CIRRELT

Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation

Optimal Resource-Constraint Project Scheduling with Overlapping Modes

François Berthaut
Lucas Grèze
Robert Pellerin
Nathalie Perrier
Adnène Hajji

February 2011

CIRRELT-2011-09

Bureaux de Montréal :
Université de Montréal
C.P. 6128, succ. Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Optimal Resource-Constraint Project Scheduling with Overlapping Modes

François Berthaut[1,2], Lucas Grèze[2], Robert Pellerin[1,2,*], Nathalie Perrier[1,2], Adnène Hajji[1,3]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Mathematics and Industrial Engineering, École Polytechnique de Montréal, P.O. Box 6079, Station Centre-ville, Montréal, Canada H3C 3A7

[3] Department of Operations and Decision Systems, Université Laval, 2325, de la Terrasse, Québec, Canada G1V 0A6

Abstract. Overlapping activities is widely used to accelerate project execution. Overlapping consist in executing in parallel two sequential activities by allowing a downstream activity to start before the end of an upstream activity based on preliminary information. In companies, overlapping is examined in resource constraints scheduling without considering interaction between activities and rework caused by alteration of information exchanged until finalized information is available at the completion of the upstream activity. By contrast, the literature deals with overlapping of couple of activities without considering a whole project with resource constraints. We here investigate the resource-constrained project scheduling problem with different feasible modes of overlapping including associated rework. We formulate the optimization problem as an integer linear programming problem. An example of a 30 activity project is provided to illustrate the utility and efficiency of this model. An optimal solution is reached within reasonable computation time. Our results also highlight the closed interaction between resource constraints and overlapping modes and confirms the relevance of jointly consider them.

Keywords. Activity overlapping, concurrent engineering, project management, project scheduling.

Acknowledgements. This work was supported by the Jarislowsky Foundation, the SNC-Lavalin firm and by the Natural Sciences and Engineering Council of Canada (NSERC). This support is gratefully acknowledged.

---

* Corresponding author: Robert.Pellerin@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2011

CIRRELT

Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation

Optimal Resource-Constraint Project Scheduling with Overlapping Modes

François Berthaut
Lucas Grèze
Robert Pellerin
Nathalie Perrier
Adnène Hajji

February 2011

CIRRELT-2011-09

Bureaux de Montréal :
Université de Montréal
C.P. 6128, succ. Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Optimal Resource-Constraint Project Scheduling with Overlapping Modes

François Berthaut[1,2], Lucas Grèze[2], Robert Pellerin[1,2,*], Nathalie Perrier[1,2], Adnène Hajji[1,3]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Mathematics and Industrial Engineering, École Polytechnique de Montréal, P.O. Box 6079, Station Centre-ville, Montréal, Canada H3C 3A7

[3] Department of Operations and Decision Systems, Université Laval, 2325, de la Terrasse, Québec, Canada G1V 0A6

Abstract. Overlapping activities is widely used to accelerate project execution. Overlapping consist in executing in parallel two sequential activities by allowing a downstream activity to start before the end of an upstream activity based on preliminary information. In companies, overlapping is examined in resource constraints scheduling without considering interaction between activities and rework caused by alteration of information exchanged until finalized information is available at the completion of the upstream activity. By contrast, the literature deals with overlapping of couple of activities without considering a whole project with resource constraints. We here investigate the resource-constrained project scheduling problem with different feasible modes of overlapping including associated rework. We formulate the optimization problem as an integer linear programming problem. An example of a 30 activity project is provided to illustrate the utility and efficiency of this model. An optimal solution is reached within reasonable computation time. Our results also highlight the closed interaction between resource constraints and overlapping modes and confirms the relevance of jointly consider them.

Keywords. Activity overlapping, concurrent engineering, project management, project scheduling.

Acknowledgements. This work was supported by the Jarislowsky Foundation, the SNC-Lavalin firm and by the Natural Sciences and Engineering Council of Canada (NSERC). This support is gratefully acknowledged.

---

* Corresponding author: Robert.Pellerin@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2011

1 Introduction

The RCPSP (Resource-Constrained Project Scheduling Problem) has been addressed in numerous papers. Various models attempt to minimize project time completion while considering limited resources [1]. Hartmann and Briskorn [2] have also presented an overview of different RCPSP extensions. Among extensions addressed in the literature, different practices have been developed to reduce time of project execution in order to establish a baseline schedule or to modify it following project delay during its execution through overlapping, crashing and substitution of activities [13]. In engineering projects, overlapping is considered as “a core technique for saving development time” [3][4] [5]. Overlapping consists in starting an activity before receiving all the final information required. This practice often causes future rework and modification as new information is gained in subsequent activities. As such, the total reduction of time is the difference between overlapped time and rework. Nevertheless, most models in the literature consider crashing as a strategy to reduce the project makespan without taking into account possible rework.

