Journal of Science and Engineering Technology, Vol. 4, No. 2, pp. 63-74 (2008)

Two Heuristic Algorithms for a Multi-Mode Resource-Constrained Multi-Project Scheduling Problem

CHING-CHIH TSENG
Department of Business Administration, Da-Yeh University
No. 112, Shanjiao Rd., Dacun, Changhua, Taiwan 51591, R.O.C.

ABSTRACT

Many studies on the resource-constrained project-scheduling problem have been published, but literature further considering multi-mode or multi-project issues often occurring in the real world is rather scarce. In this research, two heuristic algorithms are developed to solve a multi-mode resource-constrained multi-project scheduling problem (MMRCMPSP). The first, a parallel scheduling algorithm (PSA), includes a combination of an activity- and a mode-priority rule; the second is a genetic algorithm (GA). The solutions obtained by the former algorithm with the best activity- and mode-priority rule combination are used as a baseline to compare those obtained by the latter. On four sets of test problems, twenty combinations of rules are compared to determine the best one for deriving the most likely best solution in the proposed PSA. In each set, two resource-availability levels and two due-date levels are considered. Due to certain priority rules' having probabilistic factors, a two-stage sampling method is introduced to ensure that more reliable computational results are obtained. Finally, the solutions obtained by the proposed PSA having the best activity- and mode- priority rule combination are compared on the test problems with those obtained by the proposed GA. Finally, conclusions are drawn from the computational results.

Key Words: multi-project scheduling, parallel scheduling algorithm, genetic algorithm, two-stage sampling

求解多模式資源限制多專案排程問題之二個啟發式演算法

曾清枝
大葉大學企業管理學系
彰化縣大村鄉山腳路 112 號

摘　要

至今有大量關於資源限制排程問題被發表，但對於探討多模式甚至於多專案之文獻卻相對稀少。因此，本文針對多模式資源限制多專案排程問題提出二個啓發式演算法，分別是結合作業和模式優先派遣法組合之平行排程演算法和遺傳演算法。依據前者演算法在四組模擬測試問題中比較 20 個作業和模式組合後，選出最可能求得最佳解之作業和模式最佳組合當作後者演

C.E.P.S.

算法之比較基礎。在每一組問題中，考慮二個資源可用量水準和二個限制工期水準。由於某些排程法則含有機率因子，兩階段抽樣方法被導入以確保可獲得更可靠的模擬結果。最後本文就所提出平行排程演算法使用最佳作業和模式優先派遣法組合，和所提出之遺傳演算法作模擬比較，結果發現所提出之遺傳演算法優於所提出平行排程演算法。

關鍵詞：多專案排程，平行排程演算法，遺傳演算法，兩階段抽樣方法

I. INTRODUCTION

A resource constrained project scheduling problem (RCPSP) is important and challenging to both practitioners and mathematicians. In project management practice, more than one project is often performed simultaneously under limited resources within the same organization. Planners are generally concerned with a number of different decision criteria, often conflicting among each other, according to their importance and priorities. Hence, the allocation of scarce resources among different projects to achieve the optimization of an objective function is an important consideration for a project manager.

The multi-mode resource-constrained multi-project scheduling problem (MMRCMPSP) is a generalized case of the RCPSP. The MMRCMPSP consists of a number of projects, defined as collections of activities performed in one of several ways under given precedence relationships and limited amounts of various types of resources. The RCPSP belongs to the class of NP-hard problems identified by Blazewicz et al [2], their general form, MMRCMPSP, also being NP-hard. Many publications have provided various approaches for investigating diverse versions of project scheduling problems. Most methods concentrate on a single project problem in which each activity is processed in only one way, called a single mode, or more than one way, called multi-mode. In multiple modes each activity can be processed in one of several ways, wherein each execution mode is characterized by a known duration and given resource requirements. The multi-modes RCPSP was introduced by Elmaghraby [9]. Many exact and heuristic algorithms have been proposed by researchers [1, 3-5, 13, 20, 25, 27, 28]. Kolisch and Padman [15] surveyed a vast literature in this area from a perspective that integrates models, data, and optimal and heuristic algorithms for the major classes of project scheduling problems.

Concerning the resource constrained multi-project scheduling problem (RCMPSP), only a few studies have been published. The pioneering work on multi-project scheduling was done by Pritsker et al. [26], who proposed a zero-one programming approach. Kurtulus and Davis [16], when considering the multi-project scheduling problem, provided a categorization process based on two powerful project summary measures. Kurtulus and Narula [17] examined the performance of ten scheduling rules for a multi-project environment. Lova et al. [21] developed a multicriteria heuristic that lexicographically improves two criteria: one-time type such as mean project delay or multiproject duration increase; and no one-time type such as project splitting, in-process inventory, resource leveling or idle resources.

Lee and Lei [19] presented efficient algorithms for solving several special cases of such multi-project scheduling problems by imposing controllable project duration and hard-resource budget constraints. Lova and Tormas [22] examined the effect of two schedule generation schemes and five activity priority rules in single-project and multi-project environment. Kim et al. [23] proposed a hybrid genetic algorithm with fuzzy logic controller to solve the RCMPSP. The multi-project scheduling environment which they considered is in a series rather than parallel. The basic idea of their proposed approach is to use a serial method which considers priority-based encoding to find an appropriate order of activities. Then, a hybrid genetic algorithm with a fuzzy logic controller comprising specific heuristics is used to determine the early finish times for multiple projects. Goncalves et al. [12] presented a genetic algorithm for the RCMPSP, the chromosome representation of the problem being based on random keys. The schedules are constructed by using a heuristic that builds parameterized active schedules based on priorities, delay times, and release dates defined by the genetic algorithm. However, there appears to be no literature on MMRCMPSP.

In this research two heuristic algorithms are proposed to solve the MMRCMPSP. The first, called a parallel scheduling algorithm, includes a combination of an activity- and a mode-priority rules; the second, a genetic algorithm. The solutions obtained by the former algorithm with the best activity- and mode- priority rule combination are used as a baseline to compare those by the latter one.

