AN ANT COLONY APPROACH TO RESOURCE-CONSTRAINED PROJECT SCHEDULING PROBLEMS

Yun-Chia Liang*, Angela H.L. Chen**, Wen-Ching Kao* and Chiuh-Cheng Chyu*

**Department of Industrial Engineering and Management, Yuan Ze University*

No 135 Yuan-Tung Road, Chung-Li, Taoyuan County, Taiwan 320, R.O.C.

***Department of Business Administration, Nanya Institute of Technology*

No 414 Sec 3 Chung-Shan E Road, Chung-Li, Taoyuan County, Taiwan 320, R.O.C.

**ycliang@saturn.yzu.edu.tw*

ABSTRACT

Much research effort has been drawn on the resource-constrained project scheduling problems (RCPSP), a well-known NP-hard problem, in the past. The success of applying ACO method to other scheduling problems has inspired our study to extend similar investigation to the RCPSP with single or mixed resource requirements. Hence, this paper proposes an ant colony optimization (ACO) algorithm for the resource-constrained project scheduling problems (RCPSP). The activity on network (AON) is considered and the forward-parallel method is used for the activity selection. The ACT (maximum ACTim values first) rule is employed in solution construction procedure. Online and offline pheromone updating are applied to enhance the diversification and intensification of the search respectively. In addition, the proposed algorithm is tested on a well-known suite of benchmark problems from the project scheduling library (PSPLIB). Computational results on test problems with 120 jobs show that the ant colony optimization method provides competitive solution quality with moderate computational expense in comparison with other heuristics in the literature.

Key Words: Combinatorial Optimisation, Project Scheduling.

1. INTRODUCTION

There has been a tremendous increase in research of methods and optimal solutions for the RCPSP, from standard project scheduling methods such as critical path method (CPM) and program evaluation and review technique (PERT), to different kinds of heuristics (e.g. priority-based methods), till recent metaheuristics like simulated annealing, genetic algorithm, and tabu search. Meanwhile traditional project scheduling methods arrange activities based on the assumption of unlimited capacity of resources, Such assumption lacks its aptness in practice. Hence, modern approaches have incorporated more realistic limitation of the resources' availabilities; for example, resources are renewable or non-renewable, and are limited on total project basis or on per-period basis. Without surprise, research topics surrounding the project scheduling problems in general become ever more challenging just as what Garey and Johnson [1979] have proven the computational complexity of the RCPSP is an NP-hard type.

Techniques to solve RCPSP can be broadly classified into three categories: exact methods, priority rules, and meta-heuristic methods. The examples of the exact methods include integer programming (IP) [Wiest 1963], Branch-and-Bound (B&B) [Davis and Heidorn 1971,

Dorndorf *et al.* 2000], and dynamic programming (DP) [Khamooshi 1999, Webster and Azizoglu 2001]. These exact methods guarantee to obtain the optimum yet limited to the small and moderate problem size. Then, the examples of priority rules include minimum slack first (SLK), maximum slack first (MASK), minimum latest finish time (LFT), smallest resource demand (SRD), maximum actim value first (ACT), etc. Priority rules can generate acceptable results within a reasonable time. Survey and comparison of priority rules can be seen in [Chiu and Tsai 1993, Kolisch 1996, Klein 2000]. Finally, the examples of Meta-heuristic methods consist of genetic algorithm (GA) [Leon and Ramamoorthy 1995, Hartmann 1998, 1999], hybrid fuzzy and GA [Kim *et al.* 2003], tabu search (TS) [Baar *et al.* 1998], simulated annealing (SA) [Bouleimen and Lecocq 1998], and ant colony optimisation [Merkle *et al.* 2002]. Particularly, the implementation of an ant system algorithm for RCPSP in Merkle *et al.* [2002] paper involved both serial and parallel ants to generate schedules, and applied a 2-opt local search to improve the solutions. Test results of their algorithm have shown excellent performance on a set of well-known benchmark problems. More detailed literature surveys are provided by [Brucker *et al.* 1999, Hartmann and Kolisch 2000].

