Multi-mode project selection, scheduling and resource allocation with time dependent returns

Seyed Mohsen Sajadi [a], Fatemeh Pourdehghan [b]

[a] *School of Industrial Engineering, Islamic Azad University, South Tehran Branch, Tehran, Iran*
[b] *School of Industrial and Systems Engineering, University of Missouri, Columbia, USA*

Abstract

In this paper, we address the problem of multi-mode project selection and scheduling with time-dependent returns. We first develop a mathematical model to formulate the problem, followed by the introduction of a branch-and-bound algorithm designed to solve it. To enhance the efficiency of the implicit enumeration algorithm, we incorporate several fathoming techniques and priority rules. Finally, the performance of the algorithm is evaluated in a numerical example.

Keywords: *Project Selection, Project Scheduling, Multi-Mode RCPSP, Implicit Enumeration Algorithm.*

1. Introduction and background

A project is defined as a complex effort based on some relevant activities which have been budgeted and scheduled by organizations to achieve special objectives. A project portfolio is a group of projects sponsored and/or managed by an organization. Projects should be completed with limited resources of labor, finances, time, etc. Hence, sometimes there are not enough resources to execute all profitable projects [1].

Industrial projects usually include innovation, technology advancement, process improvement, re-engineering and new product development. It is assumed that project returns are sensitive to completion time. Thus, delay in launching high-tech products can reduce the company's future profit significantly. On the other hand, entering the market early can bring the organization numerous advantages [2]. An organization has a large possible R&D projects pool to pursue. In addition, the organization should select a subset of projects to execute, taking into account resource limitations and time-dependent returns.

In literature, project selection models usually do not include project scheduling as a part of decision making. These models are aimed to select a project portfolio from the pool of projects, in order to maximize the expected returns considering budget constraints. Taylor et al. [3] considered resource allocation. Gabriel et al. [4] and Golmohammadi and Pajoutan [5] included the risk in selection process. Rabbani et al. [6] described a multi-objective project selection problem with maximizing the total benefit, minimizing the total cost and minimizing the total risk. Liesiö et al. [7] considered uncertainty in the project selection problem and formulated the problem as a robust model where complete information of costs and project interdependencies are not available.

Project scheduling is of high importance in projects management. Lack of sufficient resources as well as precedence relations between project's activities, cause the project scheduling to be quite a difficult task. Scheduling of a project to complete it as soon as possible by considering limited resources is known as resource constrained project scheduling problem (RCPSP). The problem of classic RCPSP assumes that each project includes several activities, and each activity has a fixed duration for completion and needs specified amount of resources in every period of time [2].

Among recent studies, Hartmann and Briskorn [8] developed different definitions for RCPSP problems. Moreover, Herroelen and Leus [9] provided a review on uncertainty in scheduling problems. Debels and Vanhoucke [10] allowed preemptive activities in discrete points of time. Drezet and Billaut [11] set resources as variables of each time period.

One of the most used objective functions in RCPSP problems is makespan minimization [8, 12, 13]. Neumann et al. [14] proposed their model with minimization of the weighted tardiness as the objective. Calhoun et al. [15] considered projects rescheduling. In their study, projects that need rescheduling due to reasons such as delays are also concerned. Achuthan and Hardjawidjaja [16] presented a problem with the objective function of minimum of total project costs and considered the cost of earliness and tardiness costs. Chen and Askin [2] and Vanhoucke et al. [17] considered maximizing the net present value (NPV) as objective function. Nudtasomboon and Randhawa [18] considered several objective including makespan, weighted tardiness, resource leveling and nonrenewable resource consumption together. Gomes et al. [19] considered precedence relations in a multi-objective RCPSP to minimize makespan and total weighted starting time of each activity, and then used five multi-objective metaheuristic algorithms to solve the problem.

The RCPSP problem normally assumes that each activity is executable in only one mode. In the real world, each activity may have several different modes to execute, and each mode can have its own resource requirement and time duration. More resource usage results in lower duration to execute the activity. In literature reviews, multi-mode RCPSP is often called MRCPSP [8].

Heilmann [24] and Sabzehparvar and Seyed-Hosseini [25] also included generalized precedence constraints into the problem. Erenguc et al. [26] introduced crashable mode where the time duration of a mode can be reduced by bearing additional costs, which is an intrinsic concept of the "mode". Węglarz et al. [22] examined scheduling problems with finite or infinite number of activity processing modes. Kyriakidis et al. [23] offered models for single-mode and multi-mode problems. Mika et al. [27] considered three types for setup times in MRCPSP: sequence-independent, sequence-dependent and schedule-dependent setup times. Messelis and De Causmaecker [28] investigated the structure of an automatic selection tool for the MRCPSP and proposed an algorithm that individually provided higher performance than all of its components. Peteghem and Vanhoucke [29] conducted a survey of the metaheuristic algorithms adopted to solve the MRCPSP and compared their newly proposed benchmark dataset and the existing ones.

Enumeration algorithms based on the depth-first search have become a standard method to solve the RCPSP problems [12, 13]. From other studies on branch-and-bound algorithms to solve similar problems, one can refer to Heilmann [24], in which a branch-and-bound algorithm is provided for the multi-mode RCPSP problem with minimum and maximum time lags. Bianco and Caramia [30] provided the branch and bound algorithm for a project scheduling problem considering scarce resources condition, and generalized precedence relations to minimize makespan. Also, Chen and Askin [2] presented an implicit enumeration algorithm in view of the resource allocation and time dependent returns. The algorithm is enhanced by fathoming rules.

To solve a scheduling problem, different algorithms have been used. Pourdehghan et al. [34] applied particle swarm optimization and genetic algorithm to minimize a linear combination of total completion time and maximum lateness of jobs. They proposed a mixed-integer mathematical model and introduced a clustering-based approach as a data mining tool to identify promising search areas. There are few studies considering both project selection and scheduling. Coffin and Taylor [31] first select a group of projects then schedule them using multiple criteria techniques. If selected projects exceed the completion time, some project may be replaced with projects with shorter duration. Therefore, the decision space is not fully explored to find an optimum solution. Kolisch and Meyer [32] integrated project selection and scheduling in pharmaceutical research projects. The resource usage is time-dependent and the objective function is net present value. Chen and Askin [2] developed a multi-project problem including two types of decision variables. First, a set of given projects and projects that should be executed is selected. In the second step, the projects are scheduled considering the generalized precedence relations and renewable resource constraints. Project selection and scheduling is done simultaneously to maximize the net present value of profits. Liu and Wang [33] optimized a problem of project selection with multiple projects and the time scheduling using the constraint programming.

As we reviewed, there are only few papers considering both project selection and scheduling. Although multi-mode concept is widely considered in project management, even these few papers consider single-mode projects. This paper extends the problem of project selection and scheduling to the case of multi-mode. It first formulates the problem by a mixed integer linear programming model. To solve the problem, a branch-and-bound algorithm is developed. This algorithm is equipped with some dominance rules.

The rest of the paper is organized as follows. Section 2 builds a mathematical model for the problem. Section 3 develops a branch-and-bound algorithm. Section 3 shows the procedure of the proposed algorithm by applying it to a numerical example. Section 4 evaluates the performance of the proposed algorithms. Section 5 finally concludes the paper.

2. Mathematical model

In this paper, a mixed integer linear mathematical model has been presented for multi-mode project selection, scheduling and resource allocation with time dependent returns problem. In this problem, there are N projects, each of which contains J_i activities and every activity can be executed in *M* modes. Each project should be started and completed in one mode, hence changing mode and preemption is not allowed. Meanwhile, there are *K* resources which their availability is limited in every period of time. Also, financial returns are time dependent and if delay or late execution of a project occurs, its profit will decline accordingly. Here the objective is selecting a proper project and scheduling it, considering the precedence relations and available resources constraints, to maximize the profit obtained through executing the projects.

This model includes two groups of decision variables: decision variables of project selection, and variables of project scheduling. It should be noted that the first and the last activity of a project is its dummy start and dummy end respectively. Therefore, their duration and resource requirements are equal to zero.