The structure of this paper is as follows: in Section 2 the considered problem is described. In Section 3, a parallel scheduling algorithm for the MMRCMPSP and some activity- and mode- priority rules are introduced. In Section 4, a genetic algorithm for the MMRCMPSP is developed. Section

5 consists of three parts: Part one describes a test problems generator for MMRCMPSP; part two introduces a two-stage sampling procedure to determine necessary replications to obtain a more reliable best combination of activity- and mode-priority rules which are the most likely to find the maximal solution of MMRCPSP; part three provides a comparison of the GA and the PSA including the best rule combination.

II. PROBLEM STATEMENT

The considered problem consists of I parallel projects, each project i=1,…, I, being composed of J_i activities ij, j=1,…, J_i. Each project i also has two common dummy activities: source and sink. Activity ij in project i may be performed in one of the modes m=1,…, M_{ij}. Each activity, once initiated in a specific mode, must be finished without changing the mode. The activities are interrelated by two kinds of constraints, the first being precedence constraints, which force each activity ij in project i to be scheduled after all its predecessors P_{ij} are finished. The second type consists of resource constraints, which restrain the activities to be processed, subject to the limited availability of resources. While being processed, activity ij in project i performed in mode m requires q_{ijmr} units of renewable resource type r= 1,.., R, during each period of its non-preemptive duration d_{ijm}. Resource type r has a fixed and limited available amount Q_r.

For project i, it is assumed that a known and predetermined due date, DD_i, exists. In comparing to the completion time, C_i, of project i, two types of costs, *early completion incentives*, EC_i, and *late completion penalties*, LC_i, are considered. The former represents benefits obtained by completing a project earlier than the due date; whereas, the latter represents penalties for missing the due date. This problem is common in practice, and is confronted by contractors, engineering firms, maintenance crews, research and development teams, and similar organizations, all of which must simultaneously manage a number of projects subject to finite resources and due date constraints.

The objective is to maximize the weighted profit of projects. Thus, the objective function can be written as

$$\text{Maximize} \quad \sum_{i=1}^{I}\left[EC_i\left(DD_i - C_i\right)^+ - LC_i\left(C_i - DD_i\right)^+\right] \quad (1)$$

Before the two proposed heuristic algorithms are introduced, the early start time and late finish time activity ij are defined as

$$ES_{ij} = \begin{cases} 0 & if \quad P_{ij} = \phi, \\ max_{ip} \quad \{(ES_{ip} + min_m\{d_{ijm} \mid m = 1,\ldots,M_{ij}\}) : ip \in P_{ij}\} & otherwise. \end{cases}$$

(2)

$$LF_{ij} = \begin{cases} \max\{DD_i, SCPL_i\} & if \quad S_{ij} = \phi, \\ \min_{is}\{(LF_{is} - \min\{d_{ijm} \mid m = 1,\ldots,M_{ij}\}) : is \in S_{ij}\} & otherwise. \end{cases}$$

(3)

The term ES_{ij} (LF_{ij}) denotes the early start time (late finish time) of activity ij; P_{ij} (Sij) denotes the set of immediate predecessor (successor) activities to activity ij; $SCPL_i$ is the critical path length of project i having the shortest duration mode for each activity.

As above-mentioned in introduction section, MMRCPSP belongs to NP-hard problem. The currently most powerful optimization procedures are unable to optimally solve highly resource-constrained projects with more than 20 activities and three execution modes per activity within reasonable computation times [27]. This is the reason that two heuristic algorithms are proposed.

III. PARALLEL SCHEDULING ALGORITHM

During the scheduling of multiple projects, the activities in an eligible set (a set of activities eligible to be scheduled at the current time), EJ, cannot be processed concurrently due to a resource conflict. (Activities are said to be resource conflict when their requirements exceed the currently available level of at least one type of resource.) In this situation the concern is how to decide which activity in EJ should be scheduled first and in which mode it should be performed. On the basis of resource restrictions, the necessary sequence should be decided by a predetermined rule if any resource conflict occurs among the currently schedulable activities.

The parallel scheduling algorithm (PSA), first formulated by Fendley [10] to deal with RCMPSP, developed to solve the MMRCMPSP is described as follows. Feasible solutions to this problem can be obtained by using the PSA. Initially, set the clock time, t_{now} = 0, and place all activities ij of each project in the set of unscheduled activities, U. At t_{now} = 0, or any subsequent time that an activity is finished, a new EJ is specified. If a conflict in resources in EJ does not occur, all activities are added to the active set (i.e., the set of activities currently being processed). If such a conflict occurs, decide which activity in EJ should be processed first according to one of the activity priority rules described in the first subsection 3.1 below. Concurrently, the mode which should be assigned to the selected activity can be decided by using one of the mode priority rules, also described in subsection 3.1. Subtract the remaining available amount of resource type k by the amount of resource type k required by the selected activity. Then, place the selected activity in the active set. Update the new EJ and repeat the process (i.e., to determine whether any other activity in the new eligible set can now be processed). If the remaining available amount of any resource type k is not enough to process any activity of EJ or if the eligible set is empty, t_{now} is advanced to the earliest completion time of activities currently in the active set, the finished job is removed from the active set, and an EJ set is formed. A schedule is completed when U is empty; the value of t_{now} is the completion time of a project.

1. Activity Priority Rules

It is known that many activity priority rules exist. Here four representative rules are considered.

A. RAN-A (Randomly choose activity) rule: Randomly choose an activity from EJ to be processed. This rule serves as a benchmark against which all other rules could be evaluated.

B. MINLFT (Minimum Late Finish Time) rule: Schedule an activity with the minimum value of

$$\min_{ij} \quad LF_{ij}, \quad ij \in EJ. \tag{4}$$

Remark: Talbot [3] compared eight deterministic heuristic scheduling rules for multi-mode RCPSP and found that the MINLFT rule behaved better than the others.

