Resource Constrained Equipment Support Project Duration Optimization

Wang Jiancheng
Equipment Academy
Beijing, China
e-mail: wjianch@sohu.com

Lü Yaping
96311 Troops
Huaihua, China
e-mail: lüyap@sohu.com

Wei Ming
AVIC Computing Technique Research Institute
Xi'an, China
e-mail: wm631@163.com

Abstract—Equipment support resources are the substantial prerequisites for decision-making in equipment support task. The renewable resources especially the scarce ones are usually limited in quantity. On account of the limited resource units that can be assigned to activities, the project completion time will most likely be greater than the duration that critical path method has originally calculated, without taking into account the resource usage limit. To ensure that the task will be completed within the scheduled task duration and to increase the feasibility of the decision making of an equipment support task, the task has to be scheduled under resource constraint, which belongs to resource constrained project schedule problem (RCPSP). Specific components such as initialization, crossover, mutation, and repairing operators, et al., of the genetic algorithms are designed to solve the problem. The efficiency of the adopted method is demonstrated by obtaining the optimal solutions to a benchmark instance with medium scale.

Keywords-resource constrained; project duration optimization; equipment support; genetic algorithm; network planning; activity on arc network; initialization; operator

I. INTRODUCTION

Equipment support resources are substantial conditions which affect not only the execution of equipment support task but also the equipment support decision in task scheduling. The resources can be classified either as renewable or non-renewable. Non-renewable resources are consumed by the activities while renewable resources are not consumed and keep the same stock along the entire lifetime of the project. However, the renewable resources especially the scarce ones such as storage space and manpower, et al., are usually limited in quantity. On account of the limited resource units that can be assigned to activities, the project completion time after resource allocation will most likely be greater than the duration that critical path method (CPM) has originally calculated, without taking into account the resource usage limit. It is obvious that an equipment support decision with inadequate consideration of the resource usability will result in an unexpected ending. For decision-makers in a task of equipment support, the activities of which must be scheduled under resource constrained conditions such that the project will be completed within the scheduled task duration, which is advantageous in organizing and carrying out of the equipment support task in line with the predetermined schedule. How to properly arrange the activities in a task under constraint of limited resources belongs to resource constrained project schedule problem (RCPSP) [1, 2]. The objective of resource constrained project schedule problem is to properly schedule dependent activities over time such that the task duration is minimized while the resource requirements have to be met.

Since its advent, RCPSP has been widely studied over the past few decades. The suggested algorithms can be classified into two categories: exact solution methods and heuristic algorithms.

Mingozzi et al. [3] and Brucker et al. [4] developed branch and bound algorithms, and Gavish and Pirkul [5] used dynamic programming. Blazewicz et al. proved that RCPSP is nondeterministic polynomial (NP)-hard in the strong sense such that the computation time for obtaining the optimal solution using exact algorithms can be extremely long for project even of medium scale. As a result, many procedures have been proposed in the past to overcome the combinatorial explosion problem. The approaches are grouped into priority rule-based methods, classical metaheuristics, non-standard metaheuristics, and other heuristics. Of the classical metaheuristics, there are genetic algorithms (GAs), tabu search (TS), simulated annealing (SA), and ant systems (AS), scatter search (SS), filter-and-fan approach [6-11]. For a detailed description of basic components of heuristics such as schedule generation schemes, priority rules, and representations, the reader is referred to [12-15].

This work is focused on a genetic based metaheuristic algorithm to handle the RCPSP in decision making of the equipment support task.

The paper is organized as follows: In Section II the resource constrained project schedule problem is briefly formulated, in section III the basic components of genetic algorithms are discussed. Effectiveness demonstration of the algorithm with a numerical example is given in Section IV. Section V concludes the work.

