An Ant Colony Optimization Approach for the Single Machine Total Tardiness Problem[†]

Andreas Bauer
Bernd Bullnheimer[‡]
Richard F. Hartl
Christine Strauss

Department of Management Science
University of Vienna

Abstract

Machine scheduling is a central task in production planning. In general it means the problem of scheduling job operations on a given number of available machines. In this paper we consider a machine scheduling problem with one machine, the Single Machine Total Tardiness Problem. To solve this NP-hard problem, we apply the ant colony optimization metaphor, a recently developed meta-heuristic that has proven its potential for various other combinatorial optimization problems. We test our algorithm using 125 benchmark problems and present computational results.

1 Introduction

Ant Colony Optimization (ACO) is a rather new meta-heuristic introduced in the early nineties (cf. [6, 7, 11, 15]) and has successfully been applied to several combinatorial optimization problems (cf. e.g. [4, 5, 8, 9, 16, 21, 25]). In this paper we apply ACO to the Single Machine Total Tardiness Problem

[†]We would like to thank Herbert Dawid and Marco Dorigo for their contributions to this research. Financial support from the Austrian Science Foundation under Grant F 01005 "Adaptive Information Systems and Modelling in Economics and Management Science" is gratefully acknowledged.

[‡]Corresponding author: Bernd Bullnheimer, Department of Management Science, University of Vienna, Brünner Strasse 72, A-1210 Vienna, Austria; fax: +43-1-29128-504; email: bernd.bullnheimer@univie.ac.at.

(SMTTP), a well-known NP-hard combinatorial optimization problem that has been tackled by various optimization procedures (cf. [10, 19, 24, 27, 31, 33]) and heuristic approaches (cf. [20, 23, 26, 28, 34]).

The purpose of this study is twofold: we present another evidence of the versatility of ACO, and at the same time propose a meta-heuristic approach to solve the SMTTP that can compete with leading heuristics. We tested different heuristic "desirabilities", such as Shortest Processing Time (SPT), Earliest Due Date (EDD), Modified Due Date (MDD) and Look-ahead-MDD (L-MDD). We used the most efficient heuristic information, MDD, for our computational study. Furthermore we tested different alternatives to improve the solution generated with ACO by local search: swapping of any two jobs and swapping of consecutive jobs analogous to the adjacent pair interchange (API) strategy. We made experiments on some initial design alternatives using a small example with 10 jobs from literature [23] and tested, analyzed and interpreted ACO-variants, local search alternatives and parameter settings on over 250 problem instances. In this paper we report on computational experiments on 125 benchmark problems of 100 jobs each.

The remainder of the paper is organized as follows: in the following section 2 the Single Machine Total Tardiness Problem is introduced in formal terms and a short overview on relevant literature and state of the art is given. In section 3 we briefly describe the Ant Colony Optimization metaphor and present the SMTTP adaption of an ACO-algorithm on a general basis (section 4). Finally, in section 5 algorithmic details and computational results are presented, before we conclude in section 6.

2 The Single Machine Total Tardiness Problem

2.1 The Problem

The Single Machine Total Tardiness Problem (SMTTP) considers n jobs to be processed without pre-emption on a single machine that can handle only one job at a time. Every job j $(j = 1, \ldots, n)$ is available for processing at time zero and requires a positive processing time p_j. Let further d_j, C_j and $T_j = max\{0,\ C_j - d_j\}$ be the due date, the completion time and the tardiness of job j, respectively. The objective of the SMTTP is to find a processing sequence out of the possible sequences (permutations of n jobs) including all jobs that minimizes the total tardiness $\sum_{j=1}^{n} T_j$. The SMTTP is a special case of the generalized total weighted tardiness problem (TWTP) where different weights are assigned to each job.