C. STOCOM-A (Stochastic Construction Method for activity selection) rule: Randomly select an activity from EJ in accordance with its corresponding probability in proportion to γ_{ij}. In this regret-based biased random priority rule proposed by Drexl and Gruenewald [8], the regret value of a candidate activity ij measures the worst-case consequence that might arise from selecting another activity. Choose activity ij with proportional to γ_{ij}. γ_{ij} is defined as follows.

$$\gamma_{ij} = \left(\max\{LF_{ik} | ik \in EJ\} - LF_{ij} + \varepsilon\right)^{\alpha} \qquad ij \in EJ\,. \tag{5}$$

This equation estimates the worst-case consequence of not scheduling activity ij with respect to LF_{ij}. The expression $\varepsilon > 0$ makes γ_{ij} always positive; whereas, $\alpha \geq 0$ transforms the term (.) in an exponential way, thus diminishing or forcing the difference between LF for $\alpha < 0$ or $\alpha > 0$. As shown in [8], the optimal solutions occur with a high frequency at various problem instances when $\alpha = 2$ and $\varepsilon = 1$.

D. MINSL (Minimum Slack) rule: Schedule an activity with the minimum value of

$$\min_{ij} \quad LF_{ij} - \sum_{m=1}^{M_{ij}} \frac{d_{ijm}}{M_{ij}} - t_{now}, \quad ij \in EJ. \tag{6}$$

Where t_{now} denotes the current clock time previously mentioned in Section 3. This rule was found effective in multi-project scheduling by many authors addressed in Davis and Patterson [6].

2. Mode Priority Rules

Once activity ij is chosen, assign mode m by one of the following rules.

A. RAN-M (Randomly choose mode) rule: Randomly perform activity ij in mode m by derived from a uniform distribution from 1 to M_{ij}.

B. MINDM (Minimum duration mode) rule: Perform activity ij in mode m by using the minimum value of

$$\min_{m} \quad d_{ijm}, m = 1,\ldots, M_{ij} \tag{7}$$

C. STOCOM-M (Stochastic Construction Method for mode selection) rule: As in activity priority rule 3, assign mode m to process activity ij in proportion to ω_{ijm}. Randomly choose mode m^*, which in proportion to ω_{ijm} for every activity $ij \in EJ$ and m=1, M_{ij}. ω_{ijm} is defined as

$$\omega_{ijm} = (\max\{d_{ijl} | l = 1,\ldots, M_{ij}\} - d_{ijm} + \varepsilon)^{\alpha}, m = 1,\ldots, M_{ij} \tag{8}$$

The meanings of α and ε are the same as in activity priority rule 3.

D. MAXDM (Maximum Duration Mode) rule: Perform activity ij in mode m by using the maximum value of

$$\max_{m} \quad d_{ijm}, m = 1,\ldots, M_{ij}. \tag{9}$$

E. MINRDM (Minimum Resource Demand Mode): Perform activity ij in mode m by using the minimum value of

$$\min_{m} \quad \sum_{r=1}^{R} q_{ijmr}\, d_{ijm}, m = 1,\ldots, M_{ij}. \tag{10}$$

IV. GENETIC ALGORITHM FOR MULTI-PROJECT SCHEDULING PROBLEM

Genetic Algorithms (GAs) are optimization techniques for functions defined over finite domains. First proposed by Holland [14], GAs have been successfully applied to a wide variety of problems [11]. In this study, a genetic algorithm for

C.E.P.S.

MMRCMPSP is presented. The approach is based on the incorporation of problem knowledge from the application domain into the genetic algorithm. This approach particularly leads to a new complex non-standard representational scheme for chromosomes that comprises all information relevant to the search task. The introduction of this expanded representation requires the definition of new domain-dependent crossover and mutation operators that take advantage of the additional information represented in the chromosomes.

1. Direct Representation of Problem

In a direct problem representation the MMRCMPSP itself is used as a chromosome. No decoding procedure is therefore necessary. All information relevant to the MMRCMPSP at hand is included in the problem representation. The proposed GA is the method that implements a search since the represented information comprises the entire search space, thus, neither a transformation procedure nor a schedule builder is further necessary.

Complete information of a schedule for the MMRCMPSP consists of an activity priority rule, activities and their corresponding modes in each individual project. The activity priority rule is randomly chosen from the rules described in subsection 3.1; whereas the processing mode is uniformly generated from its available modes. The scheme of the direct representation is sketched as shown in Figure 1. E.g. P1/A11Mα indicates that mode α is assigned to activity 11 of project 1.

The MMRCMPSP is represented as an activity priority rule (abbreviated as j-r) plus a list of projects/activities/modes, as shown in Fig. 1. The quality of a chromosome (i.e., its schedule) is measured by its fitness value (i.e., the weighted profit from the projects).

2. Initialization

The initialization of a start generation of complete and consistent schedules can be accomplished by the following rules:

A. For each chromosome, first randomly select one of the activity priority rules described in subsection 3.1 and place the selection in locus 1; then place each project as a locus in ascending order.

B. For each locus except the first, the activity number is placed in ascending order, and the mode of each activity is randomly assigned from its available modes.

After one chromosome is built, the proposed PSA is employed to schedule this multi-project problem by using the selected activity priority rule under the precedence and resource constraints; thus, the fit value of this chromosome can be obtained.

3. Sorting

Once one generation consisting of a certain number of feasible schedules is constructed, the chromosomes are sorted in descending order by their fitness values (i.e., weighted profit).

4. Genetic Operators

A. Crossover

The crossover operator generates two offspring schedule by combining features of two selected parent schedules. The scheme of the crossover operator is determined by selecting two parent schedulers, one being the schedule having the largest fitness value and the other being randomly chosen. The crossover operator has two parts:

- A number $k \in (1,I)$ chosen at random;
- One offspring schedule consisting of the former projects (i.e., from 1 to k) of parent 1 and the latter (i.e., from k+1 to I) of parent 2. The other schedule is composed of the former projects (i.e., from 1 to k) of parent 2 and the latter (i.e., from k+1 to I) of parent 1. For example, if k = 3 is chosen, two offspring schedules shown in Figure 2, can be obtained. If k is equal to 0, crossover the activity priority rule should be mutually crossed over.

