Hindawi Publishing Corporation
Advances in Operations Research
Volume 2014, Article ID 740670, 10 pages
http://dx.doi.org/10.1155/2014/740670

Research Article

Multimode Preemptive Resource Investment Problem Subject to Due Dates for Activities: Formulation and Solution Procedure

Behrouz Afshar-Nadjafi and Mohammad Arani

Department of Industrial Engineering, Faculty of Industrial and Mechanical Engineering, Islamic Azad University, Qazvin Branch, Qazvin 34185-14161, Iran

Correspondence should be addressed to Behrouz Afshar-Nadjafi; afsharnb@alum.sharif.edu

Received 29 June 2014; Revised 13 September 2014; Accepted 16 September 2014; Published 29 September 2014

Academic Editor: Walter J. Gutjahr

The preemptive Multimode resource investment problem is investigated. The Objective is to minimize the total renewable/nonrenewable resource costs and earliness-tardiness costs by a given project deadline and due dates for activities. In this problem setting preemption is allowed with no setup cost or time. The project contains activities interrelated by finish-start type precedence relations with a time lag of zero, which require a set of renewable and nonrenewable resources. The problem formed in this way is an NP-hard. A mixed integer programming formulation is proposed for the problem and parameters tuned genetic algorithm (GA) is proposed to solve it. To evaluate the performance of the proposed algorithm, 120 test problems are used. Comparative statistical results reveal that the proposed GA is efficient and effective in terms of the objective function and computational times.

1. Introduction

The resource constrained project scheduling problem (RCPSP) is a known combinatorial optimization problem which is NP-hard [1]. The objective of RCPSP is to minimize the makespan of the project while the renewable resources availabilities are considered given. In the literature there are several exact methods and heuristics that solve the RCPSP [2–9].

In the multimode version of the RCPSP (MRCPSP), each activity can be performed in one out of a set of modes, with a specific activity duration and resource requirements. The problem involves the selection of a mode for each activity and the determination of the activity start or finish times such that project makespan is minimized. However, the precedence and resource constraints should be satisfied. MRCPSP is also NP-hard as generalization of the RCPSP. In recent years several algorithms have been proposed to solve MRCPSP [10–20].

The resource investment problem (RIP) is a close variant of RCPSP. This problem consists of determination of the activities start times and renewable resources availabilities such that the total cost of the resources is minimized subject to a given project deadline. In RIP, project's makespan is not forgotten but it is controlled with a predetermined deadline as a constraint. This problem was introduced by Möhring [21]. He showed that the problem is NP-hard. Rangaswamy [22] proposed a B&B for the RIP and applied it to the same instance set used by Demeulemeester [23]. Drexl and Kimms proposed two lower bound procedures for the RIP based on Lagrangian relaxation and column generation methods [24]. Other exact procedures are proposed by Demeulemeester [23] and Rodrigues and Yamashita [25]. Yamashita et al. [26] proposed a multistart heuristic based on the scatter search. Shadrokh and Kianfar presented a genetic algorithm (GA) to solve the RIP when tardiness of the project is permitted with penalty [27]. Ranjbar et al. developed a path relinking procedure and a genetic algorithm (GA) [28]. Van Peteghem and Vanhoucke presented an artificial immune system algorithm for the problem [29].

The basic project scheduling problems assume that each activity, once started, will be executed until its completion.

Preemptive project scheduling problem refers to the scheduling problem which allows activities to be preempted at any time instance and restarted later on at no additional cost or time. In the case in which activities are preemptive, modeling and solving such a problem as a classical nonpreemptive RCPSP or RIP may lead to poor solutions. Such situations where preemption of an activity is beneficial or necessary are typical, for example, in industry processes where processing units, like reactors or filters, have to be cleaned after the completion of certain subactivities. In the textile industry, preemptions are inevitable. When the fabric type is changed on a machine, the wrap chain must be replaced which indicates the necessity of preemption. The literature on solution methods for the preemptive project scheduling problems is relatively scant. For the preemptive RCPSP, we refer to [30–34]. For the preemptive MRCPSP, Buddhakulsomsiri and Kim proved that preemption is very effective to improve the optimal project makespan in the presence of resource vacations and temporary resource unavailability [35]. Van Peteghem and Vanhoucke have proposed a genetic algorithm for the MRCPSP and its extension to the preempted case [36].

Minimization of the earliness-tardiness penalties of the project activities is a nonregular performance measure in just-in-time (JIT) environments. In this problem, activities have an individual due date with associated unit earliness and unit tardiness penalty. Traditionally, tardiness penalties due to delivery after a contractually arranged due date are considered. Tardiness leads to customer complaints, loss of reputation and profits, monetary penalties, or goodwill damages. Costs of earliness include extra storage requirements, idle times, implicitly incur opportunity costs, and storage costs due to insurance, theft, perishing, and bounded capital for the case of an activity is completed before the due date. Therefore, the minimization of the earliness-tardiness (E/T) costs is attractive in order to meet the requests of practice. For some recent solution procedures considering earliness-tardiness penalties we refer to [37–40].

There are some shortcomings in the basic RIP which is considered in this paper simultaneously. First, activities in the basic RIP are assumed nonpreemptive, while this assumption is not true in practice. Second, in the basic RIP, the determination of only renewable resources availabilities is investigated. Third, the basic RIP supposes single execution mode for activities. Last, due dates for activities are neglected. Therefore, the contribution of this paper is threefold: first, a mixed integer programming formulation is developed for the preemptive multimode RIP with due dates for the activities. We call this problem P-MRIP-WET. This model is not considered in the past literature. Second, a new efficient parameter-tuned GA is developed for the problem due to NP-hardness of the problem. Finally, the effectiveness of the proposed method to solve the P-MRIP-WET is analyzed statistically.

The reminder of the paper is organized as follows. Section 2 describes the problem P-MRIP-WET and mixed integer formulation for it. Section 3 explains the steps of proposed genetic algorithm (GA) to solve the problem. Section 4 contains the computational results and performance evaluation of proposed GA. Finally, Section 5 concludes the paper.