2.2 State of the Art

The SMTTP is NP-hard [17], therefore it is very difficult to solve large problems optimally. Optimization techniques that were applied to this problem are branch and bound as well as dynamic programming algorithms (enumerative algorithms were applied to the general total weighted tardiness problem). Emmons [18] developed several theorems and dominance rules that can be used to restrict the search effort of a branch and bound algorithm. Fisher [19] based his dual variable Lagrangian problem on Emmons dominance rules. Schrage and Baker [31], and Lawler [24] applied a dynamic programming approach to the SMTTP, whereas Potts and Van Wassenhove [27] combined Lawlers decomposition theorem with the approach of Schrage and Baker to implement an efficient algorithm. More recently, Szwarc and Mukhopadhyay [33], and Della Croce et al. [10] presented branch and bound procedures, that are based on findings of Emmons and Lawler.

For real-world applications, e.g. in a flexible manufacturing system (FMS) environment, heuristics are for obvious reasons, like computational effort, much more appropriate than optimization procedures. Wilkerson and Irwin [34] perform an adjacent pair interchange (API) to improve a feasible basic solution (WI heuristic). Fry et al. [20] advanced the WI heuristic by selecting the best of nine adjacent pair interchange strategies. Holsenback and Russell [23] developed a heuristic that is not limited to pairwise swaps; it is based on the net benefit of relocation (NBR) and uses also Emmons dominance theorems. Panwalkar et al. [26] developed the PSK heuristic that seems to be inferior as "the NBR heuristic, when properly coded, performs significantly better than the P-S-K heuristic" (cf. [29], p. 543).

3 Ant Colony Optimization

The behaviour of real ants searching for food was the impetus for a new search strategy to solve combinatorial optimization problems that is known as Ant Colony Optimization (ACO) [6, 7, 11, 12, 13, 16, 15]. Real ants are able to communicate information concerning food sources via an aromatic essence, called pheromone. Depending on the distance and the quality of a discovered food source they mark the path leading to that food source by laying down varying quantities of pheromone. Other ants observe the pheromone trail and are attracted to follow it. Thus, the path is marked again, reinforced and will attract even more ants to follow the trail. Paths leading to close, rich food sources will be more frequented and therefore the pheromone trails on such paths will grow faster. The described behavioural process of real ant colonies is the basis to solve combinatorial optimization problems using simulation with artificial ants: instead of searching their natural environment for food artificial ants search the solution space in order to generate high-quality solutions. The objective values correspond to the quality of the food sources and an adaptive memory is the analogy of the pheromone trail. To navigate through the set of feasible solutions artificial ants are able to use a local heuristic function[1].

In [22] a more formal representation of an ant algorithm is presented: agents (ants) perform random *walks* in a *construction graph* and these walks represent feasible solutions of the underlying combinatorial optimization problem. A walk consists of several "node-to-node" moves and these moves are performed on basis of *transition probabilities*.

For some combinatorial optimization problems like e.g. the traveling salesman problem (TSP), the corresponding construction graph is obvious or even identical to the underlying graph. As will be shown in the next section, this is not the case for the SMTTP.

4 Adapting ACO for SMTTP

The Ant Colony Optimization Meta-Heuristic has proven to find good solutions for NP-hard, combinatorial optimization problems, thus the application to the SMTTP seemed reasonable and promising. In a feasible solution of the SMTTP each job appears exactly once in the processing sequence, in an optimal solution the total tardiness has to be minimized.

An exact representation of the SMTTP leads to a construction graph with a high complexity (for n jobs the graph contains 2^n vertices and $2^n \cdot \frac{n}{2}$ edges), where each transition is represented by a unique arc [1].

To reduce the complexity of the construction graph some (historical) information is neglected and a move represents the decision to schedule job j as i-th job in the sequence, no matter which jobs have been scheduled previously. Thus, we use only a two dimensional matrix to represent the desirability to chose job j to be processed as the i-th job.