B. Mutation

The mutation operator must be able to alter all information represented in the chromosome. It must also provide the possibility of reintroducing lost genetic material. Thus, two types are devised, one to change modes for selected activities, the other to generate a new schedule, as follows. Randomly select one schedule;

- Randomly select n_i (less than J_i) activities for each project i;
- Change the mode for the selected activities.

As in Figure 3, suppose that the activities marked by an asterisk (*) are selected, thus their modes are changed at

j-r	P1				P2			…	PI		
	A_{11}Mα	A_{12}Mβ	…	A_{1J1} Mγ	A_{21}Mα	…	A_{2J2}Mγ	…	A_{I1}Mα	…	A_{IJI} Mγ

Fig. 1. Direct representation of a schedule

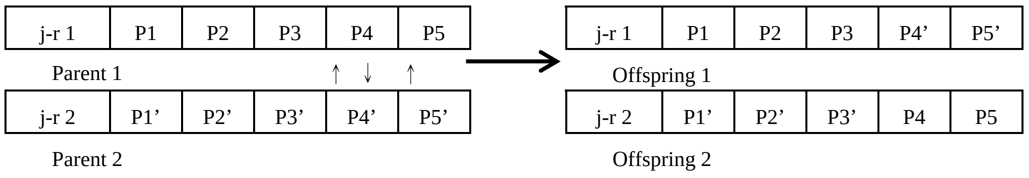

Fig. 2. Example of crossover

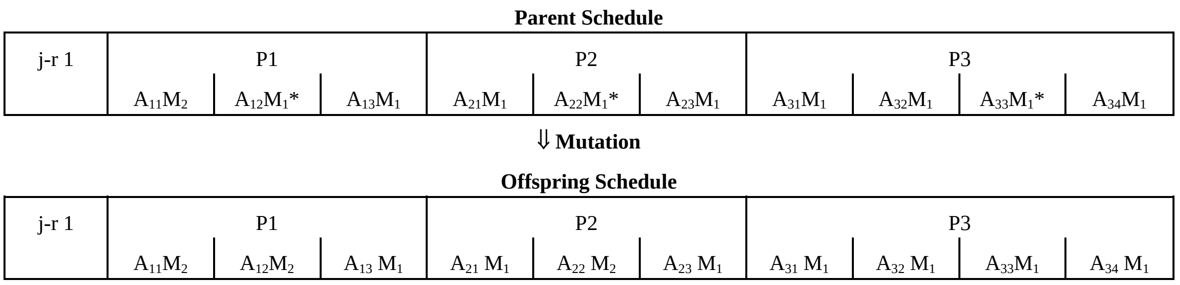

Fig. 3. Example of mutation

random within the available scheme.

5. New Generation

With an appropriate proportion, the operators described above can produce a new generation:

A. Duplicate P_1 offspring schedules from the parent having the largest fitness values (i.e., elitist preserving strategies).
B. Produce P_2 offspring schedules by using crossover operator.
C. Produce P_3 offspring schedules by using the mutation operator.
D. Generate P_4 offspring schedules at random.

$P_1+P_2+P_3+P_4=P$, with P indicating the number of schedules in one generation.

V. COMPUTATIONAL RESULTS

The GA as well as PSA have been coded in C++ and performed on a Pentium III PC with 1.13 GHz clock-pulse and 256 MB RAM.

1. Generator of a Problem Instance

An MMRCMPSP problem instance generator, tailored from previous work [6], is proposed for generation by the following step.

A. Set a problem size I, i.e., the number of projects.
B. Set the number of activities J_i of project i, which is uniformly generated from the interval [(MAX-J/2) +1, MAX-J], where MAX-J, a predetermined value, denotes the maximal number of activities of a project in a problem instance.
C. Set the complexity of project i, F_i, defined as the number of arcs (precedence relations) divided by the number of nodes (activities), of project i.
D. Generate an activity network for each project i by using the random activity network generator proposed by Demeulemeester et al. [7] when J_i and F_i are given.
E. Uniformly generated the (integer) number of modes of activity ij, M_{ij} from the interval [2, 3].
F. Fixed the total number of renewable resource types R = 3.
G. Uniformly generated the (integer) duration d_{ijm} of activity ij, performed in mode m, from the interval [5, 10].
H. Uniformly generated the amount q_{ijmr} of resource r required by activity ij, performed in mode m, from the interval [1, 5].
I. Determine the availability Q_r of renewable resource type r by multiplying the peak resource requirement

$$Q'_r=\max\{\,q_{ijmr}\mid i=1,\ldots,I,\ ij=1,\ldots,J_i,\ m=1,\ldots,M_{ij}\}\quad (11)$$

with θ, thus obtaining $Q_r=Q'_r\cdot\theta$, $r=1,\ldots,R$ with $2.0\le\theta\le 4.0$.

When given J_i and F_i, a network of project i can be generated. In this research, I = 5 and F_i = 2.5 are assumed for all problem instances.

2. Identification of the Most Likely Best Combination of Rules

The main experimental factors considered in this study are two resource-availability levels (θ = 2.0, 4.0), two due-date tightness levels (for each project i, the due date DD_i is set at one time or three times the corresponding $LCPL_i$, for convenience, denoted as *DD=LCPL* and *DD=3LCPL*; $LCPL_i$ is the critical path length of project i having the longest duration mode for each activity). In each scenario (i.e., a combination of these two factors), 20 policies, combining the different activity priority rules with the mode priority rules, are implemented. Here the objective is to identify the best policy for obtaining the maximal weighted project profit in each scenario. During the simulation process, an immediate problem resides in certain priority rules, the probabilistic factors of which will affect the reliable output of the simulation. In the following a two-stage sampling method is introduced to overcome this problem.

As described in [18, pp. 596-597], the two-stage sampling method stated below has a favorable property wherein, with a probability of at least P_r*, the expected response of the selected best policy will be larger than that of the second-best policy with $d*$, where $d*$ denotes the indifference amount. This method can determine how many replications of a policy should be performed to attain a reliable output from the simulation. In the first stage, set a fixed number of replications for each policy, then use the resulting sample mean and variance estimates to determine how many more replications for each policy are necessary in the second stage to reach a reliable decision.

