# Bi-Criterion Optimization with Multi Colony Ant Algorithms

Steffen Iredi, Daniel Merkle, and Martin Middendorf*

Institute AIFB,
University of Karlsruhe,
D-76128 Karlsruhe, Germany
iredi@informatik.uni-hannover.de
{merkle,middendorf}@aifb.uni-karlsruhe.de

**Abstract.** In this paper we propose a new approach to solve bi-criterion optimization problems with ant algorithms where several colonies of ants cooperate in finding good solutions. We introduce two methods for cooperation between the colonies and compare them with a multistart ant algorithm that corresponds to the case of no cooperation. Heterogeneous colonies are used in the algorithm, i.e. the ants differ in their preferences between the two criteria. Every colony uses two pheromone matrices — each suitable for one optimization criterion. As a test problem we use the Single Machine Total Tardiness problem with changeover costs.

## 1 Introduction

Ant Colony Optimization (ACO) is an evolutionary approach that has been applied successfully to solve various combinatorial optimization problems (for an overview see Dorigo and Di Caro [4]). In ACO ants that found a good solution mark their paths through the decision space by putting some amount of pheromone along the path. The following ants of the next generation are attracted by the pheromone so that they will search in the solution space near good solutions.

Much work has been done to apply evolutionary methods to solve multi-criterion optimization problems (see [13] for an overview). But only a few of this works used the ACO principle. Mariano and Morales [9] proposed an ant algorithm where for each objective there exists one colony of ants. In particular, they studied problems where every objective is influenced only by parts of a solution, i.e. an objective can be determined knowing only the relevant part of a solution. The objectives are assumed to be ordered by importance. In every generation ant $k$ from colony $i$ receives a (partial) solution from ant $k$ of colony $i-1$. The ant then tries to improve (or extend) the (partial) solution with respect to criterion $i$. When the solutions have passed through all colonies those solutions that are in

* Corresponding author. Part of this work was done while the author stayed at the Institute of Computer Science at the University of Hannover.

E. Zitzler et al. (Eds.): EMO 2001, LNCS 1993, pp. 359–372, 2001.

the nondominated front are allowed to update the pheromone information. Gambardella et al. [7] developed an ant algorithm for a bi-criterion vehicle routing problem. They used two ant colonies — one for each criterion. The two colonies share a common global best solution which is used for pheromone update in both colonies. Criterion 1 — the number of vehicles — is considered to be more important than criterion 2 — the total travel time of the tours. Colony 1 tries to find a solution with one vehicle less than the global best solution while colony 2 tries to improve the global best solution with respect to criterion 2. Whenever colony 1 finds a new global best solution both colonies start anew (with the new global best solution). Gagné et al. [6] tested a multi-criterion approach for solving a single machine total tardiness problem with changeover costs and two additional criteria. In their approach the changeover costs were considered to be most important. The idea was to construct heuristic values for the decisions of the ants that take all criteria into account. The amount of pheromone that an ant adds to the pheromone matrix depends solely to changeover costs of the solution. All the above mentioned approaches assume that the different criteria can be ordered by importance and in the multi colony approaches there is always one colony for every objective.

In this paper we study ACO methods for multi-criterion optimization when the objectives can not be ordered by importance. The aim is to find different solutions which cover the Pareto-optimal front. A multi colony approach is proposed where the ant colonies are forced to search in different regions of the nondominated front. It should be noted, that multi colony ant algorithms have been studied before by some authors to parallelize ACO algorithms (a short overview is given in [12]). We use heterogeneous colonies where the ants in a colony weight the relative importance of the two optimization criteria differently so that they are able to find different solutions along the Pareto front. Cooperation between the colonies is done by exchanging solutions in the global nondominated front that are in regions which "belong to other colonies".

Our test problem, the Single Machine Total Tardiness Problem (SMTTP) with changeover costs is described in section 2. A short introduction to ant algorithms for solving the single objective versions of our test problem are given in Section 3. Our ACO approaches for bi-criteria optimization problems are described in Section 4. The multi colony approaches are explained in Section 5. The tests instances and parameters are described in Section 6. The Results are discussed in Section 7 and conclusions are given in Section 8.

## 2 The Test Problem

In this paper we use the Single Machine Total Tardiness Problem (SMTTP) with changeover costs as our bi-criterion test problem. The problem is defined as follows.

- Given: $n$ jobs, where job $j$, $1 \leq j \leq n$ has a processing time $p_j$ and a due date $d_j$ and where for every pair of jobs $i, j$, $i \neq j$ there are changeover costs $c(i, j)$ that have to be paid when $j$ is the direct successor of $i$ in a schedule.