4.1 Solution generation

To construct a feasible solution the artificial ants successively choose jobs to be appended to the actual sub-sequence, until all jobs are scheduled. Each

[1] For a more detailed introduction of the Ant Colony Optimization metaphor the reader is referred to [12, 16].

ant decides independently which job j should be the i-th job in the sequence, i.e., each ant generates a complete solution. For this selection process, the ants use problem-specific, heuristic information, denoted by η_{ij}, as well as pheromone trails, denoted by τ_{ij}, where the former is an indicator on how good the choice of that job *seems to be*, and the latter indicates how good the choice of that job *was*[2]. The transition probability $\mathcal{P}_{ij}$ that job j is selected to be processed on position i is formally given by:

$$\mathcal{P}_{ij} = \begin{cases} \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{h \in \Omega} [\tau_{ih}]^\alpha [\eta_{ih}]^\beta} & \text{if } j \in \Omega \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where Ω is the set of non-scheduled jobs.

The most obvious heuristic information to be used is based on the Earliest Due Date (EDD) heuristic, which sorts and schedules jobs according to ascending due dates. In this case, the heuristic information would read as follows:

$$\text{EDD} \qquad \eta_{ij} = \frac{1}{d_j} \tag{2}$$

Another possibility[3] is the so-called Modified Due Date (MDD) heuristic, which is derived from Theorem 1 by Emmons [18]. Here, the jobs are iteratively scheduled. After a job is scheduled, all remaining non-scheduled jobs are again sorted in ascending order, but according to modified due dates. The job yielding the lowest modified due date is appended to the sub-sequence generated so far. The modified due dates are given by $max\{\mathcal{T} + p_j,\ d_j\}$, where $\mathcal{T}$ is the total processing time of all jobs already scheduled. Thus, for an ant colony optimization approach, this advanced heuristic value can formally be written as:

$$\text{MDD} \qquad \eta_{ij} = \frac{1}{max\{\mathcal{T} + p_j,\ d_j\}} \tag{3}$$

4.2 Trail update

After all ants have followed the selection process described above and thus constructed a complete solution to the problem, the pheromone trails are

[2] These two matrices are only two dimensional as a consequence of the reduction in complexity.

[3] Further priority rules like shortest processing time (SPT) or Look-ahead MDD (L-MDD) that can be used as heuristic information are discussed in [1].

globally updated. Depending on solution quality, i.e., depending on total tardiness, the corresponding pheromone trails are increased. At the same time evaporation reduces the pheromone trails. This global trail update can formally be expressed as follows:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \rho\Delta\tau_{ij}(t) \tag{4}$$

where $\rho \in [0,1]$ is a parameter that controls the pheromone decay, and $\Delta\tau_{ij}$ is amount of pheromone trail added to τ_{ij} by the ants. As for the heuristic information, there are several possibilities regarding the quantity $\Delta\tau_{ij}$ in ACO algorithms. In early versions [16], each ant contributed to the trail update, whereas later versions suggested to rank ants according to quality (AS_{rank}) [3], or to restrict the update to the best ant (ACS) [14], possibly combined with an upper and lower bound for the trail levels ($\mathcal{MM}AS$) [32]. In this paper we followed the ant colony system (ACS) idea, i.e., only the best ant contributes to the pheromone trail update. Thus, we have $\Delta\tau_{ij}(t) = 1/T^*$ for all edges (i,j) belonging to the best solution found so far, where T^* is the total tardiness of that best solution.

Besides the global update described above, there is also a local trail update in ACS. After an artificial ant has selected a job to be appended to the existing sub-sequence, the corresponding pheromone trail is updated as follows:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \rho\tau_0 \tag{5}$$

with the initial trail intensity

$$\tau_0 = \frac{1}{n \cdot T_{EDD}}$$

where T_{EDD} is the total tardiness for a processing sequence generated according to the EDD rule.

4.3 Local Search

Ant Colony Optimization algorithms have been shown to work very well if combined with a local search procedure [4, 5, 14, 32]. In such a case, a local search procedure is applied to the solutions generated by the artificial ants before updating the pheromone trails.