Indexes:

i: Set of available projects, *i*={1,2,…,*N*}

j: Set of activities, *j*={1,2,…,*J*}

m: Set of projects' modes, *m*={1,2,…,*M*}

k: Set of resources type, , *k*={1,2,…,*K*}

t: Set of time periods, *t*={1,2,…,*T*}

Parameters:

N: Number of candidate projects.

M: Number of activity's mode.

K: Number of resource type.

J_i: Final activity for project *i*.

T: time upper bound for projects' completion.

EF_{ij}: Early finish time for activity *j* of project *i*.

LF_{ij}: Last finish time for activity *j* of project *i*, given *T*.

P(*i*,*t*): Expected profit if project *i* completes in period *t*.

d_{ijm}: Duration of activity *j* of project *i* in mode *m*.

S(*ij*): Set of immediate successor activity *j* of project *i*.

r_{ijkm}: Resource *k* requirement for activity *j* of project *i* in mode *m*.

R_{kt}: Resource *k* available in period *t*.

Decision variables:

Y_{im} = 1 if project *i* selected in mode *m*, 0 Otherwise.

C_{ijmt} = 1 if activity *j* of project *i* completes in mode *m* at time *t*, 0 Otherwise.

The presented model is as follows.

$$max \quad \sum_{i=1}^{N}\sum_{m=1}^{M}\sum_{t=EF_{iJ_i}}^{LF_{iJ_i}} P(i,t)\,.\,C_{iJ_imt} \qquad (1)$$

$s.t.$

$$\sum_{m=1}^{M} Y_{im} \le 1 \qquad \forall i = 1,\dots,N; \quad (2)$$

$$\sum_{t=EF_{ij}}^{LF_{ij}} C_{ijmt} = Y_{im} \qquad \forall i = 1, \dots, N;\ \forall j = 1, \dots, J_i;\ \forall m = 1, \dots, M; \tag{3}$$

$$\sum_{t=EF_{ih}}^{LF_{ih}} (t - d_{ihm}).C_{ihmt} \geq 0 \qquad \forall i = 1, \dots, N;\ \forall m = 1, \dots, M;\ ih \in S(i1); \tag{4}$$

$$\sum_{t=EF_{ij}}^{LF_{ij}} t\,.C_{ijmt} \leq \sum_{t=EF_{ih}}^{LF_{ih}} (t - d_{ihm}).C_{ihmt} \qquad \forall i = 1, \dots, N;\ \forall j = 2, \dots, J_i - 1;\ \forall m = 1, \dots, M;\ ih \in S(ij); \tag{5}$$

$$\sum_{i=1}^{N} \sum_{j=2}^{J_i-1} \sum_{m=1}^{M} r_{ijkm} \sum_{q=\max\{t,EF_{ij}\}}^{min\{t+d_{ijm}-1,LF_{ij}\}} C_{ijmq} \leq R_{kt} \qquad \forall k = 1, \dots, K;\ \forall t = 1, \dots, T; \tag{6}$$

$$Y_{im} \in \{0,1\} \qquad \forall i = 1, \dots, N;\ \forall m = 1, \dots, M; \tag{7}$$

$$C_{ijmt} \in \{0,1\} \qquad \forall i = 1, \dots, N;\ \forall j = 1, \dots, J_i;\ \forall m = 1, \dots, M;\ \forall t = EF_{ij}, \dots, LF_{ij}; \tag{8}$$

Objective function (1) maximizes the total profits obtained by completing one of the modes of the selected projects. Constraint (2) assures that all activities of project are started and completed in one mode. Constraint (3) ensures that all activities of the selected project have been performed in one of their modes. Also, it prevents the activities of unselected projects from being executed. Constraints (4) and (5) control the precedence relations between activities. Constraint (6) limits the available resources of any type for each period of time. Constraints (7) and (8), defines the type of decision variables.

Due to the computational complexities, these problems, even with just one mode to execute, are NP-Hard problems [2]. Furthermore, the number of model's decision variables is *N.M+N.J.M.T*, all of which are binary variables, and the upper bound for the number of constraints is $N+N.J.M+N.J^2.M+K.T$.

3. Implicit enumeration algorithm

3.1. Basic enumeration algorithm

The algorithm to be considered in this section, considers all possible permutations in execution of projects in order to reach the highest possible profit. Enumeration algorithm explores a tree which includes all permutations of sequences of projects in different modes. Each node represents a project and branching to a new node means adding projects to time scheduling plan using the remaining available resources. Depth first search method indicates sequence of projects. In this method, the first branch found in exploration is selected as the next project. It explores as far as possible along each branch before backtracking, and continues the work to complete all projects and accordingly specifies the sequences. It should be noted that this method selects and schedules the projects simultaneously, and the optimum solution is not required to be scheduled again.

This algorithm has been improved using fathoming rules for pruning the tree's branches. Thus, implicit enumeration of projects is a heuristic method, unless all possible permutations of projects have been considered at the time scheduling. Chen and Askin [2] have established the basis of the proposed algorithms and fathoming rules used in this paper.

Symbols used to define a node in these algorithms are as follows:

n: Node number

P_n: Parent node number for node n

i_n: Project scheduled in node n

f_n: Profit earned by the project scheduled in node n

cf_n: Cumulative profit in node n

t_j: Activity j completion time

$[R_{kt}]_n$: Remaining resources of type k in period t for node n

F_n: Set of finished projects up to node n

E_n: Set of eligible projects for node n

V_n: Set of visited projects for node n

U_n: Set of unvisited projects for node n

Z: Set to prevent execute projects in multiple modes

Basic enumeration algorithm is as follows.

Step 1: Initialization

Let n=0 and f_0=cf_0=0. Let $[R_{kt}]_0$=available units of resource k in period t. Let the best node, defined as L, be the node 0. Projects are define as (i-m), i.e. project i is executed in mode m. Let F_0=V_0={empty} and E_0=U_0={11,12,…,1M,21,22,…,2M,…,N1,N2,…,NM}. Go to step 2.

Step 2: Branching

Check if U_n is empty. If yes, go to step 3. Otherwise, the first project in the first mode in U_n will be chosen. The project in this stage is named as im. Check if Project i in mode m is a member of Z. if yes, Let U_n=U_n-{im}, E_n=E_n-{im}, V_n=V_n+{im} and repeat step 2. Otherwise, schedule project i's activities in mode m to complete as soon as possible with resource available at node n, $[R_{kt}]_n$. Assume activity j for project i in mode m complete in period t_j (j = 1,…,J_i), the profit of project i in mode m is f and the remaining resources are $[R_{kt}]_x$. Let U_n=U_n-{im} and V_n=V_n+{im}. Branch into a new a, where a=current largest node number+1. Let P_a=n, i_a=im, f_a=f, cf_a=cf_n+f_a, t_{imj}=t_j, $[R_{kt}]_a$=$[R_{kt}]_x$. Let F_a=F_n+{im}, E_a=E_n-{im}, V_a={empty}, U_a=E_a and Z=Z+{i1,…,iM}. Check if cumulative profit of the new node is better than the best node (cf_a>cf_L). If yes, update the best node be the new node, i.e. L=a. Finally, let the new node be the current node, i.e. n=a. Repeat step 2.

Step 3: Backtracking

Check if current node is node 0. If yes, stop and specify the best sequence by traveling from the node 0 to best node L. Otherwise, for project i that scheduled in node n, let Z=Z-{i1,…,iM}, current node to be its parent node (n=p_n) and go to step 2.

3.2. Projects priority rules

In this section, some rules have been suggested to prioritize the projects before running the algorithm. These priority rules reduce the computation time and the number of explored nodes. Before running the algorithm one of these rules is selected, then the projects and modes are ordered accordingly. This initial sequence influences on the algorithm, and the algorithm explores the projects and nodes in the same order.

Priority rule 1 (maximum profit): Initially, the profit from the execution of every project for each mode in minimum makespan is determined. Then, the maximum amount of profit resulting from the execution of each project is calculated and prioritized based on the maximum profit in descending order. After prioritization of the projects, modes of each project are also arranged in descending order of profit.

Priority rule 2 (minimum project duration): At first, required time duration to execute each project for each mode is designated. Then, minimum duration of each project is obtained and accordingly is ordered ascending. Modes of each project are sorted based on required duration in ascending order after prioritization of projects.

