International Journal of Systems Science
Vol. 36, No. 5, 15 April 2005, 255–266

On solving permutation scheduling problems with ant colony optimization

DANIEL MERKLE and MARTIN MIDDENDORF*

Department of Computer Science, University of Leipzig, D-04109 Leipzig, Germany

(*Received 6 August 2001; revised 1 January 2003; accepted 13 January 2005*)

A new approach for solving permutation scheduling problems with ant colony optimization (ACO) is proposed in this paper. The approach assumes that no precedence constraints between the jobs have to be fulfilled. It is tested with an ACO algorithm for the single-machine total weighted deviation problem. In the new approach the ants allocate the places in the schedule not sequentially, as in the standard approach, but in random order. This leads to a better utilization of the pheromone information. It is shown by experiments that adequate combinations between the standard approach which can profit from list scheduling heuristics and the new approach perform particularly well.

Keywords: Ant colony optimization; Permutation problems; Single machine scheduling; Single machine total tardiness problem; Pheromone information

1. Introduction

The ant colony optimization (ACO) metaheuristic has been applied successfully to various combinatorial optimization problems such as scheduling problems, vehicle routeing problems or assignment problems (for an introduction and overview see Dorigo *et al*. 1996, Dorigo and Di Caro 1999, Dorigo and Stützle 2004, Merkle and Middendorf 2004). Scheduling problems that have been solved with ACO are the job-shop problem (Colorni *et al*. 1994, Van Der Zwann and Marques 1999), the flow-shop problem (Stützle 1998), the single-machine total tardiness problem (SMTTP) and its weighted variant (Bauer *et al*. 1999, den Besten *et al*. 1999, 2000, Merkle and Middendorf 2000), the resource constrained project scheduling problem (Merkle *et al*. 2002), the group shop scheduling (Blum 2002) and scheduling problems from industry (Gagne *et al*. 2000). In ACO, several generations of artificial ants search for good solutions. Information exchange between the ants is based on principles of communicative behaviour that are found in real ant colonies. Every ant builds up a solution step by step, going through several probabilistic decisions until a solution is found. Ants that have found a good solution mark their paths through the decision space by putting some amount of pheromone on the edges of the path. The following ants of the next generation are attracted by the pheromone so that they will search in the solution space near the good solutions. In addition to the pheromone values the ants will usually be guided by some problem specific heuristic for evaluating the possible decisions.

It has already been shown that ACO can solve permutation scheduling problems such as the single-machine total weighted tardiness problem (SMTWTP) (Bauer *et al*. 1999, Den Besten *et al*. 2000, Merkle and Middendorf 2000) and flow-shop problems (Stützle 1998) very successfully. A comparison between ACO and other heuristics on a set of benchmark problems from the OR-library (2001) for the SMTWTP was made by den Besten *et al*. (2000). ACO was able to find for all 125 test instances with 100 jobs the best-known solutions. This was significantly better than the best-known tabu search method for SMTWTP. Only iterated dynasearch reached a similar performance as ACO.

There is a general approach to solving permutation scheduling problems such as the SMTWTP and flow-shop problems with ACO. Starting with the first place of the schedule, every ant constructs a solution by deciding iteratively which job is at the next place.

*Corresponding author. Email: middendorf@informatik.uni-leipzig.de

For every place i in the schedule and every job j there is pheromone information τ_{ij} about the desirability to put job j at place i. This approach is natural since for many permutation scheduling problems there exist good list scheduling heuristics which can be used by the ants in addition to the pheromone information. All ACO algorithms that have been proposed so far for the SMTTP, the SMTWTP and the flow-shop problem follow this approach (Bauer *et al.* 1999, den Besten *et al.*, 1999, 2000, Merkle and Middendorf 2000).

In this paper we identify a disadvantage of the standard approach to solving permutation scheduling problems with ACO. We propose a new approach that circumvents this disadvantage of the standard approach. Moreover, we show how to combine the standard approach which can profit from list scheduling heuristics with the new approach. Since the ACO algorithm (den Besten *et al.* 2000) that uses the standard approach was already able to find the best solutions for all the large benchmark instances for the SMTWTP in the OR-Library (2001) we tested our method on a somewhat more difficult problem where, in addition to the weighted tardiness costs, weighted earliness costs also have to be considered. This problem is called the single-machine total weighted deviation problem (SMTWDP).

The paper is organized as follows. The definition of the SMTWDP is given in section 2. The standard decision mechanism of the ants to build a solution is compared with the new approach in section 3. In section 4 we describe the ACO algorithm for the SMTWDP. Some variants and further aspects of the ACO algorithm are considered in section 5. The choice of the parameter values of the algorithms that are used in the test runs and the test instances are described in section 6. Experimental results are reported in section 7 and a conclusion is given in section 8.

2. The single-machine total weighted deviation problem

The SMTWDP is to find for a given set of jobs with due dates a one-machine schedule that minimizes the sum of the total weighted earliness and total weighted tardiness. Formally, the SMTWDP is to find for n jobs, where job j, $1 \le j \le n$, has a processing time p_j, a due date d_j and two weights h_j and w_j, a non-pre-emptive one-machine schedule that minimizes $D = \Sigma_{j=1}^{n}(h_j \max\{0, d_j - C_j\} + w_j \max\{0, C_j - d_j\})$ where C_j is the completion time of job j. D is called the total weighted deviation of the schedule. In this paper we do not allow idle times in the schedule between the jobs. Observe that $h_j$max $\{0,\ d_j - C_j\}$ is the weighted earliness of a job and w_j max $\{0,\ C_j - d_j\}$ is its weighted tardiness. It is known that SMTWDP is NP hard in the strong sense even when all weights are the same (Garey and Johnson 1979). Note that the SMTWTP problem is easier since it can be solved in pseudopolynomial time when all weights are equals (Garey and Gohnson 1979). A short overview of research that studied combinations of earliness and tardiness criteria has been given by Wu *et al.* (2000).

3. Decisions of the ants

In this section we identify a problem of the standard decision process in ACO algorithms for permutation scheduling problems. A new approach is proposed and compared with the standard approach. In order to make the discussion easier we assume that pheromone information already exists and that no heuristic information is used. The whole ACO algorithms are then explained in the following section.