2. Problem Description

The preemptive multimode resource investment problem with weighted earliness-tardiness penalties (P-MRIP-WET) involves the scheduling of project activities on a set K of renewable resource types and a set K' of nonrenewable resource types. Each activity i is performed in a mode m_i, which is chosen out of a set of M_i different execution modes, that is, with different durations and resource requirements. The duration of activity i, when executed in mode m_i, is d_{im_i}. Each activity i in mode m_i requires $r^{\rho}_{im_ik}$ units of renewable resource type k ($k = 1, \ldots, K$) during each time unit of its execution. For each renewable resource k, the availability R^{ρ}_{k} is constant throughout the project horizon. Activity i, executed in mode m_i, will also use $r^{\nu}_{im_ik'}$ nonrenewable resource units ($k' = 1, \ldots, K'$) of the total available nonrenewable resource $R^{\nu}_{k'}$.

In sequent, assume a project represented in AON format by a directed graph $G = \{N, A\}$ where the set of nodes, N, represents activities and the set of arcs, A, represents finish-start precedence constraints with a time lag of zero. The preemptable activities are numbered from the dummy start activity 0 to the dummy end activity $n + 1$ and are topologically ordered; that is, each successor of an activity has a larger activity number than the activity itself.

Also, we consider discrete time points for the preemption. As a rule, a preemptive problem is characterized by a complicated structure of its optimal solutions. When preemption is allowed at arbitrary times, the problem turns out to be intractable [41]. In this situation, when the overall number of preemptions is unlimited and the set of admissible points for preemptions has continuous cardinality, we cannot utilize exact enumerative algorithms, unless a nontrivial preliminary analysis of properties of optimal solutions is performed. Such analysis reduces the original infinite set of possible points for preemption to a finite set. This reduction allows us to solve the problem by direct enumeration. This analysis is performed for some scheduling problems. Baptiste et al. prove that a wide class of preemptive scheduling models, including both machine and project scheduling models, have the "*integer preemption property*"; for any problem instance with integral input data, there exists an optimal schedule where all preemptions (as well as starting and completion times of jobs/activities) occur at integer time points. This conclusion is held for objective functions such as total weighted earliness-tardiness and total weighted number of late jobs [42, 43].

The objective of the P-MRIP-WET is to schedule a number of activities, in order to minimize the total cost of the resources availabilities and earliness-tardiness penalties. A schedule S is defined by a vector of activity start times and is said to be feasible if all precedence relations, project deadline, and renewable and nonrenewable resource constraints are satisfied. However, execution modes m_i for activities, available resource capacities, and start times for activities have to be determined. We have the Notations section for P-MRIP-WET.

The variables x_{im_it} can only be defined over the time interval of the activity in question $t \in [\mathrm{EST}_i, \mathrm{LFT}_i - 1]$. These limits are determined using the traditional forward and backward pass calculations considering duration of activity i based on the execution mode with lowest duration. The backward pass calculation is started from a fixed project deadline DL. In this paper, earliest finish time of dummy end activity EFT_{n+1} multiplied by 1.3 is considered as project deadline. EFT_{n+1} is computed using the traditional forward calculations considering duration of activity i based on execution mode with highest duration.

It is clear that an activity with duration of 0 is never in progress and thus does not have a corresponding decision variable which is set to 1. This problem, however, can be easily overcome: the dummy start and end activity are assigned a dummy mode with duration of 1. Also, the other parameters for dummy modes are assumed 0. All other activities with zero duration can be eliminated, provided that the corresponding precedence relations are adjusted appropriately. The resulting schedule may be transferred into a schedule for the original problem by removing the dummy start and end activity, and one time unit left shifting.

Using the above notation, the P-MRIP-WET can be mathematically formulated as follows:

$$\begin{aligned}\mathrm{Min}\ Z = &\sum_{i=1}^{n} \left(e_i * E_i + t_i * T_i\right) \\ &+ \sum_{k=1}^{K} \left(C_k^{\rho} * R_k^{\rho}\right) + \sum_{k'=1}^{K'} \left(C_{k'}^{\nu} * R_{k'}^{\nu}\right).\end{aligned} \tag{1}$$

The objective in (1) is to minimize the total cost of the project resources and earliness-tardiness penalties. First part is weighted earliness-tardiness cost. Second part is costs related to renewable resources which are available and renewable from period to period. Only the total resource use at every time instant is constrained. Third part is costs related to nonrenewable resources which are available on a total project basis, with limited consumption availability for the entire project. Consider the following:

$$\begin{aligned}&\sum_{m_i=1}^{|M_i|} x_{im_it} \le 1, \quad i = 1, \ldots, n, \\ &t = \mathrm{EST}_i, \ldots, \mathrm{LFT}_i - 1.\end{aligned} \tag{2}$$

Equation (2) guarantees that each activity i can be in progress with at most one mode in each time unit:

$$\begin{aligned}&\sum_{t=\mathrm{EST}_i}^{\mathrm{LFT}_i - 1} x_{im_it} = d_{im_i} * y_{im_i}, \\ &i = 1, \ldots, n, \quad m_i = 1, \ldots, \left|M_i\right|.\end{aligned} \tag{3}$$

Equation (3) ensures that each activity i should be in progress for d_{im_i} time units in assigned mode m_i:

$$\sum_{m_i=1}^{|M_i|} y_{im_i} = 1, \quad i = 1, \ldots, n. \tag{4}$$

Equation (4) specifies that only one execution mode is allowed for each activity i:

$$\begin{aligned}&\sum_{i=1}^{n} \sum_{m_i=1}^{|M_i|} r_{im_ik}^{\rho} * x_{im_it} \le R_k^{\rho}, \\ &t = \mathrm{EST}_i, \ldots, \mathrm{LFT}_i - 1, \ k = 1, \ldots, K.\end{aligned} \tag{5}$$

Constraint set in (5) takes care of the renewable resources limitations:

$$\sum_{i=1}^{n} \sum_{m_i=1}^{|M_i|} r_{im_ik'}^{\nu} * y_{im_i} \le R_{k'}^{\nu}, \quad k' = 1, \ldots, K'. \tag{6}$$

Constraint set in (6) takes care of the nonrenewable resources limitations:

$$\begin{aligned}&S_{i\,\mathrm{Min}} \le t * x_{im_it} + \lambda * \left(1 - x_{im_it}\right), \\ &i = 1, \ldots, n, \ m_i = 1, \ldots, \left|M_i\right|, \\ &t = \mathrm{EST}_i, \ldots, \mathrm{LFT}_i - 1.\end{aligned} \tag{7}$$

