Two Metaheuristics for Multiobjective Stochastic Combinatorial Optimization

Walter J. Gutjahr

Dept. of Statistics and Decision Support Systems, University of Vienna

Abstract. Two general-purpose metaheuristic algorithms for solving multiobjective stochastic combinatorial optimization problems are introduced: SP-ACO (based on the Ant Colony Optimization paradigm) which combines the previously developed algorithms S-ACO and P-ACO, and SPSA, which extends Pareto Simulated Annealing to the stochastic case. Both approaches are tested on random instances of a TSP with time windows and stochastic service times.

Keywords. Ant colony optimization, combinatorial optimization, multiobjective decision analysis, simulated annealing, stochastic optimization.

1 Introduction

Recently, two branches of combinatorial optimization have undergone a particularly dynamic evolution due to a strong application pull on the one hand, a technology push triggered by increased computer power on the other hand. The one of these branches is *multiobjective* combinatorial optimization (MOCO) pursuing the aim to support decision makers in the choice among a finite, but large number of alternatives without imposing the necessity of an *a priori* assignment of weights to different objectives (see, e.g., Ehrgott and Gandibleux [6]). Especially the solution of MOCO problems by *metaheuristics* has recently attracted the attention of many researchers [8]. The other of the mentioned branches is *stochastic* combinatorial optimization (SCO), which promises to support decision making under a type of uncertainty that can be represented by some suitable stochastic model. In many cases, simulation is used as an auxiliary tool for solving SCO problems. Efficient methods of diverse kind have been developed in the past for SCO and simulation-supported optimization (for a recent survey, see Fu [7]). Also in this area, metaheuristic techniques are gaining importance.

Interestingly enough, despite the large body of literature in both the MOCO and the SCO area, there are only few papers combining both features, although it would be desirable to be able to cope with problems that incorporate both multiple objectives *and* uncertainty. The scarcity of literature in this intersection has also been noted in [7] and in [1]. *Very* few articles deal with problems of this combined type by *metaheuristic* techniques. Baesler and Sepulveda [1] report on a multiobjective simulation-optimization problem in layout and scheduling, which they solve by a modification of a genetic algorithm (GA); their approach relies

on goal programming. Also Hughes [11] uses a GA approach, but he refers to the paradigm of finding nondominated (Pareto-optimal) solutions. He adapts ranking procedures applied in standard multiobjective GAs to the situation where function evaluations are subject to random noise.

The aim of this paper is to present two general-purpose metaheuristic solution algorithms SP-ACO and SPSA, determining approximations to the Pareto-optimal set for instances from a large class of MOSCO (multiobjective stochastic combinatorial optimization) problems. In Section 2, the considered type of MOSCO problems is defined. Section 3 presents the new algorithms SP-ACO and SPSA. The first, SP-ACO, is based on the ant colony optimization (ACO) paradigm (see [5]), whereas the second, SPSA, an extension of the PSA algorithm by Czyzak and Jaszkiewicz [2], draws on the well-known simulated annealing metaheuristic. In Section 4, we outline the application of both approaches to randomly generated instances of a bi-objective stochastic travelling salesperson problem with time windows and stochastic service times, and report on the obtained experimental results. Section 5 contains conclusions.

2 MOSCO Problem Formulation and Objective Function Estimation

Both approaches are designed for the heuristic solution of MOSCO problems of the following very general form:

$$\text{Minimize } (F_1(x), \ldots, F_R(x)) \quad \text{subject to} \quad x \in S \tag{1}$$

with $F_r(x) = \mathrm{E}\,(f_r(x,\omega)) \quad (r = 1, \ldots, R)$. Therein, x is the decision variable, f_r is the r-th cost function, R is the number of objectives (cost functions), ω denotes the influence of randomness, E denotes the mathematical expectation, and S is a finite set of feasible decisions.

A solution $x' \in S$ *dominates* a solution $x \in S$, if $F_r(x') \leq F_r(x)$ for all $r = 1, \ldots, R$, and if there is at least an index r such that $F_r(x') < F_r(x)$. A solution $x \in S$ is called dominated resp. nondominated by a set $S' \subseteq S$ of solutions, if there is an $x' \in S'$ such that x' dominates x, resp. if there is no such $x' \in S'$. A solution x is called *Pareto-optimal* if it is nondominated by S. The *Pareto-optimal set* is the set of Pareto-optimal solutions. As an *exact* solution of (1), we consider the Pareto-optimal set defined by the MOCO problem (1). Since the proposed algorithms are heuristics, it cannot be expected that they will in general produce the Pareto-optimal set. They output *approximations* to this set. Concerning quality evaluation of these approximations, we refer the reader to Section 4.

