An Ant Colony Optimization Application to the Single Machine Total Weighted Tardiness Problem

Matthijs den Besten[1], Thomas Stützle[2] and Marco Dorigo[3]

[1] *University of Amsterdam, Department of Computer Science, Kruislaan 403, NL-1098 SJ Amsterdam, The Netherlands*
mdbesten@wins.uva.nl

[2] *Darmstadt University of Technology, Computer Science Department, Alexanderstr. 10, 64283 Darmstadt, Germany*
stuetzle@informatik.tu-darmstadt.de

[3] *IRIDIA, Université Libre de Bruxelles, Avenue Franklin Roosevelt 50, CP 194/6, B-1050 Brussels, Belgium*
mdorigo@ulb.ac.be

Abstract

In this extended abstract we present an algorithm based on the Ant Colony Optimization (ACO) metaheuristic for the single machine total weighted tardiness problem, a well known NP–hard scheduling problem. Our ACO algorithm is currently among the best algorithms known for this problem type. In particular, we will discuss three elements that enable it to find very good solutions quickly. These are a powerful local search algorithm, the use of candidate lists that guide the ants' solution construction, and a heterogeneous ant colony where ants apply various local search variants concurrently.

1 Introduction

The single machine total weighted tardiness problem (SMTWTP) is an NP–hard scheduling problem [9] for which instances with more than 50 jobs often can not be solved to optimality with state–of–the–art branch and bound algorithms [1]. Hence, to solve large problem instances one needs approximation algorithms which obtain near-optimal solutions in short time. Recently, algorithms falling into the framework of the Ant Colony Optimization (ACO) metaheuristic have shown very promising performance on a large number of combinatorial optimization problems; for some of them they are among the top performing algorithms (see [6, 7] for an overview). In this abstract we extend the list of successful applications of ACO algorithms to include the SMTWTP. In the following, we first introduce the problem, then we explain our algorithm in depth and give some computational results. We conclude with some final remarks and indicate directions to extend our work.

2 The problem

In the SMTWTP n jobs have to be processed on a single machine. Associated to each job j is a processing time p_j, a due date d_j and a weight w_j. The jobs are all available for processing right from the start. The tardiness of job j is $T_j = \max\{0, C_j - d_j\}$ where C_j is the completion time of job j in the current job sequence; the total weighted tardiness is given by $\sum_{i=1}^{n} w_i T_i$. The goal of the SMTWTP is to find a job sequence which minimizes the sum of the weighted tardiness.

Because the applicability of exact algorithms is limited to small problems, much research has focused on approximation algorithms [13, 4]. The best performance to date has been reported for *iterated dynasearch*

[3]. This algorithm employs a special kind of local search using techniques adapted from dynamic programming which is applied inside an iterated local search algorithm.

3 The algorithm

Our ACO algorithm is an adaptation of Ant Colony System (ACS) [8] for the SMTWTP. A feasible solution for the SMTWTP, also called a *sequence*, consists of a permutation of the jobs. When applied to the SMTWTP, each ant starts with an empty sequence and then iteratively appends an unscheduled job to the partial sequence constructed so far. The heuristic information the ants use in addition to pheromone is either the jobs' apparent urgency [12] or it is based on the modified due date rule which was used in an earlier application of ACO [2] to the *unweighted* single machine total tardiness problem (the unweighted version is much easier to solve than the weighted one). Local search is applied to all solutions the ants construct in each iteration. To achieve best possible performance we found that local search, candidate lists, and heterogeneous ants play an important role as explained in the following.

3.1 Effective local search

Local search for the SMTWTP starts from some initial sequence and repeatedly tries to improve the current sequence with neighboring solutions. The local search applied by ACS-SMTWTP is a concatenation of a local search in the *interchange* neighborhood that considers exchanges of jobs placed at the ith and jth position ($i \neq j$) and a local search in the *insert* neighborhood that uses removals of the job at the ith position and inserts it in the jth position ($i \neq j$). The procedure has two variants, *interchange+insert* and *insert+interchange*, where the naming denotes the order in which the neighborhoods are searched. The idea of changing the neighborhood during the search is systematically applied by the Variable Neighborhood Search (VNS) metaheuristic [11]. We found that our VNS local search is very effective for the SMTWTP and is in large part responsible for the very good performance of our algorithm.

3.2 Candidate list

When closer examining the solution construction, we noted that if job j is not put on the position i for which desirability $\tau_{ij}(t) \cdot {\eta_{ij}}^{\beta}$ is maximal, often it is put at one of the last positions in the sequence. Because the absolute position of a job is important, we introduced a candidate list [8, 14] (another technique to alleviate this problem has been introduced in [10]). In particular, the candidate list is defined dynamically during the solution construction and comprises the first still not sequenced *cand* jobs of the global–best solution π^*. The next job is then chosen among those of the candidate list in the case of a probabilistic choice of the next job to be appended. We found that with $cand = 20$ a reasonably good performance is obtained.

3.3 Heterogeneity

We noticed an instance dependence of the appropriate choice for the local search algorithm. We addressed this problem considering an ACS algorithm using a heterogeneous colony of ants in which half of the ants apply *interchange+insert* and the other half apply *insert+interchange* local search. We found that a heterogeneous colony performed better than the worse of the two homogeneous colonies and on some instances even outperformed the best homogeneous colony [5].

Table 1: The table reports some basic statistics on the distribution of the average computation times to solve instances of the three problem sets. Given are the number of jobs, the average time (averaged over the 125 instances for each test set) to solve the benchmark set (t_{avg}) and the standard deviation (σ_t), the average time to solve the easiest and the hardest instance (t_{min} and t_{max}, respectively), and the quantils of the average time to solve a given percentage of the instances. Q_x indicates the average time to solve x% of the benchmark instances.

