A note on time/cost tradeoff curve generation for project scheduling with multi-mode resource availability costs

Denise Sato Yamashita

Reinaldo Morabito

Production Engineering Department, Universidade Federal de São Carlos

13565-905, São Carlos, SP, Brazil

E-mail: denisesy@dep.ufscar.br, morabito@power.ufscar.br

Abstract: In this note we combine two known algorithms and show how they can be used in order to generate tradeoff curves between time and cost for deterministic project scheduling problems with multiple modes and resource availability costs. The approach can handle linear and non-linear non-decreasing cost functions and it is based on the exact algorithm presented in Demeulemeester (1995) for the resource availability cost problem without multiple modes. As the problem is NP-hard, the method is computationally viable to solve only problems of a moderate size. The performance of the combined algorithm is evaluated solving different problem instances generated by the software Progen, and the solutions are compared to the ones generated by a modeling language with the solver CPLEX.

Keywords: project scheduling, resource availability cost, multiple modes, time/cost tradeoff curve, exact algorithm.

1. Introduction

This note deals with a deterministic project scheduling problem of minimizing resource availability costs considering the project completion time, the precedence relations among the activities and the multiple modes to execute an activity. The special case of this problem where there is only one mode to execute each activity is known as the *resource availability cost* *problem* (RACP). The RACP was introduced in Möhring (1984) where it was shown to be NP-hard. Drexl and Kimms (2001) point out that the solution of the RACP for different project deadlines provides interesting time/cost tradeoffs, which is valuable information for negotiating the price of a project.

The RACP differs from the *resource constrained project scheduling problem* (RCPSP) in that the former seeks to minimize a non-decreasing discrete cost function of unlimited resources while completing the project by a given deadline, whereas the latter seeks to minimize the project deadline or makespan (i.e., the completion time of the last activity of the project) with limited resources. Surveys on exact and heuristic methods for the RCPSP are found in Herroelen et al. (1998), Kolisch and Hartmann (1998), Brucker et al. (1999), Kolisch and Padman (2001), Kolisch and Hartmann (2006) and Lancaster and Ozbayrak (2007).

The literature on solution methods for the RACP is relatively scarce. Möhring (1984) presented an exact approach for the RACP and applied it to a bridge construction. Demeulemeester (1995) proposed an exact algorithm for the RACP that solves a sequence of RCPSPs, here called *D-algorithm* for short. Drexl and Kimms (2001) proposed two lower bound procedures for the RACP based on Lagrangean relaxation and column generation methods. Yamashita et al. (2006) developed a scatter search procedure for the RACP and applied it to a number of test instances. We are not aware of exact solution methods for the problem addressed here, the *multi-mode resource availability cost problem* (MMRACP), which extends the RACP by considering multiple modes to execute an activity. To the best of our knowledge, there is only one heuristic procedure for the MMRACP recently proposed in Hsu and Kim (2005), which is based on priority rules. The project scheduling model becomes more general and realistic as it considers multiple modes to execute the activities, obviously at the expense of an increase in the number of decision variables. Recent studies on project scheduling models with multiple modes are found in e.g. Alcaraz et al., (2003), Bouleimen and Lecocq (2003), Brucker and Knust (2003), Mika et al. (2005), Zhu et al. (2006), Buddhakulsomsiri and Kim (2007).

In this note, we combine two known exact algorithm and show how it can be used to solve the MMRACP and generate tradeoff curves between the project completion time and the costs of resource requirements. As Chassiakos and Sakellaropoulos (2005) point out, time-cost analysis is an important element of project scheduling, in particular, for lengthy and costly construction projects. Our approach is based on the D-algorithm and it uses the branch-and-bound method presented in Sprecher and Drexl (1998) to solve the *multi-mode resource constrained project scheduling problems* (MMRCPSP) involved in the algorithm. As the MMRACP is NP-hard, the algorithm is computationally reasonable to solve only problems of a moderate size. We also explore the potential of the approach to handle nonlinear non-decreasing cost functions concerning resource availabilities. To evaluate the method, a number of test instances were generated using Progen (Kolisch and Sprecher, 1996), software that has been extensively used for various project scheduling problems. The performance of the combined algorithm is compared to the modeling language GAMS (Brooke et al., 1998) with the solver CPLEX.

The remainder of the note is organized as follows. Section 2 contains a brief description of the problem and an illustrative example. In Section 3, we describe how the D-algorithm was adapted to solve the MMRACP and generate tradeoff curves between the project deadline and the resource availability costs. Computational tests are reported in Section 4 and concluding remarks are presented in Section 5.

2. Problem description

Consider a project with n activities subject to finish-start precedence relations $(h,j) \in H$, where activities 1 and n are dummy activities that indicate the starting and finishing time of the project, respectively. Each activity j if executed in mode i ($i = 1, \ldots, M_j$) has a duration d_{ji} and requires r_{jik} units of resource type k (k = 1, ..., m) over its duration. We assume that the resources are renewable, like workforce and machines.

Let D represent the project deadline and $C_k(a_k)$ denote a linear or non-linear non-decreasing cost function associated with the availability a_k of resource type k. A mathematical model for the MMRACP, based on the model presented in Talbot (1982) for the MMRCPSP, is described below. The decision variables are the finishing times of the activities and the resource availabilities:

- $$x_{jit} = \begin{cases} 1, \text{ if activity } j = 1,\ldots,n \text{ is processed in mode } i = 1,\ldots,M_j \text{ and finishes at} \\ \text{ time instant } t = 1,\ldots,D. \\ 0, \text{ otherwise.} \end{cases}$$

- $a = (a_1,\ldots, a_m)$ is the vector of resource availability variables, where a_k is the availability of resource type k, k = 1,..., m.

The remaining input data for the MMRACP is:

- LF_j = the latest finishing time in which activity j can be processed, without violating project deadline D, satisfying the precedence relations among the activities in H, and considering unlimited resource availabilities. LF_j can be found straightforward by working backwards through the network, starting from dummy activity n, which is scheduled at time instant D. The remaining activities are scheduled iteratively, as soon as all their successors have already been scheduled. Since we can have more than one mode to execute an activity, in order to compute LF_j we select the mode that yields the smallest duration for activity j.

