

# Resource Constrained Project Scheduling Using Particle Swarm Optimization

Jiancheng Wang<sup>\*</sup>

Equipment Academy, Beijing 101416, China

<sup>\*</sup>Corresponding author

**Abstract**—Project scheduling is the significant technique to attain the lean management, and applications can be found in such fields as construction engineering, equipment support, software development, etc. An activity-on-arrow (AOA) version of the resource constrained project scheduling problem (RCPSP) with the objective of minimizing project duration, derived from the decision making of the equipment support task, is formulated as a combination optimization problem and solved using the priority-based particle swarm optimization (PSO). The activity priorities are represented by particle positions and a serial scheduling scheme (SSS) is utilized to transform particle-represented priorities to an active schedule according to the precedence and resource constraints so that the project duration corresponding to each particle can be evaluated. The framework of the PSO scheme for the RCPSB is developed. Simulation is provided so as to investigate the performance of the priority-based PSO approach for the RCPSB. The optimal solution to a benchmark instance is obtained and compared to that of genetic algorithm, consistency between them showing the effectiveness and efficiency of the particle swarm optimization method.

**Keywords**—resource constrained; project duration optimization; equipment support; particle swarm optimization; network planning; activity-on-arrow network; serial scheduling scheme; local ring topology

## I. INTRODUCTION

Feature of resources is one of the characteristics of a project scheduling problem [1]. Traditional critical path method (CPM) and program evaluation and review technique (PERT) are usually adopted as the major tools to solve the project scheduling problems with the assumption of unlimited resource availability. However, this assumption may be impractical in many construction circumstances since only a fixed amount of resources are available or the cost of acquiring additional resources is very high [2]. How to schedule the activities in a task under constraint of limited resources belongs to resource constrained project schedule problem (RCPSP) [3,4]. Due to the limited resource units which can be assigned to activities of a project, the project makespan will most likely be greater than the duration that CPM has originally calculated, without taking into account the resource usage limit. Giving no or inadequate consideration of the resource usability and blindly treating RCPSP as the project schedule problem with no resource constraints will result in an unexpected ending. In the field of equipment support, for a decision-makers to schedule a task of equipment supporting with several kinds of limited resources, the activities must be scheduled under resource constrained

conditions such that the task will be completed within the scheduled task duration, which is advantageous in organizing and carrying out of equipment support tasks in line with the predetermined schedule [5].

Various solution-solving schemes such as exact, heuristic and metaheuristic methods have been developed for solving RCPSP since its advent over the past few decades [2,3,6–9].

Exact methods include the zero-one integer programming (IP), dynamic programming (DP), and implicit enumeration with branch-and-bound (B&B). The exact approaches may be computationally infeasible or face “combinatorial explosion” problem if the project under study is larger or more complicated. It has been proved that RCPSP is nondeterministic polynomial (NP)-hard [10] in the strong sense such that the computation time for obtaining the optimal solution using exact algorithms would be extremely long for problem even of medium scale. Therefore many heuristics have been developed.

Heuristic methods for the RCPSP are aimed at searching for optima in efficient ways. Though priority rules such as shortest activity duration (SAD), minimum late finish time (MILFT), minimum total float (MINTF), greatest resource demand (GRD), et al., can be adopted in heuristic methods, the performances of heuristics depend on problem characteristics and it is quite difficult to predict, beforehand, the most efficient heuristic for a given problem. Moreover, the heuristic methods may be trapped within local optima, and no priority rule dominates all others or performs consistently better than others.

As a new generation of heuristic algorithms, metaheuristic methods [11–15] normally include simulated annealing (SA), tabu search (TS), ant systems (AS), scatter search (SS), filter-and-fan (F&F) approach, genetic algorithm (GA), and particle swarm optimization (PSO). Of the metaheuristic methods, in addition to the advantages the metaheuristic methods have, including computational feasibility and effectiveness, PSO shows its uniqueness such as easy implementation and consistency in performance [7]. However, PSO has been seldom applied to solve the RCPSP diagramed particularly as an activity-on-arrow (AOA) network.