No. jobs	t_{avg}	σ_t	t_{min}	t_{max}	Q_{10}	Q_{25}	Q_{50}	Q_{75}	Q_{90}
40	0.088	0.021	0.004	1.72	0.004	0.008	0.013	0.122	0.215
50	0.32	0.869	0.006	10.74	0.007	0.016	0.031	0.353	0.606
100	6.99	7.019	0.018	86.26	0.028	0.070	3.190	7.499	19.30

4 Results

We tested ACS-SMTWTP on the SMTWTP benchmark instances from ORLIB available at http://www.ms.ic.ac.uk/info.html which consists of three sets of 125 instances with 40, 50, and 100 jobs, respectively. For the SMTWTP it is known that, depending on the parameter settings in the random generation of instances, some of the instances are rather easily solved while others present a much larger challenge to approximation algorithms. This is reflected in the fact that by a simple application of our local search algorithm for the 100-job instances we could find in about 45 cases the best known solution values. Therefore, we felt that the only real challenge would be to find the best known solutions on *all* instances. To account for this fact, we run ACS-SMTWTP 25 times on each instance and we set the maximal CPU time to 1200 seconds on a Pentium III, 450MHz processor, which was enough for ACS-SMTWTP to find the best known solutions for every instance in each single run. In Table 1 we give indicative statistics on the average time taken by ACS-SMTWTP to solve the instances.

ACS-SMTWTP appears to perform significantly better than most other previously proposed algorithms for the SMTWTP. For example, ACS-SMTWTP always finds the best-known solutions for the 100 job instances, while the best performing tabu search algorithm in [4] could only find 103 of the best-known solutions at that time. The only algorithm reaching a performance similar to that obtained with ACS-SMTWTP appears to be iterated dynasearch [3].

5 Conclusion

In this extended abstract we have presented ACS-SMTWTP, an effective ACO algorithm for the single machine total weighted tardiness problem. Our algorithm could find, for all known benchmark instances available in ORLIB, the optimal or best-known solutions within reasonable computation times. There are several reasons for the very good performance of the ACO approach. Among them are the use of a very effective local search algorithm, a heterogeneous colony of ants which increases the robustness of the algorithm and the application of candidate lists in the solution construction. In future work we will test our algorithm on larger SMTWTP instances and will extend it to a wider range of single-machine scheduling problems. Additionally, our results using a heterogeneous colony of ants suggest that a similar approach may improve ACO algorithms in other applications where a strong instance dependence of the best algorithm configuration has been noted.

References

[1] T. S. Abdul-Razaq, C. N. Potts, and L. N. Van Wassenhove. A survey of algorithms for the single machine total weighted tardiness scheduling problem. *Discrete Applied Mathematics*, 26:235–253, 1990.

[2] A. Bauer, B. Bullnheimer, R. F. Hartl, and C. Strauss. An ant colony optimization approach for the single machine total tardiness problem. In *Proceedings of the 1999 Congress on Evolutionary Computation (CEC'99)*, pages 1445–1450. IEEE Press, 1999.

[3] R. K. Congram, C. N. Potts, and S. L. Van de Velde. An iterated dynasearch algorithm for the single–machine total weighted tardiness scheduling problem. Technical report, Faculty of Mathematical Studies, University of Southampton, Southampton, UK, December 1998.

[4] H. A. J. Crauwels, C. N. Potts, and L. N. Van Wassenhove. Local search heuristics for the single machine total weighted tardiness scheduling problem. *INFORMS Journal on Computing*, 10(3):341–350, 1998.

[5] M. den Besten, T. Stützle, and M. Dorigo. Scheduling single machines by ants. Technical Report IRIDIA/99-16, Université Libre de Bruxelles, Belgium, 1999.

[6] M. Dorigo and G. Di Caro. The Ant Colony Optimization meta-heuristic. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*, pages 11–32. McGraw Hill, London, UK, 1999.

[7] M. Dorigo, G. Di Caro, and L. M. Gambardella. Ant algorithms for discrete optimization. *Artificial Life*, 5(2):137–172, 1999.

[8] M. Dorigo and L. M. Gambardella. Ant Colony System: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66, 1997.

[9] J. K. Lenstra, A. H. G. Rinnooy Kan, and P. Bruckner. Complexity of machine scheduling problems. In P. L. Hammer, E. L. Johnson, B. H. Korte, and G. L. Nemhauser, editors, *Studies in Integer Programming*, volume 1 of *Annals of Discrete Mathematics*, pages 343–362. North-Holland, Amsterdam, NL, 1977.

[10] D. Merkle and M. Middendorf. An ant algorithm with a new pheromone evaluation rule for total tardiness problems. In S. Cagnoni et al., editor, *Real-World Applications of Evolutionary Computing*, volume 1803 of *LNCS*, pages 287–296. Springer, Berlin, 2000.

[11] N. Mladenović and P. Hansen. Variable neighborhood search. *Computers & Operations Research*, 24:1097–1100, 1997.

[12] T. E. Morton, R. M. Rachamadugu, and A. Vepsalainen. Accurate myopic heuristics for tardiness scheduling. GSIA Working Paper 36-84, Carnegie–Mellon University, PA, 1984.

[13] C. N. Potts and L. N. Van Wassenhove. Single machine tardiness sequencing heuristics. *IIE Transactions*, 23:346–354, 1991.

[14] T. Stützle. An ant approach to the flow shop problem. In *Proceedings of the 6th European Congress on Intelligent Techniques & Soft Computing (EUFIT'98)*, volume 3, pages 1560–1564. Verlag Mainz, Aachen, 1998.