Priority rule 3 (maximum profit/resources): In this rule, the rate of maximum profit obtained from the execution of the project with respect to the required resources of each project is determined in all of modes, and then is arranged in the same way as the priority rule 1.

3.3. Fathoming rules

Fathoming rules have been developed for pruning tree's branches. As a sequence of projects traversed in the tree, these rules determine the next node (project). Backtracking from a project is calculated based on the remaining available resources according to the solution of RCPSP problems. Using these rules can reduce unnecessary computations and exploration of non-optimal nodes significantly and consequently increases the speed of the algorithm.

By increasing the completion time, the project profit will whether decrease or remain at existing value and under no circumstances it will increase. If profit of execution of project i in mode m in node n is non-positive, then fathoming rules can be used. Suppose the node a is the parent of the node n. In all nodes branched from node a, the remaining resources cannot be more than node a. However, none of the nodes branching out of the node n, can reach the optimal solution due to consumption of resources by non-profitable project i. These resources are wasted, without increasing the profit. Continuation of branching of this node by other sequences of the node a would be dominated, hence the algorithm should backtrack to the node a. Fathoming rule 1 is resulted from the above definitions. And with improvement in the first, fathoming rule 2 will be obtained.

Fathoming rule 1 (FR1): Eliminate project i in mode m from E_a, if profit f of project i in mode m at node n is such that $f \leq 0$, where a is the parent node of n. Then backtrack to node a.

Fathoming rule 2 (FR2): If $f \leq 0$ at node n, record $\{i, [R_{kt}]_n\}$. Each time branch into a new node a, if found a record that $\{h, [R_{kt}]_n\}$ where for all k and t that $[R_{kt}]_n \geq [R_{kt}]_a$, eliminate any project h from E_a.

In this phase the existing node n is being compared to a previous node which was q. If $cf_n \leq cf_q$, E_n is a subset of E_{iq} (initial E_q, i.e. set E_q when node q is first generated), and the $[R_{kt}]_n \leq [R_{kt}]_q$ for all k and t, then the maximum profit resulted from continuing the node n, equals the best amount of profit from continuing the node q. Therefore, fathoming the node n and pruning the branch, the algorithm is prevented exploring more nodes and thus backtracks to the parent node n (i.e. P_n). Fathoming rule 3 is obtained from the above definitions. Fathoming rule 4 is obtained by improving the fathoming rule 3.

Fathoming rule 3 (FR3): Fathom any branch from current node n and backtrack to parent node (P_n) if there exists a previous node q such that: (1) $cf_n \leq cf_q$, (2) $[R_{kt}]_n \leq [R_{kt}]_q$ for all k and t, (3) E_n is a subset of E^i_q.

Fathoming rule 4 (FR4): Fathom any branch from current node n and backtrack to parent node (P_n) if there exists a previous node q such that: (1) $cf_n \leq cf_q$, (2) $[R_{kt}]_n \leq [R_{kt}]_q$ for all k and t, (3) F_n is same as F_q.

3.4. Proposed implicit enumeration algorithm

In this algorithm, fathoming rules FR1 and FR2 checked before the branching into a new node and fathoming rules FR3 and FR4 checked after the branching. Implicit enumeration algorithm's step with fathoming and priority rules proposed as follows:

Step 1: Initialization

First, projects and modes sort by one of priority rules. Let $n = 0$ and $f_0 = cf_0 = 0$. Let $[R_{kt}]_0$ = available units of resource k in period t. Let the best node, defined as L, be node 0. Projects define as im, i.e. project i execute in mode m. Let $F_0 = V_0 = \{\text{empty}\}$ and $E_0 = U_0 = \{11,12,\ldots,1M,21,22,\ldots,2M,\ldots,N1,N2,\ldots,NM\}$. Go to step 2.

Step 2: Branching

Check if U_n is empty. If yes, go to step 3. Otherwise, pick first project in first mode in U_n, name project im. Check if Project i in mode m is a member of Z. if yes, Let $U_n = U_n - \{im\}$, $E_n = E_n - \{im\}$, $V_n = V_n + \{im\}$ and repeat step 2. Otherwise, schedule project i's activities in mode m to complete as soon as possible with resource available at node n, $[R_{kt}]_n$. Assume activity j for project i in mode m complete in period t_j ($j = 1,\ldots,J_i$), the profit of project i in mode m is f and the remaining resources are $[R_{kt}]_x$. Update $U_n = U_n - \{im\}$ and $V_n = V_n + \{im\}$. Check if $f \leq 0$. If yes, invoke FR1 and FR2. Let $E_n = E_n - \{im\}$, Record $\{i,[R_{kt}]_n\}$ and repeat step 2. Otherwise, branch into a new a, where a = current largest node number + 1. Let $P_a = n$, $i_a = im$, $f_a = f$, $cf_a = cf_n + f_a$, $t_{imj} = t_j$, $[R_{kt}]_a = [R_{kt}]_x$. Let $F_a = F_n + \{im\}$, $E^i_a = E_n - \{im\}$, $E_a = E^i_a$ - {any fathomable project via FR2}, $V_a = \{\text{empty}\}$, $U_a = E_a$ and $Z = Z + \{i1,\ldots,iM\}$. Check if cumulative profit of the new node is better than the best node ($cf_a > cf_L$). If yes, update the best node be the new node, i.e. $L = a$. Check if new node a is fathomed via FR3 or FR4. If yes, Let $U_n = U_a = \{\text{empty}\}$. Finally, let the new node be the current node, i.e. $n = a$. Repeat step 2.

Step 3: Backtracking

Check if current node is node 0. If yes, stop and specify the best sequence by traveling from the node 0 to best node L. Otherwise, for project i that scheduled in node n, let $Z=Z-\{i1,\ldots,iM\}$, current node to be its parent node ($n=p_n$) and go to step 2.

Figure 1 shows the implicit enumeration algorithm's flowchart.