- EF_j = the earliest finishing time in which activity j can be completed without violating the precedence relations among the activities, and ignoring the resource constraints. Similar to LF_j, it is computed straightforwardly by forward scheduling, starting with dummy activity 1, which is scheduled at time instant 0. Activities are then scheduled as soon as their predecessors have been scheduled.

$$\text{Minimize} \qquad \sum_{k=1}^{m} C_k(a_k) \tag{1}$$

$$\sum_{i=1}^{M_j} \sum_{t=EF_j}^{LF_j} x_{jit} = 1, \quad j = 1,\ldots,n \tag{2}$$

$$\sum_{i=1}^{M_h} \sum_{t=EF_h}^{LF_h} t \cdot x_{hit} \leq \sum_{i=1}^{M_j} \sum_{t=EF_j}^{LF_j} (t - d_{ji}) \cdot x_{jit}, \quad j = 1,\ldots,n, \text{ and all } h,\ (h, j) \in H \tag{3}$$

$$\sum_{j=1}^{n} \sum_{i=1}^{M_j} \sum_{b=t}^{t+d_{ji}-1} r_{jik} x_{jib} \leq a_k, \quad k = 1,\ldots,m, \quad t = 1,\ldots,D \tag{4}$$

$$x_{jit} \in \{0,1\}, \quad j = 1,\ldots,n, \quad i = 1,\ldots,M_j, \quad t = 1,\ldots,D \tag{5}$$

$$a_k \geq 0, \quad a_k \in Z, \quad k = 1,\ldots,m \tag{6}$$

In this formulation of the MMRACP, (1) refers to the objective function that minimizes the total resource availability cost of the project. Restrictions (2) impose that each activity is executed exactly once and in only one mode. Restrictions (3) guarantee that the precedence relations among the activities is respected, and restrictions (4) guarantee that the total amount of type-*k* resources required by the activities in a time instant *t* does not exceed the amount of resources available for the project. Finally, restrictions (5) define the binary decision variables and restrictions (6) assure that the resource availability variables are non-negative integers. Note in formulation (1)-(6) that the completion time of the project is considered implicitly, since $t \leq D$ for all binary variables x_{jit}.

An illustrative example

We present an illustrative example for the MMRACP. Figure 1 shows the precedence relations among the activities for a project instance with n = 7 activities and m = 3 resource types, called example 1. Table 1 presents the duration d_{ji} and the amount of resources r_{jik} required by each activity j, when executed in mode i. The resource availability cost of the project is a linear function given by $c_1a_1 + c_2\ a_2 + c_3a_3$, where $c_1 = 1$, $c_2 = 5$, $c_3 = 2$, and the deadline of the project is $D = 20$. If the resource availabilities for the project are $a_1 = 2$, $a_2 = 4$ and $a_3 = 6$, then the corresponding cost associated with these availabilities is equal to 34.

Figure 1. Precedence relations for example 1.

Table 1. Input data for example 1

An infeasible schedule for this project is displayed in Figure 2. The horizontal axis of Figure 2 shows the starting and finishing times of each activity, and the vertical axis shows the amount of resources of type k = 1, 2, 3, available for the project. Note that activity 4 is executed in mode 2, while the other activities are executed in mode 1. This schedule is infeasible with respect to the project's deadline, $D = 20$, however, it is feasible with respect to the maximum resource usage (which has to be smaller than 2 units for resource type 1, 4 units for resource type 2 and 6 units for resource type 3).

Figure 2. Gantt diagram for example 1

Models (1)-(6) assume that the decision maker has a deadline D for the project and the goal is to find out the scheduling of the activities and the resource availabilities, so that the cost of resource requirements is minimized and the project can be done within this deadline. Alternatively, there can be situations where the decision maker has a limited budget $\overline{C}$ for the project and he or she wishes to minimize the completion time of the project within this budget limit. This situation is described by model (7)-(9) below. Note that the objective function (7) now minimizes the project completion time and restriction (9) refers to the budget limitation of the project. Both models (1)-(6) and (7)-(9) could be used to generate tradeoff curves between time and cost, nevertheless, in this note we choose arbitrarily the first model.

$$\text{Minimize} \quad \sum_{t=EF_n}^{LF_n} t \cdot x_{n1t} \tag{7}$$

subject to (2), (3), (4), (5) and (6) (8)

$$\sum_{k=1}^{m} C_k(a_k) \leq \overline{C} \tag{9}$$

3. Generating time/cost tradeoff curves for the MMRACP

In this section we describe how we combine the D-algorithm and the algorithm proposed by Sprecher and Drexl (1998) to consider multiple modes of execution for the activities and to generate time/cost tradeoff curves for the MMRACP. Since the solution procedure relies on solving a sequence of MMRCPSPs, first we present a mathematical model for the MMRCPSP. The main differences between the MMRACP and the MMRCPSP are the objective function and the resource availability variables a_k, $k = 1,\ldots, m$. In the MMRCPSP, the objective function is to minimize the project makespan and the resource availability is an input data for the problem.

It can be formulated as (Talbot, 1982):

$$\text{Minimize} \quad \sum_{t=EF_n}^{LF_n} t \cdot x_{n1t} \tag{10}$$

subject to (2), (3), (4) and (5). (11)

As in model (1)-(6), model (10)-(11) includes the constraints (2)-(5) to ensure that each activity is executed exactly once and in only one mode, and to make the project feasible with respect to the precedence relations and resource limitation. It is worth noting that, unlike model (1)-(6), this model does not include restrictions (6), since the resource availabilities a_k, $k = 1,\ldots m,$ are parameters rather than variables.

Hartmann (2000) compared several branch-and-bound algorithms proposed in the literature to solve the MMRCPSP (model (10)-(11)), and ranked the algorithm proposed by Sprecher and Drexl (1998) among the best. Hence, in this work the MMRCPSP is solved by the branch-and-bound algorithm in Sprecher and Drexl (1998). The algorithm generates a search tree, the nodes of which are partial schedules, and branching occurs by selecting the next activity that is scheduled, as well as its execution mode.