– Find: A non-preemptive one machine schedule that minimizes the value of $T = \sum_{j=1}^{n} \max\{0, C_j - d_j\}$ where $C_j$ is the completion time of job $j$ and that also minimizes the sum of the changeover costs $C = \sum_{i=1}^{n-1} c(j_i, j_{i+1})$ where $j_1, j_2, \ldots, j_n$ is the sequence of jobs in the schedule.

$T$ is called the total tardiness of the schedule and $C$ is the cost of the schedule. It is known that SMTTP is NP-hard in the weak sense [5] and is solvable in pseudo-polynomial time [8]. The problem to find a schedule that minimizes only the changeover costs is equivalent to the asymmetric Shortest Hamiltonian Path problem which is NP-complete in the strong sense.

## 3 Ant Algorithms for Single-Criteria Optimization Problems

### 3.1 Total Tardiness Minimization

A variant of the ACO algorithm of Merkle and Middendorf [10] for the SMTTP is described in this section (other ACO approaches for SMTTP can be found in [1,3]). In every generation each of $m$ ants constructs one solution. An ant selects the jobs in the order in which they will appear in the schedule. For the selection of a job the ant uses heuristic information as well as pheromone information. The heuristic information, denoted by $\eta_{ij}$, and the pheromone information, denoted by $\tau_{ij}$, are an indicator of how good it seems to have job $j$ at place $i$ of the schedule. The heuristic value is generated by some problem dependent heuristic whereas the pheromone information stems from former ants that have found good solutions.

The next job is chosen from the set $\mathcal{S}$ of jobs that have not been scheduled so far according to the probability distribution that is determined by

$$p_{ij} = \frac{[\tau_{ij}]^{\alpha} [\eta_{ij}]^{\beta}}{\sum_{h \in \mathcal{S}} [\tau_{ih}]^{\alpha} [\eta_{ih}]^{\beta}} \tag{1}$$

The heuristic values $\eta_{ij}$ are computed according the following modified due date rule

$$\eta_{ij} = \frac{1}{\max\{\mathcal{T} + p_j, d_j\} - \mathcal{T}} \tag{2}$$

where $\mathcal{T}$ is the total processing time of all jobs already scheduled. The best solution found so far is then used to update the pheromone matrix. But before the update is done some of the old pheromone is evaporated according to

$$\tau_{ij} = (1 - \rho) \cdot \tau_{ij}$$

The reason for this is that old pheromone should not have a too strong influence on the future. Then, for every job $j$ in the schedule of the best solution found so far some amount of pheromone $\Delta$ is added to element $\tau_{ij}$ of the pheromone

matrix where $i$ is the place of job $j$ in the schedule. The algorithm stops when some stopping criterion is met, e.g. a certain number of generations has been done. The pseudocode for the algorithm is given below.

**Ant Algorithm 1** Total Tardiness Minimization

```
repeat
  for ant k ∈ {1,...,m} do
    S = {1,2,...,n} {set of nonscheduled jobs}
    for i = 1 to n do
      choose job j ∈ S with probability p_ij
      S := S − {j}
    end for
  end for
  for all (i,j) do
    τ_ij ⇐ (1 − ρ) · τ_ij {evaporate pheromone}
  end for
  for all (i,j) ∈ best solution do
    τ_ij ⇐ τ_ij + Δ {update pheromone}
  end for
until stopping criterion is met
```

It has to be mentioned that we use in this paper a simplified version of the original ant algorithm for the SMTTP problem. For an example no local pheromone update is done by the ants, i.e. the ants do not change the pheromone values during their search for a solution. Moreover, we do not use local optimization. The reason for using a simplified variant is that we want to concentrate on the algorithmic aspects that are relevant for solving a bi-criterion problem with multiple colonies.

### 3.2 Changeover Cost Minimization

An ant algorithm for the single machine scheduling problem where only the changeover costs are relevant could be similar to the algorithm described above. But the pheromone information should be different. For the total tardiness value it is important on which place in the schedule a job is. Whereas, for the changeover costs it is more important which job is the predecessor of a job. Hence, a pheromone matrix is used where $\tau_{ij}$ is the desirability that job $j$ comes after job $i$ in the schedule. Moreover, an additional dummy node 0 is introduced so that $c(0, j) = 0$ for every job $j \in [1, n]$. Then $\tau_{0j}$ is the desirability to start with job $j$. As heuristic information we use $\eta_{ij} = 1/c(i, j)$ for $i, j \in [1, n]$ and $\eta_{0j} = 1$ for $j \in [1, n]$.

# 4 Ant Algorithm for Bi-Criterion Optimization Problems

In this section we describe how our ant algorithms works in case of a single colony of ants.

