Anchored reactive and proactive solutions to the CPM-scheduling problem

Pascale Bendotti, Philippe Chrétienne, Pierre Fouilhoux, Alain Quilliot

PII: S0377-2217(17)30112-1
DOI: 10.1016/j.ejor.2017.02.007
Reference: EOR 14243

To appear in: *European Journal of Operational Research*

Received date: 19 May 2016
Revised date: 21 December 2016
Accepted date: 8 February 2017

Please cite this article as: Pascale Bendotti, Philippe Chrétienne, Pierre Fouilhoux, Alain Quilliot, Anchored reactive and proactive solutions to the CPM-scheduling problem, *European Journal of Operational Research* (2017), doi: 10.1016/j.ejor.2017.02.007

Highlights

- Precedence constrained scheduling under uncertain durations,
- Combinatorial criterion for robust optimization,
- Anchored solutions for proactive precedence constrained scheduling,
- Complexity and polynomial special cases.

Anchored reactive and proactive solutions to the CPM-scheduling problem

Pascale Bendotti[a,b], Philippe Chrétienne[b,*], Pierre Fouilhoux[b], Alain Quilliot[c]

[a] *EDF R&D, Département OSIRIS, 1 avenue du Général de Gaulle, 92141 Clamart cedex, France*
[b] *Sorbonne Universités, Université Pierre et Marie Curie, LIP6 CNRS UMR 7606, 4 place Jussieu 75005 Paris, France*
[c] *Université Blaise Pascal, LIMOS CNRS UMR 6158, F-63173, Aubie, France*

Abstract

In a combinatorial optimization problem under uncertainty, it is never the case that the real instance is exactly the baseline instance that has been solved earlier. The anchorage level is the number of individual decisions with the same value in the solutions of the baseline and the real instances. We consider the case of CPM-scheduling with simple precedence constraints when the job durations of the real instance may be different than those of the baseline instance. We show that, given a solution of the baseline instance, computing a reactive solution of the real instance with a maximum anchorage level is a polynomial problem. This maximum level is called the anchorage strength of the baseline solution with respect to the real instance. We also prove that this latter problem becomes NP-hard when the real schedule must satisfy time windows constraints. We finally consider the problem of finding a proactive solution of the baseline instance whose guaranteed anchorage strength is maximum with respect to a subset of real instances. When each real duration belongs to a known uncertainty interval, we show that such a proactive solution (possibly with a deadline constraint) can be polynomially computed.

*Corresponding author. Correspondence to: Université Pierre et Marie Curie, Laboratoire LIP6, Boîte courrier 169, 4 Place Jussieu, 75252 Paris cedex 05, France Tel: +33 1 44 27 54 42; fax: +33 1 44 27 70 00

Email addresses: pascale.bendotti@edf.fr (Pascale Bendotti), philippe.chretienne@lip6.fr (Philippe Chrétienne), pierre.fouilhoux@lip6.fr (Pierre Fouilhoux), alain.quilliot@isima.fr (Alain Quilliot)

Keywords:

1. Introduction

In the real world it is never the case that the instance of a problem to be solved is exactly known. How to take uncertainty into account in a combinatorial problem induces more and more research works.

Let us consider a combinatorial optimization problem Π and an instance I of Π. A solution $x = (x_1, \ldots, x_n)$ of the solution set $\mathcal{S}_I$ of I is a vector of n individual decisions. Problem Π is then to find an optimal solution $x \in \mathcal{S}_I$ minimizing a cost function c_I.

Given a *baseline instance* I of Π, we assume that a (not necessarily optimal) *baseline solution* $x \in \mathcal{S}_I$ has been chosen in advance. However, due to unexpected changes in the instance parameters, the *real instance* to be solved is no longer I but $\widehat{I}$. In situations where modifying any planned decision value x_i has a significant cost impact, the objective is now to find a solution $y \in \mathcal{S}_{\widehat{I}}$ such that the number of equally-valued decisions in x and y is maximum. Given a solution $x \in \mathcal{S}_I$ and a solution $y \in \mathcal{S}_{\widehat{I}}$, we define the *anchorage level* $\sigma(x, y)$ as the number of equally-valued decisions in x and y. Given a solution $x \in \mathcal{S}_I$ and an instance $\widehat{I}$, the *anchorage strength* $S(x, \widehat{I})$ of x with respect to instance $\widehat{I}$ is then $S(x, \widehat{I}) = \max_{y \in \mathcal{S}_{\widehat{I}}} \sigma(x, y)$. Note that, contrary to the well-known stability criterion (Herroelen and Leus, 2004) that measures the absolute deviation between the solutions of the baseline and the real instances, the anchorage level is a discrete measure based on the combinatorial structure of the problem.

A first problem occurring in practice is when the real instance $\widehat{I}$ is revealed some time after the baseline solution $x \in \mathcal{S}_I$ has been chosen: the *Anchor-Reactive* problem is then to determine a solution $y \in \mathcal{S}_{\widehat{I}}$ with a maximum anchorage level $\sigma(x, y)$.

A second problem occurring in practice is, when the real instance $\widehat{I}$ is not known, to decide which solution x of the baseline instance I to choose. However, it is realistic to assume that the subset R_Π of the possible real instances of Π is

known. In that context, the value $S^*(x, R_\Pi) = \min_{\widehat{I} \in R_\Pi} S(x, \widehat{I})$ is the *guaranteed anchorage strength* of the baseline solution x. The *Anchor-Proactive* problem is then to find the solution that provides a maximum guaranteed anchorage strength, thus defining the *anchored proactive solution*.

Note that weighted variants of these problems can easily be derived where weights are associated with the decisions.

More specifically we focus on the CPM-scheduling problem. In a baseline instance I of the CPM-scheduling problem, n jobs $J_1, \ldots, J_n$ have to be scheduled so that a set of *precedence constraints* between these jobs are satisfied. Each job J_i has a positive *expected duration* p_i and must be processed without any interruption. A *baseline schedule* x associates with each job J_i a non negative starting time x_i such that for every precedence constraint (J_i, J_j, p_i), $x_j \geq x_i + p_i$. The makespan of x is the value $\max_{i \in \{1,\ldots,n\}}\{x_i + p_i\}$. Finding a baseline schedule with a minimum makespan is a classical polynomial problem. However, the *real instance* $\widehat{I}$ is defined by the same set of jobs and the same precedence constraints as I but in the real instance $\widehat{I}$ the duration of jobs J_i is $p_i + \delta_i$ where the *disruption* δ_i is such that $p_i + \delta_i$ is non negative. In this context, the anchorage level between x and y is then the number of jobs with the same starting times in both schedules.

The first problem is called the *Anchor-Reactive CPM-Scheduling Problem* (ARSP) and is to find a schedule y of maximum anchorage level with respect to x, *i.e.,* sharing with x a maximum number of equally-valued starting times. The second problem is called the *Anchor-Proactive CPM-Scheduling Problem* (APSP) and is to find a schedule x of maximal guaranteed anchorage strength over the real instances corresponding to a known uncertainty disruption domain Δ.