In practice, overlapping depends not only on dependency between activities but also on information exchange policy between upstream and downstream activity and progress evolution. Two groups of models have been developed in the literature to analyze overlapping interactions. First, many authors consider only couples of activities and no resource constraints to establish the best trade-off between overlapping and rework. For instance, Krishnan et al. [7] developed a model-based framework to manage the overlapping of coupled activities. This model introduced the concept of information evolution and downstream sensitivity to describe interaction between both activities. Information evolution refers to the upstream generated information useful for downstream activities. Downstream sensitivity refers to the impact of a change in upstream activity on the downstream activity. The more the impact is significant, the more the sensitivity is high. Relying on these concepts, Krishnan [8] defined different types of appropriated overlapping strategies: iterative, preemptive, distributive and divisive overlapping. In a similar manner, Bogus et al. [9] identified appropriate strategies to efficiently implement overlapping in practice. Lin et al. [6] also improved the overlapping model by incorporating the downstream progress evolution and determined the optimal overlap amount. These models assume that overlapping parameters can be derived from historical data.

Other approaches have considered whole projects instead of coupled of activities under the assumption that relation between overlapping amount and rework is preliminary known for overlappable activities. They mostly use design structure matrix (DSM) to represent dependencies, to minimize feedbacks, and to identify overlapping opportunities between activities. DSMs were introduced by Steward [10]. Among other models, Gerk and Qassim [13] developed an analytic project acceleration linear model via activity crashing, overlapping and substitution with resource constraints. Wang and Lin [11] developed a stochastic overlapping process model to assess schedule risks. Their simulation model considers iterations and probabilities of rework. Iterations are mostly defined as interaction between design activities which lead to rework caused by feedbacks from downstream activities. However, their model does not take into account resource constraints. Cho and Eppinger [12] also introduced a simulation model with stochastic activity durations, overlapping, iterations, rework and considered resource constraints for some activities. They showed that theses constraints can delay some overlapped activities and delay the project. All these papers assume a simple linear relationship between rework and overlapping amount with an upper and lower bound. It is also important to note that Lin et al. [6] stated that companies determine overlapping strategies on an ad hoc basis without always considering rework and interaction between activities.

In summary, there are no RCPSP model that yet considers a realistic relationship between rework and overlapping amount. The objective of this paper is to extend the classic RCPSP with a realistic overlapping model in order to partially fill this gap. We here assume that the information flow is unidirectional from upstream to downstream activities. Consequently, the rework caused by execution of activities based on preliminary information is only assigned to the downstream activities of the identified overlappable activities. Information exchange is assumed to be costless and instantaneous. The main difference with the aforementioned overlapping models is that overlapping is restricted to a set of feasible overlap amounts, instead of considering a continuous and linear relation between overlap amount and rework. We formulate this resource-constrained project scheduling problem with overlapping modes as an integer linear programming problem, which is closed to the classical multi-mode RCPSP model [1]. This model allows finding an optimal makespan in reasonable calculation time.

The remainder of the paper is organized as follow. Section 2 first describes the problem statement and assumptions. An illustrative example and computational results are then presented in section 3. The paper concludes with recommendations for future work in section 4.

2 Problem Statement

A project is defined by a set of activities, S, including two fictitious activities 0 and n+1, which correspond to the project start and project end, respectively, with zero processing time. We denote by d_j the nominal processing time of activity j considering that all the final information required from preceding activities are available at its start; in other word, if activity j is processed without overlapping. All the symbols and their definitions used along this paper are presented in Table 1.

Table 1: Symbols and definitions

Symbol	Definition
S	Set of activities
n	Number of non-dummy activities
E	Set of temporal or precedence constraints
$i \rightarrow j$ (i, j)	Precedence constraint
d_j	Processing time of activity j
A	Set of couples of overlappable activities
A_j	Set of immediate predecessors of activity j that are overlappable with activity j
P_j	Set of immediate predecessors of activity j that are not overlappable with activity j
$Pred(j)$	Set of immediate predecessors of activity j
R	Set of renewable resources
R_k	Constant amount of available units of renewable resource k
R_{jk}	Per period usage of activity j of renewable resource k
n_{ij}	Number of precedence modes of the couple (i,j)
β_{ijn}	Amount of overlap duration between activities i and j in precedence mode n, expressed as a fraction of d_i
μ_{ijn}	Expected amount of rework in the downstream activity j when activities i and j are overlapped in precedence mode n
m_j	Number of execution modes of activity j
α_{ijm}	Amount of overlap duration between activities i and j in execution mode m, expressed as a fraction of d_i
r_{jm}	Expected amount of rework in activity j in execution mode m
T	Upper bound of the project's makespan
$t = 0,..,T$	Periods
EF_j	Earliest possible finish time of activity j
LF_j	Latest possible finish time of activity j

2.1 Precedence constraints

Frequently used project-planning methods provide graphic descriptions of task workflows in the form of the so-called activity-on-node or activity-on-arc networks. These networks depict the logical execution sequence of dependent (sequential) activities and independent (parallel or concurrent) activities. However, these tools fail to incorporate interdependent-type relation, activities' iterations and to model information flows between activities.