In our context, it is not necessary that $\mathrm{E}\,(f_r(x,\omega))$ can be computed numerically. Instead, *sampling* is used for estimating this quantity: For this purpose, draw N random *scenarios* $\omega_1, \ldots, \omega_N$ independently from each other. A *sample estimate* of $F_r(x) = \mathrm{E}\,(f_r(x,\omega))$ is given by

$$\mathcal{E}F_r(x) \;=\; \frac{1}{N} \sum_{\nu=1}^{N} f_r(x,\omega_\nu) \;\approx \mathrm{E}\,(f_r(x,\omega)). \tag{2}$$

3 Algorithms

3.1 The SP-ACO Algorithm

There exits several articles extending the ACO metaheuristic to multiobjective *or* to stochastic problems; see Dorigo and Stützle [5] for a survey. For our combined approach, we rely on the following formerly developed basic techniques: In [9], [10], an algorithm S-ACO for the heuristic solution of *single-objective* stochastic combinatorial problems has been proposed. The algorithm SP-ACO (Stochastic Pareto Ant Colony Optimization) presented here is an extension of S-ACO to the multiobjective case, combining it with the P-ACO algorithm developed for MOCO problems in [3], [4]. S-ACO and SP-ACO work based on the encoding of a given problem instance as a *construction graph* $\mathcal{C}$, a directed graph with a distinguished start node. The stepwise construction of a solution is represented by a self-avoiding random walk in $\mathcal{C}$, beginning in the start node. There may be additional rules defining particular nodes as infeasible after a certain partial walk has been traversed. When there is no feasible unvisited successor node anymore, the walk stops and is decoded as a complete solution for the problem. The conceptual unit performing such a walk is called an *ant*.

The encoding must assign exactly one feasible solution to each feasible walk. Vice versa, to each feasible solution *at least* one feasible walk (possibly more) must correspond. Given that the indicated condition is satisfied, we may consider a walk as a solution, denote it again by the symbol x and consider S as the set of feasible walks.

The probability p_{kl} that an ant goes from a node k to a feasible successor node l is chosen as proportional to $\tau_{kl} \cdot \eta_{kl}(u)$, where τ_{kl} is the so-called *pheromone value*, a memory value storing how good step (k, l) has been in previous runs, and $\eta_{kl}(u)$ is the so-called *visibility*, a pre-evaluation of how good step (k, l) will presumably be, based on some problem-specific heuristic. $\eta_{kl}(u)$ is allowed to depend on the partial walk u performed so far. In the experimental investigations in this paper, we did not use nontrivial visibility values, setting $\eta_{kl}(u) = 1$ in each case. For this reason, the role of the visibility (which can improve solution quality) will not be discussed here. For details, we refer the reader to [5].

Whether a continuation (k, l) of a partial walk u ending with node k is feasible or not is defined in accordance with the condition above that node l is not yet contained in u, and that none of the (eventual) additional rules specifies l as infeasible after u has been traversed.

In a loop, a predefined number Γ of random walks of ants according to the procedure above are performed sequentially. These Γ walks form together a *round* of the process. The single-objective heuristic S-ACO determines in each round a *round-winner*. This is done by comparing all walks that have been performed in this round on one random scenario ω, drawn specifically for this round. In the multiobjective context of SP-ACO, the determination of a round winner necessitates that a *unique* objective function for ranking the solutions produced by the walks of the ants is defined for this round. We do this by taking a *weighted average* of the cost functions $f_1, \ldots, f_R$. The weights $w_1, \ldots, w_R$ are drawn ran-

domly at the beginning of a process phase called *period*. A period contains several rounds in which solutions are gradually improved w.r.t. the current weights. In the next period, new weights are drawn; this process is iterated.