## 4.1 Two Pheromone Matrices

Different optimization criteria may need different pheromone information as explained the last section. Therefore we propose to use two pheromone matrices, one for each criterion. Then every ant can work differently on the two matrices.

For the SMTTP with changeover costs we use a pheromone matrix $M = (\tau_{ij})$ for the total tardiness criterion where $\tau_{ij}$ is the desirability that job $j$ is on place $i$ of the schedule. For the changeover cost criterion a pheromone matrix $M' = (\tau'_{ij})$ is used where $\tau'_{ij}$ is the desirability that job $j$ comes after job $i$ in the schedule.

## 4.2 Heterogeneous Colony

To force the ants to search in different regions of the Pareto front each of the $m$ ants in the colony weights the relative importance of the two optimization criteria differently when making its decisions. More exactly, ant $k$, $k \in [1, m]$ in the colony uses $\lambda_k = \frac{k-1}{m-1}$. Every ant makes its decision according to the following probabilities:

$$p_{ij} = \frac{\tau_{ij}^{\lambda\alpha} \cdot \tau_{ij}'^{(1-\lambda)\alpha} \cdot \eta_{ij}^{\lambda\beta} \cdot \eta_{ij}'^{(1-\lambda)\beta}}{\sum_{h \in \mathcal{S}} \tau_{ih}^{\lambda\alpha} \cdot \tau_{ih}'^{(1-\lambda)\alpha} \cdot \eta_{ih}^{\lambda\beta} \cdot \eta_{ih}'^{(1-\lambda)\beta}} \tag{3}$$

where $\eta_{ij}$ is the heuristic information that corresponds to the tardiness criterion (cmp. Equation 2) and $\eta'_{ij}$ is the heuristic information that corresponds to the changeover criterion (cmp. Subsection 3.2). Thus, in the extreme cases the ant $m$ with $\lambda = 1$ considers only the first criterion whereas ant 1 with $\lambda = 0$ considers only the second criterion.

Merkle and Middendorf [10] proposed an alternative method for pheromone evaluation where the pheromone values corresponding to older decisions are taken into account. Instead of using $\tau_{ij}$ in formula 1 they used $\sum_{k=1}^{i} \tau_{kj}$. This so called summation evaluation method was successfully applied to several scheduling problems. For the weighted version of SMTTP a combination of standard evaluation and a weighted version of summation evaluation has been shown to be very successful in [11]. Formally, this combined version uses

$$\tau_{ij}^{*} := c \cdot x_i \cdot \tau_{ij} + (1 - c) \cdot y_i \cdot \sum_{k=1}^{i} \gamma^{i-k} \tau_{kj} \tag{4}$$

where $c$ is the parameter that determines the relative influence of weighted summation evaluation, $\gamma$ is the parameter that determines the relative influence of pheromone values corresponding to older decisions. Parameters $x_i :=$

$\sum_{h\in\mathcal{S}}\sum_{k=1}^{i}\gamma^{i-k}\tau_{kh}$ and $y_i := \sum_{h\in\mathcal{S}}\tau_{ih}$ are factors to adjust the relative influence of local and summation evaluation. Observe, that for $c = 1$ the standard evaluation is obtained and for $c = 0$ pure summation evaluation.

Therefore it might be advantageous to use different methods for the evaluation of the two pheromone matrices. If e.g. summation evaluation is used for the first criterion the probabilities used by the ant are

$$p_{ij} = \frac{(\tau^*_{ij})^{\lambda\alpha}\cdot\tau'^{(1-\lambda)\alpha}_{ij}\cdot\eta^{\lambda\beta}_{ij}\cdot\eta'^{(1-\lambda)\beta}_{ij}}{\sum_{h\in\mathcal{S}}(\tau^*_{ih})^{\lambda\alpha}\cdot\tau'^{(1-\lambda)\alpha}_{ih}\cdot\eta^{\lambda\beta}_{ih}\cdot\eta'^{(1-\lambda)\beta}_{ih}} \tag{5}$$

### 4.3 Pheromone Update

When all $m$ ants of a generation have found a solution it has to be decided which of the ants are allowed to update. Here we propose that all ants in the nondominated front of the actual generation are allowed to update. An ant that updates will update both pheromone matrices $M$ and $M'$. Note, that this rule makes sense only when there are not too few ants in a colony. With only very few ants in a colony the ants differ much by their $\lambda$-values. Hence, no real competition about best solutions occurs since the ants will search in different regions of the nondominated front. Thus even the ants with weak solutions will have good changes to do an update. One way to solve this problem is to allow only those ants to update that have found solutions which are good compared to the nondominated front of all solutions that have been found so far.