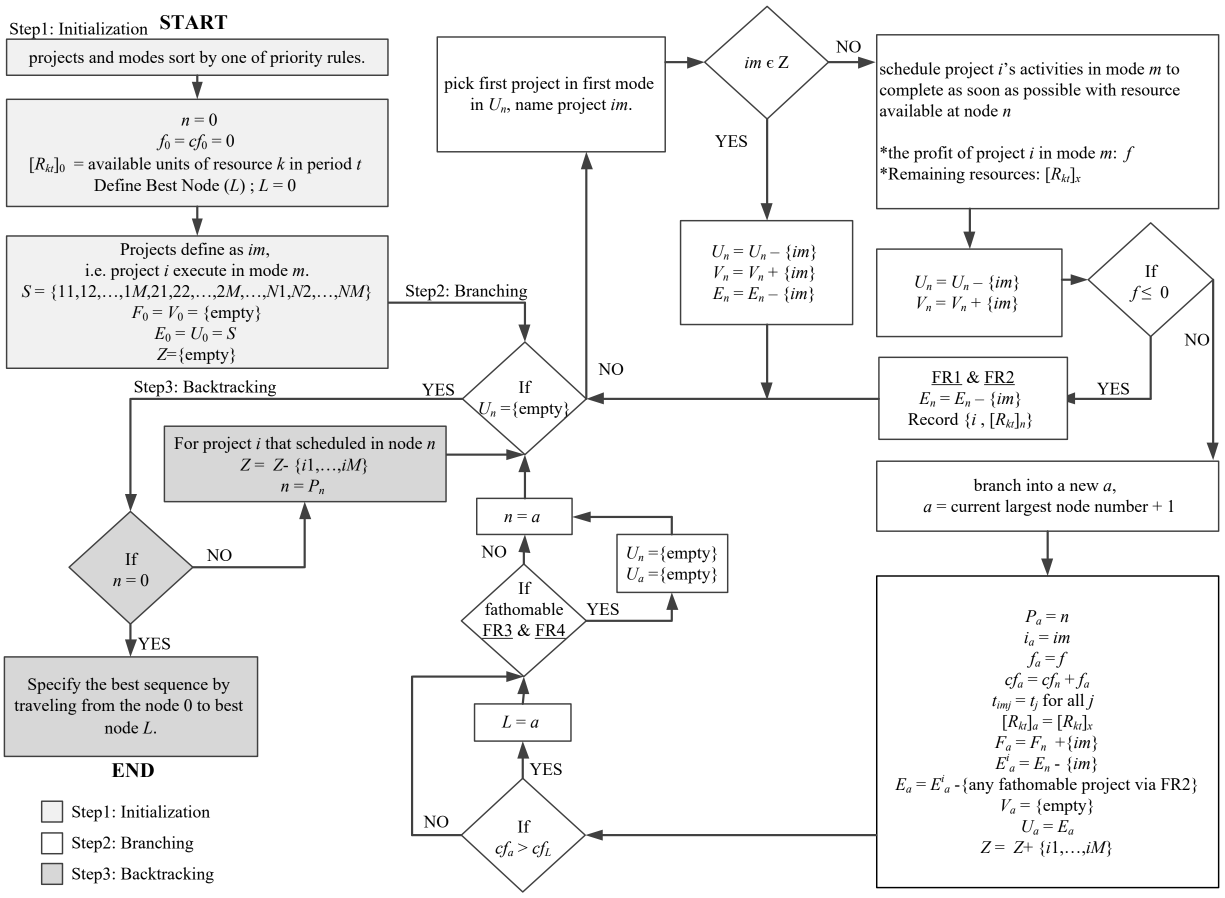
Step1: Initialization
START
projects and modes sort by one of priority rules.
$n = 0$
$f_0 = cf_0 = 0$
$[R_{kt}]_0$ = available units of resource k in period t
Define Best Node (L) ; $L = 0$
Projects define as im,
i.e. project i execute in mode m.
$S = \{11,12,\ldots,1M,21,22,\ldots,2M,\ldots,N1,N2,\ldots,NM\}$
$F_0 = V_0 = \{\text{empty}\}$
$E_0 = U_0 = S$
$Z=\{\text{empty}\}$
Step2: Branching
pick first project in first mode in U_n, name project im.
$im \in Z$
NO
YES
schedule project i's activities in mode m to complete as soon as possible with resource available at node n
*the profit of project i in mode m: f
*Remaining resources: $[R_{kt}]_x$
$U_n = U_n - \{im\}$
$V_n = V_n + \{im\}$
$E_n = E_n - \{im\}$
$U_n = U_n - \{im\}$
$V_n = V_n + \{im\}$
If $f \leq 0$
NO
YES
FR1 & FR2
$E_n = E_n - \{im\}$
Record $\{i , [R_{kt}]_n\}$
If $U_n = \{\text{empty}\}$
NO
YES
Step3: Backtracking
For project i that scheduled in node n
$Z = Z- \{i1,\ldots,iM\}$
$n = P_n$
If $n = 0$
NO
YES
Specify the best sequence by traveling from the node 0 to best node L.
END
$n = a$
NO
$U_n = \{\text{empty}\}$
$U_a = \{\text{empty}\}$
If fathomable FR3 & FR4
YES
$L = a$
YES
NO
If $cf_a > cf_L$
branch into a new a,
a = current largest node number + 1
$P_a = n$
$i_a = im$
$f_a = f$
$cf_a = cf_n + f_a$
$t_{imj} = t_j$ for all j
$[R_{kt}]_a = [R_{kt}]_x$
$F_a = F_n + \{im\}$
$E^i_a = E_n - \{im\}$
$E_a = E^i_a$ -{any fathomable project via FR2}
$V_a = \{\text{empty}\}$
$U_a = E_a$
$Z = Z+ \{i1,\ldots,iM\}$
☐ Step1: Initialization
☐ Step2: Branching
▨ Step3: Backtracking

Figure 1. Implicit enumeration algorithm's flowchart

4. Numerical example

In this section, a numerical example has been presented to show how to execute the projects in alternative modes, time dependent returns and also proposed implicit enumeration algorithm. In this example assume that there are four projects and each project can be executed in two modes. First step in proposed algorithm is giving priority to projects. With this priority rule the number of explored nodes can be reduced. In this example, the priority rule of maximum profit has been used. Duration time and the amount of required resources of projects in each mode of execution are sorted as priority and have been shown in table 1 as well as the amount of profit resulted from the completion of project in different periods. It should be noted that the amount of available resources has been considered to be four units in each period of time and the entire project to be made up of three activities, such that the first activity is project's dummy start and the third activity can be project's dummy end. Also the planning horizon length has been assumed to be seven time periods.

Table 1
Projects' expected profit and modes' duration and resource requirement

Project	First Mode		Second Mode		Expected Profit						
	Duration	Resource Requirement	Duration	Resource Requirement	1	2	3	4	5	6	7
1	1	4	3	3	10	7	4	0	-1	-1	-2
2	1	4	2	1	9	7	6	2	1	0	-1
3	2	3	4	2	9	8	4	1	0	-1	-2
4	2	2	3	1	8	5	2	0	0	-1	-1

To describe the execution in alternative modes, the project 1 is considered lonely. As shown in table 1, the project no.1 can be executed in the first mode, and in this case, it takes one time period to complete and make 10 units of profit. It also requires four units of resource per unit of time. Whereas if the project no.1 executes in the second mode, then it will be completed in the third time period and fewer profit equals four units will be brought. Of course, it should be noted that less resources, three units of resource per time unit, is also required. At this point, projects should be considered together and simultaneously. Therefore, the projects should be executed in modes which could make maximum profit according to the limited available resources.

First, in node 1, the project 1 is chosen in its first mode. This project makes 10 units of profit. The subsequent step is to move to the next node. At this node, the second project is selected in its second mode. This choice turns 6 units of profit. Likewise, project 3 in its first mode has been selected and makes 1 unit of profit. Then, given that the profit shown by the project number 4 returns a negative value, it will not be chosen and the algorithm returns to the parent node (node 2). Since in this node, selecting project 3 in mode 2 and project 4 in its all modes is not possible based on given fathoming rules, the algorithm returns to the parent node (node 1) and then among all available projects, selects a new project (project 3 in mode 1). This procedure will then continue until the ceasing condition is reached.

Example is solved by using proposed implicit enumeration algorithm's steps. Finally, it forms a tree with nodes that determine the sequence of the projects. Implicit enumeration algorithm tree has been shown in Figure 2 completely. In this figure, projects are shown as (*i*-*m*), i.e. Project *i* is executed in mode *m*. The process of implicit enumeration algorithm to demonstrate the algorithm performance has been also shown thoroughly in table 2. In this process, all steps and fathomed have been listed. Projects should be executed in modes which could have maximum profit regarding the limited resources. Maximum profit is (+ 20) unit, which has been gained at the node 9. Thus at first, the project no.1 is executed in mode 1, then the second project will be run in mode 2 and then the third project will be executed in the mode 1. It is apparently the 4th project is not executed.

To obtain the optimum profit in this example, 59 nodes in the tree are explored. Whereas if all of the nodes had checked regardless of the fathoming rules, 632 nodes would have been required to be explored. It should be noted that total number of nodes for basic enumeration without fathoming rules, for *i* projects and *m* modes is calculated as $\sum_j m^j \binom{i}{j} j!$. The number of nodes of algorithm is independent of the number of activities of project, however the number of activities of project is effective in the solve time of algorithm. The number of nodes with regard to fathoming rules depends on the data of problem (eg, initial profit and the rate of maximum decrease in the profit in every time period).

In this example, the number of explored nodes in other priority rules is as follow: 67 nodes to prioritize based on maximum profit/resource priority rule, 60 nodes to prioritize according to the minimum duration priority rule and 82 nodes to execute the algorithm regardless of any of the priority rules.

5. Computational results

In this section, for different parameters, random problems have been created and the results of solving these problems have been presented. The data of this kind of problems were randomly generated from a uniform distribution. First, number of project (N), number of modes (M) and maximum number of project's activity (J) are specified. Then, each parameter is generated as shown in table 3. It can be noted that all obtained values are rounded up.

