good performance of the algorithm is obtained at the price of the memory usage. Taking into account the capacities of present computers this fact does not significantly limits application of MOGLS.

The approximations of the nondominated sets generated in this experiment, as well as the C++ code used to calculate the quality measures are available on the Internet at www-idss.cs.put.poznan.pl./~jaszkiewicz/mokp/.

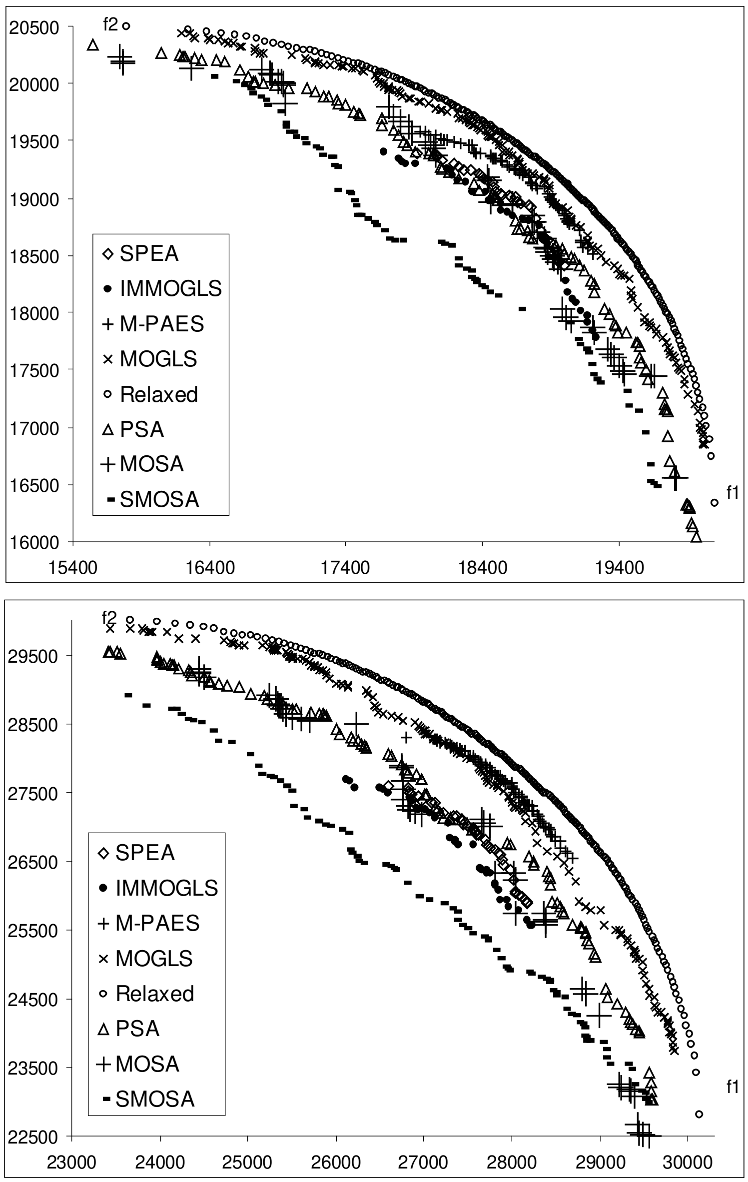

Figure 40. Nondominated points obtained for the bi-objective instances with 500 (top chart) and 750 (bottom chart) items. Each set of points was obtained by a single (first) run of a given method. The set of points denoted by "relaxed" contains unattainable points obtained by optimization of the scalarizing functions on the relaxed problem. The results for instance with 250 items are not presented because the sets of points overlap and the figure is hardly perceptible

10. Applications

This section describes some of the applications of the proposed methodology. In addition, the proposed methods have been applied to the capital budgeting [23], the design of a cellular manufacturing system [25] and to the agricultural project scheduling [61].

10.1. Nurse scheduling

The goal of nurse scheduling is to determine for each day in the scheduling period the assignment of working shifts to particular nurses. The use of computer systems in personnel scheduling was proposed already in sixties [68]. Computer-assisted nurse scheduling systems are used in order to:

- reduce the workload necessary to build the schedules manually,
- minimize personnel requirements,
- satisfy preferences of the nurses,
- balance the workload of particular nurses.

Such systems are usually based on either operational research models or artificial intelligence models [95]. In the first case the nurse scheduling problems are defined as (usually single objective) discrete mathematical programming problems. The problems can be solved by exact methods (see e.g. [129]) or heuristic methods (see e.g. [11]). Because of usually high computational complexity of such problems the heuristic approaches seem to be especially promising in practice. The artificial intelligence models include knowledge-based systems (see e.g. [116]) and constraint-based programming systems (see e.g. [27]). Such approaches are especially useful when the constraints defining feasible schedules are tight and the main difficulty is in finding schedules satisfying all of them.

A nurse scheduling system has to be adapted to the local conditions, i.e. legal regulations and traditions on the country, region and even hospital level. Polish regulations do not imply many constraints on the schedules. In result, the employer has a lot of flexibility in formulating the schedules and does not need to increase the number of nurses in order to satisfy the constraints. The schedules, however, are usually poor from the personnel's point of view. They include very long and very short (even single day) working periods, different working shifts in consecutive days and single days-off. Such situation is not only uncomfortable for the nurses, it also deteriorates quality of the work and is one of the reasons of relatively low personnel stability in Polish hospitals.

The main aim of the application described in this section is to propose schedules that will improve conditions of the work and balance the workload of particular persons without increasing the staff requirements. The problem is formulated in terms of multiple objective combinatorial optimization. In order to solve it the two-phase approach was used (see section 7.1). Pareto simulated annealing was used in the first phase and the interactive trichotomy segmentation was used in the second phase.

10.1.1. Description of the nurse scheduling problem

The nurse scheduling problem described in this paper was formulated for one of Poznań's hospitals (Szpital Przemienienia Pańskiego) [78]. The data of the surgery unit are used. In this case, the role of the decision maker plays the head nurse of the unit.

The scheduling period is one month. The typical staff in a single unit is about 20 nurses. There are three different shifts: morning, afternoon and night shifts. It was agreed that the following goals would be used in the optimization process:

- the working days should be grouped in periods of desired length, e.g. five days (goal 1),
- the nurses should work on the same shift in consecutive working days (goal 2),
- the days-off should be grouped together (goal 3),
- the numbers of morning, afternoon and night shifts assigned to particular nurses should be balanced (goal 4),
- the surplus of nurses assigned to the morning shifts on week-days should be balanced among different days in the scheduling period (goal 5).

The first three goals tend to make the working pattern in the hospital more similar to that of other organizations. The fourth goal assures equity among nurses. The goal 5 is related to the fact that number of available person-days in a given unit may be greater than the requirements. In this case it is possible to have a surplus of nurses on morning shifts on week-days.

The input data defining the problem are:

- the length of the scheduling period,
- the numbers of nurses required on particular shifts defined for each day in the scheduling period,
- the number of available nurses,
- types of contracts of particular nurses (e.g. full-time, fifty per cent time),
- information about a priori known absences of particular nurses (e.g. holidays),
- wishes of particular nurses.

As it was mentioned above, Polish legal regulations do not imply many constraints on the schedules. The regulations state that:

- the number of consecutive working days cannot be greater than eight,
- after a night shift a person cannot work on a morning or afternoon shift on the next day,
- at least one of the consecutive three Sundays has to be free.

The above regulations are used as constraints.

Other constraints used in the system are:

- the number of working days in the scheduling period assigned to each nurse has to be equal to the number of working days resulting from her/his type of contract,
- the requirements for numbers of nurses working on particular shifts on particular days have to be satisfied.

Each nurse can state a specified by the head nurse number of wishes. A wish consists in stating that on a given day the nurse wants to work on particular shift or have a day-off. The wishes can be taken into account as hard constraints or neglected. The head nurse may also decide to take into account some of the wishes only.

Decision variables

Let i=1,...,I be the index of a nurse. Let j=1,...,J be the index of a day in the scheduling period. A schedule is defined by the following decision variables:

$$x_{ij} = \begin{cases} m & \text{if the } i\text{-th nurse works on the morning shift on } j\text{-th day} \\ a & \text{if the } i\text{-th nurse works on the afternoon shift on } j\text{-th day} \\ n & \text{if the } i\text{-th nurse works on the night shift on } j\text{-th day} \\ f & \text{if the } i\text{-th nurse has the day-off on } j\text{-th day} \\ b & \text{if the } i\text{-th nurse is absent on } j\text{-th day} \end{cases}$$

Objectives

In order to achieve mentioned above goals several objective functions are defined. Each of the functions corresponds to one of the goals.

Goal 1

The objective function f_1 corresponding to the goal 1 measures the average deviation of the length of each working period (composed of consecutive working days) from its desired length. Let l=1,...,L be the index of a working period that appears in the schedule. Let C_l be the length (number of days) of l-th period. This objective is formulated in the following way:

$$f_1 = \frac{\sum_{l=1}^{L} |C_l - C|}{L},$$

where C is the desired length of a working period. This objective should be minimized. It achieves value zero if all the working periods have the desired length. The data from the previous period are taken into account while calculating lengths of the working periods.

Goal 2

The objective function f_2 corresponding to goal 2 measures the average number of, so called, changes, i.e. situations when in two consecutive days a person works on different shifts. This objective is formulated in the following way:

$$f_2 = \frac{\sum_{i=1}^{I}\sum_{j=1}^{J} \begin{cases} 1 & \text{if } x_{ij}, x_{ij-1} \in \{m,a,n\} \text{ and } x_{ij} \neq x_{ij-1} \\ 0 & \text{otherwise} \end{cases}}{I}.$$

This objective should be minimized. It achieves value zero if there are no changes in the scheduling period. The data from the previous period are taken into account while calculating value of this objective.

Goal 3

The objective function f_3 corresponding to goal 3 measures the average number of days in a free period. Let k=1,...,K be the index of a free period that appears in the schedule. Let F_k be the length (number of days) of k-th period. This objective is formulated in the following way:

$$f_3 = \frac{\sum_{k=1}^{K} F_k}{K}.$$

This objective should be maximized.

Goal 4

The objective function f_4 corresponding to goal 4 measures the average deviation of the percentage of shift types assigned to particular nurses from the average number of shift types assigned to a single person. Let av_s, $s=\{m,a,n\}$, be the average percentage of shifts of type s in the scheduling period:

$$av_s = \frac{\sum_{i=1}^{I}\sum_{j=1}^{J} x_{sij}}{\sum_{s\in\{m,a,n\}}\sum_{i=1}^{J}\sum_{j=1}^{J} x_{sij}} \times 100\% \qquad s \in \{m,a,n\}.$$

Where:

$$x_{sij} = \begin{cases} 1 & \text{if } x_{ij} = s \\ 0 & \text{otherwise} \end{cases} \quad s = \{m,a,n\}, i = 1,...,I, \; j = 1,...,J \; .$$

This objective is formulated in the following way:

$$f_4 = \frac{\sum_{i=1}^{I}\left(\sum_{s\in\{m,a,n\}} \left| av_s - \frac{\sum_{j=1}^{J} x_{sij}}{\sum_{s\in\{m,a,n\}}\sum_{j=1}^{J} x_{sij}} \times 100\% \right| \right)}{I},$$

This objective should be minimized. It achieves value zero if the types of working shifts are completely balanced among the nurses.

Goal 5

The objective function f_5 corresponding to goal 5 measures the average deviation of the nurses' surplus on particular days from the average surplus. Let r_{mj}, j=1,...,J, be the required number of nurses on the morning shift on the j-th day. Let as be the average surplus of nurses on morning shifts in the scheduling period:

$$as = \frac{\sum_{j\in P}\left(\sum_{i=1}^{I} x_{mij} - r_{mj} \right)}{|P|},$$

where P is the set of all weekdays in the scheduling period, i.e. the set of Mondays, Tuesdays, Wednesdays, Thursdays and Fridays.

This objective is formulated in the following way:

$$f_5 = \frac{\sum_{j\in P} \left| \left(\sum_{i=1}^{I} x_{mij} - r_{mj} \right) - as \right|}{|P|}.$$

This objective should be minimized. It achieves value zero if the surplus of nurses is uniformly allocated among weekdays.

The nurse scheduling problem is a routine task that has to be solved every month, so, we can assume that the decision maker's preferences do not change significantly in particular months. The objectives defined above are consistent with the goals of the optimization. Numerical values of the objectives, however, are not easy to interpret by the head nurse. For example, it may not be easy to judge if 1 day of average deviation of the length of each working period from its desired length is a good or a bad value. The head nurse needs to gain some knowledge about reasonable values of the objectives in order to be able to compare different solution and express her preferences. She can learn about the problem by analyzing the values of the objectives calculated for some past schedules that were build manually.

10.1.2. Adaptation of Pareto simulated annealing to the nurse scheduling problem

The starting feasible solution is constructed by a constraint-based programming technique. Since it is a relatively time consuming approach the starting generating sample is composed of a number of equal solutions. Since PSA is a probabilistic method, after some iterations the generating solutions become significantly different.

Three different elementary types of moves are used. Each of them transforms a feasible solution to another feasible solution. One of the types is selected randomly with equal probability at each step of the procedure.