Constraint set in (7) computes the start time of first time unit of activity i:

$$\begin{aligned}&S_{i\,\mathrm{Max}} \ge t * x_{im_it}, \quad i = 1, \ldots, n, \\ &m_i = 1, \ldots, \left|M_i\right|, \ t = \mathrm{EST}_i, \ldots, \mathrm{LFT}_i - 1.\end{aligned} \tag{8}$$

Constraint set in (8) computes the start time of last time unit of activity i:

$$S_{i\,\mathrm{Max}} + 1 \le S_{j\,\mathrm{Min}}, \quad \forall (i, j) \in A. \tag{9}$$

Equation (9) denotes the finish to start zero time lag precedence relations constraints:

$$E_i = \mathrm{Max}\left\{0, \mathrm{DD}_i - \left(S_{i\,\mathrm{Max}} + 1\right)\right\}, \quad i = 1, \ldots, n. \tag{10}$$

Equation (10) computes the earliness for each activity i:

$$T_i = \mathrm{Max}\left\{0, \left(S_{i\,\mathrm{Max}} + 1\right) - \mathrm{DD}_i\right\}, \quad i = 1, \ldots, n. \tag{11}$$

Equation (11) computes the tardiness for each activity i:

$$S_{n+1,\mathrm{Max}} \le \mathrm{DL}. \tag{12}$$

Constraint in (12) guarantees that project deadline is not violated:

$$\begin{aligned}&y_{im_i} \in \{0, 1\}, x_{im_it} \in \{0, 1\}, \quad i = 1, \ldots, n, \\ &m_i = 1, \ldots, \left|M_i\right|, \ t = \mathrm{EST}_i, \ldots, \mathrm{LFT}_i - 1.\end{aligned} \tag{13}$$

Equation (13) specifies that the decision variables y_{im_i} and x_{im_it} are binary:

$$\begin{aligned}&E_i, T_i, R_k^{\rho}, R_{k'}^{\nu}, S_{i\,\mathrm{Min}}, S_{i\,\mathrm{Max}} \in \mathrm{int}^{+}, \\ &i = 1, \ldots, n, \ k = 1, \ldots, K, \ k' = 1, \ldots, K'.\end{aligned} \tag{14}$$

Equation (14) specifies that the decision variables E_i, T_i, $R_k^{\rho}, R_{k'}^{\nu}, S_{i\,\mathrm{Min}}$ and $S_{i\,\mathrm{Max}}$ are integers.

3. Proposed GA to Solve P-MRIP-WET

In this section we describe the genetic algorithm (GA) which is the well-known metaheuristic that has been successfully applied to a noticeable number of project scheduling problems [6, 15, 18, 19, 27, 36, 38], to solve P-MRIP-WET. In order to increase quality of the proposed GA, we implement two efficient local searches. We also consider results obtained from CPLEX software version 12.3 and randomly generated feasible solutions, to provide comparable computational efforts for the proposed GA.

Solution representation, selection, and reproduction are the basic elements of GA. These elements must be well defined and adapted to a specific problem. Before the execution of the proposed GA all nonexecutable and inefficient modes can be omitted in order to reduce the search space [44]. An execution mode m_i is called nonexecutable if its execution would violate the renewable resource constraints in any schedule. Also, a mode is called inefficient if there is another mode of the same activity with the same or higher duration and no more requirements for all resources.

3.1. Solution Representation. Two important representations for schedules are the random-key (RK) and the activity-list (AL) representation. It is deduced from experimental tests that procedures based on (AL) representation outperform the other procedures [45]. In this study the AL representation is used to encode a project schedule and the parallel schedule generation scheme (PSGS) to decode the schedule representation to a schedule. We represent a feasible solution by an $n + N' + K + K'$ element vector (I).

The first n element represents execution modes for activities (mode list). In the sequel it is assumed that each original activity i with assigned mode m_i is replaced with d_{im_i} activities with unit duration. Each duration unit $j = 1, \ldots, d_{im_i}$ of an activity i is predecessor of duration unit $(j-1)$. Consequently, project new network has $N' = \sum_{i=1}^{n} d_{im_i}$ activities with duration 1 and same resource requirements as original activities. Only duration unit $d_{im_i}^{\text{th}}$ has fixed due date DD_i. Also, without loss of generality, for duration units $j = 1, \ldots, (d_{im_i} - 1)$ of an activity i due dates are assumed 0 with no earliness-tardiness penalty.

The second N' element represents a precedence-feasible permutation of activities (activity list). The third K element is renewable resource capacities (resource list). The last K' element is nonrenewable resource capacities which is determined based on assigned modes in mode list. Consider the following:

$$I = \left\{ \left(m^{J^1}, m^{J^2}, \ldots, m^{J^n} \right), \left(J^1, J^2, \ldots, J^{N'} \right), \left(R_1^{\rho}, R_2^{\rho}, \ldots, R_K^{\rho} \right), \left(R_1^{\upsilon}, R_2^{\upsilon}, \ldots, R_{K'}^{\upsilon} \right) \right\}. \quad (15)$$

Having obtained a feasible solution represented by the vector described above, the starting times of all activities are defined by parallel SGS. The SGS determines how a feasible schedule is constructed by assigning a starting time to the activities with respect to the assigned modes and the resource availabilities. The PSGS sequentially iterates over the different decision points at which activities can be added to the schedule until a feasible complete schedule is obtained. At each decision point, the unscheduled activities whose predecessors have completed are considered (in the order of the activity list) and are scheduled with assigned modes (specified in the mode list) such that resource availabilities (specified in the resource list) are not violated.