First-stage Set $n_0 \geq 2$ replications for each policy i =1, 2,…,ρ (ρ represents the number of policies) and define the first-stage sample mean, $\overline{X}_i^{(1)}(n_0)$, and variance, $S_i^2(n_0)$, as

$$\overline{X}_i^{(1)}(n_0)=\frac{\sum_{j=1}^{n_0} X_{ij}}{n_0} \tag{12}$$

and

$$S_i^2(n_0)=\frac{\sum_{j=1}^{n_0}\left[X_{ij}-\overline{X}_i^{(1)}(n_0)\right]^2}{n_0-1} \tag{13}$$

Then compute the total replications, N_i, necessitated for policy i as

$$N_i=\max\left\{n_0+1,\left\lceil\frac{\tau^2 s_I^2(n_0)}{\left(d^*\right)^2}\right\rceil\right\} \tag{14}$$

where $\lceil X\rceil$ is the smallest integer greater than or equal to the real number X, and τ (which depends on ρ, P_r*, and n_0) is a constant ([18]).

Second-stage Next, N_i - n_0 additional replications of policy i are generated, for which the second-stage sample mean is computed as

$$\overline{X}_i^{(2)}(N_i-n_0)=\frac{\sum_{j=n_0+1}^{N_i} X_{ij}}{N_i-n_0} \tag{15}$$

Then define the weight, W_{i1}, as

$$W_{i1}=\frac{n_0}{N_i}\left[1+\sqrt{1-\frac{N_i}{n_0}\left(1-\frac{(N_i-n_0)(d^*)^2}{\tau^2 S_i^2(n_0)}\right)}\right] \tag{16}$$

and set W_{i2} = 1 - W_{i1} for i = 1,2,…,ρ. The weighted sample mean, $\tilde{X}_i(N_i)$ of policy i can be computed as

$$\tilde{X}_i(N_i)=W_{i1}\overline{X}_i^{(1)}(n_0)+W_{i2}\overline{X}_i^{(2)}(N_i-n_0) \tag{17}$$

Finally, select the policy i with the largest $\tilde{X}_i(N_i)$ among i=1,…, ρ.

The objective is to select a policy having the largest sample mean (i.e., the maximal average weighted profit) at a surety level higher than 100$P*$ =95 percent that the correct selection has been made, provided that $d*$ = 5. For each scenario an MMRCPSP instance with any MAX-J=20 to 50 with F_i =2.5 and I =5 is generated. Concurrently, the value of each EC_i (early completion incentive) is set at 5, and the LC_i (late completion penalty) is set at 4 for each project i. For each problem instance, first set n_0 =30; then, compute τ =3.92 (see [18, p. 130]). Thus, the results from the first-sampling, $\overline{X}_i^{(1)}(30)$ and $S_i^2(30)$, can be obtained. When given the respective values of $S_i^2(30)$, τ, and d*, compute the total sample size N_i for each policy; then, make N_i - 30 additional replications for each policy. For example, in a combination of MINLFT and STOCOM-M, the total sample size is 288; hence, 258 replications are needed to generate in the second stage (see Table 1). Once N_i - 30 replications are produced, the second-stage sample mean $\overline{X}_i^{(2)}(N_i-30)$ can be obtained. Finally, calculate the values of W_{i1} and W_{i2} for each policy and the weighted sample mean $\tilde{X}_i(N_i)$.

Here an example of a two-stage sampling method for MAX-J=30 is given in Table 1. The overall results from the weighted sample means in all the scenarios for MAX-J=20 to 50 with F_i =2.5 and I =5 are shown in Tables 2-5.

The best policy which obtains the largest weighted sample means (the bold numerals in Tables 2-5) of all scenarios for MAX-J=20 to 50 is summarized in Table 6. On this Table, among the activity priority rules, MINSL is the best; whereas, MINDM and MINRDM are superior to other mode

Table 1. Example of two-stage sampling method (MAX-J=30, θ=4.0, *DD=LCPL*)

Policy *i*		$\overline{X}_i^{(1)}(30)$	$S_i^2(30)$	N_i	$\overline{X}_i^{(1)}(N_i-30)$	W_{i1}	W_{i2}	$\tilde{X}_i$
A.P.R	M.P.R							
RAN-A	RAN-M	-402.1	4504.5	43	-405.7	0.73	0.27	-402.5
	MINDM	-347.2	5354.6	51	-331.7	0.63	0.37	-341.5
	STOCOM-M	-396.4	3210.0	31	-540.0	0.98	0.02	-399.1
	MAXDM	-536.4	7926.4	76	-502.7	0.41	0.59	-516.7
	MINRDM	-202.2	4139.1	39	-151.3	0.83	0.17	-193.4
STOCOM-A	RAN-M	-369.6	4525.1	43	-396.0	0.74	0.26	-376.3
	MINDM	-336.0	4404.4	42	-348.0	0.75	0.25	-339.0
	STOCOM-M	-383.0	6475.2	62	-384.0	0.51	0.49	-383.5
	MAXDM	-507.0	9721.9	93	-518.9	0.35	0.65	-514.7
	MINRDM	-202.2	2726.4	31	-234.0	1.04	-0.04	-200.8
MINLFT	RAN-M	-473.0	15321.3	147	-460.8	0.22	0.78	-463.5
	MINDM	-474.0	0.0	31	-474.0	1.00	0.00	-474.0
	STOCOM-M	-435.0	30060.6	288	-454.3	0.12	0.88	-452.0
	MAXDM	-684.0	0.0	31	-684.0	1.00	0.00	-684.0
	MINRDM	-114.0	0.0	31	-114.0	1.00	0.00	-114.0
MINSL	RAN-M	74.0	1569.9	31	96.0	1.14	-0.14	70.7
	MINDM	168.0	0.0	31	168.0	1.00	0.00	168.0
	STOCOM-M	73.2	1572.6	31	96.0	1.15	-0.15	69.8
	MAXDM	-18.0	0.0	31	-18.0	1.00	0.00	-18.0
	MINRDM	96.0	0.0	31	96.0	1.00	0.00	96.0