These two problems arise in real-life projects when unexpected events occur, thus introducing variability in the job duration. Whenever the schedule x is not feasible for the real instance, a trivial approach is to find a schedule of the real instance $\widehat{I}$ with a minimum makespan. In that case, it could occur that all jobs are rescheduled. However it could be just as crucial to schedule a job

at its planned starting time as it is to terminate the project as planned. The need for maintaining as many starting times of x as possible could arise in a multi-project environment to make advance booking for resources to guarantee their availability. Other sources could be hard delivery dates for suppliers or subcontractors. The authors in (Herroelen *et al*, 2002) point out that rescheduling in such a context requires extreme care as some dates cannot be revised. An example arises in the context of scheduling maintenance jobs during an outage of a production unit. A well known objective is minimizing the outage duration. The associated economic value at stake is the cost of a substitute production. The substitution cost of a nuclear power plant is worth over 20,000 euros per day while an outage last several weeks and a schedule x involves over 10,000 jobs. While the outage duration is a major concern, the starting times for each job plays also an important role from an economic point of view. Actually each starting time is used to plan the corresponding specific resources required or the availability of subcontractors. It is thus of high importance to set the starting times for the corresponding operations well in advance and to execute each jobs at its planned starting time. In this context a relevant characteristic for the schedule of the real instance is its ability to accommodate disruptions by time-shifting as few jobs as possible with respect to the schedule of the baseline instance.

In Section 2 a short review of optimization problems under uncertainty is presented along with some background on CPM-scheduling and some useful notations. In Section 3 a polynomial algorithm for the ARSP is presented. In Section 4 an ARSP variant including time windows is considered and proved to be NP-complete. In Section 5 the disruption δ is assumed to be unknown and to belong to a known uncertainty domain Δ. In the case when Δ is a set of known intervals, we prove that finding a deadline constrained proactive schedule with maximum guaranteed anchorage strength over Δ is still a polynomial problem. The conclusion gives some research directions relative to other problems in Operational Research that could benefit from an anchored variant and fit in the proposed framework.

2. State of the art

In this section we propose a brief review of the research activity dedicated to the so-called *robust optimization* approach which refers to optimizing under uncertainty. In this context, we will show how the new paradigm of anchorage level compares to the stability measure sometimes used in robust optimization. Since we focus on CPM-scheduling problems under uncertainty, we also propose a short review of the corresponding works, and finally some background on CPM-scheduling.

2.1. Optimization under uncertainty

In the general sense the aim of robust optimization is to find a feasible solution to a problem while the uncertainties take values within a given set. A standard approach for the resolution of a robust problem involves creating a deterministic equivalent, called the robust counterpart. The practical difficulty then depends on the tractability of computing the robust counterpart (Ben-Tal *et al*, 2009).

Soyster (1973) considered a simple uncertainty set where a robust solution is feasible for all possible values in the uncertainty set, this problem defining the *robust worst-case*. Other authors extended this framework of robust counterpart optimization (Ben-Tal and Nemirovski, 1999; El-Ghaoui *et al*, 1998; Bertsimas and Sim, 2003).

While the aim is to come up with a baseline solution before uncertainty occurs, most robust techniques try to find solutions flexible to changes to the data in a multi-stage process. At a first stage "here and now" decisions are determined, while "wait and see" decisions are determined as recourse to the first stage decisions at later stages when uncertainty occurs. Note that the global cost in a robust framework takes into account both the cost of the baseline solution and the subsequent costs induced by the further stages. Robust solutions often induce a high "here and now" cost in the hope to get low "wait and see" costs. Several articles dealing with one-stage or multi-stage robust optimization

propose methods to find robust solutions with a stabilizing feature (Moazeni, 2013). Typically a deviation from the first stage decisions is considered either in some additional constraints or in the objective with a penalty factor. In this article, we propose another criterion relying on the combinatorial structure of the underlying optimization problem. Indeed anchored proactive solutions are expected to induce a low "here and now" cost with bounded "wait and see" costs.

2.2. Project scheduling under uncertainty

Even though a large majority of project scheduling efforts has been devoted to solve scheduling problems in a deterministic setting, a research track emerged in the literature to consider project scheduling under uncertainty (see (Demmeulemeester and Herroelen, 2002) for an extensive review of project scheduling under uncertainty). A first approach called *proactive baseline scheduling* is to find a schedule that would account for anticipated disruptions, thus incorporating safety in the baseline schedule. Note that such a schedule is sometimes called stable pre-schedule (Herroelen and Leus, 2004). A second approach re-optimizes the baseline schedule when unexpected events occur (see Herroelen and Leus (2004) for a review on proactive and reactive scheduling techniques).

A classical way to prevent any job to be time-shifted in a CPM-scheduling problem is to find a *robust schedule* that is guaranteed to be feasible for all anticipated disruptions that lie within a given uncertainty set. Robust scheduling reduces to find a proactive baseline schedule before uncertainty occurs. A typical extension is to find a schedule flexible to changes to the input data. A robust schedule often induces a large makespan. For CPM-scheduling projects, Minoux (2008) proposed a 2-stage robust LP model to specify the uncertainty set for job durations and showed that the resulting robust optimization problem can be efficiently solved in polynomial time. Such a model provides a way to control the degree of conservatism through a budget parameter, which bears some similarities with that first introduced by Bertsimas and Sim (2003). However it does not provide a way to control its anchorage level, which is our main concern in this article.

2.3. CPM-scheduling

Let us consider a baseline instance of the CPM-scheduling problem with n jobs $J_1, \ldots, J_n$ whose durations are $p_1, \ldots, p_n$ and a set of generalized precedence constraints (J_i, J_j, a_{ij}), where $a_{ij} \in \mathbb{R}$. Such an instance can also be defined by an arc-valued graph denoted by $G(a)$ whose nodes are the jobs $J_1, \ldots, J_n$ and whose valued arcs are the precedence constraints (J_i, J_j, a_{ij}). The following well-known Property 2.1 gives a necessary and sufficient condition for at least a schedule to exist (see (Pinedo, 2002)).

Property 2.1. *The set of schedules of $G(a)$ is not empty if and only if $G(a)$ has no positive circuit.*

In the case of simple precedence constraints where $a_{ij} = p_i$, the graph $G(a)$ can be simply denoted by $G(p)$ and has no circuit. We assume without loss of generality that if (J_i, J_j) is an arc, we have $i < j$. Whenever it is clear from the context, we will use node i for job J_i. Note that, in this case, Property 2.1 says that an instance has a schedule if the graph $G(p)$ has no circuit.

It is usual to add a *fictive* job J_0 (the project starting event) with null duration, and the additional constraints $(J_0, J_i, 0)$ for $i \in \{1, \ldots, n\}$, and a fictive job J_{n+1} (the project ending event) with null duration and the additional constraints (J_i, J_{n+1}, p_i) for $i \in \{1, \ldots, n\}$.

Consider an illustrative example with five jobs along with their expected durations $p = (4, 6, 3, 7, 1)$. Its corresponding valued graph $G(p)$ is shown in Figure 1. Note that only the necessary arcs incident to nodes J_0 and J_6 are represented.

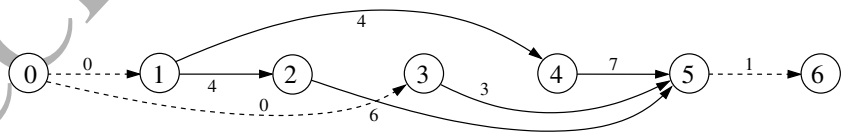

Figure 1: The valued graph $G(p)$ of an instance with five jobs

In the rest of the article, the following notations will be widely used. Let $G(a)$ be a valued graph where every arc (i, j) of $G(a)$ is valued by a_{ij}.

The *length* $\sum_{(i,j)\in P} a_{ij}$ of a path P of $G(a)$ is denoted by $l_{G(a)}(P)$. Moreover, if $G(a)$ has no positive circuit, the *length of a longest path* from node i to node j is denoted by $L_{G(a)}(i,j)$ with the convention that $L_{G(a)}(i,j) = -\infty$ if there is no such path.

