A Hybrid Genetic Algorithm for the Resource Constrained Multi-Project Scheduling Problem

Jinghua Li, Wenjian Liu
School of Mechatronics Engineering, Harbin Institute of Technology
P.O.Box 306, Harbin Institute of Technology,
Harbin, 150001, China
Jinghua@hit.edu.cn, likewalls@163.com

Abstract

Due to the widespread availability of the internet, large scale distributed projects in manufacturing are becoming popular. Besides resource constraints, there exist precedence constraints among activities within each project. This paper presents a hybrid genetic algorithm to solve the resource-constrained multi-project scheduling problem (RCMPS), which is well known NP-hard problem. Objectives described in this paper are to minimize total project time of multiple projects. The chromosome representation of the problem is based on activity lists. The proposed algorithm was operated in two phases. In the first phase, the feasible schedules are constructed as the initialization of the algorithm by permutation based simulation and priority rules. In the second phase, this feasible schedule was optimized by genetic algorithm, thus a better approximate solution was obtained. Finally, after comparing several different algorithms, the validity of proposed algorithm is shown by a practical example.

1. Introduction

With widespread availability of the Internet, large scale distributed projects in manufacturing are becoming popular. Project scheduling play important role in project management. It involves the allocation of the given resources to projects to determine the start and completion times of the detailed activities. There may be multiple projects contending for limited resources, which makes the solution process more complex. The allocation of scarce resources then becomes a major objective of the problem and several compromises have to be made to solve the problem to the desired level of optimality.

Tools to aid in project scheduling, once activity durations, precedence relationships, and the levels of each resource are known, have existed for some time. Such tools include Gantt charts and the networking tools, such as Critical Path Method (CPM) and the Program Evaluation and Review Technique (PERT). As valuable as these tools are, they have serious limitations for project activity scheduling in practice. Their use assumes unlimited resources for assignment to project activities exactly when required. Furthermore, they are applied to only one project at a time. In many practical environments where project scheduling is an important activity, resources are constrained in number and more than one project is active at any one time.

We propose a mathematical model for scheduling activities in multiple projects which are obeyed multiple resource constraints, precedence constraints in RCMPS environment, and then we develop a new hybrid genetic algorithm to solve the RCMPS problems which are well know NP-hard problem. The paper proceeds as follows: Section 2 provides the related research on RCMPS and genetic algorithm, Section 3 describes model of the resource -constrained multi-project scheduling problem and mathematical model of RCMPS. In Section 4, a hybrid genetic algorithm for solving the mathematical model is described. Section 5 gives the computational experiments and results. The conclusion follows in Section 6.

2. Literature Review

The RCMPS is a generalization of the resource constrained project scheduling problem, RCPSP. The RCPSP has been treated by multiple approaches. In contrast, for the RCMPS problem there are only few studies involving the scheduling of several projects.

It has been shown by Blazewicz et al. [1] that the RCPSP, as a generalization of the classical job shop scheduling problem, belongs to the class of NP-hard optimization problems. The RCMPS problem, as a generalization of the RCPSP, is therefore also NP-hard. In order to obtain optimal or suboptimal solutions, are classified into two major categories.

(a) Exact methods. These results explore the full space of the scheduling alternatives. Usually they are based on branch and bound procedures to avoid full enumeration. Recent proposals were presented by Brucker et al. [2].

(b) Heuristics. These algorithms do not guarantee the solution of the optimum but they tend to faster. Most of the heuristics methods used for solving resource constrained multi-project scheduling problems belong to the class of priority rule based methods. Recently, several procedures based on tabu search, simulated annealing or genetic methods have been applied to the RCPMP [3].

Genetic Algorithms were first used by Davis [4] for the RCPSP as well as many other NP-Hard scheduling
and sequencing problems. The application of GAs in project scheduling has expanded in recent years. Their differences lie in three aspects mainly. First, the ways to encode the individuals; Second, the methods to ensure activity-precedence constrains; and third, the strategies to improve the performance of GA. It is showed in [5] that GA-Based solutions can get better quality of schedules, although there execution times are much higher than the other alterations. However, few studies have focused on the application of GAs in multi-project scheduling. Kim et al. [6] developed a hybrid genetic algorithm with fuzzy logic controller to solve RCMPSP. This approach is based on the design of genetic operators with fuzzy logic controller (FLC) through initializing the revised serial method which outperforms the non-preemptive scheduling with precedence and resources constraints. Goncalves [7] presented a genetic algorithm for RCMPSP based on randomkeys, and a new schedule generation procedure that creates parameterized active schedules.