Note: A.P.R denotes activity priority rule; M.P.R., mode priority rule

Table 2. Weighted sample means of project duration (MAX-J=20)

Activity priority rule	Mode priority rule				
	RAN-M	MINDM	STOCOM-M	MAXDM	MINDRM
			θ= 2.0, *DD=LCPL*		
RAN-A	-789.9	-743.9	-692.9	-1026.3	-481.0
STOCOM-A	-589.1	-546.8	-525.5	-821.6	-293.4
MINLFT	-1080.6	-996.0	-952.0	-1530.0	-756.0
MINSL	-36.8	**12.0**	11.1	-84.0	0.0
			θ= 2.0, *DD=3LCPL*		
RAN-A	-44.9	-47.4	69.9	-271.1	469.3
STOCOM-A	907.1	964.2	946.7	694.1	1171.1
MINLFT	-412.2	-282.0	-331.9	-780.0	-60.0
MINSL	1392.1	**1428.0**	1389.5	1248.0	1416.0
			θ= 4.0, *DD=LCPL*		
RAN-A	-153.0	-124.6	-153.4	-259.9	-48.1
STOCOM-A	-168.3	-153.3	-165.6	-252.0	-43.0
MINLFT	-337.5	-300.0	-344.8	-552.0	-120.0
MINSL	28.8	**108.0**	60.8	-48.0	108.0
			θ= 4.0, *DD=3LCPL*		
RAN-A	1107.7	1138.7	1150.4	887.2	1346.6
STOCOM-A	1352.2	1364.0	1414.7	1243.9	1459.6
MINLFT	676.5	732.0	815.9	348.0	1224.0
MINSL	1465.8	**1524.0**	1509.9	1320.0	**1524.0**

priority rules. Hence, for each scenario the corresponding priority rules are selected for use in a parallel scheduling algorithm to find the maximal average weighted profit.

3. Comparison of Parallel Heuristic Scheduling and Proposed Genetic Algorithms

In a GA, how to determine the values of various parameters such as population size, crossover rate, and

Table 3. Weighted sample means of project duration (MAX-J=30)

Activity priority rule	Mode priority rule				
	RAN-M	MINDM	STOCOM-M	MAXDM	MINDRM
			θ = 2.0, *DD=LCPL*		
RAN-A	-1406.7	-1360.7	-1304.3	-1665.0	-976.9
STOCOM-A	-1451.3	-1296.1	-1286.8	-1732.8	-950.2
MINLFT	-1553.1	-1248.0	-1301.9	-2112.0	-1092.0
MINSL	-64.4	-24.0	-12.6	**54.0**	48.0
			θ = 2.0, *DD=3LCPL*		
RAN-A	-426.8	-314.3	-348.9	-708.2	126.6
STOCOM-A	-106.1	-37.7	33.1	-370.9	470.5
MINLFT	-101.1	-12.0	-17.9	-342.0	324.0
MINSL	1883.3	1920.0	1886.7	1860.0	**2016.0**
			θ = 4.0, *DD=LCPL*		
RAN-A	-402.5	-341.5	-399.1	-516.7	-193.4
STOCOM-A	-376.3	-339.0	-383.5	-514.7	-200.8
MINLFT	-463.5	-474.0	-452.0	-684.0	-114.0
MINSL	70.7	**168.0**	69.8	-18.0	96.0
			θ = 4.0, *DD=3LCPL*		
RAN-A	1191.9	1283.4	1294.3	930.1	1585.1
STOCOM-A	1437.1	1552.2	1559.3	1186.8	1706.4
MINLFT	1399.1	1644.0	1498.0	1392.0	1656.0
MINSL	2039.0	**2136.0**	2118.3	1932.0	2064.0

Table 4. Weighted sample means of project duration (MAX-J=40)

Activity priority rule	Mode priority rule				
	RAN-M	MINDM	STOCOM-M	MAXDM	MINDRM
			θ = 2.0, *DD=LCPL*		
RAN-A	-2079.7	-1873.2	-1851.1	-2552.3	-1312.7
STOCOM-A	-2058.1	-1940.6	1886.2	2646.9	-1335.1
MINLFT	-967.4	-862.0	-826.7	-1368.0	-690.0
MINSL	-52.7	**60.0**	25.0	-90.0	-42.0
			θ = 2.0, *DD=3LCPL*		
RAN-A	-927.8	-720.1	-699.1	-1400.2	-158.1
STOCOM-A	-699.7	-493.6	-515.2	-1089.8	65.3
MINLFT	-495.6	-234.0	-216.8	-1044.0	-18.0
MINSL	2211.3	**2364.0**	2341.3	2124.0	2220.0
			θ = 4.0, *DD=LCPL*		
RAN-A	-627.3	-527.6	-619.8	-858.4	-335.4
STOCOM-A	-603.1	-536.1	-601.2	-883.8	-345.9
MINLFT	-240.7	-132.0	-236.4	-492.0	-306.0
MINSL	92.4	**240.0**	95.8	-60.0	36.0
			θ = 4.0, *DD=3LCPL*		
RAN-A	1065.1	1252.6	1257.8	567.2	1622.5
STOCOM-A	1299.8	1454.7	1471.6	891.5	1819.8
MINLFT	1132.9	1500.0	1367.4	624.0	1572.0
MINSL	2393.2	**2544.0**	2507.8	2184.0	2340.0

mutation rate is a very complicated problem. A good combination of GA parameters for producing a stable and robust result depends on an excellent design for the experiment; however, such a combination is beyond the scope of this study.