The enumeration procedure starts by scheduling activity 1 at time instant $t = 0$. At each level of the search tree, we determine the set of activities that were already scheduled (partial schedule) and the set of *candidate activities*, that is, activities that were not scheduled yet but are precedence feasible. The next step is to select an activity from the set of candidate activities and a respective execution mode. This generates a new branch in the search tree. We proceed this way until all activities are scheduled, and therefore, we obtain a makespan for the project. More details of this branch and bound algorithm for the MMRCPSP can be found in Sprecher and Drexl (1998).

Solution procedure of the MMRACP

In order to adapt the D-algorithm to handle multiple modes of execution for the activities and to generate a time/cost tradeoff curve, two main modifications are made to the original algorithm:

- Instead of solving a sequence of RCPSPs, the algorithm solves a sequence of MMRCPSPs.
- A procedure to generate a time/cost tradeoff curve is added to the algorithm.

Since the modified algorithm does not alter the main steps of the D-algorithm, and the MMRCPSPs involved are solved by an exact solution procedure, the algorithm is exact for the MMRACP.

The algorithm starts by specifying a deadline range, (D_{min} , D_{max}). In the next step, the current deadline value is set to D_{max} and a lower bound limit for the amount of resources available is computed. Once the resource availability is fixed, a MMRCPSP (model (10)-(11)) is solved. If it is possible to find a feasible solution for the MMRCPSP, then we have found an optimal solution for the MMRACP. Otherwise, we create a set of candidate solutions *ES*, where a candidate solution $a = (a_1,\ldots, a_m)$ is the vector of resource availability variables. Initially, this set consists of candidates´ solutions generated from a lower bound solution, marginally increasing its resource availabilities. These new solutions are inserted in *ES* if they are efficient points. Point $(a_1,\ldots,a_m)$ is an efficient point if there is no other point $(a_1^{'},\ldots,a_m^{'})$ in the solution space such as $a_k^{'} \leq a_k$ for all k = 1,.., m. For example, suppose we have *ES* ={(5,2,3), (4,5,7), (5,5,1)}, then (4,6,7) is not an efficient point, since point (4,5,7) is already in *ES*.

The next step is to solve the MMRCPSP corresponding to the cheapest efficient point. If the MMRCPSP is infeasible with respect to the current deadline, this efficient point is eliminated from *ES* and the algorithm generates m solutions from this infeasible solution, increasing the resource availabilities, one resource type at a time, by one unit. These new solutions are included in *ES* if they are efficient points and the search proceeds by solving the MMRCPSP corresponding to the cheapest efficient point. This process is repeated until a feasible solution is found, that is, the project finishes before the current deadline and the resource limits are not violated at any time while the project is being executed. This is the optimal solution for the MMRACP for the current deadline.

The procedure to generate the time/cost tradeoff curve starts by decreasing the current deadline value by one unit (or more, if it suits the decision maker) and repeating the process of examining *ES*, until finding the cheapest efficient point that finishes before the current deadline. Note that, rather than creating a new set *ES* from scratch, we save some computational time by using the set found in the previous iteration as a starting point in the new iteration. The procedure stops when the current deadline is smaller than D_{min}. It should be noted that this algorithm is computationally viable in situations where it is reasonable to consider the resource availability as integer numbers and in relatively small amounts (e.g., few dozens of workers, instead of hundreds or thousands of workers).

A concise pseudo code for the solution procedure is presented in Figure 3, followed by a discussion of its application to the illustrative example of Section 2. Step 1 computes a simple lower bound ($b_1,...,b_m$) for the vector of resource availabilities ($a_1,...,a_m$). The main goal of Step 2 is to improve the lower bound b_k: first we choose a resource type k and set $a_k = b_k$, while the remaining m-1 resource types l, $l \neq k$, are set to their upper bound value $a_l = u_l$, and then we solve a MMRCPSP for this resource availability ($a_1,...,a_m$). If this solution is feasible, the lower bound b_k cannot be improved, otherwise, a_k (the variable that was initially set to b_k) is increased by one unit, i.e., $a_k = b_k+1$, and a new MMRCPSP is solved for this new vector of resource availabilities ($a_1,...,a_m$). This procedure is repeated until a feasible solution is found. Note that this procedure results in an optimal solution for the MMRACP if we have only one resource type ($m = 1$).

In Step 3, the procedure starts by generating candidate solutions by combining two resource types: resource type 1 and resource type k, $k \neq 1$. Initially, these two resource types receive the value of the lower bound, $a_1 = b_1$ and $a_k = b_k$, and the remaining resource types receive the upper bound value. The objective is to find a set of tuples (a_1, a_k) that result in a feasible schedule, such as if we decrease one of the resource types, a_1 or a_k, the solution becomes infeasible. For each resource type k, $k \neq 1$, these values are stored in set PES_k, that consists of tuples (a_1, a_k) where the second element is the minimum amount of resource type k necessary when only a_1 units of resource type 1 are available (where a_1 is the first element of the tuple). Set PES_k is updated as follows: we check if there is already a tuple in PES_k, where the first element is equal to a_1. If this is the case, the second element of the tuple is replaced by a_k. Otherwise, a new tuple is added to PES_k where a_1 is the value of the first element and a_k is the value of the second element. Set EP is a set related to PES_k, and it contains all values a_1 for which a tuple was added to PES_k. Set EP is updated as follows: we check if there is an element in *EP* equal to the current value a_1. If such an element does not exist, then it is added to the set. Note that if $m = 2$, Step 3 finds an optimal solution for the MMRACP.

In Step 4, we create set *ES*, a set of the candidate solutions obtained from the elements of PES_k. Note that these solutions are lower bounds for the MMRACP. For each element $a_1 \in EP$, we seek tuples in PES_k, where the first element is the greatest element smaller than or equal to a_1, and we set a_k to the value of the second element, $k = 1, 2.,..., m$. Then, we check if this lower bound can be included in *ES*. If there is no element in *ES* with the same resource values, and if there is no other element in *ES* that dominates the current solution, that is, there is no $y = (y_1,..., y_m)$, $y \in ES$, such as, $y_k \le a_k$, for all $k = 1,..., m$, then the current element is inserted in *ES*. Otherwise, no other element is inserted in *ES*.