3.2. Initial Generation. The GA starts with producing the initial generation. As used in the literature [19], the initial generation can be created randomly. In proposed GA, each solution in initial generation is randomly created as follows. First, we assign a randomly selected mode to each activity i. Second, we generate an empty $N' = \sum_{i=1}^{n} d_{im_i}$ elements vector. To fill this vector an activity is randomly selected from set of activities that are not selected before and all their precedence are already met. This process is repeated until the activity list is full. Third, renewable resource k availability is randomly selected between lower and upper bound of R_k^{ρ}. Finally, the nonrenewable resource capacity k' is computed based on assigned modes; that is, summation of resource requirement of the activities in the assigned modes for nonrenewable resource k' is considered as $R_{k'}^{\upsilon}$. The capacity of the renewable resource type k cannot be less than $\underline{R_k^{\rho}} = \text{Max}_i\{\text{Min}_{m_i}(r_{im_ik}^{\rho})\}$, while the upper bound of the capacity of the renewable resource type k, $\overline{R_k^{\rho}}$, can be determined by unconstrained scheduling of activities at their earliest start times.

3.3. Reproduction. Once the initial generation is created, the process continues in an iterative way in order to obtain the next high quality generation. The better solutions of the generation will have higher possibilities to survive for the reproduction. In proposed GA, the reproduction process consists of four phases. In the first one, the codified segments of two selected solutions are exchanged following the crossover operators. In the second phase, the selected solutions are randomly changed using the mutation operators. In the third phase, some improved solutions created by knowledge based local searches are inserted to the next generation. In the last phase, the current generation is sorted by best to worst fitness function. Then P_r percent top solutions from the current generation are copied to the next generation. In this way, these operators lead to the exploitation of good solutions and to the exploration of new areas of search space.

The selected solutions for crossover and mutation operators are called parents. The parents are selected based on their fitness values. We used the roulette wheel scheme, where each solution has a probability of selection that is directly proportional to its fitness value. The selected solutions are grouped in couples randomly. Then the crossover operators act with a probability P_c for each couple, and the mutation operators act on each child created by the crossover operators with a probability P_m. However, P_{LS} percent of improved solutions created by two local searches are inserted to the next generation. In order to preserve the constant population size, we consider $P_r = (1 - P_c - P_m - P_{\text{LS}})$.

3.4. Crossover Operators. Crossover operators, depend on the type of solution representation. In the proposed GA, one of the following operators randomly acts on the selected couple.

(i) Two integers v and w are randomly selected from set $\{1, \ldots, n\}$. Then the execution modes of original activities between v and w are crossed to form other two solutions. Subsequently, the activity list and resource capacities are updated.

(ii) Two integers p and q are randomly selected from set $\{1, \ldots, K\}$. Then the renewable resource availabilities p and q are crossed to form other two solutions.

(iii) Two integers u and z are randomly selected from set $\{1, \ldots, N'\}$. Then the activity list of parent solutions are divided in three parts and they are combined to form two news solutions (Figure 1). A, B, and C are three parts (subset of activities) of parent 1 and D, E, and F are three parts (subset of activities) of parent 2. First part (D) of child solution 1 is copied from parent solution 2. The next parts (a and b) of child 1 are taken from two first parts of parent solution 1 (A and B) in the same order. However, the activities that have already been added from the parent 2 may not be considered again. Then the nonrepetitive activities in C from parent solution 1 which have not appeared in F from parent solution 2 are added to part c of child 1. Finally, the activities that have not existed in child 1 are taken from the part F of parent 2 in the same order (f). The activity list of child 2 is produced similarly from parent 1 and parent 2. This operator guarantees the feasibility of the new children solutions because, as it is clear in Figure 1, the priority of activities in child 1 is D-a-b-c-f which preserves the priority of activities in parent 1 (A-B-C) and parent 2 (D-E-F). Similarly, the priority of activities in child 2 is A-d-e-f-c which preserves the priority of activities in parent 1 (A-B-C) and parent 2 (D-E-F).

3.5. Mutation Operators. In the proposed GA, one of the following operators randomly acts on the selected child created with the crossover operations.

(i) An integer i is randomly selected from set $\{1, \ldots, n\}$. Then an execution mode different from the current mode of original activity J^i is assigned to it. Subsequently, the activity list and resource capacities are updated.

(ii) An integer j is randomly selected from set $\{1, \ldots, N'\}$. Then two predecessors of activity J^j are randomly selected and their positions in activity list are exchanged. Also, some activities between these positions are shifted to left or right such that feasibility of resulting solution is preserved.

3.6. Intelligent Local Searches. The use of local searches can improve the evolution speed of GA. Two following knowledge based local searches are employed on selected parents.

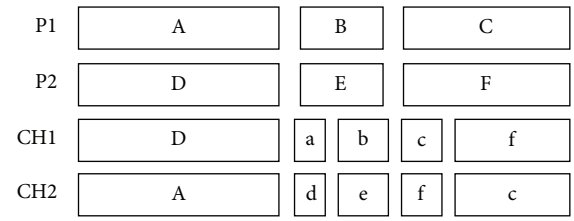

Figure 1: Third crossover operator.

(i) Resource Based Local Search (LS1). The percentage of project activities which are on time or have earliness (denoted by g) is computed. A random real number (denoted by r) is selected from interval $(0, 1)$. Then, an integer y is randomly selected from set $\{1, \ldots, K\}$. If $r \leq g$ the availability of renewable resource y is reduced (else it is increased) one unit. If the current availability of renewable resource y is equal to lower (upper) bound of R_y^ρ, a different resource is randomly selected. This procedure is continued until the fitness of new solution is better than the current solution (and project deadline is not violated) or all resources are equal to lower (upper) bounds. The idea behind this operator is to balance between earliness-tardiness cost and the cost of the renewable resources. In doing so, a renewable resource is randomly selected and reduced (increased) with a probability which has direct (inverse) relation with the percentage of project activities which are on time or have earliness. Clearly, a low amount of resource availabilities leads to more tardiness and less earliness and resources cost.

(ii) Activity Based Local Search (LS2). This local search is applied if at least one activity with earliness exists in current solution. All activities with earliness are identified and their start times are calculated by parallel SGS. Let s_f and s_l denote the start time of first and last activities which have earliness, respectively. A random integer x is selected from set $\{s_f, \ldots, s_l - 1\}$. Suppose x' denotes the first time interval larger than x which is empty without any in-progress activity. Time interval $[x, x + 1]$ is left empty without any in-progress activity and all activities between x and x' are shifted to right one unit time such that x' is full. The fitness of new solution is evaluated. If it is better than the current solution and project deadline is not violated, it replaces the current one.