Once an MMRCMPSP instance has been generated, first solve it by the PSA, then by the GA, in which stage 40 chromosomes (i.e., schedules) are initialized as the first generation. Then find the maximal solution through 100 generations. For simplicity, a new generation with P_1=25, P_2=15, P_3=5 and P_4=5 is set (Here, these values of the

Table 5. Weighted sample means of project duration (MAX-J=50)

Activity priority rule	Mode priority rule				
	RAN-M	MINDM	STOCOM-M	MAXDM	MINDRM
			$\theta = 2.0$, *DD=LCPL*		
RAN-A	-2586.5	-2314.1	-2294.6	-3242.4	-1687.8
STOCOM-A	-2623.8	-2347.9	-2367.7	-3361.2	-1722.6
MINLFT	-1139.2	-1044.0	-1027.3	-1686.0	-378.0
MINSL	-54.6	36.0	35.3	-108.0	**72.0**
			$\theta = 2.0$, *DD=3LCPL*		
RAN-A	-1062.5	-790.1	-770.6	-1718.4	-108.7
STOCOM-A	-1135.1	-830.9	-838.6	-1875.5	-267.3
MINLFT	2537.5	2556.0	2584.4	2436.0	2880.0
MINSL	2939.7	3084.0	3029.9	2832.0	**3120.0**
			$\theta = 4.0$, *DD=LCPL*		
RAN-A	-770.5	-663.3	-638.6	-1073.0	-422.1
STOCOM-A	-803.2	-643.5	-637.9	-1075.2	-415.7
MINLFT	-433.2	-324.0	-360.6	-576.0	-546.0
MINSL	118.4	**312.0**	244.2	0.0	156.0
			$\theta = 4.0$, *DD=3LCPL*		
RAN-A	1475.7	1746.0	1738.4	907.1	2215.9
STOCOM-A	1336.2	1691.2	1652.2	731.3	2051.2
MINLFT	3011.5	3180.0	3123.0	2952.0	3156.0
MINSL	3162.7	**3360.0**	3303.5	3048.0	3204.0

Table 6. Best activity- and mode- priority rules for each problem instance

	$\theta = 2.0$				$\theta = 4.0$			
MAX-J	*DD = 3LCPL*		*DD=LCPL*		*DD = 3LCPL*		*DD=LCPL*	
	A	B	A	B	A	B	A	B
20	MINSL	MINDM	MINSL	MINDM	MINSL	MINDM	MINSL	MINDM
30	MINSL	MINRDM	MINSL	MINDM	MINSL	MINDM	MINSL	MINDM
40	MINSL	MINDM	MINSL	MINDM	MINSL	MINDM	MINSL	MINDM
50	MINSL	MINRDM	MINSL	MINRDM	MINSL	MINDM	MINSL	MINDM

Note: A represents the probably-best activity priority rule
B represents the probably-best mode priority rule

parameters are finally determined on the basis of some simulation effort. Even though the solutions obtained by using these values in the GA are not optimal one, they will be quasi-optimal). For convenience, the solutions obtained by the PSA (described in section 3) are denoted as *SPSA*. In the following subsection, the *SPSA* are compared with the *SGA* (solutions derived by the proposed genetic algorithm). To examine and evaluate the performance the proposed GA, a measure is defined as

$$\phi = (SGA - SPSA) / SPSA * 100\% \qquad (18)$$

Here different combinations of resource-availability levels (θ =2.0, 4.0) and two different due-date tight levels (*DD=LCPL*, *DD=3LCPL*) are considered. The early completion incentive value is assumed to be 4; the late completion penalty is 5. For each combination, given MAX-J=20, 30 MMRCPSP instances are generated. Each instance is solved by the proposed parallel scheduling and the proposed GA, the ϕ value being calculated. Classify these 30 ϕ values into three classes (i.e., $\phi > 10\%$, $10\% > \phi > 0\ \%$, $\phi < 0$) to evaluate the performance of the GA. The final results from a different MAX-J = 20 to 50 are summarized in Table 7.

It can be observed from Table 7 that most of the generated problem instances solved by the GA are better than those solved by the PSA. Moreover, many ϕ are greater than 10% in the majority of cases, except in the combination of $\theta =$ 4.0, *DD = 3LCPL*. From the computational results based on the different experimental scenarios and generated problem instances, the results obtained by the GA are better than those obtained by the PSA in most cases. Furthermore, as smaller value of θ is set (i.e., less resource availability can be

Table 7. Comparison of *SPSA* and *SGA*

			$\theta = 2.0$			
		DD=3LCPL			*DD=LCPL*	
MAX-J	$\phi \geq 10\%$	$10\% > \phi \geq 0\%$	$\phi < 0\%$	$\phi \geq 10\%$	$10\% > \phi \geq 0\%$	$\phi < 0\%$
20	18	11	1	18	8	4
30	5	16	9	13	11	6
40	2	18	10	17	7	6
50	24	3	3	6	16	8
			$\theta = 4.0$			
		DD=3LCPL			*DD=LCPL*	
MAX-J	$\phi \geq 10\%$	$10\% > \phi \geq 0\%$	$\phi < 0\%$	$\phi \geq 10\%$	$10\% > \phi \geq 0\%$	$\phi < 0\%$
20	0	27	3	27	2	1
30	0	25	5	20	1	9
40	0	17	13	17	7	6
50	0	22	8	10	5	15

obtained) and due date of project completion is set tighter; the proposed GA appears to outperform PSA more obviously. Therefore, a simple design of GA can play a role of solving a MMRCMPSP in relatively effective manner.

Considering the computation times of these two algorithms, the PSA is certainly superior to the GA owing to the number of their computational iterations. Under the same problem instance, constructing only one schedule is needed in the PSA, whereas, generating 30 schedules and performing 100 generations are needed in the GA. In case of θ = 2.0, *DD=LCPL*, MAX-J = 20, the average times of the PSA and the GA are 0.0549 (0.139) and 22.833 (52.410) (seconds), respectively. The values in the parentheses are the case of MAX-J = 50. Therefore, more efficient and robust GA is left for future study.

VI. CONCLUSION

In this study, two heuristic algorithms have been developed to solve a multi-mode resource-constrained multi-project scheduling problem (MMRCMPSP). The first is a parallel scheduling algorithm developed to find an approximately optimal solution; the second, a genetic algorithm. The proposed genetic algorithm is based on a direct representation of candidate solutions and new genetic operators. The MMRCMPSP itself has been used as a chromosome, and augmented recombination operators have been designed to work on the non-standard representation. All relevant domain information has been contained in the problem representation with the intention being that the genetic algorithm can operate on the entire search space.