From the time when Marco Dorigo first introduced the Ant System (AS), the earliest version of the ant colony optimisation (ACO) methods, in his dissertation in 1992 [Dorigo 1992], ACO has been successfully applied to different optimisation problems including continuous and combinatorial problems such as travelling salesperson (TSP) [Dorigo *et al.* 1996, Dorigo and Gammbardella 1997, Stützle and Hoos 1999], quadratic assignment (QAP) [Stützle and Hoos 1999, Maniezzo and Colorni 1999], vehicle routing (VRP) [Bullnheimer *et al.* 1999, Liang *et al.* 2003], redundancy allocation (RAP) [Liang and Smith 1999, 2004] and scheduling [Colorni *et al.* 1994, Stützle 1998, Bauer *et al.* 1999, Liang and Smith 2000, Merkle *et al.* 2002]. Since there hasn't been any dominant solution methodology in the RCPSP, the opportunity for ACO to be a competitive optimisation method exists. Furthermore, the RCPSP lends itself to appropriate ACO problem representations. Therefore, this paper's primary objective is to develop an ACO algorithm for the single-mode RCPSP, then to evaluate the performance of our proposed algorithm with the best-reported algorithms.

The content of this paper is organized as follows. Section 2 introduces an ant colony optimisation algorithm for RCPSP. Section 3 provides comparisons of test results on benchmark problems with other methods. Finally, section 4 brings to the conclusion of this research.

2. PROBLEM DEFINITION

This study considers the structure of the project as a so-called activity-on-node (AON) network where the nodes represent the activities, durations are node weights and the arcs denote the precedence relations between activities. The network is acyclic and numerically labelled. Preemption is not allowed. There are scarce renewable resources. The RCPSP is, thus, stated as follows: A single project consists of a set of activities where each activity has to be performed in one prescribed way (mode) using specified amounts of the resources provided for the completion of the project. The prescribed way (mode) refers as the activities' schedule, which must satisfy the precedence relations between activities and resource availability per time period. Furthermore, the mathematical model of the single-mode resource-constrained project-scheduling problem is formulated as follows:

$$Min \left[\underset{j \in J}{Max} \left(t_a \times x_{jt_a} \right) \right] \tag{1}$$

Subject to

$$\sum_{t_a = EF_j}^{LF_j} x_{jt_a} = 1 \qquad j = 1, \ldots, J \tag{2}$$

$$\sum_{t_a=EF_h}^{LF_h} t_a \times x_{ht_a} \le \sum_{t_a=EF_j}^{LF_j} (t_a - d_j) x_{jt_a} \qquad h \in P_j,\ j = 2,\ldots, J \qquad (3)$$

$$\sum_{j=1}^{J} k_{jr} \sum_{q=max\{t_a, EF_j\}}^{min\{t_a+d_j-1, EF_j\}} x_{jq} \le K_r \qquad r \in R,\ t_a = 1,\ldots, T^* \qquad (4)$$

$$x_{jt_a} \in \{0,1\} \qquad j=1,\ldots, J,\ t_a = EF_j,\ldots, LF_j \qquad (5)$$

Notation

J set of activities
j index for activities, $j = 1, 2,\ldots, J$
R set of renewable resources
R index for renewable resources, $r \in R$
ES_j earliest start time of activity j
LS_j latest start time of activity j
EF_j earliest finish time of activity j
LF_j latest finish time of activity j
T^* upper bound on the project's makespan
K_r number of units of renewable resource r, $r \in R$, available in time unit t_a, $t_a = 1,\ldots, T^*$
P_j set of immediate predecessors of activity j
d_j duration of activity j
k_{jr} number of units of renewable resource r, $r \in R$, used by activity j
$x_{jt_a} = \begin{cases} 1 & \text{if activity } j \text{ is completed at the end of period } t_a \\ 0 & \text{otherwise} \end{cases}$
n total number of activities
US the set of unscheduled activities that satisfies both precedence and resource constraints
η_l local heuristic information of activity l
τ_{il} pheromone trail intensity of activity l following its immediate predecessor i
P_{il} transition probability of activity l following its immediate predecessor i
α a parameter that controls the relative importance of pheromone trails
β a parameter that controls the relative importance of local heuristic information
q $\in [0,1]$, a uniformly generated random number
q_0 $\in [0,1]$, a parameter that controls the ratio of exploration and exploitation
ACT_j ACT value of activity j
$ACT_j^* = \dfrac{ACT_j}{\max_{j \in J}(ACT_j)}$, scaled ACT value of activity j
ρ $\in [0,1]$, a parameter that controls the persistence of pheromone trails
τ_0 initial pheromone trail intensity
Q a parameter that controls the magnitude of pheromone contribution
DP^* the makespan of the best ant
$\Delta\tau_{il}^e = \dfrac{Q}{DP^*}$, quantity of pheromone added to τ_{il} by the elitist ant