To give every generation of ants the same influence the amount of pheromone that is added to a pheromone matrix is the same in every generation. Therefore, every ant is allowed to update an amount of $\tau_{ij} = \tau_{ij} + 1/l$ where $l$ is the number of ants that are allowed to update in the actual generation.

## 5 The Multi Colony Approach

In our multi colony ant algorithm several colonies of ants cooperate and specialize to find good solutions in different regions of the Pareto front. All $p$ colonies have the same number of $m/p$ ants.

### 5.1 Pheromone Update

In the single colony algorithm only ants in the nondominated front of the colony are allowed to update. Hence, a reasonable way for pheromone update in the multi colony algorithm is that only those ants update that found a solution which is in the local nondominated front of the colony. This corresponds to the case were there is no cooperation between the colonies. Therefore the results are the same as with a multistart approach where a single colony ant algorithm is run several times and the global nondominated front at the end is determined from the nondominated fronts of all runs. In the following we describe how to introduce collaboration between the colonies. For this, the ants in a generation put their

solutions in a global solution pool that is shared by all colonies. The pool is used to determine the nondominated front of all solutions in that generation. Then, only ants that found a solution which is in the global nondominated front are allowed to update. We study two different methods to determine in which colony an ant should update the pheromone matrix:

1. Method 1 – update by origin: an ant updates only in its own colony (compare Figure 1).
2. Method 2 – update by region in the nondominated front: the sequence of solutions along the nondominated front is split into $p$ parts of equal size. Ants that have found solutions in the $i$th part update in colony $i$, $i \in [1, p]$ (compare Figure 2). More formally, the solutions in the nondominated front are sorted with respect to the first criterion (it does not matter whether the list is sorted according to the first or the second criterion). Let $L$ be the sorted list. The sorted list is then split into parts $L_1, L_2, \ldots, L_p$ so that their size differs by at most one. All ants that found solutions in list $L_i$, $i \in [1, p]$ will update the pheromone matrix of colony $i$.

The first method imposes a stronger selection pressure on the ants that are allowed to update. It is not enough for an ant to have a solution in the local nondominated front of its colony. Instead, the solution must be in the global nondominated front. This method might be advantageous because other colonies help to detect which of the solutions in the local nondominated front of a colony might be weak. An interesting observation is that the update by origin method might also enforce the colonies to search in different regions of the nondominated front. It is more likely that a solution from the local nondominated front of a colony might also be in the global nondominated front when only a few solutions from other colonies are in the same region. Hence, it is more likely that an ant with solutions in less dense areas of the nondominated front will be allowed to update and thereby will influence the further search process.

The aim of method 2 is to explicitly guide the ant colonies to search in different regions of the Pareto front.

### 5.2 Heterogeneous Colonies

As in the single colony algorithm the ants in a colony use different $\lambda$-values, i.e. when making their decisions they weight the relative importance of the two optimization criteria differently. More exactly, ant $k$ in colony $i$, $i \in [1, p]$ uses $\lambda_k = \frac{k-1}{m/p-1}$, $k \in [1, m/p]$. We call this rule 1 for defining the $\lambda$-values.

An alternative could be to use different $\lambda$-values in the colonies so that $\lambda$-values of the ants in the colonies are in different subintervalls of $[0, 1]$. Thus the colonies weight the optimization criteria differently.

– Rule 2 — disjoint $\lambda$-intervalls: ant $k$, $k \in [1, m/p]$ in colony $i$ has $\lambda$-value $(i-1) \cdot \frac{m}{p} + k$.

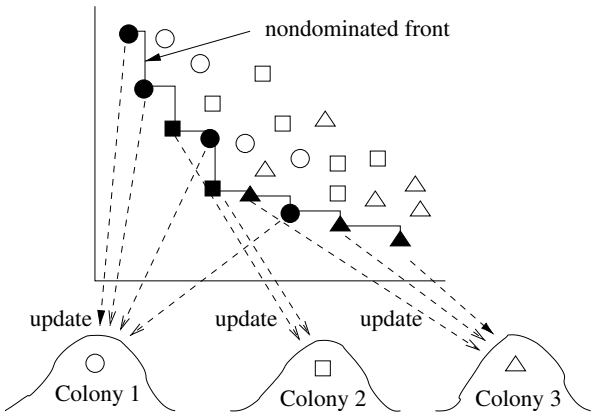


**Fig. 1.** Update by origin: Every ant with a solution in the nondominated front updates in its own colony.

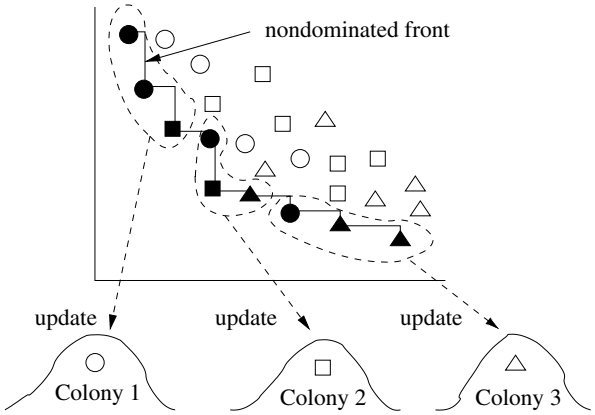


**Fig. 2.** Update by region in the nondominated front: Ants with a solution in the nondominated front update in the colony that are corresponds to the region of the solution.