In Step 5, the resource availability is the one defined by the element in *ES* with the smallest cost. If this resource availability yields a feasible solution, then an optimal solution for the current deadline is found. Otherwise, the algorithm generates *m* solutions from this infeasible solution, increasing the resource availabilities, one resource type at a time, by one unit. These new solutions are included in *ES,* if they are efficient points.

In Step 6, we generate the time/cost tradeoff curve. As we reach this step, we already have a solution for the MMRACP for the current deadline, and one iteration of the algorithm is completed. We proceed by updating the current deadline, decreasing it by at least one unit. The algorithm stops if the current deadline is smaller than D_{min} (other stop criteria can be also used, for instance, a computational time limit or a maximum cost for the project). If the stop criterion is not satisfied, we generate a new point of the tradeoff curve. First, we update the lower bound value b_k using a similar procedure as the one described in Step 2. Note that, since the current deadline is smaller than the deadline of the previous iteration, the new lower bound value b_k is greater than or equal to the lower bounds computed in the previous iteration. Therefore, we do not have to compute b_k from scratch, instead, we start a new computation of b_k using the value of b_k obtained in the previous iteration. After b_k is updated, we examine list *ES* and we update the elements $y_k \in ES$ where $y_k \leq b_k$ for some k = 1,…,m, by increasing the value of y_k to b_k. The new costs of the updated elements in ES are computed, the set ES is resorted and Step 5 is repeated.

Figure 3. Pseudo code of the combined algorithm

Algorithm application to the illustrative example

In order to illustrate the application of the algorithm, we use the example presented in Section 2.

Step 1: We set the current deadline to D = 20. The lower bound is computed: $b_1 = 1,\ b_2 = 3,\ b_3 = 3$.

Step 2: The upper bound value is computed by $u_k = \sum_{j=1}^{n} \max_i (r_{jik})$, thus u_1 = 7, u_2 = 11 and u_3 = 11. The objective in this step is to improve the lower bound value. It starts by assigning a lower bound value to resource type 1, and the remaining resource types receive the upper bound value, in other words, the resulting vector of resource availabilities is given by (a_1, a_2, a_3) = (1,11,11). Then, a MMRCPSP is solved for this vector of resource availabilities, and we obtain a makespan value that exceeds the project's deadline. Therefore, resource type 1 is increased by one unit and we determine the makespan for (a_1, a_2, a_3) = (2,11,11). This solution is feasible and therefore, a new lower bound for resource type 1 is given by $b_1 = 2$. This procedure to determine the lower bound is repeated for resource type 2: we evaluate (7,3,11) and we obtain an infeasible solution. We increase the availability of resource type 2 and we obtain (7,4,11), which is a feasible solution, thus, the new lower bound for resource type 2 is $b_1 = 4$. We proceed in this way for resource type 3 and we find the value $b_3 = 4$. The lower bounds for each resource type at the end of Step 2 are $b_1 = 2, b_2 = 4, b_3 = 4$.

Step 3: We start with resource type 2, k = 2, and we enumerate all pairs ($a_{1,}$ a_2) that result in a feasible solution, such as if we decrease one of these resource types by one unit, the solution becomes infeasible, that is, (a_1-1, a_2, a_3) and ($a_{1,}$ a_2-1, a_3) are infeasible. As in the D-algorithm, we begin with solution $(a_1, a_2, a_3) = (b_1, b_2, u_3) = (2,4,11)$. Resource a_2 is increased by one unit and a MMRCPSP is solved for this vector of resource availability. Since it yields an unfeasible solution, a_2 is increased again, that is, a_2 = 6, and the corresponding vector of resource availability results in a feasible solution. After updating PES_2, we have PES_2 = {(2,6)} and EP = {2}. The availability of resource a_1 is increased by one unit, a_1 = 2+1 = 3, and a_2 is decreased by one unit, until the solution becomes infeasible or a lower bound for this resource type is reached. Thus we have,

- (3,5,11) – feasible solution, PES_2 = {(2,6),(3,5)} and EP = {2,3}.
- (3,4,11) – feasible solution, PES_2 = {(2,6),(3,4)} and EP = {2,3}.

Note that a_2 = b_2, therefore a_2 cannot be reduced anymore as it reached the lower bound value b_2. This procedure is repeated for k = 3:

- (2,11,4) – infeasible solution, then increase resource type 3 by one unit
- (2,11,5) – infeasible solution, then increase resource type 3 by one unit
- (2,11,6) – feasible solution, PES_3 = {(2,6)} and EP = {2,3} (2 is already in EP)

The procedure proceeds as described in the algorithm and it yields PES_3 = {(2,6), (3,4)} and EP = {2,3}.

Step 4: The first element in EP is 2, and from set PES_2 we obtain a_2 = 6, and from set PES_3 we obtain a_3 = 6, therefore, ES = {(2,6,6)} with cost: $1 \cdot 2 + 5 \cdot 6 + 2 \cdot 6 = 44$. The second element in EP is 3, thus, ES = {(3,4,4), (2,6,6)}, where (3,4,4) yields a cost of: $1 \cdot 3 + 5 \cdot 4 + 2 \cdot 4 = 31$. Note that ES is sorted by increasing cost.