This work focuses on a particle swarm optimization based metaheuristic algorithm to solve the RCPSP, derived from the decision making of the equipment support task, which is represented as an AOA network.

The rest of this paper is organized as follows. The resource constrained project schedule problem is briefly formulated in

Section II. Section III is devoted to the basic components of PSO algorithms. Numerical simulation is given in Section IV to demonstrate the effectiveness of the algorithm. Finally, the conclusions are given in Section V.

## II. FORMULATION

Resource constrained project can be diagrammed with an activity-on-arrow network. The task is consisted of  $N$  activities in all including the dummy ones, their durations being zeros. The set of all activities of the project is denoted by  $A' = \{1, 2, \dots, n, \dots, N\}$ . The adjacent matrix of the project is denoted  $L$ , which is an upper right triangular matrix due to the fact that the AOA network is a directed acyclic diagram (DAG). The element of  $L$ ,  $l_{ij}$ , is 1 if there exists an edge or activity between nodes  $i$  and  $j$ , or 0 otherwise. Except for the one dimensional index expression of an activity in the project, a node pair index expression of an activity,  $(i, j)$ , is an alternate, and the set of all activities of this type expression is denoted by  $A^N = \{(i, j) | l_{ij} = 1; i = 1, 2, \dots, N-1; j = i+1, i+2, \dots, N\}$ . Between the two kinds of index expression, there exists a one-to-one mapping for an activity. Under the node pair index denotation,  $(i, j)$  denotes an activity between the pair of nodes  $i$  and  $j$ , and there exists no more than one activity between each pair of nodes. The sets of all immediate preceding and succeeding activities of an activity  $(i, j)$  is denoted by  $P_{(i,j)}$  and  $S_{(i,j)}$ , respectively.

For resource constrained project scheduling problem, activities use renewable resources that are available in limited capacities. Suppose the number of renewable resource types is  $K$ . While being processed, activity  $(i, j)$  requires  $r_{(i,j)}^{k,0}$  units of resource of type  $k = 1, 2, \dots, K$ , during each period of its non-preemptable duration  $d_{(i,j)}$ . Resource  $k$  has a limited availability of  $R_k$ , constant through the project execution. The parameters  $d_{(i,j)}$ ,  $r_{(i,j)}^{k,0}$ , and  $R_k$  are assumed to be nonnegative, deterministic, and integer.

The resource constrained project scheduling problem that considers the renewable resources, nonpreemption and minimizing of project duration is formulated as a discrete optimization problem as in (1), (2), and (3).

$$\min T_p = \{\max_{(i,j) \in A^N} t_{(i,j)}^{AF}\} \quad (1)$$

Subject to:

$$t_{(h,i)}^{AF} \leq t_{(i,j)}^{AF} - d_{(i,j)}, \forall (h,i) \in P_{(i,j)}; (i,j) \in A^N \quad (2)$$

$$\sum_{A_i} r_{(i,j)}^k(t) \leq R_k, k = 1, 2, \dots, K; t = t_{(i,j)}^{AS}; (i,j) \in A^N \quad (3)$$

In which,  $T_p$  is the project duration,  $t_{(i,j)}^{AS}$  and  $t_{(i,j)}^{AF}$ ,  $(i,j) \in A^N$ , are respectively the actual start and finish time of activity  $(i,j)$ ;  $A_t$  is the set of ongoing activities in period  $t$ , as defined in (4).

$$A_t = \{(i,j) | (i,j) \in A^N, t_{(i,j)}^{AF} - d_{(i,j)} + 1 \leq t \leq t_{(i,j)}^{AF}\} \quad (4)$$

And  $r_{(i,j)}^k(t)$  is the amount of resource  $k$  required by activity  $(i,j)$  at time instant  $t$ , as in (5).