– Rule 3 — overlapping $\lambda$-intervalls: the $\lambda$-intervall of colony $i$ overlaps by 50% with the $\lambda$-intervall of colony $i-1$ and colony $i+1$. Formally, colony $i$ has ants with $\lambda$-values in $[(i-1)/(p+1), (i+1)/(p+1)]$ (compare Figure 3).

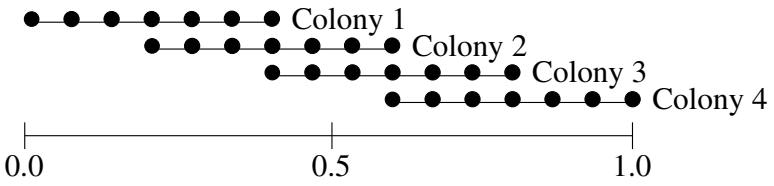


**Fig. 3.** $\lambda$-values when using rule 3: 4 colonies with 7 ants each.

## 6 Test Instances and Parameters

We tested our ant algorithms on problem instances where the jobs and their deadlines were generated after the following rule that is often used to create instances for the SMTTP [2]: for each job $j \in [1, 100]$ an integer processing time $p_j$ is taken randomly from the interval $[1, 100]$ and an integer due date $d_j$ is taken randomly from the interval

$$\left[\sum_{j=1}^{100} p_j \cdot (1 - TF - \frac{RDD}{2}), \sum_{j=1}^{100} p_j \cdot (1 - TF + \frac{RDD}{2})\right]$$

The value RDD (relative range of due dates) determines the length of the interval from which the due dates were taken. TF (tardiness factor) determines the relative position of the centre of this interval between 0 and $\sum_{j=1}^{100} p_j$. We chose the values for TF from the set $\{0.4, 0.5, 0.6\}$ and RDD was set to $RDD = 0.6$. The changeover costs between the jobs were chosen randomly from one of the sets $[1, 100]$ and $[50, 100]$.

The parameters used for the test runs are: $\alpha = 1$, $\beta = 1$, $\rho = 0.02$. Pheromone evaluation was done according to formula 5 where a combination between summation evaluation and standard evaluation for matrix 1 is used. For the corresponding parameters the values $c = 0.6$ and $\gamma = 0.9$ were used (these values were shown to be suitable for the weighted SMTTP [11]). The number of ants in every generation was $m = 100$. When using several colonies the 100 ants were distributed equally to the colonies. Every element of the pheromone matrices was initialized with 1.0. Every test was performed with 11 runs. Every run was stopped after 300 generations.

## 7 Results

The performance of the multi colony approach was tested on 6 problem instances: three instances with changeover costs in $[1, 100]$ and three instances with changeover costs in $[50, 100]$ (and $TF \in \{0.4, 0.5, 0.6\}$). The outcome of each single run of the ant algorithm is the subset of all nondominated solutions in the set of all solutions found during the run of the algorithm. The median attainment surfaces for runs with 1 colony and 10 colonies are shown in Figure 4 (the median attainment surface is the median line of all the attainment surfaces connecting the pareto front in every of the 11 runs). For the tests with 10 colonies we used pheromone update method 2 (update by region in the pareto front) and $\lambda$-rule 2 with overlapping $\lambda$-intervalls. The figure shows that the median attainment surfaces of the runs with 10 colonies are nearly always better than those for the 1 colony runs. Only for two instances with changeover costs in $[50, 100]$ the median attainment surfaces of the 1 colony runs are slightly better in a small region. But for these instances the 10 colonies found solutions with much smaller changeover costs.