For our ant approach to the SMTTP a pairwise swap was considered. The pairwise swap is a 2-opt-like strategy that may either be applied to the set of all jobs scheduled or, alternatively, only to adjacent jobs which have a predecessor-successor-relation. The latter is based on the Adjacent Pairwise

Interchange Strategy (API) of Wilkerson and Irwin [34] and is less costly regarding computational effort than the former.

Concerning the local search procedure, it is possible to use one of the two strategies described above, (1) to all sequences generated during an iteration or (2) to the best sequence of the current iteration (iteration-best-method). Again, the alternative that seems to be more promising (1) is also the more expensive alternative in terms of computation time. We will address this topic in the next section, when we present our computational results.

5 Results

The computational tests we present in this section were performed on 125 benchmark problems[4] from Potts and Van Wassenhove [28], which were generated as follows: for each job j ($j = 1, \ldots, n$) an integer processing time p_j from the uniform distribution [1, 100] and an integer due date d_j from the uniform distribution

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

are generated. Here, the hardness of a problem is determined by the tardiness factor, denoted by TF and the relative range of due dates, denoted by RDD. The values for TF and RDD are taken from the set {0.2, 0.4, 0.6, 0.8, 1.0}. The problem set contains five instances of size $n = 100$ for each combination of TF and RDD, i.e., 125 instances in total[5].

As mentioned in section 4.2 we implemented ACS, a more recent ACO algorithm that uses a somewhat different transition rule as opposed to the general one presented in equation (1) in section 4.1. Here an artificial ant selects job j to be processed on position i as follows:

$$j = \begin{cases} \arg\max_{h \in \Omega} \left\{ [\tau_{ih}]^{\alpha} [\eta_{ih}]^{\beta} \right\} & \text{if } q \leq q_0 \\ J & \text{if } q > q_0 \end{cases} \tag{6}$$

where $q_0 \in [0, 1]$ is a tunable parameter, q is a random variable, uniformly distributed over [0,1], and J is a job selected according to (1). In other words, for $q \leq q_0$ the "best" job in terms of trail and desirability is *exploited*, whereas for $q > q_0$, a "good" job is *explored*.

[4] We would like to thank H.A.J. Crauwels for providing the benchmark problems as well as their optimal solutions.

[5] We applied ACO to test data sets with smaller problem sizes, too, obtaining results of comparable quality.

In our tests we used MDD for the heuristic information η_{ij}. We had 20 artificial ants and the following parameter settings: $\alpha = 1$, $\beta = 2$, $\rho = 0.1$ and $q_0 = 0.9$. Preliminary tests showed that applying the "pairwise swap of all jobs" local search procedure to each solution generated by an ant yields the best results, but is computationally very expensive. For that reason we used the less costly iteration-best variant, i.e., the local search procedure was applied only to the best solution of each iteration.

Table 1 gives the results generated by our ant algorithm in detail. We

		ACO (best run)		ACO (5 runs)	
TF	RDD	Number of optimal solutions	Mean relative deviation	Number of optimal solutions	Mean relative deviation
0.2	0.2	5	0.00%	25	0.0000%
0.2	0.4	5	0.00%	24	0.0026%
0.2	0.6	5	0.00%	24	0.0007%
0.2	0.8	5	0.00%	23	0.0006%
0.2	1.0	5	0.00%	25	0.0000%
0.4	0.2	5	0.00%	25	0.0000%
0.4	0.4	5	0.00%	23	0.0032%
0.4	0.6	5	0.00%	23	0.0086%
0.4	0.8	5	0.00%	25	0.0000%
0.4	1.0	5	0.00%	25	0.0000%
0.6	0.2	5	0.00%	25	0.0000%
0.6	0.4	5	0.00%	25	0.0000%
0.6	0.6	5	0.00%	24	0.0012%
0.6	0.8	5	0.00%	25	0.0000%
0.6	1.0	5	0.00%	25	0.0000%
0.8	0.2	5	0.00%	25	0.0000%
0.8	0.4	5	0.00%	25	0.0000%
0.8	0.6	5	0.00%	25	0.0000%
0.8	0.8	5	0.00%	25	0.0000%
0.8	1.0	5	0.00%	25	0.0000%
1.0	0.2	5	0.00%	25	0.0000%
1.0	0.4	5	0.00%	25	0.0000%
1.0	0.6	5	0.00%	25	0.0000%
1.0	0.8	5	0.00%	25	0.0000%
1.0	1.0	5	0.00%	25	0.0000%
Ø		125/125	0.00	616/625	0.0007