The *complementary graph* of $G(a)$ is the graph $\overline{G}(a)$ with the same node set as $G(a)$ and whose arcs are all pairs of distinct nodes that are not an arc in $G(a)$.

A *poset* (or partially ordered set) on $\{1,\ldots,n\}$ is a relation $\mathcal{R}$ which is reflexive, anti-symmetric and transitive. A subset $A \subset \{1,\ldots,n\}$ is an *antichain* of the poset if $(e,e') \notin \mathcal{R}$ and $(e',e) \notin \mathcal{R}$ for all $e \neq e' \in A$.

3. The Anchor-Reactive CPM-Scheduling Problem

In this section, we consider the reactive case where the disruption δ is known and we show that the ARSP can be solved in polynomial time.

Given an instance $(G(p), p, x, \delta)$ of the ARSP, let us consider a subset H of $\{J_1,\ldots,J_n\}$. A subset H of $\{J_1,\ldots,J_n\}$ is said to be *x-compatible* if there is a schedule y of $G(p+\delta)$ such that the jobs of H have the same starting times in x and y.

We introduce a technical valued graph $G^H(p+\delta)$ defined from $G(p+\delta)$ as follows: for each $J_i \in H$, the constraint $(J_0, J_i, 0)$ is replaced by (J_0, J_i, x_i) (the arc (J_0, J_i) is called a *forward arc*) and a new constraint $(J_i, J_0, -x_i)$ is created (the arc (J_i, J_0) is called a *backward arc*).

For example, by referring to the instance shown in Figure 1, Figure 2 shows the valued graph $G^H(p+\delta)$ relative to disruption $\delta = (2,1,0,-1,1)$, $x = (0,5,1,6,13)$ and $H = \{J_1, J_3, J_4, J_5\}$. Note that only the necessary arcs incident to node J_6 are represented. The minimum makespan of the baseline instance is 12 while that of the real instance is 15. Since the makespan of baseline schedule x is 14, at least one job must be with a different starting time in a real instance. As the considered subset H contains four jobs, it corresponds to a schedule of minimum anchorage level with respect to x.

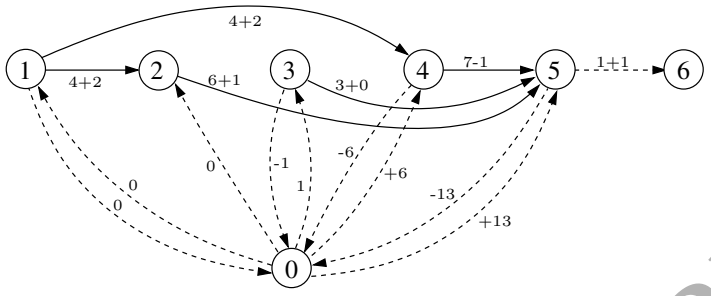

Figure 2: A valued graph $G^H(p+\delta)$

Note that graph $G^H(p+\delta)$ corresponds to a particular CPM-scheduling instance where the starting times of jobs in H are equal in every schedule y as they keep their values from x: indeed, the two constraints $y_j \geq y_0 + x_i$ and $y_0 \geq y_i - x_i$ imply $y_j = y_0 + x_i$. The following technical lemma directly comes from Property 2.1.

Lemma 3.1. *A subset H of $\{J_1, \ldots, J_n\}$ is x-compatible if and only if $G^H(p+\delta)$ has no positive simple circuit.*

A more precise analysis of the circuits of $G^H(p+\delta)$ leads to the following equivalent condition.

Property 3.2. *A subset H of $\{J_1, \ldots, J_n\}$ is x-compatible if and only if, for any $J_i \in H \cup \{J_0\}$ and $J_j \in H \setminus \{J_i\}$ such that there is a path from J_i to J_j in $G(p)$, we have $x_i + L_{G(p+\delta)}(i,j) \leq x_j$.*

Proof. Let us first show the $\Leftarrow$ part.
First remark that, since the only arc (J_i, J_j) of $G^H(p+\delta)$ with $j < i$ are backward arcs (J_i, J_0), then J_0 belongs to every simple circuit of $G^H(p+\delta)$. Let C be an arbitrary simple circuit of $G^H(p+\delta)$ and let (J_0, J_i) and (J_j, J_0) be the arcs of C incident to J_0. We then consider two cases:
- If (J_0, J_i) is not a forward arc, then C is made of the arc (J_0, J_i) valued by 0, followed by a path P of $G^H(p+\delta)$ from J_i to J_j, followed by the backward arc (J_j, J_0) valued by $-x_j$. Since $J_j \in H$, we have from the assumption that

$L_{G(p+\delta)}(0,j) - x_j \leq 0$. Moreover since P is also a path of $G(p+\delta)$, we get

$$l_{G(p+\delta)}(C) = 0 + l_{G(p+\delta)}(P) - x_j \leq L_{G(p+\delta)}(i,j) - x_j \leq 0.$$

- If (J_0, J_i) is a forward arc, then C is made of the arc (J_0, J_i) valued by x_i, followed by a path P of $G^H(p+\delta)$ from J_i to J_j, followed by the backward arc (J_j, J_0) valued by $-x_j$. Since $J_i \in H$ and $J_j \in H$, we have from the assumption $L_{G(p+\delta)}(i,j) - x_j + x_i \leq 0$ and we get :

$$l_{G(p+\delta)}(C) = x_i + l_{G(p+\delta)}(P) - x_j \leq x_i + L_{G(p+\delta)}(i,j) - x_j \leq 0.$$

In both cases, we can see that cycle C is not positive, then, from Lemma 3.1, H is x-compatible.

Let us now show the $\Rightarrow$ part.

Let $J_i \in H \cup \{J_0\}$ and $J_j \in H \setminus \{J_i\}$ such that there is a path from J_i to J_j in $G(p)$. Let P be the longest path from J_i to J_j in $G^H(p+\delta)$. Let P' be a longest path from J_i to J_j in $G_{p+\delta}$, since P' is a path of $G^H(p+\delta)$ whose arcs have values lower than or equal to arcs of $G^H(p+\delta)$, then $L_{G(p+\delta)}(i,j) \leq L_{G^H(p+\delta)}(i,j)$.

If $i = 0$, then we have $x_i = 0$. Let C be the simple cycle $(P, (J_j, J_0))$ of $G^H(p+\delta)$, we know from Lemma 3.1 that this cycle is not positive, thus $L_{G^H(p+\delta)}(0,j) - x_j \leq 0$. Since $L_{G(p+\delta)}(0,j) \leq L_{G^H(p+\delta)}(0,j)$, we get $x_i + L_{G(p+\delta)}(0,j) - x_j \leq 0$.

If $i \in \{1, \ldots, n\}$, let C be the simple circuit $((J_0, J_i), P, (J_j, J_0))$. We have $l_{G^H(p+\delta)}(C) = l_{G^H(p+\delta)}(J_0, J_i) + L_{G^H(p+\delta)}(i,j) - x_j$. We know from Lemma 3.1 that $l_{G^H(p+\delta)}(C) \leq 0$, and then $l_{G^H(p+\delta)}(J_0, J_i) + L_{G(p+\delta)}(i,j) - x_j$. Since (J_0, J_i) is a forward arc, we finally get $x_i + L_{G(p+\delta)}(i,j) - x_j \leq 0$. □

For example, by referring to the valued graph shown in Figure 2, jobs J_1 and J_3 have no predecessors (besides the fictive jobs), so their starting times are not subject to disruption effects. Job J_2 is scheduled at 5 in the baseline schedule x, while its predecessor J_1 takes $p_1 + d_1 = 6$ units of time to process. Note that if J_2 were included in H, then circuit (J_0, J_1, J_2, J_0) would pass through the newly added backward arc (J_2, J_0) valued by -5, thus would be with positive value 1. Job J_4 is scheduled at $x_4 = 6$ and its predecessor J_1 terminates at 6. Hence the

new added constraints (J_0, J_4, x_4) and $(J_4, J_0, -x_4)$ do not induce any positive valued circuit in $G^H(p+\delta)$. Similarly for J_5 scheduled at $x_5 = 13$. Note that job J_4 takes one less unit of time to process since $\delta_4 = -1$. Hence it does not affect the sign of any circuit as long as its real duration $p_4 + \delta_4$ remains non negative. Since graph $G^H(p+\delta)$ features only no positive circuits, H is x-compatible.