Step 5: We start by selecting the first element of ES, (3,4,4). As this solution is infeasible, then the resource availabilities are increased, yielding the following resource vectors (see Figure 4): (4,4,4,**32**), (3,5,4,**36**), (3,4,5,**33**), where the first three numbers are the resource availabilities and the last number in bold is the cost of the project. Since these elements do not belong to *ES* and they are efficient points, they can be added to *ES* = {(4,4,4,**32**), (3,4,5,**33**), (3,5,4,**36**), (2,6,6,**44**)}. The next element to be examined is (4,4,4,**32**), and it yields a feasible solution when the MMRCPSP is solved. Therefore, we generate solutions: (5,4,4,**33**), (4,5,4,**37**), and (4,4,5,**34**). Solution (4,5,4,**37**) is dominated by (3,5,4,**36**) and solution (4,4,5,**34**) is dominated by (3,4,5,**33**) and therefore, they are not included in *ES*={(3,4,5,**33**), (5,4,4,**33**), (3,5,4,**36**), (2,6,6,**44**)}. Following the order of the elements in *ES*, we examine (3,4,5,**33**), and we find out that it is infeasible, and the following solutions are generated (4,4,5,**34**), (3,5,5,**38**) (it is not added to *ES* because it is not an efficient point) and (3,4,6,**35**), thus, *ES*={(5,4,4,**33**), (4,4,5,**34**), (3,4,6,**35**), (3,5,4,**36**), (2,6,6,**44**)}. Now, we determine the makespan for (5,4,4,**33**), and it yields an infeasible solution, hence, set *ES* is updated, *ES*={(4,4,5,**34**), (6,4,4,**34**), (3,4,6,**35**), (3,5,4,**36**), (2,6,6,**44**)}. Next, we examine (4,4,5,**34**), which results in an infeasible solution, and the new set *ES* is given by *ES*={(6,4,4,**34**), (3,4,6,**35**), (5,4,5,**35**), (3,5,4,**36**), (2,6,6,**44**)}. The algorithm proceeds by determining the makespan for the cheapest solution, (6,4,4,**34**). This solution is also infeasible and *ES*={(3,4,6,**35**), (5,4,5,**35**), (7,4,4,**35**), (3,5,4,**36**), (6,5,4,**39**), (2,6,6,**44**)}. Finally, the solution (3,4,6,**35**) has a schedule that finishes before the project's deadline, and therefore, this is an optimal vector of resource availabilities for the current deadline.

Figure 4. Illustration of Step 5: infeasible solutions are marked with "I" and solutions that are not efficient points (dominated solutions) are marked with "D".

Step 6: Tradeoff curve: Solution (3,4,6,**35**) has a schedule that finishes at time instant 20, therefore, we update the deadline value to *D* = 19 and we start a new iteration. The lower bounds from the previous iteration are $b_1 = 2, b_2 = 4, b_3 = 4$. If we take $(a_1, a_2, a_3) = (b_1, u_2, u_3)$ and we solve an RCPSMM for *D* = 19, we obtain an infeasible solution, hence, b_1 is increased to $b_1 = 3$.

We solve a RCPSMM for the vector of resource availabilities $(b_1 = 3, u_2, u_3)$ and we obtain a feasible solution, therefore value 3 is the new lower bound for resource type 1. We proceed by updating b_2, which is increased from $b_2 = 4$ to $b_2 = 6$, and b_3 is increased from $b_3 = 4$ to $b_3 = 6$. Set *ES* is updated and sorted, and the algorithm proceeds by determining the makespan for the next solution with the smallest cost, (3,6,6,**45**), which results in 15 (smaller than *D* = 19), and we add this point to the tradeoff curve. This step is repeated until the stop criterion is satisfied.

4. Computational experiments

In this section, we present the computational results obtained by the combined algorithm presented in section 3. The experiments were performed on a PC Pentium 4, 2.6 GHz, 512 Mbyte RAM. All procedures were coded in C^{++} language. The MMRACP and MMRCPSP share most input data and therefore it is relatively easy to adapt existing instances of the MMRCPSP to the MMRACP.

We use Progen (Kolisch and Sprecher, 1996) to generate instances for the MMRCPSP and adapt them to the MMRACP. The *network complexity* (*NC*) and *resource factor* (*RF*) are two important input parameters to Progen. *NC* controls the average number of immediate successors of an activity. *RF*, which ranges between 0 and 1, controls the percentage of resource types required by an activity. For example, if *RF* = 1, every activity in the project requires all *m* types of resources, while *RF* = 0 indicates that activities do not require any type of resources.

To complete the construction of a problem instance, it is also necessary to determine a completion time for the project. This deadline is computed as: $D = DF \cdot EF_n$, where *DF* is the deadline factor and EF_n is the earliest finishing time for the project. The minimum and maximum deadlines D_{min} and D_{max} are obtained by setting *DF* = 1.0 and *DF* = 1.5, respectively. For each instance, costs c_k are real numbers drawn from the uniform distribution U[1,10], and the duration of the activities are integer values sorted and rounded from the uniform distribution U[1,10].

During the search, if the algorithm does not generate all tradeoff points for the specified deadline range, the search is interrupted after reaching 3,600 seconds. The parameters used to generate the test instances are:

- *RF*: 0.25, 0.5, 0.75 and 1.0
- *NC*: 1.5, 1.8 and 2.1
- $n = 15$, $m = 4$, $M_j = 3$, $j = 1,\ldots,n$.

We have also tested two types of objective functions:

- Linear: $\sum_{k=1}^{m} C_k(a_k) = \sum_{k=1}^{m} c_k a_k$
- Nonlinear: $\sum_{k=1}^{m} C_k(a_k) = \sum_{k=1}^{m} c_k a_k^2$

For each objective function, we generated instances for all *RF* and *NC* values, that is, a test set of $4 \times 3 = 12$ problem instances. In order to assess the quality of the combined algorithm, we have also solved the instances using the modeling language GAMS (version 2.0.10.0) (Brooke *et al.*, 1998) and the solver CPLEX (version 7.0) with default parameter settings. To solve model (1)-(6) with the nonlinear cost function in GAMS/CPLEX, we transformed it into a linear model by adding binary variables:

$$z_{kl} = \begin{cases} 1, \text{ if there are } l \ (l = 1,2,\ldots,L_k) \text{ resources of type } k \text{ available for the project} \\ 0, \text{ otherwise.} \end{cases}$$

where L_k is an upper bound on the amount of type k resource.

$$\text{Minimize} \qquad \sum_{k=1}^{m} c_k \sum_{l=1}^{L_k} l^2 z_{kl} \tag{12}$$

subject to (2), (3), (4), (5), (6), and (13)

$$\sum_{l=1}^{L_k} z_{kl} = 1, \qquad k = 1,\ldots,m \tag{14}$$

$$z_{kl} \in \{0,1\}, \quad k = 1,\ldots,m, \quad l = 1,\ldots,L_k \tag{15}$$

In this formulation, (12) is equivalent to the quadratic cost function defined before, and restrictions (2)-(6) are defined in Section 2. Restrictions (14) guarantee that for each resource type k, only one value of z_{kl} is equal to 1, and restrictions (15) define the binary variables z_{kl}.