Procedure SP-ACO

$\tau_{kl}^{(r)} := 1$ for all (k,l) and for all $r = 1, \ldots, R$;
initialize the solution set X as the empty set;
for period $\pi = 1$ **to** Π {
  draw weights $w_1, \ldots, w_R$ randomly;
  $\tau := \sum_{r=1}^{R} w_r \tau^{(r)}$;
  for round $m = 1$ **to** M {
    for ant $\gamma = 1, \ldots, \Gamma$ {
      set position k equal to start node of $\mathcal{C}$;
      set u equal to the empty list;
      while (a feasible continuation (k,l) of u exists) {
        select successor node l with probability

$$p_{kl} = \begin{cases} 0, \text{ if } (k,l) \text{ is infeasible,} \\ \tau_{kl} \cdot \eta_{kl}(u) \,/\, \left(\sum_{(k,r)} \tau_{kr} \cdot \eta_{kr}(u) \right), \text{ else,} \end{cases}$$

        the sum being over all feasible (k,r);
        set $k := l$, and append l to u; }
      $x_\gamma := u$; }
    based on one random scenario ω and objective function
      $f(x,\omega) = \sum_{r=1}^{R} w_r f_r(x,\omega)$,
    select the best walk x out of $x_1, \ldots, x_\Gamma$;
    if $(m = 1)$ set $\hat{x} := x$; // candidate for best solution in period
    else {
      based on random scenarios $\omega_1, \ldots, \omega_{N_m}$, compute sample estimate
        $\mathcal{E}(F(x) - F(\hat{x})) = \frac{1}{N_m} \sum_{\nu=1}^{N_m} \sum_{r=1}^{R} w_r (f_r(x,\omega_\nu) - f_r(\hat{x},\omega_\nu))$;
      if $(\mathcal{E}(F(x) - F(\hat{x})) < 0)$ set $\hat{x} := x$; }
    evaporation: $\tau^{(r)} := (1-\rho)\, \tau^{(r)}$ for all r;
    global-best reinforcement: $\tau_{kl}^{(r)} := \tau_{kl}^{(r)} + c_1 w_r$ for all $(k,l) \in \hat{x}$ and all r;
    round-best reinforcement: $\tau_{kl}^{(r)} := \tau_{kl}^{(r)} + c_2 w_r$ for all $(k,l) \in x$ and all r;
    $\tau := \sum_{r=1}^{R} w_r \tau^{(r)}$;
    based on a sample of size $N^{(c)}$, evaluate estimates $\mathcal{E}F_1(\hat{x}), \ldots, \mathcal{E}F_r(\hat{x})$;
    if ($\hat{x}$ nondominated by X according to computed estimates of size $N^{(c)}$)
      add $\hat{x}$ to X and remove dominated elements from X; } }

Fig. 1. Pseudocode SP-ACO.

In an ACO implementation for a deterministic problem, it is customary to store the best solution seen so far in a special variable. A crucial difference to the deterministic case is that in the stochastic context, it is not possible anymore to decide with certainty whether a current solution x is better than the solution currently considered as the best found, $\hat{x}$, or not. To make a tentative decision by sampling, we perform a *tournament*: After a current round-winner x has

been determined, x is compared with the solution considered as the overall best solution so far in this period, $\hat{x}$. For evaluating the solutions, a weighted average F of the objective functions $F_1, \ldots, F_R$ with the current weights $w_1, \ldots, w_R$ is used, and estimates for the values of the functions F_r are determined from a sample consisting of N_m randomly drawn scenarios ω_ν which are used by both solutions. Also these scenarios are round-specific, i.e., in the next round, new scenarios will be drawn. The larger N_m, the more reliable is the decision. The winner of the comparison is stored as the new "global-best" $\hat{x}$. In [9] it is shown that the sample size N_m should be increased as a linear function of the round number m to enable a convergence result for the single-objective case.

Next, the solution $\hat{x}$ considered so far as the best of the current period as well as the current round-winner are reinforced on each of their arcs by pheromone increments, after a certain fraction ρ ("evaporation factor") of pheromone has been removed from each arc. The parameters $c_1 > 0$ and $c_2 > 0$ in the algorithm determine the amount of pheromone increment on global-best and round-best walks, respectively.

We take account of the different objective functions $F_1, \ldots, F_R$ by assigning a *separate* pheromone matrix $\tau^{(r)} = (\tau_{kl}^{(r)})$ to each objective r. Global-best and round-best reinforcement is done in each of these pheromone matrices with the current respective weight w_r. For the transition from a node k to a feasible successor node l, the guiding pheromone values τ_{kl} must be computed as a weighted mean of the objective-specific pheromone values $\tau_{kl}^{(r)}$. As weights we take again the current values w_r.