From Property 3.2 we will finally show that a subset H of $\{J_1, \ldots, J_n\}$ is x-compatible if and only if $H \cup \{J_0\}$ is an antichain of a specific poset of $\{J_0, \ldots, J_n\}$.

Let $\mathcal{R}_{G(p),x,\delta}$ be the relation on $\{J_0, \ldots, J_n\}$ defined as follows: $(J_i, J_j) \in \mathcal{R}_{G(p),x,\delta}$ if $(i = j)$ or if there is a simple path from J_i to J_j in $G(p)$ such that $x_j < x_i + L_{G(p+\delta)}(i, j)$.

$\mathcal{R}_{G(p),x,\delta}$ is clearly reflexive. $\mathcal{R}_{G(p),x,\delta}$ is anti-symmetric since $G(p)$ has no circuit. Let us show that $\mathcal{R}_{G(p),x,\delta}$ is also transitive. Assume that $(J_i, J_j) \in \mathcal{R}_{G(p),x,\delta}$ and $(J_j, J_k) \in \mathcal{R}_{G(p),x,\delta}$. If $i = j = k$, then we clearly have $(J_i, J_k) \in \mathcal{R}_{G(p),x,\delta}$. If $i = j$ and $j \neq k$, there is a simple path from J_i to J_k in $G(p)$ such that $x_k < x_i + L_{G(p+\delta)}(i, k)$. The same reasoning applies if $i \neq j$ and $j = k$. Finally, if i, j and k are all distinct, then, since $G(p)$ has no circuit, there is a simple path from J_i to J_k in $G(p)$. Moreover, we have $x_j < x_i + L_{G(p+\delta)}(i, j)$ and $x_k < x_j + L_{G(p+\delta)}(j, k)$ so that $x_k < x_i + L_{G(p+\delta)}(i, j) + L_{G(p+\delta)}(j, k) \leq L_{G(p+\delta)}(i, k)$. We then have proved that $\mathcal{R}_{G(p),x,\delta}$ is a poset on $\{J_0, \ldots, J_n\}$.

We then can state the main result of this section.

Theorem 3.3. *A subset H of $\{J_1, \ldots, J_n\}$ is x-compatible if and only if $H \cup \{J_0\}$ is an antichain of the poset $\mathcal{R}_{G(p),x,\delta}$ of $\{J_0, \ldots, J_n\}$.*

Proof. Assume that $H \cup \{J_0\}$ is an antichain of the poset $\mathcal{R}_{G(p),x,\delta}$ of $\{J_0, \ldots, J_n\}$ and let J_i and J_j be two jobs such that $i \neq j$, $J_i \in H \cup \{J_0\}$ and $J_j \in H$. If there is a path from J_i to J_j in $G(p)$, then from the definition of $\mathcal{R}_{G(p),x,\delta}$, we necessarily have $x_i + L_{G(p+\delta)}(i, j) \leq x_j$. So, we conclude from Property 3.2 that H is x-compatible.

Assume now that H is x-compatible and let J_i and J_j be two distinct jobs in $H \cup \{J_0\}$. If $(J_i, J_j) \in \mathcal{R}_{G(p),x,\delta}$, then there is a simple path J_i to J_j in $G(p)$ and $x_j < x_i + L_{G(p+\delta)}(i,j)$. So, from Property 3.2, H is not x-compatible, a contradiction. The same reasoning applies if $(J_j, J_i) \in \mathcal{R}_{G(p),x,\delta}$. □

From Theorem 3.3, we know that an x-compatible subset of maximum cardinality in $\{J_1, \ldots, J_n\}$ can be found by searching for a maximum antichain containing J_0 in the poset $\mathcal{R}_{G(p),x,\delta}$. Since $\mathcal{R}_{G(p),x,\delta}$ may be computed in $O(n^3)$, it comes from Dilworth's theorem in (Dilworth, 1950), that finding a maximum x-compatible subset H_{max} can also be done in $O(n^3)$. By definition the size of H_{max} is the anchorage level of a schedule of $G(p+\delta)$ with respect to schedule x. Finally, such a schedule can be obtained with a linear time breadth first search on graph $G^{H_{max}}(p+\delta)$, thus leading to the following theorem.

Theorem 3.4. *The ARSP can be solved in $O(n^3)$.*

4. The Anchor-Reactive CPM-Scheduling Problem with Time Windows

We now consider an ARSP variant including time windows, called the *ARSP with Time Windows* (ARSPTW), where the jobs must satisfy a global time-window constraint such that in the new schedule y each job must be processed within the time window $[A, B]$. Here we will simply consider the special case where $A = 0$.

An instance of the ARSPTW is thus denoted by $(G(p), p, \delta, x, B)$ where $G(p)$, p, δ and x are as defined in an instance of the ARSP, and where each job in the new schedule y must be completed before B, *i.e.*, y is a schedule of $G(p+\delta)$ such that $y_{n+1} \leq B$. Without loss of generality we assume that every job in schedule x is also processed before B (otherwise all jobs have to be rescheduled).

To set the complexity of the ARSPTW, we will use a reduction from the Maximum Complete Bipartite Subgraph (MCBS). This problem is to decide whether a bipartite graph $G = (L \cup R, E)$ with n (non isolated) nodes has

a complete bipartite subgraph with at least k nodes. The MCBS is an NP-complete problem (Garey and Johnson, 1979).

We will consider a special case of the ARSP where

- $G = (L \cup R, E)$ is a bipartite graph,
- $p_i = 1$ for $i \in L \cup R$,
- $\delta_i = 1$ (resp. 0) for $i \in L$ (resp. R),
- $B = 4$,
- $x_i \in \{1, 2\}$ for $i \in L \cup R$.

Let us denote by $\overline{\Pi}$ the decision version of this special case of the ARSPTW where it must be decided whether an instance $(G(p), p, \delta, x, B)$ has a schedule y with an anchorage level at least k with respect to schedule x. Thus an instance of $\overline{\Pi}$ will be denoted by $(G(p), p, \delta, x, B, k)$.

Consider now an instance $(G(p), p, \delta, x, B, k)$ of $\overline{\Pi}$ and suppose that this instance admits a schedule y. We can first observe that every isolated node i of $L \cup R$ may be scheduled such that $y_i = x_i$. So, we will assume without loss of generality that G has no isolated node. Consequently we necessarily have $y_i \in \{0, 1\}$ for all $i \in L$ and $y_i \in \{2, 3\}$ for all $i \in R$. We then denote by L_0 (resp. L_1) the subset of the jobs of L scheduled at time 0 (resp. 1) in y. Let us also denote by R_2 (resp. R_3) the subset of the jobs of R scheduled at time 2 (resp. 3) in y. Figure 3 illustrates this definition for an instance with nine jobs where subsets $L_1 = \{4, 5\}$ and $R_2 = \{7, 8\}$.

