Jaszkiewicz A., Hapke M., Kominek P. (2001), Performance of multiple objective evolutionary algorithms on distribution system design problem – computational experiment, in: E. Zitzler, K. Deb, L. Thiele, C.A. Coello Coello, D. Corne (eds.) *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, **1993**, Springer, Berlin, 241-255.
Draft version

Performance of multiple objective evolutionary algorithms on a distribution system design problem - computational experiment

Andrzej Jaszkiewicz, Maciej Hapke, Paweł Kominek*

Institute of Computing Science, Poznań University of Technology
ul. Piotrowo 3a, 60-965 Poznań, Poland
{Jaszkiewicz|Hapke|Kominek}@cs.put.poznan.pl

Abstract. The paper presents a comparative experiment with four multiple objective evolutionary algorithms on a real life combinatorial optimization problem. The test problem corresponds to the design of a distribution system. The experiment compares performance of a Pareto ranking based multiple objective genetic algorithm (Pareto GA), multiple objective multiple start local search (MOMSLS), multiple objective genetic local search (MOGLS) and an extension of Pareto GA involving local search (Pareto GLS). The results of the experiment clearly indicate that the methods hybridizing recombination and local search operators by far outperform methods that use one of the operators alone. Furthermore, MOGLS outperforms Pareto GLS.

1 Introduction

Multiple objective metaheuristics, e.g. multiple objective versions of genetic algorithms, simulated annealing or tabu search constitute one of the most active directions of research in multiple objective optimization [17]. Methods of this kind are claimed to be able to solve both continuous nonlinear optimization problems and large scale combinatorial optimization problems. A significant number of real life applications of multiple objective metaheuristics to continuous nonlinear problems (e.g. in engineering design) have been reported in the literature. At the same time the number of successful applications to real life multiple objective combinatorial optimization problems seems to be rather limited [17]. In our opinion, this is not due to the lack of potential applications but due to the difficulty of the problems.

In this paper we describe a comparative experiment on a real life combinatorial optimization problem corresponding to a distribution system design. The problem was analyzed in a project in which the authors were involved. 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 parameters' values but have the same mathematical form, see e.g.[15]),
- 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.

* This work has been supported by KBN grant No. 8T11F00619.

Four multiple objective metaheuristics have been tested on this problem. We used Pareto ranking based multiple objective genetic algorithm proposed by Fonseca and Flemming [3] (Pareto GA), multiple objective multiple start local search (MOMSLS) with random weight vectors, multiple objective genetic local search (MOGLS) [7] and an extension of Pareto GA involving local search (Pareto GLS). Two different quality measures were used in this experiment - coverage of two approximations to the nondominated set and estimation of the expected value of weighted Chebycheff scalarizing function on an approximation. In addition running time of the methods was taken into account. All the methods used the same solution encoding, the recombination and local search operators and the same way of constructing initial solutions. All the methods were implemented with MOEALIB++ library [16] and shared most of the C++ code.

The paper is organized in the following way. In the next section some basic definitions are given. The distribution system design problem is described in the third section. In the fourth section the four methods used in the experiment are briefly described. The fifth section describes customization of the methods to the problem. Design and the results of the experiment are characterized in the sixth section. The last section contains conclusions.

2 Problem statement and basic definitions

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

$$\min\{f_1(\mathbf{x}) = z_1, ..., f_J(\mathbf{x}) = z_J\} \quad \text{(P1)}$$
$$\text{s.t.} \quad \mathbf{x} \in D,$$

where: *solution* $\mathbf{x} = [x_1, ..., x_I]$ is a vector of *decision variables*, D is the set of feasible solutions. If each decision variable takes discrete values from a finite set the problem is combinatorial.

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

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

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

An *approximation to 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.

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

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

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

Range equalization factors [11, ch. 8.4.2] are defined in the following way:

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

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 Chebycheff scalarizing functions are defined in the following way:

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

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 Chebycheff scalarizing function of this type has at least one global optimum (minimum) belonging to the set of efficient solutions. For each efficient solutions **x** there exists a weighted Chebycheff scalarizing function s such that **x** is global optimum (minimum) of s [11, ch. 14.8].

Linear scalarizing functions are defined in the following way:

$$s(\mathbf{z}, \Lambda) = \sum_j \lambda_j z_j = \sum_j \lambda_j f_j(\mathbf{x}).$$

Each weighted Chebycheff scalarizing function of this type has at least one global optimum (minimum) belonging to the set of efficient solutions [11, ch. 14.8].