II. FORMULATION

An equipment support task can be modeled using an activity-on-arrow (AOA) network. The task is consisted of N activities in all including the dummy ones. The set of all activities of the project is denoted by $A = (a_1, \ldots, a_N)$, and the first and the last activity, a_1 and a_N, are, by convention, defined as dummy activities, i.e., activities that require zero execution time. The adjacent matrix of the activities is denoted X, which is an upper right triangular matrix due to the fact that the AOA network is a directed acyclic diagram (DAG). The element of X, x_{ij}, is 1 if there exists an edge or activity between node i and j, or 0 otherwise. Except for the symbol expression of an activity in the project, a node pair denotation of an activity in AOA network is an alternative. Under the node pair denotation, i–j denotes an activity between the node pair (i, j), and there is no more than one activity between each pair of nodes. The set of all activities of the project is alternatively denoted by $A=\{i–j|x_{ij}=1\}$. Sets of all immediate predecessor and successor activities of an activity i–j are denoted by $P_{i–j}$ and $S_{i–j}$, respectively. For resource constrained project scheduling problem (RCPSP), activities are related by two types of constraints. The first one is a precedence relation between i–j and $P_{i–j}$ which forces activity i–j not to be started before any one of its immediate predecessor activities $P_{i–j}$ has been finished. This type of constraint can be modeled using a precedence graph. The constraint of the second type is related to the renewable resource requirements. Activities use renewable resources that are provided in limited capacities. 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, …, K, during every time instant of its non-preemptable duration $d_{i–j}$. Resource type k has a limited capacity of R_k, constant through the project execution, which cannot be violated at any time period, i.e., the sum of resource usage of all ongoing activities A_t in time period t should not exceed R_k units of resource type k =1, 2, …, K. The parameters $d_{i–j}$, $r_{i–j}^{k,0}$, and R_k are assumed to be nonnegative, deterministic, and integer.

The decision variable in a schedule of RCPSP is the actual finish time of activity i–j, $t_{i–j}^{AF}$. The objective for RCPSP is to find a schedule of the task such that the task duration is minimized subject to precedence and resource constraints. The model is formulated as an optimization problem as in (1a) and (1b).

$$\text{Minimize} \quad T = \max\{t_{i–j}^{AF} \mid i – j \in A\} \tag{1a}$$

$$\text{Subject to:} \begin{cases} t_{i–j}^{AF} = t_{i–j}^{AS} + d_{i–j} \\ t_{i–j}^{AS} = \min\{t \mid t > \phi_{h–i}^{\max}, b_{t+1}^{k}, b_{t+2}^{k}, \cdots, b_{t+d_{i–j}}^{k} \ge r_{i–j}^{k}, k = 1,2,...,K\} \\ r_{i–j}^{k}(t) = \begin{cases} r_{i–j}^{k,0}, & t_{i–j}^{AS} < t \le t_{i–j}^{AS} + d_{i–j} \\ 0, & t \le t_{i–j}^{AS} \text{ or } t > t_{i–j}^{AS} + d_{i–j} \end{cases} \\ \phi_{h–i}^{\max} = \max\{t_{h–i}^{AF} \mid h – i \in P_{i–j}\} \end{cases} \tag{1b}$$

In the model, $t_{i–j}^{AS}$ and h–$i \in P_{i–j}$ are the actual start time and one of the immediate predecessor activities of activity i–j respectively, and b_t^k is the as-yet-unused resource units of type k at time point t.

III. GENETIC ALGORITHM

GA is essentially an iteration procedure that operates on sets of decision variables, and it start with expression of the chromosome.

A. Chromosome Expression

An indirect expression of a chromosome is adopted, in which an ordered activity list is a chromosome. Though an activity in AOA network can be named by either an integer or an edge between a node pair, we choose here to use the former for simplicity and ease of description. Thus, a chromosome is essentially composed of N integers, corresponding to N activities, and it stands for a possible order with which all activities of the task are to be scheduled. Suppose C is one of the chromosome in current generation, and c_p, p = 1, 2, …, N, is the activity in gene position p, the chromosome can then be expressed as follows.

$$C = [c_1, c_2, \ldots, c_p, \ldots, c_N]$$

Due to the utilization of indirect coding scheme, the chromosome could be precedence infeasible which results from the mutation operator. If it happens, the repairing operator is used to rectify the logic relationships among activities. As for when the activities in a precedence feasible chromosome will be scheduled to actually start, it is subject to the condition of resource constraint.

Once the chromosome structure is properly expressed, population can be initialized using the designed structure.

B. Initialization

Due to the existence of both precedence and resource constraints, the usually adopted random way to initialize the population will yield a considerable amount of infeasible schedules. To guarantee to generate precedence and resource feasible chromosome in initial population, a single-pass strategy is used alternatively. It fills a schedule from left to right with schedulable activities one at a time in a fixed order. The schedulable activities S_t are the set of activities each being as-yet-unscheduled, and the set of the immediate predecessors of each being a subset of the set of the as-yet-scheduled or partial scheduled activities PS_t. At any point in decision of selecting an activity from the set

of schedulable activities, it assigns priority among competing activities on a purely random basis. The set of the partial scheduled activities is updated whenever an activity is scheduled. The initialization is complemented in an iterative way until all activities have been scheduled. The procedure is shown below.

