Noname manuscript No.
(will be inserted by the editor)

The Resource-Constrained Activity Insertion Problem with Minimum and Maximum Time Lags

Christian Artigues · Cyril Briand

Received: date / Accepted: date

Abstract We consider the resource-constrained activity insertion problem with minimum and maximum time lags. The problem consists in inserting a single activity in a partial schedule while preserving its structure represented through resource flow networks and minimizing the makespan increase caused by the insertion. In the general case, we show that finding a feasible insertion that minimizes the project duration is NP-hard. When only minimum time lags are considered and when activity durations are strictly positive, we show the problem is polynomially solvable, generalizing previously established results on activity insertion for the standard resource-constrained project scheduling problem.

Keywords esource-constrained project scheduling · minimum and maximum time lags · activity insertion problem · complexity

1 Introduction

In the standard resource-constrained project scheduling problem (RCPSP), the precedence relations are simple: an activity cannot start before the end of all its predecessors. In other words, between the start time of an activity A_i and the start time of its successor A_j there is a minimal distance (or minimum time lag) equal to the duration of A_i. The RCPSP with minimum and maximum time lags (RCPSP/max) involves generalized precedence relations where the minimum time lag between A_i and A_j can be any non-negative value and where there is possibly a maximum allowed time lag between the start time of A_i and the start time of A_j.

Independently of makespan minimization, simply finding a resource-feasible schedule that respects both the minimum and maximum time lags is NP-hard [3] whereas, for the standard RCPSP, this problem is polynomial. For that reason, the RCPSP/max problem has been extensively studied in the scheduling literature. For a thorough analysis of this problem, we recommend the book [12]. In this paper, unlike most approaches

C. Artigues · C. Briand
LAAS-CNRS, Université de Toulouse, 7 avenue du Colonel Roche, 31077 Toulouse, France
E-mail: artigues,briand@laas.fr

that focused on finding a global schedule that minimizes the project duration, we illustrate the difficulties brought by the maximum time lags by considering the apparently simple problem of inserting a single activity inside an existing schedule.

This problem aims at finding an insertion satisfying the mininum/maximum time lags and resource constraints and preserving the partial schedule structure. The objective is to minimize the project duration increase. Such a problem arises in local search procedures for reinsertion neighborhoods that unschedule an activity and insert it at another position [2,4,13]. For such procedures finding a compromise between the quality of the reinsertion operator and its computational requirements is a fundamental issue. Since the last decades, there as been a growing interest in the predictive-reactive scheduling framework to take account of uncertainty in scheduling. We refer to recent surveys in the context of manufacturing systems [14] and project management [15]. Inserting efficiently unexpected activities in an existing schedule is among the central issues raised by predictive-reactive scheduling reactive [2,1,5,8].

The interest of preserving the structure of the partial schedule is twofold. First, it makes it possible to decrease the size of the search space, which is necessary for both neighborhood search and reactive scheduling. Second, this policy tends to minimize the disturbances during the on-line project execution phase, which is referred to as ensuring the schedule stability.

In [4,13] among others, operation insertion problems are solved in a generalized job-shop. In [9], the job insertion problem in job-shop scheduling is studied. In [8] and [10], insertion problems in a general disjunctive scheduling framework capturing a variety of job shop scheduling problems and insertion types are considered. A polyedral characterization is provided and lower and upper bounding procedures are developed. Since disjunctive and job-shop scheduling are particular cases of the RCPSP where all resources have unit availability, the proposed procedures cannot be used to solve the problem considered in this paper.

For the RCPSP and the RCPSP/max, we can define the structure of the schedule by means of resource flow networks, which induce additional precedence constraints preventing resource over-subscription. With such a structure, the activity insertion problem has been addressed already for the standard RCPSP in [1,2]. An $O(n^2m)$ optimal insertion procedure has been proposed where n is the number of activities and m is the number of resources. In [1], a tabu search procedure based on the reinsertion of a critical activity in the current resource flow network has been designed and obtained good results to solve the standard RCPSP. Hence, an important issue is to establish whether this approach can be extended to minimum and maximum time lags for proposing a new class of neighborhood search method for the RCPSP/max.

In this paper, we show that the considered insertion problem is unfortunately NP-hard in the general case. However, when maximum time lags are ignored, we propose a new polynomial algorithm, generalizing the results obtained in [1,2] to the RCPSP with minimum time lags.

The resource constrained activity insertion problem is presented in Section 2. The concept of feasible insertion is introduced in Section 3. Feasibility conditions are described in Section 4. The problem complexity is given in Section 5. The particular case where only minimum time lags are considered is studied in Section 6. Concluding remarks are drawn in Section 7.

2 Problem statement

Activities constituting the project are identified by set $V = \{A_0, \ldots, A_{n+1}\}$. Activity A_0 represents by convention the start of the schedule and activity A_{n+1} represents symmetrically the end of the schedule. The set of non-dummy activities is identified by $\mathcal{A} = \{A_1, \ldots, A_n\}$. p_i denotes the duration of activity i with $p_0 = p_{n+1} = 0$. We assume in this paper that $p_i > 0, \forall A_i \in \mathcal{A}$.