In this paper, the objectives of the RCMPSP are to minimize total project time of multiple projects. Total project time is the sum of the completion times for all projects which scheduled the activities such that precedence and resource constraints are obeyed in multiple projects. The resource constraints refer to limited renewable resources are necessary for carrying out the project activities.

We propose a mathematical model for scheduling activities in multiple projects which are obeyed multiple resource constraints, precedence constraints in RCMPSP environment, and then we develop a new hybrid genetic algorithm with to solve the RCMPSP problems which are well known NP-hard problem.

### 3. Problem description and mathematical Model

The following initial assumptions are made firstly: (a) The RCMPSP consists of multiple projects which are a number of activities (tasks) with known processing time and multiple resources; (b) The RCMPSP must be finished without changing the project, when once initiated in a specific project (precedence constraints of multiple projects); (c) Start time of each activity is dependent upon the completion of some other activities (precedence constraints of activities). After finishing a specific activity, next activity must be also started in a project; (d) The multiple resources are available in limited quantities but renewable from period to period; (e) Activities cannot be interrupted, there is only one execution mode for each activity; (f) the managerial objective is to minimize the total project time for all projects.

Our formulation uses the following notations:

- \( i \) project index, \( i = 0, \ldots, I + 1 \) (or \( \Pi_i = 0, \ldots, I + 1 \) if \( \Pi_i \) is a dummy project).
- \( j \) activity index in each project, \( j = 0, \ldots, J + 1 \)

The objective function (1) minimizes the total project time that is the sum of the completion times for all projects. Resource constraint (2) correspond to resource constrains regarding nonrenewable resources. In constraint (3), activity \( j \) in each project must not be started before all activities of precedence \( j \) are finished (precedence relations between related activities). Lastly, precedence constraint (4) states precedence relations between related projects.

### 4. Genetic algorithms

Genetic algorithms (GA) are stochastic search techniques for approximating ‘optimal’ solutions within complex search spaces [8]. In this research, the general procedure for GA [9] was modified. Figure 1 describes the proposed GA process to solve RCMPSP. The proposed algorithm was operated in two phases. In the first phase, the feasible schedules are constructed as the initialization of the algorithm by permutation based simulation and priority rules. In the second phase, this feasible schedule was optimized by genetic algorithm, thus a better approximate solution was obtained. A refresh operator was added in order to assign the activities in the domain of time and resource by accommodating precedence and resource constraints.

#### 4.1 Chromosome representation

Generally, the chromosome representation methods...
A chromosome makespan. In the first GA variant to be examined, a sequence must be completed to determine the overall total activity number. Each chromosome as an activity an activity ID number. Chromosome length is set to the unique Identification Number and each gene represents can be either direct or indirect. This paper uses indirect rules based sampling scheme collectively, in our case the solution quality, we employ priority rules or priority of precedence feasible individuals. To obtain better simulation [10] is used to produce an initial population of infeasible solutions. So a permutation based function could result in the generation of a large number of constrained resource allocation problems have a small no worse than the conventional heuristic does. Highly genetic algorithm with priority rules can guarantee to do considered again. We obtain already been taken from the mother may not be substring 1, after deleting activities that appear in substring 1, Step 3: Find the exclusive substring from parent 2 after deleting activities that appear in substring 1, named substring 1.

4.3. Crossover

In order to minimize the total project time, maximize the alternative schedule with optimal total project time, we use genetic operators for RCMPSP. The crossover operators used are a one-point uniform crossover [11] and a Union crossover operator 3(UX3) [12].