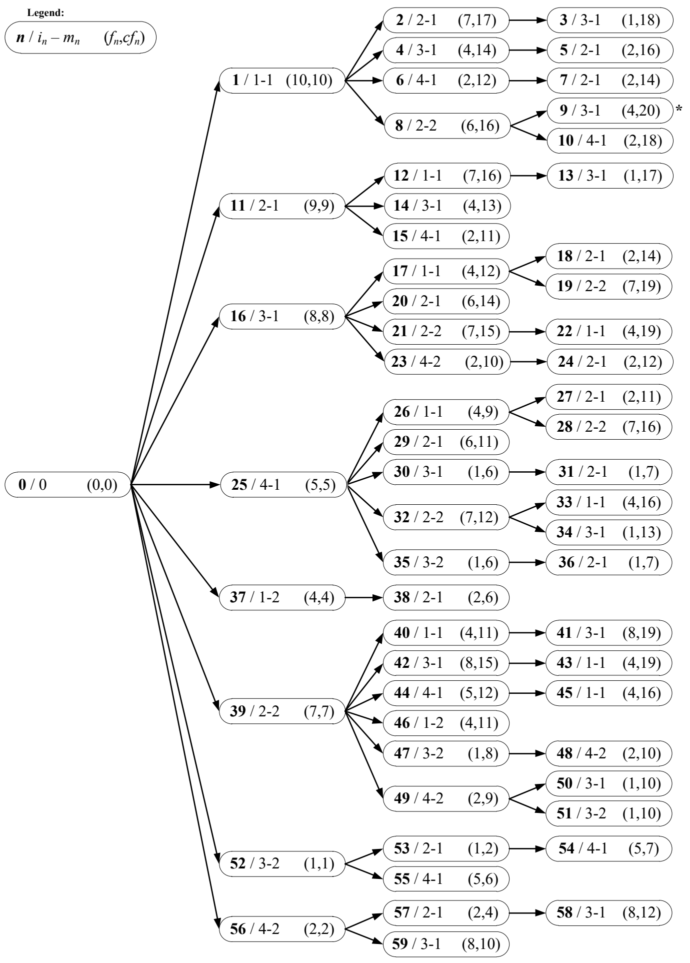

Figure 2. Implicit enumeration tree

Table 2
Implicit enumeration algorithm process

Node	Scheduled Project	Scheduled Mode	Cumulative Profit	Remark
1	1	1	10	New best node. FR1 (projects 3-2, 4-2).
2	2	1	17	New best node. FR1 (projects 3-2, 4-1, 4-2).
3	3	1	18	New best node. FR1 (projects 4-1, 4-2).
4	3	1	14	-
5	2	1	16	FR3 via node 2 (project 2-1).
6	4	1	12	-
7	2	1	14	FR3 via node 2 (project 2-1).
8	2	2	16	FR1 (projects 3-2, 4-2).
9	3	1	20	*Best node*. FR1 (projects 4-1, 4-2).
10	4	1	18	FR1 (projects 3-1, 3-2).
11	2	1	9	FR1 (projects 1-2, 3-2, 4-2).
12	1	1	16	-
13	3	1	17	FR2 via node 2 (projects 4-1, 4-2). FR3 or FR4 via node 3 (project 3-1).
14	3	1	13	FR1 (projects 1-1, 1-2, 4-1, 4-2).
15	4	1	11	FR1 (projects 1-1, 1-2, 3-1, 3-2).
16	3	1	8	FR1 (projects 1-2, 4-1).
17	1	1	12	FR1 (projects 4-1, 4-2).
18	2	1	14	FR1 (projects 4-1, 4-2).
19	2	2	19	FR1 (project 4-2).
20	2	1	14	FR1 (projects 1-1, 1-2, 4-1, 4-2).
21	2	2	15	FR1 (project 4-2).
22	1	1	19	FR1 (project 4-2).
23	4	2	10	FR1 (project 1-1).
24	2	1	12	FR2 via node 23 (projects 1-1, 1-2). FR3 via node 2 (project 2-1).
25	4	1	5	FR1 (project 1-2).
26	1	1	9	FR1 (projects 3-1, 3-2).
27	2	1	11	FR1 (projects 3-1, 3-2).
28	2	2	16	FR1 (project 3-2).
29	2	1	11	FR1 (projects 1-1, 1-2, 3-1, 3-2).
30	3	1	6	FR1 (project 1-1).
31	2	1	7	FR2 via node 30 (projects 1-1, 1-2). FR3 via node 27 (project 2-1).
32	2	2	12	FR1 (project 3-2).
33	1	1	16	FR1 (projects 3-1, 3-2).
34	3	1	13	FR1 (project 1-1).
35	3	2	6	FR1 (project 1-1).
36	2	1	7	FR2 via node 35 (projects 1-1, 1-2). FR3 via node 2 (project 2-1).
37	1	2	4	-
38	2	1	6	FR2 via node 16 (projects 4-1, 4-2). FR3 via node 27 (project 2-1).
39	2	2	7	-
40	1	1	11	-
41	3	1	19	FR4 via node 3 (project 3-1).
42	3	1	15	-
43	1	1	19	FR4 via node 22 (project 1-1).
44	4	1	12	-
45	1	1	16	FR4 via node 33 (project 1-1).
46	1	2	11	FR1 (projects 3-1, 3-2, 4-1, 4-2).
47	3	2	8	FR1 (projects 1-1, 1-2, 4-1). FR2 via node 16 (projects 4-1, 4-2).
48	4	2	10	FR3 via node 16 (project 4-2).
49	4	2	9	FR1 (projects 1-1, 1-2).
50	3	1	10	FR1 (project 1-2).
51	3	2	10	-
52	3	2	1	FR1 (project 1-1).
53	2	1	2	FR2 via node 52 (projects 1-1, 1-2).
54	4	1	7	FR2 via node 35 (projects 1-1, 1-2). FR3 via node 6 (project 4-1).
55	4	1	6	FR3 via node 6 (project 4-1).
56	4	2	2	FR1 (project 1-1).
57	2	1	4	FR2 via node 56 (projects 1-1, 1-2).
58	3	1	12	FR2 via node 23 (projects 1-1, 1-2). FR3 via node 3 (project 3-1).
59	3	1	10	FR3 via node 4 (project 3-1).

In this section, in order to investigate this algorithm, 1000 random problems have been generated in different ranges. These ranges are divided base on explored nodes by basic enumeration algorithm. Generated problems for the number of projects, the number of activities and modes of each project and also fixed profit decrease rate for each problem have been solved by all priority rules. Proposed algorithms has been solved with MATLAB 8.1 and has been implemented on a computer with Core i7 2.20GHz CPU and 4GB physical memory (RAM).

Table 3
Random problem generation data

Parameter	*Distribution*	*Description*
J_i	Uniform[$0.7\times J$, J]	Depends on (J)
T	Uniform[$3\times N\times J$, $7\times N\times J$]	Depends on (J) and (N)
d_{ijm}	Uniform[1 , 7]	
r_{ijkm}	Uniform[1 , 10]	
R_{kt}	Uniform[10 , 15]	
$P(i,t)$	Uniform[$0.5\times T$, T]	t=1 (Profit in first time period)
$P(i,t)$	Uniform[$(1-\alpha)\times P(i,t-1)-1\times\alpha$, $P(i,t-1)$]	t>1 - Depends on previous period's profit and the rate of maximum decrease in the profit in every time period (α)

5.1. Algorithms computational time

It should be noted that the total number of nodes for basic enumeration algorithm without fathoming rules, for project i with m modes is always constant and is calculated as $\sum_j m^j \binom{i}{j} j!$. The number of nodes with respect to fathoming rules depends on problem's data e.g. the initial profit, time periods and profit decrease rate. By increasing of the explored nodes, computational time will be also increased. The number of nodes of algorithm is independent of the number of projects' activities, with which the computational cost has a linear relationship.

Increasing the number of projects and modes increases the algorithms' explored nodes gap which results in increasing of the computational time gap. In larger size problems, implicit enumeration algorithm finds the optimum solution with about 0.01% of explored nodes and time of the basic enumeration algorithm.