The above RCPSP model in Equation 1 forms the core problem in our study. Basically, while minimizing the project's makespan is its primarily objective, Equation 2 ensures that exactly one completion time is assigned to each activity, and the precedence relations between

activities are also guaranteed in Equation 3. Equation 4 describes that the capacity of the different types of renewable resources cannot be exceeded by the requirements of the scheduled activities per time unit. Then, Equation 5 defines the binary decision variables. In addition, the upper bound of project's makespan (T^*) is determined by summing up the maximum activity durations. Finally, given T^*, the earliest start time (*ES*) and the earliest finish time (*EF*) can be obtained via the forward pass in the critical path method (CPM); as well as, the latest start time (*LS*) and the latest finish time (*LF*) can be calculated using backward pass of the CPM. Thus, the actual finish time of activity *j* will fall within the range of $[EF_j, LF_j]$.

Some typical assumptions for RCPSP are considered as follows:

- Durations of activities are known and fixed.
- No pre-emption of activities is allowed.
- Precedence relations between the activities may exist.
- The quantities and the types of required resources for each activity are known and fixed.
- Both durations of activities and available resources are integer and indivisible.
- Activities can only be performed in a pre-determined mode, *i.e.*, single mode.

3. AN ANT COLONY OPTIMIZATION ALGORITHM

In an ACO algorithm, after setting parameter values and initialising pheromone trails, the ant colony constructs solutions by applying a state transition rule. Online and offline pheromone update rule are employed during each iteration, and the process continues until a stopping criterion is reached. The study applies the ACO-RCPSP algorithm follow this particular algorithmic scheme given below:

Set all parameters and initialize the pheromone trails
Loop
Sub-Loop
Generate the schedule based on the state transition rule
Apply the online pheromone update rule
Continue until all ants have been generated
Evaluate all solutions during the iteration and record the best one
Apply the offline pheromone update rule
Continue until the stopping criterion is reached

3.1. State Transition Rule

Each ant generates a complete feasible solution, *i.e.*, a project schedule. The parallel schedule generation method is employed in this research. That is, to construct a solution the artificial ants successively choose activities to be appended to the current sub-schedule starting from time unit of one, until all activities are scheduled. For activity selection, the ants use local heuristic information, denoted by η_l, as well as pheromone trails, denoted by τ_{il}. The former is an indicator of how good the choice of an activity seems to be, and the latter indicates how good the choice of an activity was. In order to balance the exploitation of good solutions and the exploration of the search space, the state transition rule shown below is used for the solution construction process where activity *v* is selected to be performed following its immediate predecessor *i* in the schedule.

$$v = \begin{cases} \arg \max_{l \in US} [(\tau_{il})^{\alpha} (\eta_l)^{\beta}] & q \le q_0 \\ V & q > q_0 \end{cases} \quad (6)$$

and V is selected according to the transition probability given by

$$P_{iv} = \begin{cases} \dfrac{(\tau_{iv})^{\alpha} (\eta_v)^{\beta}}{\sum_{l \in US} (\tau_{il})^{\alpha} (\eta_l)^{\beta}} & v \in US \\ 0 & Otherwise \end{cases} \quad (7)$$

where α and β are parameters that control the relative weight of pheromone and local heuristic, respectively, *US* refers to the set of unscheduled activities satisfying both precedence and resource requirements, q is a random number uniformly generated between 0 and 1, and q_0 is a parameter which determines the relative importance of exploitation versus exploration. When $q \le q_0$ an exploitation of the knowledge available about the problem (the local heuristic knowledge about the choice of activities) and the learned knowledge memorized in the form of pheromone trails are used, whereas $q > q_0$ favours more (random) exploration.