$$r_{(i,j)}^k(t) = \begin{cases} r_{(i,j)}^{k,0}, & t_{(i,j)}^{AS} < t \leq t_{(i,j)}^{AS} + d_{(i,j)} \\ 0, & t \leq t_{(i,j)}^{AS} \text{ or } t > t_{(i,j)}^{AS} + d_{(i,j)} \end{cases} \quad (5)$$

The decision variable of the RCPSP is the actual finish time,  $t_{(i,j)}^{AF}$ ,  $(i,j) \in A$ , and the objective function (1) minimizes the makespan of the project. Constraints (2) take into consideration the precedence relations between activities  $(h,i)$  and  $(i,j)$ , where  $(h,i)$  immediately precedes  $(i,j)$ . Finally, constraint set (3) limits the total resource usage at the actual start time of activity  $(i,j)$  within the maximum available amount.

## III. PSO ALGORITHM

### A. Overall Framework

The framework of PSO algorithm is developed as in Algorithm 1. It is implemented in an iteration style. Initialization is done first, after which the iterations begin. Within a loop, each particle is transformed and evaluated and  $P_m$  is updated, which is a prerequisite to update  $P_{g(m)}$  using either the lbest ring model or the gbest model. Values of necessary variables are saved and terminating condition is checked to decide either to begin the next iteration or to end the whole iterations. If terminating condition is satisfied, necessary quantities such as the current best sequence, the corresponding makespan of the project, et al., are outputted; otherwise, the velocity and position are updated to proceed to next iteration of the optimization process.

ALGORITHM 1.	THE OVERALL FRAMEWORK OF PSO
initialize $P_m$	
initialize $P_{g(m)}$	
<b>for</b> each particle $m$ <b>do</b>	
<b>for</b> each dimension $n$ <b>do</b>	
initialize $x_{mn}$ with random positions	
initialize $v_{mn}$ with random velocities	
<b>end for</b>	
<b>end for</b>	
$t = 1$	
<b>for</b> each time step $t$ <b>do</b>	
<b>for</b> each particle $m$ in the swarm <b>do</b>	
transform each particle to an active schedule using SSS	
calculate project duration $T_p$ using (6)	
update $P_m$	
<b>end for</b>	
save average project duration of this iteration	
save shortest project duration of this iteration	
save the current best schedule so far, and the corresponding	
current shortest project duration so far	
check terminating condition and branch accordingly	
<b>for</b> each particle $m$ in the swarm <b>do</b>	
update $P_{g(m)}$ using lbest model	
<b>for</b> each dimension $n$ <b>do</b>	
update $v_{mn}$ & $x_{mn}$ using (9) and (10)	
<b>end for</b>	
<b>end for</b>	
$t = t + 1$	
<b>end for</b>	
output results	

### B. Particle Solution Representation

In general, there are two forms of solution representations: permutation-based (or activity list) representation and priority-based (or random key) representation, for the RCPSP. The latter form is adopted here. The value  $x_{mn}$  of the  $n$ th location of particle  $m$  stands for the priority of the activity with index  $n$ , with which it competes with its competitors in decision set to be scheduled as earlier as possible.

### C. Schedule Generation Using SSS

The serial scheduling scheme (SSS) is used to transform the particle-represented priorities to an active schedule based on precedence constraints and available resources from period to period so that the particle can be evaluated during searching for an optimal schedule.