Table 4
Comparison algorithms by average computational time (by seconds)

i	m	Enum. Run Time	Proposed Algorithm Run Time	Time Gap (%)	i	m	Enum. Run Time	Proposed Algorithm Run Time	Time Gap (%)	i	m	Enum. Run Time	Proposed Algorithm Run Time	Time Gap (%)
3	2	0.02	0.02	0.0	4	2	0.09	0.03	63.4	5	2	3.65	0.20	94.5
3	4	0.09	0.03	60.9	4	4	2.62	0.22	91.6	5	4	1763	1.57	99.9
3	6	0.34	0.10	71.1	4	6	84.07	1.60	98.1	5	6	2943	2.51	99.9
3	8	1.02	0.19	81.4	4	8	779.08	5.02	99.4	5	8	-	5.07	-
3	10	2.51	0.23	91.0	4	10	4169.50	6.11	99.9	5	10	-	10.78	-

Table 5
Average proposed algorithm computational time (by seconds) per different parameters value

i	*j*	*m*	Run Time	*i*	*j*	*m*	Run Time	*i*	*j*	*m*	Run Time
2	5	5	0.03	5	3	5	1.58	5	5	2	0.20
3	5	5	0.14	5	4	5	1.82	5	5	4	1.57
4	5	5	0.44	5	6	5	2.00	5	5	6	2.51
5	5	5	2.18	5	8	5	5.78	5	5	8	5.07
6	5	5	15.65	5	10	5	9.12	5	5	10	10.78
7	5	5	146.47	5	12	5	15.21	5	5	12	35.25
8	5	5	922.88	5	14	5	26.13	5	5	14	60.60
9	5	5	2572.70	5	16	5	38.31	5	5	16	132.00
10	5	5	4677.99	5	18	5	50.81	5	5	18	217.59
11	5	5	7631.1	5	20	5	73.24	5	5	20	363.63

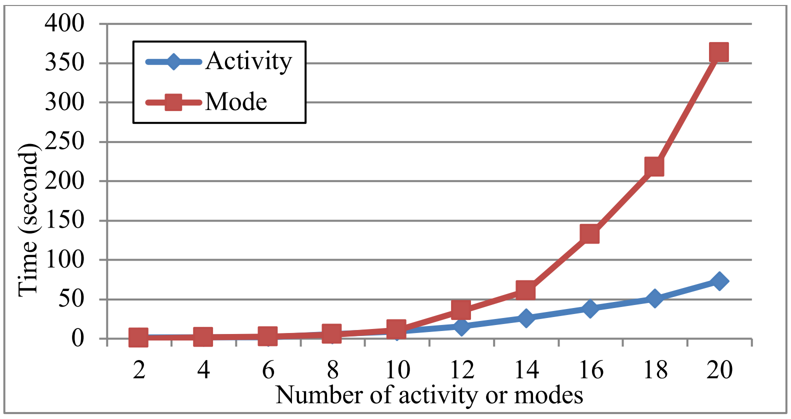

Figure 3. Average model computational time per different activity and mode value

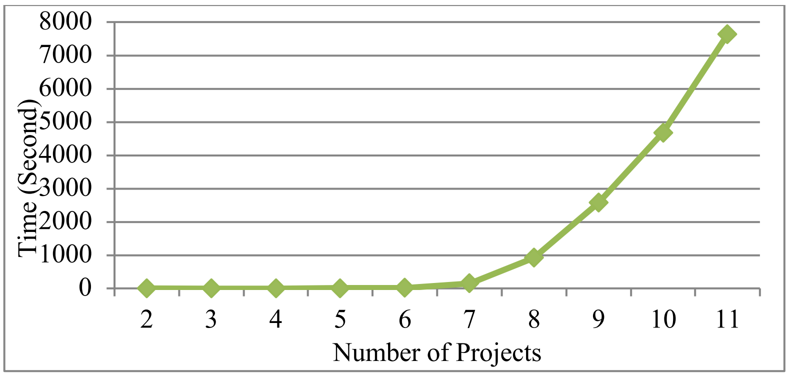

Figure 4. Average model computational time per different number of projects

In this section α is set to 0.2. Also, maximum profit priority rule is used. Other data are randomly generated for different parameters. Table 4 presents the computational time of the enumeration and implicit enumeration algorithm, as well as the time gap between them for different values of *m* and *i*. As can be seen in the table, increase in the number of projects and modes causes the time gap and the explored nodes in either of the algorithms increase.

The average computational time in second for different values of parameters is given in Table 5. In each part of this table, two parameters are considered fixed and one other parameter has been increased. As shown in Figure 3 and 4, the increment of computational time of problem rises exponentially for enhancement of the number of projects and project's modes. However, by increasing the number of projects, computational time may increase with greater rate than by increasing of the number of projects' modes. The increase of the computational time caused by increasing the number of project's activities follows a linear trend which is rather insignificant when compared to those that caused by increasing the number of projects or modes. Also, the number of time periods, the number of different types of resources and the rate of maximum decrease in the profit in every time period influence the solution time. However, these influences are less than those of main parameters.

5.2. Priority rules comparison

Problems with basic enumeration algorithm ("Enum."), an implicit enumeration algorithm have been solved through maximum profit priority rule, maximum profit/resource priority rule, minimum duration priority rule and without priority rules in random orders. The results in table 6 and figure 5 have been presented based on the average of explored nodes and percentage of improvement of explored nodes for priority rules.

As can be seen, by increasing the explored nodes in basic enumeration algorithm, the gap between explored nodes grows larger and applying priority rules can improve time and the number of explored nodes of algorithm. On average, in comparison with proposed algorithm without priority rules and in random order, the maximum profit priority rule by 45.25%, the maximum profit/resource priority rule by 38.43%, and the minimum duration priority rule by 41.22% make an improvement and decrease in the number of explored nodes. Since the max profit priority rule creates much better improvement, so this rule shall be used further in this study.

Table 6
Comparison priority rules by average explored node

Priority	Row	1	2	3	4	5	Average Gap
	Range	10^2-10^3	10^3-10^4	10^4-10^5	10^5-10^6	10^6-10^7	
Enumeration	Enum.	562	5171	56358	178137	2602109	
No Priority	Node	119.37	536.65	1801.75	3656.33	9330.06	
	Gap with Enum.	78.76%	89.62%	96.80%	97.95%	99.64%	92.55%
Max Profit	Node	95.12	327.45	910.37	1538.16	3776.37	
	Gap with Rand.	20.31%	38.98%	49.47%	57.93%	59.52%	45.25%
	Gap with Enum.	83.07%	93.67%	98.38%	99.14%	99.85%	94.82%
Max Profit/Resource	Node	98.62	352.85	984.62	1920.75	4878.43	
	Gap with Rand.	17.38%	34.25%	45.35%	47.47%	47.71%	38.43%
	Gap with Enum.	82.45%	93.18%	98.25%	98.92%	99.81%	94.52%
Min Duration	Node	100.25	335.4	931	1771.75	4410.75	
	Gap with Rand.	16.02%	37.50%	48.33%	51.54%	52.73%	41.22%
	Gap with Enum	82.16%	93.51%	98.35%	99.01%	99.83%	94.57%

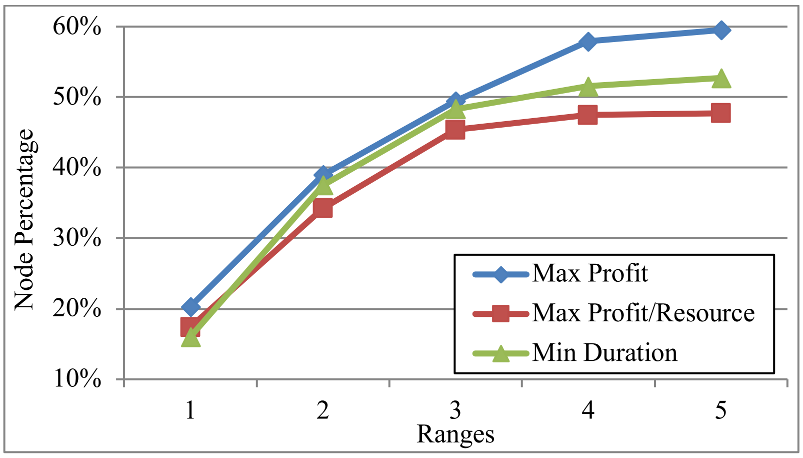

Figure 5. Node percentage gap explored by priority rules

5.3. Maximum profit decrease rate and fathoming rules

In this section, explored nodes by the two algorithms, frequency of using fathoming rules and improvement percentage are discussed. Generated problems are clustered in different ranges and results for different ranges are shown in table 7. As can be seen from the table, the number of proposed nodes of algorithm decreases by increasing the rate of maximum profit decrease (α) and the optimum solution is acquired more quickly. By increasing the number of projects and modes, the number of explored nodes will increase. The greater number of projects and modes, the larger the difference between the two algorithms given that the implicit enumeration algorithm will attain the optimum solution through the exploration of much smaller number of nodes.