Table 2. Computational times (in seconds) for the linear model

Table 3. Computational times (in seconds) for the non-linear model

Tables 2 and 3 compare the computational runtimes (in seconds) of the combined algorithm to GAMS/CPLEX for the linear and non-linear models, respectively. Note in Table 2 that the combined algorithm solves more instances (11 out of 12) than GAMS/CPLEX (6 out of 12) within the time limit of 3,600 seconds. The algorithm is faster than GAMS/CPLEX as the *RF* value increases, that is, as the activities require more types of resources and the problem becomes more constrained.

Regarding Table 3, the combined algorithm solved 11 (out of 12) instances within the time limit, while GAMS/CPLEX solved only 4 instances. The algorithm is faster in all test instances of Table 3. In most examples, the computational times increase with the *RF* values. By comparing Tables 2 and 3, it is interesting to note that GAMS/CPLEX has a worse performance when solving a non-linear model than solving a linear model, due to the increase in the number of variables of model (12)-(15).

Figure 5 shows a graph of the time/cost tradeoff curve for one of the examples of Table 2 with 15 activities, 4 resources, $RF = 0.5$, $NC = 1.5$. The initial deadline D_{min} and the maximum deadline D_{max} values were set to 19 and 28, respectively. Note that the greatest decrease of cost occurs for the smallest deadlines values, from 19 to 21. For $D > 24$, we can observe that the cost values seem to converge. The tradeoff curves of other examples of the Tables have a similar behavior. These curves are interesting as they provide the decision maker with a quantitative tradeoff analysis between the project deadline and the resource availability cost.

Figure 5. Graph illustrating the time/cost tradeoff for an instance in Table 2.

5. Conclusions

In this note we combine two known exact algorithms and show how they can be used to generate time/cost tradeoff curves for the multiple mode resource availability cost problem (MMRACP). The method can handle linear and non-linear non-decreasing cost functions of the resource availabilities, assuming that the resources are renewable. To the best of our knowledge, there are no exact methods to solve the MMRACP in the literature. The algorithm combines the exact algorithm presented in Demeulemeester (1995) to solve the RACP, and the branch-and-bound method presented in Sprecher and Drexl (1998) to solve the different MMRCPSPs involved in the procedure. The project scheduling model becomes more general and realistic as it considers multiple modes to execute the activities, obviously at the expense of an increase in the number of decision variables.

The test results showed that the combined algorithm is computationally feasible to generate time/cost tradeoff curves for problems of a moderate size. It is worth noting that this algorithm can only be dealt with in situations where it is reasonable to consider the resource availabilities as integer numbers and in relatively small amounts. As mentioned, time/cost tradeoff curves are interesting as they provide the decision maker with a quantitative tradeoff analysis between the project deadline and the required resource availability costs.

The solutions obtained by the combined algorithm were compared to the ones of software GAMS/CPLEX, as we did not find other exact methods for the MMRACP in the literature. The algorithm was faster than GAMS/CPLEX in 18 (out of 24) test instances, while

GAMS/CPLEX was faster than the algorithm in 4 (out of 24) instances. In particular, the algorithm is faster than GAMS/CPLEX as the activities require more types of resources and the problem becomes more constrained. The computational results were mostly encouraging for the test instances with non-linear cost function, in which the combined algorithm was always faster than GAMS/CPLEX. Therefore, we believe that the algorithm is a fine starting point for the development of more efficient exact approaches to the MMRACP. An interesting perspective for future research is the extension of the method to deal with non-renewable resources.

Acknowledgements: The authors thank the anonymous referees for their helpful comments and suggestions. This research was partially supported by CNPq (grant 150337/2004-3, 522973/95-7) and FAPESP (grants 07/00209-6, 2006/03496-3).

References

Alcaraz, J., Maroto, C., Ruiz, R., 2003, 'Solving the multi-mode resource constrained project scheduling problem with genetic algorithms', *Journal of the Operational Research Society*, Vol. 54, No. 6, pp. 614-626.

Bouleimen, K., Lecocq, H., 2003, 'A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version' *European Journal of Operational Reseach*, Vol. 149, No. 1, pp. 268-281.

Brooke, A., Kendrik, D., Meeraus, A., & Raman, R., 1998. *GAMS – A user's guide*, The Scientific Press.

Brucker, P., Drexl, A., Möhring, R., Neumann, K. and Pesch, E., 1999. 'Resource constrained project scheduling: Notation, classification, models, and methods', *European Journal of Operational Research*, Vol. 112, No. 1, pp. 3-41.

Brucker, P., Knust, S., 2003, 'Lower bounds for resource-constrained project scheduling problems', *European Journal of Operational Research*, Vol. 149, No.2, pp. 302-313.

Buddhakulsomsiri, J., Kim, D.S., 2007, ‘Priority rule-based heuristic for multi-mode resource-constrained project scheduling problems with resource vacations and activity splitting’, *European Journal of Operational Research*, Vol. 178, No. 2, pp. 374-390.

Chassiakos, A.P., Sakellaropoulos, S.P., 2005, ‘Time-Cost optimization of construction projects with generalized activity constraints’, *Journal of Construction Engineering and Management*, Vol. 10, pp. 1115-1124.

Demeulemeester, E., 1995, ‘Minimizing resource availability costs in time-limited project networks’, *Management Science*, Vol. 41, No. 10, pp. 1590-1598.

Drexl, A. and Kimms, A., 2001, ‘Optimization guided lower and upper bounds for the resource investment problem’, *Journal of the Operational Research Society*, Vol. 52, No. 3, pp. 340-351.

Hartmann, S., 2000, *Project scheduling under limited resources: models, methods, and applications*, Lecture Notes in Economics and Mathematical Systems 478, Springer-Verlag, Berlin.