3.7. Parameters Tuning. The performance of metaheuristics depends excessively on the value of their parameters. In this paper the Taguchi experimental design is used to tune the parameters of GA. Taguchi divides factors into controllable and noise factors and offers a set of orthogonal arrays for designing experiments of quality improvement. Although there is no direct control of noise factors, the Taguchi method determines the optimal level of controllable factors and minimizes the effect of noise [46]. In the proposed GA, the factors that should be tuned are nPop, nIt, P_c, P_m, and P_{LS}, where nPop is the population size, nIt is the number of

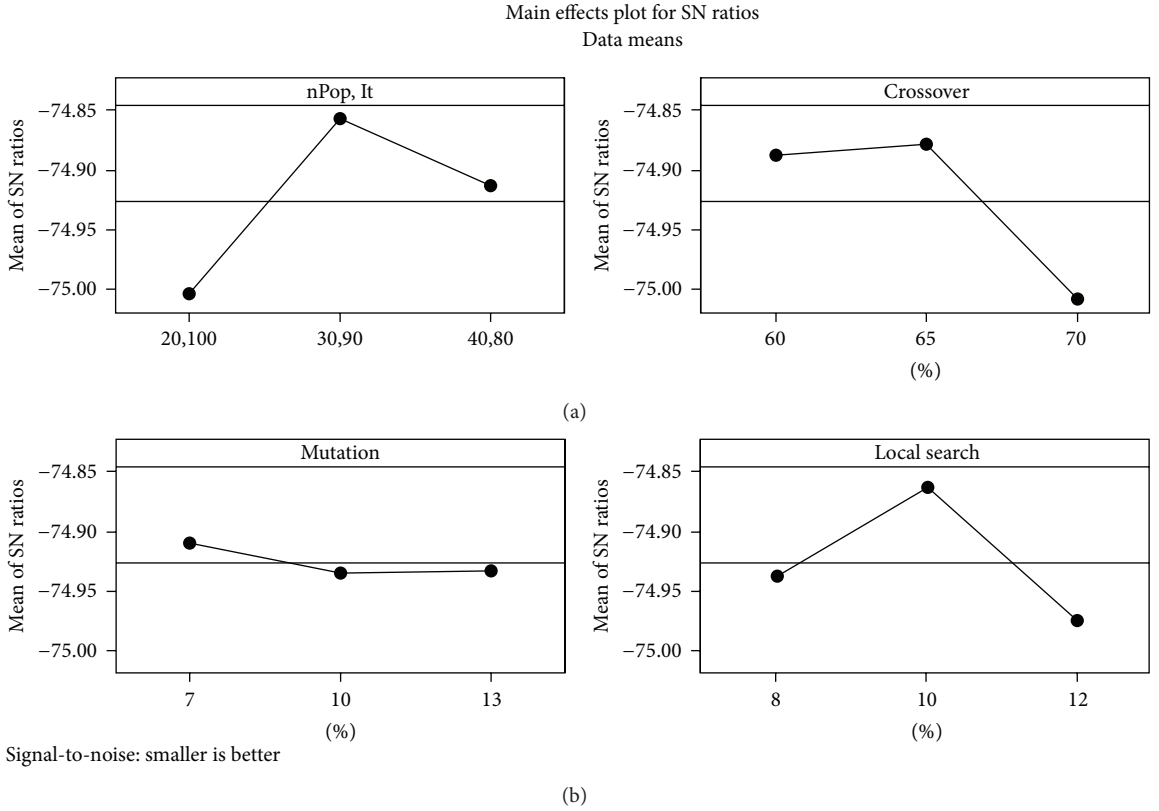

Figure 2: The mean S/N ratio plot for each level of the factors.

iterations (generations), and P_c, P_m, and P_{LS} are the crossover, mutation, and local search probabilities. A set of 27 randomly generated problems with 20 nondummy activities are used for parameter tuning. Using MINITAB software version 16, based on a L27 orthogonal array design, the average S/N ratio is obtained at each level (Figure 2). Different levels of these factors and the optimal levels (in bold) of *n*Pop, *n*It, P_c, P_m, and P_{LS} are shown in Table 1.

Table 1: Factors and factors levels.

Factor	Number of levels	Level 1	Level 2	Level 3
*n*Pop, *n*It	3	20, 100	**30, 90**	40, 80
P_c	3	60%	**65%**	70%
P_m	3	**7%**	10%	13%
P_{LS}	3	8%	**10%**	12%

3.8. Stopping Criterion. The procedure is continued until a predetermined number of generations are produced. We obtained good results by indexing the number of generations to the size of the problem, that is, use of the small number of generations for small problems and large number of generations for larger problems. So, the tuned values of *n*Pop and *n*It (90, 30) are used for problems with about 20 activities. However, these values are increased proportionally for larger sizes.

4. Performance Evaluation

4.1. Test Problems. In order to validate the proposed GA algorithm, a set of 120 problems was generated by the generator ProGen developed by Drexl et al. [47], using the parameters given in Table 2.

For each combination of problem parameters 5 problems were generated. The deadlines were generated in the same way as described in Section 2 by setting $DL = 1.3 * EFT_{n+1}$. The due dates were generated in the same way as Vanhoucke et al. [37]. First, a maximum due date was obtained by multiplying the critical path length by 1.5 for each project. Then, random numbers generated between 1 and maximum due date. The numbers are sorted and assigned to the activities in increasing order.

4.2. Experimental Results. The proposed GA were coded in Borland C++ 5.02 and executed on a personal computer with an Intel Core2Dou, 2.5 GHz processor, and 3 GB memory. Table 3 present the computational results of the proposed algorithm. For problems with 20 subactivities, the results are compared with the optimal solutions obtained by CPLEX. However, for problems with 50 and 80 subactivities which CPLEX is unable to solve, the results are compared with the best randomly generated solution (BRGS). Proposed GA

Table 2: The parameter settings for the problem set.