Move 1

In the move of the first type the assignments of working shifts or free-days in two randomly selected days are exchanged between two randomly selected nurses. Let $i', i'' \in \{1,...,I\}$ be the indices of the selected nurses and $j', j'' \in \{1,...,J\}$ be the indices of selected days. The selection has to be performed in a way that assures that the following constraint is satisfied:

$$x_{i'j'} \neq x_{i'j''} \wedge x_{i''j'} \neq x_{i''j''} \wedge x_{i'j'} = x_{i''j''} \wedge x_{i'j''} = x_{i''j'},$$

i.e. each of the nurses has different assignments on the selected days and the assignment of each nurse on the first day is equal to the assignment of the other nurse on the second day. This constraint assures that the requirements will be satisfied after this move. Before performing this move it is also necessary to check if it would not violate some other constraints, e.g. if the number of consecutive working days would not greater than eight. If some constraints are violated another two nurses and another two days are selected randomly.

This move can modify the values of objectives f_1, f_2 and f_3. It does not change the values of objectives f_4 and f_5.

Move 2

In the move of the second type the assignments of working shifts on a randomly selected day are exchanged between two randomly selected nurses. The two persons cannot have free-days on the selected day and they must have different assignments on the selected day.

This move can modify the values of objectives f_2 and f_4. It does not change the values of objectives f_1, f_3 and f_5.

Table 13. Percentages of moves of particular types that improved, worsened or did not change values of particular objectives

Type of move	% of	f_1	f_2	f_3	f_4	f_5
Move 1	improving steps	24.2	34.7	23.1	0	0
	worsening steps	25.9	36.6	22.2	0	0
	steps without change	49.9	28.7	54.7	100	100
Move 2	improving steps	0	32.9	0	41.1	0
	worsening steps	0	37.4	0	40.0	0
	steps without change	100	29.7	100	18.9	100
Move 3	improving steps	34.1	20.2	28.1	0	23.6
	worsening steps	36.4	20.2	27.2	0	23.3
	steps without change	29.5	59.6	44.7	100	53.1
Composite move	improving steps	19.4	29.3	17.1	13.7	7.9
	worsening steps	20.8	31.4	16.5	13.3	7.8
	steps without change	59.8	39.3	66.5	73.0	84.4

Move 3

In the move of the third type the assignments of a randomly selected nurse are exchanged between two randomly selected days. On one of the days the nurse must work on morning shift and on the other day she/he must have a day-off. This move may violate the requirement that the number of consecutive working days is not greater than eight. So, violation of this constraint must be tested before performing this move. If the constraint is violated another nurse and another two days are selected randomly.

This move can modify the values of objectives f_1, f_2, f_3 and f_5. It does not change the value of objective f_4.

As it was mentioned above, particular moves cannot influence values of some objectives. In order to obtain more information about influence of particular moves on the objectives a special experiment was performed. In the experiment 500 000 steps with each type of moves were executed. For each type of moves and for each objective percentages of steps that improved, worsened and did not change the value of the objective were calculated. The results of the experiment are summarized in Table 13.

PSA requires that the elementary moves modify all the objectives. So, no type of moves can be used alone in the procedure. The term "composite move" used in Table 13 corresponds to the situation when the type of move is selected randomly with equal probability in each iteration. This approach is used in the PSA procedure.

10.1.3. Example

The approach described above has been implemented in a prototype decision support system. The system is composed of two main modules: the computational module and the user interface module. The computational module is implemented in standard C++. The user interface module is implemented under Ms Windows environment. It allows the user to:

- handle of workers database,
- specify scheduling period's parameters,
- specify wishes of nurses,
- set PSA parameters,
- search for the best compromise in the set of solutions generated by PSA with the interactive trichotomy segmentation.

Table 14. Numbers of nurses required on particular shifts during the scheduling period

Shift	9.6	10.6	11.6	12.6	13.6	14.6	15.6	16.6	17.6	18.6	19.6
Morning	2	4	5	4	4	4	3	2	4	4	5
Afternoon	2	2	3	2	2	2	2	2	2	2	3
Night	2	2	2	2	2	2	2	2	2	2	2
Shift	20.6	21.6	22.6	23.6	24.6	25.6	26.6	27.6	28.6	29.6	30.6
Morning	4	4	3	2	4	4	5	4	4	3	2
Afternoon	2	2	2	2	2	2	3	2	2	2	2
Night	2	2	2	2	2	2	2	2	2	2	2
Shift	1.7	2.7	3.7	4.7	5.7	6.7	7.7	8.7	9.7		
Morning	4	4	5	4	4	3	2	4	4		
afternoon	2	2	3	2	2	2	2	2	2		
Night	2	2	2	2	2	2	2	2	2		

Table 15. Wishes of the nurses

Person	Day	Wish
Szukała	9.6.96	day-off
Kąkol	25.6.96	morning shift
Zimowska	16.6.96	day-off
Kudełka	28.6.96	afternoon shift
Kokot	3.7.96	day-off
Zachmaun	5.7.96	day-off
Stroińska	29.6.96	morning shift
Matysiak	10.6.96	day-off

Table 16. Ranges of objectives found by the PSA procedure

Objective	Best value	Worst value
f_1	0.77	3.08
f_2	0.5	6.21
f_3	2.5	1.48
f_4	9.18%	27.11%
f_5	1	1.45

In order to illustrate the approach an exemplary problem will be solved. Assume that the work of 14 nurses is to be scheduled in the period since 9.6.96 to 9.7.96. Thirteen of the nurses have full-time contracts. The requirements for particular days are given in table 2. Two persons have holidays in this period. Kąkol is absent since 9.6.96 till 15.6.96.

Chelińska is absent since 1.7.96 till 9.7.96. The wishes expressed by some nurses are given in Table 15.

The problem was solved by the PSA method. 16 generating solutions were used. 200 000 steps of the procedure were performed. The method generated 2017 different approximately non-dominated schedules. Table 16 summarizes the ranges of particular objectives in this set of solutions.

Table 17. Exemplary schedule found by the system

Person	9.6	10.6	11.6	12.6	13.6	14.6	15.6	16.6	17.6	18.6	19.6	20.6	21.6	22.6	23.6	24.6
Szukała	f	a	a	m	m	m	f	f	m	m	m	m	m	f	n	n
Mykowska	m	m	m	m	f	n	n	n	n	n	f	f	f	f	f	m
Kąkol	b	b	b	b	b	b	b	a	a	a	a	f	f	f	m	m
Zimowska	f	f	m	m	n	n	n	f	f	f	a	a	a	f	m	m
Piszcz	a	a	a	a	a	a	a	f	m	m	m	f	m	m	n	f
Kudełka	a	m	m	m	f	f	f	m	m	m	f	n	n	n	f	a
Kokot	n	n	n	n	f	f	a	a	m	m	m	m	m	m	f	m
Kokot	m	m	m	m	m	m	f	f	f	m	m	m	m	m	f	f
Toporek	f	m	m	m	m	m	m	f	a	a	a	a	a	f	f	n
Zachmaun	f	f	a	a	a	a	f	f	m	m	m	m	m	f	f	m
Stroińska	f	m	m	m	m	m	f	n	n	n	n	f	f	a	a	a
Chelińska	f	f	m	m	m	m	m	m	f	f	n	n	n	n	f	f
Młynarz	n	n	n	n	n	f	f	f	m	m	m	m	m	f	f	f
Matysiak	f	f	m	m	m	m	m	f	f	f	f	m	m	a	a	m
Person	25.6	26.6	27.6	28.6	29.6	30.6	1.7	2.7	3.7	4.7	5.7	6.7	7.7	8.7	9.7	
Szukała	n	n	f	f	f	f	m	m	m	a	a	f	a	m	m	
Mykowska	m	m	m	f	f	a	a	a	a	f	m	m	m	m	m	
Kąkol	m	m	m	m	f	f	f	m	m	m	n	n	f	m	m	
Zimowska	m	m	m	f	a	m	m	m	n	n	n	f	f	a	a	
Piszcz	f	a	n	n	n	f	m	m	m	f	f	f	f	m	m	
Kudełka	a	a	a	a	f	f	m	m	m	m	m	f	f	n	n	
Kokot	m	m	m	m	f	f	a	a	a	m	m	f	f	f	f	
Kokot	f	f	a	a	a	a	n	n	f	f	m	m	m	m	m	
Toporek	n	n	n	n	f	f	f	f	m	m	m	m	f	m	m	
Zachmaun	m	m	m	m	n	n	n	f	f	m	f	n	n	n	n	
Stroińska	a	f	m	m	m	f	f	m	m	m	m	a	n	f	f	
Chelińska	f	a	m	m	m	n	b	b	b	b	b	b	b	b	b	
Młynarz	m	m	m	m	m	f	f	f	a	a	a	a	a	a	a	
Matysiak	f	f	f	f	f	m	m	n	n	n	f	f	f	m	m	

Table 17 contains one of the schedules found by the system. The values of particular objectives calculated for this schedule are as follows: f_1=1.08, f_2=1.5, f_3=2.21, f_4=18.12%, f_5=1.11. The wishes of the nurses were treated as hard constraints, so, all of them are satisfied. The form in which the schedule is presented is similar to that traditionally used in the hospital.

10.1.4. Practical evaluation

The proposed approach was evaluated in the surgery unit of the hospital Przemienienia Pańskiego in Poznań. All the objectives were calculated for a number of real schedules created manually in previous months by a head nurse. The head nurse has several years of experience in this task. For the same scheduling periods PSA procedure was run.

One of the goals of the experiment was to test whether it is possible to generate solutions that will dominate the solutions created manually by an experienced person, i.e. whether it is possible to find schedules that will be better on some objectives and not worse on the other objectives than the solutions created manually. It turned-out that it requires about 100 000 steps (elementary moves made by all solutions together) of the procedure to find such solutions. This number of moves requires about half an hour on PC with Pentium 120 MHz. The solutions that dominated the manually created ones were slightly better on objective f_2 and significantly better on objectives f_1, f_3, f_4 and f_5. In relatively short time, necessary to perform about 10 000 moves, it was possible to find solutions significantly better on objectives f_1, f_3, f_4 and f_5 but worse on objective f_2 than the solutions created manually. This was caused by the fact that solutions created by the head nurse were always very good from the point of view of the second objective. We observed that the head nurse often inserted a single day-off between two working days when a nurse works on different shifts. This resulted in schedules with a very low number of changes, i.e. situations when in two consecutive days a person works on different shifts. At the same time, however, the schedules included many single days-off and relatively short working periods, even single working days. The schedules generated by the system were evaluated high by the head nurse and personnel of the unit.

10.2. Multiple objective software project scheduling under fuzziness

Software projects are repeatedly late [45], [79]. Pressman [130] lists the following main reasons of the delays:

- unrealistic deadlines,
- poor requirements definitions and changes to the requirements definitions during the project,
- errors in cost and resource estimation,
- overlooking important risks,
- technical difficulties,
- poor communication in the development team,
- too late recognition of delays with respect to the schedule and the lack of repair actions.

Many of the above reasons are related to project scheduling and monitoring. Brooks stated that software projects get delayed "one day at a time" [10]. This means that even large delays usually have no single main reason but are results of many small delays of elementary activities.

Time parameters of software project activities obviously cannot be predicted precisely. As it was mentioned in section 8, the uncertainty of the time parameters is caused by the fact that most software production tasks are human-intensive as well as very high innovation rate in the software industry. Furthermore, unlike in some other engineering disciplines, e.g. in civil engineering, production of software is unique even at the level of elementary activities. Thus, the use of historical data for probabilistic modeling is usually impossible. In practice, it appears that software project managers are often able to estimate approximate duration of particular activities. Their estimations can be naturally modeled in the form of fuzzy numbers [59], [60].

Classic approaches to project scheduling assume that the only objective of the project manager is to minimize the completion time of the project. Although, this objective is certainly very important in practice project managers often take into account other aspects, e.g. project cost, resource requirements, etc.

Together with Hapke, Kominek and Słowiński we have been working on software project scheduling for several years. The work resulted in the development of both new methodology and software tools for this area of application [62]. In this section, we concentrate on the methodology for interactive analysis of multiple objective software project scheduling under fuzziness. This work resulted in a system called FMPS (fuzzy multiple objective project scheduling) that integrates implementations of fuzzy PSA and fuzzy LBS methods (see section 8) and Fuzzy Project tool [62]. In addition, FMPS has been integrated with an UML based CASE tool through the fuzzy SISOP system [54].

10.2.1. Project scheduling under fuzziness

The multiple objective fuzzy project scheduling problem can be characterized by set of resources $\boldsymbol{R}$ to be allocated among precedence constrained activities from set $\boldsymbol{Z}$, and by set $\boldsymbol{C}$ of project performance measures (objectives, criteria) see [149], [178].

Set $\boldsymbol{R}$ is composed of resources belonging to three categories:

- *renewable* resources with a limited usage,
- *nonrenewable* resources with a limited consumption,
- *doubly constrained* resources with a limited consumption and usage.

Activities from set $\boldsymbol{Z}$ have discrete resource requirements. They can be performed in several performing modes. Each performing mode of an activity is defined by different resource requirements and by different duration. Moreover, for each activity, ready time is known. It is assumed that all the time parameters of activities are, in general, uncertain and modeled by fuzzy numbers.

The multiple performing modes of activities imply the multiple objective nature of the problem (time – cost trade-offs). In the project scheduling problem the following performing measures are usually considered: project makespan (completion time), resource utilization smoothness, maximum lateness, mean flow time, net present value, project cost. Let us notice that in our case, because of uncertain time parameters of activities, the values of time and cost objectives from set $\boldsymbol{C}$ are also uncertain, so, their values are fuzzy numbers.