ALG-1: Initial Population Generation

```
begin
  ip←1
  while ip≤pop_size do
    ig←1
    PS_t←{1}
    while ig≤N−1 do
      S_t←{j|P_j ⊂ PS_t, j ∉ PS_t}
      j← ∀ i ∈ S_t
      PS_t←PS_t ∪ {j}
      ig←ig+1
    end while
    ip←ip+1
  end while
end
```

C. Evaluation

During each generation, the individuals in the current population are evaluated. To evaluate an individual, the chromosome is decoded first. When decoding, both the precedence and the resource constraints are enforced. Each activity is scheduled only if all of its predecessors are completed and its resource requirements are met. By decoding, the actual start time of activity i–j, t_{i-j}^{AS}, is obtained.

Since the RCPSP is formulated as a minimization problem, and the roulette wheel selection is a fitness-proportional selection, a transformation is utilized to map the natural objective function to a fitness function in order to ensure the fitter individual has a lower objective function value.

The transformation is implemented by either a relative fitness function as in (2) or an absolute fitness function as in (3).

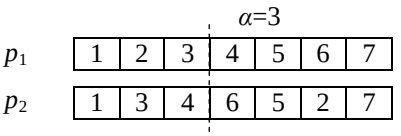
$$f(\boldsymbol{C}) = \frac{g_{\max} - g(\boldsymbol{C}) + \gamma}{g_{\max} - g_{\min} + \gamma} \tag{2}$$

$$f(\boldsymbol{C}) = G_{\max} - g(\boldsymbol{C}) \tag{3}$$

For small scale problem, both of them can be used to evaluate the chromosome, whereas the absolute one is preferred to evaluate chromosomes of medium scale problem or even those of large scale.

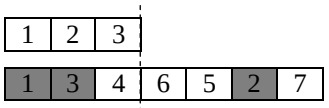
D. Crossover Operator

Due to elaborately designed initialization of population, all the chromosomes are precedence feasible. To prevent from interrupting the precedence feasibility of a schedule at the early stage, an efficient crossover operator is utilized to implement the crossover operation of two precedence feasible chromosomes. The remarkable merit of the technique is that the offspring is still precedence feasible after crossover. The operation is sketched in Fig. 1.

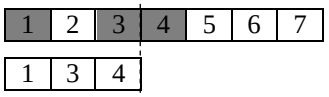


A random position is selected as α=3, for parents of a pair of chromosomes, therefore the crossover operation is executed between the third and the fourth gene positions of parents p_1 and p_2.

(a) Selecting crossover position

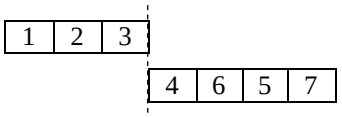


(i) Group p_2 and the first 3 genes of p_1 together. Shadowed genes in p_2 are those which are already included in the first 3 gene positions in p_1.

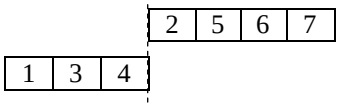


(ii) Group p_1 and the first 3 genes of p_2 together. Shadowed genes in p_1 are those which are already included in the first 3 gene positions in p_2.

(b) Grouping

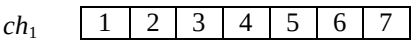


(i) In the first group, delete from p_2 the genes with shadows. From left to right, collect the genes left in p_2 into a chromosome block with consecutive gene positions and the index to starting gene position being 4.

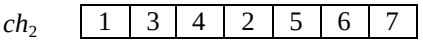


(ii) In the second group, delete from p_1 the genes with shadows. From left to right, collect the genes left in p_1 into a chromosome block with consecutive gene positions and the index to starting gene position being 4.

(c) Deleting genes with shadows



(i) Add the genes in chromosome block collected from p_2 to the tail of the first 3 gene positions of p_1 to get child ch_1.



(ii) Add the genes in chromosome block collected from p_1 to the tail of the first 3 gene positions of p_2 to get child ch_2.

(d) Connecting substrings to form offspring

Figure 1. Crossover operation.

E. Mutation Operator

Mutation operator is designed with the intensive search strategy. It is incorporated with the neighborhood technique to try to make a schedule to be λ-optimum which

means it is better than any other solutions in the neighborhood. Unlike in crossover operation, in mutation operation, a schedule may become infeasible, and the repairing operator is therefore needed to resolve the precedence conflict for the illegal neighbors.