### D. Determining Best Positions

For a particle in the swarm, by transforming using serial method, the actual finish times,  $t_{(i,j)}^{AF}$  of activity  $(i,j) \in A^N$ , is obtained, and the project duration corresponding to the particle,  $T_p$  is computed as in (6).

$$T_p = \{\max_{(i,j) \in A^N} t_{(i,j)}^{AF}\} \quad (6)$$

After all the particles at current iteration have been evaluated, the best position that the  $m$ th particle has individually found so far is as in (7).

$$P_m = \{p_{m1}, p_{m2}, \Lambda, p_{mn}, \Lambda, p_{mN}\} \quad (7)$$

And the best position found so far by the  $m$ th particle within its any neighbors using either the lbest ring model or the gbest model is as in (8).

$$P_{g(m)} = \{p_{g(m)1}, p_{g(m)2}, \Lambda, p_{g(m)n}, \Lambda, p_{g(m)N}\} \quad (8)$$

Both positions will be used in updating the velocity and the position of each particle in the swarm.

### E. Updating Mechanism

At each step of the PSO algorithm, the entire swarm evolves by updating the velocity and position of each particle  $m$  in every dimension  $n$  by the following rules [7,16].

$$\begin{aligned} v_{mn}(t+1) = & w(t+1)v_{mn}(t) \\ & + c_1 r_1 (p_{mn} - x_{mn}(t)) \\ & + c_2 r_2 (p_{g(m)n} - x_{mn}(t)) \end{aligned} \quad (9)$$

$$x_{mn}(t+1) = v_{mn}(t+1) + x_{mn}(t) \quad (10)$$

Where  $c_1$  and  $c_2$  are learning factors;  $r_1$  and  $r_2$  are random numbers between 0 and 1;  $w(t)$  is the inertia weight used to control the impact of the previous velocities on the current velocity, influencing the trade-off between the global and local experiences. The updated particles will be used for next iteration.

## IV. SIMULATIONS EXPERIMENT

In order to investigate the PSO-based method for the RCPSP, a benchmark instance is analyzed using the algorithm given in Section III, whose precedence relation of activities is diagramed as an AOA network as shown in Figure 1. The project considers three types of renewable resources and consists of 39 activities including 14 dummy activities.

The parameters  $P_{(i,j)}$ ,  $d_{(i,j)}$ , and  $r_{(i,j)}^{k,0}$ ,  $k = 1, 2, 3$ , are given in Table 1. The evolutionary parameters for PSO algorithm are  $w=1.0$ ,  $c_1=1.0$ ,  $c_2=1.0$ ,  $swarmSize=50$ ,  $maxGen=800$ , and the member number of the neighborhood in the lbest ring model is set as 2.

TABLE I. PARAMETERS OF ALL ACTIVITIES

$n$	$(i,j)$	$P_{(i,j)}$	$d_{(i,j)}$	$r_{(i,j)}^{1,0}$	$r_{(i,j)}^{2,0}$	$r_{(i,j)}^{3,0}$
1	(1,2)		0	0	0	0
2	(2,3)	1	5	3	5	2
3	(2,4)	1	5	5	4	3
4	(2,5)	1	3	5	2	2
5	(3,4)	2	0	0	0	0
6	(4,6)	3,5	4	4	1	4
7	(4,9)	3,5	2	2	4	4
8	(5,7)	4	1	5	5	4
9	(5,8)	4	6	3	5	2
10	(6,10)	6	0	0	0	0
11	(6,13)	6	1	4	1	4
12	(7,11)	8	0	0	0	0
13	(7,12)	8	3	3	3	2
14	(8,10)	9	0	0	0	0
15	(8,11)	9	0	0	0	0
16	(9,10)	7	0	0	0	0
17	(9,11)	7	0	0	0	0
18	(10,14)	10, 14, 16	6	3	2	2
19	(11,16)	12, 15, 17	3	3	2	4
20	(12,17)	13	4	1	5	4
21	(13,15)	11	6	2	2	2
22	(14,16)	18	0	0	0	0
23	(14,18)	18	3	5	5	4
24	(15,22)	21	3	3	2	3
25	(16,19)	19, 22	3	1	4	4
26	(17,18)	20	0	0	0	0
27	(17,23)	20	1	2	4	6
28	(18,21)	23, 26	4	4	5	4
29	(19,20)	25	3	5	3	3
30	(20,22)	29	0	0	0	0
31	(20,24)	29	4	1	0	4
32	(21,23)	28	0	0	0	0
33	(21,25)	28	1	2	2	1
34	(22,25)	24, 30	4	1	6	2
35	(23,25)	27, 32	6	3	2	1
36	(24,25)	31	0	0	0	0
37	(24,26)	31	3	2	2	2
38	(25,26)	33, 34, 35, 36	3	0	1	3
39	(26,27)	37, 38	0	0	0	0

