MULTI-MODE RESOURCE CONSTRAINED PROJECT SCHEDULING PROBLEM INCLUDING MULTI-SKILL LABOR (MRCPSP-MS)-MODEL AND A SOLUTION METHOD

Mónica A. Santos[1], Anabela P. Tereso[2]

Abstract: The problem that we address in this chapter is an extension of the Resource-Constrained Project Scheduling Problem (RCPSP). It belongs to the class of project scheduling problems with multi-level (or multi-mode) activities, that permit an activity to be processed by resources operating at appropriate modes, where each mode belongs to a different resource level and incurs different cost and duration. Each activity must be allocated exactly one unit of each required resource, and the resource unit may be used at any of its specified levels. The processing time of an activity is given by the maximum of the durations that would result from different resources allocated to that activity. The objective is to find an optimal solution that minimizes the overall project cost, given a delivery date. A penalty is incurred for tardiness beyond the specified delivery date, or a bonus is accrued for early completion. We present a mathematical programming formulation as an accurate problem definition. A Filtered Beam Search (FBS)-based method is used to solve the problem. It was implemented using the C# language. Results of our experimentations on the use of this method are also presented.

1. Introduction

The Resource-Constrained Project Scheduling problem that we address in this chapter involves multi-level activities, where each activity can be processed using one of several modes that are available for each resource, with each mode of a resource belonging to a different skill level and incurring different cost and duration. This class of problems is an extension of the Resource-Constrained Project Scheduling Problem (RCPSP), which has been shown to be NP-hard (Blazewicz et al., 1983). Some of the earlier methods proposed for the solution of the project scheduling problem include: CPM (Critical Path Method) (Kelley and Morgan, 1959), and PERT (Program Evaluation and Review Technique) (MacCrimmon and Ryavec, 1964; Clark, 1962), resource allocation method (Davis, 1966), resource leveling procedures (Bandelloni et al., 1994, Zimmermann and Engelhardt, 1998), Monte-Carlo simulation-based methods (Metropolis et al., 1953; Ragsdale, 1989), and those based on criticality indices (Dodin and Elmaghraby, 1985), among others. Ever since, there have been mathematical models proposed for more complex problems. The search for optimal solutions for the RCPSP has focused on the use of Integer Programming (IP) (Pritsker et al. 1969, Berthold et al. 2010, Nemhauser and Wolsey 1988), Dynamic Programming (DP) (Bellman and Dreyfus, 1959, Elmaghraby et al., 1992), and Branch and Bound (B&B) techniques.

The presence of binary variables in a problem has led researchers to develop B&B-based procedures for its solution. The success of this technique depends on the branching scheme and on the tightness of the bound used. Kis et al. (2005) explored the scheduling problem, where the need for resources for each activity varies in proportion to the intensity of the activity itself. To formalize the problem, they used an integer linear programming model and proposed a B&B-based algorithm to find an optimal solution. However, B&B procedures are inadequate for real-size problems, despite their efficiency relative to a frontal attack on the discrete optimization problem. The need to solve real problems in reasonable CPU times, have led researchers to develop heuristic procedures.

The heuristics used belong to one of two classes: priority rule-based methods or meta-heuristic approaches. The priority rule-based methods build a plan by selecting activities from a range of activities available successively so that all activities are sequenced (Boctor, 1993; Dean et al. 1992; Heilmann, 2000). The meta-heuristic-based methods begin with an initial solution, and then, search for its

[1] University of Minho, 4710-057 Braga, Portugal Email: monica.lasalete@gmail.com

[2] University of Minho, 4710-057 Braga, Portugal Email: anabelat@dps.uminho.pt

improvement by defining an appropriate neighborhood. There are still two types of heuristics, series heuristics where the priority of the activities is predetermined and remains fixed, and parallel heuristics where the priority is updated each time an activity is scheduled for processing. Other types of heuristics found in the literature, considered as a sub-field of meta-heuristics, are tabu search (Arroub et al., 2010), simulated annealing (Mika et al., 2005) and genetic algorithms (Gonçalves et al. 2004). Tseng (2008) has also discussed the use of genetic algorithms applied to the Multi-Project, Multi-Mode RCPSP.

For the multi-mode RCPSP, it has been shown that, for highly resource-constrained projects with more than 20 activities and three modes in each activity, it is difficult to find optimal schedules (Hartmann, 2001). The heuristic methods are simple to understand, easy to apply, and are capable of clarifying the scheduling process. Typically, heuristic methods consider each activity's impact on a specific objective by sorting the activities competing for resources and allocating only some of them the resources needed for scheduling in a period. Besides, meta-heuristic make few or no assumptions about the problem and have the advantage of performance consistency and the ability to determine global optimal solutions. However, the search for optimal solution within feasible solutions makes meta-heuristic methods spend more computational time than heuristic methods (Zhang et al., 2006).

The objective of our study is to minimize the total project cost given a due date that includes a bonus for early completion or a penalty for tardiness. In several resource-constrained scheduling problem models found in the literature, there are two important aspects present in any model: the objective and the constraints. The objective may be based on time, such as minimizing the project duration (Boctor 1990, Heilmann, 2000, Basnet et al., 2001), or on economic aspects, such as minimizing the project cost (Tereso et al. 2004, Tereso et al. 2006, Mika et al., 2005). However, time-based objectives are often in conflict with cost-based objectives. A recurrent situation encountered in practice is the need to complete a project by its due date and maximize profit. Ozdmar and Ulusoy (1995) reported in their survey of the literature, studies where the Net Present Value (NPV) is maximized while the due date is a 'hard' constraint (Patterson et al. 1989, 1990). There are several other multi-objective studies in the literature where efficient solutions regarding time and cost targets are generated. Guldemond et al. (2008) presented a study related to the problem of scheduling projects with strict deadline jobs, defined as a Time-Constrained Project Scheduling Problem (TCPSP), where a non-regular objective function is used. The original RCPSP uses regular objective functions, like minimizing the makespan, but several non-regular objectives have become popular like maximizing NPV. Vanhoucke et al. 2000 define regular and non-regular measures of performance: "A regular measure of performance is a nondecreasing function of the activity completion times, while for a nonregular measure of performance the above definition does not hold".

Kazaz and Sepil (1996) have presented a Mixed Integer Linear Program (MIP) formulation with Benders decomposition for a project scheduling problem where the cash flows do not occur at some event realization times, but as progress payments at the end of some time periods. In Sepil and Ortaç (1997), three different heuristic rules were developed to solve the same problem, extended with renewable resource constraints. Padman and Dayanand (1997) allow the decision-maker to set progress-payment points and Etgar et al. (1997) incorporate elements of bonus/penalty structures. As the costs incurred depend on the activities in progress while scheduling is based on non-cost related considerations, the researchers explicitly included cash-flows-resources-constraints in their formulations. Elmaghraby and Herroelen (1990) employed the following property of an optimal solution that maximizes the NPV: the activities with positive cash flows should be scheduled as soon as possible and those with negative cash flow as late as possible. They argue that the faster completion of the project is not necessarily the optimal solution with regard to maximizing the NPV. In Mika et al. (2005) study, a positive flow is associated with each activity. The objective is to maximize the NPV of all cash flows of the project. They used two meta-heuristics that are widely used in research: Simulated Annealing (SA) and Tabu Search (TS). Our problem objective is cost-based and we have a bonus/penalty structure, but we do not consider the NPV objective.

In Ulusoy and Cebelli's (2000) approach to payment scheduling problem (using a multi-mode RCPSP), the amount and timing of the payments made by the client and received by the contractor are determined so as to achieve an equitable solution. An equitable solution is defined as one where both the contractor and the client deviate from their respective ideal solutions by an equal percentage. The ideal solutions for the contractor and the client result from having a lump sum payment at the start and end of the project, respectively. A double-loop genetic algorithm (GA) is proposed to solve an equitable solution. The outer loop represents the client and the inner loop the contractor. The inner loop corresponds to a multi-mode RCPSP with the objective of maximizing the contractor's NPV for a given payment distribution. When searching for an equitable solution, information flows between the outer and inner loops regarding the payment distribution over the event nodes and the timing of these payments.

Willis (1985) described requirements for modeling realistic resources. These requirements include the variable need of resources according to the duration of the activity, variable availability of resources over the period of the project, and different operational modes for the activities. A discrete time/resource function implies the representation of an activity in different modes of operation. Each mode of operation has its own duration and amount of renewable and non-renewable resource requirements.

The number of activities in a project determines the size of the project, and also, it contributes to the complexity of a scheduling problem. Another fact contributing to the complexity of a project is the precedence relations among the activities. As shown by Elmaghraby and Herroelen (1980), projects with the same number of activities and the same number of activity relationships can have varying degrees of difficulty. So, there is not a straightforward association between project complexity and problem difficulty. Meanwhile, many project complexity measures have been proposed (Herroelen, 2006).