3.1. *The standard approach*

The standard ACO approach is that ants build up a schedule by always extending an already fixed prefix. Thus, in the case of permutation scheduling problems the ant decides first which job is the first in the schedule, and then it decides which job is the second and so forth. The decisions are made randomly according to the pheromone information. For n jobs there is pheromone information τ_{ij}, $i, j \in [1; n]$, and the ith decision of an ant is then which job j to put on which place i in the schedule. If S is the set of unscheduled jobs, the probability p_{ij} to choose job $j \in S$ is $p_{ij} = \tau_{ij}/\Sigma_{h \in S}\tau_{ih}$. In the following we identify a disadvantage of this approach.

The general principle of ACO algorithms is that the pheromone information τ_{ij}, $i,j \in [1:n]$, reflects the outcomes of the decisions which have been made by former ants that found good solutions. The ants of the actual generation should use this pheromone information in an adequate way. Hence their decisions should be made according to the probability distribution that is determined by the relative size of the pheromone values corresponding to the possible outcomes of the decision.

The following observation shows a general problem for ACO algorithms to follow this guideline. Let us consider an example with $n = 3$ jobs and the pheromone values given by the following matrix:

$$|\tau_{ij}|_{i,\,j \in [1:3]} = \begin{vmatrix} 1/2 & 1/3 & 1/6 \\ 1/6 & 1/3 & 1/2 \\ 1/3 & 1/3 & 1/3 \end{vmatrix}.$$

This matrix is called the pheromone matrix. Then, for the first place in the schedule an ant simply chooses job $j \in [1:3]$ with probability $p_{1j} = \tau_{1j}$. Hence, the probability to choose job j equals exactly the relative amount of pheromone τ_{1j} for job j in the first row of the matrix. However, for the decision on which job to place in the second place the probabilities are different from the relative amount of pheromone in the second row. This is because the outcome of the decision for the first place has to be considered. In particular, $p_{21} = p_{12}[\tau_{21}/(\tau_{21} + \tau_{23})] + p_{13}[\tau_{21}/(\tau_{21} + \tau_{22})] = 25/180 < 1/6$. Similarly, $p_{22} = 56/180 < 1/3$ and $p_{23} = 99/180 > 1/2$. This example has shown that the first decision of an ant correctly reflects the intention of the ACO heuristic, namely that the probability distribution for the decision reflects the relative size of the pheromone values. However, as the example shows, this is not necessarily true for the following decisions.

In the following we show that the situation is given worse. For typical pheromone matrices that occur when solving permutation scheduling problems there is a systematic bias in the probability distributions that are used by the ants with respect to the distributions that correspond to the values in the rows of the pheromone matrix. A typical situation for problems with total earliness and tardiness costs such as the SMTWDP is that for every job there is a more or less preferred place in the schedule and the costs usually become higher the more the place of a job differs from the preferred place.

As an example we study an idealized SMTWDP instance with $n = 30$ jobs and where each job j has processing time $p_j = 1$, due date $d_j = j$, tardiness weight $w_j = 1$ and earliness weight $h_j = 1$. Clearly, the only optimal solution with total costs 0 has job j at place j in the schedule. We consider a hypothetical pheromone matrix that is similar to pheromone matrices that occur after some generations in the ACO algorithm when the largest pheromone values are centred around the pheromone values that correspond to the optimal decisions. The pheromone matrix is illustrated in figure 1 and formally defined as follows. For $i \in [3, 28]$, let $\tau_{ii} = 2/5$, $\tau_{i,i-1} = \tau_{i,i+1} = 1/5$, $\tau_{i,i-2} = \tau_{i,i+2} = 1/10$ and $\tau_{ij} = 1/20$ if $j \notin \{i-2, i-1, i+1, i+2\}$. In rows $i \in \{1, 2, 29, 30\}$ we define the pheromone values so that the sums of the pheromone values in all rows and columns of the matrix are equal because this is often the case in real ACO algorithms (in this example the sum is 9/4). Let $\tau_{11} = 11/20$, $\tau_{22} = 2/5$ and $\tau_{21} = \tau_{12} = 1/4$, $\tau_{30,30} = 11/20$, $\tau_{29,29} = 2/5$ and $\tau_{29,30} = \tau_{30,29} = 1/4$.

We let 100 000 ants construct a solution using the test matrix. Denote by m_{ij} the number of ants that have decided to place job j at place i of the schedule. Since in every row of the pheromone matrix the sum of the τ values is 9/4, the pheromone matrix suggests that $m^*_{ij} := \tau_{ij} \times (4/9) \times 100\,000$ ants should have placed job j at place i. Let $d_{ij} := m_{ij} - m^*_{ij}$ be the deviation of the observed values from the suggested values. The matrix $\mathbf{D}^+ := [\max\{0, d_{ij}\}]$ of the positive deviations and the matrix $\mathbf{D} = [\min\{0, d_{ij}\}]$ of the negative deviations are shown in figures 2 and 3 respectively. Note that the grey values in the figures show relative values between the maximum and the minimum value in each matrix and cannot be compared directly between the matrices.

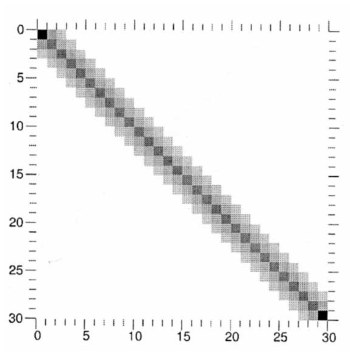
Figure 1. Pheronome matrix of the example problem: white = 1/20; black = 11/20.

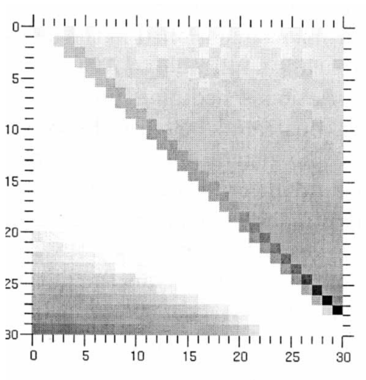
Figure 2. Matrix $\mathbf{L}^+$: white = 0, black = 5689.