Table 1: Results of ACO for SMTTP ($n = 100$ jobs)

report the best solution found as well as the result obtained for five ACO test runs for each instance, leading to 625 runs in total. Our ant algorithm generated optimal solutions for all 125 problem instances. The 616 optimal solutions (out of 625) and the low average deviation of 0.0007% may be interpreted as an indicator of the robustness of the approach.

Table 2 shows a comparison of results generated with several leading problem-specific heuristics[6]. From [28] we took the results for the following procedures:

- Wilkerson-Irwin (WI)
- four decomposition heuristics: DEC/EDD, DEC/WI/S, DEC/WI/D, DEC/RS500
- two descent methods: DES and DESO
- two simulated annealing approaches: SALM and SAPW

and from [29] we took the results for NBR, modified NBR (M-NBR), PSK and composite M-NBR/PSK[7].

For each procedure the size of the data set, the number of optimal solutions found and the mean relative deviation from the optimum is given[8]. Russell and Holsenback [29] performed their experiments on a reduced set of 85 instances[9] containing 100 jobs each. Only our ACO algorithm was able to solve all 125 test instances optimally. The M-NBR/PSK composite heuristic, which applies M-NBR as well as PSK and selects the better of the two solutions obtained, performs comparably well in terms of deviation, but solves only 31 out of 85 instances optimally. Regarding the number of optimal solutions DEC/WI/D (98 out of 125) comes closest, but is still outperformed by our ant algorithm.

A simulated annealing approach was developed by Ben-Daya and Al-Fawzan [2] and applied to selected problem instances (100 jobs) with a tardiness factor TF=0.6 and a range of due dates RDD $\in \{0.4, 0.8\}$. Their approach yielded an average deviation of 0.75% only (cf. [2], p. 65) whereas ACO has an average deviation of 0.00%.

[6]The results presented were not obtained for identical problem instances, but all problem instances were generated using the method by Potts and Van Wassenhove [28], which we briefly described above.

[7]See [28] and [29] for details concerning these procedures.

[8]We do not report actual computation times as our implementation was not meant for high-speed computing but for maximum flexibility. Based on our experience we estimate computation times of 10s for an efficient PC-implementation. This means that our ACO algorithm is not as fast as the other heuristics, but this drawback is more than compensated by the better results.

[9]For TF=1.0 no instances at all, and for TF=0.2 only 10 instances with RDD $\in \{0.2, 0.4\}$ were generated.

Procedure	Data set size	Number of optimal solutions	Mean relative deviation
WI[a]	125	54	1.28%
DEC/EDD[a]	125	44	4.03%
DEC/WI/S[a]	125	61	0.59%
DEC/WI/D[a]	125	98	0.16%
DEC/RS500[a]	125	78	0.15%
DES[a]	125	43	0.19%
DESO[a]	125	80	0.20%
SALM[a]	125	23	0.84%
SAPW[a]	125	37	0.73%
NBR[b]	85	16	1.91%
M-NBR[b]	85	24	1.21%
PSK[b]	85	22	2.04%
M-NBR/PSK[b]	85	31	0.0075%
ACO	125	125	0.0000%

[a] Potts and Van Wassenhove (1991)
[b] Russel and Holsenback (1997a)

Table 2: Comparison of ACO with leading heuristics

6 Conclusion

Ant Colony Optimization (ACO) has shown good results in various applications, e.g. traveling salesman, vehicle routing, graph coloring, quadratic assignment or job shop. We developed an ACO algorithm to solve a special machine scheduling problem, the Single Machine Total Tardiness Problem. Integrating a problem-specific procedure, the Modified Due Date (MDD), as the heuristic criteria and performing a pairwise swap as a local search procedure yielded very good results for large problem instances. Our approach outperformed all leading heuristics significantly.