Control parameter	Value
Number of nondummy activities (average number of subactivities)	4 (20), 10 (50), 15 (80)
Number of execution modes	2, 3
Number of renewable resource types	2, 3
Number of nonrenewable resource types	2, 3
Activity durations	$\{1, \ldots, 10\}$
Unit earliness and tardiness costs	[1, 20]
Unit renewable and nonrenewable resource costs	[1, 20]
Number of initial and terminal activities	3
Maximal number of predecessors and successors	3
Coefficient of network complexity (CNC)	1.5
Activity renewable resource (per period) demand	$\{0, \ldots, 10\}$
Activity nonrenewable resource (per period) demand	$\{0, \ldots, 10\}$

executed 10 times for each problem to obtain more reliable data. The following notations are used in Table 3:

NPOC: number of problems optimally solved by CPLEX in 1000 second;

NPMG: number of runs of problems for which GA was able to find optimum solution;

ACT-CPLEX: average convergence time for CPLEX (in seconds);

ACT-GA: average convergence time for the GA (in seconds);

NBRGS: number of runs for which BRGS was able to conquest the GA at same CPU time;

ARD-GA: average relative deviation percentages for the GA;

ARD-BRGS: average relative deviation percentages for the BRGS.

Relative deviation (RD) percentage for each problem is obtained by the following formula:

$$\mathrm{RD} = \frac{Z - Z^*}{Z^*}, \tag{16}$$

where Z is the value of objective function and Z^* is the optimal solution obtained by CPLEX. In the cases for which CPLEX is not able to solve the problem optimally, Z^* is the best obtained solution by the GA or the best randomly generated solution.

Table 3 reveals that when the number of subactivities is equal to 20, all 40 problems can be solved optimally by CPLEX 12.3 within 1000 second. It is noticeable that when the number of subactivities is equal to 20, the mathematical model contains averagely 57 variables and 139 constraints.

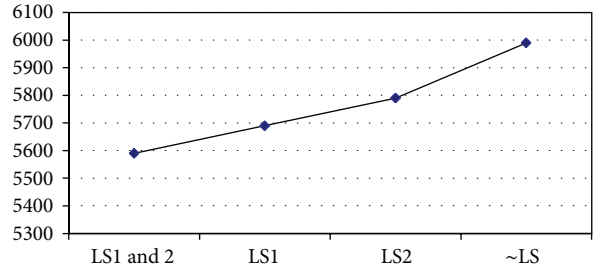

Figure 3: The effect of local searches on quality of GA.

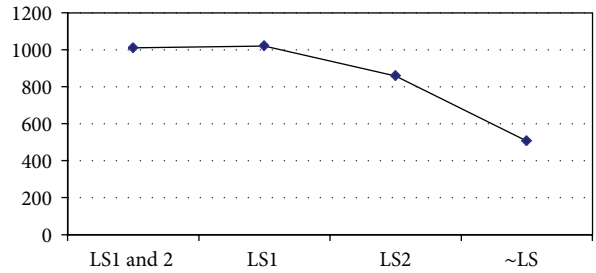

Figure 4: The effect of local searches on speed of GA.

Average CPU-time for CPLEX indicates that when the number of resource types is increased the complexity of problem is increased. Also, Table 3 shows that when the number of subactivities is greater than 20, while the CPLEX is unable to solve the problem, there is a solution by the proposed GA in a satisfying CPU time. ARD for the GA algorithm are not high. This means that proposed GA gives robust solutions. Also, NPMG for the GA indicates that too many of executions of problems with 20 subactivities reach to the optimum solution. Also, Table 3 shows that best randomly generated solutions (BRGS) conquest GA only in 1.75% runs (21 out of 1200 runs). Also, Table 3 indicates that solutions obtained by BRGS are not reliable measured by ARD. Consequently, these results confirm the advantages of the proposed GA to solve the P-MRIP-WET.

4.3. Sensitivity Analysis on Local Searches. In order to evaluate the effect of the knowledge based local searches on speed and quality of GA a sensitivity analysis is done. The results are demonstrated in Figures 3 and 4. Figure 3 shows that, by simultaneously using LS1 and LS2, the proposed GA has best quality measured by objective function. While, by only using LS1 and LS2, objective function will be 1.79% and 3.57% worse, respectively. However, applying GA without any local searches leads to solutions about 6.94% worse than simultaneously using LS1 and LS2.

It is clear from Figure 4, by simultaneously using LS1 and LS2, that the proposed GA has lowest speed measured by CPU time. By only using LS1 speed of algorithm reduces slightly, while by only using LS2 algorithm speeds up considerably (about 18%). Also, applying GA without any local searches leads to about 55% more speed than simultaneously using LS1 and LS2.

Table 3: Comparison of the results obtained by the CPLEX, GA, and BRGS.

Number of subactivities	Number of modes	Number of problems	CPLEX		BRGS		GA		
			NPOC	ACT-CPLEX	NBRGS	ARD-BRGS	NPMG	ARD-GA	ACT-GA
20	2	20	20	321.276	8	28.541%	193	1.38%	38.264
20	3	20	20	564.539	5	43.760%	184	1.51%	69.370
50	2	20	0	—	3	39.918%	—	2.27%	151.581
50	3	20	0	—	2	57.364%	—	2.64%	214.602
80	2	20	0	—	2	71.382%	—	2.80%	270.431
80	3	20	0	—	1	102.475%	—	3.79%	425.206

5. Summary and Conclusions

In this paper, we attempted to solve the preemptive multimode resource availability cost problem with due dates for the activities (P-MRIP-WET). The objective is to schedule the activities in order to minimize the total cost of the resources availabilities and earliness-tardiness penalties subject to the precedence constraints and a fixed deadline on project. This problem has not been studied ever before. The problem was described with an integer programming model, and then the genetic algorithm (GA) was proposed to solve it. However, two knowledge based local searches are proposed to improve the evolution speed of GA. The parameters of proposed GA are tuned based on Taguchi experimental design. The performance of the proposed algorithm on 120 test problems was compared with the results of the CPLEX and best randomly generated solutions (BRGS). From the computation results, we could clearly see that the proposed GA could efficiently solve the project scheduling problem.