The following Lemma gives the key idea of this section.

Lemma 4.1. *Given a schedule y of $\overline{\Pi}$ and the corresponding subsets L_0, L_1, R_2 and R_3 describing y, then $L_1 \cup R_2$ induces a complete bipartite subgraph of the complementary graph $\overline{G}$ of G. Moreover the anchorage level is $\sigma(x, y) = |L_1| + |R_2|$.*

Proof. Let us consider a schedule y of $\overline{\Pi}$ and the corresponding subsets L_0, L_1, R_2 and R_3. Clearly for any $i \in L_1$ and any $j \in R_2$, we do not have $(i, j) \in E$. If $\overline{G}$ is the complementary graph of G, the subgraph of $\overline{G}$ induced by $L_1 \cup R_2$ is a

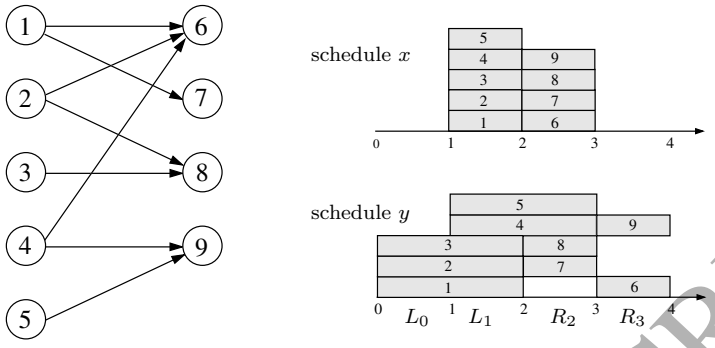

Figure 3: Schedules x and y of a $\overline{\Pi}$ instance

complete bipartite subgraph. By construction, jobs of $L_1 \cup R_2$ are the only jobs sharing the same starting times in both x and y, thus $\sigma(x, y) = |L_1| + |R_2|$. □

Theorem 4.2. *$\overline{\Pi}$ is an NP-complete problem.*

Proof. Let (G, k) be an MCBS instance where $G = (L \cup R, E)$ is a bipartite graph with n (non isolated) nodes and k is a positive integer. We define the following instance $f(G) = (\overline{G}(p), p, \delta, x, B, k)$ of $\overline{\Pi}$ as follows:

- $\overline{G}(p)$ is the complementary graph of G,
- $p_i = 1$ for all $i \in L \cup R$,
- $\delta_i = 1$ (resp. 0) for all $i \in L$ (resp. $i \in R$),
- $x_i = 1$ (resp. $x_i = 2$) for all $i \in L$ (resp. $i \in R$),
- $B = 4$.

Let us first suppose that y is a schedule of $\overline{G}(p+\delta)$ with anchorage level $\sigma(x, y) = k$, we then consider the corresponding subsets L_1 and R_2. By Lemma 4.1 the subgraph of $\overline{G}(p + \delta)$ induced by $L_1 \cup R_2$ is complete and has k nodes.
Conversely, let us now suppose that instance G of the MCBS admits a complete bipartite subgraph induced by $(L' \cup R')$ whose number of nodes $|L'| + |R'| \geq k$. We then define a schedule y of $\overline{G}(p + \delta)$ described by subsets $L_0 = L \setminus L'$, $L_1 = L'$, $R_2 = R'$ and $R_3 = R \setminus R'$. From Lemma 4.1,we then have $\sigma(x, y) \geq k$. Finally, since $f(G)$ may be computed from G in polynomial time, the MCBS polynomially reduces to $\overline{\Pi}$ and $\overline{\Pi}$ is then NP-complete. □

Since the length of the special case $\overline{\Pi}$ of the ARSPTW is polynomially equivalent to the length of the ARSPTW and Π is NP-complete, then we have the following theorem.

Theorem 4.3. *The ARSP with Time Windows is NP-hard.*

5. The Anchor-Proactive CPM-Scheduling Problem

In the previous sections, the ARSP(TW) finds a reactive schedule when the disruption δ is known. In this section, the aim is to compute a schedule of the baseline instance before any disruption is known. The disruption δ is now supposed to be unknown but belongs to an uncertainty domain Δ. Each real instance $\widehat{I}$ (resp. real instance set R_{APSP}) is now defined by a disruption δ (resp. uncertainty domain Δ).

Then a decision maker faces two important questions: which guarantee of anchorage strength is offered by a given baseline schedule x? and which baseline schedule x offers a minimum anchorage strength guarantee that is maximum?

Given a schedule x of a baseline instance and a disruption $\delta \in \Delta$, the anchorage strength of a given schedule x of $G(p)$ then writes $S(x,\delta) = \max_{y \text{ schedule of } G(p+\delta)} \sigma(x,y)$ and its guaranteed anchorage strength is $S^*(x,\Delta) = \min_{\delta\in\Delta} S(x,\delta)$, thus answering the first decision maker's question.

As to answer the second question, the problem would be to find an *anchored proactive schedule* x^*_Δ satisfying $S^*(x^*_\Delta,\Delta) \geq S^*(x,\Delta)$ for all potential schedule x, *i.e.*,

$$S^*(x^*_\Delta,\Delta) = \max_{\substack{x \text{ schedule} \\ \text{of } G(p)}} \min_{\delta\in\Delta} \max_{\substack{y \text{ schedule} \\ \text{of } G(p+\delta)}} \sigma(x,y)$$

The complexity of the problems associated to both questions depends of the combinatorial structure of Δ. In the following Subsection 5.1, a special case for the uncertainty structure is considered, thus leading us to consider the so-called robust worst-case in the uncertainty domain.

5.1. The Anchor-proactive CPM-scheduling problem in the robust worst-case

In this section we consider the special case when $\Delta = \prod_{i\in\{1,\dots,n\}}[\delta_i^-, \delta_i^+]$ where $[\delta_i^-, \delta_i^+]$ is a known interval of non-negative real values.

Let x be a given schedule of $G(p)$. To compute $S^*(x, \Delta)$, we prove the following monotonicity property of $S(x, \delta)$ as a function of δ.

Property 5.1. *If $\delta' \geq \delta$, then $S(x, \delta') \leq S(x, \delta)$.*

Proof. From Theorem 3.3, we know that $S(x, \delta)$ is equal to the number of jobs of a maximum antichain containing 0 of the poset $\mathcal{R}_{G(p),x,\delta}$ on $\{0, 1, \dots, n\}$. Let us show that $\mathcal{R}_{G(p),x,\delta} \subseteq \mathcal{R}_{G(p),x,\delta'}$. If $i \neq j$ and $(J_i, J_j) \in \mathcal{R}_{G(p),x,\delta}$, then there is a path from J_i to J_j in $G(p)$ and $x_j < x_i + L_{G(p+\delta)}(i, j)$. Since $\delta \leq \delta'$, we also have $L_{G(p+\delta)}(i, j) \leq L_{G(p+\delta')}(i, j)$ and $x_j < x_i + L_{G(p+\delta')}(i, j)$. So $(J_i, J_j) \in \mathcal{R}_{G(p),x,\delta'}$. Thus an antichain of the poset $\mathcal{R}_{G(p),x,\delta'}$ is also an antichain of the poset $\mathcal{R}_{G(p),x,\delta}$ and we conclude that $S(x, \delta') \leq S(x, \delta)$. □

So, given a schedule x of $G(p)$, it directly comes from Theorem 3.3 that $S^*(x, \Delta) = S(x, \delta^+)$ and can then be computed in polynomial time from the result of Section 3.