References

[1] Bauer, A. (1998): *Ant Colony Optimization for the Single Machine Total Tardiness Problem*, unpublished Master Thesis (in german), Department of Management Science, University of Vienna, Austria.

[2] Ben-Daya, M. and M. Al-Fawzan (1996): A Simulated Annealing Approach for the One-Machine Mean Tardiness Scheduling Problem, *European Journal of Operational Research* **93** pp. 61-67.

[3] Bullnheimer, B., R.F. Hartl and C. Strauss (1997): *A New Rank Based Version of the Ant System - A Computational Study*, Working Paper No. 1, SFB Adaptive Information Systems and Modelling in Economics and Management Science, to appear in *CEJOR*.

[4] Bullnheimer, B., R.F. Hartl and C. Strauss (1997): *An Improved Ant System Algorithm for the Vehicle Routing Problem*, POM Working Paper No. 10/97, University of Vienna, to appear in *Annals of Operations Research*.

[5] Bullnheimer, B., R.F. Hartl and C. Strauss (1998): *Applying the Ant System to the Vehicle Routing Problem*, in: S. Voss, S. Martello, I. H. Osman and C. Roucairol, eds, Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization, Kluwer, Boston, pp. 109-120.

[6] Colorni, A., M. Dorigo and V. Maniezzo (1991): *Distributed Optimization by Ant Colonies*, Proceedings of European Conference on Artificial Life ECAL'91, Paris, France, Elsevier Publishing, pp. 134-142.

[7] Colorni, A., M. Dorigo and V. Maniezzo (1992): *Investigation of Some Properties of an Ant Algorithm*, Proceedings of Parallel Problem Solving from Nature, Brussels, Elsevier, pp. 509-520.

[8] Colorni, A., M. Dorigo, V. Maniezzo and M. Trubian (1994): Ant System for Job-Shop Scheduling, *JORBEL - Belgian Journal of Operations Research, Statistics and Computer Science* **34** (1) pp. 39-53.

[9] Costa, D. and A. Hertz (1997): Ants can Colour Graphs, *Journal of the Operational Research Society* **48** pp. 295-305.

[10] Della Croce, F., R. Tadei, P. Baracco and A. Grosso (1998): A New Decomposition Approach for the Single Machine Total Tardiness Scheduling Problem, *Journal of the Operational Research Society* **49** pp. 1101-1106.

[11] Dorigo, M. (1992): *Optimization, Learning and Natural Algorithms*, Ph.D. Dissertation (in italian), Politecnico di Milano, Milano, Italy.

[12] Dorigo, M. and G. Di Caro (1999): *The Ant Colony Optimization Meta-Heuristic*, in: D. Corne, M. Dorigo and F. Glover, eds, New Ideas in Optimization, McGraw-Hill.

[13] Dorigo M., G. Di Caro and L.M. Gambardella (1998): *Ant Algorithms for Discrete Optimization*, Technical Report IRIDIA/98-10, Universite Libre de Bruxelles, Belgium. to appear in *Artificial Life*.

[14] Dorigo, M. and L.M. Gambardella (1997): Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem, *IEEE Transactions on Evolutionary Computation* **1** pp. 53-66.

[15] Dorigo, M., V. Maniezzo and A. Colorni (1991): *Positive Feedback as a Search Strategy*, Technical Report 91-016, Dip. Elettronica, Politecnico di Milano, Italy.