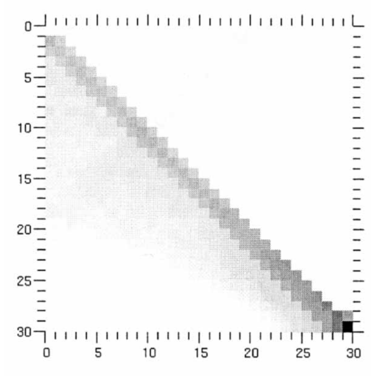
Figure 3. Matrix $\mathbf{L}^-$: white = 0, black = −24 554.

It can be seen from the figures that there is a systematic bias in the decisions of the ants. The negative values d_{ij} occur along the diagonal which means that the ants tend to choose these values, corresponding to good decisions that cause only small deviations not sufficiently often. The positive values occur above the diagonal and also in the bottom left corner. The reason for the high values above the diagonal is that the high pheromone values along the diagonal cause smaller probabilities to be chosen below the diagonal (and therefore to higher probabilities to the right of the diagonal). The reason for the high values in the bottom left corner can be explained as follows. Assume that an ant does not schedule a job when its high pheromone values occur. Then it will often happen that the job is placed near the end of the schedule when there are only a few alternatives left.

3.2. *The new approach*

In the following we propose a method to cope with the problems mentioned in the last section. The new approach is applicable to ACO algorithms that solve permutation scheduling problems where no precedence constraints between the jobs have to be considered. The general idea is that every ant determines in a random order over the places which job is assigned to the next place in the schedule. Ants that decide according to a random sequence are called random ants. The advantage of the random decision sequence is that every place has the same chance to be the first that is assigned a job. Because there is no bias in the first decision, the average of the decisions of the ants will better reflect the information that is contained in the pheromone matrix. It is likely that this will improve the optimization behaviour compared with an ACO algorithm following the standard approach.

To compare the new approach with the standard approach we let 100 000 random ants construct a solution using the test pheromone matrix from the last section. Let m_{ij}^{R} be the number of random ants that have decided to place job j at place i of the schedule and let $d_{ij}^{\mathrm{R}} := m_{ij}^{\mathrm{B}} - m_{ij}^{*}$. The matrix $\mathbf{D}^{\mathrm{R}+} := [\max\{0, d_{ij}^{\mathrm{R}}\}]$ of the positive deviations and the matrix $\mathbf{D}^{\mathrm{R}-} := [\min\{0, d_{ij}^{\mathrm{R}}\}]$ of the negative deviations are shown in figures 4 and 5 respectively.

Similar to the results for the standard approach the negative values d_{ij} occur along the diagonal, which means that the ants tend to choose these values, corresponding to good decisions with small deviations not sufficiently often. However, in contrast with the standard approach the positive values are randomly distributed in the rest of the matrix. There is no clear bias to some specific region. This is an advantage

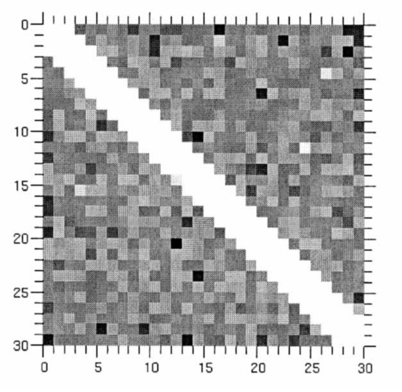
Figure 4. Matrix $\mathbf{L}^{\mathrm{R}+}$: white = 0, black = 278.

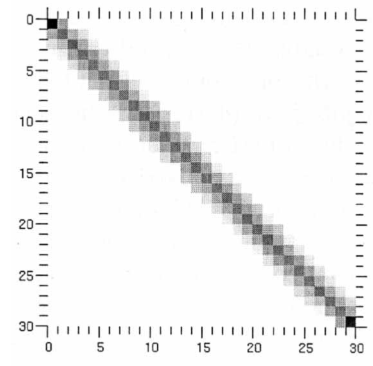
Figure 5. Matrix $\mathbf{L}^{\mathrm{R}-}$: white = 0, black = −3 574.

because it indicates that an ACO algorithm using random ants might find good solutions.

Comparing the absolute value gives further indications for the strength of the random ants. For the standard ants the sum of all positive values d_{ij} in M (which equals the absolute value of the sum of all negative values m_{ij}) was 418 418, that is about 14.0% of all 3 000 000 decisions made by the ants. For the random ants the sum of all positive values d_{ij}^{R} in M was only 112 209, that is only about 3.7% of all 3 000 000 decisions by the ants.

The weighted deviation of all decisions of the standard ants that exceed (fall below) their expected value was 4 597 388 (968 992 respectively); that is, the average deviation of each of these decisions was 10.99 (2.32 respectively). Compared with that, the weighted deviation of all decisions of the random ants that exceed (fall below) their expected value was only 1 336 926 (−55 480 respectively). The average deviation of each of these decisions was 11.91 (0.50 respectively) which is similar to the corresponding values for the standard ants. The maximum positive (negative) value d_{ij} was 5 689 (−24 544 respectively) and the maximum positive (negative) value d_{ij}^{R} was 278 (−3 574 respectively). The comparison of the absolute values also shows that the random ant approach leads to decisions that better reflect the probabilities which correspond to relative size of the pheromone values. Moreover, it does not have such a strong bias as the standard approach.

It should be mentioned that some care has to be taken with an interpretation of our findings. Since many ACO algorithms use a selection mechanism that allows only the best ant in a generation to update the pheromone values, it is not necessarily true in such a case that generations of ants which are not good on the average lead to an algorithm that is not good. Moreover, the most important contribution of the ants in a generation is not always to only find a good solution. It is also important which of the pheromone values will be increased because this influences the next generation of ants. When the pheromone update mechanism introduces a systematic bias it could become more and more unlikely that the following generations will find the optimal solution. Nevertheless, since the random ants found solutions that better reflect the pheromone information and since the deviations show no strong bias, it is likely that they can contribute to a good optimization behaviour.

3.3. *Combinations of both approaches*

It should be mentioned that there is a problem with the random ant approach. An ant that assigns jobs to the places in a random order cannot usually profit from heuristics that are based on list scheduling. Since there exist good list scheduling heuristics for many permutation scheduling problems and heuristic information is in general important for the optimization behaviour of ACO algorithms, it is not clear whether the new approach (without a heuristic) performs better than a standard ACO algorithm that uses a powerful list scheduling heuristic.

