Reactive single-machine scheduling to maintain a maximum number of starting times

Philippe Chrétienne[1]

Université Pierre et Marie Curie, LIP6, F-75005 Paris, France
{philippe.chretienne}@lip6.fr

Mots-clés : *single-machine scheduling, complexity, dynamic programming*

1 Introduction

It is often the case that the real durations of an instance of a scheduling problem are not equal to those from which a baseline schedule has been planned. So, one has to determine a schedule of the real instance that minimizes a cost that measures the gap between the planned schedule and the schedule chosen for the real instance. A measure that has been already studied is the sum of the absolute deviations of the corresponding starting times [2]. However, in many cases, the cost is mainly impacted by the number of tasks that are not scheduled at the same date in the two schedules. The problem is then to find a schedule of the real instance such that the number of tasks scheduled at the same date in both schedules is maximum. It has been shown in [1] that this problem can be polynomially solved for CPM scheduling instances and when the real durations are longer that those of the planned instance.
In this paper, we consider the single-machine framework. We first study the so-called COMPATIBILITY problem where it must be decided if the tasks of a given subset of the real instance may be processed at their planned starting times. We show that COMPATIBILITY is NP-complete in the strong sense and that even the special case COMPATIBILITY(1), where a single task has to be processed at its planned starting time is NP-complete. Coming back to our optimization problem MAXANCHOR where we search for a compatible subset with a maximum cardinality, we show, by polynomially reducing COMPATIBILITY(1) to MAXANCHOR, that MAXANCHOR is NP-complete.
We then study the discrete-preemptive variants called respectively DPR-COMPATIBILITY and DPR-MAXANCHOR where the processing of a task may be splitted into more than one intervals with integer ends. We show that DPR-COMPATIBILITY may be solved in $O(n)$ time and we provide an $O(n^2)$ dynamic programming algorithm solving DPR-MAXANCHOR.

2 Complexity aspects

Assume that given an instance of n tasks $1, \cdots, n$, whose planned non-negative integer durations are respectively $p_1, \cdots, p_n$, a baseline schedule $(b_1, \cdots, b_n)$ has been chosen to process these tasks on a single machine. It is assumed that $b_i, i \in \{1, \cdots, n\}$ is a non negative integer. Some time after schedule b has been chosen, it occurs that the real duration of task i is no longer p_i but $p_i + \delta_i$ where δ_i is a non-negative integer. The COMPATIBILITY problem is to decide, given a subset $H = \{h_1, \cdots, h_K\}$ of $\{1, \cdots, n\}$ and a common deadline D, whether there is a schedule of the real instance such the tasks of H are scheduled at their planned starting time. Using a pseudopolynomial reduction from 3-PARTITION, it is easy to show that

Propriété 1 *COMPATIBILITY is NP-complete in the strong sense.*

The problem COMPATIBILITY(1) is the special case of COMPATIBILITY when H is a singleton. Again, using a polynomial reduction from PARTITION, it is easy to show that

Propriété 2 *COMPATIBILITY(1) is NP-complete.*

The problem MAXANCHOR is to decide, given an integer $K \leq n$, if there is a compatible subset of tasks whose cardinality is at least K. Using the following polynomial reduction of COMPATIBILITY(1) where :
- the instance of COMPATIBILITY(1) is denoted by (n, b, p, δ, M, h) (where it is assumed w.l.o.g. that $M < b_h + \sum_{i=1}^{n}(p_i + \delta_i)$) and
- the corresponding instance of MAXANCHOR is defined by a set of $n-1$ jobs A_i for $i \in \{1, \cdots, n\} \setminus \{h\}$ with $\hat{p}(A_i) = 0$, $\hat{\delta}(A_i) = p_i + \delta_i$, $\hat{b}(A_i) = 0$; one job A_h with $\hat{p}(A_h) = p_h$, $\hat{\delta}(A_h) = \delta_h$ and $\hat{b}(A_h) = b_h$; $\hat{D} = D$; $K = 2$; we get that

Propriété 3 *MAXANCHOR is NP-complete.*

3 Solving the discrete-preemptive versions

In the discrete-preemptive version DPR-COMPATIBILITY and DPR-MAXANCHOR, the first time unit during which every task i with $p_i > 0$ of a compatible subset must be run is the interval $[b_i, b_i + 1]$. For the sake of simplicity, we will assume in the rest of the paper that each task has a positive duration and that $b_1 < \cdots < b_n$.

3.1 A linear algorithm for DPR-COMPATIBILITY

Solving DPR-COMPATIBILITY is quite easy. Assume that $H = \{h_1, \cdots, h_K\}$ and that $h_1 < \cdots < h_K$. The first step assigns each unit-time interval $[b_i, b_i + 1]$ for $i \in H$ to the task i. In the second step, starting from time b_1, each non-busy time unit is assigned to the first ready compatible task if there is one. In the third step, starting from time 0, each remaining non-busy time unit is assigned to a non already completed task of $\{1, \cdots, n\} \setminus H$ while there is one. It is easy to prove that this algorithm, which may be implemented in linear time, finds a schedule such that each task h_k starts at time b_{h_k}. Morever this schedule has a minimum makespan.

3.2 An $O(n^2)$ algorithm for DPR-MAXANCHOR

In order to solve DPR-MAXANCHOR by a dynamic programming scheme, we define for $i \in \{1, \cdots, n\}$ and $k \in \{0, \cdots, i\}$ the function $m(i, k)$ where $m(i, k)$ is the minimum makespan of a schedule of the tasks subset $\{1, \cdots, i\}$ containing a compatible subset of size k. It is easy to see that $m(i, 0) = \sum_{j=1}^{i} p_j + \delta_j$ and that $m(i, i) = \mu_i$ where the μ_i's are given by $\mu_0 = 0$ and the recurrence formula for $j \in \{1, \cdots, i\}$: $\mu_j = \max(b_j, \mu_{j-1}) + p_j + \delta_j$. Considering the two distinct cases when task i belongs to an optimal compatible subset or not, we proved the following recurrence formula :

$$m(i, k) = \min(m(i-1, k), p_i + \delta_i + \max(b_i, m(i-1, k-1))).$$

The maximum size of a compatible subset is then given by $\max(k \in \{1, \cdots, n\} | m(n, k) \leq D)$.

4 Conclusion

The problematic studied in this paper does not only concern scheduling problems but all combinatorial problems where a solution of a planned instance must be chosen before the real instance is actually known. This is far from being rare in practice and gives rise to a lot of promising and quite interesting research directions.

Références

[1] Pascale Bendotti, Philippe Chrétienne, Pierre Fouilhoux and Alain Quilliot. *PERT : Recherche d'ordonnancements stables en présence de durées incertaines.* ROADEF 2016.

[2] Herroelen, W and Leus, R. *The construction of stable project baseline schedules.* Technical Report Kathilieke Universiteit Leuven, 2004.