The actual start time  $t_{(i,j)}^{AS}$  and the actual finish time  $t_{(i,j)}^{AF}$  of the optimum schedule  $S^*$  obtained using the PSO algorithm is listed in Table 2. The optimal duration of the project is 64,

which is the same as what was obtained using GA method [5]. The resource allocation profile of the optimum schedule  $S^*$  is shown in Figure 2. The evolution process of the algorithm is shown in Figure 3.

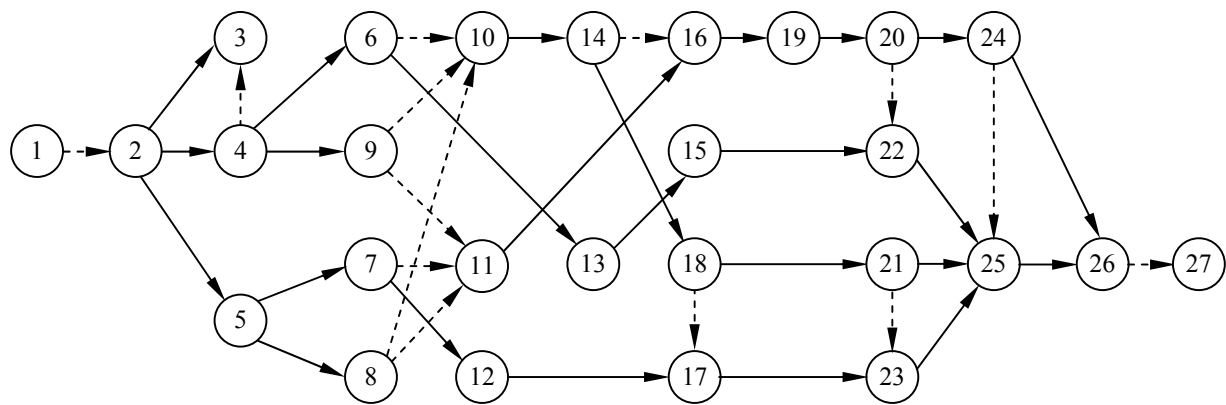
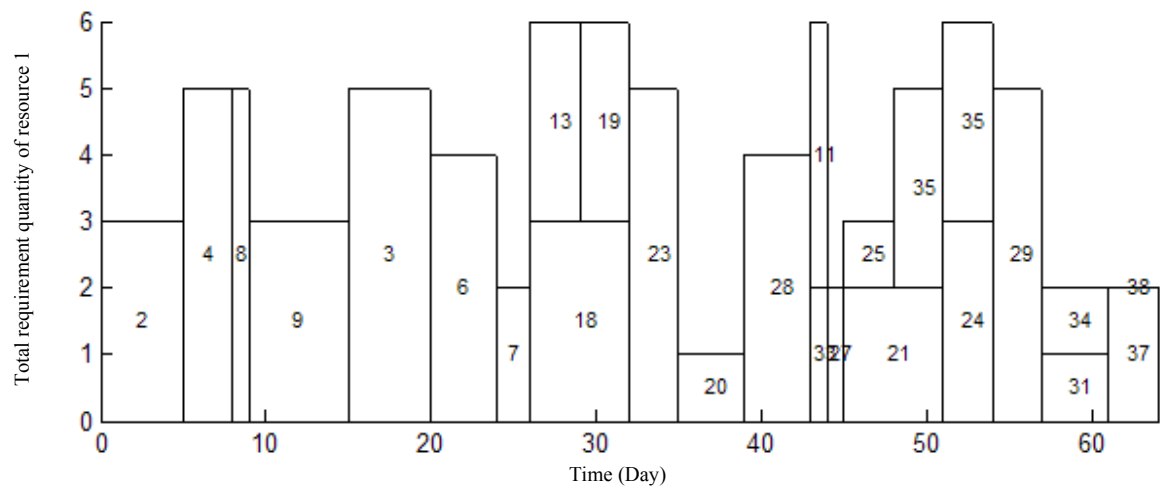