Chiu and Tsai [1993] evaluated the performances of different heuristics. Their experimental results concluded the maximum ACTim value first (ACT) rule outperforms other 7 heuristics in single-project makespan minimization problem. Therefore, the ACT rule is adopted as the local heuristic in this research. The ACT value of activity *j* (ACT_j) is determined by calculating the sum of durations in the longest path from the activity *j* to the last activity. Then the scaled ACT value (ACT_j^*) of activity *j* is obtained as shown in Equation 8. The activity with higher ACT^* value has better chance to be selected by ants.

$$\eta_j = ACT^*_j = \frac{ACT_j}{\max_{j \in J}(ACT_j)} \quad (8)$$

3.2. Pheromone Updating

The pheromone update rule consists of two phases – online updating and offline updating. The purpose of online updating is to decay the pheromone intensity of the selected move to encourage exploration. Online updating is after an ant constructs a solution by

$$\tau_{il}^{new} = \rho \cdot \tau_{il}^{old} + (1 - \rho) \cdot \tau_o \quad (9)$$

where $\rho \in [0,1]$ is a parameter that controls the pheromone persistence, *i.e.*, $1-\rho$ represents the proportion of the pheromone evaporated. τ_0 represents the initial pheromone intensities.

After all ants have followed the selection process described above and thus constructed a complete schedule, the pheromone trails are updated offline. Depending on solution quality, *i.e.*, depending on makespan, the corresponding pheromone trails are increased. At the same

time evaporation reduces the pheromone trails. The offline trail update can formally be expressed as follows:

$$\tau_{il}^{new} = \rho \cdot \tau_{il}^{old} + (1-\rho) \cdot \Delta\tau_{il}^{e} \qquad (10)$$

where $\Delta\tau_{il}^{e}$ is the amount of pheromone trail added to τ_{il} by the elitist ants. That is $\Delta\tau_{il}^{e} = \frac{Q}{DP^{*}}$ for all combinations (*i*, *l*) belonging to the best solution found so far, where DP^{*} is the project makespan of the best ant so far, and *Q* denotes a constant that controls the magnitude of the pheromone contribution.

4. COMPUTATIONAL RESULTS

Kolisch and Sprecher [1996] proposed sets of resource-constrained scheduling benchmark problems. For single-mode RCPSP, the PSPLIB provides test problems with four different sizes (*i.e.*, 30, 60, 90, and 120 activities). According to the number of resources each activity uses, these problems can be further divided into single resource problem (SRP) and mix resource problem (MRP). That is activities in SRP need only one resource to perform while in MRP multi-resources are required. SRP instances of 120 and MRP instances of 360 are generated for all 30, 60, and 90-activity problems. The number of instances for 120-activity problem is 150 on SRP and 450 on MRP. Table 1 summarizes the number of instances for each problem sizes in this study. The optimal solutions to 30-activity problems are provided. The upper and lower bounds on project's makespan are also provided for other problem sizes. Both files and optimal solutions (or the upper/lower bounds) of the test problems can be downloaded from the Project Scheduling Library (PSPLIB) web site (URL: http://www.bwl.uni-kiel.de/Prod/psplib/datasm.html).

Table 1. List of Test Problem Groups

	Problem Size (number of activities)			
Type of Instances	**30**	**60**	**90**	**120**
SRP	120	120	120	150
MRP	360	360	360	450

The ACO-RCPSP is coded in Visual C++ 6.0 and run using an Intel Pentium IV 2.4GHz PC with 512 MB RAM. All computations use real float point precision without rounding or truncating values. The parameters in this ACO-RCPSP algorithm from the preliminary research is set up as follows: $\alpha = 1$, $\beta = 0.5$, $q_0 = 0.1$, and $\rho = 0.95$ for SRP, and $\alpha = 0.5$, $\beta = 2$, $q_0 = 0.7$, and $\rho = 0.95$ for MRP. ACT is chosen as the local heuristic, online updating is used, and the offline updating allowing only the best ant to contribute is considered. Total *n*/5 ants are allowed in each iteration. For example, in 30-activity problems, the population size is 6 per run; thus, 12 per run for 60-activity problem, 18 for 90, and 24 for 120. The algorithms terminate when (this criterion is determined based on the trade-off between the computational expense and the solution quality) the best ant has not changed for consecutive 10 iterations. Each instance runs ten times based on different random number seeds.

4.1 Comparison with Pure ACT

To verify the performance of ACO-RCPSP algorithm, the study conducts a comparison with the suggested best priority rule - Pure ACT [Chiu and Tsai 1993], and the conclusion is

depicted in this section. The performance measure is calculated by [(ACO-RCPSP or ACT – LB) / (LB)] × 100% where LB denotes the lower bound provided in the PSPLIB and for 30-activity instances LB is replaced by the optimum. The percentages on the performance measures of both algorithms are tabulated (see Table 2). Our results have indicated that ACO-RCPSP outperforms pure ACT in all cases. It concludes that a global optimiser of ACO-RCPSP using ACT as a local heuristic indeed improves the performance of local optimiser ACT.