A valuated activity-on-node graph $G(V, E, l)$ is defined where nodes correspond to activities and arcs correspond to precedence relations. Each arc $(A_i, A_j) \in E$ is valuated by an integer time lag l_{ij}. $l_{ij} \geq 0$ corresponds to a minimum time lag of l_{ij} units stating that A_j has to start at least l_{ij} time units after the start time of A_i. In the standard RCPSP, only minimum time lags are considered and $l_{ij} = p_i$ for each arc $(A_i, A_j) \in E$. $l_{ij} \leq 0$ corresponds to a time lag of $-l_{ij}$ units stating that A_i has to start at the latest l_{ij} time units after the start time of A_j. We assume that G is strongly connected and that the longest path from A_0 to any activity A_i, $i \neq 0$ is non negative while the longest path from any activity A_i, $i \neq n+1$ to A_{n+1} is greater than of equal to p_i. Resource constraints are defined as in the standard RCPSP: $\mathcal{R} = \{R_1, \ldots, R_m\}$ denotes the set of m resources. B_k denotes the availability of R_k. $b_{i,k}$ represents the amount of resource R_k used during the execution of i.

A solution S is a schedule giving the start time S_i of each activity A_i. The makespan of a schedule S is equal to S_{n+1}, the start time of the end activity. The RCPSP with minimum and maximum time lags is the problem (P) of finding a non-preemptive schedule S of minimal makespan S_{n+1} subject to precedence and resource constraints.

$$\min S_{n+1}$$
$$S_j - S_i \geq l_{ij} \quad \forall (A_i, A_j) \in E$$
$$\sum_{A_i \in \mathcal{A}_t} b_{i,k} \leq B_k \quad \forall R_k \in \mathcal{R}, \forall t \geq 0$$

By convention we set $S_0 = 0$ as time origin. A schedule S is feasible if it satisfies the generalized precedence constraints 2 and the resource constraints 2 where $\mathcal{A}_t = \{i \in \mathcal{A} \,|S_i \leq t < S_i + p_i\}$ represents the set of non-dummy activities in process at time t. Note that an activity cannot be interrupted once it is started, i.e. preemption is not allowed.

(P) does not necessarily have a solution and, unfortunately, deciding whether (P) has a solution is NP-complete [3].

Infeasibility can be due to an inconsistency of the precedence constraints E, independently of the resource constraints, which is easy to detect. Indeed, there is no solution to (P) if there is a cycle of positive length in $G(V, E, l)$. The Floyd-Warshall (FW) algorithm can be used to detect in $\mathcal{O}(n^3)$ time such a cycle. In the remaining of the paper we consider only problems for which the temporal constraints are consistent. Moreover, a (tentative) upper bound UB on the makespan is available where $l_{(n+1)0} = -UB$.

In case there is no cycle of positive length in $G(V, E, l)$ and since G is strongly connected, a distance matrix $(\delta_{i,j})_{i,j \in V^2}$ can be defined where $\delta_{i,j}$ is the length of the longest path between i and j in $G(V, E, l)$. $(\delta_{i,j})$ values are returned by FW. Given, the distance matrix δ, (P) can be written alternatively:

$$\min S_{n+1}$$

$$S_j - S_i \geq \delta_{i,j} \quad \forall (A_i, A_j) \in V^2, i \neq j$$

$$\sum_{A_i \in \mathcal{A}_t} b_{i,k} \leq B_k \quad \forall R_k \in \mathcal{R}, \forall t \geq 0$$

An instance of (P) is defined by tuple (n, m, p, δ, b, B).

We use the concept of resource-flow network to represent the solutions to (P) [1,2, 6,11,12]. A resource flow f is a $(n+2) \times (n+2) \times m$ matrix verifying equations (1-3) defined below:

$$f_{i,j,k} \geq 0 \quad \forall A_i \in \mathcal{A} \cup \{A_0\}, \forall A_j \in \mathcal{A} \cup \{A_{n+1}\}, \forall R_k \in \mathcal{R} \tag{1}$$

$$\sum_{A_i \in \mathcal{A} \cup \{A_{n+1}\}} f_{0,i,k} = \sum_{A_i \in \mathcal{A} \cup \{A_0\}} f_{i,n+1,k} = B_k \quad \forall R_k \in \mathcal{R} \tag{2}$$

$$\sum_{A_j \in \mathcal{A} \cup \{A_{n+1}\}} f_{i,j,k} = \sum_{A_j \in \mathcal{A} \cup \{A_0\}} f_{j,i,k} = b_{i,k} \quad \forall A_i \in \mathcal{A}, \forall R_k \in \mathcal{R} \tag{3}$$

$f_{i,j,k}$ denotes the number of resource R_k units transferred from activity A_i to activity A_j.