$\tilde{t} := \tilde{t}_0$
repeat
  Compose a set $Q(\tilde{t})$ of those activities which have not been scheduled yet and whose immediate predecessors have been completed by time $\tilde{t}$;
  for each activity Z_i from $Q(\tilde{t})$, in the order of the priority list **do**
  begin
    if Z_i's resource requirements $\leq$ resource availability **then**
    if $\tilde{r}_i <<= \tilde{t}$ **then**
    begin
      Z_i's start time: $\tilde{s}_i := \tilde{t}$
      Z_i's finish time: $\tilde{ft}_i := \tilde{s}_i \oplus \tilde{p}_i$
      remove Z_i from $Q(\tilde{t})$
      allocate required resources to Z_i
      insert $\tilde{ft}_i$ into set B
    end
    else
      insert $\tilde{r}_i$ into set B
  end
  $\tilde{t} := \max(\tilde{t}, \tilde{l})$
  if $\tilde{l}$ is a finish time of any activity **then**
    update resource availability
  remove $\tilde{l}$ from set B
until all activities from the priority list are completed

where:

$\tilde{t}$ - current time moment in which the an allocation of resources to an activity is considered

$\tilde{l}$ - least (in the sense of WCR) value from the set B,

set $B = \{ \tilde{f}_i : \forall Z_i \in S(\tilde{t}) \} \cup \{ \tilde{r}_i : \forall Z_i \in R(\tilde{t}) \}$,

$S(\tilde{t})$ – the set of activities scheduled before $\tilde{t}$, $\tilde{s}_i <<= \tilde{t}$,

$R(\tilde{t})$ – the subset of $Q(\tilde{t})$ of activities not ready in $\tilde{t}$, $\tilde{r}_i \geq \tilde{t}$.

Figure 41. Fuzzy parallel scheduling procedure

In solving NP-hard project scheduling problems a parallel scheduling procedure has been found the most efficient heuristic. Its important feature is that allocation of resources to activities is considered in subsequent events, in which the procedure checks whether the considered resource can be allocated to activities. A generalization of the parallel procedure to fuzzy time parameters has been proposed by Hapke and Słowiński [64]. The generalization consists in answering two questions:

- how to determine the sequence of fuzzy time moments in which new activity (activities) is (are) to be considered,
- how to determine the time moment in which a renewable or doubly constrained resource becomes available to other activities (activity has been completed by the current fuzzy time moment).

The goal was to construct a procedure that would give feasible schedules for *every realization* of fuzzy time parameters. In order to avoid the violation of precedence and resource constraints, the WCR has to be substituted by the Strong Comparison Rule – SCR,

$$\tilde{A} >>= \tilde{B} \Leftrightarrow \max(\tilde{A}, \tilde{B}) = \tilde{A}.$$

It is obvious that SCR includes WCR. The fuzzy parallel scheduling procedure is presented in Figure 41.

10.2.2. Adaptation of fuzzy PSA to multiple objective project scheduling under fuzziness

The adaptation of fuzzy version of PSA to the problem of multiple objective project scheduling under fuzziness requires the definition of neighborhood. The neighbor of the current schedule is constructed according to the algorithm presented in Figure 42. The starting solutions are constructed by assigning to the activities random priorities and then applying the fuzzy parallel scheduling procedure (see Figure 41).

draw an activity Z_x at random from **Z**
copy to the new schedule all the activities Z_i : $\tilde{f}_i <<= \tilde{s}_x$
create a set W of activities Z_j whose start times $\tilde{s}_i >> \tilde{s}_x$ and satisfy the
    precedence constraints in moment $\tilde{s}_x$ whose $\tilde{r}_i <<= \tilde{s}_x$
draw an activity Z_y at random from W, draw its performing mode number
    at random from {1,2,..., z} and schedule it in moment $\tilde{s}_x$
schedule the rest of activities in order of their start times of the current
    schedule taking activity Z_x instead of Z_y according to the fuzzy
    scheduling procedure

where: $\tilde{s}_i$ and $\tilde{f}_i$ are the start and finish times of activity Z_i

Figure 42. The algorithm for drawing a neighborhood solution

10.2.3. Example

The presented approach has been used in a number of small and medium size software projects in Poznan University of Technology and in a number of co-operating companies. In addition evaluation version of the implemented software [62] has been distributed to a number of organization and companies. At present we are continuing the work on integration of the proposed methodology with industry standard products for project scheduling.

In order to illustrate the presented approach we present its application to scheduling of an exemplary software project of building a commercial data base system. The project consists of 51 precedence related activities (see Figure 43). The activities require resources belonging to two categories: renewable and non-renewable (see Table 18). Table 18 includes two groups of resources: programmers and senior programmers. Some activities can be performed both by the former and the latter. Performing an activity by a programmer is cheaper but making the same by a senior programmer takes less time. Thus, the existence of multiple performing modes implies multiple objective nature of the problem.

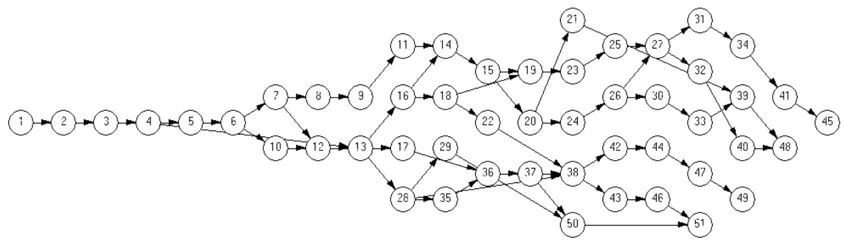

Figure 43. Precedence related software project activities

Project manager (PM) is interested in minimization of three objectives:

- the project completion time (*Cmax*) expressed in days,
- the total project cost (*Cost*) expressed in $,
- the resource smoothness rate (*Rs*) calculated as the average deviation from the average resource usage expressed in resource units.

The first two objectives take fuzzy values, while the last one takes crisp values.

Table 18. The set of available resources

Name	Category	Availability	Cost ($/h)
Business Analyst	renewable	12	30
Technical Editor	renewable	8	25
Product Manager	renewable	8	35
Production Specialist	renewable	6	15
Programmer	renewable	25	28
Senior Programmer	renewable	15	32
Software Engineer	renewable	10	31
Tester	renewable	28	23
Trainer	renewable	8	30
Translation Specialist	renewable	16	25
Writer	renewable	16	24
Money	non-renewable	470.000$	—

In order to solve the exemplary problem the presented two-phase approach has been used (see section 7.1). In the first phase the fuzzy version of PSA has been used, while in the second phase the fuzzy version of LBS has been used.

The following parameters have been set for the fuzzy PSA procedure:

- conditions of changing the temperature (number of iterations at one temperature level) 2000
- the size of the sample of the generating solutions 8

- starting temperature 0.1
- stopping condition temperature ≤ 0.001
- the temperature changing factor 0.9

The fuzzy PSA procedure has generated a set of 592 WCR-Pareto optimal solutions.

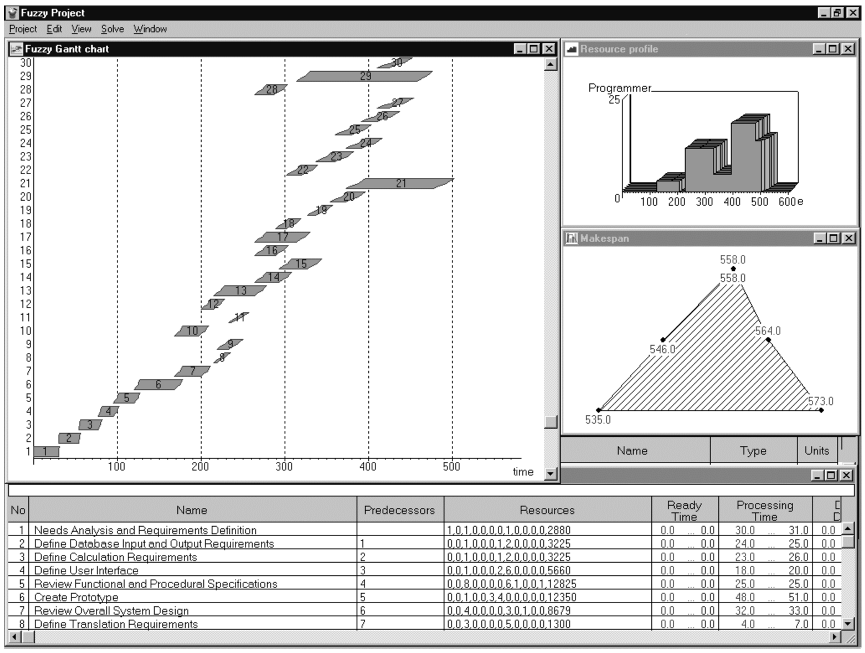

Figure 44. The best compromise fuzzy schedule

We illustrate the work with the fuzzy LBS procedure by presenting an exemplary session with the method. The procedure starts by calculating the ideal and the nadir points giving the PM information about possible range of each objective. The two points are presented below (the value of each fuzzy objective is represented in two ways – by fuzzy number and its defuzzified counterpart according to the function $\Im$):

Point	*Cmax*	*Rs*	*Cost*
Ideal	**546**	**3.5**	**436866**
	525,536,548,548,555,565		428391,431835,436656,436656,440788,448365
Nadir	**660**	**7.2**	**453670**
	645,652,659,659,667,682		442847,448248,454420,454420,458277,464449

Then the PM defines the reference point:
Cmax = 590
Rs = 5.5
Cost = 440000

The reference point is then projected onto the set of WCR-Pareto optimal solutions. This results in the following starting middle point:

Cmax	*Rs*	*Cost*
601	**5.7**	**439669**
589,595,603,603,606,611		430831,435220,441071,441071,443266,446923

Then the PM expresses his/her preferential information by specifying the indifference and preference thresholds on each objective:

Objective	q_j	p_j
Cmax	5	14
Rs	1	2
Cost	300	1000

This means that solutions differing by less than 5 days are treated by the PM as indifferent. A schedule shorter than 14 days is strongly preferred by the PM. In the outranking neighborhood, three characteristic points have been found (each point locally minimizes one of the three objectives):

Solution	*Cmax*	*Rs*	*Cost*
1.	**601**	**5.7**	**439669**
	589,595,603,603,606,611		430831,435220,441071,441071,443266,446923
2.	**596**	**7.0**	**439454**
	577,587,598,598,602,613		425743,433121,441238,441238,444189,452305
3.	**606**	**6.9**	**438945**
	588,597,608,608,613,624		425868,432386,440353,440353,443974,451941

In comparison with the reference point, the above solutions have low values of the project cost. However the project completion time is not satisfactory to the PM. Thus, he/she decides to change the reference point by decreasing *Cmax* at the expense of *Rs* and *Cost* co-ordinates. A new reference point is as follows:

Cmax = 560
Rs = 6
Cost = 450000

As expected, the new middle point is better on *Cmax* objective:

Cmax	*Rs*	*Cost*
562	**6.2**	**449392**
541,552,564,564,571,580		432621,441417,451013,451013,456611,463808

Then, the PM changes thresholds on *Cost* objective. He/she finds indifferent two schedules differing by less than 400$ and strongly prefers a schedule being cheaper by 1500$. Moreover, he/she changes the preference threshold on *Cmax* to 10 days.

Objective	q_j	p_j
Cmax	5	10
Rs	1	2
Cost	400	1500

This results in the following three characteristic points:

Solution	*Cmax*	*Rs*	*Cost*
1.	**546**	**7.2**	**445969**
	525,536,548,548,555,565		428751,437735,447535,447535,453251,461418
2.	**567**	**6.2**	**449136**
	547,557,569,569,577,585		433039,440956,450456,450456,456789,463123
3.	**570**	**7.1**	**441755**
	551,562,573,573,578,588		426489,435003,443518,443518,447388,455128

The PM chooses point #1 as the new middle point. The corresponding characteristic points are as follows:

Solution	*Cmax*	*Rs*	*Cost*
1.	**546**	**7.2**	**445969**
	525,536,548,548,555,565		428751,437735,447535,447535,453251,461418
2.	**556**	**7.1**	**443770**
	535,546,558,558,564,573		427307,436093,445678,445678,450470,457658
3.	**556**	**7.1**	**443770**
	535,546,558,558,564,573		427307,436093,445678,445678,450470,457658

Then the PM wants to see some schedules being between points #1 and #2 in the current outranking neighborhood:

Solution	*Cmax*	*Rs*	*Cost*
1.	**546**	**7.2**	**445969**
	525,536,548,548,555,565		428751,437735,447535,447535,453251,461418
2.	**551**	**7.2**	**445715**
	530,541,553,553,560,571		428576,437471,447175,447175,452835,461730
3.	**552**	**7.2**	**445691**
	531,542,554,554,562,572		428391,437265,446946,446946,453401,461468
4.	**556**	**7.1**	**443770**
	535,546,558,558,564,573		427307,436093,445678,445678,450470,457658

Finally, the PM chooses point #4 as the best compromise schedule. Taking into account crisp equivalents of each objective value the PM can see that the schedule is about 10 days longer than the shortest possible and about 104 days shorter than the longest possible. It has almost the worst (the highest) resource smoothness rate. The corresponding project cost is about 6904$ higher than the cheapest possible and about 9900$ lower than the most expensive.