Therefore, we propose to combine both approaches. Some ants should decide according to the standard sequential order and use the (list scheduling) heuristic while the other ants make decisions according to a random order without using heuristic information, but care has to be taken when two types of ants work together. When both types of ant are in the same generation, it might often be the case that ants of one type are better than the others. Then, only the ants of the better type will have a chance to update the pheromone matrix, while the other ants are useless. We assume here that only the better ants are allowed to update the pheromone information. Note that in the first ACO algorithms that appeared in the literature, usually all ants are allowed to update so that the amount of pheromone depends on the quality of the solution that was found by the ant. However, it has been shown meanwhile that for most problems it is advantageous to allow only the best ant to update (see, for example, Dorigo and Di Caro (1999) and Dorigo and Stützle (2004). We circumvent this problem by using ants of different types in different generations so that competition occurs only between ants of the same type.

4. The ant colony optimization algorithm for the single-machine total weighted deviation problem

The ACO algorithm for the SMTWDP is explained in detail in this section. It is an iterative algorithm where in every generation (iteration) each of m ants constructs one solution for the SMTWDP. We use two types of ant. The so-called sequential ants select the jobs in the order in which they will appear in the schedule, whereas the random ants as introduced in section 3 select the jobs according to some random order in which they will appear in the schedule. For the selection of a job both types of ant use pheromone information. The sequential ants use also some heuristic information. Heuristic information, denoted by η_{ij}, and pheromone information, denoted by τ_{ij}, are indicators 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 pheromone information stems from former ants that have found good solutions. The next job is chosen according to the probability distribution over the set of unscheduled jobs S determined by

$$p_{ij} = \frac{\tau_{ij}\eta_{ij}}{\Sigma_{h\in S}\tau_{ih}\eta_{ih}} \quad \text{or} \quad p_{ij} = \frac{\tau_{ij}}{\Sigma_{h\in S}\tau_{ih}}$$

for the sequential ants and the random ants respectively.

The heuristic values η_{ij} are computed according to a heuristic that has been obtained by modifying a heuristic proposed by Ow and Morton (1998). The idea is that a sequential ant chooses the next job from the set of jobs that already exceed their due date (with respect to the sum of the processing times of all jobs that are scheduled so far) or will exceed it when they are scheduled next (if there exists such a job). For every one of these jobs the costs will become higher when it is scheduled later. From these jobs the shorter jobs and those with a high tardiness weight should be scheduled first. Hence, for $d_j \leq T + p_j$, where T is the sum of the processing times of all jobs that have already been scheduled, the heuristic value is (recall that w_j is the tardiness weight of job j)

$$n_{ij} = \frac{w_j}{p_j}.$$

Some of the jobs that will not exceed their due date when they are scheduled next might exceed it when they are scheduled after some other job that is scheduled next. To be able to consider those jobs we apply some foresight to the heuristic apply. Every job j that has a processing time which is smaller than the average processing time and where the due date exceeds $T+p_j$ by at most $(\bar{p}-p_j)/[w_j/(w_j+h_j)]$ is given some positive heuristic value. Observe that the last value becomes larger for jobs which have tardiness costs that are relatively high compared with the earliness costs. The assigned heuristic values fall linearly with increasing due date from (w_j/p_j) to zero in the interval $[T+p_j,\ T+p_j+(\bar{p}-p_j)[w_j/(w_j+h_j)]$. Hence, if $d_j \in [T+p_j,\ T+p_j+\max\{0, \bar{p}-p_j\}/[w_j/(w_j+h_j)]]$, then

$$n_{ij} = \left(1 - \frac{d_j - T - p_j}{\max\{0, \bar{p}-p_j\}[w_j/(w_j+h_j)]}\right)\frac{w_j}{p_j}.$$

When each remaining job j has a due date $d_j > T+p_j+(\bar{p}+p_j)[w_j/(w_j+h_j)]$, then the heuristic value η_{ij} is defined by (recall that h_j is the earliness weight of job j)

$$\eta_{ij} = \frac{p_j}{h_j}$$

so that jobs with a long processing time and small earliness weight will be preferred.

After all m ants of the generation have constructed a solution, the ant that found the best solution in that generation is allowed to update the pheromone matrix but, before that, some of the old pheromone is evaporated according to

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

The reason for this is that old pheromone should not have too strong an influence on the future. Then, for every job j in the schedule of the best solution found in the generation, some amount of pheromone is added to element τ_{ij} of the pheromone matrix where i is the place of job j in the schedule. The amount of pheromone added is $1/D$, where D is the total deviation of the schedule, that is

$$\tau_{ij} = \tau_{ij} + \frac{1}{D}.$$

The algorithm stops when a certain number of generations has been achieved. We tested different modes of alternation between generations of sequential ants and generations of random ants.

5. Additional aspects and variants

Some variants of the ACO algorithm that were described in the last section which concern the pheromone evaluation and the type of ants used are described in this section.

5.1. *Pheromone summation rule*

An alternative way to use the pheromone information is explained in the following. It was proposed by Merkle and Middendorf (2000) for the SMTWTP. Since the SMTWTP is the variant of the SMTWDP where all weights h_j are zero, that is only the tardiness of a job counts, we use this pheromone evaluation method (called pheromone summation evaluation, by Merkle and Middendorf (2000)) here also.

The following problem occurs when using the relative pheromone values directly as the probability to choose the next job. Assume that by chance an ant chooses to put some job h at place i of the schedule that has a low pheromone value τ_{ih} (instead of job j that has a high pheromone value τ_{ij}). Then in order to have a high chance to still end up with a good solution it will probably be necessary for the ant to place job j not too late in the schedule when j has a small due date. To handle this problem it was proposed by Merkle and Middendorf (2000) to let the pheromone value τ_{ij} also influence later decisions when choosing a job for some place $l > i$. A simple way to guarantee this influence is to use the sum of all pheromone values for every job from the first row of the matrix up to row i when deciding about the job for place i. Then a sequential ant chooses the next job for place i in the schedule according to the probability distribution over S determined by

$$p_{ij} = \frac{(\Sigma_{k=1}^{i}\tau_{kj})\eta_{ij}}{\Sigma_{h\in S}[(\Sigma_{k=1}^{i}\tau_{kh})\eta_{ih}]}. \quad (1)$$

5.2. *Backward ants*

Merkle and Middendorf (1999) proposed the use of (sequential) forward and (sequential) backward ants for solving the shortest common supersequence problem. Here we study which kind of ants, forward ants or backward ants, are better for the different types of problem instances. Backward ants construct solutions by assigning jobs to the places of the schedule in reverse order; that is they first decide which job is the last. Clearly, for the sequential backward ants the heuristic that they use has to be modified accordingly. This is done as follows.