The average number of applying the fathoming rules for different ranges and for different rates of maximum profit decrease (α) is shown in Figure 6. As can be seen, the application of these rules increases by increasing the number of projects, modes, and as a result nodes. According to the figure 6a, the fathoming rule 1 initially increases and then decreases by increasing α. This is because by declining the amount of projects' profit and rising the number of non-positive projects, fewer fathoming rules will be used for fathoming larger number nodes.

The fathoming rule 2 is considered as a supplement to the fathoming rule 1. According to the figure 6b, the use of the rule increases for different rates. However, it must be added that according to the figure, first, higher rates of α have greater values in FR2 and while the range grows this order changes reversely and lower α's rates witness faster increase and as result gain greater values than higher α. This is because the fathoming rule 1 closes many nodes and therefore, the possibility of using f other rules declines to its minimum. However, the fathoming rules 1 and 2 are depended on each other and by increasing the use of the fathoming rule 1, the fathoming rule 2 also increases.

Fathoming rule 3 and 4 are almost similar. However, the fathoming rule 3 is used more, because it comes earlier than the rule 4 in the process of checking the algorithm. If the fathoming rule 3 is not used in the algorithm, the fathoming rule 4 should be checked. However, as can be seen in figures 6c and 6d, the usage of both rules reduces by increasing

α. This decrease is due to the use of the fathoming rule 1 and the fact that it fathoms a large number of nodes.

Table 7
Average fathoming rules used for different ranges and decrease profit rates

α	Range	FR1	FR2	FR3	FR4	Node	Enum. Node	Gap
0.1	10^2-10^3	14.62	0	33.7	3.08	117.62	400.6	61.75%
	10^3-10^4	37.17	0.22	149.05	22.05	514.85	5171.6	88.76%
	10^4-10^5	180.5	4.5	611.28	108.35	1797.07	59160	96.93%
	10^5-10^6	170.58	1.5	816.37	109.08	2415.54	178137.3	98.52%
	10^6-10^7	553.2	4.5	3249.65	484.75	8662.75	3182118.6	99.66%
	10^7-10^8	664.93	6.06	6335	1073.31	16864.69	26473343	99.93%
	10^8-10^9	2364.12	23.06	15796.38	2584.75	41282.06	504798834	99.99%
Average		569.3	5.69	3855.92	626.48	10236.37	76385309	92.22%
0.2	10^2-10^3	36.12	0.25	19.58	2.26	73.37	400.6	76.84%
	10^3-10^4	152.57	1.4	93.85	12.37	321.85	5171.6	92.29%
	10^4-10^5	233.14	5.14	342.92	55	1010	59160	98.04%
	10^5-10^6	554.37	10	469	71.62	1415.83	178137.3	99.13%
	10^6-10^7	1433.45	53.45	1269.5	155.45	3370.35	3182118.6	99.87%
	10^7-10^8	3094.31	86.43	2815.5	375.06	7376.12	26473343	99.97%
	10^8-10^9	5919.93	317.37	6433.56	929.37	16454.13	504798834	99.99%
Average		1631.98	67.72	1634.84	228.73	4288.80	76385309	95.16%
0.3	10^2-10^3	38.91	0.54	14.37	2	57.2	400.6	78.02%
	10^3-10^4	173.85	5.35	62.8	8.05	212.62	5171.6	95.11%
	10^4-10^5	355.5	31.21	227.85	24.42	659.07	59160	98.77%
	10^5-10^6	613.2	19	304.54	38.16	901.08	178137.3	99.47%
	10^6-10^7	1465.8	90.8	693.75	75.1	1818	3182118.6	99.93%
	10^7-10^8	2462.31	150.87	1501	231.18	4086.31	26473343	99.98%
	10^8-10^9	4868	303.81	3872.12	492	9680.68	504798834	99.99%
Average		1425.37	85.94	953.77	124.41	2487.85	76385309	95.90%
0.4	10^2-10^3	35.62	0.66	10.2	1.08	39.91	400.6	87.75%
	10^3-10^4	134.4	4.1	41.1	5.42	141.85	5171.6	96.75%
	10^4-10^5	239.851	14.71	183.71	20.57	515.57	59160	99.12%
	10^5-10^6	602.95	34.79	144	13.5	417.95	178137.3	99.73%
	10^6-10^7	1056.55	85.25	417.2	64.3	1150.5	3182118.6	99.96%
	10^7-10^8	2305.81	276.93	953.31	140.87	2513.25	26473343	99.99%
	10^8-10^9	4552.93	738	1565.31	160.12	3964.06	504798834	99.99%
Average		1275.44	164.92	473.54	57.98	1249.01	76385309	97.61%
0.5	10^2-10^3	27.5	0.58	4.58	0.41	20.83	400.6	91.24%
	10^3-10^4	105.67	4.45	24.15	3.22	82.82	5171.6	97.74%
	10^4-10^5	193.28	19.28	61.5	11.64	193.5	59160	99.58%
	10^5-10^6	464.12	35.5	91.16	10.62	263.87	178137.3	99.84%
	10^6-10^7	732.15	74.4	180.3	24.85	504.65	3182118.6	99.98%
	10^7-10^8	1942.12	147.12	446.81	52.5	1185.93	26473343	99.99%
	10^8-10^9	3635.37	572.31	1020	109.25	2612.81	504798834	99.99%
Average		1014.31	121.95	261.21	30.35	694.91	76385309	98.34%

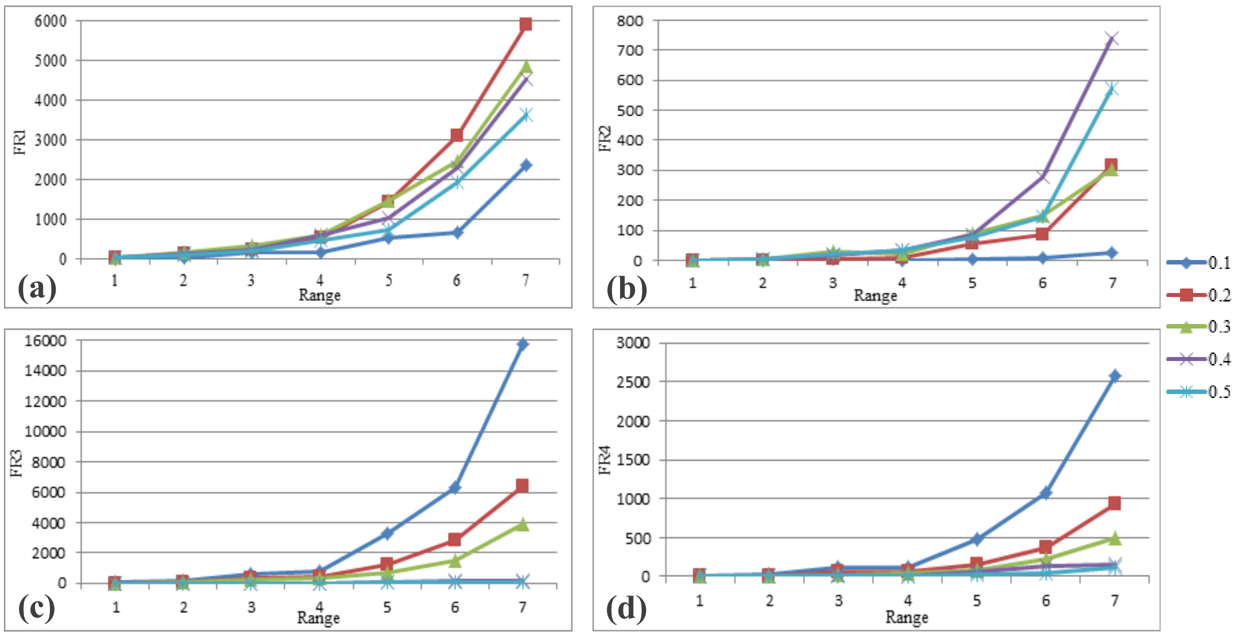

Figure 6. Average Fathoming rules used for different α

6. Conclusion

This study has focused on the problem of selecting and scheduling of multi-mode projects with resource allocation and time dependent returns. The assumption of multi-mode execution of projects along limited available resources and profit returns of projects that are depended to completion time makes the problem closer to a real world problem. In addition, the project selection and scheduling of the selected projects are done simultaneously. Selecting inappropriate projects will result in losing the profit obtainable from choosing suitable projects, in addition to the costs spent on limited resources for improper projects.