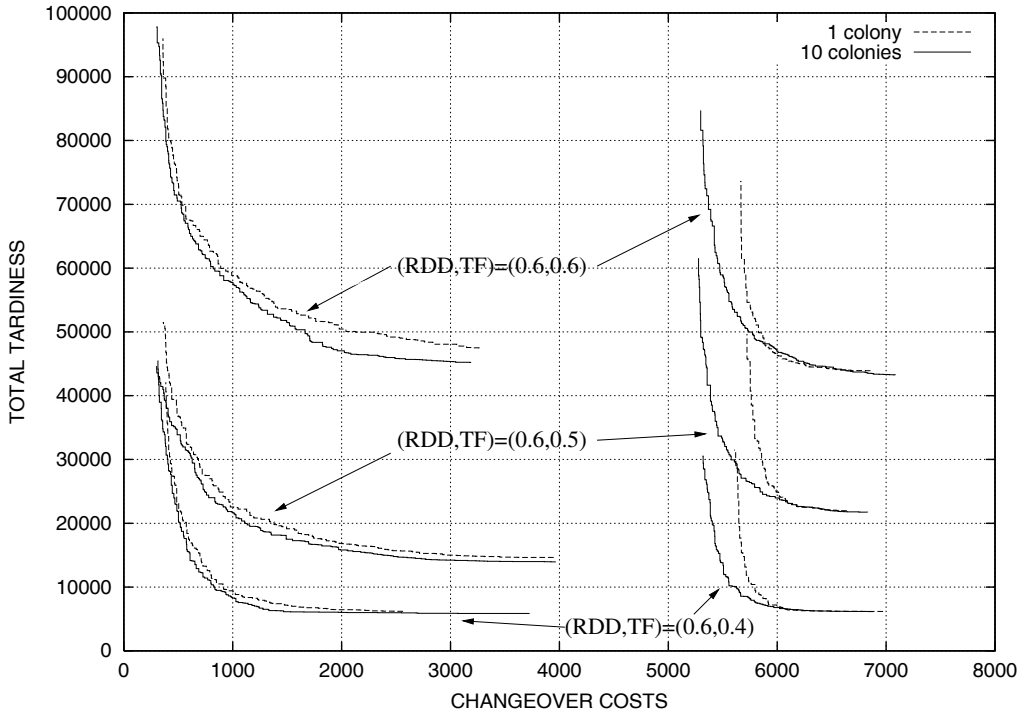


**Fig. 4.** Median attainment surfaces obtained with 1 colony and 10 colonies. Left part: three instances with changeover costs in [1,100]. Right part: three instances with changeover costs in [50,100].

In the following we present some results where we compare the different pheromone update methods and the $\lambda$-rules. For these tests we used the problem instance with $TF = 0.6$ and changeover costs in $[1, 100]$.

Figure 5 shows the convergence behaviour of the 10 colonies algorithm. The median attainment surfaces obtained after different numbers of generations are depicted. It can be seen that the median attainment surfaces after 200 generations and 300 generations differ not much. The results obtained for 1 colony were similar. Hence, our other results that were all obtained after 300 generations should not change with a higher number of generations.

The median attainment surfaces obtained with the different pheromone update methods are shown in Figure 6. The simple multistart strategy without cooperation between the colonies is worst along the whole surfaces. For the smaller costs values method 1 (update by origin) and method 2 (update by region in the nondominated front) show nearly the same performance. But median attainment surface when using method 2 is the best for medium and small total tardiness values.

Figure 7 shows from which colony the ants stem that found solutions which are in the final nondominated front (i.e. after 300 generations). It can be seen that the pheromone update method 2 (by region in the nondominated front) forces all colonies to specialize to those regions from where the ants come that

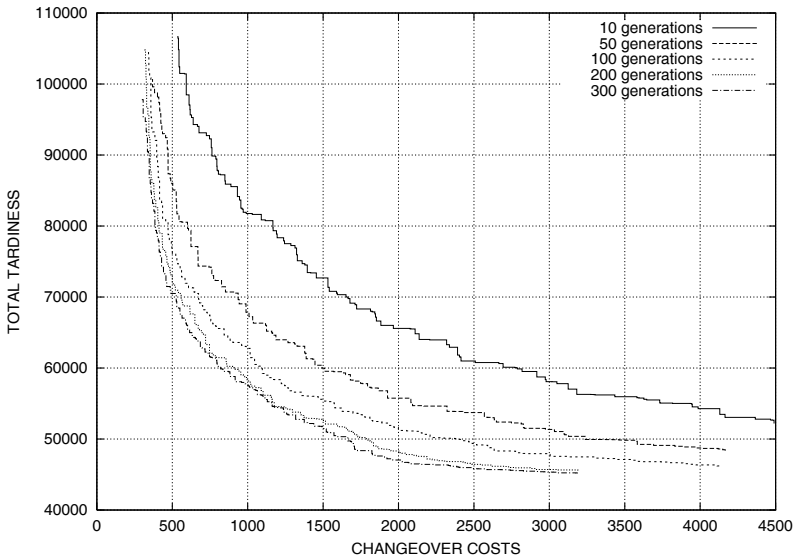


**Fig. 5.** Convergence behaviour with 10 colonies: median attainment surfaces are shown for generations 10, 50, 100, 200 and 300.