The Design Structure Matrix (DSM) representation can handle these additional relations between activities with the broader concept of information sharing [14]. Information exchange between activities can occur at the beginning, the middle or the end of an activity and includes both tangible and intangible types such as parts, part dimensions, and bill of materials, which constitute the outputs from an upstream activity and are required to begin the work of a downstream activity. A DSM is a square matrix where rows and columns represent activities. The DSM aims to represent the information flows for a given subset of activities and constitutes the first step in analyzing potential feedbacks. Indeed, feedback information exchanges from downstream to upstream activities correspond to design modification requests due to inability to meet target design requirements or design flaws detected in downstream stages [11]. Any feedback information exchange from downstream activities lead to modifications and reworks performed by the upstream activities to accommodate these changes and iterations between upstream and downstream activities can virtually occur to fix the problems identified. In order to minimize feedbacks, the DSM can be partitioned using block triangularization algorithm to obtain a unidirectional sequence of information exchange [14]. As a last resort, activities can be aggregated or decomposed into lower-level activities to eliminate feedbacks.

In this paper, we assume that such preliminary studies have been conducted to identify the nature of relations between activities and to determine a feasible sequence of activities without any feedback from downstream activities. The project is then only composed of independent and dependent activities and the resulting information flow within the project between activities is assumed to be unidirectional from upstream to downstream activities.

The analysis of information exchanges between dependent couple activities enables to categorize them into non-overlappable and overlappable ones. The former represents the case where a downstream activity requires the final output information from an upstream activity to be executed or the completion of the upstream activity. The latter represents the case where a downstream activity can begin with preliminary information and receives final update at the end of the upstream activity. This relation provides the opportunity to overlap two activities so that a downstream activity can start before an upstream activity is finished. While the non-overlappable activities are connected with the classical finish-to-start precedence constraint, the overlappable ones are connected with a finish-to-start-plus-lead time precedence constraint where the lead-time accounts for the amount of overlap.

In the remainder of the paper, we denote by A and P the sets of couples of overlappable and non-overlappable activities, respectively. Similarly for each activity j, Aj and P_j represent the set of immediate predecessors that are overlappable and non-overlappable with activity j, respectively. The set of precedence constraints in the project and the set of immediate predecessors of activity j are defined by:

$$E = A \bigcup P$$
$$Pred\,(j) = A_j \bigcup P_j \,, \forall j \in S \qquad (1)$$

2.2 Model of the overlapping process

Figure 1 shows the overlapping process of two activities i and j in A. The downstream activity j starts with preliminary inputs from the downstream activity i. The amount of overlap, α_{ij}, is expressed as a fraction of the downstream activity's duration. As the upstream activity proceeds, its information evolves to its final form and is released to the downstream activity j at its completion. This approach implies that the traditional pattern of exchange of finalized information at the end of the upstream activity is altered to a more frequent exchange of evolving information during the overlap period. However, additional rework is often necessary to accommodate the changes in the upstream information in the downstream development.

The expected duration of this rework is denoted by r_{ij}. The total amount of time required to execute both activities, D_{ij}, is expressed as follows:

$$D_{ij} = d_i + d_j \cdot (1 - \alpha_{ij}) + r_{ij} \tag{2}$$

If $d_j \geq d_i$, the amount of overlap is usually bounded by the fraction d_i/d_j in order to prevent the downstream activity to start before the upstream activity. If overlapping was not applied, the total amount of time required to execute both activities would simply be $D_{ij} = d_i + d_j$. Depending on the nature of the activities, there may exist a trade-off between time gains from overlapping and rework.

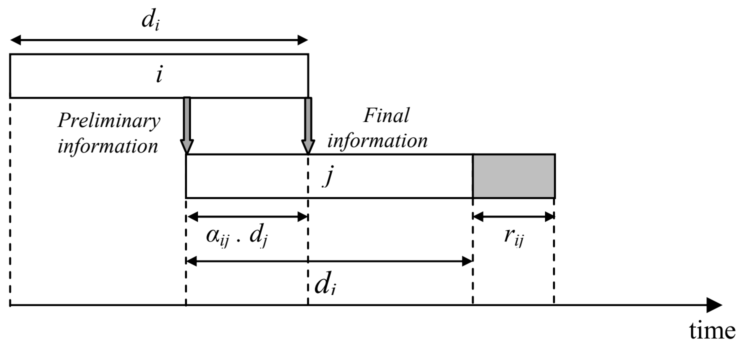

Fig. 1. Overlapping process of two activities

The main issue with the overlapping problem is to quantify the amount of rework as a function of the amount of overlap. A significant part of the literature in overlapping is dedicated to the determination of the optimal overlap amount for two activities without resource constraints. Indeed, the overlapping problem requires exploring the behavior and interaction of activities during their processes. Krishnan et al. [7] presented a pioneer paper in this field. They proposed a model of dependency based on the upstream information evolution which characterizes the refinement of information from its preliminary form to a final value and the downstream sensitivity which represents the duration of a downstream iteration to incorporate upstream changes. Loch and Terwiesch [15] adapted these concepts by considering the upstream evolution as the rate of modifications in the upstream and the downstream sensitivity as the impact of a modification on downstream rework, and jointly analyzed overlapping and communication policies between two activities. Roemer et al. [16] introduced the concept of probability of rework as a function of the overlap duration which encompasses both the evolution and sensitivity models proposed in [7]. While Krishnan et al. [7] and Loch and Terwiesch [15] addressed the project's makespan minimization problem, Roemer et al. [16], Lin et al. [17] and Lin et al. [6] extended the model of Loch and Tewiesch [15] to examine the time-cost tradeoffs in overlapping. Communication policies should be considered along overlapping if the information exchange between activities require non-negligible time and cost [6]. We here assume that information exchange is costless and instantaneous.