Table 2. Comparison between Pure ACT and ACO-RCPSP

Type of Instances	30 activities		60 activities		90 activities		120 activities	
	ACT	ACO-RCPSP	ACT	ACO-RCPSP	ACT	ACO-RCPSP	ACT	ACO-RCPSP
SRP	20.08%	11.43%	43.36%	22.61%	52.95%	30.21%	63.82%	35.94%
MRP	28.40%	17.62%	43.94%	25.15%	54.03%	29.77%	55.67%	36.23%

4.2 Comparison with AS-RCPSP

We compare the results of ACO-RCPSP with the results of AS-RCPSP by [Merkle *et al.* 2002]. Our proposed ACO-RCPSP algorithm is different from AS-RCPSP in several areas such as its population size, sequencing method, search direction, stopping criterion, etc. For example, ACO-RCPSP uses $n/5$ (*i.e.*, 24 in 120-activity) forward ants to construct solutions in parallel while AS-RCPSP employs two types of ants – forward and backward – with 5 ants each to construct the schedules in serial or parallel methods. AS-RCPSP uses 2-opt local search to improve the solution quality, and the total number of solutions visited are 5000 as the stopping criterion. ACO-RCPSP does not apply the local search in consideration of the computational expense and employs a stopping criterion to check the convergence of the best ant performance. Table 3 shows the detailed difference between both algorithms.

Table 3. Differences between ACO-RCPSP and AS-RCPSP Structures (for 120-activity Problem)

Item	ACO-RCPSP	AS-RCPSP
Population size	24	10
Sequencing method	Parallel	Serial or Parallel
Search direction	Forward	Forward and Backward
Local heuristic information	ACT	nLST, nLFT, etc.
Local search	n/a	2-opt
Stopping criteria	Best ants has not changed for 10 iterations	Total number of 5000 solutions

Besides AS-RCPSP, Merkle *et al.* also developed a simpler version s-AS-RCPSP. The difference is that only forward ants are used in this s-AS-RCPSP. The comparison of these three ACO algorithms is listed in Table 4. Deviations from the lower bound among average, best, and worst performances are provided. Average performance of ACO-RCPSP is slightly worse than AS-RCPSP, but the best performance of ACO-RCPSP outperforms AS-RCPSP by about 4%. While compared to s-AS-RCPSP, ACO-RCPSP is an obvious winner in both average and the best categories, and performs comparably in the worst cases. The average number of solutions searched in ACO-RCPSP is 460 (a population size of 20 ants with approximately 23 iterations) that are only 1/10 of the ones in AS-RCPSP and its variations.

Based on the above comparisons, ACO-RCPSP shows competitive performance in a very effectively computational way.

Table 4. Comparison between ACO-RCPSP, AS-RCPSP, and s-AS-RCPSP

Performance Measures	ACO-RCPSP		AS-RCPSP	s-AS-RCPSP	
	SRP	MRP		serial	parallel
Average deviation from the lower bound	35.94%	36.23%	35.43%	37.1%	38.0%
Standard deviation	0.116	0.164	0.046	n/a	n/a
Smallest deviation from the lower bound	31.26%	30.57%	35.35%	36.7%	37.6%
Largest deviation from the lower bound	40.48%	41.78%	35.50%	40.3%	38.6%

4.3 Comparison with Other Meta-Heuristics

ACO-RCPSP is further compared with other well-known meta-heuristics in the literature. Five genetic algorithms of [Leon and Ramamoorthy 1995], [Hartmann 1998], and [Hartmann 1999], a simulated annealing algorithm of [Bouleimen and Lecocq 1998], and an ant colony optimisation algorithm of [Merkle *et al.* 2002] are included in this study. All meta-heuristics are asked to construct and evaluate 5000 solutions except ACO-RCPSP (discussed in previous section). The performance of ACO-RCPSP is only slightly worse than AS-RCPSP and self-adapting GA, but outperforms rest of meta-heuristics. Based on the number and quality of solutions evaluated, ACO-RCPSP can be considered one of the most effective and efficient methods.