After reinforcement, it is checked whether the current global winner solution $\hat{x}$ can be added to the current set X of candidates for the approximation to the Pareto-optimal set. For this purpose, an estimation of the objective function values $F_1(\hat{x}), \ldots, F_R(\hat{x})$ based on a random sample of constant size $N^{(c)}$, where $N^{(c)}$ is comparably large, is performed. If $\hat{x}$ turns out as nondominated by X according to these estimates, $\hat{x}$ is added to X, and solutions in X dominated by $\hat{x}$ are removed from X. The objective function estimates obtained from the sample of size $N^{(c)}$ are assigned to $\hat{x}$ in the list of the elements of X for future dominance comparisons with new candidates for the solution set X.

3.2 The SPSA Algorithm

In this subsection, we present an extension of the PSA (Pareto Simulated Annealing) algorithm by Czyzak and Jaszkiewicz [2], designed for solving MOCO problems, to an algorithm SPSA (Stochastic Pareto Simulated Annealing) for the solution of MOSCO problems. Since PSA is already a well-established technique, we keep the description short, focusing on the points by which SPSA extends PSA. The pseudocode of SPSA is given in Fig. 2.

PSA uses a search set (here denoted by Θ) exploring the solution space governed by a mechanism that (i) drives the search points towards the Pareto-optimal set, and (ii) favors diversification by forcing points that lie close to each other in the solution space to "specialize" on different objectives. The last effect

is achieved by a suitable modification of the weights assigned to the objective functions: the weights are increased for those objectives for which a current search point x is better than a near-by search point x', and decreased for the others.

Procedure SPSA

initialize the search set Θ by s random feasible solutions;
initialize the solution set X as the empty set;
initialize $N^{(2)}$ by N_{init};
for $i = 1$ **to** s
  if (ith solution x_i in Θ is nondominated by X, based on sample size $N^{(1)}$)
    add x_i to X and remove dominated elements from X;
initialize temperature parameter T;
repeat until (termination criterion is met) {
  for $l = 1$ **to** L {
    for $i = 1$ **to** s {
      for $r = 1$ **to** R
        compute sample estimate $\mathcal{E}F_r(x_i)$ based on sample size $N^{(3)}$;
      construct a random feasible neighbor solution y to x_i;
      select $x' \in \Theta$ nondominated by x_i with minimum distance to x_i;
      if (first run or x' not found) {
        for $r = 1$ **to** R
          draw random weight w_{ir};
        normalize weights w_{ir} to $\sum_r w_{ir} = 1$; }
      else {
        for $r = 1$ **to** R {
          compute sample estimate $\mathcal{E}F_r(x')$ based on sample size $N^{(3)}$;
          if ($\mathcal{E}F_r(x_i) < \mathcal{E}F_r(x')$) $w_{ir} := aw_{ir}$; **else** $w_{ir} := w_{ir}/a$;
          normalize weights w_{ir} to $\sum_r w_{ir} = 1$; } }
      for $r = 1$ **to** R
        compute sample estimate $\mathcal{E}(F_r(x_i) - F_r(y))$ based on sample size $N^{(2)}$;
        with probability $\min(1, \exp(\sum_r w_{ir}\mathcal{E}(F_r(x_i) - F_r(y))/T)$ {
          $x_i := y$;
          if ($T < T_c$ and y nondominated by X, based on sample of size $N^{(1)}$)
            add y to X and remove dominated elements from X; } } }
  $T := bT$;
  increase $N^{(2)}$ by N_{inc}; }

Fig. 2. Pseudocode SPSA.

Basically, our extension SPSA works as PSA, with the exception that at any time when an objective function evaluation is necessary, an estimation based on sampling is done. For the sample estimate, we use the notation of (2). Three sample size parameters $N^{(1)}$, $N^{(2)}$ and $N^{(3)}$ are used. $N^{(1)}$ and $N^{(3)}$ are constants, $N^{(2)}$ is a variable. $N^{(1)}$ corresponds to the sample size $N^{(c)}$ in the SP-ACO algorithm in that it is applied for deciding whether a candidate solution is (presumably) nondominated by the current elements of the solution set X. As $N^{(c)}$

in the case of SP-ACO, $N^{(1)}$ must be high, because the estimates are not revised anymore at a later time. In the experiments, it turned out that better results were achieved by considering insertion into X only after the temperature parameter T has fallen below some threshold T_c. Sample size $N^{(2)}$ is used for deciding whether or not a neighbor solution is to be accepted. Since by the simulated annealing philosophy, in a late phase (low temparature), neighbor solutions that are not better than the current solution should be rejected with a high probability, it is important that the estimation accuracy for the difference of the objective function values is gradually increased during the process, similarly as we gradually increase N_m in the SP-ACO algorithm. The third sample size, $N^{(3)}$, is used for getting estimates of the objective function values of a current solution $x \in \Theta$ compared to those of the closest neighbor x' of x in Θ.