[16] Dorigo, M., V. Maniezzo and A. Colorni (1996): Ant System: Optimization by a Colony of Cooperating Agents, *IEEE Transactions on Systems, Man, and Cybernetics* **26** (1) pp. 29-41.

[17] Du, J. and J.Y.-T. Leung (1990): Minimizing Total Tardiness on One Machine is NP-Hard, *Mathematics of Operations Research* **15** pp. 483-495.

[18] Emmons, H. (1969): One Machine Sequencing to Minimize Certain Functions of Tardiness, *Operations Research* **17** pp. 701-715.

[19] Fisher, M.L. (1976): A Dual Algorithm for the One-Machine Scheduling Problem, *Mathematical Programming* **11** pp. 229-251.

[20] Fry, T.D., L. Vicens, K. Macleod and S. Fernandez (1989): A Heuristic Solution Procedure to Minimize $\bar{T}$ on a Single Machine, *Journal of the Operational Research Society* **40** pp. 293-297.

[21] Gambardella, L.M. and M. Dorigo (1997): *HAS-SOP: An Hybrid Ant System for the Sequential Ordering Problem*, Technical Report IDSIA/11-97, IDSIA, Lugano, Switzerland.

[22] Gutjahr, W.J. (1998): *A Generalized Ant System and its Convergence*, Technical Report 98-10, Department of Statistics, Operations Research and Computer Science, University of Vienna, Austria.

[23] Holsenback, J.E. and R.M. Russell (1992): A Heuristic Algorithm for Sequencing on One Machine to Minimize Total Tardiness *Journal of the Operational Research Society* **43** pp. 53-62.

[24] Lawler, E.L. (1977): A Pseudopolynomial Algorithm for Sequencing Jobs to Minimize Total Tardiness *Annals of Discrete Mathematics* **1** pp. 331-342.

[25] Maniezzo, V., A. Colorni and M. Dorigo (1994): *The Ant System applied to the Quadratic Assignment Problem*, Technical Report No. 94-28, IRIDIA, Brussels, Belgium.

[26] Panwalkar, S.S., M.L. Smith and C.P. Koulamas (1993): A Heuristic for the Single Machine Tardiness Problem, *European Journal of Operational Research* **70** pp. 304-310.

[27] Potts, C.N. and L.N. Van Wassenhove (1982): A Decomposition Algorithm for the Single Machine Tardiness Problem, *Operations Research Letters* **32** pp. 177-181.

[28] Potts, C.N. and L.N. Van Wassenhove (1991): Single Machine Tardiness Sequencing Heuristics, *IIE Transactions* **23** pp. 346-354.

[29] Russell, R.M. and J.E. Holsenback (1997a): Evaluation of Greedy, Myopic and Less-Greedy Heuristics for the Single Machine, Total Tardiness Problem, *Journal of the Operational Research Society* **48** pp. 640-646.

[30] Russell, R.M. and J.E. Holsenback (1997b): Evaluation of leading heuristics for the single machine tardiness problem, *European Journal of Operational Research* **96** pp. 538-545.

[31] Schrage, L.E. and K.R. Baker (1978): Dynamic Programming Solution of Sequencing Problems with Precedence Constraints, *Operations Research* **26** pp. 444-449.

[32] Stützle, T. and H. Hoos (1997): *The $\mathcal{MAX}$-$\mathcal{MIN}$ - Ant System and Local Search for the Traveling Salesman Problem*, Proceedings of IEEE 4th International Conference on Evolutionary Computation ICEC'97, IEEE Press, pp. 308-313.

[33] Szwarc, W. and S. Mukhopadhyay (1996): Decomposition of the Single Machine Total Tardiness Problem, *Operations Research Letters* **19** pp. 243-250.

[34] Wilkerson, L.J, and Irwin, J.D. (1971): An Improved Algorithm for Scheduling Independent Tasks, *AIIE Transactions* **3** pp. 333-342.