Herroelen W., De Reyck B. and Demeulemeester E., 1998, ‘Resource-constrained project scheduling: a survey of recent developments’, *Computers & Operations Research*, Vol. 25, No. 4, pp. 279-302.

Hsu, C-C. and Kim, D.S., 2005, ‘A new heuristic for the multi-mode resource investment problem’, *Journal of the Operational Research Society*, Vol. 56, No. 4, pp. 406-413.

Kolisch R. and Hartmann S., 1998, ‘Heuristic algorithms for the resource-constrained project scheduling problem: Classification and computational analysis’, in *Project Scheduling: Recent Models, Algorithms, and Applications*, ed. J. Weglarz, Kluwer Academic Publishers, Norwell, MA.

Kolisch, R., Hartmann, S., 2006, ‘Experimental investigation of heuristics for resource-constrained project scheduling: An update’, *European Journal of Operational Research*, Vol. 174, No. 1, pp. 23-37.

Kolisch, R. and Padman, R., 2001, ‘An integrated survey of deterministic project scheduling’, *Omega: The International Journal of Management Science*, Vol. 29, No. 3, pp. 249-272.

Kolisch, R. and Sprecher, A., 1996, ‘PSPLIB- A project scheduling library’, *European Journal of Operational Research*, Vol. 96, No. 1, pp. 205-216.

Lancaster, J., Ozbayrak, M., 2007, ‘Evolutionary algorithms applied to project scheduling problems – a survey of the state-of-the-art’, *International Journal of Production Research*, Vol. 45, No. 2, pp. 425-450.

Mika, M., Waligóra, G., Weglarz, J., 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*, Vol. 164, No. 3, pp. 639-668.

Möhring, R.F., 1984, ‘Minimizing costs of resource requirements in project networks subject to a fixed completion time’, *Operations Research*, Vol. 32, No. 1, pp. 89-120.

Sprecher, A. and Drexl, A., 1998, ‘Multi-mode resource-constrained project scheduling by a simple, general and powerful sequencing algorithm’, *European Journal of Operational Research*, Vol.107, No. 2, pp. 431-450.

Talbot, F. B., 1982, ‘Resource-constrained project scheduling problem with time-resource tradeoffs: The nonpreemptive case’, *Management Science*, Vol. 28, No. 10, pp. 1197-1210.

Yamashita, D.S., Armentano, V.A. and Laguna, M., 2006, ‘Scatter search for project scheduling with resource availability cost’, *European Journal of Operational Research*, Vol. 169, No. 2, pp. 623-637.

Zhu, G.D., Bard, J.F., Yu, G., 2006, ‘A branch-and-cut procedure for the multimode resource-constrained project-scheduling problem’, *Informs Journal on Computing*, Vol. 18, No. 3, pp.377-390.

Author Bios:

Reinaldo Morabito is an Associate Professor, Production Engineering Department, Federal University of Sao Carlos, Brazil. He earned a B.S. in civil engineering from State University of Campinas, a M.Sc. in operations research, and a Ph.D. in transportation engineering, both from University of Sao Paulo, Brazil. He was a visiting scholar at the Sloan School of Management, M.I.T., Cambridge, MA. His research interests include cutting and packing problems, queueing networks applied to manufacturing systems, probabilistic location problems, and logistics and transportation planning.

Denise Yamashita holds a pos-doc position at the Production Engineering Department, Federal University of Sao Carlos, Brazil. She earned a B.S. in mathematics from State University of Campinas, Brazil, and a M.Sc. and a Ph.D. in electrical engineering from the same university. Her research interests include exact and heuristic search methods for scheduling and cutting and packing problems.

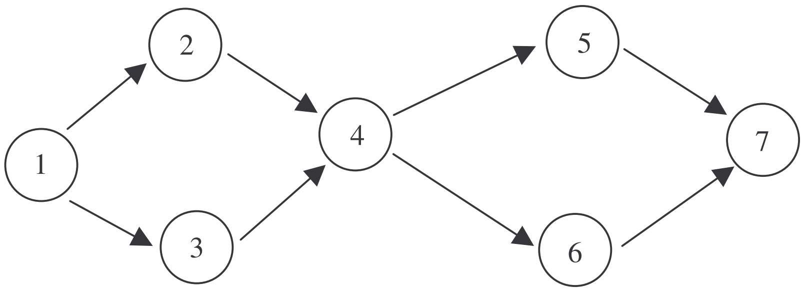

Figure 1. Precedence relations for example 1.