Cmax	*Rs*	*Cost*
556	**7.1**	**443770**
535,546,558,558,564,573		427307,436093,445678,445678,450470,457658

The schedule can be presented in the fuzzy Gantt chart given in Figure 44.

10.3. Logistic distribution system design – application and a comparative experiment

10.3.1. Description of the problem

An international company operating in Poland is going to completely rebuild its distribution system [88]. The company entered Polish market at the beginning of 90's and acquired two production plants located in two large Polish cities. The company focuses on the production and sales of cosmetics, detergents and washing articles. The annual turnover of the company is roughly $100 million.

The distribution system is going to be composed of up to 39 distribution centers (warehouses) placed is some predefined potential locations. Each distribution center will deliver company's products to clients located in one or more of 49 regions of Poland. The regions correspond to the old administrative districts. The goal of the study is to select appropriate locations for distribution centers and assignment of regions to the distribution centers. The following three objectives are taken into account:

- minimization of the total annual distribution cost, including the transportation cost, the warehousing cost and the cost of locked-up capital,
- minimization of the worst case riding time,
- minimization of the number of distribution centers (minimization of the complexity of the system).

The problem shares some characteristic features with many other real-life problems:

- it involves objectives having different mathematical formulation (many test problems involve objectives that differ by parameter values but have the same mathematical form),
- although it has some similarities with the classic combinatorial optimization problems, e.g. location problems, it has a number of specific element, making the use of standard operational research methods unfeasible.

Thus, we decided to perform also a comparative experiment on the real-life problem.

Below we present mathematical programming formulation of the problem:

Data

The following data are used in the model:

I – the number of potential locations of distribution centers,

J – the number of regions that have to be assigned to the distribution centers,

L – the number of vehicle types used for transportation from production plants to distribution centers,

DA_j – annual demand of region j for products of production plant A in [pallets], $j = 1,...,J$,

DB_j – annual demand of region j for products of production plant B in [pallets], $j = 1,...,J$,

TC_{ij} – average transportation cost from distribution center at location i to region j in [PLN/pallet], $i = 1,...,I$, $j = 1,...,J$,

TCA_{jl} – average transportation cost from production plant A to distribution center at location i by vehicle of l-th type in [PLN/pallet], $i = 1,...,I$, $l=1,...,L$,

TCB_{jl} – average transportation cost from production plant B to distribution center at location i by vehicle of l-th type in [PLN/pallet], $i = 1,...,I$, $l=1,...,L$,

TT_{ij} – average travel time from distribution center at location i to the first customer in region j in [min], $i = 1,...,I$, $j = 1,...,J$,

PHC_i – cost of pallet handling in distribution center at location i, $i = 1,...,I$,

CRT – current average pallet rotation time in [days], i.e. the average number of days that a pallet spends in the current distribution system,

CCA – average daily cost of locked-up capital per pallet produced in production plant A in [PLN/day],

CCB – average daily cost of locked-up capital per pallet produced in production plant B in [PLN/day],

$MSSA$ – annual cost of minimum safety stock of pallets produced in production plant A to be maintained in each distribution center

$MSSB$ – annual cost of minimum safety stock of pallets produced in production plant B to be maintained in each distribution center

ML_l – capacity of vehicles of l-th type used for transportation from production plants to distribution centers, $l=1,...,L$,

DY – average number of working days in a year.

Decision variables

Two groups of decision variables are considered:
$y_i \in \{0, 1\}$, $i = 1,...,I$, equals to one if distribution center at location i is included in the plan, and 0 otherwise.
$x_{ij} \in \{0, 1\}$, $i = 1,...,I$, $j = 1,...,J$, equals to one if region j is assigned to distribution center at location i, and 0 otherwise.

Constraints

The first group of constraints assures that regions are assigned only to distribution centers included in the plan:

$$x_{ij} \le y_i , \qquad i = 1,...,I, j = 1,...,J.$$

The second group of constraints assures that each region is assigned to exactly one distribution center:

$$\sum_{i=1}^{I} x_{ij} = 1 , \qquad j = 1,...,J.$$

Objective functions

Three objectives are considered in the model:

- *TDC* – total annual cost of distribution,
- *MRT* – maximum riding time,
- *NDC* – number of distribution centers.

All the objectives are minimized.

The first objective is defined in the following way:

$$TDC = \sum_{i=1}^{I} \min_{l}\{ATCA_{il} + ACCA_{il}\} + \min_{l}\{ATCB_{il} + ACCB_{il}\} + APHC_i$$

where $ATCA_{il}$ ($ATCB_{il}$) denotes annual transportation cost of products from production plant A (B) to distribution center at location i with the use of vehicles of l-th type, $ACCA_{il}$ ($ACCB_{il}$) denotes annual cost of capital locked-up in pallets of products from production plant A (B) at location i with the use of vehicles of l-th type, and $APHC_i$ denotes total annual cost of pallets handling in the distribution centers.

Annual transportation cost takes into account transportation from production plants to distribution centers and transportation from distribution centers to the customers in the regions. It is defined in the following way:

$$ATCA_{il} = TCA_i \sum_{j=1}^{J} x_{ij} DA_j + \sum_{j=1}^{J} x_{ij} TC_{ij} DA_j \text{ , } ATCB_{il} = TCB_i \sum_{j=1}^{J} x_{ij} DB_j + \sum_{j=1}^{J} x_{ij} TC_{ij} DB_j \text{ .}$$

Annual cost of locked-up capital is defined in the following way:

$$ACCA_{il} = \left(\max\left\{ y_i \left(MSSA + \sum_{j=1}^{J} x_{ij} DA_j CCA\, DHA_{il} \right), \sum_{j=1}^{J} x_{ij} DA_j CCA \left(CRT + DHA_{il} \right) \right\} \right),$$

$$ACCB_{il} = \left(\max\left\{ y_i \left(MSSB + \sum_{j=1}^{J} x_{ij} DB_j CCB\, DHB_{il} \right), \sum_{j=1}^{J} x_{ij} DB_j CCB \left(CRT + DHB_{il} \right) \right\} \right) \quad .$$

where DHA_{il} and DHB_{il} are average headway of deliveries for production plants A and B, respectively, with the use of vehicles of l-th type. They are defined in the following way:

$$DHA_i = ML_l / \left(\sum_{j=1}^{J} x_{ij} DA_j / DY \right), \; DHB_i = ML_l / \left(\sum_{j=1}^{J} x_{ij} DB_j / DY \right).$$

Annual cost of pallets handling in the distribution centers is defined in the following way:

$$APHC_i = y_i PHC_i \left(\sum_{j=1}^{J} x_{ij} DA_j + \sum_{j=1}^{J} x_{ij} DA_j \right).$$

Maximum riding time to the first customer on a route is defined in the following way:

$$MRT = \max\{ x_{ij} TT_{ij} \}.$$

Number of distribution centers is defined in the following way:

$$NDC = \sum_{i=1}^{I} y_i \text{ .}$$

10.3.2. Adaptation of MOGLS to the distribution system design problem

The problem seems to be well suited for the use of a genetic local search algorithm. It is intuitively clear that good solutions should have many common features. For example, it is natural to expect regions to be assigned to near distribution centers and to expect distribution centers to be located in regions with large demands. Furthermore the local move described below modifies only a small part of solution what allows to improve the effectiveness of local search.

The solutions are encoded by I lists of regions. The i-th list contains regions assigned to the distribution center at location i. If the list is empty there is no distribution center at this location (no minimum safety stock is maintained).

Initial solutions are constructed by assigning each region to a distribution center at a randomly selected location.

for each potential location of a distribution center
    if the distribution center at this location has non-empty list of assigned regions in both parents
        Mark the location as *preliminary selected*
    else if the distribution center at this location has non-empty list of assigned regions in one of the parents
        Mark the location as *preliminary selected* with probability 0.5
for each region
    if the region is assigned in both parents to a distribution center at the same location
        Assign the region to the distribution center at the same location
    else /* *The region is assigned to distribution centers at different locations* */
        if both the locations are marked as *preliminary selected*
            Select at random one of the locations and assign the region to the distribution center at this location
        else if one of the locations is marked as *preliminary selected*
            Assign the region to the distribution center at this location
        else /* *None of the locations is marked as preliminary selected* */
            Select a location at random. If it is not marked as *preliminary selected* grant it this mark. Assign the region to the distribution center at this location

Figure 45. The algorithm of the recombination operator

The idea of the recombination operator is to preserve important features common to both parents, i.e.

- common selections of distribution centers locations,
- common assignments of regions to distribution centers at the same locations.

The details of the recombination operator algorithm are presented in Figure 45.

Local search is used to improve the value of the weighted linear scalarizing function. We have tested also a version of local search that improves weighted Tchebycheff scalarizing function but it resulted in worse performance of all the algorithms.

Local search uses neighborhood move that shifts a region from a distribution center at a location to a distribution center at another location.

Greedy version of local search is used. It tests the neighborhood moves in random order and performs the first improving move. The local search algorithm is stopped when no improving move is found after testing all the possible neighborhood moves, which means that a local optimum is achieved.

The mutation operator shifts a randomly selected region to a distribution center at a randomly selected potential location.

10.3.3. Computational experiment

Four multiple objective metaheuristics have been tested on this problem. We used Pareto ranking-based multiple objective genetic algorithm proposed by Fonseca and Flemming [35] (Pareto GA), multiple objective multiple start local search (MOMSLS) with random weight vectors, multiple objective genetic local search (MOGLS) and an extension of Pareto GA involving local search (Pareto GLS). Two different quality measures were used in this experiment – coverage of two approximations of the nondominated set Q_4 and the estimation of expected value of weighted Tchebycheff scalarizing function on an approximation – $\hat{R}$. The parameter k of measure $\hat{R}$ was set to 50. In addition, running time of the methods was taken into account. All the methods used the same solution encoding, the same recombination and local search operators and the same way of constructing initial solutions. All the methods were implemented with MOEALib++ library (see Appendix) and shared most of the C++ code.

Pareto GLS hybridizes Pareto GA with local search. The starting population is composed of local optima obtained in the same way as in MOGLS and MOMSLS. In addition, after each recombination local search is applied to each offspring. The scalarizing function to be optimized is drawn at random in the same way as in MOGLS (see Figure 18).

Table 19. Average running times of the methods

MOGLS		Pareto GLS		MOMSLS		Pareto GA	
Number of recombinations	Average running time [s]	Number of recombinations	Running time [s]	Number of local searches	Running time [s]	Number of generations	Running time [s]
60	66.1	60	67	120	116.1	40	15.7
120	72	120	76.3	180	174.6	80	34.4
180	76.9	180	84.6	240	233.2	120	53.6
240	81	240	92.6	300	291.6	160	73.2
300	85	300	100.8	360	350.8	200	92.7
360	89.2	360	108.9	420	409.7	240	111.4
420	92.7	420	117.2	480	468.3	280	130.8
480	96.3	480	125.7	540	526.9	320	150
540	100	540	134.2	600	585.8	360	168.5
600	103.9	600	142	660	644.4	400	187.1

Each method has been run 10 times. MOGLS has used temporary population of size 30. The number of initial solutions has been set equal to 60. This value has been set with the approach proposed in section 6.4. This number of initial solutions assures that on average 30 best solutions on a randomly chosen weighted linear scalarizing function in the initial set are of the same quality as 30 local optima of the same function. The sets of potentially Pareto-optimal solutions generated in each run have been saved after 60, 120,..., 600 recombinations.

Pareto GLS has used population of size 60. In other words, Pareto GLS has been starting by generating the same number of local optima as MOGLS. The number of generations has been set to 11. This has assured the same number of recombinations as in the case of MOGLS. The sets of potentially Pareto-optimal solutions have been saved after the same numbers of recombinations as in the case of MOGLS. The mutation probability has been set equal to 0.1.

Table 20. Results of comparison using expected value of weighted Tchebycheff scalarizing

MOGLS			Pareto GLS		
Number of recombinations	Value of quality measure $\hat{R}$	Standard deviation of quality measure $\hat{R}$	Number of recombinations	Value of quality measure $\hat{R}$	Standard deviation of quality measure $\hat{R}$
60	0.099	0.005	60	0.111	0.006
120	0.089	0.004	120	0.101	0.004
180	0.082	0.006	180	0.093	0.007
240	0.078	0.005	240	0.088	0.006
300	0.077	0.005	300	0.084	0.006
360	0.075	0.004	360	0.082	0.006
420	0.072	0.004	420	0.080	0.004
480	0.070	0.003	480	0.078	0.004
540	0.068	0.002	540	0.078	0.004
600	0.067	0.003	600	0.077	0.005
MOMSLS			Pareto GA		
Number of local searches	Value of quality measure $\hat{R}$	Standard deviation of quality measure $\hat{R}$	Number of generations	Value of quality measure $\hat{R}$	Standard deviation of quality measure $\hat{R}$
120	0.236	0.009	40	0.402	0.023
180	0.233	0.008	80	0.389	0.028
240	0.232	0.007	120	0.387	0.027
300	0.229	0.008	160	0.383	0.025
360	0.227	0.006	200	0.378	0.024
420	0.227	0.007	240	0.371	0.022
480	0.226	0.006	280	0.367	0.023
540	0.226	0.006	320	0.363	0.024
600	0.224	0.006	360	0.357	0.020
660	0.224	0.006	400	0.355	0.021

MOMSLS has been performing 660 local searches, i.e. the same number of local searches as MOGLS and Pareto GLS. The sets of potentially Pareto-optimal solutions have been saved after 120, 180,..., 660 local searches, i.e. after the same number of local searches as MOGLS and Pareto GLS.