A sequential backward ant starts by choosing the last job that finishes at time $\sum_{j=1}^{n} p_j$. It always choose the next job from the set of jobs that already exceed their due date (if there exists such a job). Of these jobs the shorter jobs and those with a high earliness weight should be scheduled first. Hence, for $d_j \geq T$, where T is the sum of the processing times of all remaining jobs that have not been scheduled so far, the heuristic value is

$$\eta_{ij} = \frac{h_j}{p_j}.$$

Similarly, for the sequential forward ants, some of the jobs which will not exceed their due date when they are scheduled next will also be considered. All those jobs that have a processing time which is smaller than the average and where the due date falls not before $T - (\bar{p} - p_j)[h_j/(w_j + h_j)]$ are given a positive heuristic value. If $d_j \in [T - \max\{0, \bar{p} - p_j\}[h_j/(w_j + h_j)T]$ then

$$\eta_{ij} = \left(1 - \frac{T - d_j}{\max\{0, \bar{p} - p_j\}[h_j/(w_j + h_j)]}\right)\frac{h_j}{p_j}.$$

6. Test instances and parameters

We tested the different variants of the algorithm on SMTWDP instances of size 100 jobs. The instances were generated as follows: for each job $j \in [1, 100]$, an integer processing time p_j is taken randomly from the interval $[10, 100]$, the earliness weight h_j is taken randomly from the interval $[1e, 2e]$, the tardiness weight w_j is taken randomly from the interval $[1t, 2t]$, and an integer due date d_j is taken randomly from the interval

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

where TF is the tardiness factor and RDD is the relative range of due dates.

Note that this rule was also used for creating the benchmark instances for the SMTWTP that can be found in the OR-Library (2001), the parameters e and t allow control of the average influence of the earliness and tardiness weights for a problem instance. The RDD value determines the length of the interval from which the due dates were taken. TF determines the relative position of the centre of this interval between 0 and the sum of the processing times $\Sigma_{j=1}^{100} p_j$. The values for TF are chosen from the set $\{0.2, 0.4, 0.6, 0.8\}$. RDD was set 0.4; that is the due dates cover a range of 40% of the computation interval. For (e, t) we tested the combinations $(5, 1)$, $(3, 1)$, $(1, 1)$, $(1, 3)$ and $(1, 5)$. For every test and each combination of TF and (e, t) we used a set of at least 15 test instances. The parameter ρ was set to 0.01 and the number of ants in every generation was $m = 20$. Every run was stopped when the average solution quality found by the ants in a generation has not changed over 100 generations or after 20 000 generations.

We use the following notation for the different versions of the ant algorithm: F-A, only sequential forward ants are used as described in section 4; B-A, only sequential backward ants are used as described in section 5.2 (the corresponding versions where the ants use the pheromone summations rule as in section 5.1 are called ΣF-A and ΣB-A respectively; R-A, only random ants are used. We call these algorithms the uniform variants because all generations of ants work in the same manner. The following algorithms are called heterogeneous because they use generations of ants that work differently. Combinations between F-A, B-A, ΣF-A and ΣB-A with R-A where exactly the even generations work according to R-A are denoted by FR-A, BR-A, ΣFR-A and ΣBR-A respectively.

7. Experimental results

In this section we present experimental results on the performance of the different ACO algorithms. Moreover, some specific aspects such as the best number of ants per generation and the best relative number of generations with random ants for the heterogeneous algorithms are discussed.

7.1. *Performance of the uniform algorithms*

A comparison between the relative optimization behaviours of the uniform algorithms F-A, B-A, ΣF-A, ΣB-A and R-A is depicted in figure 6. The figure shows for each algorithm and each combination of tardiness factor TF and (e, t) its relative performance measured as percentage average loss in solution quality compared with the solution of the best of these five

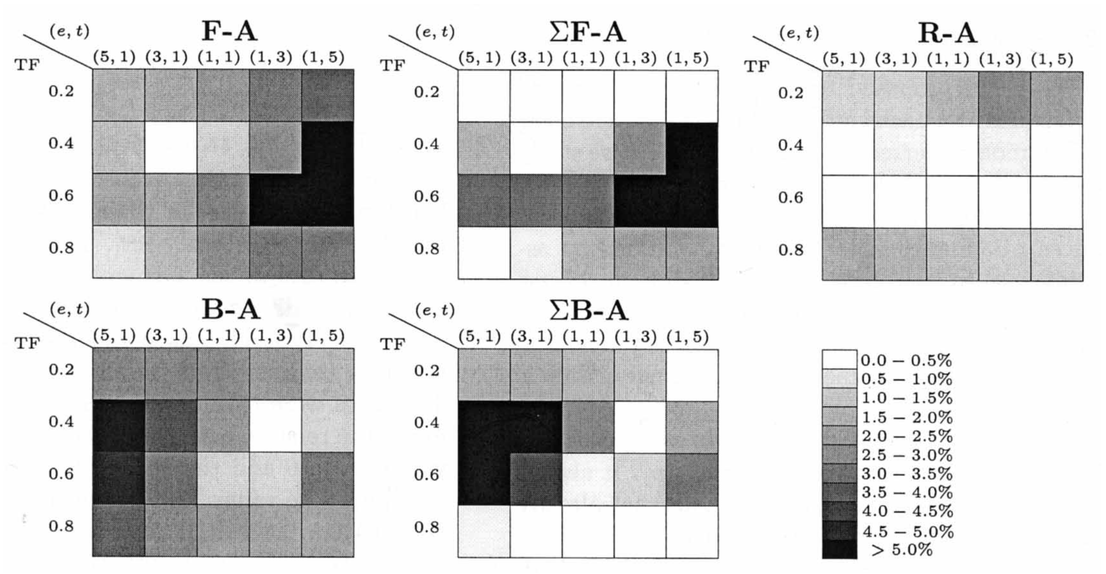

Figure 6. Relative performances of F-A, B-A, ΣF-A, ΣB-A and R-A. For each algorithm and combination of tardiness factor TF and (e, t) value the average percentage loss in solution quality compared with the best-performing algorithm of the five variants for that combination of TF and (e, t) values is shown. Thus for each combination of TF and (e, t) the best-performing algorithm variant has a white box.