Notations

A: Set of arcs of acyclic digraph representing the project

N: Set of nodes of acyclic digraph representing the project, $|N| = n$

n: Number of nondummy activities, index by i

K: Number of renewable resource(s), index by k

K': Number of nonrenewable resource(s), index by k'

M_i: Set of execution modes for activity $i, i \in N$

$|M_i|$: Number of execution modes for activity i, index by m_i

d_{im_i}: Duration of activity i in mode m_i, $i \in N$, $m_i \in M_i$

$r^{\rho}_{im_ik}$: Resource requirement of activity i in mode m_i for renewable resource type k, $k = 1, \ldots, K$, $i \in N$, $m_i \in M_i$

$r^{\nu}_{im_ik'}$: Resource requirement of activity i in mode m_i for nonrenewable resource type k', $k' = 1, \ldots, K'$, $i \in N$, $m_i \in M_i$

DD_i: Due date of activity i, $i \in N$

DL: Project deadline

e_i: Per unit earliness cost of activity i, $i \in N$

t_i: Per unit tardiness cost of activity i, $i \in N$

C^{ρ}_k: Unit cost of renewable resource type k, $k = 1, \ldots, K$

$C^{\nu}_{k'}$: Unit cost of nonrenewable resource type k', $k' = 1, \ldots, K'$

EST_i: Earliest start time of activity i (computed by assuming infinite resource capacities)

LFT_i: Latest finish time of activity i (computed by assuming infinite resource capacities)

Z: Objective function (total cost of the resources availabilities and earliness-tardiness penalties)

R^{ρ}_k: Constant availability of renewable resource type k throughout the project horizon, $k = 1, \ldots, K$ (integer decision variable)

$R^{\nu}_{k'}$: Total availability of nonrenewable resource type k', $k' = 1, \ldots, K'$ (integer decision variable)

E_i: Earliness of activity i (integer decision variable)

T_i: Tardiness of activity i (integer decision variable)

$S_{i\,\mathrm{Min}}$: Start time of first time unit of activity i (integer decision variable)

$S_{i\,\mathrm{Max}}$: Start time of last time unit of activity i (integer decision variable)

x_{im_it}: 1, if activity i in mode m_i is in progress at time interval $[t, t+1]$, 0, otherwise (binary decision variable)

y_{im_i}: 1, if activity i is executed in mode m_i, 0, otherwise (binary decision variable).

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] J. Blazewicz, J. K. Lenstra, and A. H. Rinnooy Kan, "Scheduling subject to resource constraints: classification and complexity," *Discrete Applied Mathematics*, vol. 5, no. 1, pp. 11–24, 1983.

[2] R. Kolisch and S. Hartmann, "Experimental investigation of heuristics for resource-constrained project scheduling: an update," *European Journal of Operational Research*, vol. 174, no. 1, pp. 23–37, 2006.

[3] H. Zhang, H. Li, and C. M. Tam, "Particle swarm optimization for resource-constrained project scheduling," *International Journal of Project Management*, vol. 24, no. 1, pp. 83–92, 2006.

[4] J. R. Montoya-Torres, E. Gutierrez-Franco, and C. Pirachicán-Mayorga, "Project scheduling with limited resources using a genetic algorithm," *International Journal of Project Management*, vol. 28, no. 6, pp. 619–628, 2010.

[5] S. Hartmann and D. Briskorn, "A survey of variants and extensions of the resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 207, no. 1, pp. 1–14, 2010.

[6] A. Agarwal, S. Colak, and S. Erenguc, "A neurogenetic approach for the resource-constrained project scheduling problem," *Computers & Operations Research*, vol. 38, no. 1, pp. 44–50, 2011.

[7] C. Fang and L. Wang, "An effective shuffled frog-leaping algorithm for resource-constrained project scheduling problem," *Computers & Operations Research*, vol. 39, no. 5, pp. 890–901, 2012.

[8] O. Koné, "New approaches for solving the resource-constrained project scheduling problem," *4OR*, vol. 10, no. 1, pp. 105–106, 2012.

[9] D. C. Paraskevopoulos, C. D. Tarantilis, and G. Ioannou, "Solving project scheduling problems with resource constraints via an event list-based evolutionary algorithm," *Expert Systems with Applications*, vol. 39, no. 4, pp. 3983–3994, 2012.

[10] G. Zhu, J. F. Bard, and G. Yu, "A branch-and-cut procedure for the multimode resource-constrained project-scheduling problem," *INFORMS Journal on Computing*, vol. 18, no. 3, pp. 377–390, 2006.

[11] H. Zhang, C. M. Tam, and H. Li, "Multimode project scheduling based on particle swarm optimization," *Computer-Aided Civil and Infrastructure Engineering*, vol. 21, no. 2, pp. 93–103, 2006.

[12] A. Lova, P. Tormos, and F. Barber, "Multi-mode resource constrained project scheduling: scheduling schemes, priority rules and mode selection rules," *Inteligencia Artificial*, vol. 10, no. 30, pp. 69–86, 2006.

[13] B. Jarboui, N. Damak, P. Siarry, and A. Rebai, "A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems," *Applied Mathematics and Computation*, vol. 195, no. 1, pp. 299–308, 2008.

[14] M. Ranjbar, B. de Reyck, and F. Kianfar, "A hybrid scatter search for the discrete time/resource trade-off problem in project scheduling," *European Journal of Operational Research*, vol. 193, no. 1, pp. 35–48, 2009.

[15] A. Lova, P. Tormos, M. Cervantes, and F. Barber, "An efficient hybrid genetic algorithm for scheduling projects with resource constraints and multiple execution modes," *International Journal of Production Economics*, vol. 117, no. 2, pp. 302–316, 2009.

[16] J. Coelho and M. Vanhoucke, "Multi-mode resource-constrained project scheduling using RCPSP and SAT solvers," *European Journal of Operational Research*, vol. 213, no. 1, pp. 73–82, 2011.

[17] M. Ranjbar, "An optimal NPV project scheduling with fixed work content and payment on milestones," *International Journal of Industrial Engineering & Production Research*, vol. 22, no. 3, pp. 181–186, 2011.

[18] A. Barrios, F. Ballestin, and V. Valls, "A double genetic algorithm for the MRCPSP/max," *Computers & Operations Research*, vol. 38, no. 1, pp. 33–43, 2011.

[19] B. Afshar-Nadjafi, A. Rahimi, and H. Karimi, "A genetic algorithm for mode identity and the resource constrained project scheduling problem," *Scientia Iranica*, vol. 20, no. 3, pp. 824–831, 2013.