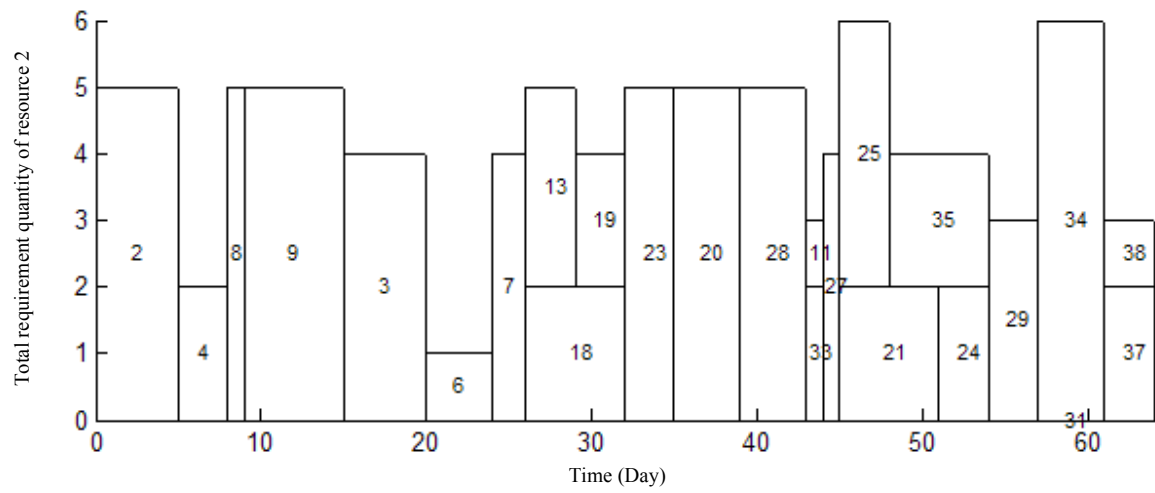
FIGURE I. AOA NETWORK OF THE BENCHMARK INSTANCE PROBLEM.