From flow f, a set of precedence constraints (or arcs) $F(f)$ can be defined as

$$F(f) = \{(A_i, A_j) \in V^2 | \exists R_k \in \mathcal{R}, f_{i,j,k} > 0\}$$

$F(f)$ is the set of precedence constraints (or arcs) induced by flow f. We say that A_j is a resource successor of A_i and A_i is a resource predecessor of A_j if $(A_i, A_j) \in F(f)$.

From arc set $F(f)$, we define the graph induced by flow f as

$$\mathcal{G}(f) = G\left(V, E \cup F(f), L(f)\right)$$

where arc valuation $L(f) : E \cup F(f) \rightarrow \mathbb{N}$ is a mapping defined by extending valuation l as follows.

$$L_{ij}(f) = \begin{cases} l_{ij} & \text{if } (A_i, A_j) \in E, (A_i, A_j) \notin F(f) \\ p_i & \text{if } (A_i, A_j) \in F(f), (A_i, A_j) \notin E \\ \max(p_i, l_{ij}) & \text{if } (A_i, A_j) \in E \cap F(f) \end{cases}$$

Now, let $\left(\Delta_{i,j}(f)\right)_{A_i, A_j \in V}$ be defined as the distance matrix induced with flow f. $\Delta_{i,j}(f)$ is the length of the longest path from A_i to A_j in $\mathcal{G}(f)$. The precedence constraints $F(f)$ induced by the flow are consistent if and only if there is no cycle of positive length in $\mathcal{G}(f)$. In particular, $\Delta_{0,i}(f)$, $\forall A_i \in V$ denotes the earliest start schedule associated with the flow f.

Proposition 1 *(P) can be defined as the problem of finding a resource flow f verifying (1-3), such that $\mathcal{G}(f)$ contains no cycle of positive length and such that $\Delta_{0,n+1}(f)$ is minimal.*

Proof The arguments used in Theorem 1 and 2 in [11] hold. See also Section 2.13 in [12]. □

Hence, a flow f verifying (1-3), such that $\mathcal{G}(f)$ contains no cycle of positive length is said to be feasible and is considered in what follows as a solution to (P) giving feasible schedule $\left(\Delta_{0,i}(f)\right)_{A_i \in V}$.

To define the resource constrained activity insertion problem (RCAIP), we consider a partial solution flow f in which all activities but one, denoted by activity A_x $(0 < x < n+1)$, have been scheduled. This amounts to considering a complete solution f to problem (P_{-x}) identical to (P) except that $b_{x,k} = 0$, $\forall R_k \in \mathcal{R}$. The RCAIP amounts to compute a solution to (P) by inserting activity A_x in the flow f associated with the solution of (P_{-x}) in such a way that the resource flow assigned to A_x can only be rerouted from f. We give a formal definition of a rerouted flow.

Definition 1 A flow f' is rerouted from a flow f for an activity A_x if there exists values $q_{i,j,k}$, for all $i, j \in V$ and for all $R_k \in \mathcal{R}$ such that $0 \leq q_{i,j,k} \leq f_{i,j,k}$ and

$$f'^{k}_{xj} = \sum_{A_i \in V} q_{i,j,k} \qquad \forall A_j \in V, \forall R_k \in \mathcal{R} \tag{4}$$

$$f'^{k}_{ix} = \sum_{A_j \in V} q_{i,j,k} \qquad \forall A_i \in V, \forall R_k \in \mathcal{R} \tag{5}$$

$$f'^{k}_{ij} = f_{i,j,k} - q_{i,j,k} \qquad \forall A_i, A_j \in V, \forall R_k \in \mathcal{R} \tag{6}$$

From this definition, we define the *resource-constrained activity insertion problem* (P_x).

Definition 2 (RCAIP) Given a solution flow f to problem (P_{-x}), the resource-constrained activity insertion problem (P_x) amounts to find a rerouted flow f' for A_x such that f' is a solution flow to (P) and $\Delta_{0,n+1}(f')$ is minimized.

We have $\Delta_{0,n+1}(f') \geq \Delta_{0,n+1}(f)$ since by transitivity, any precedence constraint induced by f is also induced by f'. Note that an instance of RCAIP is given by tuple $(n, m, p, \delta, b, B, x, f)$.

Consider an example issued from [12], comprising five real activities and a single resource of three units . Durations and resource demands are given in Table 1. Minimum and maximum time lags and the corresponding the project network are displayed on the left side of Figure 1.