Pareto GA has used population of size 100 and has been performing 400 generations. Since this method does not use local search it performs more recombinations per seconds than Pareto GLS. The above parameters assured running times comparable to MOGLS and Pareto GLS. The mutation probability has been set equal to 0.1.

Table 19 presents average running times of the methods. The dispersion of the running times after a given number of iterations is not given, it has been, however, very low. Note that MOMSLS has required much more time than MOGLS and Pareto GLS to perform the same number of local searches. This is caused by the fact that solutions obtained by recombination are much closer to local optima than random solutions. Thus, local search needs less iterations to achieve a local optimum.

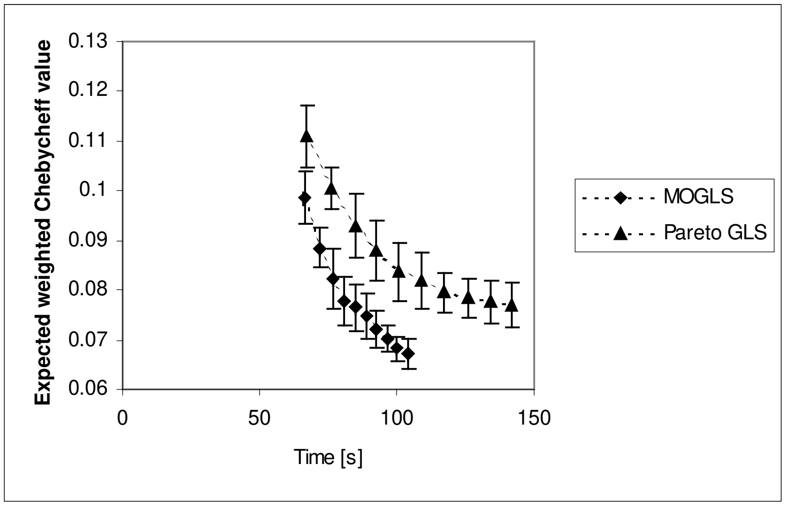

Figure 46. Results of comparison with expected value of weighted Tchebycheff scalarizing function. The chart shows ± standard deviation range of the quality measure

Table 20 presents results of evaluation with the expected value of weighted Tchebycheff scalarizing quality measure $\hat{R}$. The results clearly indicate that the two methods hybridizing recombination and local search operators by far outperform methods that use one of the operators alone. In particular the comparison of Pareto GLS, Pareto GA and MOMSLS is a clear example of the synergy of the two operators. Pareto GLS is a simple combination of the two latter methods and does not include any step that cannot be found in Pareto GA or MOMSLS.

Furthermore, MOGLS outperforms Pareto GLS. It generates better approximations of the nondominated set in shorter times. In our opinion, this is caused by the fact that MOGLS selects solutions for recombination on the basis of the same scalarizing function that is optimized by local search. Furthermore, the solutions are selected from a larger set of known solutions. On the other hand, MOGLS has higher memory requirements than Pareto GLS. Figure 46 illustrates the performance of MOGLS and Pareto GLS.

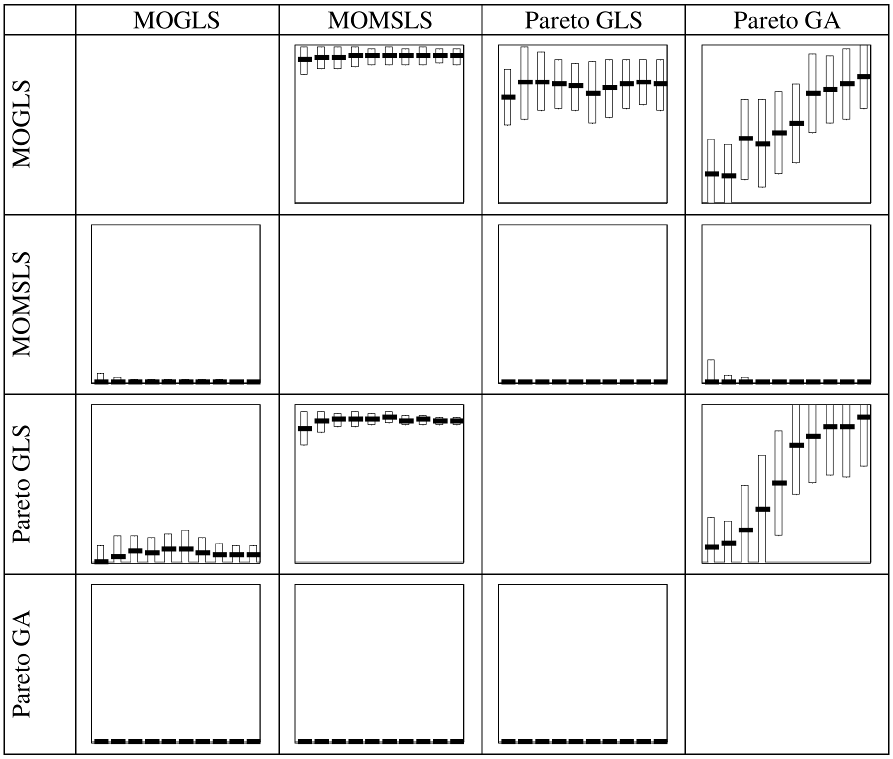

Figure 47. Results of the comparison using the coverage measure Q_4. Each chart contains 10 box plots representing the distribution of Q_4 values for a certain ordered pair of algorithms. The 10 box plots correspond to the ten sets of potentially Pareto-optimal solutions saved by each method in each run (the first set on the left). The scale is 0 at the bottom and 1 at the top for each chart. Chart in row of algorithm *A1* and column of algorithm *A2* presents values of coverage of approximations generated *A2* by approximations generated by *A1*. The charts show ± standard deviation range of the quality measure

Figure 47 presents results of the comparison using the coverage measure. The results indicate the same ranking of the four methods as the evaluation with the expected value of weighted Tchebycheff scalarizing quality measure. Note, that no methods is able to produce approximations of nondominated set covering 100% of approximations generated by other methods. This indicates that no method is able to find all Pareto-optimal solutions.

Summarizing, the results of the computational experiment indicate that:

- hybridization of recombination and local search operators boosts the performance of multiple objective evolutionary algorithms on the distribution system design problem,
- the choice of solutions for recombination has crucial influence on the performance of genetic local search algorithm.

The number of potentially Pareto optimal solutions extracted from the results of all the methods was relatively low. 55 potentially Pareto optimal solutions have been generated. In this case, the DMs (management of the production company) did not need further support in selection of the best compromise solution. The company's magement decided to implement one of the potentially Pareto optimal solutions. An additional analysis has been performed in order to find an optimal plan of rebuilding the distribution system. In other words, the goal of the analysis was to find the best path from the current state to the desired distribution system allowing step by step introduction of new distribution centers.

11. Summary and conclusions

In this work we have presented a review of recently emerged methodology of solving multiple objective combinatorial optimization problems with the use of metaheuristic algorithms. In particular, two new multiple objective metaheuristics proposed by the author have been described in detail. The effectiveness of the methods has been tested in a number of computational experiments. The experiment on 0/1 multiple knapsack problem allowed direct comparison with the results of previous comparative studies of Zitzler and Thiele [188] and Knowles and Corne [101], [102]. All the experiments used the same set of instances and compatible adaptation of the methods to the problem. Taking into account all the experiments, the two methods presented in this work were the best performers among twelve representative multiple objective metaheuristics. The results of this and other experiments indicate that the use of the dominance relation only for guiding the heuristic search is sufficient to assure convergence toward the nondominated set but does not guarantee dispersion over all regions of the set. On the contrary, weighted scalarizing functions allow to achieve both convergence and dispersion over the nondominated set.

In order to evaluate the quality of results of multiple objective metaheuristics we used both measures used in previous experiments as well as new measures based on our work on evaluation of approximations of the nondominated set. A unique advantage of the proposed quality measure is that it allows calculation of a lower bound that may be used to estimate the gap between the quality of an approximation and the quality of the actual nondominated set even if the latter is unknown. Furthermore, the quality measure gives evaluations concordant with intuitive assessments. For example, it differentiates the quality of approximations that cover only small regions of the nondominated set.

In order to support the DM in selection of the best compromise solution we have proposed a two-phase approach to interactive analysis of multiple objective combinatorial optimization problems. In this approach, the approximately Pareto-optimal solutions are generated off-line in the first phase. Two methods proposed by the author for interactive analysis of discrete choice problems applicable in the second phase have also been described. In addition, a new technique for evaluation of computational effectiveness of the off-line generation of approximately Pareto-optimal solutions has been proposed. The technique compares the computational requirements of the off-line generation with the computational requirements of the on-line generation necessary to achieve the same quality of approximation. Computational experiments described in this work indicate that despite other advantages of the off-line generation it may also be competitive from the computational effectiveness point of view.

In order to handle uncertainty of parameters of multiple objective combinatorial optimization problems Pareto simulated annealing and Light Beam Search-discrete have been adapted to handle fuzzy scores. The idea of the adaptation is quite general and may be used in the case of many other multiple objective metaheuristics and interactive procedures.

The proposed methodology has been applied to a number of real-life multiple objective combinatorial optimization problems.

An interesting direction for further research is to more precisely define the classes of problems appropriate for particular methods. The "No free lunch" theorem for optimization [177] clearly indicates that no method may outperform all the other methods on all problems. Obviously, the issue of defining the areas of application for particular methods is not specific to multiple objective metaheuristics. This question has not yet been answered in the case of single objective methods. We expect that hybrid genetic algorithms will perform well on globally convex problems, i.e. problems whose good solutions close in the objective space are also similar in the decision space.

Note that global convexity depends on the definition of similarity in the decision space. Furthermore, the recombination operator should preserve the important features of the parents. In [91] we have proposed an approach that allows to develop effective recombination operators on the basis of experimental studies of correlation between the appearance of particular solutions features and the solutions quality. The approach has been applied in single objective case. As it was mentioned in the introduction, real-life problems often involve objectives with different mathematical definitions. This is true in the case of all the real-life applications described in this work. Thus, it may so happen that the quality on particular objectives is correlated with the appearance of different solutions features. In result, different recombination operators would be appropriate for particular objectives. Similar situation may concern neighborhood operators and other aspects of adapting the metaheuristics to a given problem. For example, different objectives may require different neighborhood operators and/or different methods for constructing the initial solutions. In fact, this situation occurred in the case of the nurse scheduling problem where particular types of moves were able to modify only some of the five objectives. In general, it may be typical that in practice the best operators for particular objectives will be significantly different. The question arises in what way multiple objective metaheuristics may take advantage of the various operators. This issue seems to be almost neglected in the literature. According to our knowledge, the only approach that takes into account different nature of objectives has been proposed by Iredi et al. [74]. They applied an ant colonies algorithm to a bi-objective scheduling problem. They used two different pheromone matrices for the two objectives. The influence of the two matrices depends on the objective weights. In the future, we are going to take into account different nature of objectives in the proposed methods.

In the case of some problems we have observed that the use of weighted linear functions in local optimization improves the performance of multiple objective genetic local search in comparison with the use of weighted Tchebycheff functions. On the other hand, because of the properties of Tchebycheff functions their use is a more general approach. An interesting further research direction would be to study the reasons for difficulties in local optimization of Tchebycheff functions and to propose approaches that would facilitate it. An interesting approach is to optimize composite functions, which allows to preserve some advantages of the Tchebycheff functions [51]. Furthermore, Hansen [51] proposed to use a number of candidate local moves being the best on the linear function. The best move on the Tchebycheff function is then selected from the set of candidate moves. This approach is effective if the evaluation of changes of linear functions is faster than the evaluation of changes of Tchebycheff functions. Recently Levin [112] has proposed a lexicographical local search approach to single objective problems with min-max or max-min objectives that could be applied to optimization of Tchebycheff functions.

In the current version of the multiple objective genetic local search algorithm we assume that local search uses a constant weight vector in each run. An interesting approach would be to allow some small random perturbations of the weight vectors during local search. The comparison of Pareto simulated annealing and Ulungu's et al. multiple objective simulated annealing suggests that small perturbations of the weight vectors may improve the performance of local search.

An interesting direction of research is development of multiple objective versions of new single objective metaheuristics, e.g. social learning algorithm [96]. The algorithm uses analogy to social education systems. In this case, various search directions in the objective space may be interpreted as various specializations of the schools/teachers.

In the future we are also going to continue comparative experiments with multiple objective metaheuristics and to apply these methods to other practical problems.

Streszczenie w języku polskim

Wielokryterialne algorytmy metaheurystyczne do optymalizacji kombinatorycznej

Problemy wielokryterialnej optymalizacji kombinatorycznej (WOK) pojawiają się wielu obszarach, np. w szeregowaniu produkcji, harmonogramowaniu przedsięwzięć, harmonogramowaniu czasu pracy, układaniu planów zajęć, projektowaniu instalacji produkcyjnych, ustalaniu tras pojazdów, telekomunikacji, planowaniu inwestycji i wielu innych. Liczba udanych rzeczywistych zastosowań WOK jest jednak stosunkowo mała w porównaniu z innymi obszarami optymalizacji wielokryterialnej i optymalizacji kombinatorycznej. Najwyraźniej WOK stanowi jeden z najtrudniejszych obszarów optymalizacji.

