Lower Bounds for the Multi-Skill Project Scheduling Problem

Emmanuel Néron
Laboratoire d'Informatique de l'Université de Tours /E3I
64 av. Jean Portalis, 37200 Tours FRANCE
neron@univ-tours.fr

Abstract

This paper deals with an extension of the classical Resource Constrained Project Scheduling Problem (RCPSP). We present a new type of resource constraints in which staff members are involved. On the one hand, we focus on staff members, each of them having several skills, i.e, is able to perform more than one kind of activity. On the other hand, an activity has specific skill requirements that must be satisfied. To solve this problem, we propose two lower bounds. The first one uses a linear programming scheme proposed for the RCPSP and the second one is based on energetic reasoning.

1. Presentation of the Multi-Skill Project Scheduling Problem

The Multi-Skill Project Scheduling Problem (MSPSP) mixes both the classical RCPSP, and the Multi-Purpose Machine model [3], [4]. From the RCPSP, we use the project description, and we add new resource constraints inspired by the Multi-Purpose Machine model. For instance let us consider that resources are staff members having more than one skill, and that each activity needs a "given amount" of skills to be performed. Thus scheduling an activity at time *t*, requires matching its skill requirements with the skills of the staff members that are available at *t*. Our goal is to minimize the overall project duration, i.e, $\min(C_{max})$.

Figure 1 presents a 4-activities and 4-members example with a feasible solution. Table 1.a gives the processing times of actives along with their skill requirements. Table 1.b, describes staff members in terms of skills. Figure 1.a, presents the precedence constraints between activities. Figure 1.b, shows a feasible solution.

Table 1.a : Activity definition

	A_1	A_2	A_3	A_4
Processing time	2	5	3	3
Programmer	-	1	2	1
DB Designer	1	-	-	1
Webmaster	1	1	-	-

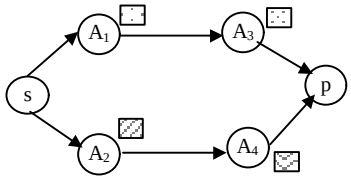

Figure 1.a : Precedence Constraints

Table 1.b : Person definition

	P_1	P_2	P_3	P_4
Programmer	-	Yes	Yes	Yes
DB Designer	Yes	-	-	-
Webmaster	Yes	Yes	-	Yes

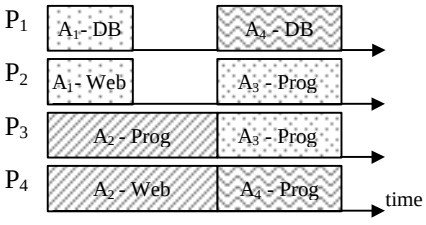

Figure 1.b : A feasible solution

Figure 1 : Example of Multi-Skill Project Scheduling Problem (MSPSP)

In the example above, one can see that activity A_3 cannot start at time 2, because it requires 2 programmers, while the available staff members at this time cannot meet this skill requirement (P_1 is not a programmer).

This model is an extension of the classical RCPSP: if we assume that all members only have one skill, we get classical resource constraints. This model can also be seen as a specific case of the Multi-Mode RCPSP [8]. The main reason to justify this new model is the huge number of modes (more than two hundred feasible modes for a medium size project) that would be necessary if we chose to model the same feasible resource assignments with the Multiple-Mode model. For instance, consider activity A_2 of the example presented in Table 1. This activity requires a Programmer (who can be P_2, P_3, P_4) and a Webmaster (who can be P_1, P_2, P_4). If we use a Multi-Mode model in which persons are considered as resources, there are six valid modes for A_3 : {(P_1, P_2), (P_2, P_4), (P_3, P_1), (P_3, P_2), (P_3, P_4), (P_1, P_4)}. For all of them the processing time for the activity is the same, only resource assignments change from a mode to another. Therefore, a decision maker that has to build a schedule would not be able to list all valid modes for the project. In our model, the listing of modes is "implicit" thanks to the notion of skill.

Let us present MSPSP notations:

- $I = \{A_1,..., A_n\}$: the set of activities to be processed without preemption.
- p_i : the processing time of A_i.
- $G= (V,E,d)$: the precedence graph in which there is a node (i) associated to each activity A_i. A starting dummy activity (s) and an ending dummy activity (p) are added. $(i,j)\in E$ if there is a precedence constraints between A_i and A_j, in that case $d_{i,j}$ which is the valuation of $(i,j) \in E$, is equal to p_i.
- $\{S_1,..., S_K\}$: the set of skills.
- $\{P_1,..., P_M\}$: the set of staff members.
- $S_{m,k} = 1$ if person P_m has the skill S_k and 0, otherwise. $\forall m\in[1..M]\ \Sigma_k\ S_{m,k} \geq 1$ indicates that a person has at least one skill.
- $b_{i,k}$: the number of persons with the skill S_k, needed to perform activity A_i,
- r_i : the release date of A_i is the longest path in G from the starting dummy activity (s) to the node (i).
- q_i : the tail of A_i is the longest path in G from the node (i) to the ending dummy activity (p), minus the processing time of A_i.