algorithms for the specific combination of TF and (e, t). It can be seen that the algorithm R-A with only random ants performed in general quite well. It was best in about half of the cases where the due dates lie in the middle of the scheduling interval (TF = 0.4 and TF = 0.6). ΣF-A performed best when the due dates are more at the end (TF = 0.2) and high earliness costs have to be paid for the jobs that are scheduled first. The opposite is true for ΣB-A. For relative high earliness weights (large e values) ΣF-A performs better than for relative high tardiness weights (large t values). Again, the opposite is true for ΣB-A. Analogous remarks hold for F-A and B-A which performed, in general, worse than their counterparts with the pheromone summation evaluation. These results coincide with an observation of den Besten *et al.* (2000) that for the SMTWTP the instances with high TF values are more difficult than those with smaller TF values (using only forward ants). Our results help to explain this phenomenon since they show that the preferred working direction of the ants should depend on the type of the problem instances so that ants make the most important decisions first. This observation is of general interest for ACO algorithms.

7.2. *Performance of the heterogenous algorithms*

Comparing the heterogenous ACO algorithms that use two types of ant (FR-A, BR-A, ΣFR-A and ΣBR-A) with their uniform counterparts with a single type of ant (F-A, B-A, ΣF-A and ΣB-A) our test results show that the two-type variants are in every case better than the corresponding single-type variants. A comparison of the performance of the heterogenous algorithms and the uniform algorithm R-A is given in figure 7. When due dates are more at the end (TF = 0.2) or the beginning (TF = 0.8), algorithms ΣFR-A and ΣBR-A are the best variants. It is interesting that in these cases the sequential ants which use the pheromone summation evaluation can profit from the combination with the random ants because ΣF-A and ΣB-A alone are better than R-A in these cases. For due dates more in the middle (TF = 0.4 and TF = 0.6), algorithm FR-A is the best when the earliness weights are relatively large compared with the tardiness weights. For the opposite case where earliness weights are relatively small compared with the tardiness weights, BR-A is the best variant. This shows again that important decisions that can save high costs should be made first.

In order to make our algorithms comparable with those of other researchers we also give the absolute performance results of the algorithms for SMTWDP and different combinations of tardiness factor TF and (e, t). Table 1 shows for the different combinations of TF and (e, t) which algorithm variant performed best (of all the uniform and heterogeneous variants F-A, B-A, ΣF-A, ΣB-A, R-A, FR-A, BR-A, ΣFR-A and ΣBR-A) and the corresponding obtained solution quality. It follows from the results in the table that in all cases a heterogeneous algorithm performs better than all uniform algorithms. Observe that for every row in the table in the left-hand part (right-hand part)

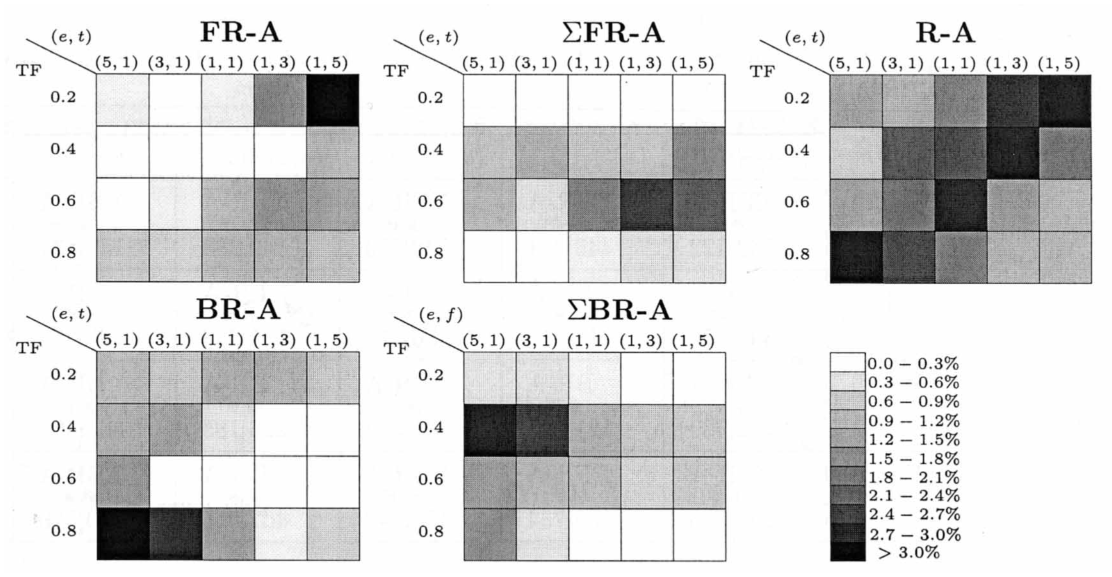

Figure 7. Relative performances of FR-A, BR-A, ΣFR-A, ΣBR-A and R-A. For each algorithm and combination of tardiness factor TF and (e, t) value the average percentage loss in solution quality compared with the best-performing algorithm of the five variants for that combination of TF and (e, t) values is shown. Thus for each combination of TF and (e, t) the best-performing algorithm variant has a white box.

Table 1. Average total deviation for the SMTWDP for the best algorithm of F-A, B-A, ΣF-A, ΣB-A, R-A, FR-A, BR-A, ΣFR-A and ΣBR-A.

	Best algorithm and average total deviation for the following (e, t) values				
TF	(5, 1)	(3, 1)	(1, 1)	(1, 3)	(1, 5)
0.2	ΣFR-A	ΣFR-A	ΣFR-A	ΣBR-A	ΣBR-A
	739 122	443 851	148 453	149 288	149 834
0.4	FR-A	FR-A	FR-A	FR-A	BR-A
	349 575	218 472	86 738	124 358	160 203
0.6	FR-A	BR-A	BR-A	BR-A	BR-A
	152 589	121 241	87 874	226 185	363 407
0.8	ΣFR-A	ΣFR-A	ΣBR-A	ΣBR-A	ΣBR-A
	155 756	155 7340	154 640	462 516	770 590

there are algorithms with the sequential ants that are forward ants (back-ward ants). This shows again that the preferred working direction of the ants should consider important decisions first.