Table 1 Activity durations and resource demands

A_i	0	1	2	3	4	5	6
p_i	0	6	4	2	4	2	0
b_i	0	1	2	2	2	3	0

We consider a partial schedule for (P) in which activity A_1 is not scheduled, corresponding to a complete solution of (P_{-1}), displayed in Figure 2. The resource flow corresponding to this schedule is displayed on the right side of Figure 1. Thin arcs correspond to the original precedence constraints only while thick arcs are induced by the flow (the flow value is displayed between braces). Plain thick arcs belong to $E \cup F(f)$ while thick dotted arcs are only induced by the flow. In flow f, solution to (P_{-1}), presented in Figure 1, the non-zero flow amounts are $f_{0,3} = 2$, $f_{0,5} = 1$, $f_{3,4} = 2$,

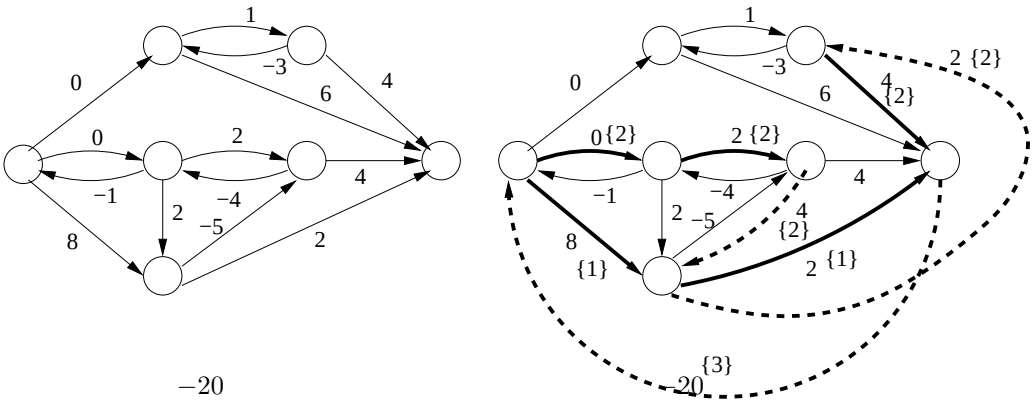

Fig. 1 Minimum/maximum time lags (left side) and flow f solution to (P_{-1}) (right side)

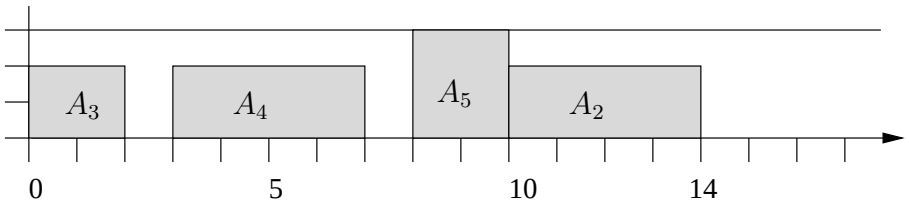

Fig. 2 Gantt chart of the solution to (P_{-1})

$f_{4,5} = 2$, $f_{5,6} = 1$, $f_{5,2} = 2$, $f_{2,6} = 2$. The activity candidate for insertion A_1 is such that $b_i = 1$. Hence, to insert A_1 into f according to the insertion scheme described above, we have to find activities A_i and A_j such that the amount of flow taken from $f_{i,j}$ and rerouted to A_1 is $q_{i,j} = 1$. The reader may check that among the non zero flow amounts listed above, the only valid candidates are $A_i = A_5$ and $A_j = A_6$. Indeed, inserting A_1 before activity A_5 or between A_2 and A_5 violates constraint $S_2 \leq S_1 + 3$ while inserting A_1 after A_2 violates constraint $S_2 \geq S_1 + 1$. It follows that we obtain the solution to P_1 displayed in Figure 3 of makespan 16.

Fig. 3 Gantt chart of the solution to P_1

3 Feasible insertions

Since the original flow f, feasible for problem (P_{-x}), is an input data of insertion problem P_x, we will denote $\Delta_{i,j}(f)$, $\mathcal{G}(f)$, $F(f)$ and $L_{ij}(f)$ as $\Delta_{i,j}$, $\mathcal{G}$, F and L_{ij}, respectively.

We define an insertion as an ordered pair of disjoint activity sets (α, β) such that $(\alpha, \beta) \in V \setminus \{A_x, A_{n+1}\} \times V \setminus \{A_0, A_x\}$, $\alpha \cap \beta = \emptyset$.

An insertion induces a set of arcs

$$F(\alpha, \beta) = \{(A_a, A_x)\}_{A_a \in \alpha} \cup \{(A_x, A_b)\}_{A_b \in \beta}.$$

We consider graph

$$\mathcal{G}(\alpha, \beta) = G\left(V, E \cup F \cup F(\alpha, \beta), L(\alpha, \beta)\right)$$

where $L(\alpha, \beta)$ denotes the arc valuation induced by (α, β) and is defined as follows:

$$L_{ij}(\alpha, \beta) = \begin{cases} p_i & \text{if } A_i \in \alpha, A_j = A_x, (A_i, A_x) \notin E \\ \max(p_i, l_{ix}) & \text{if } A_i \in \alpha, A_j = A_x, (A_i, A_x) \in E \\ p_x & \text{if } A_i = A_x, A_j \in \beta, (A_x, A_j) \notin E \\ \max(p_x, l_{xj}) & \text{if } A_i = A_x, A_j \in \beta, (A_x, A_j) \in E \\ L_{ij} & \text{if } A_i \in V \setminus \alpha, A_j \in V \setminus \beta \end{cases}$$

Let $\Delta_{i,j}(\alpha, \beta)$ denotes the length of the longest path from A_i to A_j in $\mathcal{G}(\alpha, \beta)$. Let $Q_k(\alpha, \beta)$ denote the amount of resource k units available for insertion in (α, β). Vector $Q(\alpha, \beta)$ is called the insertion capacity. We have

$$Q_k(\alpha, \beta) = \sum_{A_i \in \alpha, A_j \in \beta} f_{i,j,k}$$

We claim that an insertion (α, β) verifying

$$Q_k(\alpha, \beta) \geq b_{x,k}, \forall R_k \in \mathcal{R} \tag{7}$$

and such that there is no positive length cycle in $\mathcal{G}(\alpha, \beta)$ is feasible in the sense it yields a feasible solution f' to (P_x).