After this presentation of the Multi-Skill Project Scheduling Problem, we present two lower bounds for this problem.

2. Two lower bounds for MSPSP

In this section we present two lower bounds for MSPSP. Let's not forget that computing lower bounds for the RCPSP is a challenging problem. Lower bounds are useful, first to prove the efficiency of heuristics, and eventually to be used in branch-and-bound methods. The two lower bounds that we propose are destructive [5] in the sense that they are used to determine if a given number LB is a valid lower bound for the project duration. Once LB is fixed, a deadline d_i, is associated to each activity ($d_i = LB - q_i$).

2.1 Extension of a general linear programming scheme from RCPSP to MSPSP

The linear lower bound that we present is an adaptation of a linear programming scheme proposed by Carlier and Néron [2] for the RCPSP, which is based on a time-horizon decomposition into successive intervals. The first step is the computation of time-intervals. We assume that release dates (r_i) and deadlines (d_i) have been computed according to the precedence constraints and a given integer LB. Let $T = U_{i\in I}\ \{r_i, d_i\} = \{t_1, t_2,...,t_{L+1}\}$. We assume that T is sorted in a non-decreasing order and that all time points are different. Let:

- $\forall l\in[1..L]$, $e_l = [t_l, t_{l+1})$. L denotes the number of consecutive time intervals that must be taken into account. e_l is the l-th interval and t_l is the starting point of time-interval e_l.

- $\forall l \in [1..L]$, $\forall i \in [1..n]$, $x_{i,l}$ is the absolute part of A_i performed during e_l. $x_{i,l}$ are variables for our linear program,
- $\forall l \in [1..L]$, $\forall m \in [1...M]$, $\forall k \in [1..K]$, $\delta^l_{m,k}$, the time P_m spent during $[t_l, t_{l+1}]$ performing skill S_k. $\delta^l_{m,k}$ are variables for our linear program.

The first constraint (1) implies that the parts of activities are positive:

$$\forall i \in I\ ,\ \forall l \in [1..L],\ x_{i,l} \geq 0 \qquad (1)$$

(2) ensure that the activities are completely performed:

$$\forall i \in I,\ \sum_{l=1}^{L} x_{i,l} = p_i \qquad (2)$$

(3)-(5) are used to model that an activity must be performed within its time-window. Moreover, for each interval, the part of the activity performed during this interval must not be larger than the size of the interval itself. Notice that (4) and (5) are linear constraints since r_i, d_i and t_l are known beforehand (they are data in our linear programming formulation)

$$\forall l \in [1..L],\ \forall i \in I,\ x_{i,l} \leq t_{l+1} - t_l \qquad (3)$$

$$\forall i \in I,\ \forall l \in [1..L],\ \text{if } d_i \leq t_l \text{ then } x_{i,l} = 0 \qquad (4)$$

$$\forall i \in I,\ \forall l \in [1..L],\ \text{if } r_i \geq t_{l+1} \text{ then } x_{i,l} = 0 \qquad (5)$$

The three following equations are used to model the resource constraints. Skill requirements of activities must be met for each interval (6). A time interval being given, a staff member cannot work longer than the size of this time-interval (7). A staff member can perform a given skill only if he has it. (8)

$$\forall k \in [1..K],\ \forall l \in [1..L] \quad \sum_{i \in I} x_{i,l} \cdot b_{i,k} \leq \sum_{m=1}^{M} \delta^l_{m,k} \qquad (6)$$

$$\forall l \in [1..L],\ \forall m \in [1..M] \quad \sum_{k=1}^{K} \delta^l_{m,k} \leq t_{l+1} - t_l \qquad (7)$$

$$\forall l \in [1..L],\ \forall m \in [1..M],\ \forall k \in [1..K], \quad \delta^l_{m,k} \leq S_{m,k} \cdot (t_{l+1} - t_l) \qquad (8).$$

Proposition 1 : If the linear programming formulation given by (1)-(8) has no solution then LB+1 is a valid lower bound of the project duration.

Remember that deadlines are computed according to LB and the precedence graph. Then if there is no solution, there is at least one non-valid deadline, thus there is no solution with a makespan equal to LB.