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.*

3 The distribution system design problem

An international company operating in Poland is going to completely rebuild its distribution system. The company entered the 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. 85% of the turnover is generated at the Polish local market and the remaining 15% is an export to Eastern European countries. At the Polish market the company's products are mainly sold to wholesalers (60% of sales) and the chains of large retailers (hipermarkets, supermarkets) – 20% of sales.

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 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 transportation cost, warehousesing cost and 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).

Below we present mathematical programming formulation of the problem:

3.1 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 (DB_j)- annual demand of region j for products of production plant A (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} (TCB_{jl})- average transportation cost from production plant A (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 (CCB)- average daily cost of locked-up capital per pallet produced in production plant A (B) in [PLN/day],
$MSSA$ ($MSSB$)– annual cost of minimum safety stock of pallets produced in production plant A (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.

3.2 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.

3.3 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:

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

3.4 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:

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

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, and.

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:

$$ATCA_{il} = TCA_i \sum_{j=1}^{J} x_{ij} DA_j + \sum_{j=1}^{J} x_{ij} TC_{ij} DA_j \ .$$

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

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

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

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

where DHA_{il} and DHB_{il} are average headways 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:

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

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:

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

4 Methods used in the experiment

4.1 Multiple objective genetic local search

Genetic Local Search (GLS) is a metaheuristic algorithm that hybridizes recombination operators with local search, or more generally, with other local improvement heuristics. Other frequently used names are Memetic Algorithms or Hybrid Genetic Algorithms. 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 efficient in combinatorial optimization. In particular, methods of this type were able to improve the best known results on a number of well studied combinatorial optimization problems, e.g. Travelling Salesperson Problem [10], Graph Coloring Problem [2] and Quadratic Assignment Problem [13]. Thus, in our opinion, development of multiple objective versions of GLS is one of the most promising directions of research.

In this paper, we use multiple objective genetic local search (MOGLS) algorithm proposed by the first author in [7]. In [7] the algorithm was tested on multiple objective Travelling Salesperson Problem and outperformed MOGLS algorithm of Ishibuchi and Murata [6], MOGLS based on the idea of MOSA method proposed by Ulungu et al. [12] and Pareto ranking-based evolutionary algorithm [3] proposed by Fonseca and Fleming. In [8] the algorithm outperformed Strength Pareto Approach (SPEA) algorithm [15] and Memetic-Pareto Archive Evolution Strategy (M-PAES) [9] on the 0/1 multiple objective knapsack problem. Below we describe main ideas of the method.

The goal of multiple objective metaheuristics is to generate good approximations to the nondominated set. Of course, the best possible approximation is the whole nondominated set itself. Note that all weighted Chebycheff scalarizing functions have optima in the nondominated set and each nondominated point is an optimum of a weighted Chebycheff scalarizing functions (see section 2). Thus, finding the whole nondominated set is equivalent to finding optima of all weighted Chebycheff scalarizing functions. Hence, we reformulate the goal of multiple objective metaheuristics as simultaneous optimization of all weighted Chebycheff scalarizing functions. In fact, it is enough to consider all weighted Chebycheff scalarizing functions with normalized weight vectors.

MOGLS implements the idea of simultaneous optimization of all weighted Chebycheff scalarizing functions with normalized weight vectors by random choice of a 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 heuristic that improves locally the value of the current scalarizing function. An approximation of the ideal point composed of the best known values on each objective is used as the reference point. The reference point changes (improves) during the run of the method. The normalized weight vectors are drawn at random with uniform probability distribution $p(\Lambda)$, i.e. a distribution for which:

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

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

In our opinion, the main reason of the good performance of the single objective GLS is that the method combines features of two good solutions. Offsprings 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. In each iteration, our MOGLS algorithm constructs a temporary population composed of K solutions being the best among known solutions on the current scalarizing function. Then, two different solutions are drawn for recombination from the temporary population with uniform probability. The algorithm of MOGLS is summarized in Fig. 1.

The outcome of MOGLS is an approximation to the nondominated set *PP* composed of all potentially nondominated points (and corresponding potentially Pareto-optimal solutions) generated by the method. Updating the set of potentially nondominated points *PP* with solution $\mathbf{x}$ consists of:

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

In general, the scalarizing functions with random normalized weight vectors should be applied to normalized objective function values (see section 2). The objectives could e.g. be normalized with the range equalization factors based on the objectives' ranges in *PP*. This approach is used in the experiment.