[20] E. N. Afruzi, E. Roghanian, A. A. Najafi, and M. Mazinani, "A multi-mode resource-constrained discrete time-cost tradeoff problem solving using an adjusted fuzzy dominance genetic algorithm," *Scientia Iranica*, vol. 20, no. 3, pp. 931–944, 2013.

[21] R. H. Möhring, "Minimizing costs of resource requirements in project networks subject to a fixed completion time," *Operations Research*, vol. 32, no. 1, pp. 89–120, 1984.

[22] B. Rangaswamy, *Multiple resource planning and allocation in resource- constrained project networks [Ph.D. thesis]*, Graduate School of Business, University of Colorado, 1998.

[23] E. Demeulemeester, "Minimizing resource availability costs in time-limited project networks," *Management Science*, vol. 41, no. 10, pp. 1590–1598, 1995.

[24] A. Drexl and A. Kimms, "Optimization guided lower and upper bounds for the resource investment problem," *Journal of the Operational Research Society*, vol. 52, no. 3, pp. 340–351, 2001.

[25] S. B. Rodrigues and D. S. Yamashita, "An exact algorithm for minimizing resource availability costs in project scheduling," *European Journal of Operational Research*, vol. 206, no. 3, pp. 562–568, 2010.

[26] D. S. Yamashita, V. Amaral Armentano, and M. Laguna, "Scatter search for project scheduling with resource availability cost," *European Journal of Operational Research*, vol. 169, no. 2, pp. 623–637, 2006.

[27] S. Shadrokh and F. Kianfar, "A genetic algorithm for resource investment project scheduling problem, tardiness permitted with penalty," *European Journal of Operational Research*, vol. 181, no. 1, pp. 86–101, 2007.

[28] M. Ranjbar, F. Kianfar, and S. Shadrokh, "Solving the resource availability cost problem in project scheduling by path relinking and genetic algorithm," *Applied Mathematics and Computation*, vol. 196, no. 2, pp. 879–888, 2008.

[29] V. Van Peteghem and M. Vanhoucke, "An artificial immune system algorithm for the resource availability cost problem," *Flexible Services and Manufacturing Journal*, vol. 25, no. 1-2, pp. 122–144, 2013.

[30] L. Kaplan, *Resource constrained project scheduling with preemption of jobs [Ph.D. thesis]*, University of Michigan, Ann Arbor, Mich, USA, 1988.

[31] E. L. Demeulemeester and W. S. Herroelen, "An efficient optimal solution procedure for the preemptive resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 90, no. 2, pp. 334–348, 1996.

[32] F. Ballestín, V. Valls, and S. Quintanilla, "Pre-emption in resource-constrained project scheduling," *European Journal of Operational Research*, vol. 189, no. 3, pp. 1136–1152, 2008.

[33] M. Vanhoucke and D. Debels, "The impact of various activity assumptions on the lead time and resource utilization of resource-constrained projects," *Computers and Industrial Engineering*, vol. 54, no. 1, pp. 140–154, 2008.

[34] J. Damay, A. Quilliot, and E. Sanlaville, "Linear programming based algorithms for preemptive and non-preemptive RCPSP," *European Journal of Operational Research*, vol. 182, no. 3, pp. 1012–1022, 2007.

[35] J. Buddhakulsomsiri and D. S. Kim, "Properties of multi-mode resource-constrained project scheduling problems with resource vacations and activity splitting," *European Journal of Operational Research*, vol. 175, no. 1, pp. 279–295, 2006.

[36] V. Van Peteghem and M. Vanhoucke, "A genetic algorithm for the preemptive and non-preemptive multi-mode resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 201, no. 2, pp. 409–418, 2010.

[37] M. Vanhoucke, E. Demeulemeester, and W. Herroelen, "An exact procedure for the resouce-constrained weighted earliness-tardiness project scheduling problem," *Annals of Operations Research*, vol. 102, pp. 179–196, 2001.

[38] Y. Khoshjahan, A. A. Najafi, and B. Afshar-Nadjafi, "Resource constrained project scheduling problem with discounted earliness-tardiness penalties: mathematical modeling and solving procedure," *Computers and Industrial Engineering*, vol. 66, no. 2, pp. 293–300, 2013.

[39] N. Runge and F. Sourd, "A new model for the preemptive earliness-tardiness scheduling problem," *Computers & Operations Research*, vol. 36, no. 7, pp. 2242–2249, 2009.

[40] B. Afshar Nadjafi and S. Shadrokh, "A branch and bound algorithm for the weighted earliness-tardiness project scheduling problem with generalized precedence relations," *Scientia Iranica*, vol. 16, no. 1 E, pp. 55–64, 2009.

[41] K. Bülbül, P. Kaminsky, and C. Yano, "Preemption in single machine earliness/tardiness scheduling," *Journal of Scheduling*, vol. 10, no. 4-5, pp. 271–292, 2007.

[42] P. Baptiste, J. Carlier, A. Kononov, M. Queyranne, S. Sevastyanov, and M. Sviridenko, "Structural properties of optimal preemptive schedules," *Diskretnyi Analizi Issledovanie Operatsii*, vol. 16, no. 1, pp. 3–36, 2009.

[43] P. Baptiste, J. Carlier, A. Kononov, M. Queyranne, S. Sevastyanov, and M. Sviridenko, "Properties of optimal schedules in preemptive shop scheduling," *Discrete Applied Mathematics*, vol. 159, no. 5, pp. 272–280, 2011.

[44] A. Sprecher, S. Hartmann, and A. Drexl, "An exact algorithm for project scheduling with multiple modes," *OR Spektrum. Quantitative Approaches in Management*, vol. 19, no. 3, pp. 195–203, 1997.

[45] 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, no. 2, pp. 394–407, 2000.

[46] G. Taguchi, *Introduction to Quality Engineering*, Asian Productivity Organization, Tokyo, Japan, 1986.

[47] A. Drexl, R. Nissen, J. H. Patterson, and F. Salewski, "ProGen/πx—an instance generator for resource constrained project scheduling problems with partially renewable resources and further extensions," *European Journal of Operational Research*, vol. 125, pp. 59–72, 2000.