Z uwagi na sukces metaheurystyk w optymalizacji jednokryterialnej wielu badaczy zaproponowało zastosowanie metaheurystyk w przypadku wielokryterialnym. Od czasu opracowania *Vector Evaluated Genetic Algorithm* w 1985 roku w literaturze opisano wiele nowych wielokryterialnych metaheurystyk. Celem metod tego typu jest wygenerowanie zbioru rozwiązań optymalnych w sensie Pareto będącego przybliżeniem całości lub części zbioru niezdominowanego. Obecnie wielokryterialne metaheurystyki należą do jednego z najbardziej aktywnych obszarów badań w ramach optymalizacji wielokryterialnej, a wykorzystanie wielokryterialnych metaheurystyk stanowi jedno z najbardziej obiecujących podejść do WOK.

Celem niniejszej pracy jest rozwinięcie metodyki wielokryterialnych metaheurystyk do problemów WOK. W pracy przedstawiono aktualny stan badań w zakresie metodyki wielokryterialnych metaheurystyk ze szczególnym uwzględnieniem charakterystyki głównych algorytmów oraz zagadnień oceny tego typu metod. Miary jakości zaproponowane przez innych autorów mają z reguły charakter kardynalny lub geometryczny. W tej pracy zaprezentowano nowe podejście do oceny przybliżeń zbioru niezdominowanego, oparte na założeniach co do możliwych preferencji decydenta. Następnie opisano dwie nowe wielokryterialne metody metaheurystyczne zaproponowane przez autora, oparte na znanych metodach jednokryterialnych – symulowanym wyżarzaniu i algorytmie genetycznym z przeszukiwaniem lokalnym. Przedstawiono także dwufazowe podejście do dialogowej analizy problemów WOK. W pierwszej fazie z wykorzystaniem wielokryterialnej metaheurystyki jest generowany zbiór rozwiązań w przybliżeniu optymalnych w sensie Pareto. W fazie drugiej powyższy zbiór jest analizowany z wykorzystaniem metody dialogowej do problemów wielokryterialnego wyboru ze skończonym zbiorem wariantów. W pracy opisano także dwie nowe metody dialogowe do problemów tego typu, zaproponowane przez autora. Zaprezentowane metody zostały rozszerzone na problemy WOK w warunkach niepewności. Metody te przetestowano w kilku eksperymentach obliczeniowych. Wyniki tych eksperymentów wskazują, że metody te są konkurencyjne w stosunku do reprezentatywnych metod opisanych w literaturze. W pracy przytoczono także przykładowe zastosowania zaproponowanych metod w zagadnieniach harmonogramowania czasu pracy, harmonogramowania przedsięwzięć programistycznych oraz projektowania struktury systemu dystrybucji materiałowej. W podsumowaniu wskazano kierunki dalszych prac badawczych.

References

[1] Ackley D. H. (1987), *A connectionist machine for genetic hillclimbing*. Kluwer Academic Press, Boston.

[2] Assad A.A. (1988), Modeling and implementation issues in vehicle routing, in: B.L. Golden and A.A. Assad (eds.), *Vehicle routing: Methods and Studies*, North-Holland, Amsterdam, 7-46.

[3] Balicki J., Kitowski Z. (to appear), Multicriteria evolutionary algorithm with tabu search for task assignment, in: *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, Springer, Berlin.

[4] Ben Abdelaziz F., Krichen S. (1997), *A tabu search heuristic for multiple objective knapsack problems*, Ructor Research Report, **RR 28-97**.

[5] Bogardi J.J., Gupta A.D. and Jiang H.Z. (1991), Search Beam Method – a promising way to define nondominated solutions in multi-objective groundwater development, *Water Resources Development*, **7**, 4, 247-258.

[6] Boese K., Kahng A., Muddu S. (1994), A new adaptive multistart technique for combinatorial global optimisation, *Operations Research Letters*, **16**, 101-113.

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

[8] Borges P.C. (2000), CHESS – changing horizon efficient set search: a simple principle for multiobjective optimisation, *Journal of Heuristics*, **6**, 405-418.

[9] Brockhoff K. (1985), Experimental test of MODM algorithms in a modular approach, *European Journal of Operational Research*, **22**, 159-166.

[10] Brooks F. (1995), *The Mythical Man Month*, Addison-Wesley, New York.

[11] Brusco M.J., Jacobs L.W. (1993), A simulated annealing approach to flexible labor scheduling problems, *Journal of the Operational Research Society*, **44**, 1191-1200.

[12] Buchanan J.T. (1994), An experimental evaluation of interactive MCDM methods and the decision making process, *Journal of the Operational Research Society*, **45**, 1050-1059.

[13] Buchanan J.T. (1997), A naive approach for solving MCDM problems: The GUESS method, *Journal of the Operational Research Society*, **48**, 202-206.

[14] Buchanan J.T., Corner J.L. (1997), The effects of anchoring in interactive MCDM solution methods, *Computers and Operations Research*, **24**(10), 907-918.

[15] Buchanan J.T., Daellenbach H.G. (1987), A comparative evaluation of interactive solution methods for multiple objective decision models, *European Journal of Operational Research*, **29**, 353-359.

[16] Chanas S. (1987), Fuzzy optimization in networks, in: J. Kacprzyk, S.A. Orlovsky (eds.), *Optimization models using fuzzy sets and possibility theory*, Reidel Publishing Company, Dordrecht, 303-327.

[17] Choo E.U., Atkins D.R. (1980), An interactive algorithm for multicriteria programming. *Computers and Operations Research*, **7**, 81-87.

[18] Coello C.A. (1999), A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques, *Knowledge and Information Systems. An International Journal*, **1** (3), 269-308.

[19] Cvetkovic D., Parmee I., Webb E. (1998), Multi-objective Optimisation and Preliminary Airframe Design, in: I. Parmee (ed.), *The Integration of Evolutionary and Adaptive Computing Technologies with Product/System Design and Realisation*, Plymouth, United Kingdom, Plymouth Engineering Design Centre, Springer, 255-267.

[20] Czyżak P., Hapke M., Jaszkiewicz A. (1994), *Application of the Pareto-Simulated Annealing to the multiple criteria shortest path problem*, Technical Report, Institute of Computing Science, Poznan University of Technology, **94/03**.

[21] Czyżak P., Jaszkiewicz A. (1995), A metaheuristic procedure for multiple objective combinatorial optimization problems, (in Polish), in: R. Kulikowski, L. Bogdan (eds.), *Decision support. Expert systems*, Instytut Badań Systemowych PAN, Warszawa, 85-94,

[22] Czyżak P., Jaszkiewicz A. (1995), *Pareto Simulated Annealing. Computational experiment*, (in Polish), Technical Report, Institute of Computing Science, Poznan University of Technology, **003/95**.

[23] Czyżak P., Jaszkiewicz A. (1996), A multiobjective metaheuristic approach to the localization of a chain of petrol stations by the capital budgeting model, *Control and Cybernetics*, **25**, 1, 177-187.

[24] Czyżak P., Jaszkiewicz A. (1997), Pareto Simulated Annealing, in: G. Fandel, T. Gal (eds.), *Multiple Criteria Decision Making. Proceedings of the XIIth International Conference, Hagen (Germany)*, Springer, Berlin, 297-307.

[25] Czyżak P., Jaszkiewicz A. (1997), The Multiobjective Metheuristic Approach for Optimization of Complex Manufacturing Systems, in: G. Fandel, T. Gal (eds.), *Multiple Criteria Decision Making. Proceedings of the XIIth International Conference, Hagen (Germany)*, Springer, Berlin, 591-592.

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

[27] Darmoni S.J., Fajner A., Mahe N., Leforestier A., Vondracek M., Stelian O., Baldenweck M. (1995), Horoplan: computer-assisted nurse scheduling using constraint-based programming, *Journal of the Society for Health Systems*, **5**, 41-54.

[28] Dasgupta D., Michalewicz Z. (1997), *Evolutionary Algorithms in Engineering Applications*, Springer, Berlin.

[29] Dempster A.P. (1967), Upper and lower probabilities induced by a multivalued mapping, *Ann. Math. Statist.*, **38**, 325-339.

[30] Dubois D., Prade H. (1980), *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, New York.

[31] Dubois D., Prade H. (1987), The mean value of a fuzzy number, *Fuzzy Sets and Systems*, **24**, 279-300.

[32] Dyer J.S. (1973), An empirical investigation of man-machine interactive approach to the solution of the multiple criteria problem, in: J. Cochrane, M. Zeleny (eds.), *Multiple Criteria Decision making*, Univeristy of S.C. Press, Columbia/S.C., 202-216.

[33] Ehrgott M., Gandibleux X. (2000), *An Annotated Bibliography of Multi-objective Combinatorial Optimization,* Technical Report 62/2000, Fachbereich Mathematik, Universitat Kaiserslautern, Kaiserslautern, Germany.

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

[35] 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.

[36] Fonseca C.M., Fleming P.J. (1995), An overview of evolutionary algorithms in multiobjctive optimisation, *Evolutionary Computation*, **3**, 1, 1-16.

[37] Fonseca C.M., Fleming P.J. (1996), On the Performance Assessment and Comparison of Stochastic Multiobjective Optimizers, in: H.-M. Voigt, W Ebeling., I. Rechenberg, H.-P. Schwefel (eds.), *Parallel Problem Solving from Nature – PPSN IV*, Lecture Notes in Computer Science, Springer, Berlin, 584-593.

[38] Fortemps P., Roubens M. (1996), Ranking and defuzzification methods based on area compensation, *Fuzzy Sets and Systems*, **82**, 319-330.

[39] Franz L.S., Lee S.M. (1980), A goal programming based interactive decision support system, *Lecture Notes in Economics and Mathematical Systems*, **190**, 110-115.

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

[41] Galinier P., Hao J.-K. (1999), *Hybrid evolutionary algorithms for graph coloring*. Technical Report, Parc Scientifique Georges Besse, Nimes (To appear in *Journal of Combinatorial Optimization*).

[42] Gambardella L.M., Taillard É. (1999), MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows, in: D. Corne, M. Dorigo, F. Glover (eds.), *New Ideas in Optimization*, McGraw-Hill, London, 63-76.

[43] Gandibleux X., Fréville A. (2000), Tabu search based procedure for solving the 0-1 multiobjective knapsack problem: the two objectives case, *Journal of Heuristics*, **6**, 361-383.

[44] Gandibleux X., Mezdaoui N., Fréville A. (1997), A multiobjective tabu search procedure to solve combinatorial optimization problems, in: R. Caballero, F. Ruiz, R. Steuer (eds.), *Advances in Multiple Objective and Goal Programming*. Lecture Notes in Economics and Mathematical Systems, 455, Springer, 291-300.

[45] Genuchten M. van (1991), Why is software late? An empirical study of reasons for delay in software development, *IEEE Transactions on Software Engineering*, **17**, 6, 582-590.

[46] Goldberg D.E. (1988), *Genetic Algorithms in Search, Optimization, and Machine Learning*, Reading, Addison-Wesley.

[47] Goldberg D. E., Segrest P. (1987), Finite Markov chain analysis of genetic algorithms, in: J.J. Grefenstette (ed.), *Genetic algorithms and their applications. Proceedings of the Second International Conference on Genetic Algorithms*, Hillsdale, NJ, 1-8.

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

[49] Hajela P., Lin C. Y. (1992), Genetic search strategies in multicriterion optimal design, *Structural Optimization*, **4**, 99-107.

[50] Hansen M.P. (1998), *Metaheuristics for multiple objective combinatorial optimization*, Ph.D. Thesis, IMM-PHS-1998-45, Technical University of Denmark, Lyngby.

[51] Hansen M.P. (2000), Use of substitute scalarizing functions to guide a local search based heuristic: the case of moTSP, *Journal of Heuristic*s, **6**, 3, 419-430.

[52] Hansen M.P., Jaszkiewicz A. (1998), *Evaluating the quality of approximations to the nondominated set*, Working paper, Institute of Mathematical Modelling Technical University of Denmark, IMM-REP-1998-7.

[53] Hapke M. (1997), *Fuzzy multi-objective project scheduling*, Ph.D. Thesis, Institute of Computing Science, Poznan University of Technology (in Polish).

[54] Hapke M., Jaszkiewicz A., Kominek P., Strugalski A. (1998), *Software project scheduling with fuzzy durations based on CASE tools information*, Research Report no. RA-004/98, Institute of Computing Sciences, Poznan University of Technology.

[55] Hapke M., Jaszkiewicz A., Słowiński R. (1994), Fuzzy Project Scheduling System for Software Development, *Fuzzy Sets and Systems*, **67**, 101-117.

[56] Hapke M., Jaszkiewicz A., Słowinski R. (1996), Interactive analysis of multiple-criteria project scheduling problems, in: *Proceedings of the The Fifth International Workshop on Project Management and Scheduling – EURO PMS'96*, Poznań, Poland, 11-13.04, 107-110.