(a) The first crossover operator is called one-point crossover. We consider two individuals selected for crossover, a mother M and a father F. Then we draw a random integer with . Now two new individuals, a daughter D and a son S, are produced from the parents. We first consider D which is defined as follows: In the task sequence of D, the positions in D is taken from the father. However, the tasks that have already been taken from the mother may not be considered again. We obtain

\[ j_i^D = j_i^M \]

The task sequence of positions in D is taken from the father. However, the tasks that have already been taken from the mother may not be considered again. We obtain

\[ m = \min \{m \mid j_m^F \notin \{j_1^M, \ldots, j_{m-1}^M, j_{m+1}^M, \ldots, j_n^M\}, m = 1, 2, \ldots, J \} \]

(b) Another crossover procedure used is based on a Union crossover 3 (UX3) method. Union crossover 3 (UX3) is essentially a random shuffle of two exclusive substrings that maintain precedence feasibility. UX3 is performed as follows:

Step 1: Given two parent strings 1 and 2, select two positions along both parents randomly through the whole string.

Step 2: Select the middle substring defined between the two crossing sites from parent 1, named substring 1.

Step 3: Find the exclusive substring from parent 2 after deleting activities that appear in substring 1, named substring 2.

Step 4: Randomly select activities from the two exclusive substrings; reselect activities if the immediate predecessors are not selected.

Step 5: Fill the activity index into the offspring chromosome, positioned from left to right.

Step 6: Repeat Steps 3 and 4 until all chromosomes of population size are generated.

4.2. Initial population production

Incorporate heuristics into initialization to generate well-adapted initial population. In this way, a hybrid genetic algorithm with priority rules can guarantee to do no worse than the conventional heuristic does. Highly constrained resource allocation problems have a small feasible search space. Therefore random generation of strings and incorporating a penalty into the objective function could result in the generation of a large number of infeasible solutions. So a permutation based simulation [10] is used to produce an initial population of precedence feasible individuals. To obtain better solution quality, we employ priority rules or priority rules based sampling scheme collectively, in our case the latest finish time(LFT), the latest finish time(LST) and ACTIM, for choosing an eligible activity. This procedure proceeds as follows:

Step 1: Start the dummy source activity at time 0.

Step 2: Randomly select a activity from all unselected task pool, and check if its immediate predecessor(s) are already selected. If not yet selected, continue this random selection until a satisfying activity is found.

Step 3: Compute the set of the eligible activities and use priority rule LFT to select the eligible activity with the highest priority.

Step 4: After updating the eligible set, the second activity to be scheduled is selected using priority rule LST.

Step 5: Repeatedly scheduling an unscheduled activity by MSLK, a feasible active schedule is obtained.

Step 6: Repeat 2, 3, 4, 5 until all chromosomes of population size are generated.
4.4. Mutation

Given a activity list based individual \( V_i \), the mutation operator modifies the related activity sequence as follows: For all positions \( k = 1, \ldots, J-1 \), activities \( j^k_i \) and \( j^k_i + 1 \), are exchanged with a probability of \( p_{\text{mutation}} \) if the result is a activity sequence which fulfills the precedence assumption.

The mutation operator may create activity sequences (i.e. gene combinations) that could not have been produced by the crossover operator. However, it should be noted that performing a mutation on an individual does not necessarily change the related schedule. This is due to the redundancy in the genetic representation: For example, interchanging two activities in the activity sequence which have the same start time changes the individual, but not the related schedule.

4.5. Refresh

Activity starting times, and resource constraints are considered by another operator, named Refresh [13]. This operator acts to set the starting time of each activity and to obtain the makespan of the completion of all parallel projects for each individual real coded string of the population. The makespan is then used for the fitness measurement for evaluating members of the population. Based on the ordered lists of activities given as the chromosome representation, it allocates the limited resources to activities in turn and keeps track of when resources become available. The steps to this process are as follows:

**Step 1:** Initialize the available time of all resources to 0
**Step 2:** Start with the first activity index listed in the chromosome, compute
\[
\max \left\{ S_d + \sum_{j \in S_p} P_j, \max \{ S_j + P_j, \} \right\}
\]
where, \( j \) is the index of the activity last performed by the resource to use.
\( j \) is the index immediate preceding activity of this activity.
**Step 3:** Repeat the step 1 and 2 until all the tasks have been scanned and computed and the makespan of string is then obtained.