Table 5. Comparison between ACO-RCPSP and Other Heuristics

Algorithms	Authors/Publication year	Average deviation from the lower bound (%)
AS-RCSPSP	Merkle *et al.*/2002	35.43
Self-adapting GA	Hartmann/1999	35.60
ACO-RCPSP	This research	36.16*
GA1	Hartmann/1998	36.74
SA	Bouleimen and Lecocq/1998	37.68
GA2	Hartmann/1998	38.49
GA3	Leon and Ramamoorthy/1995	40.69
GA4	Hartmann/1998	42.25

Note: the average deviation from the lower bound (%) of ACO-RCPSP considers both SRP and MRP results in Table 4.

5. CONCLUSION

Much research effort has been drawn on the resource-constrained project scheduling problems (RCPSP), a well-known NP-hard problem. Since ACO has been successfully applied to different optimisation problems and no dominant solution methodology in the RCPSP, this study aims to develop an ACO algorithm for the single-mode RCPSP, then to evaluate the performance of the developed algorithm with the best-reported algorithms. A modern meta-heuristic – ant colony optimisation algorithm is developed. Parallel and forward pass are employed to determine the schedule of project. The ACT rule, one of the most effective priority rules, is applied as the local heuristic information in the state transition rule. Both online and offline pheromone update rules are used to enhance the exploration and

exploitation of search space. Computational results show that the proposed algorithm (ACO-RCPSP) is superior to pure ACT rule in all problem sizes. While comparing to other meta-heuristics, ACO-RCPSP generates competitive performance considering the least number of solutions it evaluates. Compared to another ACO algorithm, AS-RCPSP, ACO-RCPSP outperforms AS-RCPSP in the best performance measure. In summary, ACO-RCPSP shows its promising potential in RCPSP, and it should be easily extended to other types of RCPSP such as multi-mode, multi-projects, or non-renewable resources.

REFERENCES

Baar, T., P. Brucker, and S. Knust (1998), "Tabu-Search Algorithms and Lower Bounds for the Resource-Constrained Project Scheduling Problem," In S. Voss, S. Martello, I. Osman, C. Roucairol (eds.), *Meta-Heuristics*: *Advances and Trends in Local Search Paradigms for Optimization*, Kluwer Academic Publishers, Boston, 1-8.

Bauer, A., B. Bullnheimer, R. F. Hartl, and C. Strauss (1999), "Minimizing Total Tardiness on a Single Machine Using Ant Colony Optimization," *Proceedings of the 1999 Congress on Evolutionary Computation*, Washington, D.C., 1445–1450.

Bouleimen, K., and H. Lecocq (1998), "A New Efficient Simulated Annealing Algorithm for Resource-Constrained Project Scheduling Problem," Technical Report, Service de Robotique et Automatisation, Université de Liége, Liége, Belgium.

Brucker. P, A. Drexl, R. H. Möhring, K. Neumann, and E. Pesch (1999), "Resource-Constrained Project Scheduling: Notation, Classification, Models, and Methods," *European Journal of Operational Research*, **112.1,** 3-41.

Chiu, H.-N. and D.-M. Tsai (1993), "A Comparison of Single-Project and Multi-Project Approaches in Resource-Constrained Multi-Project Scheduling Problems," *Journal of the Chinese Institute of Industrial Engineers*, **10.3,** 171-179.

Colorni, A., M. Dorigo, V. Maniezzo, and M. Trubian (1994), "Ant System for Job-Shop Scheduling," *Belgium Journal of Operations Research, Statistics and Computer Science*, **34.1,** 39-53.

Davis, E. W. and G. E. Heidorn (1971), "An Algorithm for Optimal Project Scheduling under Multiple Constraints," *Management Science*, **7.1,** 803-816.

Dorigo, M. (1992), *Optimization, Learning and Natural Algorithms*, Ph.D. Thesis, Politecnico di Milano, Italy.

Dorigo, M. and L. M. Gammbardella (1997), "Ant Colony System: A Cooperative Learning Approach to the Travelling Salesman Problem," *IEEE Transactions on Evolutionary Computation*, **1.1,** 53-66.

Dorigo, M., V. Maniezzo, and A. Colorni (1996), "Ant System: Optimization by a Colony of Cooperating Agents," *IEEE Transactions on System, Man, and Cybernetics-Part B: Cybernetics*, **26.1,** 29-41.