TABLE II. OPTIMUM SCHEDUL AND ITS ACTUAL START TIME AND ACTUAL FINISH TIME

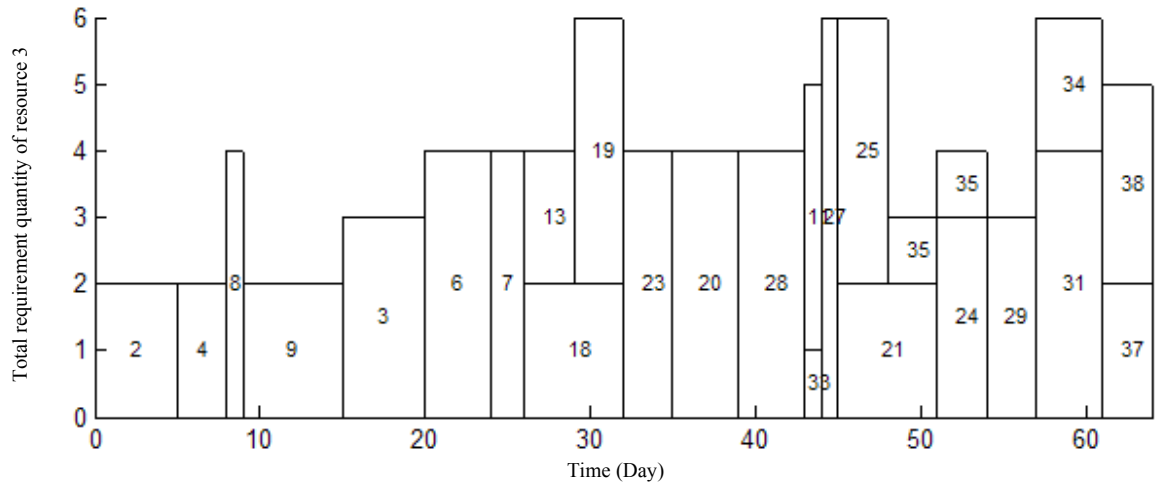
$n$	1	2	5	4	8	9	14	12	15	3	6	7	10	16	18	23	13	20	26	28	33	27	17	19	11	21	24	22	25	29	30	31	36	34	37	32	35	38	39
$t_{(i,j)}^{AS}$	0	0	5	5	8	9	15	9	15	15	20	24	24	26	26	32	26	35	39	39	43	44	26	29	43	45	51	32	45	54	57	57	61	57	61	43	48	61	64
$t_{(i,j)}^{AF}$	0	5	5	8	9	15	15	9	15	20	24	26	24	26	32	35	29	39	39	43	44	45	26	32	44	51	54	32	48	57	57	61	61	61	64	43	54	64	64



(a) Allocation profile of resource 1



(b) Allocation profile of resource 2



(c) Allocation profile of resource 3

FIGURE II. RESOURCE ALLOCATION PROFILES OF THE OPTIMUM SCHEDULE  $S^*$ .

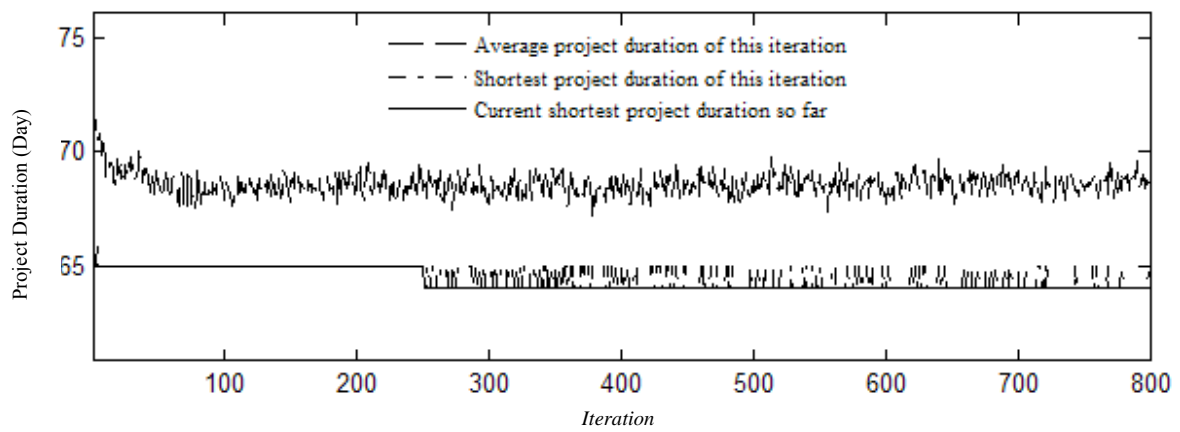


FIGURE III. EVOLUTION PROCESS OF THE PSO.

## V. CONCLUSIONS

The priority-based particle swarm optimization algorithm is utilized to solve the resource constrained project scheduling problem of activity-on-arrow version derived from the decision making of the equipment support task. Simulation result shows the effectiveness of the proposed algorithm for solution of instances with medium scale. However, different particle representation of particle swarm optimization may have different performance. Hence, future effort will be made to solve the RCPSP using permutation-based particle swarm optimization aiming at comparing the solution performance of priority-based PSO with that of the permutation-based one, respectively.