Consider Algorithm 1 (GENERATEFLOWFROMINSERTION) which compute flow f' by rerouting the flow sent from α to β. The algorithm runs in $O(n^2 m)$. Q_k denote at each inner step the total amount of flow already rerouted to A_x. Flow f' is feasible for (P_x) if (α, β) verifies (7) and if $\mathcal{G}(f')$ does not contain any positive length cycle. For the latter consition, we have $F(f') \subseteq F(\alpha, \beta) \cup F$. hence if $\mathcal{G}(\alpha, \beta)$ does not have any positive length cycle, $\mathcal{G}(f')$ does not have any positive length cycle neither and $\Delta_{0,n+1}(f') \leq \Delta_{0,n+1}(\alpha, \beta)$.

Furthermore, given a flow f', solution to P_x rerouted from f through resource amounts $q_{i,j,k}$, an insertion (α, β) can be defined by setting $\alpha = \{A_i \in V | \exists A_j \in V, \exists R_k \in \mathcal{R}, q_{i,j,k} > 0\}$ and $\beta = \{A_i \in V | \exists A_j \in V, \exists R_k \in \mathcal{R}, q_{j,i,k} > 0\}$. (α, β) is such that $\mathcal{G}(\alpha, \beta) = \mathcal{G}(f')$ and $\Delta_{0,n+1}(f') \leq \Delta_{0,n+1}(\alpha, \beta)$.

From this reduction, we can restrict the search space to pairs of activity sets (α, β) without considering explicitly amounts of rerouted flow $q_{i,j,k}$. It follows a simplified definition of the RCAIP.

Algorithm 1 generateFlowFromInsertion(α, β) Compute flow f' from insertion (α, β)

$q_{i,j,k} \leftarrow 0$, for all $A_i, A_j \in V$, for all $R_k \in \mathcal{R}$.
$Q_k \leftarrow 0$, for all $R_k \in \mathcal{R}$
for $A_i \in \alpha$ **do**
  for $A_j \in \beta$ **do**
    for $R_k \in \mathcal{R}$ **do**
      $q_{i,j,k} \leftarrow \max(0, b_{x,k} - Q_k)$
      $Q_k \leftarrow Q_k + q_{i,j,k}$
    end for
  end for
end for
Compute f' from f and q with equations (4-6)

Definition 3 (RCAIP) Given a solution flow f to problem (P_{-x}), the resource-constrained activity insertion problem (P_x) amounts to find an insertion (ordered pair of activity sets) (α, β) verifying capacity condition (7), such that $\mathcal{G}(\alpha, \beta)$ does not include a positive length cycle and such that $\Delta_{0,n+1}(\alpha, \beta)$ is minimized.

Algorithm 1 shows that, given a feasible insertion (α, β), only a subset of α may actually sends flow to A_x after insertion. Hence α is called the set of possible resource successors. Symetrically, only a subset of β may received flow from A_x and we call β the set of possible resource successors. In fact, when we call an insertion feasible it will issue a feasible flow for the range of possible values $b_{x,k} \in \{0, \ldots, Q_k(\alpha, \beta)\}$, for all $R_k \in \mathcal{R}$. This feature may be of interest in presence of uncertainty on the resource demand of the inserted activity.

4 Insertion feasibility conditions

In this section we establish feasibility conditions of a given insertion (α, β).

The decision variant of (P_x) is considered. The problem is to decide whether a feasible solution with a makespan not greater than v exists. For any instance I of the RCAIP (P_x), the problem is equivalent to deciding whether there is a feasible solution to (P_x) with $l_{(n+1)0} = -v$. Deciding whether an instance I of P_x has a solution amounts to deciding whether there exist a feasible insertion (α, β). If $v < \Delta_{0,n+1}$, there is already a positive length cycle in $\mathcal{G}$ and, consequently, the answer is no. In the remaining of the section, we assume $v \geq \Delta_{0,n+1}$.

Since only arcs from $F(\alpha, \beta)$ are added by the insertion, there is only three disjoint sets of elementary cycles that can appear in $\mathcal{G}(\alpha, \beta)$, all involving activity A_x.