4 Experimental Results on a TSPTW-SST

For first computational experiments with the described algorithms, we used a bi-objective TSP with Time Windows and Stochastic Service Times (TSPTW-SST). A set of customers $\{1, \ldots, n\}$ and a distance matrix $D = (d_{ij})$ are given. Distances are interpreted as driving times. Let us imagine that the travelling person is a service engineer. To each customer i, a time window $[a_i, b_i]$ can be assigned, indicating that customer i requests a visit by the service engineer starting at time t_i with $a_i \leq t_i \leq b_i$. If the service engineer arrives at customer i at a time t_i before time a_i, (s)he must wait until time a_i. If (s)he arrives after time b_i, a *tardiness* of amount $t_i - b_i$ is registered. Not every customer needs to have a time window for the visit. The service at customer i takes some time Y_i, where Y_i is a random variable with known distribution. After finishing the service at customer i, the service engineer drives to the next customer on the list given by the chosen permutation x of customers. The aim is to minimize the exepected values of two objectives: (i) the sum of total driving time and total waiting time, (ii) the sum of the tardiness values $(t_i - b_i)^+$.

Besides SP-ACO and SPSA, we also implemented a complete enumeration procedure combined with brute force simulation (CE/BFS, sample size 10^6) to evaluate proposed solutions, and a random search (RS) procedure with constant sample size per solution.

We generated 12 different problem instances at random. The problem size was chosen as $n = 9$ in all these instances, which was the largest number of customers for which we were able to compute the Pareto-optimal set by the CE/BFS approach within reasonable time. (On a PC Pentium 2.4 GHz, this took about 5 hours per test instance; the sample size was chosen as high as 10^6 to reach a sufficient accuracy of the objective function evaluations.) Test instances with $n = 9$ may seem as very small, but we would like to emphasize that in the combined setting of two different objectives combined with simulation-based objective function determination, the problem is highly nontrivial already for this instance size.

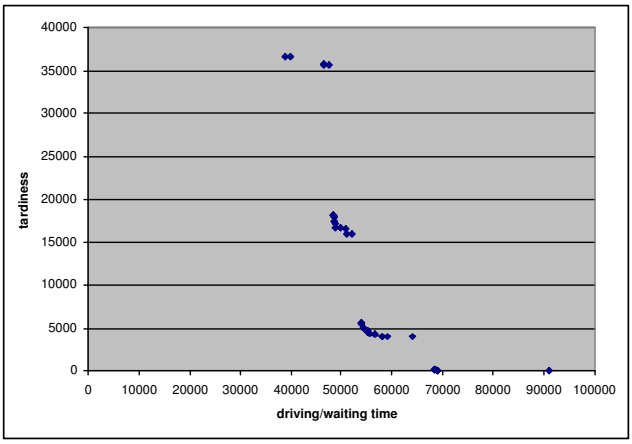

Fig. 3. Pareto front for the first instance of $p_{TW} = 0.2$, $[\sigma_{min}, \sigma_{max}] = [0, 40]$, time units multiplied by 1000.

In the case of each problem instance, $n = 9$ customer points were selected uniformly at random from a square. Distances were computed as Euclidean distances between these points. It was assumed that traversing an edge of the square takes 10 time units. For each customer, a random decision was made on wether or not to assign a time window, using a fixed probability p_{TW} for the existence of a time window. If a time window was assigned, its length was selected uniformly at random between 6 and 60 time units, and its start time was selected uniformly at random between time 0 and the maximum time such that the whole time window was still contained in an interval of 120 time units. The service time distributions were chosen as uniform distributions on the interval between two values σ_{min} and σ_{max}.