Dorndorf, U., E. Pesch, and T. Phan Huy (2000), "A Branch and Bound Algorithm for the Resource-Constrained Project Scheduling Problem," *Mathematical Methods in Operations Research*, **52**, 413-439.

Garey, M. R. and D. S. Johnson (1979), *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, CA.

Hartmann, S. (1998), "A Competitive Genetic Algorithm for Resource-Constrained Project Scheduling," *Naval Research Logistics*, **45.7,** 733-750.

Hartmann, S. (1999), "Self-Adapting Genetic Algorithm with an Application to Project Scheduling," University of Kiel, Germany, Technical Report 506.

Hartmann, S., and R. Kolisch (2000), "Experimental Evaluation of State-of-the-Art Heuristics for the Resource-Constrained Project Scheduling Problem," *European Journal of Operational Research*, **127.2,** 394-407.

Khamooshi, H. (1999), "Dynamic Priority–Dynamic Programming Scheduling Method $(DP)^2SM$: A Dynamic Approach to Resource-Constraint Project Scheduling," *International Journal of Project Management*, **17.6,** 383-391.

Kim, K. W., M. Gen, and G. Yamazaki (2003), "Hybrid Genetic Algorithm with Fuzzy Logic for Resource-Constrained Project Scheduling," *Applied Soft Computing*, **2.3,** 174-188.

Klein, R. (2000), "Bidirectional Planning: Improving Priority Rule-Based Heuristics for Scheduling Resource-Constrained Projects," *European Journal of Operational Research*, **127.3,** 619-638.

Kolisch, R. (1996), "Efficient Priority Rules for the Resource-Constrained Project Scheduling Problem," *Journal of Operations Management*, **14.3,** 179-192.

Kolisch, R. and A. Sprecher (1996), "PSPLIB – A Project Scheduling Problem Library," *European Journal of Operational Research*, **96**, 205-216.

Leon, V. J., and B. Ramamoorthy (1995), "Strength and Adaptability of Problem Space Based Neighborhoods for Resource-Constrained Scheduling," *OR Spektrum*, **17**, 173-182.

Liang, Y.-C., M.-H. Lo, and A. H. L. Chen (2003), "An Ant Colony Approach for the Vehicle Routing Problem under Distance and Capacity Constraints," *Proceedings of the 8th Annual International Conference on Industrial Engineering - Theory, Applications and Practice*, November 10-12, Las Vegas, Nevada, 600-605.

Liang, Y.-C. and A. E. Smith (1999), "An Ant System Approach to Redundancy Allocation," *Proceedings of the 1999 Congress on Evolutionary Computation*, Washington D.C., July 1999, 1478-1484.

Liang, Y.-C. and A. E. Smith (2000), "Ant Colony Optimization for Constrained Combinatorial Problems," *Proceedings of the 5th Annual International Conference on Industrial Engineering - Theory, Applications and Practice*, Hsinchu, Taiwan, ID 296.

Liang, Y.-C. and A. E. Smith (2004), "An Ant Colony Optimization Algorithm for the Redundancy Allocation Problem (RAP)," *IEEE Transactions on Reliability*, **53.3**, 417-423.

Maniezzo, V. and A. Colorni (1999), "The Ant System Applied to the Quadratic Assignment Problem," *IEEE Transactions on Knowledge and Data Engineering*, **11.5,** 769-778.

Merkle, D., M. Middendorf and H. Schmeck (2002), "Ant Colony for Resource-Constrained Project Scheduling," *IEEE Transactions on Evolutionary Computation*, **6.4,** 333-346.

Stützle, T. (1998), "An Ant Approach to the Flow Shop Problem," *Proceedings of the 6th European Congress on Intelligent Techniques & Soft Computing*, Verlag Mainz, Aachen, 1560-1564.

Stützle, T. and H. H. Hoos (1999), "The MAX-MIN Ant System and Local Search for Combinatorial Optimization Problems," in S. Voss, S. Martello, I. H. Osman, and C. Roucairol (eds.), *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer, 313-329.

Wiest, J. D. (1963), *The Scheduling of Large Project with Limited Resource*, Unpublished Ph.D. Thesis, Carnegie Institute of Technology.

Webster, S. and M. Azizoglu (2001), "Dynamic Programming Algorithms for Scheduling Parallel Machines with Family Setup Time," *Computer and Operations Research*, **28**, 127-137.