The original idea of the method was to store all generated solutions in the current set of solutions *CS* from which temporary 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 $K \times S$. In each iteration the newly generated solution is added to the beginning of the queue if it is better than the worst solution in the temporary population and different to all solutions in the temporary population. If the size of the queue is bigger than $K \times S$ then the last solution from the queue is removed.

4.2 Multiple start local search algorithm

This simple algorithm corresponds to the ***Initialization***, ***Generation of the first approximation of the ideal point*** and ***Generation of the initial set of solutions*** phases of the MOGLS algorithm. At each iteration the algorithm draws at random a weight

vector, constructs new random solution and applies to it local search. The outcome of the method is the set of potentially Pareto-optimal solutions found among the local optima. The algorithm is denoted by MOMSLS.

Parameters: K – size of the temporary population, optional parameter S – number of initial solutions, stopping criterion
Initialization:
The set of potentially Pareto-optimal solution PP:=∅
The current set of solutions CS:=∅
Generation of the first approximation of the ideal point:
for each objective f_j
Construct randomly a new feasible solution **x**
Improve locally objective f_j starting from 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**
Improve locally the scalarizing function s with weight vector Λ starting from solution **x** obtaining **x**'
Add **x**' to the current set of solutions CS
Update set PP with **x**'
until CS contains S solutions
Main loop:
repeat
Draw at random a weight vector Λ
From CS select K different solutions being the best on scalarizing function s forming temporary population TP
Draw at random with uniform probability two solutions $\mathbf{x}_1$ and $\mathbf{x}_2$ from TP.
Recombine $\mathbf{x}_1$ and $\mathbf{x}_2$ obtaining $\mathbf{x}_3$
Improve locally the scalarizing function s with weight vector Λ starting from solution $\mathbf{x}_3$ obtaining $\mathbf{x}_3$'
if $\mathbf{x}_3$' is better on s than the worst solution in TP and different in the decision space to all solutions in TP **then**
Add $\mathbf{x}_3$' to the current set of solutions CS
Update set PP with $\mathbf{x}_3$'
until the stopping criterion is met
Outcome: the set of potentially Pareto-optimal solutions PP.

Fig. 1. Algorithm of multiple objective genetic local search - MOGLS.

4.3 Pareto ranking based multiple objective genetic algorithm

At present probably the most often used multiple objective metaheuristics are multiple objective genetic algorithms based on Pareto ranking (compare [14] ch. 3.3.2.2 and 3.3.2.3 and [17]). Although algorithms of this kind differ in some details, they share

the idea of assigning solutions fitness on the basis of a ranking induced by the dominance relation proposed for the first time in [4] In this paper we use the particular implementation of the idea proposed in [3]. The algorithm uses fitness sharing in the objective space [3].

The algorithm proposed in [3] was extended by the use of the set of potentially Pareto-optimal solutions *PP* (secondary population) updated with each newly generated solution. The algorithm is denoted by Pareto GA.

4.4 Pareto ranking based multiple objective genetic local search algorithm

This algorithm 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. The algorithm is denoted by Pareto GLS.

5 Adaptation of the methods to the distribution system design problem

The solutions are encoded by I lists of regions. 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.

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

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

The details of the recombination operator algorithm are presented in Fig. 2.

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 Chebycheff scalarizing function but it resulted in worse performance of all the algorithms. Note that one of the quality measures is average value of the weighted Chebycheff scalarizing function. The value of the measure was, however, always better if the linear function was used in local search.

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

for each potential location of 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
        else if one of the locations is marked as *preliminary selected*
            Assign the region to the distribution center at this location
        else
            Mark a randomly selected location as *preliminary selected* and assign the region to the distribution center at this location

Fig. 2. The algorithm of the recombination operator.

6 Computational experiment

6.1 Quality measures

One of the quality measures used in the experiment is the coverage of two approximations to the nondominated set. Let A and B be two approximations. The coverage measure is defined as:

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

The value $C(A, B) = 1$ means that all points in B are dominated by or equal to (covered by) some points in A. The value $C(A, B) = 0$ means that none point in B is covered by any point in A. Note that in general $C(A, B) \neq C(B, A)$. This quality measure was used in [15] and [8].

As the second quality measure we use the estimation of the expected value of weighted Chebycheff scalarizing function on approximation A over the set of normalized weight vectors Λ [5]:

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

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.