Let us now suppose that we are able to compute x^+ schedule of $G(p)$ whose value $S(x, \delta^+)$ is maximum over every schedule x of $G(p)$. From the definition of x^+, for every schedule x of $G(p)$ we have $S(x^+, \delta^+) \geq S(x, \delta^+)$. Since $S^*(x^+, \Delta) = S(x^+, \delta^+)$ and $S^*(x, \Delta) = S(x, \delta^+)$, we get that $S^*(x^+, \Delta) \geq S^*(x, \Delta)$, thus showing that x^+ is an anchored proactive schedule x^*_Δ.

Note that this problem is trivial since the schedules of $G(p+\delta^+)$ are schedules of $G(p)$. Indeed every schedule of $G(p+\delta^+)$ is an optimal solution with anchorage level equal to n. Furthermore the makespan of x^*_Δ may be overly large as it corresponds to the robust worst-case considered in Soyster (1973). Recall such an approach is bound to very conservative solutions. To alleviate such a shortcoming a deadline constraint will be considered in Subsection 5.2.

5.2. The Deadline constrained anchor-proactive CPM-scheduling problem in the robust worst-case

Consider $\delta \in \Delta$ with the special structure described in Subsection 5.1. The following *deadline constraint* is considered : the makespan of a baseline schedule must be equal to M, where M is a natural number such that $L_{G(p)}(0, n+1) \leq M < L_{G(p+\delta^+)}(0, n+1)$. The *Deadline constrained Anchor-Proactive CPM-Scheduling Problem* (DAPSP) is to find a anchored proactive schedule x^*_Δ over the schedules x of $G(p)$ with $x_{n+1} \leq M$.

In this section, we will prove the following result.

Theorem 5.2. *The Deadline constrained APSP in the robust worst-case can be solved in* $O(n^3)$.

The proof of Theorem 5.2 is derived from that presented to solve the ARSP where δ is known. Indeed, from the monotonicity Property 5.1, solving the DAPSP for all $\delta \in \Delta$ reduces to consider the disruption in the worst case denoted by δ^+.

To prove Theorem 5.2, we first need to introduce a new optimization problem (P) as follows. We consider graph $G(p)$, p and M as defined for the DAPSP and we also consider a given disruption $\delta \in \Delta$.

A *candidate solution* of an instance $I = (G, p, \delta, M)$ of the DAPSP is a pair (x, y) where

- $x = (x_0, \ldots, x_{n+1})$ is a schedule of $G(p)$ such that $x_{n+1} \leq M$,
- $y = (y_0, \ldots, y_{n+1})$ is a schedule of $G(p + \delta)$.

The problem (P) is then to find a candidate pair (x, y) so that anchorage level $\sigma(x, y)$ is maximum. We will conclude the proof of Theorem 5.2 using the fact that DAPSP reduces to problem (P) for $\delta = \delta^+$.

We now associate with the instance $I = (G, p, \delta, M)$ of (P) an instance $\widehat{G}(a)$ of the CPM-scheduling problem defined as follows:

- $\{0, \ldots, n+1\} \cup \{0', \ldots, (n+1)'\}$ are the jobs of $\widehat{G}(p)$,
- the valued subgraph of $\widehat{G}(a)$ induced by $\{0, \ldots, n+1\}$ is the same as $G(p)$,

- the valued subgraph of $\widehat{G}(a)$ induced by $\{0', \ldots, (n+1)'\}$ is the same as $G(p+\delta)$ (where each node i is replaced by node i')
- $(n+1, 0, -M)$ is an additional precedence constraint.

Note that valuation a of the arcs of $\widehat{G}(a)$ corresponds to the valuation of the arcs of $G(p)$ (resp. $G(p+\delta)$) for the subgraph induced by the nodes of $\{0, \ldots, n+1\}$ (resp. $\{0', \ldots, (n+1)'\}$).

We then have the following property.

Property 5.3. *The candidate solutions of instance $I = (G, p, \delta, M)$ of (P) are in one to one correspondence with the schedules τ of $\widehat{G}(a)$ such that $\tau(0) = \tau(0') = 0$.*

Proof. Let (x, y) be a solution of I. The corresponding τ is defined as follows: $\tau(i) = x_i$ for $i \in \{1, \ldots, n\}$ and $\tau(i') = y_i$ for $i \in \{1, \ldots, n\}$.
Let (u, v) be an arc of $\widehat{G}(a)$. If (u, v) is an arc of the subgraph of $\widehat{G}(a)$ induced by $\{0, \ldots, n+1\}$ or by the subgraph of $\widehat{G}(a)$ induced by $\{0', \ldots, (n+1)'\}$, then we clearly have $\tau(v) - \tau(u) \geq a_{uv}$. If (u, v) is the arc $(n+1, 0)$, we have $\tau(u) = x_{n+1}$, $\tau(v) = 0$ and $\tau(v) - \tau(u) = -x_{n+1}$. Since $x_{n+1} \leq M$, we get $\tau(v) - \tau(u) \geq -M = a_{uv}$. Thus τ is a schedule of $\widehat{G}(a)$. Moreover, the correspondence is one to one since a schedule τ of $\widehat{G}(a)$ is clearly the image of exactly one solution of I. □

A subset H of $\{1, \ldots, n\}$ is *compatible* if there is a candidate solution (x, y) of I such that for all $i \in H$, $x_i = y_i$. It comes directly from Property 5.3 that a subset H of $\{1, \ldots, n\}$ is compatible if there is a schedule τ of $\widehat{G}(a)$ such that $\tau(0) = \tau(0') = 0$ and for all $i \in H$, $\tau(i) = \tau(i')$.

Given a subset H of $\{1, \ldots, n\}$, we define from the instance $\widehat{G}(a)$ the instance $\widehat{G}^H(a)$ of the CPM-scheduling problem by adding, for each $i \in H \cup \{0\}$ the constraints $(i, i', 0)$ and $(i', i, 0)$. These additional arcs will be called the *transverse* arcs of $\widehat{G}^H(a)$. For example, by referring to the instance $G(p)$ shown in Figure 1, Figure 4 shows the valued graph $\widehat{G}^H(a)$ relative to $\delta^+ = (2, 2, 2, 0, 1)$, $M = 14$ and a subset $H = \{J_1, J_3, J_4\}$. Note that only the necessary arcs incident to

the nodes 0, 0′, $n+1$ and $(n+1)'$ are represented.

From the values of the minimum makespan of the worst-case instance is 16 and d the deadline M is 14 and the starting time of J_5 is 13

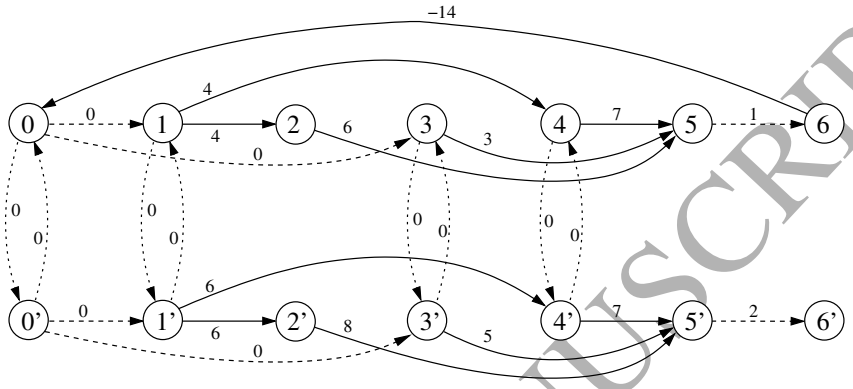

Figure 4: The valued graph $\widehat{G}^H(a)$ relative to $H = \{J_1, J_3, J_4, J_5\}$

The following property comes from the structure of the circuits of $\widehat{G}^H(a)$.