- An elementary cycle belonging to set $\mathcal{C}_1(\alpha, \beta)$ involves an arc (A_i, A_x) with $A_i \in \alpha$ and $L_{ix}(\alpha, \beta) = p_i$ and no arc (A_x, A_j) with $A_j \in \beta$. Note that in $\mathcal{G}(\alpha, \beta)$, a longest path from A_x to A_i which is not traversing any node of β is also a longest path from A_x to A_i in $\mathcal{G}$ of length $\Delta_{x,i}$ (since the path cannot traverse A_x again). Therefore, if $\mathcal{L}_1(\alpha, \beta)$ denotes the length of the longest cycles in $\mathcal{C}_1(v, \alpha, \beta)$, we have

$$\mathcal{L}_1(\alpha, \beta) = \max_{A_i \in \alpha} (p_i + \Delta_{x,i}).$$

- An elementary cycle belonging to set $\mathcal{C}_2(\alpha,\beta)$ includes an arc (A_x, A_j) with $A_j \in \beta$ and $L_{xj}(\alpha,\beta) = p_x$ and no arc (A_i, A_x) with $A_i \in \alpha$. Using the same arguments as in the previous case, the length of the longest path from A_j to A_x in $\mathcal{G}(\alpha\beta)$ is equal to $\Delta_{j,x}$. Consequently, the length of the longest cycle in $\mathcal{C}_2(\alpha,\beta)$ is

$$\mathcal{L}_2(\alpha,\beta) = p_x + \max_{A_j \in \beta} \Delta_{j,x}.$$

- Last, an elementary cycle belonging to set $\mathcal{C}_3(\alpha,\beta)$ involves two arcs $(A_i, A_x), (A_x, A_j)$ with $L_{ix}(\alpha,\beta) = p_i$ and $L_{xj}(\alpha,\beta) = p_x$. Again, the longest path in $\mathcal{G}(\alpha,\beta)$ from j to i passes only through arcs existing in $\mathcal{G}$ and therefore is of length $\Delta_{j,i}$. Hence, the length of the longest cycle in $\mathcal{C}_3(\alpha,\beta)$ is

$$\mathcal{L}_3(\alpha,\beta) = p_x + \max_{(A_i,A_j)\in\alpha\times\beta} (p_i + \Delta_{j,i}).$$

In what follows, $\mathcal{C}_q(v,\alpha,\beta)$ and $\mathcal{L}_q(v,\alpha,\beta)$ $(q = 1,2,3)$ will be simply denoted $\mathcal{C}_q$ and $\mathcal{L}_q$ when there is no ambiguity. Figure 4 illustrates the three considered cycle types.

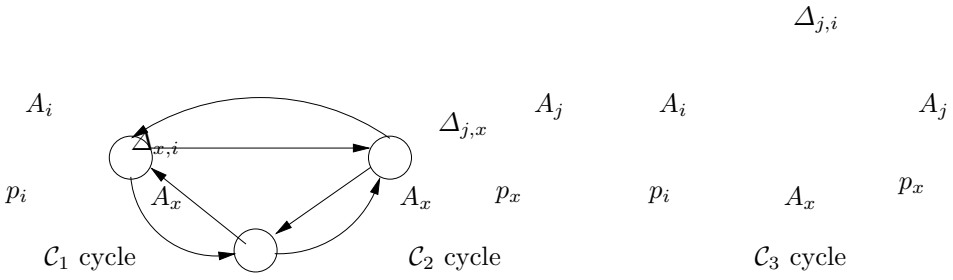

Fig. 4 The three cycle types generated by the insertion

The feasibility condition for an insertion (α,β) follows throm the definition of the three possible cycle sets.

Proposition 2 *An insertion (α,β) is feasible for (P_x) if and only if it verifies (7) and*

$$\max_{q=1,2,3} \mathcal{L}_q(\alpha,\beta) \leq 0 \tag{8}$$

5 Computational complexity

There have been previously established complexity results related to insertion. The activity insertion problem has already been addressed for the standard RCPSP in [1, 2]. The problem is polynomial and an $O(n^2m)$ optimal insertion procedure has been proposed by the authors. In [9], the authors study the job insertion problem in the standard job-shop scheduling problem and show the problem is NP-hard. In [8] and [10], a general insertion problem in disjunctive scheduling is defined. In disjunctive scheduling, resource constraints are defined through a set of disjunctive arcs $e = \{i,j\}$ such that, in any feasible solution called a selection, each disjunctive arc must be orientated yielding arc (i,j) or arc (j,i). An insertion problem corresponds to a partial orientation (selection) of the disjunctive arcs and an insertion corresponds to a complete

orientation (selection). Among other results, the authors show that when the insertion disjunctive graph has specific property, the insertion problem becomes solvable in polynomial time, in particular because feasible insertions correspond to independent sets in a bipartite graph. A disjunctive arc defines a set of activities that cannot be scheduled in parallel, i.e. a forbidden set of cardinality 2. The fundamental difference with the work of [9], [8] and [10] is that we cannot use the disjunctive graph formulation to represent our problem since in the RCPSP, the resource constraints define forbidden sets of cardinality that can be larger than 2. Hence, while insertion of single activities are generally easy in disjunctive graph models the following results show that this is not the case for our problem. Indeed, we establish the link between feasible insertions and independent sets in general graphs.