Boctor (1993) presented a heuristic procedure for the scheduling of non-preemptive resource-constrained projects, where the resource is renewable from period to period. Each activity is assumed to have a set of possible durations and resource requirements. The objective is to minimize the project duration. The heuristic used belongs to the class of priority rule-based methods. This class builds a plan by selecting activities from a range of activities available successively so that all activities are sequenced. A general framework to solve large-scale problems was suggested. The heuristic rules that can be used in this framework were evaluated, and a strategy to solve these problems efficiently was designed. Heilmann (2000) also worked with the multi-mode case in order to minimize the duration of the project. In his work, besides the different modes of execution of each activity, a maximum and minimum delay between activities is specified. He presented a priority rule-based heuristic. Basnet et al. (2001) presented a "filtered beam" search technique to generate makespan minimizing schedules, for multi-mode single resource constrained projects, where there is a single renewable resource to consider and the multi-mode consists essentially of how many people can be employed to finish an activity.

The problem presented here also belongs to the class of project scheduling problems with multi-level (or multi-mode) activities, with each activity processed by resources operating at appropriate modes, where each mode belongs to a different resource skill level, which implies different cost and duration. Usually, multi-mode RCPSP defines an explicit set of modes for each activity, with a specific activity duration and resource requirements. Our approach, however, defines a set of resource levels. Each activity may elect a level for each one of the resources required. The combination of all possible levels of each resource, required for the execution of the activity, provide the alternative modes of execution for each activity.

Santos and Tereso (2010) have presented a formal MRCPSP-MS model and a breadth-first search procedure. Consequently, Santos and Tereso (2011a, 2011b) presented an adaptation of a FBS scheme to this problem and reported result of a preliminary computational investigation. In this chapter, we present further analysis on the results obtained for networks of different sizes.

In a Breadth First Search (BFS) scheme, all the nodes (partial solutions) in the search tree are evaluated at each stage before going any deeper, subsequently realizing an exhaustive search that visits all nodes of the search tree. The branch and bound search technique can be seen as a polished breadth first search, since it applies some criteria in order to reduce the BFS complexity. Usually, it consists of keeping track of the best solution found so far, discarding a node if it cannot offer a better solution. A FBS is a heuristic B&B procedure that uses BFS but only the top best nodes are kept. At each stage of the tree, all successors for the selected nodes at the current stage are generated, but it only stores a preset number of descendent nodes at each stage, called the beam width.

The B&B and the Beam Search (BS) procedures have been typically applied to the RCPSP (Basnet et al., 2001; Demeulemeester and Herroelen, 1996; Kis, 2005). The differentiating aspects of our approach are: (i) the definition of a set of states followed by the activities; (ii) the priority rules used to solve resource conflicts; and (iii) the alternative evaluation rules used to discard undesirable "branches".

Our approach allows determination of a project solution using the BFS scheme or the FBS scheme. We implemented the proposed approach using C# language.

2. Problem description

Consider a project network in the activity-on-arc (AoA) representation: $G = (N, A)$, with $|N| = n$ (representing the events) and $|A| = m$ (representing the activities). Each activity may require the simultaneous use of several resources with their consumption dictated by the selected execution mode - each resource may be deployed at a different level. The objective is to determine the optimal allocation of resources to the activities in order to minimize the total cost incurred (due to resources + penalty for tardiness + bonus for earliness). We follow the dictum that an activity should be initiated as soon as it is sequence-feasible.

There are $|R| = \rho$ resources. A resource has a capacity of several units (say w workers or machines) and may be used at different levels, such as a 'resource' of electricians of different skill levels, or a 'resource' of milling machines but of different capacities and ages. A level might be the power of usage of a machine: high, medium or low, or the amount of hours used by a resource: half-time, normal time or extra-time. Another example would be a 'resource' of routes where the levels could be: easy, short, fast, or economic.

The different levels of a resource $r \in R$ may be represented as $L(r) = \{r_1, \ldots r_{L(r)}\}$; the number of levels varies with the resource. If resource r is utilized at level l for activity j then the processing time shall be denoted by $p(j, r, l)$. An activity normally requires the simultaneous utilization of more than one resource for its execution, but each activity must be allocated exactly one unit of each resource.

To better visualize the problem, one can summarize its characteristics in a matrix format as shown in Table 1. For illustrative purposes, we assume that any resource may have at most three levels: low (level 1), average or normal (level 2), and high (level 3). A cell entry in the matrix is the processing time $p(j, r, l)$ for activity j of resource r at level l. Due to space limitations, Table 1 exhibits the information as (j, r, l); the symbol "p" is forfeited. If an activity does not require a resource; this is indicated in the matrix by the symbol ∅ (null). The symbol b_r gives the number of units available of resource r.

Table 1: Problem characteristics with three levels.

Resources / Activities	Processing time (j, r, l)		
	$1\ (b_1)$	……	$\lvert R\rvert = \rho\ (b_\rho)$
1	$(1,1,1)\ (1,1,2)\ (1,1,3)$	…	$(1,\rho,1)\ (1,\rho,2)\ (1,\rho,3)$
2	$(2,1,1)\ (2,1,2)\ (2,1,3)$	…	$(2,\rho,1)\ (2,\rho,2)\ (2,\rho,3)$
.	.	.	.
.	.	.	.
.	.	.	.
$\lvert A\rvert = m$	$(m,1,1)\ (m,1,2)\ (m,1,3)$	…	$(m,\rho,1)\ (m,\rho,2)\ (m,\rho,3)$

b_r= the number of units available of resource r.
(j, r, l) = the processing time $p(j, r, l)$ for activity j of resource r at level l.

The processing time of an activity is given by the maximum of the durations that would result from a specific allocation of various resources. To better understand this representation, consider the miniscule project of Figure 1 with three-activities. Assume that the project requires the utilization of four resources.

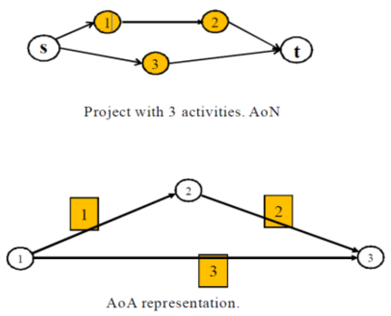


Figure 1. A project with three activities.

Let $\gamma(r,l)$ be the unitary cost of resource r at level l. Then, in each cell, there shall be a $2 \times |L(j,r)|$ matrix in which the first row gives the duration $p(j,r,l)$ and the second row the cost $c(j,r,l)$. The total cost of a resource, at each level, is obtained as the product of the resource unitary cost with the activity processing time.

Table 2: Project resource requirements, processing times and resource costs.

↓Act	Resources (availability)	→	1 (2)	2 (1)	3 (3)	4 (2)	η_j
	Res. Unit cost for each level		(1,3)	(2,5,10)	(1,3,5)	(2,4,6)	
1	$p(j,r,l)$		(14,6)	Ø	(12,8,5)	(18,12,7)	3
	$c(j,r,l)$		(14,18)		(12,24,25)	(36,48,42)	
2	$p(j,r,l)$		Ø	(7,5,3)	Ø	(8,5,4)	2
	$c(j,r,l)$			(14,25,30)		(16,20,24)	
3	$p(j,r,l)$		(20,12)	(22,16,10)	Ø	Ø	2
	$c(j,r,l)$		(20, 36)	(44,80,100)			

$p(j,r,l)$ = the processing time $p(j,r,l)$ for activity j of resource r at level l.
$c(j,r,l)$ = the total cost for activity j of resource r at level l.
η_j= count of resources required for activity j.

For brevity, Table 2 gives only the processing times and costs at each level of the resources; the identity of the activity and the resource are suppressed.

The top row indicates that there are 4 resources with varying availability: resources 1 and 4 have $b1 = b4 = 2$ units each, resource 2 has only one unit ($b2 = 1$), and resource 3 has $b3 = 3$ units. The following row specifies a cost rate for each level of the resource. A positive entry in the row corresponding to activity j indicates the resource(s) required by the activity-each activity must be allocated exactly one unit of each resource. However, the resource may be utilized at any of its specified levels. For instance, resource 2 has only 1 unit available, which can be utilized at any of its 3 levels: if, for activity 2, it is utilized at level 1 (the lowest level), then the processing time is 7 (i.e.; $p\ (2,2,1) = 7$) and the cost is 14 (i.e.; $c\ (2,2,1) = 14$); if it is utilized at level 2 (the intermediate level), then the processing time is 5 and the cost is 25; finally, if it is utilized at level 3 (the highest level), then the processing time is 3 and the cost is 30; etc.