Property 5.4. *$H \subset \{1, \ldots, n\}$ is compatible if and only if the length of every path from 0 to $n+1$ in $\widehat{G}^H(a)$ is at most M.*

Proof. We know from Property 3.1 that H is compatible if and only if $\widehat{G}^H(a)$ has no simple circuit with a positive length. But, except for the circuits $(i, i', i), i \in H \cup \{0\}$, whose length is 0, any other simple circuit is made of a simple path from 0 to $n+1$ closed by the arc $(n+1, 0)$. □

The following property gives the first characterization of a compatible subset of jobs (the proof is given in the annex).

Property 5.5. *A subset of jobs H is compatible if and only if, for all $i \in H \setminus \{0\}$, $j \in H \setminus \{i\}$ such that there is a path from i to j in $G(p)$ we have $L_{G(p)}(0, i) + L_{G(p+\delta)}(i, j) + L_{G(p)}(j, n+1) \leq M$.*

We now show that the compatible subsets of an I of (P) are the antichains containing node 0 of a poset on $\{0, 1, \ldots, n\}$. Let us define the relation $\widehat{\mathcal{R}}_{G(p),\delta}$ on $\{0, 1, \ldots, n\}$ as follows: (i, j) is in $\widehat{\mathcal{R}}_{G(p),\delta}$ if $(i = j)$ or there is a path from

i to j in $G(p)$ such that $L_{G(p)}(0,i) + L_{G(p+\delta)}(i,j) + L_{G(p)}(j,n+1) > M$. We then have the following Lemma (the proof is given in the annex).

Lemma 5.6. *$\widehat{\mathcal{R}}_{G(p),\delta}$ is a poset on $\{0,1,\ldots,n\}$.*

We now consider the antichains of $\widehat{\mathcal{R}}_{G(p),\delta}$ that contain node 0 and get the following characterization of the compatible subsets (the proof is given in the annex).

Property 5.7. *Let H be a subset of $\{1,\ldots,n\}$. H is a compatible subset of the instance (G,p,δ,M) of (P) if and only if $H \cup \{0\}$ is an antichain of $\widehat{\mathcal{R}}_{G(p),\delta}$.*

It is now easy to derive from Property 5.7 a polynomial algorithm solving an instance $I = (G,p,\delta,M)$ of (P). The first step is to derive the poset $\widehat{\mathcal{R}}_{G(p),\delta}$, which requires to calculate the values $L_{G(p)}(0,i)$, $L_{G(p)}(i,n+1)$ and $L_{G(p+\delta)}(i,j)$ for all $i,j \in \{1,\ldots,n\}$: this can be done in $O(n^3)$ time. The second step computes a maximum antichain of $\widehat{\mathcal{R}}_{G(p),\delta}$ containing node 0. It is well known from Dilworth (1950) that finding a maximum antichain is a polynomial problem solved in $O(n^3)$ time by deriving a maximum matching of a bipartite graph with $2n$ nodes. To take into account the constraint on node 0, we only need to remove from $\widehat{\mathcal{R}}_{G(p),\delta}$ all the descendants of node 0, find a maximum antichain in the corresponding subgraph and add node 0 to the obtained antichain. Finally, to get an optimal solution (x,y) from the obtained antichain H, we only have to solve the corresponding instance $\widehat{G}^H(p)$ of the CPM-scheduling problem. With every schedule of that instance is associated an optimal solution of (G,p,δ,M). We thus get an $O(n^3)$ algorithm solving (P), which finally proves Theorem 5.2.

For example, by referring to the instance $\widehat{G}^H(a)$ shown in Figure 4, a possible candidate pair (x,y) corresponding to $H = \{J_1,J_3,J_4\}$ is $x = (0,5,0,6,13,14)$ and $y = (0,6,0,6,14,16)$. It can be proved that H is a compatible subset of maximal cardinality. Hence the maximum garanteed anchorage strength of the instance with deadline $M = 14$ is three and x a anchored proactive schedule.

6. Conclusion

In this article we have defined a combinatorial optimization framework that emphasizes the role of the so-called anchored solutions to control the detrimental effects of disruptions. In the case of CPM-scheduling problems with simple precedence constraints, this framework involves defining two variants of the latter problem depending whether the disruption is known or not. Interestingly, the anchorage level is directly related to the genuine combinatorial structure of the considered scheduling problem. When the disruption is known, the anchor-reactive variant of the scheduling problem is considered. We have provided a polynomial algorithm to compute an optimal schedule of the real instance. When the new schedule should respect a global time-window constraint, finding an optimal schedule has been shown to be an NP-hard problem. Finally when the disruption is not known, the anchor-proactive variant of the scheduling problem is considered. More particularly when each duration value belongs to a known interval, it has been shown how to compute a maximum guaranteed anchored schedule for the baseline instance over the uncertainty domain and also how to account for a deadline constraint. Such a deadline constraint enables us is to control the conservatism of the resulting schedule.

Future research will naturally investigate the computation of guaranteed anchored proactive schedules for other uncertainty domains. In this work we considered one reactive stage when a new schedule is computed to account for a given disruption. How much control remains during several reactive stages is one among many other questions left for future works. The same anchor-based approach concerning the single machine is also currently studied. More generally, for a lot of optimization problems, the anchorage strength of a solution with respect to some disruptions of the baseline instance is an important issue for decision makers and deserve to be thoroughly studied. This article is, for the scheduling domain, a first step in this direction.

References

Ben-Tal, A., El Ghaoui, L. & Nemirovski, A. (2009). Robust Optimization. *Princeton Series in Applied Mathematics, Princeton University Press*, 9-16.

Ben-Tal, A. & Nemirovski, A. (1999) Robust solutions to uncertain programs. *Oper. Res. Lett.*, 25(1).

Bertsimas, D., & Sim, M. (2003). Robust discrete optimization and network flows. *Mathematical Programming Series B*, 98:49-71.

Demmeulemeester, E. and Herroelen, W. Project Scheduling - A Research Handbook. *Inter. Series in Oper. Res. and Manag. Sci., vol 49, Boston: Kluwer Academic Publishers.*

Dilworth, R.P. (1950). A decomposition theorem for partially ordered sets. *Ann. Math.*, 51(1):161-166.

El-Ghaoui, L., Oustry, F. & Lebret, H. (1998) Robust solutions to uncertain semi-definite programs. *SIAM J. Optim.*, 9(3).

Garey, M. R., & Johnson, D. S. (1979). Computers and Intractability: A Guide to the Theory of NP-Completeness *W. H. Freeman & Co, New York, NY, USA.*

Herroelen, W., & Leus, R. (2004). The construction of stable project baseline schedules *Europ. J. of Operational Res.*, 156(3):550-565.

Herroelen, W., & Leus, R. (2004). Robust and reactive project scheduling: a review and classification of procedures *Int. j. prod. res.*, 42(8):1599-1620.

Herroelen, W., & Leus, R. & Demeulemeester, E. (2002). Critical chain project scheduling: do not oversimplify. *Project Management j.*, december 2002:48-60.

Minoux, M. (2008). Robust Linear Programming with Right-Handside Uncertainty, Duality and Applications. *Encyclopedia of Optimization*, 3317-3327.

Moazeni, S., Coleman, T.F. & Li, Y. (2013). Regularized robust optimization: the optimal portfolio execution case. *Computational Optimization and Applications*, 55(2):341-377.