Experiments were carried out for the following combinations of parameter values: For p_{TW}, we considered the cases $p_{TW} = 0.2$ and $p_{TW} = 0.3$, and for the intervals $[\sigma_{min}, \sigma_{max}]$, we chose the cases $[0, 20]$, $[0, 40]$ and $[10, 30]$. For each of the 6 parameter combinations, 2 random test instances were generated, such that 12 test instances in total were produced. Fig. 3 shows a typical Pareto front (Pareto-optimal set in objective space).

To have a fair comparison, each of the heuristic algorithms SP-ACO, SPSA and RS was given 20 seconds on a PC Pentium 2.4 GHz for a single run. The parameters of each heuristic were tuned to best possible results within these 20 seconds at the first test instance for the combination $p_{TW} = 0.2$, $[\sigma_{min}, \sigma_{max}] = [0, 40]$. After that, 100 runs for each of the three heuristics were performed on each test instance and evaluated with the help of the results obtained by CE/BFS.

The tuning yielded the following parameter values: (a) SP-ACO: $M = 10$, $\Gamma = 200$, $\rho = 0.00005$, $N^{(c)} = 2000$, $c_1 = 0.0005$, $c_2 = 0.000005$. (b) SPSA: $s = 10$, $N^{(1)} = 500$, $N^{(2)}$: initial value 1, increment 1, $N^{(3)} = 1$, $a = 1.1$, $b = 0.9$. A neighbor solution y was determined by m_n random 2-opt moves, where m_n was chosen uniformly between 1 and 4. (c) RS: sample size $N_{rs} = 200$ per

randomly generated solution. Concerning the SP-ACO parameters, we remark that the value ρ may seem rather low compared to usual ACO implementations for single-objective deterministic problems. However, it should be noted that pheromone is not re-initialized at the beginning of each period, such that also low values of ρ effect distinct differences in the pheromone values towards the end of the procedure. Moreover, it can be observed that in the parameter choice above, the global pheromone increment c_1 is 100 times as high as the local pheromone increment c_2, which turned out as advantageous.

In the literature, many measures have been described for evaluating the results of multiobjective optimization heuristics (see, e.g., Jaszkiewicz [12]). Since our present results are only intended as a first experimental test, we used only one, rather basic evaluation metric, namely the ratio of Pareto-optimal solutions, as determined by CE/BFS, that are covered by one of the solutions in the output set of a considered heuristic algorithm. This is the first evaluation metric Q_1 for MOCO heuristics suggested in [12]; it goes back to Ulugu et al. [13].

In Table 1, we list the average Q_1 value over 100 test runs for each problem instance and each heuristic. In addition, for each problem instance and each pair of heuristics, we performed a (two-sided) Wilcoxon test to decide whether or not the difference between the indicated average Q_1 values for the two heuristics is statistically significant or not. Column 2 of Table 1 shows the total number of Pareto-optimal solutions, columns 3 – 5 the average Q_1 values over 100 runs, and columns 6 – 8 the results of the significance tests for the pairwise comparisons (n: no significance; s, s^* and s^{**}: significance at level $\alpha = 0.05$, 0.01 and 0.001, respectively). If a significant difference was found, the heuristic with the better results in the pairwise comparison was named. As it can be seen, both SP-ACO and SPSA outperformed RS in all cases with high significance. SP-ACO outperformed SPSA in 6 cases with high significance and was outperformed by SPSA in 3 cases, also with high significance. In the 3 remaining cases, there was no statistically significant difference between the results of SP-ACO and SPSA.

5 Conclusions

Two metaheuristics, SP-ACO and SPSA, for solving MOSCO problems have been developed and compared on a TSPTW with stochastic service times. The results do not indicate a consistent superiority of SP-ACO over SPSA or vice versa in the evaluation metric Q_1, but a random search approach is clearly outperformed by both. Future work should deal with more comprehensive outcome evaluations on extended test instance sets, and Q_1 should be supplemented by other metrics. Furthermore, extensions of other MOCO metaheuristics to the stochastic case should be included into the comparison.