4.6. Evaluation and fitness

Evaluation function plays the same role in GA as that which the environment plays in natural evolution. The objective function value is considered as the evaluation rule of every individual. Sometimes the best and the worst chromosomes will produce almost the same numbers of offspring in the next population, which cause premature convergence. In this case, the effect of natural section is therefore not obvious. A linear normalization, which converts the evaluations of chromosomes into fitness values, is used here to overcome this problem. This paper defines the fitness function of an individual \( V_i \) as following:

\[
g(V_i) = \frac{f_{\text{max}} - f(V_i) + r}{f_{\text{max}} - f_{\text{min}} + r}
\]
where \( V_i \) is the \( i \) chromosome in the next population. \( g(V_i) \) is fitness function. \( f_{\text{max}} \) is the maximum makespan of a generation. \( f_{\text{min}} \) is the minimum makespan of a generation. \( r \) is a parameter used to adjust the probability distribution of individuals.

4.7. Selection

Selection occurs for solutions showing high fitness. To avoid loss of the best chromosomes' genes, two schemes are used for this realization.

(a) The proportional selection, which can be viewed as a randomized version of the previously described ranking technique. Let \( g(V_i) \) be the fitness of an individual \( V_i \), and let pop-size denote the current population, that is, a list containing the individuals. Note that we use a list of individuals instead of a set because we explicitly allow two (or more) distinct individuals with the same genotype in a population. We restore the original population size by successively removing individuals from the population until \( \text{POP} \) individuals are left, using the following probability:

\[
P(V_i) = \frac{g(V_i)}{\sum_{a=1}^{\text{POP}} g(V_a)}
\]

(b) The elitist strategy, which fixes the potential best number loss by copying the best member of each generation into the succeeding generation;

The second strategy may increase the speed of domination of a population by a super individual as is therefore used in the initial generations to speed up the convergence.

5. Validation

The proposed algorithms are implemented in Delphi language on PC Pentium 1400 MHz clock-pulse and 256 MB RAM as operation system. The RCMPSP consists of three flexible projects. These three projects can perform 57 activities totally including two dummy activities (start and end activity). Figure 2 provides the network of RCMPSP. The parameters environment for the problem was set as follows:

- Population size = 45
- Crossover probability = 0.3
- Mutation probability = 0.001
- Maximum number of iterations = 100

It is the optimal scheduled one profile out of 149 alternative schedules about example of RCMPSP problem. We could found the best optimal solution of total project time (\( t_F = 59 \)) at 19 generation. \( V = \{1,5,4,11,14,3,8,2,13,10,11,16,7,6,23,9,18,15,19,27,20,7,24,21,29,22,30,28,31,26,32,27,33,34,35,36,38,40,37,39, \)
41,42,43,44,45,46,47,48,49,50,51]. Relevant activity start-time was [0, 5,12,3, 19,14,23,29,34,17,24,20,15,26, 20, 30, 24, 20, 31, 36, 37, 36, 31, 36, 30, 40, 42, 31, 37, 15,27,23,18,22,20,36,36,37,39,38,46,42,45,50,51,50,53, 54,57,56,58]. Relevant activity finish-time was [4,11,15,7,25,20,28,33,22,27,24,18,31,26,34,28,23,35,40,41,33,39,34,46,45,36,42,21,30,28,23,41,38,43,42, 43,48,50,48,52,53,53,56,57,59,59,63]. Figure 3 shows makespan distribution. Figure 4 compares the makespan obtained by those algorithms. Figure 4 shows that the proposed algorithms can get better solution for RCMPSP than the others.

6. Conclusions

This paper presents a new hybrid genetic algorithm to solve the RCMPSP. The proposed algorithm was operated in two phases. In the first phase, the feasible schedules are constructed as the initialization of the algorithm by permutation based simulation and priority rules. In the second phase, this feasible schedule was optimized by genetic algorithm, thus a better approximate solution was obtained. A refresh operator was added in order to assign the activities in the domain of time and resource by accommodating precedence and resource constraints. Finally, after comparing several different algorithms, the validity of proposed algorithm is shown by a practical example.

Acknowledgements

This work is partially supported by the COSTIND(Commission of Science Technology and Industry for National Defense) Research Project, China, under Contract 20030119.

References