F. Repairing Operator

Repairing operator is activated whenever a chromosome is checked as a precedence infeasible one. Given that an infeasible schedule $S_{NF} = [v_1, \ldots, v_j, \ldots, v_N]$ is detected and saved as S, and V_j is the set of the first j activities of the chromosome S, so a repairing operator is needed to make the chromosome S precedence feasible. Repairing operator is functioned by adjusting genes among positions to make sure that any activity in the set of the immediate successor activities of activity $g = v_j$, S_g, should not be placed before gene position j. In the end, the schedule S_{NF} is updated by S. The repairing process is shown in ALG-2.

ALG-2: Repairing of an Infeasible Schedule

begin
  $S \leftarrow S_{NF}$
  make v_1 and v_N feasible
  $j \leftarrow 2$
  while $j \le N-2$ **do**
    $V_j \leftarrow [v_1, v_2, \ldots, v_j]$
    $j \leftarrow j+1$
    $g \leftarrow v_j$
    $A_c \leftarrow V_{j-1} \cap S_g$
    $m \leftarrow |V_{j-1} \cap S_g|$
    if $m>0$ **then**
      move the same elements in S as those in A_c into any positions within $[j+1\ N]$
      $j \leftarrow j-m$
    end if
  end while
  $S_{NF} \leftarrow S$
end

IV. Computational Experiment

A benchmark instance [1] is solved using GA designed in section above. The algorithm is implemented in Matlab Language.

The AOA network of the problem is as shown in Fig. 2. The parameters P_{i-j}, d_{i-j}, and $r_{i-j}^{k,0}$, k = 1, 2, 3, are given in Table I. R_k = 6, k = 1, 2, 3. The evolutionary environment of the problem is that *popSize* = 50, *chromLen* = 39, P_c = 0.85, P_m = [0.55 0.35], *maxGen* = 30. The absolute fitness function is adopted. Activity *i–j*, 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 genetic algorithm is listed in Table II. The GA program is run randomly 10 times, and seven out of ten solutions have attained the exact ones of the problem, which shows the capability of the GA.

The evolution process of the problem is shown in Fig. 3. The optimum duration 64 is obtained in the 9th generation for the AOA network, which is the same as that of the AON network. It demonstrates that the proposed algorithm can find the known optimum rapidly.

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

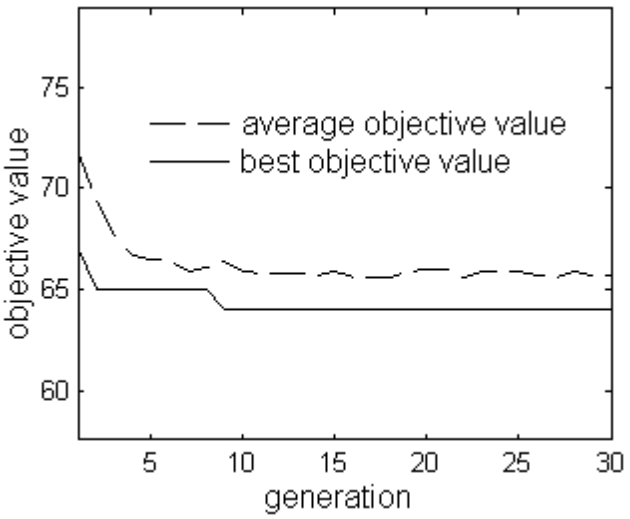


Figure 3. Evolution process of objective values vs. generations.