After proposing a mathematical model for this problem, an implicit enumeration algorithm has been presented which has been improved by several fathoming and priority rules. These rules prevent additional calculations resulted by full exploration. The results indicate that the prioritization of projects in the first step of selection is of great importance and increases the speed of the algorithm. The rate of maximum decrease in the profit in every time period has also a large impact on the fathoming rules and the number of explored nodes by the algorithm. Simultaneous use of fathoming and priority rules can reduce the number of nodes for exploration and consequently computational time up to 99.99%. Of the future research in this field can be referred to the expanse of the fathoming rules, providing new fathoming and priority rules, and taking into account the uncertainty and utilization of heuristics methods to solve these problems.

References

[1] Archer, N.P., Ghasemzadeh, F., 1999. An integrated framework for project portfolio selection, International Journal of Project Management 17, 207–216.

[2] Chen, J., Askin, R.G., 2009. Project selection, scheduling and resource allocation with time dependent returns, European Journal of Operational Research 193 (1), 23–34.

[3] Taylor, B.W., Moore, J.L., Clayton, E.R., 1982. R&D project selection and manpower allocation with integer nonlinear goal programming, Management Science 28 (10), 1149–1158.

[4] Gabriel, S.A., Ordonez, J.F., Faria, J.A., 2006. Contingency planning in project selection using multiobjective optimization and chance constraints, Journal of Infrastructure Systems 12 (2), 112–120.

[5] Golmohammadi, A., Pajoutan, M., 2011. Meta heuristics for dependent portfolio selection problem considering risk, Expert Systems with Applications 38, 5642–5649.

[6] Rabbani, M., Aramoon Bajestani, M., Baharian Khoshkhou, G., 2010. A multi-objective particle swarm optimization for project selection problem, Expert Systems with Applications 37, 315–321.

[7] Liesiö, J., Mild, P., Salo, A., 2008. Robust portfolio modeling with incomplete cost information and project interdependencies, European Journal of Operational Research 190, 679–695.

[8] Hartmann, S., Briskorn, D., 2010. A survey of variants and extensions of the resource-constrained project scheduling problem, European Journal of Operational Research 207, 1–14.

[9] Herroelen, W.S., Leus, R., 2005. Project scheduling under uncertainty: Survey and research potentials, European Journal of Operational Research 165 (2), 289–306.

[10] Debels, D., Vanhoucke, M., 2008. The impact of various activity assumptions on the lead time and resource utilization of resource-constrained projects, Computers and Industrial Engineering 54, 140–154.

[11] Drezet, L.E., Billaut, J.C., 2008. A project scheduling problem with labour constraints and time-dependent activities requirements, European Journal of Operational Research 112 (1), 217–225.

[12] Demeulemeester, E., Herroelen, W., 1992. A branch-and-bound procedure for the multiple resource-constrained project scheduling problem, Management Science 38 (12), 1803–1818.

[13] Demeulemeester, E., Herroelen, W., 1997. A branch-and-bound procedure for the generalized resource-constrained project scheduling problem, Operations Research 45(2), 201–212.

[14] Neumann, K., Schwindt, C., Zimmermann, J., 2002. Recent results on resource-constrained project scheduling with time windows: Models, solution methods, and applications, Central European Journal of Operations Research 10, 113–148.

[15] Calhoun, K.M., Deckro, R.F., Moore, J.T., Chrissis, J.W., Hove, J.C.V., 2002. Planning and re-planning in project and production scheduling, Omega – The international Journal of Management Science 30 (3), 155–170.

[16] Achuthan, N., Hardjawidjaja, A., 2001. Project scheduling under time dependent costs – A branch and bound algorithm, Annals of Operations Research 108 (1–4), 55–74.

[17] Vanhoucke, M., Demeulemeester, E.L., Herroelen, W.S., 2001. On maximizing the net present value of a project under renewable resource constraints, Management Science 47, 1113–1121.

[18] Nudtasomboon, N., Randhawa, S.U., 1997. Resource-constrained project scheduling with renewable and non-renewable resources and time-resource tradeoffs, Computers and Industrial Engineering 32 (1), 227–242.

[19] Gomes, H.C., de Assis das Neves, F., Souza, M.J.F., 2014. Multi-objective metaheuristic algorithms for the resource-constrained project scheduling problem with precedence relations, Computers & Operations Research 44, 92–104.

[20] Coelho, J., Vanhoucke, M., 2011. Multi-mode resource-constrained project scheduling using RCPSP and SAT solvers, European Journal of Operational Research 213, 73–82.

[21] Deblaere, F., Demeulemeester, E., Herroelen, W., 2011. Reactive scheduling in the multi-mode RCPSP, Computers & Operations Research 38, 63–74.

[22] Węglarz, J., Józefowska, J., Mika, M., Waligóra, G., 2011. Project scheduling with finite or infinite number of activity processing modes – A survey, European Journal of Operational Research 208, 177–205.

[23] Kyriakidis, T.S., Kopanos, G.M., Georgiadis, M.C., 2012. MILP formulations for single- and multi-mode resource-constrained projectscheduling problems, Computers and Chemical Engineering 36, 369–385

[24] Heilmann, R., 2003. A branch-and-bound procedure for the multi-mode resource-constrained project scheduling problem with minimum and maximum time lags, European Journal of Operational Research 144 (2), 348–365.

[25] Sabzehparvar, M., Seyed-Hosseini, S.M., 2008. A mathematical model for the multimode resource-constrained project scheduling problem with mode dependent time lags, Journal of Supercomputing 44 (3), 257–273.

[26] Erenguc, S., Ahn, T., Conway, D., 2001. The resource constrained project scheduling problem with multiple crashable modes: An exact solution method, Naval Research Logistics 48 (2), 107–127.

[27] Mika, M., Waligóra, G., Weglarz, J., 2008, Tabu search for multi-mode resource constrained project scheduling with schedule-dependent setup times, European Journal of Operational Research 187 (3), 1238–1250.

[28] Messelis, T., De Causmaecker, P., 2014. An automatic algorithm selection approach for the multi-mode resource-constrained project scheduling problem. European Journal of Operational Research 233, 511–528.

[29] Van Peteghem, V., Vanhoucke, M., 2014. An experimental investigation of metaheuristics for the multi-mode resource-constrained project scheduling problem on new dataset instances, European Journal of Operational Research 235, 62–72.

[30] Bianco, L., Caramia, M., 2012. An exact algorithm to minimize the makespan in project scheduling with scarce resources and generalized precedence relations, European Journal of Operational Research 219, 73–85.

[31] Coffin, M.A, Taylor, W.B., 1996. R&D project selection and scheduling with a filtered beam search approach, IIE Transactions 28, 167–176.

[32] Kolisch, R, Meyer, K., 2006. Selection and scheduling of pharmaceutical research projects." In: Jozefowsk, J., Weglarz, J. (Eds.), Perspectives in Modern Project Scheduling. Springer Science, New York, 321–344.

[33] Liu S.S., Wang, C.W., 2011. Optimizing project selection and scheduling problems with time-dependent resource constraints, Automation in Construction 20, 1110–1119.

[34] Pourdehghan, F., Fazlollahtabar, H. 2019. Meta-heuristic algorithms for a clustering-based fuzzy bi-criteria hybrid flow shop scheduling problem. *Soft Computing, 23*(22), 12103-12122.