Suppose that all four resources are allocated at their respective level 2 (normal intensity). Letting p_j denote the duration of activity j and c_j the total cost of activity j, we get:

- activity 1 shall take $p1 = max\{6, 8, 12\} = 12$ time units; $c_j = 18 + 24 + 48 = 90$ monetary units;
- activity 2 shall take $p2 = max\{5, 5\} = 5$ time units; $c_j = 25 + 20 = 45$ monetary units;
- activity 3 shall take $p3 = max\{12, 16\} = 16$ time units; $c_j = 36 + 80 = 116$ monetary units;

and the project shall consume $max\ \{12 + 5, 16\} = 17$ time units to complete. However, due to resource restrictions, activities 2 and 3 cannot be executed at the same time since resource 2 has only one unit which must be allocated to either activity. So, if allocated first to activity 2, the project should consume $max\ \{12 + 5, 17 + 16\} = 33$ time units; if allocated first to activity 3, the project should consume $max\ \{16 + 5, 16\} = 21$ time units. The latter decision should be the preferred one. So, the total resource cost of the project shall be 251 monetary units and will be finished at time $T = 21$ (assuming the project started at time $T = 0$). We also consider lateness costs and earliness gains (negative costs). If the project specified due date is $Ts = 24$, the project will finish early. If the unitary cost for earliness is equal to -10, the total cost will be $251 - 10 * 3 = 221$.

3. Mathematical Model

Briefly, the constraints of this problem are:

• Precedence relationships among the activities.
• A unit of a resource is allocated to at most one activity at any time (the unit of the resource may be idle during an interval) at one level.
• Capacity of each resource: the number of units allocated for processing at any time should not

exceed the capacity of the resource to which these units belong.
• An activity can be started only when it is sequence-feasible and all the requisite resources are available, each perhaps at its own level, and must continue at that level for all the resources without interruption or preemption.

The objective is to find an optimal solution that minimizes the overall project cost, while respecting a specified delivery date. A penalty is incurred for tardiness beyond the specified delivery date, or a reward is secured for early completion.

Consider the following variables:

Input variables
$G(N, A)$: Project network in AoA representation with a set N of nodes and a set A of activities.
n: number of nodes; $n = |N|$.
m: number of arcs or number of activities; $m = |A|$.
(i, j): activity, represented by arc (i, j).
r: resource $r \in R$.
L_r: set of levels for resource r.
$\eta_{i,j}$: the count of resources required by activity (i, j).
ρ: number of resources, $\rho = |R|$.
b_r: capacity of resource r.
$\gamma(r, l)$: marginal cost of resource r at level l (\$/period).
γ_E: marginal gain from early completion of the project (\$/period).
γ_L: marginal loss (penalty) from late completion of the project (\$/period).
$p(i, j, r, l)$: the processing time of activity (i, j) when resource r is allocated at level l (time period).
T_s: target completion time of the project (time period).

State variables
C^k: the kth uniformly directed cutset *(udc)* of the project network that is traversed by the project progression (i.e., a set of on-going activities); $k = 1, \dots, K$.
t_i : time of realization of node i (AoA representation), where node 1 is the "start node" of the project and node n its "end node" (time period).

Decision variables
$x(i, j, r, l)$: a binary variable, of value 1 if resource r is allocated to activity (i, j) at level l, and 0 otherwise.l is the level at which a resource is applied to an activity $l \in L_r$.

Output variables
c_E: earliness cost (\$).
c_T: tardiness cost. (\$).
c_{ET}: earliness-tardiness cost. (\$).
c_R : total resource cost for all project activities (\$).
TC: total cost of the project (\$).

Next, we present the relevant constraints.

We begin by defining the processing time of an activity as the maximum of the processing times imposed by the different resources. These processing times will be a function of the levels at which the resources required by the activity are allocated, and an activity cannot start before all the preceding activities have finished, we have

$$t_j - t_i \geq \max \{p(i, j, r, l) * x(i, j, r, l)\}, \forall\, i, j \in N, \forall\, r \in R, \forall\, l \in L_r\ . \tag{1}$$

The total units of a resource allocated at any time to all the activities should not exceed the capacity of the resource to which these units belong. This restriction is applicable to the activities that are concurrently active (i.e., on-going), which must lie in the same *uniformly directed cutset (udc).*

The total allocation of resource r to the active activities in the "current" *udc* C^k cannot exceed its available capacity

$$\sum_{i,j \in C^k} x(i,j,r,l) \leq b_r, \forall\, r \in R, \forall\, l \in L_r, k = 1, \dots, K\,. \tag{2}$$

A unit of a resource is allocated to an activity at only one level (the unit of the resource may be idle during an interval of time):

$$\sum_{l \in L_r} x(i,j,r,l) = 1, \forall\, i,j \in N, \forall\, r \in R\,. \tag{3}$$

An activity must be allocated all the resources it needs at some level, at which time it can be started and must continue at the same level for all the resources without interruption or preemption. This requirement is represented as follows:

$$\eta_{i,j} - \sum_{r \in R} \sum_{l \in L_r} x(i,j,r,l) = 0, \forall\, i,j \in C^K\,. \tag{4}$$

The difficulty in implementing this constraint set stems from the fact that we do not know a priori the identity of the udc's that shall be traversed during the execution of the project, since that depends on resource allocation. The allocation of the resources is bounded by their availabilities at each udc, but the latter cannot be known until after the allocation of resources has been determined. Enumerating all the udc’s and constraining the resources usage at each one would over-constrain the problem (see Ramachandra and Elmaghraby, 2006). There are several ways to resolve this difficulty, formal as well as heuristic. The formal ones are of the integer programming genre, which, when combined with the nonlinear mathematical programming model presented above, present a formidable computing burden. On the other hand, the heuristic approaches are more amenable to computing.

The objective function is composed of two parts: the cost of use of the resources, and the gain or loss due to earliness or tardiness; respectively, of the project completion time (t_n) relative to its due date.

Earliness and tardiness (delay) are defined by:

$$e \geq T_s - t_n\,, \tag{5}$$

$$d \geq t_n - T_s\,, \tag{6}$$

$$e, d \geq 0. \tag{7}$$

The costs may be evaluated as follows:

a) the cost of resource utilization in the selected level, for each activity is:

$$c_R(i,j) = \sum_{r \in R} \sum_{l \in L_r} c(i,j,r,l) * x(i,j,r,l)\,. \tag{8}$$

$$c(i,j,r,l) = \gamma(r,l) * p(i,j,r,l). \tag{9}$$

b) the earliness-tardiness costs are:

$$c_{ET} = c_E + c_T = \gamma_E \cdot e + \gamma_L \cdot d. \tag{10}$$

c) total resources cost for all activities of project:

$$c_R = \sum_{i,j \in N} c_R(i,j). \tag{11}$$

d) total cost of project:

$$TC = C_R + C_{ET} \tag{12}$$

The desired objective function may be written simply as:

$$\min TC \tag{13}$$

4. Solution method

Our initial approach to solve the problem on hand relies on a BFS scheme. In the BFS scheme, all the nodes (partial solutions) in the search tree are evaluated at each stage, before going any deeper (Figure 2), subsequently realizing an exhaustive search that visits all nodes of the search tree. The B&B search technique can be seen as a polished BFS, since it applies some criteria in order to reduce the complexity of the BFS scheme. Usually, it consists of keeping track of the best solution found so far and checking if the solution given by that node is greater than the best known solution. So, if that node cannot offer a better solution than the solution obtained so far, the node is fathomed. The B&B approach is more efficient if the bounds are tight.

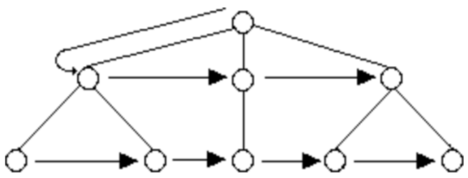


Figure 2. BFS traversal.

The B&B process consists of two procedures:

i) Subset generation;
ii) Subset elimination;

The former (subset generation) is accomplished by branching, where a set of descendent nodes is generated, thereby creating a tree-like structure. The latter (subset elimination) is realized through either bounding, where upper and lower bounds are evaluated at each node, or via feasibility checking, where the extension of a partial solution is deemed infeasible, and the branch is fathomed. A bound can be strong, which is usually harder to calculate but it accelerates finding an optimal solution, or it can be weak, which is easier to calculate but makes it slower to find the desired solution.

In our case, the objective is to minimize the total cost encountered, that is based on the bonus achieved or the penalty cost incurred while respecting or exceeding the specified due date, respectively. As a result, finding a bound depends on the following three project parameters cited: the penalty cost, bonus cost and due date. As noted above, a bound helps in reducing the search while not discarding potentially desirable branches.

A "filtered beam" search is a heuristic B&B procedure that uses breadth-first search but only the top "best" nodes are kept. At each stage of the tree, it generates all successors for the selected nodes at the current stage, but only stores a predetermined number of descendent nodes at each stage, called the beam width.