7.3. *Relative influence of random ants in heterogeneous algorithms*

Since the heterogenous algorithms were quite successful we were interested to find how many generations of random ants should be executed per sequential generation. With FR-(x, y)-A we denote the algorithm where x generations of random ants alternate with y generations of sequential forward ants. For ΣFR-(x, y)-A this notation is used analogously. Tests were carried out with $(x, y) \in \{(\infty, 0), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (0, \infty)\}$. Note that FR-$(\infty, 0)$-A is the same as R-A and that FR-$(0, \infty)$-A is the same as F-A. Note also that we do not make corresponding tests with BR-(x, y)-A and ΣBR-(x, y)-A because the results will basically be symmetric.

The test results for Fr-(x, y)-A are shown in figure 8. It can be seen that, in all cases, sequential generations and random generations should have a significant influence. The best rats are between $(5, 1)$ to $(1, 2)$ for F-(x, y)-A. Further, the earlier the due dates, the higher should be the rate of random generations. Thus, for TF values not less than 0.6, significantly

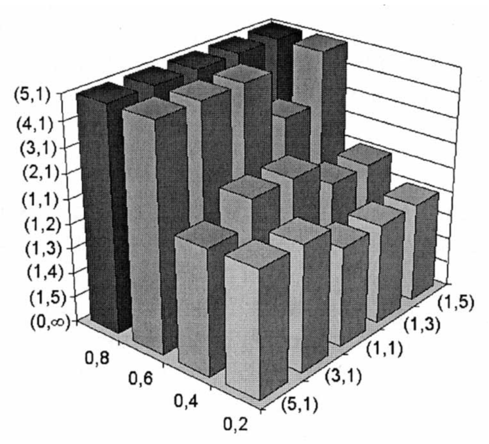

Figure 8. Best relations between number of generations with random ants and number of iterations with sequential ants for FR-(x, y)-A and different TF and (e, t) values: the heights of the bars indicate which (x, y) pair performed best.

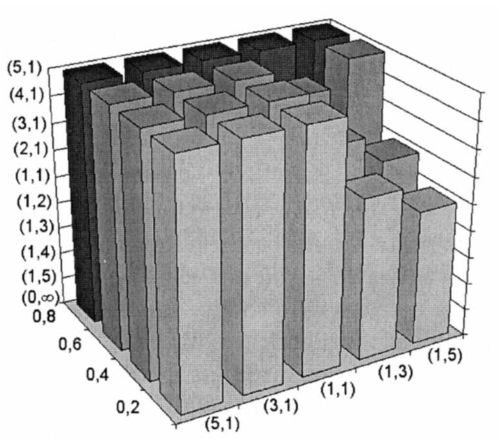

Figure 9. Best relations between number of generations with random ants and number of iterations with sequential ants for ΣFR-(x, y)-A and different TF and (e, t) values: the heights of the bars indicate which (x, y) pair performed best.

more random generations should be obtained and, for TF values not greater than 0.4 an equal (or slightly higher) number of generations with sequential ants should be determined. This corresponds to the observation that forward ants are especially good for problem instances with large due dates (i.e. with small TF values). Note that in the former case it is better to use backward ants instead of forward ants. Thus, when the pheromone summation rule is not used, we conclude the following: when using sequential generations with the best sequential ants (forward versus backward) and random generations, about the same number of generations of both types should be used (or slightly more of the former).

The test results for ΣF-(x, y)-A are depicted in figure 9. The results show that a strong influence of random generations is best for most cases. The best rates vary between $(5, 1)$ and $(1, 1)$. Only late due dates ($\mathrm{TF} = 0.4$ and $\mathrm{TF} = 0.2$) and $(e, t) = (1, 3)$ or $(e, t) = (1, 5)$ with equal (or only slightly larger) rates of sequential ant generations and random generations performed best, but these are not the cases where ΣFR-A performed best (compare table 1). Thus, when the pheromone summation rule is used, we conclude the following: when using generations with the best sequential ants (forward versus backward) and generations with random ants, the number of generations with random ants should be higher. Note that this conclusion is different from the corresponding conclusion for the case when the pheromone summation rule is not used. A possible explanation for this difference is that R-A is already better than F-A but is worse than ΣF-A in the relevant cases (see figure 6). This means that sufficient random generations must be used in the latter case before they can have a significant influence on the optimization process.

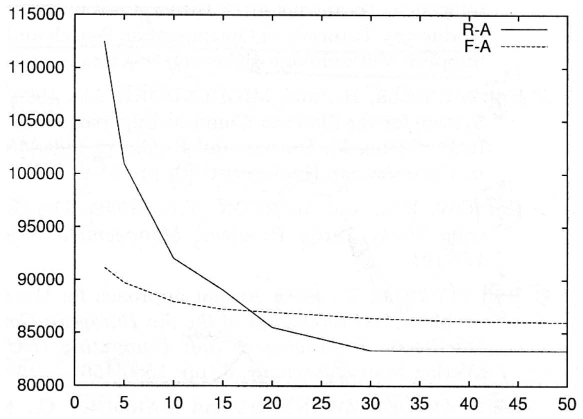

Figure 10. Best average total deviations of F-A and R-A for different numbers of ants per generation ($\mathrm{TF} = 0.4$ and $(e, t) = (1, 1)$)

7.4. *Influence of the number of ants*

Since random ants use different random sequences for their decisions, it can be expected that the solution qualities obtained by random ants in a generation show a larger variance than the solution qualities of a generation of forward ants. This makes it likely that the selection pressure of which ants are allowed to update the pheromone information should be larger for random ants than for forward ants. Thus, the number of ants in a generation should not be too small for random ants. Therefore, we tested the influence of the number of ants per generation on the optimization behaviour of the algorithms. Figure 10 shows the results for the case when $\mathrm{TF} = 0.4$ and $(e, t) = (1, 1)$. Not surprisingly, the obtained solution qualities of both algorithms increase with increasing number of ants per generation. More interesting is the fact that the results support our hypothesis because there is a strong decline in performance for R-A when there are fewer than 20 ants per generation. This is different for F-A which performs reasonably well also for a small number of ants per generation. In comparison, F-A performed better than R-A for 15 or fewer individuals per generation whereas R-A performed better for 20 or more individuals.