The objective of the parallel scheduling algorithm has been used as baseline to validate the effectiveness and performance of the second algorithm. As a baseline, Identification of the best combination of activity- and mode-priority rules which can find most likely solution for a certain objective is needed. Due to certain priority rules with probabilistic factors, a two-stage sampling method has been introduced to ensure that more reliable computational results are obtained. Finally, a problem-instance generator has been devised to generate a certain number of test problems. A genetic algorithm proposed to search a better solution of MMRCMPSP is one of the objectives of the paper, since so far there seems no good heuristic algorithm in public literature. Hence, a simple genetic algorithm is initialized; more efficient and robust GA is left for future study. From the computational results, the effectiveness and performance of the proposed genetic algorithm is validated.

ACKNOWLEDGMENTS

The author expresses appreciation to Dr. Cheryl Rutledge, Department of English, Da-Yeh University, for her editorial assistance, and two anonymous referees for their valuable comments and suggestions.

REFERENCES

1. Alcaraz, J., C. Maroto and R. Ruiz (2003) Solving the multi-mode resource-constrained project scheduling problem with genetic algorithms. *Journal of the Operational Research Society*, 54, 614-626.
2. Blazewicz, J., L. J. Kenstra and A. H. G. Kan (1983) Scheduling subject to resource constraints: Classification and complexity. *Discrete Applied Mathematics*, 5, 11-24.
3. Boctor, F. F. (1993) Heuristics for scheduling projects with resource restrictions and several resource-duration modes. *International Journal of Production Research*, 31(11), 2457-2558.

4. Boctor, F. F. (1996) A new and efficient heuristic for scheduling projects with resource restrictions and multiple execution modes. *European Journal of Operational Research*, 90, 349-361.
5. Bouleimen, K. and K. Lecocq (2003) A new efficient simulated annealing algorithm for resource-constrained project scheduling problem and its multiple mode version. *European Journal of Operational Research*, 149, 268-281.
6. Davis, E. W. and J. H. Patterson (1975) A comparison of heuristic and optimum solutions in resource-constrained project scheduling. *Management Science*, 21(8), 944-955.
7. Demeulemeester, E., B. Dodin and W. Herroelen (1993) A random activity network generator. *Operations Research*, 41, 972-980.
8. Drexl, A. and J. Gruenewald (1993) Non-preemptive multi-mode resource-constrained project scheduling. *IIE Transactions*, 25, 74-81.
9. Elmaghrary, S. E. (1977) *Activity Networks: Project Planning and Control by Network Models*, Wiley, New York, NY.
10. Fendley, L. G. (1968) Toward the development of a complete multi-project scheduling system. *Journal of Industrial Engineering*, 19(10), 505-515.
11. Goldberg, D. E. (1989) *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison Wesley, Reading, MA.
12. Goncalves, J. F, J. J. M. Mendes and M. G. C. Resende (2004) *A Genetic Algorithm for the Resource Constrained Multi-Project Scheduling Problem*, Technical Report: TD-668LM4. AT&T Labs Research.
13. Hartman, S. (1998) A competitive genetic algorithm for resource-constrained project scheduling. *Naval Research Logistics*, 45, 733-750.
14. Holland, J. H. (1976) *Adaptation in Nature and Artificial Systems*, University of Michigan Press, Ann Arbor, MI.
15. Kolisch, R. and R. Padman (2001) An integrated survey of deterministic project scheduling. *Omega*, 29, 249-272.
16. Kurtulus, I. S. and E. W. Davis (1982) Multi-project scheduling: Categorization of heuristic rules performance. *Management Science*, 28, 161-172.
17. Kurtulus, I. S. and S. C. Narula (1985) Multi-project scheduling: Analysis of project performance. *IIE Transactions*, 17, 58-66.
18. Law, A. M. and W. D. Kelton (1991) *Simulation Modeling and Analysis*, MacGraw-Hill, Singapore.
19. Lee, C. Y. and L. Lei (2001) Multiple-project scheduling with controllable project duration and hard resource constraints: Some solvable cases. *Annals of Operations Research*, 102, 287-307.
20. Lova, A., P. Tormas and F. Barber (2006) Multi-mode resource constrained project scheduling: Scheduling schemes, priority rules and mode selection rules. *Inteligencia Artificial*, 30(10), 69-86.
21. Lova, A., C. Maroto and P. Tormas (2000) A multicriteria heuristic method to improve resource allocation in multiproject scheduling. *European Journal of Operational Research*, 127, 408-424.
22. Lova, A. and P. Tormas (2001) Analysis of scheduling schemes and heuristic rules performance in resource-constrained project scheduling. *Annals of Operations Research*, 102, 263-286.
23. Kim, K. W., Y. Yun, J. Yoon, M. Gen and G. Yamazaki (2005) Hybrid genetic algorithm with adaptive abilities for resource-constrained multiple project scheduling. *Computers in Industry*, 56, 143-160.
24. Law, A. M. and W. D. Kelton (1991) *Simulation Modeling and Analysis*, MacGraw-Hill, Singapore.
25. Mori, M. and C. C. Tseng (1997) A genetic algorithm for multi-mode resource constrained project scheduling problem. *European Journal of Operational Research*, 100, 134-141.
26. Pritsker, A. A. B, L. J. Watters and P. M. Wolfe (1969) Multiproject scheduling with limited resources: A zero-one programming approach. *Management Science*, 16, 93-108.
27. Sprecher, A. and A. Drexl (1998) Multi-mode resource-constrained project scheduling by a simple, general and powerful sequencing algorithm. *European Journal of Operational Research*, 107, 431-450.
28. Talbot, F. B. (1982) Resource-constrained project scheduling with time-resource tradeoffs: The nonpreemptive case. *Management Science*, 28, 1197-1210.

Received: Jul. 05, 2007 Revised: Dec. 18, 2007
Accepted: Mar. 10, 2008