References

1. Baesler, F.F., Sepúlveda, J.A., "Multi-objective simulation optimization for a cancer treatment center", *Proc. WSC 2001*, pp. 1405-1411 (2001).

		average ratio of found sol.			significantly better		
test instance	no. sol.	aco	sa	rs	aco : sa	aco : rs	sa : rs
0.2, [0,20] / 1	22	0.337	0.327	0.184	n	aco (s^{**})	sa (s^{**})
0.2, [0,20] / 2	17	0.663	0.571	0.444	aco (s^{**})	aco (s^{**})	sa (s^{**})
0.2, [0,40] / 1	41	0.421	0.285	0.218	aco (s^{**})	aco (s^{**})	sa (s^{**})
0.2, [0,40] / 2	34	0.450	0.450	0.343	n	aco (s^{**})	sa (s^{**})
0.2, [10,30]/ 1	25	0.428	0.305	0.222	aco (s^{**})	aco (s^{**})	sa (s^{**})
0.2, [10,30]/ 2	16	0.727	0.720	0.511	n	aco (s^{**})	sa (s^{**})
0.3, [0,20] / 1	21	0.701	0.561	0.366	aco (s^{**})	aco (s^{**})	sa (s^{**})
0.3, [0,20] / 2	19	0.748	0.652	0.448	aco (s^{**})	aco (s^{**})	sa (s^{**})
0.3, [0,40] / 1	23	0.620	0.662	0.515	sa (s^{**})	aco (s^{**})	sa (s^{**})
0.3, [0,40] / 2	27	0.432	0.480	0.329	sa (s^{**})	aco (s^{**})	sa (s^{**})
0.3, [10,30]/ 1	20	0.532	0.582	0.459	sa (s^{**})	aco (s^{*})	sa (s^{**})
0.3, [10,30]/ 2	11	0.550	0.451	0.338	aco (s^{**})	aco (s^{**})	sa (s^{**})

Table 1. Results for the 12 test instances (including 100 runs per instance and method).

2. Czyzak, P., Jaszkiewicz, A., "Pareto simulated annealing — a metaheuristic technique for multiple-objective combinatorial optimization", *J. of Multi-Criteria Decision Analysis* 7, pp. 34-47 (1998).
3. Doerner, K., Gutjahr, W.J., Hartl, R.F., Strauss, C., Stummer, C., "Ant Colony Optimization in Multiobjective Portfolio Selection", *Proc. 4th Metaheuristics International Conference*, pp. 243-248 (2001).
4. Doerner, K., Gutjahr, W.J., Hartl, R.F., Strauss, C., Stummer, C., "Pareto Ant Colony Optimization: A metaheuristic approach to multiobjective portfolio selection", *Annals of Operations Research* 131, pp. 79-99 (2004).
5. Dorigo, M., Stützle, T., *Ant Colony Optimization*, MIT Press (2004).
6. Ehrgott, M., Gandibleux, X., "A Survey and Annotated Bibliography of Multiobjective Combinatorial Optimization", *OR Spektrum* 22, pp. 425-460 (2000).
7. Fu, M.C., "Optimization for simulation: theory vs. practice", *INFORMS J. on Computing* 14, pp. 192-215 (2002).
8. Gandibleu, X., Sevaux, M., Sörensen, K., T'kindt, V. (Eds.), *Metaheuristics for Multiobjective Optimization*, Springer, Berlin-Heidelberg (2004).
9. Gutjahr, W.J., "A converging ACO algorithm for stochastic combinatorial optimization", *Proc. SAGA 2003* (Stochastic Algorithms: Foundations and Applications), Springer LNCS 2827, pp. 10-25 (2003).
10. Gutjahr, W.J., "S-ACO: An ant-based approach to combinatorial optimization under uncertainty", *Proc. ANTS 2004* (4th International Workshop on Ant Colony Optimization and Swarm Intelligence), Springer LNCS 3172, pp. 238-249 (2004).
11. Hughes, E.J., "Evolutionary Multi-objective Ranking with Uncertainty and Noise", in: E. Zitzler, K. Deb, L. Thiele, C.A. Coello Coello, and D. Corne (eds.), *First International Conference on Evolutionary Multi-Criterion Optimization*, pp. 329-343, Springer LNCS No. 1993 (2001).
12. Jaszkiewicz, A., "Evaluation of multiple objective metaheuristics", in: Gandibleu, X., Sevaux, M., Sörensen, K., T'kindt, V. (Eds.), *Metaheuristics for Multiobjective Optimization*, Springer, Berlin-Heidelberg, pp. 65-89 (2004).
13. Ulugu, E.L., Teghem, J., Fortemps, Ph., Tuyttens, D., "MOSA method: a tool for solving multiobjective combinatorial optimization problems", *J. of Multi-Criteria Decision Analysis* 8, pp. 221-236 (1999).