An important finding of the aforementioned papers is that the duration of rework is a convex increasing function of the amount of overlap and that the time to complete the upstream and downstream activities is convex in the amount of overlap. The former statement is intuitive: if the amount of overlap increases, then the preliminary information at the downstream activity's start will be more unreliable and more downstream changes must be incorporated. The latter statement entails that there exists a unique amount of overlap such that the gain from overlapping exceeds the loss due to rework in an optimal manner. According to the upstream evolution, the downstream dependency, the learning effect and the activity durations, the additional rework may not evolve faster than the overlap duration when overlapping increases, which indicates that the total amount of time required to execute activities is monotonic increasing and that complete overlapping is optimal. The conditions for optimality of complete overlapping are derived in [6][16][17] under different assumptions.

2.3 Precedence and overlapping modes

In order to study the interaction between overlapping and resource constraints in the scheduling optimization problem with multiple activities including several overlapping opportunities, the relation between rework and overlap amount is required for a range of overlap amounts for each couple of overlappable activities . Indeed, the optimal overlap amounts for a resource-constraints project composed of several couples of overlappable activities are not necessarily set to the optimal values found for each couple of activities [12][13][18].

In this paper, overlapping is assumed to be defined for discrete values of overlap durations. First, this assumption is more realistic considering that scheduling is performed in practice on a period-by-period basis (i.e., hour, day, week): resource availabilities and allocations are estimated per period, while activity durations are discrete multiples of one period [1]. Second, activity progress is measured in practice according to the completion of internal milestones which corresponds to important events, such as design criteria frozen, detailed design completed, drawings finalized, or any activity deliverables. This preliminary information is issued at intermediate points and used as input for a downstream activity. Therefore, the start time of an overlapped downstream activity is restricted to a finite number of instants corresponding to upstream activities' milestones which constitutes different feasible modes for the execution of overlapping activities. Each overlapping mode is characterized by an amount of overlap expressed as a fraction of the downstream activity's duration and a rework duration.

Overlapping modes can be generalized to precedence modes in order to describe all precedence relationships between activities. For each couple of precedence constraints $i \rightarrow j$, there exists at least one precedence mode n_{ij} which corresponds to a basic finish-to start relation without overlapping. When $(i,j) \in A$, there exist additional precedence modes associated with the different overlapping strategies. Thus, precedence modes can be expressed as follows:

$$n_{ij} = 1, \quad \beta_{ij1} = 0, \ \mu_{ij1} = 0, \qquad \forall j \in S, \ \forall i \in P_j \tag{3}$$

$$n_{ij} > 1, \quad \beta_{ij1} = 0, \ \mu_{ij1} = 0 \text{ and } 0 < \beta_{ijn} < 1, \ \mu_{ijn} > 0, \ \forall n \in [2, n_{ij}], \forall j \in S, \ \forall i \in A_j \tag{4}$$

where β_{ijn} and μ_{ijn} denote the amount of overlapped time between activities i and j and the expected amount of rework in the downstream activity j when activities i and j are executed in precedence mode n=1,.., n_{ij}.

When i and j are overlappable, they can be either overlapped and executed in mode n, $n=2,\ldots,n_{ij}$, or sequentially performed in mode n=1 without overlapping. As depicted in Figure 2, it is important to note that the precedence constraints on the finish time of activities i and j will defer depending on the overlapping mode: when not overlapped, the downstream activity start time is superior or equal to the upstream activity finish time, whereas the downstream activity start time is equal to the upstream activity finish time minus one of the feasible overlap duration in the case of overlapping.

2.4 Multiple overlapping and activity modes

We assume that there is no restriction concerning the number of overlappable or non-overlappable predecessors. If an activity is overlapped by multiple upstream activities, feasible overlapping modes are assumed to be compatible. Consider for example the case of a downstream activity j with two upstream activities, denoted by i_1 and i_2. If both couples (i_1, j) and (i_2, j) are overlapped, the amount of rework in downstream activity is between