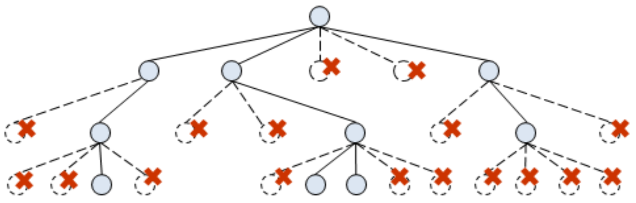


Figure 3. A beam search tree with beam width = 3.

In the proposed procedure, we have the option of using either the BFS or the FBS scheme. For the latter, we need to specify an appropriated beam width.

We consider the activities to be in one of four states: "to begin", "pending", "active" (i.e., on-going) and "finished". To get the first activities with which to initiate the process, we search all activities that do not have any predecessors. These activities are set to the state "to begin". All others are set to the state "pending".

Activities in the state "to begin" are analyzed in order to check resources availability. If we have enough resources, all activities in the state "to begin" are modified to the state "active"; otherwise, we apply, in sequence, the following rules, until the resources conflicts are resolved:

a) Give priority to activities that are precedents to a larger number of "pending" activities.
b) Give priority to activities that use fewer resources.
c) Give priority to activities in sequence of arrival to the state "to begin".

An "event" represents the starting time of one or more activities, and the project begins at event 0 in which no activity has started yet. Each activity must be allocated exactly one unit of each resource. For each active activity, we calculate all the possible combinations of levels of resources. Then, we aggregate all these combinations to get the initial combinations of allocation modes for all "active" activities. These initial combinations form branches through which we will get possible solutions for the project. All combinations have a copy of the resource availability information, and activities' current state.

If the FBS scheme is selected to obtain a solution, then:

1. If the number of combinations is less than the beam width value, all combinations are kept.
2. Otherwise, the set of combinations must be reduced to the beam width value, so some combinations need to be discarded. To evaluate the best combinations, we may pick one of the following rules:

   Select the top best combinations that have:

   - Minimum Duration.
   - Minimum Cost.
   - Minimum Cost/Duration.

   Not all combinations of the set can be directly compared because the number of activities that have been scheduled in each combination may differ. So, the combinations are grouped by the number of activities that have already been scheduled.

   Then, the combinations are compared with the others that belong to the same group. The final set is composed by a share of combinations of each group formed before.

   The ratio of each group in the final combinations set is calculated by:

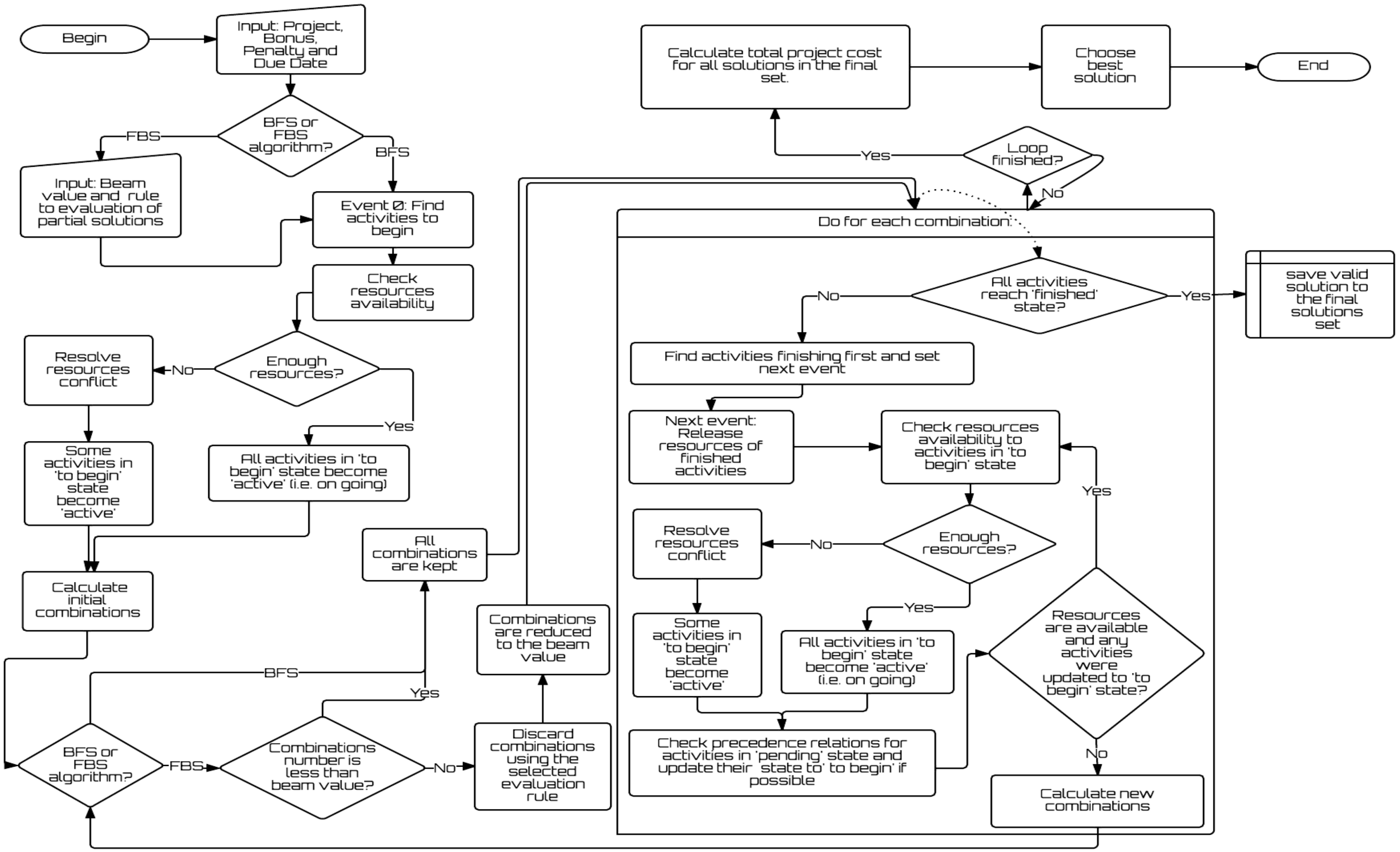
$$ratio = groupCount / totalCombinations \qquad (14)$$

In either case, we continue applying the following procedure to each combination:

3. To all activities in progress, we find the ones that will be finished first, and set that time as the next event.
4. We update activities found in step 1 to state "finished", and release all the resources being used by them.
5. For all activities in the state "to begin", we seek the ones that can begin, the same way we did when initiating the project. Activities in the state "to begin" are analyzed in order to check resource availability. If no resource conflict exists, all activities in the state "to begin" are set to state "active" and resources are set as being used; otherwise, we apply in sequence, the rules described above.
6. For all activities in the state "pending", we check for precedence relationships. For all activities that are precedence-feasible, their state is updated to state "to begin". These activities are not combined with the previous set of "to begin" activities to give priority to activities that entered first in this state.
7. If there are resources available and any pending activities were set "to begin", we apply step 5 again.
8. For all new "active" activities, we set their start time to the next event found in step 3, and determine all the possible combinations of its resource levels. Then we join all found combinations for these activities, getting new combinations to add to the actual combination being analyzed. This generates new branches for investigation.
9. We continue by applying step 1 (or 3) to each new combination until all activities are set to state "finished".
10. When all activities reach the "finished" state, we obtain a valid solution for the problem.

We evaluate the project completion time and the total cost incurred for all complete solutions, and choose the best among them.

A flowchart of the proposed solution method is shown on Figure 4.



Begin
Input: Project, Bonus, Penalty and Due Date
BFS or FBS algorithm?
FBS
BFS
Input: Beam value and rule to evaluation of partial solutions
Event 0: Find activities to begin
Check resources availability
Enough resources?
No
Yes
Resolve resources conflict
Some activities in 'to begin' state become 'active'
All activities in 'to begin' state become 'active' (i.e. on going)
Calculate initial combinations
BFS or FBS algorithm?
BFS
FBS
Combinations number is less than beam value?
Yes
No
All combinations are kept
Discard combinations using the selected evaluation rule
Combinations are reduced to the beam value
Calculate total project cost for all solutions in the final set.
Choose best solution
End
Loop finished?
Yes
No
Do for each combination
All activities reach 'finished' state?
No
Yes
save valid solution to the final solutions set
Find activities finishing first and set next event
Next event: Release resources of finished activities
Check resources availability to activities in 'to begin' state
Enough resources?
No
Yes
Resolve resources conflict
Some activities in 'to begin' state become 'active'
All activities in 'to begin' state become 'active' (i.e. on going)
Check precedence relations for activities in 'pending' state and update their state to' to begin' if possible
Resources are available and any activities were updated to 'to begin' state?
Yes
No
Calculate new combinations

Figure 4. A flowchart of the proposed solution method.