In what follows, we prove that the RCAIP is NP-hard. Hereafter, we define the decision variant of the RCAIP.

Definition 4 (RCAIP - decision variant) Given a solution flow f to problem (P_{-x}) and an integer v, does there exist an insertion (α, β) verifying conditions (7-8) and such that $\Delta_{0,n+1}(\alpha, \beta) \leq v$?

We show that the decision problem is NP-complete by reduction from the independent set problem, defined below.

Definition 5 (INDEPENDENT SET - decision variant) Given a graph $G(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of vertices and $\mathcal{E}$ is a set of edges, and an integer k, does there exist a subset of vertices $\mathcal{W} \subseteq \mathcal{V}$ of cardinality k such that $\forall x, y \in \mathcal{W}$, $(x, y) \notin \mathcal{E}$ (i.e. the vertices of $\mathcal{W}$ are pairwise independent) ?

Such a problem is known to be NP-complete [7]. Let p denotes the number of vertices: $p = |\mathcal{V}|$.

Theorem 1 *The decision variant of the RCAIP is NP-complete.*

Proof The problem is in NP since (i) for any tentative solution (α, β) checking whether constraints (7) are satisfied can be made in $O(n^2 m)$ time since $|\alpha| \leq n$ and $|\beta| \leq n$ and (ii) checking whether there is no positive length cycles in $\mathcal{C}_1$, $\mathcal{C}_2$ and $\mathcal{C}_3$ take $O(n)$, $O(n)$ and $O(n^2)$ time, respectively. Initial Matrix $(\Delta_{i,j})$ can be computed in $O(n^3)$ by the Floyd-Warshall algorithm.

In what follows, we show how to associate to any independent set problem an instance of the insertion problem with $n = 2p + 1$ non-dummy activities and a single resource of availability $B_1 = p$ (the resource index will be omitted in the rest of the demonstration).

The non-dummy activities are partitioned into three subsets $\Theta = \{A_1, \ldots, A_p\}$, $\Omega = \{A_{p+1}, \ldots, A_{2p}\}$ and $\{A_n\}$ with $A_n = A_{2p+1}$ being the activity to insert. All activities but A_n are such that $b_i = p_i = 1$, $\forall A_i \in \Theta \cup \Omega$. For A_n, we set $p_n = 3$ and $b_n = k$. The resource flow f is such that

$$\begin{aligned} f_{0,i} &= 1 \quad \forall A_i \in \Theta \\ f_{i,p+i} &= 1 \quad \forall A_i \in \Theta \\ f_{j,n+1} &= 1 \quad \forall A_j \in \Omega. \end{aligned}$$

All other resource flows are zero. The precedence constraints E are such that there is an arc (A_0, A_i) with $l_{0i} = 0$, for each $A_i \in \Theta$. There is an arc (A_j, A_{n+1}) with

$l_{j(n+1)} = 1$ for each $j \in \Omega$. There is an arc (A_i, A_{p+i}) with $l_{i(p+i)} = 2$ for each $A_i \in \Theta$. There is an arc (A_{n+1}, A_0) such that $l_{(n+1)0} = -5$ (i.e. the makespan has to be not greater than 5). For the inserted task A_n, there is an arc (A_0, A_n) and an arc (A_n, A_{n+1}) with $l_{0n} = 0$ and $l_{n(n+1)} = 3$, respectively.

The set of edges $\mathcal{E}$ of the graph of the independent set problem is arbitrarily orientated. Let $G(\mathcal{V},\mathcal{U})$ denotes the graph so obtained. Now we create in E an arc (A_{p+j}, A_i) with $l_{(p+j)i} = -3$ for each $(i, j) \in \mathcal{U}$ to represent the edges in $\mathcal{E}$. Note that the earliest schedule resulting from f is optimal for (P_{-n}) since $\Delta_{0,n+1} = \delta_{0,n+1} = 3$.

Figure 5 shows how the insertion problem instance is built from an example graph of six nodes. Below each arc the time lag is indicated and the transferred resource flow is displayed between braces.