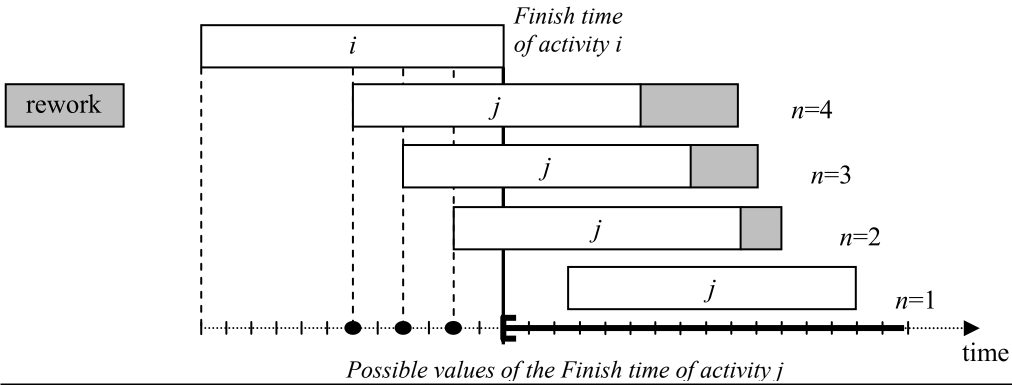

Fig. 2. Precedence constraints on the finish times of two overlappable activities *i* and *j*

the maximum of single rework and the sum of them, depending on the duplicate rework, as stated in [12]. Without loss of generality, the latter is considered in the model.

In typical projects involving engineering phases, the number of precedence and overlapping relationships may largely exceeds the number of activities. As each activity can have several overlappable or non-overlappable predecessors, we introduce the notion of execution modes associated to activities. Each activity mode represents a combination of possible precedence or overlapping modes of an activity with its overlappable or non-overlappable predecessors. Consequently, the set of activity modes for each activity is generated by a full factorial design of the precedence and overlapping modes with its predecessors. Tables 2a, 2b and 2c show the activity modes in the case of non-overlappable predecessors, only one overlappable predecessor (with four overlapping modes), and two predecessors (each with three overlapping modes), respectively.

Table 2a: execution mode of activity *j* in the case of non-overlappable predecessors

m	α_{ijm}, $\forall i \in P_j$	r_{jm}
1	0	0

Table 2b: execution mode of activity *j* in the case of one overlappable predecessor

	$i \in A_j$		$\forall k \in P_j$		
m	n_{ij}	α_{kjm}	n_{kj}	α_{ijm}	r_{jm}
1	1	*0*	1	0	*0*
2	2	β_{kj2}	1	0	μ_{kj2}
3	3	β_{kj3}	1	0	μ_{kj3}
4	4	β_{kj4}	1	0	μ_{kj4}

Table 2c: execution mode of activity *j* in the case of two overlappable predecessors

	$i_1, i_2 \in A_j$				$\forall k \in P_j$		
m	$n_{i1,j}$	$n_{i2,j}$	$\alpha_{i1,j,m}$	$\alpha_{i2,j,m}$	n_{kj}	α_{kjm}	r_{jm}
1	1	1	0	0	1	0	0
2	1	2	0	$\beta_{i2,j,2}$	1	0	$\mu_{i2,j,2}$
3	1	3	0	$\beta_{i2,j,3}$	1	0	$\mu_{i2,j,3}$
4	2	1	$\beta_{i1,j,2}$	0	1	0	$\mu_{i1,j,2}$
5	2	2	$\beta_{i1,j,2}$	$\beta_{i2,j,2}$	1	0	$\mu_{i1,j,2}+\mu_{i2,j,2}$
6	2	3	$\beta_{i1,j,2}$	$\beta_{i2,j,3}$	1	0	$\mu_{i1,j,2}+\mu_{i2,j,3}$
7	3	1	$\beta_{i1,j,3}$	0	1	0	$\mu_{i1,j,3}$
8	3	2	$\beta_{i1,j,3}$	$\beta_{i2,j,2}$	1	0	$\mu_{i1,j,3}+\mu_{i2,j,2}$
9	3	3	$\beta_{i1,j,3}$	$\beta_{i2,j,3}$	1	0	$\mu_{i1,j,3}+\mu_{i2,j,3}$

2.5 The 0-1 Integer Linear Programming model

Each activity j must finish within the time window $\{EF_j,...,LF_j\}$ with respect to the precedence relations and the activity durations. They can be derived from the traditional forward recursion and backward recursion algorithms considering that the project must start at time 0 and that T constitutes an upper bound of the project's makespan (i.e., the sum of processing times of all activities) [1]. We define the decision variables (i.e., the finish times and the overlapping modes) as follows:

$$X_{jtm} = \begin{cases} 1 & \text{if activity } j \text{ is executed in mode } m \text{ and finsished at time } t \\ 0 & \text{otherwise} \end{cases}$$

$$\forall j \in S\,,\ \forall t \in [0,T] \text{ and } \forall m \in [1, m_{ij}] \quad (5)$$

The decision on the activity modes can be classed into three cases. On the one hand, if activities (i,j) are not overlappable, the decision is simply not to overlap. On the other hand, if activities (i, j) are overlappable, these activities can be either overlapped (m > 1) or executed in series (m =1). The resource-constrained scheduling problem with overlapping can then be formulated with the 0-1 integer non-linear programming model as follow:

$$\text{Minimize} \sum_{m=1}^{m_{n+1}} \sum_{t=EF_{n+1}}^{LF_{n+1}} t \cdot X_{n+1,t,m} \quad (6)$$