5. Computational results and analysis

The proposed solution method was implemented in C#, an Object-Oriented Programming (OOP) language, using Visual Studio 2010. To construct the project network (in AoN), we used Graph#, an open source library for .Net/WPF applications that is based on a previous library QuickGraph. These libraries support GraphML that is an XML-based file format for graphs, although we defined our own particular xml format.

The following computational tests were performed on an Intel® Pentium® M @1.20GHz 1.25GB RAM.

5.1. Three-activity network

Consider a three-activity network for 4 resources, one with 2 levels and the others with 3 different levels, respectively. Assume the following parameter values for earliness and lateness costs: $\gamma_E = -10$, $\gamma_L = 20$, and the due date, $T_S = 24$.

Table 3: Three-activity network solution values obtained using the BFS scheme.

t_n	C_E	C_T	C_R	TC	Runtime (ms)
16	80	0	230	150	44

Table 4: Three-activity network solution values obtained using the FBS scheme.

Beam Width	Evaluation Type																	
	Cost						Duration						Cost/Duration					
	t_n	C_E	C_T	C_R	TC	Runtime (ms)	t_n	C_E	C_T	C_R	TC	Runtime (ms)	t_n	C_E	C_T	C_R	TC	Runtime (ms)
20	25	0	20	200	220	8	16	80	0	230	150	2	29	0	100	196	296	11
50	24	0	0	205	205	14	16	80	0	230	150	4	28	0	80	193	273	22
200	20	40	0	218	178	51	16	80	0	230	150	68	26	0	40	201	241	61
450	16	80	0	230	150	198	16	80	0	230	150	319	24	0	0	205	205	177
700	16	80	0	230	150	162	16	80	0	230	150	112	22	20	0	213	193	116
900	16	80	0	230	150	431	16	80	0	230	150	134	19	50	0	225	175	197

The BFS scheme generates 972 combinations for the three-activity network. We used a beam width between 20 and 900. As we can see by the results exhibited in Table 4, the "Duration" evaluation type was the best for this network, achieving the same result that was obtained for the BFS scheme, even for the smaller beam width. The evaluation type "Cost" performed better than the "Cost/Duration", which did not achieve the best solution even with a beam width of 900. However, "Cost/Duration" evaluation gave the lowest project cost when the bonus or penalty, $C_R = 193$ was not considered.

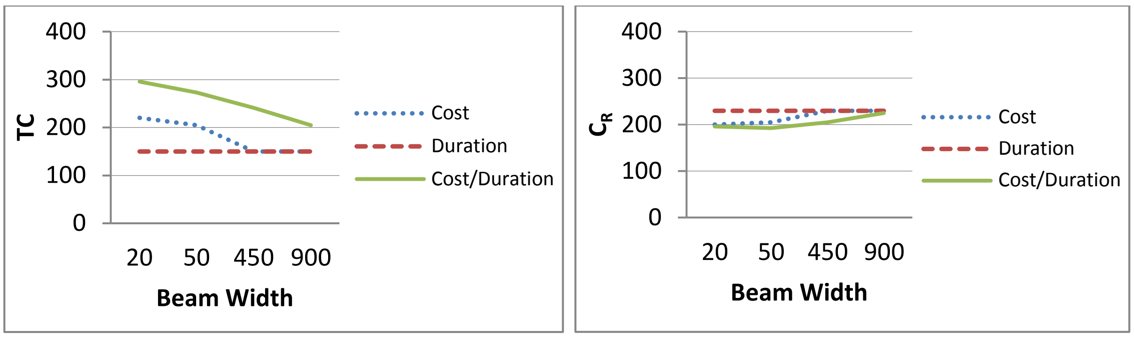


Figure 5. Variations of TC and C_R values versus beam width for evaluation types cost, duration, and cost/duration.

As can be observed from the figure above, the quality of the solutions achieved increases, i.e., the value of the TC decreases or remains equal with increase in the beam width value. This does not happen for the project cost C_R. On the contrary, this variable is highest for the best solutions. The reason for this difference has to do with the bonus and due date specified. These values make us achieve best solutions with unprofitable C_R values, because these complete the project earlier. If the earliness and lateness costs where: $\gamma_E = 0$, $\gamma_L = 0$, the best solution would be $C_R = 189$, $TC = 189$ and $t_n = 30$.

5.2. Five-activity network

Consider a five-activity network with the same resources as the three-activity network above. Assume the following parameter values for earliness and lateness costs: $\gamma_E = -10$, $\gamma_L = 20$, and the due date $T_S = 30$.

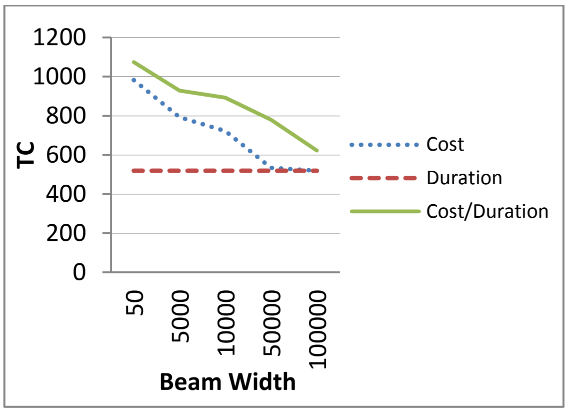
Table 5: Five-activity network solution values, obtained using the BFS scheme.

t_n	C_E	C_T	C_R	TC	Runtime (s)
36	0	120	400	520	18

Table 6: Five-activity network solution values obtained using the FBS scheme.

Beam Width	Evaluation Type																	
	Cost						Duration						Cost/Duration					
	t_n	C_E	C_T	C_R	TC	Runtime (s)	t_n	C_E	C_T	C_R	TC	Runtime (s)	t_n	C_E	C_T	C_R	TC	Runtime (s)
50	63	0	660	323	983	0.03	36	0	120	400	520	0.02	68	0	760	315	1075	0.03
500	58	0	560	333	893	0.3	36	0	120	400	520	0.31	65	0	700	319	1019	0.24
5000	52	0	440	353	793	3.6	36	0	120	400	520	4.4	60	0	600	329	929	3.71
10000	48	0	360	363	723	5.3	36	0	120	400	520	8.0	58	0	560	333	893	6.2
50000	37	0	140	395	535	17.0	36	0	120	400	520	17.8	51	0	420	360	780	14.7
100000	36	0	120	400	520	19.3	36	0	120	400	520	25.3	42	0	240	383	623	20.0

The BFS scheme generates 104,976 combinations for the five-activity network. We varied beam width between 50 and 100,000. As we can see by the results exhibited in Table 6, the "Duration" evaluation type was faster in reaching results similar to the ones obtained for the BFS scheme. The other evaluation types are far from the solution obtained for the BFS algorithm using lowest beam widths, but achieved better C_R (project cost without bonus or penalty) values, $C_R = 323$ for "Cost" and $C_R = 315$ for "Cost/Duration" type.



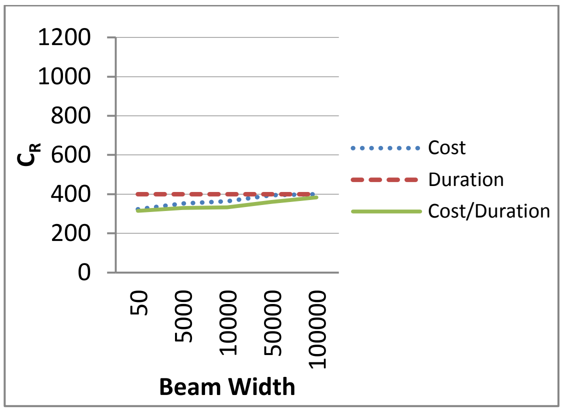


Figure 6. Variations of TC and C_R values versus beam width for evaluation types cost, duration, and cost/duration.

Again, we obtain better TC for worst C_R values, but in this case, the bonus is never materialized. For this project, our method could not find a solution with a completion time less than or equal to the due date, 30; the earliest time for completion is 36. If the earliness and lateness costs where: $\gamma_E = 0$, $\gamma_L = 0$, the best solution would be $C_R = 315$, $TC = 315$ and $t_n = 68$.

A plot of the TC and C_R values obtained for the BFS scheme, and $\gamma_E = -10$, $\gamma_L = 20$, against several due dates, is shown in Figure 7.

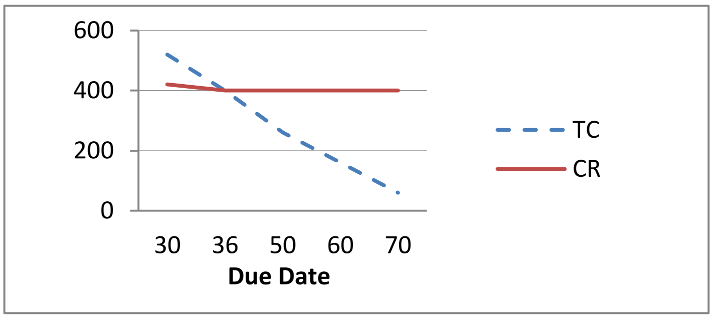


Figure 7. Variation of TC and C_R values versus due dates for the BFS scheme.