In order to estimate the expected value we use numerical integration, i.e. we calculate average value of the weighted Chebycheff scalarizing function over a large sample of systematically generated normalized weight vector. We 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 this produces $\binom{k+J-1}{J-1}$ weight vectors. The estimation of measure R is calculated as:

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

The parameter k was set to 50.

The C++ codes used to calculate the two quality measures are inlcuded in the MOEALIB++ library [16].

6.2 Experiment design

Because of confidence reasons the real life data have been slightly modified. All of them, however, are in realistic ranges.

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 [7]. This number of initial solutions assures that on average 30 best solutions on a randomly chosen weighted linear scalarizing function 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.

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 in the case of MOGLS and Pareto GLS.

Pareto GA has used a population of size 100 and has been performing 400 generations. As this method does not use local search in performs more recombinations per second 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 1. 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

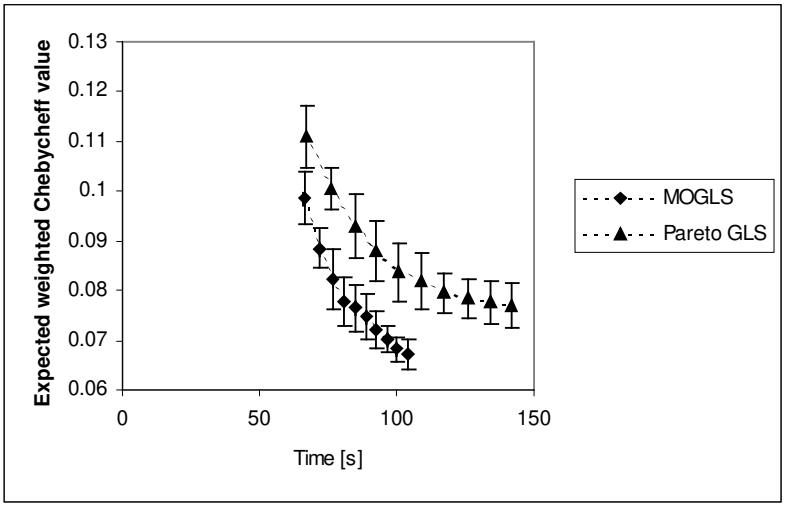

Fig. 3. Results of comparison with expected value of weighted Chebycheff scalarizing function. The chart shows ± standard deviation range of the quality measure.

6.3 Results

Table 1 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.

Table 2 presents results of evaluation with expected value of weighted Chebycheff scalarizing quality measure. 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 synergy of the two operators. Pareto GLS is a simple combination of the two latter methods and does not include any element that cannot be found in Pareto GA or MOMSLS.

Table 2. Results of comparison with expected value of weighted Chebycheff scalarizing.

MOGLS			Pareto GLS		
Number of recombinations	Value of quality measure R	Standard deviation of quality measure R	Number of recombinations	Value of quality measure R	Standard deviation of quality measure 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 R	Standard deviation of quality measure R	Number of generations	Value of quality measure R	Standard deviation of quality measure 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

Furthermore, MOGLS outperforms Pareto GLS. It generates better approximations to the nondominated set in shorter time. 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. From the other side, MOGLS has higher memory requirements than Pareto GLS. Fig. 3 illustrates performance of MOGLS and Pareto GLS.