[57] Hapke M., Jaszkiewicz A., Słowiński R. (1997), Fuzzy project scheduling with multiple criteria, in: *Proceedings of Sixth IEEE International Conference on Fuzzy Systems*, FUZZ-IEEE'97, July 1-5, Barcelona, Spain, 1277-1282.

[58] Hapke M., Jaszkiewicz A., Słowiński R. (1998), Fuzzy multi-mode resource-constrained project scheduling with multiple objectives, ch. 16, in: J. Węglarz (ed.) *Recent Advances in Project Scheduling*, Kluwer Academic Publishers, Dordrecht, 355-382.

[59] Hapke M., Jaszkiewicz A., Słowiński R. (1998), Interactive analysis of multiple-criteria project scheduling problems, *European Journal of Operational Research*, **107**, 315-324.

[60] Hapke M., Jaszkiewicz A., Słowiński R. (1998), Fuzzy approach to multi-objective project scheduling under uncertainty. in: G. Cagil, A.K. Turker, E. Oztemel, H. Taskin (eds.), *Proc. 2nd Int. Symposium on Intelligent Manufacturing Systems* (*IMS'98*), Sakarya, August 6-7, Sakarya University Press, 1998, 927-940.

[61] Hapke M., Jaszkiewicz A., Słowiński R. (2000), Pareto simulated annealing for fuzzy multi-objective combinatorial optimization, *Journal of Heuristics*, **6**, 3, 329-345.

[62] Hapke M., Kominek P., Jaszkiewicz A., Słowiński R. (2000), Integrated tools for software project scheduling under uncertainty, in: P. Brucker, S. Heitmann, J. Hurink, S. Knust (eds.), *Proc. 7th Int. Workhop on Project Management and Scheduling PMS'2000*, Osnabrueck, Germany, April 17-19, 149-151.

[63] Hapke M., Jaszkiewicz A., Żak J. (2000), The design of the physical distribution system with the application of the multiple objective mathematical programming. Case study, in: *Proceeding of the Fourth International Conference on the Multi-Objective Programming and Goal Programming MOPGP'00. Theory & Applications,* Ustroń, Poland, May 29-June 1, University of Economics in Katowice, 149-160, (to appear in Lecture Notes in Economics and Mathematical Systems).

[64] Hapke M., Słowiński R. (1996), Fuzzy priority heuristics for project scheduling, *Fuzzy Sets and Systems*, **83**, 291-299.

[65] Hilliard M. R., Liepins G. E., Palmer M., Rangarajen G. (1989), The computer as a partner in algorithmic design: Automated discovery of parameters for a multiobjective scheduling heuristic, in: R. Sharda, B.L. Golden, E. Wasil, O. Balci, W Stewart. (eds.), *Impacts of Recent Computer Advances on Operations Research.* North-Holland, New York.

[66] Holland J.H. (1975), *Adaptation in natural and artificial systems*, University of Michigan Press, Ann Arbor.

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

[68] Howell J.P. (1966), Cyclical scheduling of nursing personel, *Hospitals*, **40**, 77-85.

[69] http://lindo.com/

[70] http://www.lania.mx/~ccoello/EMOO/EMOObib.html

[71] http://www.tik.ee.ethz.ch/~zitzler/testdata.html

[72] http://www.crpc.rice.edu/softlib/tsplib/

[73] Hwang C.-L., Paidy S.R., Yoon K., Masud A.S.M. (1980), Mathematical programming with multiple objectives: A tutorial, *Computers and Operations Research*, **7**, 5-31.

[74] Iredi S., Merkle D., Middendorf M. (to appear), Bi-criterion optimization with multi colony ant algorithms, in: *Proceeding of the First International Conference on Evolutionary Multi-Criterion Optimization*. Lecture Notes in Computer Science, Springer, Berlin.

[75] Ishibuchi H., Murata T. (1998), Multi-Objective Genetic Local Search Algorithm and Its Application to Flowshop Scheduling, *IEEE Transactions on Systems, Man and Cybernetics*, **28**, 3, 392-403.

[76] Jacquet-Lagrèze E., Meziani R., Słowiński R. (1987), MOLP with an interactive assessment of a piecewise-linear utility function, *European Journal of Operational Research*, **31**, 350-357.

[77] Jaszkiewicz A. (1996), Self adapting metaheuristic procedure for multiple objective combinatorial optimization, (in Polish), *Zeszyty Naukowe Politechniki Śląskiej*, **117**, 137-147.

[78] Jaszkiewicz A. (1997), A metaheuristic approach to multiple objective nurse scheduling, *Foundations of Computing and Decision Sciences*, **22**, 3, 169-184.

[79] Jaszkiewicz A. (1997), *Software engineering*, (in Polish), Helion, Gliwice.

[80] Jaszkiewicz A. (1997), Multiple objective combinatorial optimization – survey of solution techniques, (in Polish), in: T. Trzaskalik (ed.), *Applications of operational research*, Absolwent, Łódź, 251-264.

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

[82] Jaszkiewicz A. (1999), Improving performance of genetic local search by changing local search space topology, *Foundations of Computing and Decision Sciences*, **24**, 2, 77-84.

[83] Jaszkiewicz A. (2000), On the computational effectiveness of multiple objective metaheuristics, in: *Proceeding of the Fourth International Conference on the Multi-Objective Programming and Goal Programming MOPGP'00. Theory & Applications*, Ustroń, Poland, May 29-June 1, University of Economics in Katowice, 201-214, (to appear in Lecture Notes in Economics and Mathematical Systems).

[84] Jaszkiewicz A. (2000), *On the performance of multiple objective genetic local search on the 0/1 knapsack problem. A comparative experiment*, Research report, Institute of Computing Science, Poznan University of Technology, RA-002/2000.

[85] Jaszkiewicz A. (to appear). Genetic local search for multiple objective combinatorial optimization. *European Journal of Operational Research.*

[86] Jaszkiewicz A. (to appear), Multiple objective genetic local search algorithm, in: M. Köksalan, S. Zionts (eds.), P*roceedings of the Fifteenth International Conference on Multiple Criteria Decision Making,* Ankara July 10-14, 2000.

[87] Jaszkiewicz A., Biedka T. (2000), *Tests of modified DPC recombination operator for traveling salesperson problem*, Research report, Institute of Computing Science, Poznan University of Technology, RB-032/2000.

[88] Jaszkiewicz A., Hapke M., Kominek P. (to appear), Performance of multiple objective evolutionary algorithms on distribution system design problem – computational experiment. in: *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, Springer, Berlin.

[89] Jaszkiewicz A., Ferhat A.B. (1999), Solving multiple criteria choice problems by interactive trichotomy segmentation, *European Journal of Operational Research*, **113**, 2, 271-280.

[90] Jaszkiewicz A., Kaszuba R., Kominek P. (1999), Genetic Local Search for Optimization of the Transportation System of Municipal Cleaning Service, in: A. Adamski, A. Rudnicki, J. Żak (eds.), *Proceedings of the Modelling and Management in Transportation International Conference*, Poznań, Poland, 12-16.10, 205-210.

[91] Jaszkiewicz A. Kominek P. (to appear), Developing efficient genetic local search type heuristics on the basis of global convexity tests – vehicle routing example, *European Journal of Operational Research*.

[92] Jaszkiewicz A., Słowiński R. (1995), The Light Beam Search – outranking based interactive procedure for multiple-objective mathematical programming, in: P.M. Pardalos, Y. Siskos, C. Zopounidis (eds.), *Advances in Multicriteria Analysis*, Kluwer Academic Publishers, Dordrecht, 129-146.

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

[94] Jaszkiewicz A., Słowiński R. (1999), The 'Light Beam Search' Approach – an Overview of Methodology and Applications, *European Journal of Operational Research*, **113**, 2, 300-314.

[95] Jelinek R.C., Kavois J.A. (1992), Nurse staffing and scheduling: past solutions and future directions, *Journal of the Society for Health Systems*, **3**, 75-82.

[96] Jędrzejowicz P. (1999), Social learning algorithm as a tool for solving some difficult scheduling problems, *Foundations of Computing and Decision Sciences*, **24**, 2, 51-66.

[97] Kahneman D., Slovic P., Tversky A. (1982), *Judgement under uncertainty: heuristics and biases*, Cambridge University Press, Cambridge.

[98] Keeney R.L., Raiffa H. (1976), *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*, Wiley, New York.

[99] Keller C.P., Goodchild M.F. (1988), The multiobjective vending problem: a generalisation of the traveling salesman problem, *Environ. Plann. B: Plann. Design*, **15**, 447-460.

[100] Knowles J.D., Corne D.W. (1999), The Pareto archived evolution strategy: a new baseline algorithm for multiobjective optimisation, in *1999 Congress on Evolutionary Computation*, Piscataway, NJ, IEEEE Service Center, 98-105.

[101] Knowles J.D., Corne D.W. (2000), M-PAES: A Memetic Algorithm for Multiobjective Optimization, in: *Proceedings of the 2000 Congress on Evolutionary Computation*, vol. 1, Piscataway, New Jersey, July. IEEE Service Center, 325-332.

[102] Knowles J.D., Corne D.W. (2000), A Comparison of Diverse Approaches to Memetic Multiobjective Combinatorial Optimization, in: *Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program*, Las Vegas, Nevada, July, 103-108.

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

[104] Kołodziejczyk W. (1986), Orlovsky's concept of decision-making with fuzzy preference relation – further results, *Fuzzy Sets and Systems*, **19**, 11-20.

[105] Korhonen P. (1988), A Visual Reference Direction Approach to Solving Discrete Multiple Criteria Problems, *European Journal of Operational Research*, **34**, 2, 152-159.

[106] Korhonen P., Laakso J. (1986), Solving generalized goal programming problems using a visual interactive approach, *European Journal of Operational Research*, **26**, 355-363.

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

[108] 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.

[109] Kursawe F. (1992), Evolution strategies for vector optimization, *Proceedings of the Tenth International Conference on Multiple Criteria Decision Making, Taipei 19-24.07*, vol. 1, 187-193.

[110] Laarhoven P.J.M., Aarts E.H.L. (1987), *Simulated Annealing: Theory and Applications,* Reidel, Dordrecht.

[111] Larichev O.I., Polyakov O.A., Nikiforov A.D. (1987), Muticriterion linear programming problems (analytical survey), *Journal of Economic Psychology*, **8**, 389-407.

[112] Levin A. (to appear), Lexicographical local search and the p-center problem, *European Journal of Operational Research*.
[113] Liepins G. E., Hilliard M. R., Richardson J., Palmer M. (1990), Genetic algorithms application to set covering and traveling salesman problems, in: D.E. Brown, C.C. White (eds.), *Operations research and Artificial Intelligence: The integration of problem-solving strategies,* Kluwer Academic, Norwell, Massachusetts, 29-57.
[114] Lotfi V., Stewart T.J. and Zionts S. (1992), An aspiration-level interactive model for multiple criteria decision making, *Computers and Operations Research*, **19**, 677-681.
[115] Luce D. (1956), Semiorders and a theory of utility discrimination, *Econometrica*, 24, 178-191.
[116] Lukman D., May J.H. Shuman L.J., Wolfe H.B. (1991), Knowledge-based schedule formulation and maintenance under uncertainty, *Journal of the Society for Health Systems*, **2**, 42-64.
[117] 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.
[118] Mariano C.E., Morales E. (1999), MOAQ and ant-Q algorithm for multiple objective optimization problems, in: W. Banzhaf, J. Daida, A.E. Eiben, M.H. Garzon, V. Honavar, M. Jakiela, R.E. Smith (eds.), *Proceeding of the Genetic and Evolutionary Computation Conference*, Orlando, Florida, USA, 13-17 July, vol. 1, 894-901.
[119] Merz P., Freisleben B. (1997), Genetic Local Search for the TSP: New Results, in: *Proceedings of the 1997 IEEE International Conference on Evolutionary Computation*, IEEE Press, 159-164.
[120] Michalewicz Z. (1992), *Genetic algorithms + data structures = evolution programs*, Springer, Berlin.
[121] Michalewicz Z. Arabas J. (1994), Genetic algorithms for the 0/1 knapsack problem, in: Z.W. Raś, M. Zemankova (eds.), *Methodologies for Intelligent Systems (ISMIS'94)*, Berlin, 134-143.
[122] Moscorola J., Roy B. (1977), Procédure automatique d'examen de dossiers fondée sur une segmentation trichotomique en présence de critères multiple, *RAIRO*, **11**, 2, 145-173.
[123] Mühlenbein H. (1991), Evolution in time and space – the parallel genetic algorithm, in: G.J.E. Rawlins (ed.), *Foundations of genetic algorithms*, Morgan Kaufman Publishers, 316-337.
[124] Murata T., Ishibuchi H., Tanaka H. (1996), Multi-objective genetic algorithm and its application to flowshop scheduling, *Computers Ind. Eng.*, **30**, 4, 957-968.
[125] Nabrzyski J. (2000), *User preference driven expert system for solving multiobjective project scheduling problems*, Ph.D. Thesis, Institute of Computing Science, Poznan University of Technology (in Polish).
[126] Nakayama H., Sawaragi Y. (1984), Satisficing trade-off method for multiobjective programming, in: M. Grauer, A.P. Wierzbicki (Eds.), *Interactive Decision Analysis*, LNEMS, vol. 229, Springer, Berlin, 114-122.
[127] Osman I., Laporte G. (1996), Metaheuristics: A bibliography, *Annals of Operations Research*, **65**, 513-623.