Note that starting from $T_S = 36$, the total project cost takes advantage of the bonus, independent of the particular bonus and penalty cost parameter values. So, for this project, it would be profitable to adjust the due date specified first, to a value higher than 36.

5.3. Ten-activity network

Now, consider a ten-activity network for 5 different resources, three of them with 2 possible levels, one having 5 levels and the last one with 3 elective levels. Assume the following rates for earliness and lateness costs: $\gamma_E = -15$, $\gamma_L = 20$, and the due date, $T_S = 36$.

A solution could not be achieved in a reasonable time for the BFS scheme.

Table 7: Ten-activity network solution values obtained using the FBS scheme.

Beam Width	Evaluation Type																	
	Cost						Duration						Cost/Duration					
	t_n	C_E	C_T	C_R	TC	Runtime (s)	t_n	C_E	C_T	C_R	TC	Runtime (s)	t_n	C_E	C_T	C_R	TC	Runtime (s)
50	29	15	0	406	391	0.2	24	90	0	508	418	0.2	45	0	340	456	796	0.16
500	29	15	0	406	391	3.2	26	60	0	468	408	4.1	45	0	300	444	744	3.3
1000	29	15	0	406	391	4.9	26	60	0	468	408	6.6	45	0	300	444	744	5.6
5000	27	45	0	417	372	17.2	26	60	0	468	408	31.3	44	0	320	447	727	19.4
10000	27	45	0	417	372	40.5	24	90	0	490	400	50.8	44	0	280	440	720	38.1
50000	27	45	0	405	360	723	23	90	0	480	375	635	42	0	250	451	691	355

We observed a performance decrement in runtime values. This project, besides having more activities than the previous ones, also has more resources and resources levels. Therefore, it is of higher complexity than the previous ones. The evaluation type "Cost" provides the best solutions, with a $\mathrm{TC} = 360$ for a beam width of 50,000. We achieved reasonable solutions for "Duration". However, weak solutions were obtained for "Cost/Duration".

As can be observed in Figure 8, there is no significant difference between the TC and the C_R values. However, the "Cost/Duration" values are worse, as the penalty cost for all beam width values was applied to this one.

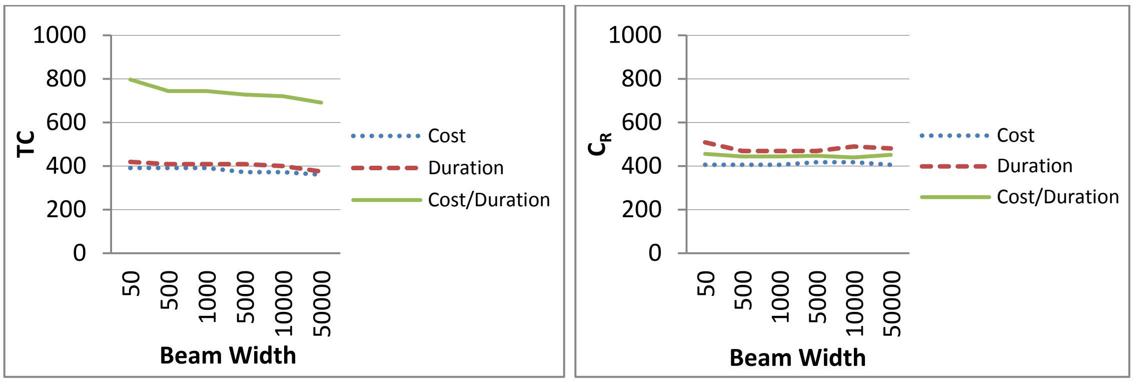


Figure 8.Variations of TC and C_R values versus beam width.

Let's analyze the TC and C_R, obtained for different due dates, using the same γ_E, and γ_L. The beam width analyzed is the 50,000, for the "Cost" and the "Duration" evaluation types.

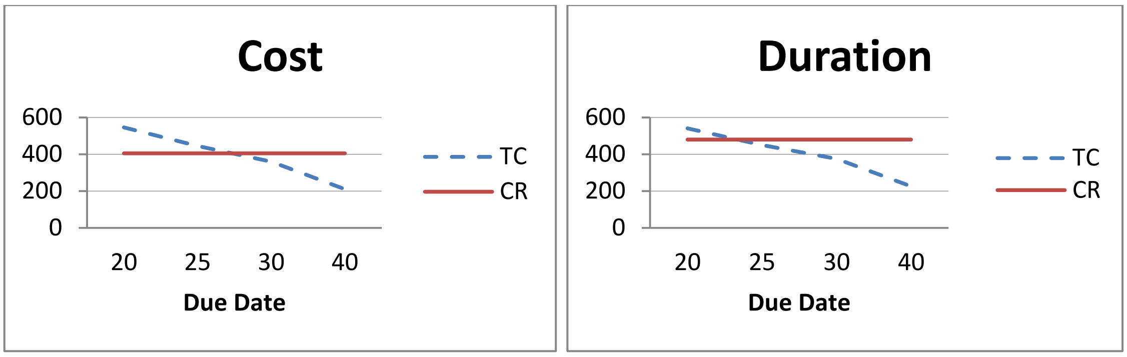


Figure 9. Variation of TC and C_R values versus due dates for evaluation types cost and duration.

For both evaluation types, there is a due date from which the total project cost takes advantage from the bonus; this due date is greater than 27 for "Cost" and greater than 23 for "Duration".

5.4. Twenty-activity network

Consider a twenty-activity network for the 4 resources with 3 different levels each, with the following parameter values for earliness and lateness costs: $\gamma_E = -10$, $\gamma_L = 20$, and the due date $T_S = 70$.

Table 8: Twenty-activity network solution totals obtained using FBS scheme.

Beam Width	Evaluation Type																	
	Cost						Duration						Cost/Duration					
	t_n	C_E	C_T	C_R	TC	Runtime (s)	t_n	C_E	C_T	C_R	TC	Runtime (s)	t_n	C_E	C_T	C_R	TC	Runtime (s)
50	135	0	1300	846	2146	2.34	70	0	0	1502	1502	1.9	135	0	1300	846	2146	1.34
500	110	0	800	902	1702	3.17	70	0	0	1502	1502	10.4	129	0	1180	865	2045	4.34
1000	110	0	800	902	1702	15.9	70	0	0	1492	1492	26.2	129	0	1180	865	2045	13.7
2000	110	0	800	902	1702	25.0	70	0	0	1479	1479	164	129	0	1180	865	2045	23.2
3000	110	0	800	902	1702	31.9	70	0	0	1479	1479	257	129	0	1180	864	2044	35.4
5000	110	0	800	902	1702	57.3	-	-	-	-	-	-	129	0	1180	864	2044	51.7
10000	110	0	800	902	1702	79.0	-	-	-	-	-	-	129	0	1180	864	2044	84.2

The proposed method could not achieve a solution using "Duration" evaluation type and beam widths of 5,000 and 10,000, because of computing memory limitations. However the best solutions were achieved by the "Duration" evaluation type, $TC = 1479$. Once again, the "Cost/Duration" type performed poorly, and the "Cost" type, even with a twenty times higher beam width, did not achieve a better solution than the one obtained by beam width of 500. The bad performance of the "Cost" type is directly related to the due date specified, 70. For a due date, $T_S = 135$, we would get a $TC = 846$, which is identical to the C_R obtained for a beam width of 50.

In Figure 10, we depict variations of the TC and C_R values, over different due dates, using the same γ_E, and γ_L values. The beam width used is 3,000, for the "Cost" and the "Duration" evaluation types.

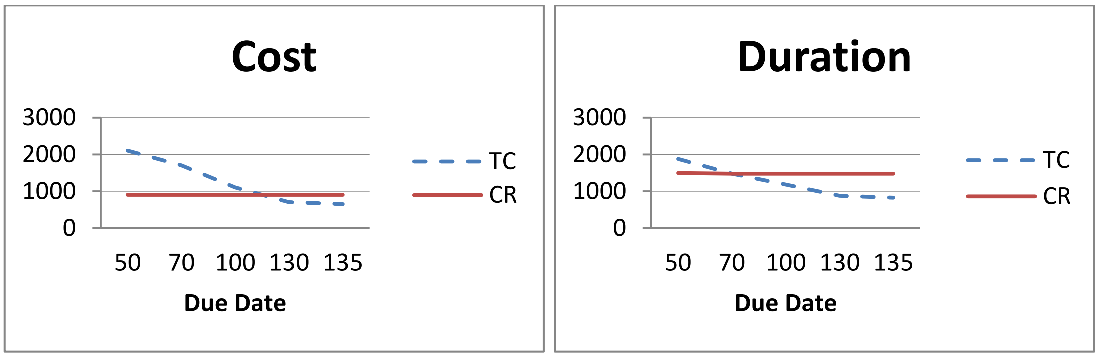


Figure 10. Variations of TC and C_R versus different due dates for evaluation types cost and duration.