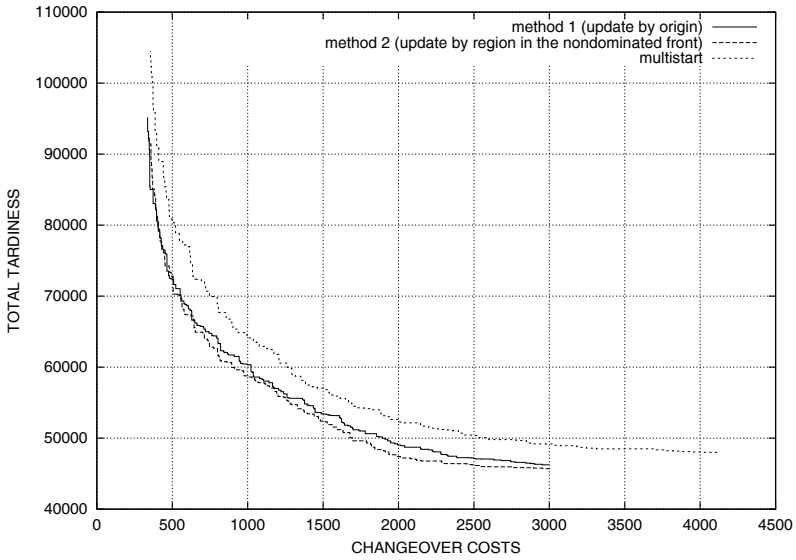


**Fig. 6.** Influence of the pheromone update method when using 10 colonies: method 1 — update by origin; method 2 — update by cutting the nondominated front; the multistart approach.

are allowed to update in that colony. Also, method 1 (update by origin) seems to force the colonies to specialize, though this effect is not so clear as for method 2. Clearly, for method 1 it can not be predicted to which region a colony will specialize.

The influence of the $\lambda$-rules when using pheromone update method 1 (by origin) is shown in Figure 8 for 2 and 5 colonies. When using rule 2 with disjoint $\lambda$-intervalls the case of 2 colonies clearly shows that colonies are forced to

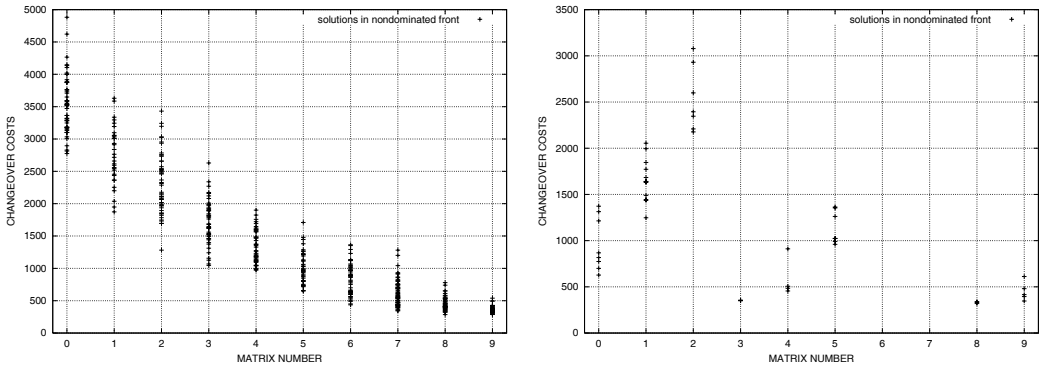


**Fig. 7.** Solutions in the nondominated front after 300 generations when using 10 colonies: changeover costs and number of the colony from which the corresponding ant stems. Left: pheromone update by region in the nondominated front (for all 11 runs). Right: pheromone update by origin (only for 1 run).

specialize to different regions of the nondominated front. The results for rule 2 in centres of these regions is quite good and better than for rule 3. But in the middle of the nondominated front and in the extreme regions the results for rule 2 are poor. Rule 3 shows a much more balanced behaviour. Rule 1 (all colonies have ants with $\lambda$-values in $[0, 1]$) perfomed worse than rule 3 along the whole median attainment surface (not shown in the figure). The results for 5 colonies are similar but the differences between the rules are not so big since the colonies can specialize to smaller regions. For 10 colonies only small differences were found (not shown here).