[128] Osyczka A., Kundu S. (1995), A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm, *Structural Optimization*, **10**, 94-99.

[129] Ozkarahan I. (1991), An integrated nurse scheduling model, *Journal of the Society for Health Systems*, **3**, 79-101.

[130] Pressman R.S. (1993), *A managers guide to software engineering*, McGraw-Hill, NewYork.

[131] Radcliffe N.J., Surry P.D. (1994), Formal memetic algorithms, in: T. Fogarty (ed.), *Evolutionary Computing: AISB Worksh*op, Springer, Berlin.

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

[133] Ritzel B. J., Wayland E. J., Ranjithan S. (1994), Using genetic algorithms to solve a multiple objective groundwater pollution containment problem, *Water Resources Research*, **30**, 5, 1589-1603.

[134] Rommelfanger H. (1990), FULPAL: An interactive method for solving (multiobjective) fuzzy linear programming problems, section 5, in: R. Słowiński, J. Teghem (eds.), *Stochastic Versus Fuzzy Approaches to Multiobjective Mathematical Programming under Uncertainty*, Kluwer Academic Publishers, Dordrecht, 279-299.

[135] Rosenblatt M.J., Sinuany-Stern Z. (1989), Generating the discrete effcient frontier to the capital budgeting problem, *Operations Research*, **37**, 3, 384-394.

[136] Rosenthal R.E. (1985), Principles of multiobjective optimization, *Decision Sciences*, **16**, 133-152.

[137] Ross T.J. (1995), *Fuzzy Logic with Engineering Applications*, McGraw-Hill, New York.

[138] Roubens M. (1990), Inequality constraints between fuzzy numbers and their use in mathematical programming, section 7, in: R. Słowiński, J. Teghem (Eds.), *Stochastic Versus Fuzzy Approaches to Multiobjective Mathematical Programming under Uncertainty*, Kluwer Academic Publishers, Dordrecht, 321-330.

[139] Roy B. (1985), *Méthodologie Multicritère d'Aide à la Décision*, Economica, Paris.

[140] Roy B. (1990), The outranking approach and the foundations of ELECTRE methods, in: C.A. Bana e Costa (ed.), *Readings in Multiple Criteria Decision Aid*, Springer, Berlin, 155-183.

[141] Roy B., Bouyssou D. (1993), *Aide Multicritère à la Décision: Méthodes et Cas*, Economica, Paris.

[142] Sayers T. (to appear), Issues in the development of a multi-objective GA to optimise traffic signal controller parameters, in: M. Köksalan, S. Zionts (eds.), *Proceedings of the 15th International Conference on MCDM,* Ankara, July 10-14.

[143] Shafer G. (1976), *A Mathematical Theory of Evidence*, Princeton University Press, Princeton, N.J.

[144] 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.

[145] Serafini P. (1987), Some considerations about computational complexity for multiobjective combinatorial problems, *Lecture Notes in Economics and Mathematical Systems,* **294**, 222-232.

[146] Serafini P. (1992), Simulated annealing for multiple objective optimization problems, in: *Proceedings of the Tenth International Conference on Multiple Criteria Decision Making,* Taipei 19-24.07, vol. 1, 87-96.

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

[148] Shelokar P.S., Adhikari S., Vakil R., Jayaraman V. K., Kulkarni B. D. (to appear), Multiobjective ant algorithm: combination of strength Pareto fitness assignment and thermodynamic clustering, *Foundations of Computing and Decision Sciences*.

[149] Słowiński R. (1981), Multiobjective network scheduling with efficient use of renewable and non renewable resources, *European Journal of Operational Research*, **7**, 265-273.

[150] Słowiński R. (1984), Review of multiple objective programming methods. Part I (in Polish), *Przegląd Statystyczny*, **31**, 47-63.

[151] Słowiński R. (1989), Multiobjective project scheduling under multiple-category resource constraint, in: R. Słowiński, J. Węglarz (eds.), *Advances in Project Scheduling*, Elsevier, Amsterdam, 151-167.

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

[153] Steuer R.E. (1977), An interactive multiple objective linear programming procedure, *TIMS Studies in the management Sciences*, **6**, 225-239.

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

[155] Steuer R.E. and Choo E.-U. (1983), An interactive weighted Tchebycheff procedure for multiple objective programming, *Mathematical Programming*, **26**, 326-344.

[156] Steuer R.E., Silverman J., Whishman A.W. (1993), A combined Tchebycheff/aspiration criterion vector interactive multiobjective programming procedure, *Management Science*, **39**, 1255-1260.

[157] 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**, 3, 462-472.

[158] Suppapitnarm A., Parks T. (to appear), Simulated annealing: an alternative approach to true multiobjective optimization, in: *Genetic and Evolutionary Computation Conference*, Orlando, Florida, 406-407.

[159] Surry P.D., Radcliffe N.J., Boyd I.D. (1995), A Multi-Objective Approach to Constrained Optimisation of Gas Supply Networks: The COMOGA Method, in: T.C. Fogarty (ed.), *Evolutionary Computing*, AISB Workshop. Selected Papers, Lecture Notes in Computer Science, Sheffield, U.K., Springer, 166-180.

[160] Taillard É. D. (1995), Comparison of iterative searches for the quadratic assignment problem, *Location Science*, **3**, 87-105.
[161] Tamaki H., Kita H., Kobayashi S. (1996), Multi-Objective Optimization by Genetic Algorithms: A Review, In: T. Fukuda, T. Furuhashi (eds.), *Proceedings of the 1996 International Conference on Evolutionary Computation (ICEC'96)*, Nagoya, Japan, IEEE, 517-522.
[162] Taner O.V., 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.
[163] Tao G., Michalewicz Z. (1998), Evolutionary Algorithm for TSP, in: A.E. Eiven, T. Bäck, M. Schoenauer, H.-P. Schwefel (eds.), *Proceedings of the 5th Parallel Problem Solving from Nature Conference*. Lecture Notes in Computer Science, **1498**, Springer, Berlin.
[164] Todd D.S., Sen P. (1997), A Multiple Criteria Genetic Algorithm for Containership Loading, in: T. Bäck (ed.), *Proceedings of the Seventh International Conference on Genetic Algorithms*, San Mateo, California, July, Morgan Kaufmann, 674-681.
[165] Törn A. (1980), A sampling-search-clustering approach for exploring the feasible/efficient solutions of MCDM problems, *Computers and Operations Research*, **7**, 67-79.
[166] Tuyttens D., Teghem J., Fortemps P., Van Nieuwenhuyse K. (2000), Performance of the MOSA method for the bicriteria assignment problem, *Journal of Heuristics*, **6**, 3, 295-310.
[167] Ulungu E.L. (1993), *Optimisation combinatoire multicritère: détermination de l'ensemble des solutions efficaces et méthodes interactives*, Thèse de doctorat, Université de Mons-Hainaut (Belgim).
[168] Ulungu E.L., Teghem J. (1994), Multi-ojective Combinatorial Optimization Problems: A Survey, *Journal of Multi-Criteria Decision Analysis*, **3**, 83-101.
[169] Ulungu E.L., Teghem J., Fortemps Ph., (1995), Heuristic for multiobjective combinatorial optimization problems by simulated annealing, in: J. Gu, G. Chen, Q. Wei, S. Wang (eds.), *MCDM: Theory and Applications 1995*, Sci-Tech, Windsor, 229-238.
[170] Ulungu E.L., Teghem J., Fortemps Ph., Tuyttens D. (1999), MOSA method: a tool for solving multiobjective combinatorial optimization problems, *Journal of Multi-Criteria Decision Analysis*, **8**, 221-236.
[171] Ulungu B., Teghem J., Ost C. (1998), Efficiency of interactive multi-objective simulated annealing through a case study, *Journal of the Operational Research Society*, **49**, 1044- 1050.
[172] Vanderpooten D. (1989), The interactive approach in MCDA: a technical framework and some basic conceptions, *Mathematical and Computer Modelling*, **12**(10), 1213-1220.
[173] Van Veldhuizen D.A. (1999), *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*, Ph.D. thesis, Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio.

[174] Van Veldhuizen D. A., Lamont G. B. (1998), Evolutionary Computation and Convergence to a Pareto Front, in: J.R. Koza (ed.), *Late Breaking Papers at the Genetic Programming 1998 Conference*, Stanford University, California, July, Stanford University Bookstore, 221-228.

[175] Viana A. Sousa J.P. de (2000), Using metaheuristics in multiobjective resource constrained project scheduling, *European Journal of Operational Research*, **120**, 359-374.

[176] Wallenius J. (1975), Comparative evaluation of some interactive approaches to multicriterion optimization, *Management Science*, **21**, 1387-1396.

[177] Wolpert D.H., Macready W. G. (1997), No free lunch theorem for optimization, *IEEE Transactions on Evolutionary Computation*, **1** (1), 67-82.

[178] Węglarz J. (1980), On certain models of resource allocation problems, *Kybernetes*, **9**, 61-66.

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

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

[181] White D.J. (1990), A bibliography on the applications of mathematical programming multiple-objective methods, *Journal of the Operational Research Society*, **41**, 669-691.

[182] White D.J. (1991), The maximal-dispertion problem, *IMA Journal of Mathematics Applied in Bussiness & Industry*, **3**, 131-140.

[183] Yu G. (1998), *Industrial Applications of Combinatorial Optimization*, Kluwer Academic Publisher, Boston.

[184] Yu W. (1992), *Electre Tri: Aspects méthodologiques et manuel d'utilisation*, Document du LAMSADE 74, Université de Paris Dauphine.

[185] Zionts S. (1981), A Multiple Criteria Method for Choosing among Discrete Alternatives, *European Journal of Operational Research*, **7**, 1, 143-147.

[186] Zionts S., Wallenius J., (1983), An interactive multiple objective linear programming method for a class of underlying nonlinear utility functions, *Management Science*, **29,** 519-529.

[187] Zitzler E. Thiele L. (1998), Multiobjective Optimization Using Evolutionary Algorithms – A Comparative Study, in: A.E. Eiben (ed.), *Parallel Problem Solving from Nature V*, Amsterdam, September, Springer, 292-301,

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

Appendix. MOEALib++ – implementation of multiple objective metaheuristics in the form of a library of C++ classes

MOEALib++ (**M**ultiple **O**bjective **E**volutionary **A**lgorithms **Lib**rary in C**++**) is a library of C++ classes that implements a number of multiple objective metaheuristics. It has been developed on the basis of former implementations of the methods described in this work – Pareto simulated annealing (PSA) and multiple objective genetic local search (MOGLS). The goals of development of the library may be summarized as follows:

- to implement the main multiple objective metaheuristics in a consistent way,
- to allow easy adaptation of the methods to a given problem,
- to support evaluation of multiple objective metaheuristics on a given problem,
- to allow easy incorporation of new multiple objective metaheuristics and to support the development of new methods.

At present the library includes the following methods:

- Pareto simulated annealing (PSA) (see section 5),
- Serafini's multiple objective simulated annealing (SMOSA) [146][147],
- Ulungu's et al. multiple objective simulated annealing (MOSA) [170],
- multiple objective genetic local search (MOGLS) (see section 6),
- Ishibuchi's and Murata's multiple objective genetic local search (IMMOGLS) [75],
- multiple objective multiple start local search (MOMSLS) (see section 10.3.3),
- Pareto ranking-based multiple objective genetic algorithm (ParetoGA); at present the implementation follows the proposition of Fonseca and Flemming [35], however, in the future it will embody other versions of Pareto ranking-based algorithms.

Each method is implemented in a separate template class that utilizes a number of common library classes. The template classes are instantiated with classes corresponding to solutions of a given problem.

In order to adapt one or more of the multiple objective metaheuristics to a given problem one has to implement a new class corresponding to the problem's solution by deriving from the library class TMOEASolution. A detailed design pattern described in the documentation of MOEALib++ illustrates the suggested way of adapting the library to a given problem.

The library is implemented in standard C++. It uses standard template library – STL. Thus, it may be compiled under any standard C++ compiler. The classes of thelibrary are presented in figure Figure 48.

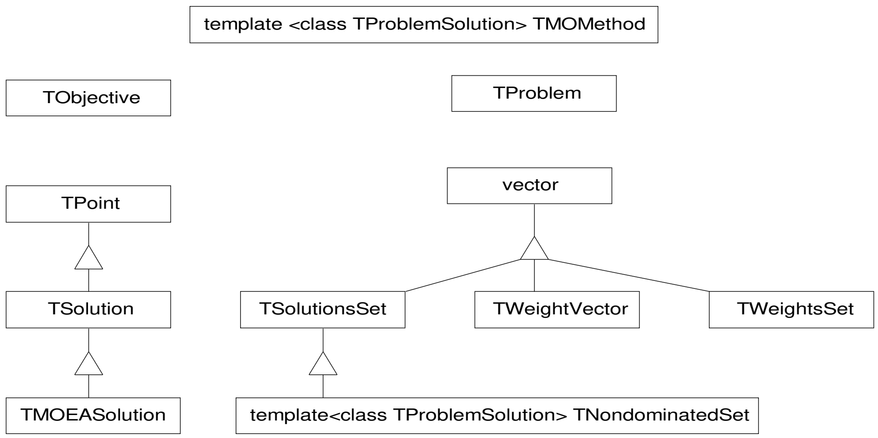

Figure 48. Classes of the MOEALib++ library