## REFERENCES

- [1] J. K. Lee and Y. D. Kim, "Search heuristics for resource constrained project scheduling," *Journal of the Operational Research Society*, vol. 47, pp. 678–689, May 1996.
- [2] H. Zhang, H. Li, and C. M. Tam, "Particle swarm optimization for resource-constrained project scheduling," *International Journal of Project Management*, vol. 24, pp. 83–92, January 2006.
- [3] E. W. DAVIS and J. H. Patterson, "A comparison of heuristic and optimum solutions in resource-constrained project scheduling," *Management Science*, vol. 21, pp. 944–955, April 1975.
- [4] R. Kolisch and S. Hartmann, "Experimental investigation of heuristics for resource-constrained project scheduling: An update," *European Journal of Operational Research*, vol. 174, pp. 23–37, October 2006.
- [5] J. C. Wang, M. Wei, and Y. P. Lü, "Resource constrained equipment support project duration optimization," in *Advances in Intelligent Systems Research*, vol. 117, Y. Y. Su, G. R. Chang, and Z. Luo, Eds. Paris: Atlantis Press, 2015, pp. 784–788.
- [6] R. Kolisch, "Serial and parallel resource-constrained project scheduling methods revisited: theory and computation," *European Journal of Operational Research*, vol. 90, pp. 320–333, April 1996.
- [7] H. Zhang, X. D. Li, H. Li, and F. L. Huang, "Particle swarm optimization-based schemes for resource-constrained project scheduling," *Automation in Construction*, vol. 14, pp. 393–404, June 2005.
- [8] F. F. Bector, "Some efficient multi-heuristic procedures for resource-constrained project scheduling," *European Journal of Operational Research*, vol. 49, pp. 3–13, November 1990.
- [9] P. Brucker, A. Drexl, R. Möhring, K. Neumann, and E. Pesch, "Resource-constrained project scheduling: Notation, classification, models, and methods," *European Journal of Operational Research*, vol. 112, pp. 3–41, January 1999.
- [10] J. Blazewicz, J. K. Lenstra, and A. H. G. Rinnooy Kan, "Scheduling subject to resource constraints: classification and complexity," *Discrete Applied Mathematics*, vol. 5, pp. 11–24, January 1983.
- [11] T. Baar, P. Brucker, and S. Knust, "Tabu-search algorithms and lower bounds for the resource-constrained project scheduling problem," in *Meta-heuristics: Advances and Trends in Local Search Paradigms for Optimization*, S. Voss, S. Martello, I. H. Osman, and C. Roucairol, Eds. Dordrecht: Kluwer Academic Publishers, 1998, pp. 1–8.
- [12] K. Bouleimen and H. Lecocq, "A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple modes version," *European Journal of Operational Research*, vol. 149, pp. 268–281, September 2003.
- [13] S. Hartmann, "A competitive genetic algorithm for resource-constrained project scheduling," *Naval Research Logistics*, vol. 45, pp. 733–750, October 1998.
- [14] E. Pinson, C. Prins, and F. Rullier, "Using tabu search for solving the resource-constrained project scheduling problem," in *Proceedings of the Fourth International Workshop on Project Management and Scheduling*, E. Demeulemeester and W. Herroelen, Eds. Leuven: Research Center for Operations Management, 1994, pp. 102–106.
- [15] S. E. Sampson and E. N. Weiss, "Local search techniques for the generalized resource-constrained project scheduling problem," *Naval Research Logistics*, vol. 40, pp. 665–675, August 1993.
- [16] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proceedings of IEEE Swarm Intelligence Symposium (SIS)*, O. Holland, J. Woods, R. D. Nardi, and A. Clark, Eds. Honolulu: IEEE Press, 2007, pp. 120–127.