2.2 A lower bound based on energetic reasoning

The energetic approach which has been formalized and evaluated both from a theoretical and an experimental point of view by Baptiste, Le Pape and Nuijten [1], aims at developing satisfiability tests and time-bound adjustments for Cumulative Scheduling Problem, to ensure that either a given schedule is not feasible or to derive some necessary conditions that any feasible schedule must satisfy. This approach has been successfully applied to other problems such as the Multi-Processor Flow-Shop [7].

Given a time interval $[t_1, t_2]$, satisfiability tests are based on the computation of the part of the activities that must be performed between time points t_1 and t_2. $W(i, t_1, t_2)$, the part of activity A_i that must be completed in time interval $[t_1, t_2]$ is called its work over the time-interval $[t_1, t_2]$. Once these mandatory parts are computed, determining if there exists a feasible person assignment to parts of activities over a given interval such that all skill requirements are met is equivalent to

solving a maximum flow problem on the graph $G(t_1, t_2)$ presented in Figure 2. Notice that the network structure does not vary from one interval to the other, only capacities on the arcs depend on the work of the activities over the considered time interval.

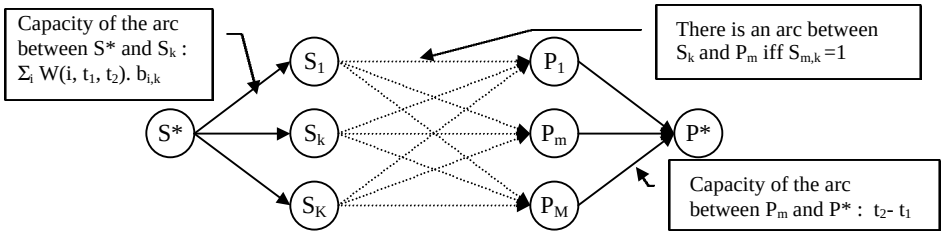

Figure 2 : $G(t_1, t_2)$ the graph for a given time interval $[t_1, t_2]$

Proposition 2 : If there exists a time interval $[t_1, t_2]$ such that the maximum flow computed on the graph $G(t_1, t_2)$ is not equal to $\Sigma_{k\in 1..K} \Sigma_{i\in I} W(i, t_1, t_2)$, then LB+1 is a valid lower bound.

Lets' remember that deadlines are computed according to LB and the precedence graph. Then if the maximum flow is not equal to $\Sigma_{k\in 1..K} \Sigma_{i\in I} W(i, t_1, t_2)$, there is at least one non-valid deadline, thus there is no solution with a makespan equal to LB.

4. Conclusion

In this paper we proposed two lower bounds for a new kind of resource constrained project scheduling problem. Multi-Skill Project Scheduling Problem can be seen as a specific case of Multi-Mode RCPSP in which the listing of valid modes is not possible within a reasonable time. Notice that these two lower bounds can easily be extended to take into account generalized precedence constraints and non-availability of persons. Experimental results on generated benchmarks will be presented to show the efficiency of our approach. Further works on this problem will focus on heuristics, based on an adaptation of the Serial Schedule Generation Scheme [6], and Tabu Search for the Multi-Skill Project Scheduling problem.

References

[1] Baptiste Ph., Le Pape C., Nuijten W. (1999) Satisfiability Tests and Time Bound Adjustments for Cumulative Scheduling Problems, *Annals of Operational Research*,3305-333.

[2] Carlier J. and Néron E, (2001), On linear lower bound for the resource constrained project scheduling problem. Submitted to *European Journal of Operational Research*.

[3] Dauzère-Pérès, S., Roux, W., and Lassère J.B., (1998), Multi-resource Shop Scheduling with Resource Flexibility, *European Journal of Operational Research*, 107;289-305.

[4] Jurish B., (1992), Scheduling Jobs in Shops with Multi-purpose Machines PhD dissertation, University of Osnabruck.

[5] Klein R., Scholl A., (1999), Computing Lower Bounds by Destructive Improvement: An Application to Resource-Constrained Project Scheduling, *European Journal of Operational Research*, 112;332-346.

[6] Kolisch R., (1996), Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation . *European Journal of Operational Research*, 90;320-222,.

[7] Néron E., Baptiste Ph. and Gupta J.N.D, (2001), Solving hybrid flow-shop using energetic reasoning, *Omega*, 29;501-511.

[8] Sprecher A. (1994), Resource-Constrained Project Scheduling, Exacts Methods for the Multi-Mode Case, *Lectures Notes in Economics and Mathematical Systems*, 409, Springer Verlag.