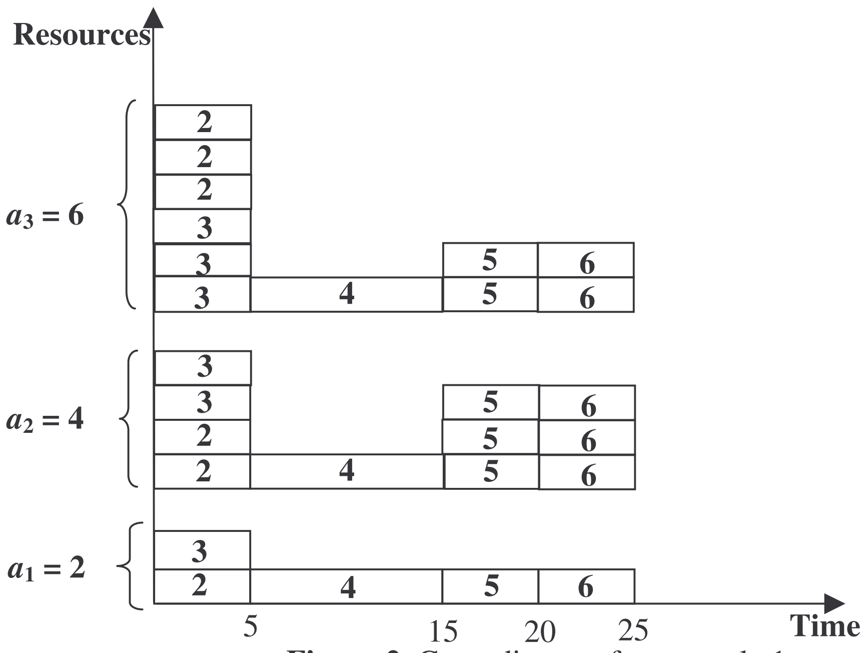

Figure 2. Gantt diagram for example 1

```
Step 1: Set $D = D_{max}$. Generate a lower bound $b_k = \min_{j,i}\{r_{jik}\}$, for each resource type $k = 1,..., m$.
Step 2: Lower bound $b_k$ is improved for each resource type $k = 1,..., m$.
Step 3: For each resource type $k = 2,...,m$, create sets $PES_k$ and $EP$.
Step 4: Create set $ES$ from set $PES_k$.
Step 5: Select a candidate solution $a$ with the smallest cost from $ES$. If the solution is feasible, then go to Step 6, as an optimal solution was found. Otherwise, generate $m$ candidate solutions by increasing the resources of the candidate solution $a$ by one unit. Include the generated solutions in $ES$ if they are efficient points. Repeat Step 5.
Step 6: Tradeoff curve. Update the deadline by decreasing it: $D \leftarrow D-1$. If $D < D_{min}$, stop. Update the lower bounds for this new deadline value. Update set $ES$ and go to Step 5.
```

Figure 3. Pseudo code of the combined algorithm

$$\begin{tikzcd}
 & & & \boxed{(3,4,4,\mathbf{31})} \arrow[dll] \arrow[d] \arrow[dr] & & \boxed{(2.6.6.\mathbf{44})} & \\
 & \boxed{(4,4,4,\mathbf{32})\text{-I}} \arrow[dl] \arrow[d] \arrow[dr] & & \boxed{(3,5,4,\mathbf{36})} & \boxed{(3,4,5,\mathbf{33})\text{-I}} \arrow[dl] \arrow[d] \arrow[dr] & & \\
\boxed{(4,4,5,\mathbf{34})\text{-D}} & \boxed{(4,5,4,\mathbf{37})\text{-D}} & \boxed{(5,4,4,\mathbf{33})\text{-I}} \arrow[dl] \arrow[d] \arrow[dr] & \boxed{(3,4,6,\mathbf{35})\text{-F}} & \boxed{(3,5,5,\mathbf{38})\text{-D}} & \boxed{(4,4,5,\mathbf{34})\text{-I}} \arrow[dl] \arrow[d] \arrow[dr] & \\
 & \boxed{(6,4,4,\mathbf{34})\text{-I}} \arrow[dl] \arrow[d] \arrow[dr] & \boxed{(5,5,4,\mathbf{38})\text{-D}} & \boxed{(5,4,5,\mathbf{35})\text{-D}} & \boxed{(5,4,5,\mathbf{35})} & \boxed{(4,5,5,\mathbf{39})\text{-D}} & \boxed{(4,4,6,\mathbf{36})\text{-D}} \\
\boxed{(7,4,4,\mathbf{35})} & \boxed{(6,5,4,\mathbf{39})\text{-D}} & \boxed{(6,4,5,\mathbf{36})\text{- D}} & & & &
\end{tikzcd}$$

Figure 4. Illustration of Step 5: infeasible solutions are marked with “I” and solutions that are not efficient points (dominated solutions) are marked with “D”.

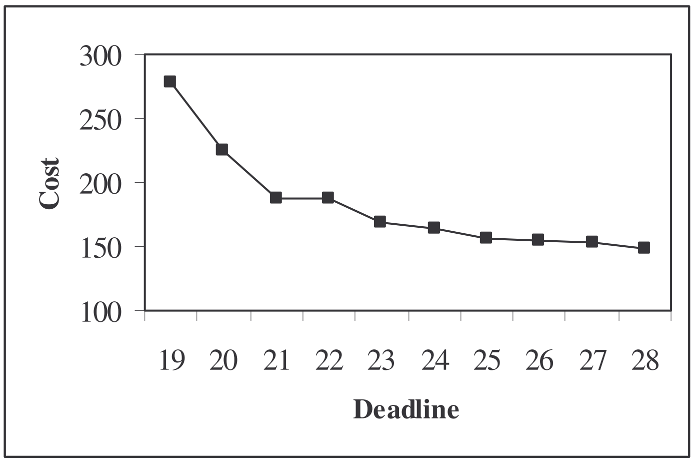

Figure 5. Graph illustrating the time/cost tradeoff for an instance in Table 2.

Table 1. Input data for example 1

Activity *j*		**Execution mode** *i*	**Duration** d_{ji}	**Resource 1** r_{ji1}	**Resource 2** r_{ji2}	**Resource 3** r_{ji3}
1	$M_1 = 1$	1	0	0	0	0
2	$M_2 = 1$	1	5	1	2	3
3	$M_3 = 1$	1	5	1	2	3
4	$M_4 = 2$	1	5	3	1	1
		2	10	1	1	1
5	$M_5 = 1$	1	5	1	3	2
6	$M_6 = 1$	1	5	1	3	2
7	$M_7 = 1$	1	0	0	0	0

Table 2. Computational times (in seconds) for the linear model

NC	*RF*	**GAMS/CPLEX**	**Combined Algorithm**
1.5	0.25	44.62	590.98
	0.50	454.85	531.67
	0.75	-	2756.25
	1.00	-	518.84
1.8	0.25	53.27	42.20
	0.50	184.12	381.36
	0.75	2645.84	1695.06
	1.00	-	1955.74
2.1	0.25	17.86	64.73
	0.50	-	42.47
	0.75	-	2064.28
	1.00	-	-

Table 3. Computational times (in seconds) for the non-linear model

NC	*RF*	**GAMS/CPLEX**	**Combined Algorithm**
1.5	0.25	1080.21	583.22
	0.50	1678.24	442.19
	0.75	-	2097.55
	1.00	-	469.84
1.8	0.25		31.49
	0.50	2501.06	418.39
	0.75	-	1209.13
	1.00	-	1529.72
2.1	0.25	327.09	36.74
	0.50	-	37.13
	0.75	-	1303.69
	1.00	-	-