Subject to :

$$\text{If } \sum_{t=EF_j}^{LF_j} X_{jt1} = 1 \text{ then } \sum_{m=1}^{m_i} \sum_{t=EF_{n+1}}^{LF_{n+1}} t \cdot X_{itm} \leq \sum_{t=EF_{n+1}}^{LF_{n+1}} \left(t - d_j\right) \cdot X_{jt1}\ , \quad \forall j \in S\,, \forall i \in Pred(j) \quad (7)$$

$$\text{If } \sum_{m=2}^{m_j} \sum_{t=EF_j}^{LF_j} X_{jt1} = 1 \text{ then } \sum_{m=1}^{m_i} \sum_{t=EF_{n+1}}^{LF_{n+1}} t \cdot X_{itm} = \sum_{m=2}^{m_j} \sum_{t=EF_{n+1}}^{LF_{n+1}} \left(t - d_j \cdot (1-\alpha_{ijm}) - r_{jm}\right) \cdot X_{jtm}\ , \quad \forall j \in A\,, \forall i \in A_j \quad (8)$$

$$\sum_{j=2}^{n} \left[R_{jk} \cdot \left(\sum_{m=1}^{m_j} \sum_{b=t}^{t+d_j-1+r_{jm}} X_{jbm} \right) \right] \leq R_k\ , \qquad \forall k \in R\ \ \forall t \in [0,T] \quad (9)$$

$$\sum_{m=1}^{m_i} \sum_{t=EF_i}^{LF_i} t \cdot X_{itm} \leq \sum_{m=1}^{m_j} \sum_{t=EF_j}^{LF_j} t \cdot X_{jtm}\ , \qquad \forall j \in S\,, \forall i \in A(j) \quad (10)$$

$$\sum_{m=1}^{m_j} \sum_{t=EF_j}^{LF_j} X_{jtm} = 1\ , \qquad \forall j \in S \quad (11)$$

$$X_{jtm} = \{0,1\}\ , \qquad \forall j \in S\,,\ \forall t \in [0,T] \text{ and } \forall m \in [1, m_{ij}] \quad (12)$$

The objective (6) minimizes the finish time of the dummy sink activity and therefore, the project's makespan. Constraints (7) represent the finish-to-start precedence constraints when activities are not overlapped. If activities are overlapped, constraints (8) state that the downstream activity must start at the upstream activity finish time minus one of the feasible overlap duration. Constraints (7) and (8) reflect the precedence and overlapping constraints presented in Figure 2. Constraints (9) define the resource constraints. Constraints (10) guarantee that the downstream activity of a couple of overlappable activities can not finish before the upstream activity's finish time. Constraints (11) ensure that each activity is assigned one activity mode and one finish time. Finally, constraints (12) define the aforementioned binary decision variables.

The 0-1 integer non-linear programming model given by the objective (6) and the constraints (7)-(12) can be transformed to a 0-1 integer linear programming model. Constraints (7) and (8) are reformulated as follows:

$$\sum_{m=1}^{m_i} \sum_{t=EF_{n+1}}^{LF_{n+1}} t \cdot X_{itm} \leq \sum_{m=1}^{m_j} \sum_{t=EF_{n+1}}^{LF_{n+1}} \left(t - d_j \cdot (1-\alpha_{ijm}) - r_{jm}\right) \cdot X_{jtm} \ , \quad \forall j \in S \ , \forall i \in Pred(j) \tag{13}$$

$$\sum_{m=1}^{m_j} \sum_{t=EF_{n+1}}^{LF_{n+1}} \alpha_{ijm} \cdot X_{jtm} \leq Y_{ij} \ , \quad \forall j \in S \ , \forall i \in A(j) \tag{14}$$

$$\sum_{m=1}^{m_i} \sum_{t=EF_i}^{LF_i} t \cdot X_{itm} \geq \left(\sum_{m=1}^{m_j} \sum_{t=EF_i}^{LF_i} \left(t - d_j (1-\alpha_{ijm}) - r_{jm}\right) \cdot X_{jtm} \right) - T \cdot (1 - Y_{ij}) \ , \quad \forall j \in S \ , \forall i \in A(j) \tag{15}$$

$$Y_{ij} = \{0,1\} \qquad \forall j \in S \ , \forall i \in Pred(j) \tag{16}$$

Note that Y_{ij} is an additional binary variable. Constraints (13) represent the finish-to-start precedence constraints, with a negative lead time in the case of overlapping. According to constraints (14), if two overlappable activities (i, j) are overlapped, then $Y_{ij} = 1$ and thus the union of constraints (13) and (15) is equivalent to the equality constraints (8). If activities (i, j) are not overlapped, then Y_{ij} is unrestricted and constraints (15) are not restrictive.

In view of significant efficiency of linear programming solutions method [19][20], and the implementation of these methods in commercial software packages, the linearization of the aforementioned precedence constraints is extremely useful.