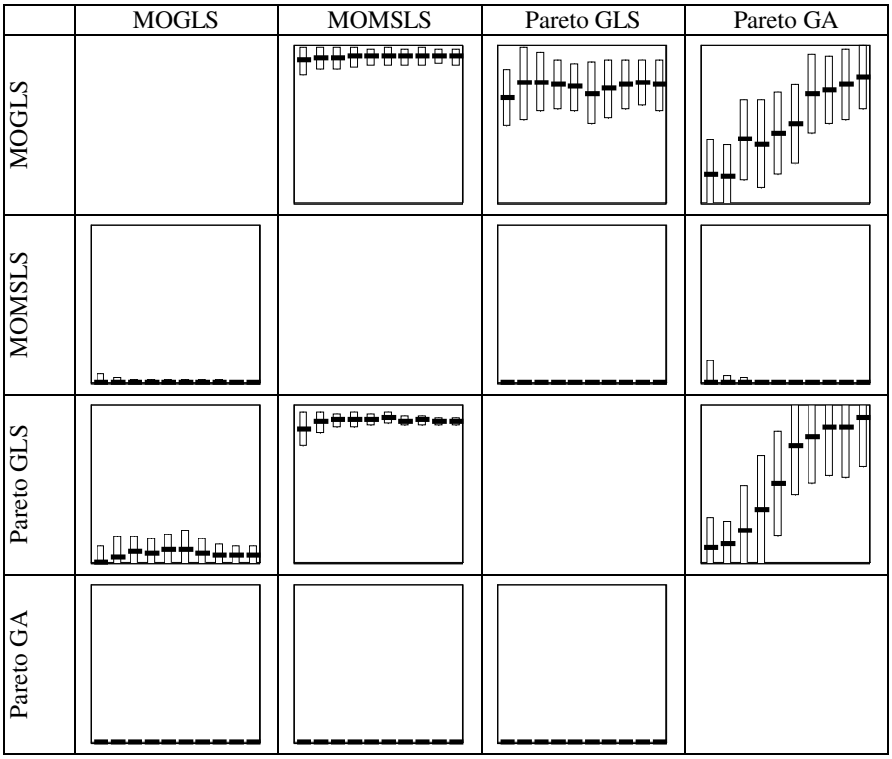

Fig. 4. Results of the comparison with coverage measure. Each chart contains 10 box plots representing the distribution of *C* values for a certain ordered pair of algorithms. The 10 box plots correspond to the ten sets of potentially Pareto-optimal solutions saved by each method in each run (the first 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.

Fig. 4 presents results of the comparison with coverage measure. The results indicate the same ranking of the four methods as the evaluation with expected value of weighted Chebycheff scalarizing quality measure. Note, that no methods is able to produce approximations to nondominated set covering 100% of approximations generated by other methods. This indicates that no method is able to find all Pareto-optimal solutions.

7 Conclusions

Four multiple objective evolutionary algorithms have been tested on a real life combinatorial optimization problem corresponding to the distribution system design. The results of the computational experiment indicate that:

- hybridization of recombination and local search operators may boost performance of multiple objective evolutionary algorithms,
- the choice of solutions for recombination has crucial influence on the performance of genetic local search algorithm.

References

1. Ackley D. H. *A connectionist machine for genetic hillclimbing*. Kluwer Academic Press, Boston, 1987.
2. Galinier P., Hao J.-K. Hybrid evolutionary algorithms for graph coloring. Technical Report, Parc Scientifique Georges Besse, Nimes, 1999.
3. 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.
4. Goldberg D.E. Genetic algorithms for search, optimization, and machine learning. Reading , MA, Addison-Wesley, 1989.
5. Hansen P.H, Jaszkiewicz A. Evaluating quality of approximations to the non-dominated set. Technical Report, Department of Mathematical Modelling, Technical University of Denmark, IMM-REP-1998-7, 1998.
6. Ishibuchi H. Murata T. Multi-Objective Genetic Local Search Algorithm and Its Application to Flowship Scheduling. IEEE Transactions on Systems, Man and Cybernetics, 28, 3, 392-403, 1998.
7. Jaszkiewicz A. Genetic local search for multiple objective combinatorial optimization, Technical Report RA-014/98, Institute of Computing Science, Poznan University of Technology, 1998
8. Jaszkiewicz A. On the performance of multiple objective genetic local search on the 0/1 knapsack problem. A comparative experiment. Submitted to IEEE Transactions on Evolutionary Computation.
9. Knowles J.D., Corne D.W. A Comparison of Diverse Approaches to Memetic Multiobjective Combinatorial Optimization, In Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program, pages 103-108, Las Vegas, Nevada, July 2000.
10. Merz P., Freisleben B., Genetic Local Search for the TSP: New Results, In Proceedings of the 1997 IEEE International Conference on Evolutionary Computation, IEEE Press, pp. 159-164, 1997.
11. Steuer R.E. Multiple Criteria Optimization - Theory, Computation and Application, Wiley, New York, 1986.
12. Ulungu E.L., Teghem J., Fortemps Ph., Tuyttens D.. MOSA method: a tool for solving multiobjective combinatorial optimization problems. Journal of Multi-Criteria Decision Analysis, 8, 221-236, 1999.
13. Taillard É. D. Comparison of iterative searches for the quadratic assignment problem, Location science, 3, 87-105, 1995.
14. Van Veldhuizen D.A. (1999). Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations. PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, May 1999.
15. Zitzler E., Thiele L. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach, IEEE Transactions on Evolutionary Computation, Vol. 3, No. 4, pp. 257-271, November, 1999.
16. www-idss.cs.put.poznan.pl./~jaszkiewicz/MOEALib
17. www.lania.mx/~ccoello/EMOO/EMOObib.html