The due dates, for which the total project cost takes advantage of the bonus, are 110 and 70, for "Cost" type and "Duration" type evaluations, respectively.

For $\gamma_E = 0$, $\gamma_L = 0$, the solution obtained for "Cost" evaluation type is $TC = C_R = 846$ and $t_n = 135$. For the "Duration" evaluation type we have $TC = C_R = 1479$ and $t_n = 70$. So, according to the penalty and bonus values, there is the possibility for adjusting the due date in order to get the advantage of the best of these solutions.

5.5. Thirty-activity network

Next, we considered a thirty-activity network for 4 different resources, 3 of them with 2 possible levels and one having 4 levels. Assume the following rates for earliness and lateness costs: $\gamma_E = -10$, $\gamma_L = 10$, and the due date $\mathrm{T_S} = 100$.

Table 9: Thirty-activity network solution values obtained using the FBS scheme.

Beam Width	Evaluation Type																	
	Cost						Duration						Cost/Duration					
	t_n	C_E	C_T	C_R	TC	Runtime (s)	t_n	C_E	C_T	C_R	TC	Runtime (s)	t_n	C_E	C_T	C_R	TC	Runtime (s)
50	87	130	0	696	566	1.2	94	60	0	668	608	1.2	101	0	10	731	741	1.3
500	87	130	0	687	557	4.2	87	130	0	746	616	2.6	100	0	0	725	725	3.3
1000	87	130	0	687	557	8.0	87	130	0	746	616	12.9	100	0	0	725	725	6.8
2000	87	130	0	687	557	47.6	87	130	0	746	616	51.8	99	10	0	688	678	4.9
5000	87	130	0	687	557	160	87	130	0	746	616	163	95	50	0	719	669	135
10000	87	130	0	687	557	242	87	130	0	746	616	346	95	50	0	697	647	178

The best solution found for this network was $TC = 557$ with evaluation type "Cost". In the solutions achieved using the "Duration" type, we obtain a better solution for the minimum beam width than that for the higher ones. For a beam width of 50, the algorithm was able to preserve lowest cost solutions, even though with a small range of branches in the search tree. The higher beam width values gave best solutions in terms of project duration, but they were inferior in terms of total project cost calculations. Again, the “Cost/Duration” type achieved worst solutions than for the other two evaluation types. The variations of TC and C_R values over different beam widths for each of the evaluation types are depicted in Figure 11.

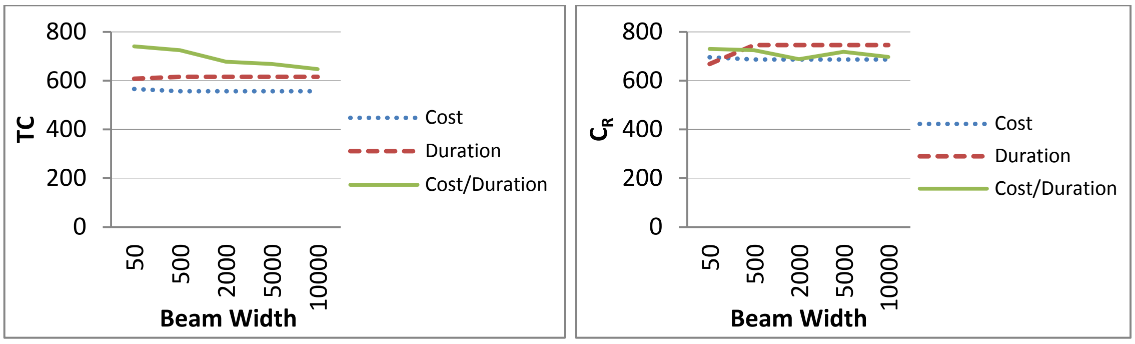


Figure 11. Variation of TC and C_R versus beam width for evaluation types cost, duration and cost/duration.

With $\gamma_E = 0$, $\gamma_L = 0$, the best solution for “Cost” is $TC = C_R = 668$ and $t_n = 94$. For the “Duration” evaluation type we have is $TC = C_R = 746$ and $t_n = 87$.

Observing the graphics of the TC and C_R values, obtained for different due dates, using $\gamma_E = -10$, $\gamma_L = 10$, and beam width of 5,000, note that they both have the same due date from which a bonus or a penalty is applied $T_S = 87$.

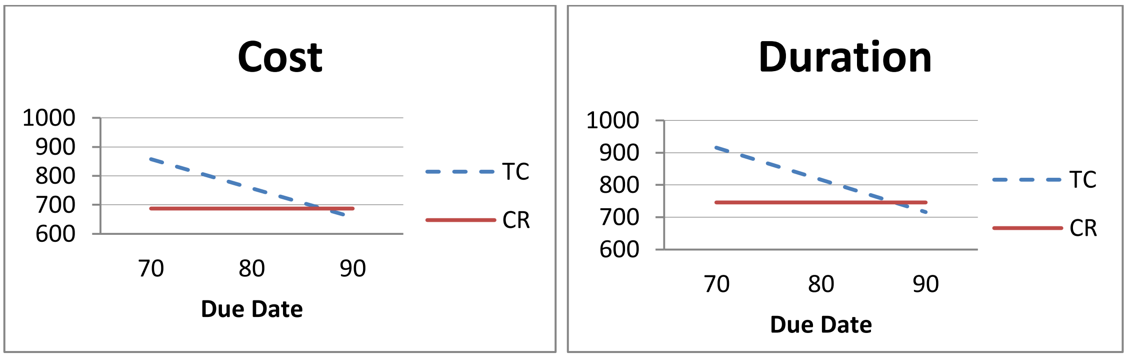


Figure 12. Variation of TC and C_R versus different Due Dates for evaluation types cost and duration.

5.6. Remark

The performance of an evaluation type seems to be intrinsically reliant on project characteristics, and especially, on the defined values of bonus, penalty and due date. For complex project networks, an increase in beam width until it is computationally feasible to obtain a solution, does not offer, necessarily, better solutions. For each project, there exists a specific due date, beyond which the bonus or the penalty is realized. Knowing the recommended solution for a project (obtained using different evaluations types) without considering the bonus, penalty or due date, can be useful, especially when there is a possibility of negotiating their values.

6. Conclusions and Further Research

The RCPSP belongs to the class of NP-hard problems (Blazewicz et al., 1983). However, this problem becomes more difficult to solve when practical issues such as multi-objective, multi-mode, and multi-project are included. A heuristic-based approach is the best approach to use in such a case.

In this chapter, we have addressed a RCPSP with multiple resource modes available at different levels. Given a due date, the objective is to allocate resources to all activities of the project so as to minimize the total cost encountered because of resource utilization, plus the net gain (bonus) accrued from finishing the project earlier or the penalty incurred for finishing the project late. We have presented a mathematical formulation for this problem, and have developed a filtered beam search-based method for its solution. This is essentially a branch-and-bound-based method except for a limited number of branches that are kept at a node.

Different criteria were used to evaluate a node, namely, cost, duration, and cost/duration. A cost-based criterion was found to generate better solutions, as expected. However, we observed that even the duration-based criterion generated good solutions (lowest cost values) for smaller beam width. The cost/duration evaluation criterion was found to always give inferior results. Although we have developed an effective procedure for the solution of this problem, yet it requires further investigation to study the problem’s inherent properties, which can further aid in obtaining solutions of better quality in reasonable CPU times.