3 Illustrative example

3.1 Data

We consider a project instance generated by Kolisch and Sprecher [21] composed of 30 non-dummy tasks and 4 renewable resources. The activity durations, resource consumptions, resource availabilities and precedence relations are similar to those presented in [22]. As no overlapping was defined in the original instance, the additional overlapping data have been generated (i.e., A, M_{ij}, α_{ijm},r_{jm}). Eight couples and two triplets of overlappable activities have been considered, as depicted in Tables 3 and 4. As a reminder, the overlapping amount and rework for non-overlappable activities are null.

Table 3 : Overlapping data for the couples of overlappable activities

Upstream activity i	Downstream activity j	Mode m	α_{ijm}	r_{jm}	Upstream activity i	Downstream activity j	Mode m	α_{ijm}	r_{jm}
3	7	1	0	0	10	16	1	0	0
		2	0.2	0			2	0.1	0
		3	0.6	1			3	0.3	0
3	8	1	0	0			4	0.4	1
		2	1/9	0			5	0.5	3
		3	3/9	1	13	18	1	0	0
		4	3/10	1			2	0.2	0
		5	4/10	3			3	0.6	1
4	10	1	0	0	11	26	1	0	0
		2	2/7	0			2	1/7	0
		3	3/7	0			3	3/7	1
		4	4/7	2			4	4/7	3
2	11	1	0	0	8	27	1	0	0
		2	1/9	0			2	0.125	0
		3	1/3	1			3	0.25	0
		4	5/9	2			4	0.375	1

Table 4: Overlapping data for the triplets of overlappable activities

Upstream activities i_1	i_2	*Downstream activity j*	*Mode m*	α_{i1jm}	α_{i2jm}	r_{jm}	Upstream activities i_1	i_2	*Downstream activity j*	*Mode m*	α_{i1jm}	α_{i2jm}	r_{jm}
11	18	20	1	0	0	0	16	17	22	1	0	0	0
			2	0	1/7	0				2	0	2/9	1
			3	0	2/7	1				3	0	3/9	1
			4	1/7	0	0				4	1/7	0	0
			5	1/7	1/7	0				5	1/7	2/9	1
			6	1/7	2/7	1				6	1/7	3/9	1
			7	3/7	0	1				7	3/7	0	1
			8	3/7	1/7	1				8	3/7	2/9	2
			9	3/7	2/7	2				9	3/7	3/9	2

3.2 Results

The illustrative case was implemented in AMPL Studio v1.6.j and solved with Cplex 12.2. In the original project extracted from PSPLIB, the optimal project makespan is 38 and 43 time units with and without resource constraints, respectively. Because of the overlapping opportunities defined in our illustrative example, some activities can start before the end of some of their predecessors based on preliminary information and with additional rework. Consequently, some overlappable activities can start earlier than scheduled in the original optimal schedule. The resulting optimal makespan with overlapping modes is 34 and 39 time units with and without resource constraints, respectively, as shown in Table 5.

Table 5: Effects of resource constraints and overlapping on the optimal makespan and the computation time

Case	Resources constraints	Overlapping Modes	Number of overlapped activities	Number of overlappables activities	Optimal makespan	CPU's Time
1	No	No	0	0	**38**	0.05
2	Yes	No	0	0	**43**	0.48
3	No	Yes	6	12	**34**	0.41
4	Yes	Yes	4	12	**39**	3.70

For the scheduling problem without resource constraints and with overlapping modes (case 3), the overlapping modes of the non-critical overlappable activities obtained in the optimal schedule are not the overlapping modes with the higher gain when overlappable activities are considered separately. Therefore, overlappable activities on the critical path are overlapped at their local optimum, as any reduction of the time to execute critical activities will decrease the project makespan. As highlighted in Table 5, only half of the set of overlappable activities are indeed overlapped for the scheduling problem without resource constraints and with overlapping modes. Even though the computational times are reasonable in all cases, Table 5 also reveals that overlapping modes significantly increase it, as it adds further complexity to the already complex case of resource-constrained scheduling problem, which is known to be a NP-hard optimization problem [20].

When resource constraints are considered, overlapping is less performed than without resource constraints. As expected, overlapping lead to additional workload and to more resource consumptions. Overlapping is thus less attractive and only one third of the set of overlappable activities are overlapped with resource constraints and overlapping modes (case 4). The overlapping modes obtained with the optimal makespan are detailed in Table 6. This confirms that overlapping and resource constraints are closely interrelated.

Table 6: Overlapping modes obtained with the optimal makespan

	without resource constraints (case 3)				with resource constraints (case 4)			
Overlapped couples or triplets of activities*	Mode	Overlap duration	Rework	Local Gain	Mode	Overlap duration	Rework	Local Gain
(3,8)	3	3	1	2	3	3	1	2
(4,10)	4	4	2	2	3	3	0	3
(2,11)	4	5	2	3	1	0	0	0
(10,16)	4	4	1	3	1	0	0	0
(13,18)	3	3	1	2	3	3	1	2
(16,17,22)	3	3	1	2	3	3	1	2

*The non-displayed couples or triplets of activities are not overlapped (i.e., executed in activity mode 1).