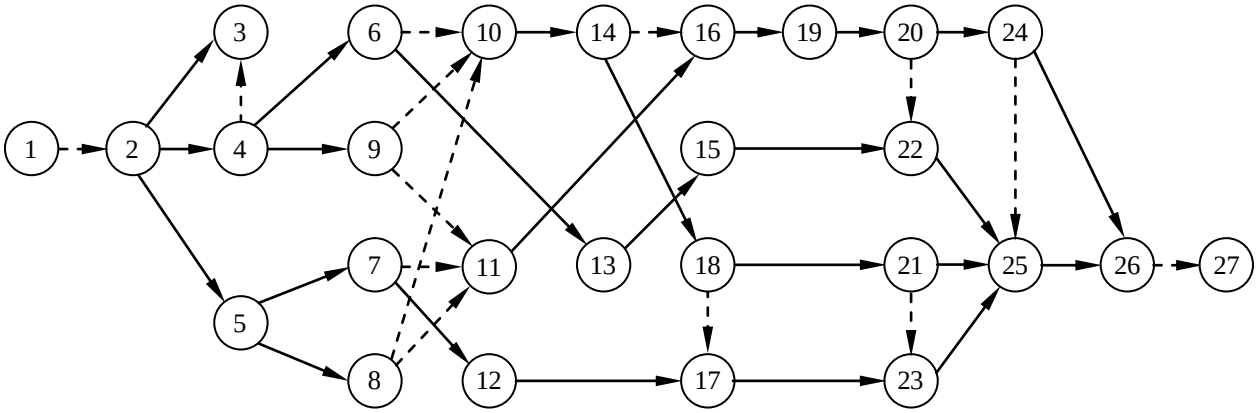


Figure 2. AOA network of an equipment support task.

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

$i–j$	1	3	4	2	8	12	9	14	5	15	6	10	7	17	16	18	23	22	13	20	27	26	28	33	11	19	21	24	32	25	35	29	31	36	30	34	37	38	39
t_{i-j}^{AS}	0	0	5	8	13	14	14	20	13	20	20	24	24	26	26	26	32	32	26	35	39	39	40	44	44	29	45	51	44	45	48	54	57	61	57	57	61	61	64
t_{i-j}^{AF}	0	5	8	13	14	14	20	20	13	20	24	24	26	26	26	32	35	32	29	39	40	39	44	45	45	32	51	54	44	48	54	57	61	61	57	61	64	64	64

V. CONCLUSIONS

Decision making of equipment support task with limited resources belongs to resource constrained project schedule problem. An especially designed genetic algorithm is utilized to find the optimum schedule. Numerical result shows the efficiency of the proposed algorithm for solution of problems with medium scale. Future efforts will be focused on incorporation of forward-backward improvement operator into this version to expand the ability of the proposed approach.

REFERENCES

[1] 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.

[2] 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, 2006.

[3] A. Mingozzi, V. Maniezzo, S. Ricciardelli, and L. Bianco, "An exact algorithm for the resource constrained project scheduling problem based on a new mathematical formulation," Manag. Sci., vol. 44, no. 5, pp.714–729, 1998.

[4] P. Brucker, S. Knust, A. Schoo, and O. Thiele, "A branch-and-bound algorithm for the resource constrained project scheduling problem," Eur. J. Oper. Res., vol. 107, no. 2, pp. 272–288, 1998.

[5] B. Gavish and H. Pirkul, "Algorithms for multi-resource generalized assignment problem," Manag. Sci., vol. 37, no. 6, pp. 695–713, 1991.

[6] T. Baar, P. Brucker, S. Knust, 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, Dordrecht, 1998, pp. 1–8.

[7] 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, 2003.

[8] S. Hartmann, "A competitive genetic algorithm for resource-constrained project scheduling," Naval Research Logistics, Vol. 45, pp. 733–750, 1998.

[9] 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, 1996.

[10] E. Pinson, C. Prins, and F. Rullier, "Using tabu search for solving the resource-constrained project scheduling problem," Proceedings of the Fourth International Workshop on Project Management and Scheduling, Leuven, Belgium, 1994, pp. 102–106.

[11] S.-E. Sampson, E.-N. Weiss, Local search techniques for the generalized resource-constrained project scheduling problem, Naval Research Logistics 40 (1993) 665–675.

[12] S. Hartmann and R. Kolisch, "Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem," European Journal of Operational Research, Vol. 127, pp. 394–407, 2000.

[13] R. Kolisch, S. Hartmann, Heuristic algorithms for solving the resource-constrained project scheduling problem: Classification and computational analysis, in: J. Weglarz (Ed.), Project Scheduling: Recent Models, Algorithms and Applications, Kluwer Academic Publishers, Berlin, 1999, pp. 147–178.

[14] H. Wang, T.-L. Li, and D. Lin, "Efficient genetic algorithm for resource-constrained project scheduling problem," Transactions of Tianjin University, Vol.16, No.5, 2010, pp. 376–382, doi: 10.1007/s12209-010-1495-y.

[15] F. Gargiulo and D. Quagliarella, "Genetic Algorithms for the Resource Constrained Project Scheduling Problem," Proc. The 13th IEEE International Symp. Computational Intelligence and Informatics (CINTI 2012), IEEE Press, Nov. 2012, pp. 39–47.