References

1. Arroub, M., Kadrou, Y., Najid, N. (2010) ‘An efficient algorithm for the multi-mode resource constrained project scheduling problem with resource flexibility’, *International Journal of Mathematics in Operational Research,* 2:6,748 – 761.
2. Bandelloni, M., Tucci, M., Rinaldi, R. (1994) ‘Optimal resource leveling using non-serial dynamic programming’, *European Journal of Operational Research*, 78: 2, 162-177.
3. Basnet, C., Tang G. e Yamaguchi T. (2001) ‘A Beam Search Heuristic for Multi-Mode Single Resource Constrained Project Scheduling’, *In proceedings of 36th Annual Conference of the Operational Research Society of New Zealand , Christchurch, NZ, Nov-Dec, 1-8.*
4. Bellman, R., Dreyfus, S. (1959) ‘Functional approximations and dynamic programming’, *Mathematical Tables and Other Aids to Computation,* 13, 247-251.
5. Berthold, T., Heinz, S., Lübbecke, M.E., Möhring, R.H., Schulz, J. (2010) ’A constraint integer programming approach for resource-constrained project scheduling’, *In proceedings of CPAIOR 2010, LNCS, June, Andrea Lodi, Michela Milano, Paolo Toth (eds), Springer*, 6140, 51-55.
6. Blazewicz, J., Lenstra, J.K. e Rinnooy Kan, A.H.G. (1983) ‘Scheduling subject to resource constraints: classification and complexity’, *Discrete Applied Mathematics*, 5:1, 11-24.
7. Boctor, F.F. (1990) ‘Some efficient multi-heuristic procedures for resource constrained project scheduling’ *European Journal of Operational Research*, 49, 3-13.
8. Boctor, F.F. (1993) ‘Heuristics for scheduling projects with resource restrictions and several resource-duration modes’, *International Journal of Production Research,* 31, 2547-2558.
9. Clark, C.E. (1962) ‘The PERT Model for the Distribution of an Activity Time’, *Operations Research*, 10:3, 405-406.
10. Davis, E.W. (1966) ‘Resource allocation in project network models - a survey’, *Journal of Industrial Engineering*, 17: 4, 17, 177-188.
11. Dean, B.V., Denzler, D.R., Watkins, J.J. (1992) ‘Multiproject staff scheduling with variable resource constraints’, *IEEE Transactions on Engineering Management,* 39, 59-72.
12. Demeulemeester, E.L., Herroelen, W.S. (1996) ‘An Efficient Optimal Solution Procedure for the Preemptive Resource-Constrained Scheduling Problem’, *European Journal of Operational Research,* 90, 334-348.
13. Dodin, B.M, Elmaghraby, S.E. (1985) ‘Approximating the Criticality Indices in the Activities in PERT Networks’, *Management Science,* 207-23.
14. Elmaghraby, S.E. (1992) ‘Resource allocation via dynamic programming in activity networks’, *European Journal of Operational Research*, 88, 50-86.
15. Elmaghraby, S.E., Herroelen, W.S. (1980), 'On the measurement of complexity in activity networks', *European Journal of Operational Research,* 5:4, 223–234.
16. Elmaghraby, S.E., Herroelen, W.S. (1990) ‘The scheduling of activities to maximize the net present value of projects’, *European Journal of Operational Research,* 49, 35-40.
17. Etgar, R., Shtub, A., LeBlanc, L.J. (1997) ‘Scheduling projects to maximize net present value - the case of time-dependent, contingent cash flows’, *European Journal of Operational Research,* 96, 90-96.
18. Gonçalves, J.F., Mendes, J.J.M., Resende, M.G.C. (2004) ‘A Genetic Algorithm for the Resource Constrained Multi-Project Scheduling Problem’, *Technical ReportTD-668LM4,* AT&T Labs Research.
19. Guldemond, T., Hurink, J., Paulus J., Schutten, J. (2008) ‘Time-constrained project scheduling’, *Journal of Scheduling,* 11: 2, 137-148.
20. Hartmann, S. (2001) ‘Project scheduling with multiple modes: a genetic algorithm’, *Annals of Operational Research,* 102 111-135.
21. Heilmann, R. (2000) ‘Resource–constrained project scheduling: a heuristic for the multi–mode case’, *OR Spektrum* 23, 335–357.
22. Herroelen, W. (2006), ‘Project scheduling–theory and practice’, *Production and Operations Management,* 14:4, 413–432.
23. Kazaz, B., C. Sepil. 1996. Project scheduling with discounted cash flows and progress payments. Journal of the Operational Research Society 47, 1262-1272.
24. Kelley, J.E., Morgan, R., Walker (1959) ‘Critical Path Planning and Scheduling’, *In proceedings of Eastern Joint Computer Conference, Boston, December 1-3, 1959, NY 1960*, 160-173.
25. Kis, T. (2005) ‘A branch-and-cut algorithm for scheduling of projects with variable-intensity activities’, *Mathematical Programming, Springer Berlin / Heidelberg,* 103:3, 515-539.
26. MacCrimmon, K.R., Ryavec, C.A. (1964) ‘An Analytical Study of the PERT Assumptions’, *Operations Research*, 12:1, 16-37.

27. Metropolis, N., Rosenbluth, A., Rosenbluth,M., Teller,A., Teller,E. (1953) ‘Equation of state calculations by fast computing machines’, *Journal of Chemical Physics,* 21, 1087–1092.
28. Mika M., Waligora G., Weglarz G. (2005) ‘Simulated annealing and tabu search for multi-mode resource-constrained project scheduling with positive discounted cash flows and different payment models’, *European Journal of Operational Research*, 164:3, 639-668.
29. Nemhauser, G. L., Wolsey, L. A. (1988) *Integer and Combinatorial Optimization,* Wiley-Interscience, Hoboken, NJ, USA.
30. Ozdamar, L., Ulusoy, G. (1995) ‘A Survey on the Resource-Constrained Project Scheduling Problem’, *IIE Transactions,* 27, 574-586.
31. Padman, R., Dayanard, N. (1997) ‘On modelling payments in projects’, *Journal of the Operational Research Society,* 48, 906-918.
32. Patterson, J.H., Slowinski, R., Talbot, F.B., Weglarz, J. (1989) ‘An algorithm for a general class of precedence and resource constrained scheduling problems’, In Slowinski, R. and Weglarz, J. (eds) *Advances in project scheduling,* Elsevier, Amsterdam, pp. 3-28.
33. Patterson, J.H., Slowinski, R., Talbot, F.B., Weglarz, J. (1990) ‘Computational experience with a backtracking algorithm for solving a general class of precedence and resource constrained scheduling problems’, *European Journal of Operational Research*, 49, 68-7.
34. Pritsker, A., Watters, L., Wolfe, P. (1969) ‘Multi-project scheduling with limited resources: a zero-one programming approach’, *Management Science*, 16, 93-108.
35. Ragsdale, C. (1989) ‘The current state of network simulation in project management theory and practice’, *Omega: The International Journal of Management Science*, 17, 21-25.
36. Ramachandra, G., Elmaghraby, S.E. (2006), ‘Sequencing precedence-related jobs on two machines to minimize the weighted completion time’, International Journal of Production Economics, 100 (1), 44-58.
37. Santos, M.A., & Tereso, A.P. (2011b) ‘On the Multi-mode, Multi-skill Resource Constrained Project Scheduling Problem - A Software Application’, In A. GasparCunha, R. Takahashi, G. Schaefer & L. Costa (eds) *Soft Computing in Industrial Applications*, 96, 239-248.
38. Santos, M.A., Tereso A.P. (2010a) ‘On the Multi-Mode, Multi-Skill Resource Constraint Project Scheduling Problem (MRCPSP-MS)’, *In proceedings of 2nd International Conference on Engineering Optimization (EngOpt 2010), Lisbon – Portugal, September 6-9.*
39. Santos, M.A., Tereso, A.P. (2011a) ‘On the multi-mode, multi-skill resource constrained project scheduling problem – computational results’, *In proceedings of ICOPEV - International Conference on Project Economic Evaluation, Guimarães – Portugal, April 28-29*.
40. Sepil, C., Ortac N. (1997) ‘Performance of the Heuristic Procedures for Constrained Projects with Progress Payments’, *Journal of the Operational Research Society,* 48, 1123-1130.
41. Tereso A.P., Araújo M.M., Elmaghraby S.E. (2004). Adaptive Resource Allocation in Multimodal Activity Networks. International Journal of Production Economics 92,1-10.
42. Tereso A.P., Mota J.R., Lameiro R.J. (2006). Adaptive Resource Allocation Technique to Stochastic Multimodal Projects: a distributed platform implementation in JAVA. Control Cybern 35, 661-686.
43. Tseng, C. (2008) ‘Two Heuristic Algorithms for a Multi-Mode Resource-Constrained Multi-Project Scheduling Problem’, *Journal of Science and Engineering Technology,* 4:2, 63-74.
44. Ulusoy, G., Cebelli, S. (2000) ‘An equitable approach to the payment scheduling problem in project management’, *European Journal of Operational Research,* 127, 262–278.
45. Vanhoucke, M., Demeulemeester, E. and Herroelen, W., (2000), ‘An exact procedure for the resource-constrained weighted earliness-tardiness project scheduling problem’, *Annals of Operations Research,* 102, 179-196.
46. Willis, R.J. (1985) ‘Critical path analysis and resource constrained project scheduling theory and practice’, *European Journal of Operational Research,* 21, 149-155.
47. Zhang H., Li H. and Tarn CM. (2006) 'Heuristic scheduling of resource-constrained', multiple-mode and repetitive projects', *Construction Management and Economics* 24, 159-169.
48. Zimmermann, J. and H. Engelhardt (1998) ‘Lower Bounds and Exact Algorithms for Resource Levelling Problems’, *Technical Report WIOR-517*, University of Karlsruhe.