Pinedo, M. (2002). Scheduling: Theory, Algorithms, and Systems. *Prentice Hall.*

Soyster, A.L. (1973). Convex programming with set-inclusive constraints and applications to inexact linear programming. *Operations Research*, 21:1154-1157.

Annex: Proof of Property 5.5

Proof. Let us denote by $\widehat{G}_0^H(p)$ (resp. $\widehat{G}_1^H(p)$) the subgraph of $\widehat{G}^H(a)$ induced by $\{0,\dots,n+1\}$ (resp. $\{0',\dots,(n+1)'\}$).

Let us first assume that $H \subset \{1,\dots,n\}$ is a compatible subset and assume i,j satisfy Conditions 1. and 2.. From Condition 2., we know there is a longest path μ' from i' to j' in $\widehat{G}_1^H(p)$ and that $l_{\widehat{G}_1^H(p)}(\mu') = L_{G(p+\delta)}(i,j)$. Let β be a longest path from 0 to i in $\widehat{G}_0^H(p)$ and let γ be a longest path from j to $n+1$ in $\widehat{G}_0^H(p)$. From these definitions, we have $l_{\widehat{G}_0^H(p)}(\beta) = L_{G(p)}(0,i)$ and $l_{\widehat{G}_0^H(p)}(\gamma) = L_{G(p)}(j,n+1)$. Since $\beta\cdot(i,i')\cdot\mu'\cdot(j',j)\cdot\gamma$ is a path from 0 to $n+1$ in $\widehat{G}^H(p)$ with length $L_{G(p)}(0,i)+L_{G(p+\delta)}(i,j)+L_{G(p)}(j,n+1)$, we have $L_{G(p)}(0,i)+L_{G(p+\delta)}(i,j)+L_{G(p)}(j,n+1) \le M$.

Conversely let μ be a simple path of $\widehat{G}^H(p)$ from 0 to $n+1$. If no arc (i,i') belongs to μ, then μ is a path of $\widehat{G}_0^H(p)$ and we have $l(\mu) \le M$. Otherwise, let (i,i') (resp. (j',j)) be the first (resp. last) transverse arc of μ. Since the nodes i and j satisfy Conditions 1. and 2., we have

$$L_{G(p)}(0,i)+L_{G(p+\delta)}(i,j)+L_{G(p)}(j,n+1) \le M. \tag{1}$$

Let us denote by $\mu[i',j']$ the subpath of μ from i' to j'. Since μ is an alternating sequence of subpaths of $\widehat{G}_0^H(p)$ and $\widehat{G}_1^H(p)$, we may define $\bar{\mu}[i',j']$ as the path of $\widehat{G}_1^H(p)$ we get when each subpath of μ that belongs to $\widehat{G}_0^H(p)$ is replaced by its corresponding path in $\widehat{G}_1^H(p)$. Figure 5 illustrates the definition of $\bar{\mu}[i',j']$.

From these definitions and since $\delta_i \ge 0$ for $i \in \{1,\dots,n\}$, we have

$$l_{\widehat{G}^H(p)}(\mu[i',j']) \le l_{\widehat{G}_1^H(p)}(\bar{\mu}[i',j']) \le L_{\widehat{G}_1^H(p)}(i',j') = L_{G(p+\delta)}(i,j).$$

From the definition of i and j, we have

$$l_{\widehat{G}^H(p)}(\mu) = l_{\widehat{G}_0^H(p)}(\mu[0,i]) + l_{\widehat{G}^H(p)}(\mu[i',j']) + l_{\widehat{G}_0^H(p)}(\mu[j,n+1]).$$

Finally, since $l_{\widehat{G}_0^H(p)}(\mu[0,i]) \le L_{G(p)}(0,i)$, $l_{\widehat{G}^H(p)}(\mu[i',j']) \le L_{G(p+\delta)}(i,j)$ and $l_{\widehat{G}_0^H(p)}(\mu[j,n+1]) \le L_{G(p)}(j,n+1)$, we conclude from inequality (1) that $l_{\widehat{G}^H(p)}(\mu) \le M$. □

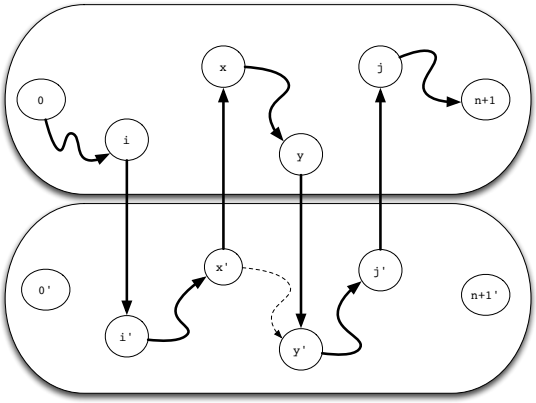

Figure 5: The subpaths $\mu[i,j]$ and $\bar{\mu}[i',j']$

Annex: Proof of Lemma 5.6

Proof. $\widehat{\mathcal{R}}_{G(p),\delta}$ is clearly reflexive. Since $G(p)$ has no circuit, $\widehat{\mathcal{R}}_{G(p),\delta}$ is also anti-symmetric. Assume now now that i, j, and k are three distinct nodes such that $(i,j) \in \widehat{\mathcal{R}}_{G(p),\delta}$ and $(j,k) \in \widehat{\mathcal{R}}_{G(p),\delta}$. We know that, in $G(p)$, there is a path μ from i to j and a path ν from j to k. So $\mu \cdot \nu$ is a path of $G(p)$ from i to k. Moreover we have

$$L_{G(p)}(0,i) + L_{G(p+\delta)}(i,j) + L_{G(p)}(j,n+1) > M \tag{2}$$

$$L_{G(p)}(0,j) + L_{G(p+\delta)}(j,k) + L_{G(p)}(k,n+1) > M \tag{3}$$

Since $L_{G(p)}(0,j) + L_{G(p)}(j,n+1) \leq L_{G(p)}(0,n+1) \leq M$ and $L_{G(p+\delta)}(i,j) + L_{G(p+\delta)}(j,k) \leq L_{G(p+\delta)}(i,k)$, we get by summing (2) and (3) that

$$L_{G(p)}(0,i) + L_{G(p+\delta)}(i,k) + L_{G(p)}(k,n+1) > M.$$

Thus $\widehat{\mathcal{R}}_{G(p),\delta}$ is a poset on $\{0,1,\ldots,n\}$. □

Annex: Proof of Lemma 5.7

Proof. Assume first that H is a compatible subset and let $i,j \in H \cup \{0\}$ be such that $i < j$. If there is no path in $G(p)$ from node J_i to node J_j, then we

have $(i,j) \notin \widehat{\mathcal{R}}_{G(p),\delta}$. If there is such a path, then we have from Property 5.5 that $L_{G(p)}(0,i) + L_{G(p+\delta)}(i,j) + L_{G(p)}(j,n+1) \leq M$ from which we conclude that $(i,j) \notin \widehat{\mathcal{R}}_{G(p),\delta}$. So $H \cup \{0\}$ is an antichain of $\widehat{\mathcal{R}}_{G(p),\delta}$.
Conversely let $H \cup \{0\}$ be an antichain of $\widehat{\mathcal{R}}_{G(p),\delta}$ and assume that $i \in H \cup \{0\}$, $j \in H$ and $i \neq j$. Since $(i,j) \notin \widehat{\mathcal{R}}_{G(p),\delta}$, we have $L_{G(p)}(0,i) + L_{G(p+\delta)}(i,j) + L_{G(p)}(j,n+1) \leq M$. So we get from Property 5.5 that H is a compatible subset of the instance (G,p,δ,M). □