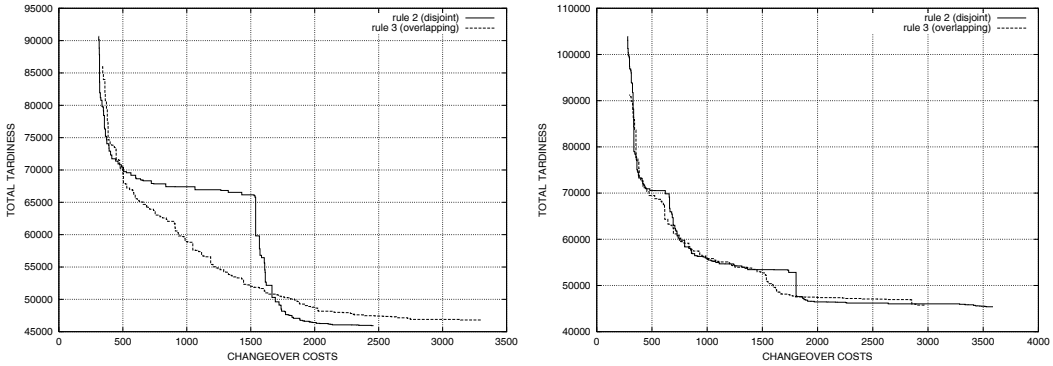


**Fig. 8.** Influence of the $\lambda$-rule when using 2 and 5 colonies: rule 2 — disjoint intervalls; rule 3 — overlapping intervalls.

Finally, we show that it is advantageous to use two pheromone matrices per colony (compare subsection 4.1): a place-job matrix suitable for minimizing the total tardiness and a job-job matrix suitable for minimizing changeover costs

(see Figure 9). When using only the place-job matrix in every colony the total tardiness values are good when changeover costs do not matter. But the ants were not able to find solutions with small changeover costs. When using only the job-job matrix in each colony we have the opposite effect: the ants can find solutions with small changeover costs but no solutions with small total tardiness values. Using two matrices per colony performed nearly always better along the whole median attainment surface than using only on of the matrices.

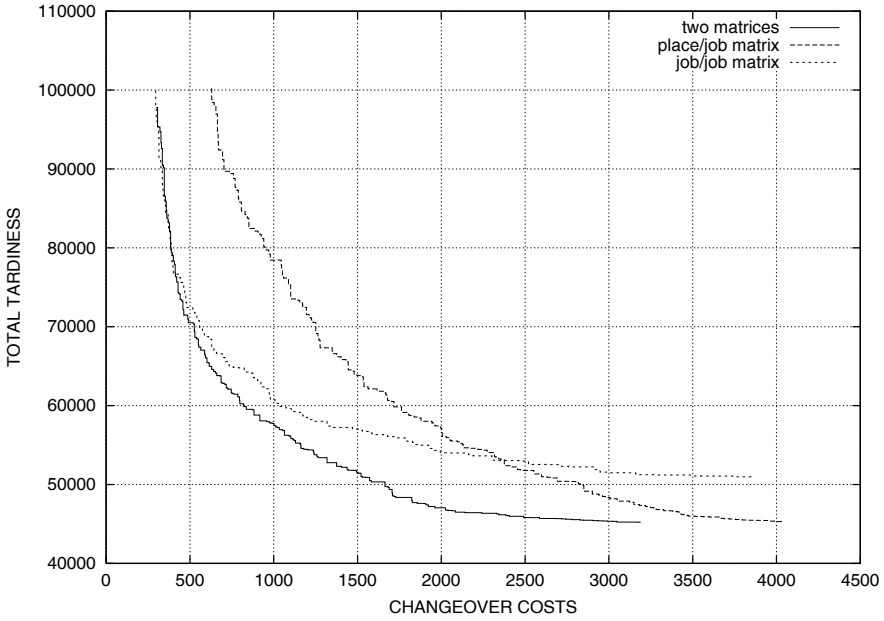


**Fig. 9.** Influence of number of matrices per colony: only one place-job matrix, only one job-job matrix, one place-job matrix and one job-job matrix. Test runs were performed with 10 colonies, pheromone by region in the nondominated front and overlapping $\lambda$-intervalls.

## 8 Conclusions and Future Work

We studied an approach to solve bi-criterion optimization problems with a multiple colony ant algorithm. It was shown that cooperation between the colonies allows to find good solutions along the whole the Pareto front. Heterogeneous colonies were used where the ants have different prefences. It was shown that the use of two different kinds of pheromone matrices in every colony — each matrix suitable for one optimization criterium — is advantagouos.

Currently we are studying methods to dynamically adapt the $\lambda$-values of the ants in a colony instead of using predetermined $\lambda$-intervalls. Moreover, the introduction of fitness sharing is under investigation. We expect that the use of elitist ants that update the pheromone matrices for solutions that are in the nondominated front of all solutions found so far will improve our results. Also, local pheromone update might be especially useful for multi-criterion optimization since it forces the ants to search for different sulutions.