4 Conclusion and Discussion

Overlapping activities is one of the most applied strategies to accelerate a project either in its early stage when the schedule baseline is set up or following project delay during its execution. Overlapping is inherently risky as it entails that downstream activities start before the information they require is available in a finalized form. However, additional workload required to accommodate the information changes transmitted by upstream activities to the overlapped downstream activities are often ignored in practice. On the other hand, in spite of all research efforts accomplished in evaluating the relation between rework and the amount of overlap and determining the optimal overlapping strategy for two activities without resource constraints [6][7][15][16][17], only few papers have incorporated overlapping in the RCPSP [12][13]. In addition, these papers studied simplified linear rework model that are not realistic.

We investigate the joint optimization of overlapping and resource-constrained project scheduling problem with the following assumptions: (1) preliminary information can be exchanged between identified overlappable activities, (2) the information flow is unidirectional from upstream to downstream activities, (3) information exchanges are costless, (4) overlapping is restricted to a finite number of feasible amounts of overlap for each couple of activities, corresponding to overlapping modes, and (5) rework is preliminary estimated for each overlapping mode. The main contribution of this paper is to present an integer linear programming model for the project scheduling problem. Such a formulation shares similarities with the traditional multi-mode resource constraint scheduling problem (MRCPSP). Considering the limit of exact solution procedure encountered with MRCSPSP, we can anticipate that solving the RCPSP with overlapping modes for larger projects, as they usually appears in practical cases, will require the use of metaheuristics or heuristics. The relaxation of the aforementioned assumptions also represents interesting perspectives.

5 References

[1] Hartmann, S. Project scheduling under limited resources. Berlin: Springer, 1999.
[2] Hartmann, S. and Briskorn, D. A survey of variants and extensions of the resource-constrained project scheduling problem. European Journal of Operational Research, 207(1):1–14, 2010.
[3] Bogus, S. M., Molenaar, K. R. and Diekmann, J.E. Concurrent engineering approach to reducing design delivery time. Journal of Construction Engineering and Management, 131(11):1179–1185, 2005.
[4] Terwiesch, C. and Loch, C.H. Measuring the effectiveness of overlapping development activities. Management Science, 45(4): 455–465, 1999.
[5] Smith, P. and Reinertsen, D. Developing Products in Half the Time. New York: Van Nostrand Reinhold, 1995.

[6] Lin, J., Chai, K. H., Brombacher, A.C. and Wong, Y.S. Optimal overlapping and functional interaction in product development. European Journal of Operational Research, 196(3):1158–1169, 2009.
[7] Krishnan, V., Eppinger, S. D. and Whitney, D.E. A model-based framework to overlap product development activities. Management Science, 43(4):437–451, 1997.
[8] Krishnan, V. Managing the simultaneous execution of coupled phases inconcurrent product development. IEEE Transactions on Engineering Management, 43(2):210–217, 1996.
[9] Bogus, S.M., Molenaar, K.R. and Diekmann, J.E. Strategies for overlapping dependent design activities. Construction Management and Economics, 24:829-837, 2006.
[10] Steward, D.V.. The design structure system: a method for managing the design of complex systems. IEEE Transactions on Engineering Management, 28:71–74, 1981.
[11] Wang, J. T. and Lin, Y. I. An overlapping process model to assess schedule risk for new product development. Computers & Industrial Engineering, 57(2):460–474.
[12] Cho, S.H. and Eppinger, S.D. A simulation-based process model for managing complex design projects. IEEE Transactions on engineering management, 52(3): 316–328, 2005.
[13] Gerk, J. E. V. and Qassim, R. Y. Project acceleration via activity crashing, overlapping, and substitution." IEEE Transactions on Engineering Management, 55(4):590–601, 2008.
[14] Browning, T.R. Applying the design structure matrix to system decomposition and integration problems: a review and new directions. IEEE Transactions on engineering management, 48(3):292–306, 2001.
[15] Loch, C.H. and Terwiesch, C. Communication and uncertainty in concurrent engineering. Management Science, 44(8):1032–1048, 1998.
[16] Roemer, T.A., Ahmadi, R. And Wang, R.H. Time-cost trade-offs in overlapped product development. Operations Research, 48(6):858–865, 2000.
[17] Lin, J., Qian, Y., Cui, W. and Miao, Z. Overlapping and communication policies in product development. European Journal of Operational Research, 201(3):737–750, 2010.
[18] Browning, T.R. and Eppinger, S.D. Modeling impacts of process architecture on cost and schedule risk in product development. IEEE Transactions on engineering management, 49(4):428–442, 2002.
[19] Demeulemeester, E., De Reyck, B. and Herroelen, W. The discrete tim/resource trade-off problem in project networks: a branch-and-bound approach. IIE Transactions, 32:1059–1069, 2000.
[20] Herroelen, W. Project scheduling – theory and practice. Production and Operations Management, 14(4):413–432, 2005.
[21] Kolisch, R. and Sprecher, A. PSPLIB – A project scheduling problem library. European Journal of Operational Research, 96(1):205–216, 1996.
[22] Pellerin, R., Sadr, J., Guevremont, M. and Rousseau, L.M. Planning of international projects with material constraints. International Conference on Industrial Engineering and Systems Management 2009, Montréal, Canada.