7.5. *Summary of experimental results*

The most important findings from the experiments and some advice that follows from these findings can be stated as follows.

(1) It depends on the problem instance whether an ACO with forward ants or backward ants performs better. As a general rule it is suggested that the working direction of the ants is chosen so that ants make important decisions early.
(2) Random ants perform quite well (i.e. better or only slightly worse) when compared with uniform algorithms with either forward or backward ants.
(3) The best heterogeneous algorithms where generations of random ants alternate with generations of sequential ants perform better than all uniform algorithms.
(4) For a heterogeneous algorithm we conclude firstly that, when not using the pheromone summation rule, about the same number of generations with random ants and sequential ants should be used, and secondly that when using the pheromone summation rule, more generations with random ants than generations with sequential ants should be used. In both cases it is assumed that the best version of sequential ants (forward ants or backward ants) is used.
(5) The number of ants per generation strongly effects the performance of algorithms with random ants and should be higher than for generations with sequential ants. As a general rule it can be said that ten ants per generation are often sufficient for sequential ants whereas at least twenty ants should be used for random.

8. Conclusion

We have proposed a new approach for solving permutation scheduling problems with ACO. Instead of using the same sequence of decisions for all ants in the new approach the ants use random sequences. Using a simple test problem it was shown that the standard approach leads to an unwanted systematic bias in the decisions of the ants whereas in the new approach the ants' decisions better reflect the pheromone information. Tests for the SMTWDP have shown that a combination between generations of ants that use the standard method of always extending a prefix of the schedule with generations of ants that allocate the places in the schedule in random order performs particularly well. We further characterized the influence of different types of problem instance on the best working direction (forward or backward) that the ants should follow and on the optimal mixture between generations of random ants and generations of ants that use the standard sequential method. In general, our approach of using generations of random ants in combination with generations of standard sequential ants is applicable to any permutation problem and it is interesting to investigate its usefulness for other types of permutation problem.

References

A. Bauer, B. Bullnheimer, R.F. Hartl and C. Strauss, "An ant colony optimization, approach for the single machine total tardiness problem", *Proceedings of the 1999 Congress on Evolutionary Computation*, Amsterdam: Elsevier, 1999, pp. 1445–1450.

C. Blum "ACO applied to group shop scheduling: a case study on intensification and diversification", *Proceedings of the 3rd International Workshop on Ant Algorithms* Lecture Notes in Computer Science, Vol. 2463, Berlin: Springer, 2002, pp. 14–27.

A. Colorni, M. Dorigo, V. Maniezzo and M. Trubian, "Ant system for job-shop scheduling", *JORBEL–Belgian Journal of Operations Research, Statistics and Computer Science*, 1994, 34, 39–53.

M.L. den Besten, T. Stützle and M. Dorigo, "Scheduling single machines by ants", Technical report IRIDIA/99-16, Institut de Recherches Interdisciplinaires et de Développements en Intellgence Artificelle, Université Libre de Bruxelles, Brussel, Belgium, 1999.

M.L. den Besten, T. Stützle and M. Dorigo, "Ant colony optimization for the total weighted tardiness problem", in *Parallel Problem Solving from Nature: 6th International Conference* Lecture Notes in Computer Science, Vol. 1917, K. Deb, G. Rudolph, X. Yao, E. Lutton, J.J. Merelo, and H.-P. Schweber, editiors, Berhin, Springer, 2000, p. 611–620.

M. Dorigo and G. Di Caro, "The ant colony optimization meta-heuristic", in *New Ideas in Optimization*, D. Corne, M. Dorigo and F. Glover, editors, New York: McGraw hill, 1999, pp. 11–32.

M. Dorigo, V. Maniezzó and A. Cólorni, "The ant system: optimization by a colony of cooperating agents", *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 1996, 26, pp. 29–41.

M. Dorigo and T. Stützle, *Ant Colony Optimization*, Boston, MA: The MIT Press, 2004.

C. Gagné, M. Gravel and W.L. Price, "Scheduling a single machine where setup times are sequence dependent using an ant-colony Heuristic", in *Abstract Proceedings of the International Workshop on Ant Algorithms*, M. Dorigo, L.M. Gambordello, M. Middendorf and T. Stützle, editors, 2000, pp. 157–160.

M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, New York: W. H. Freeman, 1979.

D. Merkle and M. Middendorf, "An ant algorithm with a new pheromone evaluation rule for total tardiness problems", *Proceedings of* *the EvoWorkshops 2000* Lecture Notes in Comptuer Science, Vol. 1803, Berlin: Springer, 2000, pp. 287–296.

D. Merkle, M. Middendorf and H. Schmeck, "Ant colony optimization for resource-constrained project scheduling", *IEEE Transactions on Evolutionary Computation*, 2002, 6, 333–346.

D. Merkle and M. Middendorf, "Swarm intelligence", in Introductory Tutorials in Optimisation, Search and Decision Support Methodology, Kluwer, 2005.

R. Michels, and M. Middendorf, "An ant system for the shortest common supersqeuence problem, in D. Corne, M. Dorigo and F. Glover, editors, New York: McGraw Hill, 1999, pp. 51–61.

OR-library, http://mscmga.ms.ic.ac.uk/jeb/orlib/wtinfo.html.

P.S. Ow and T.E. Morton "The single machine early/tardy problem", *Management Science*, 1998, 35, 177–191.

T. Stützle, "An ant approach for the flow shop problem", *Proceedings of the 6th European Congress on Intelligent Techniques* '*Soft Computing*, Aachem: Verlag Mainz, 1998, pp. 1560–1564.

S. Van Der Zwaan and C. Marques, "Ant colony optimisation for job shop scheduling", *Proceedings of the 3rd Workshop on Genetic Algorithms and Artifical Life*, 1999.

C.-C. Wu, W.C. Lee and J.M. You, "Trade-off solutions in a single machine scheduling problem for total earliness and maximum tardiness", *International Journal of Systems Science*, 2000, 31, 639–647.