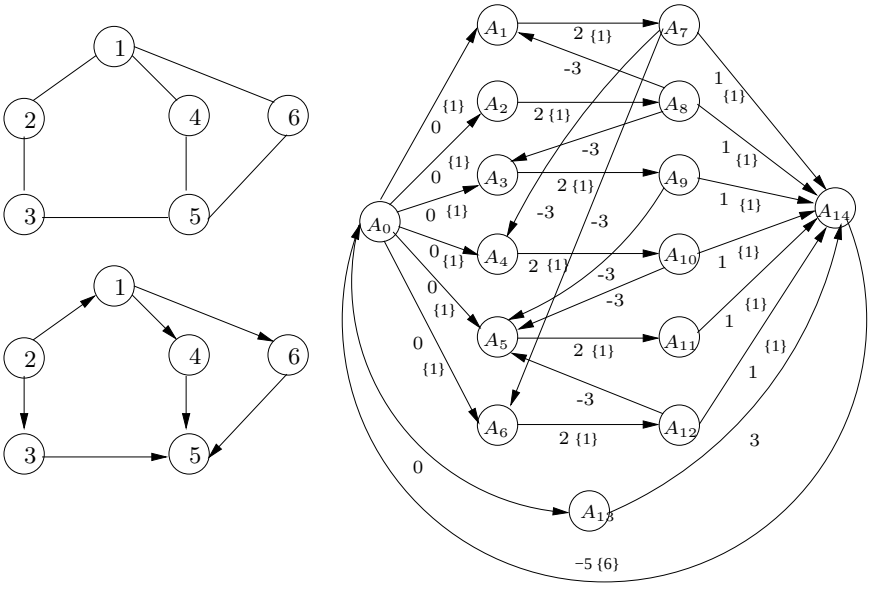

Fig. 5 Reduction from independent set

It can be easily checked that the minimal distance matrix Δ for the so-defined insertion problem instance (with $v = 5$) is such that for each pair A_i, A_j with $i \in \Theta$ and $j \in \Omega$, we have $\Delta_{j,i} = -3$ if $(j - p, i) \in \mathcal{U}$ and $\Delta_{j,i} = -4$ otherwise.

Below we show that there exists an insertion for A_n in $\mathcal{G}$ if and only if there exists an independent set of cardinality k in $G(\mathcal{V}, \mathcal{E})$. First, one can observe that if W is an independent set of cardinality k, then there is a feasible insertion (α, β). Indeed, if $\alpha = \{A_i | i \in W\}$ and $\beta = \{A_{p+i} | i \in W\}$ then, because the flow from $A_i \in \Theta$ to A_{i+p} is equal to 1, we have $\sum_{A_i \in \alpha, A_j \in \beta} f_{i,j} = k$. Consequently (α, β) verifies capacity condition (7). Moreover, there is no maximum time lags involving the inserted activity A_n and, consequently, no possible positive length cycles in $\mathcal{C}_1$ nor in $\mathcal{C}_2$. For any pair of nodes $x, y \in W$ we have no edge (x, y) in $\mathcal{E}$, therefore we have no arc $(x + p, y)$ in $\mathcal{U}$. As a result, $\forall A_j \in \beta$, $\forall A_i \in \alpha$, we have $\Delta_{j,i} = -4$. The longest cycle in $\mathcal{C}_3$ has a length equal to $\mathcal{L}_3(\alpha, \beta) = \max_{(A_i, A_j) \in \alpha \times \beta}(p_i + p_x + \Delta_{j,i})$. Since $p_i = 1$ and $p_x = 3$, we have $\mathcal{L}_3(\alpha, \beta) = 0$. It follows that the insertion problem is feasible.

Now let us show that if the insertion is feasible, there is an independent set of cardinality k. Let (α, β) denotes the feasible insertion. Any insertion such that $\alpha = \{A_0\}$ and $\beta \subseteq \Theta \neq \emptyset$ is not feasible since it will increase the project duration to 6, whereas it is bounded by 5. The same remark holds for the insertion such that $\alpha \subseteq \Omega$ and $\beta = \{A_{n+1}\}$. Therefore, A_n can only be located between some activities of Θ and some activities of Ω (i.e. $\alpha \subseteq \Theta$ and $\beta \subseteq \Omega$). For a feasible (α, β) insertion, since no positive length cycles can generated, we have $\Delta_{j,i} = -4\ \forall A_i \in \alpha, \forall A_j \in \beta$. Subsequently there is no arc $(j-p, i)$ in $\mathcal{U}$ and no edge $(j-p, i)$ in E, $\forall A_i \in \alpha$, $\forall A_j \in \beta$. Furthermore, since the insertion is feasible, we have $|\alpha| = k$ therefore α is an independent set of cardinality k. Figure 6 illustrates the correspondence between an independent set (bold nodes) and an insertion. □

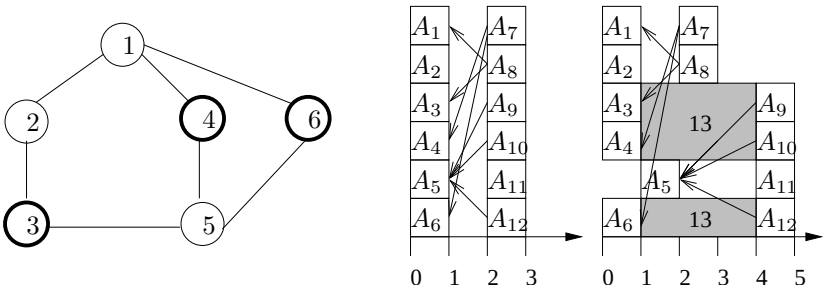

Fig. 6 A feasible independent set and its corresponding feasible insertion

While for the standard RCPSP, Artigues *et al.* proposed a polynomial algorithm to solve the insertion problem [1,2], the introduction of minimum and maximum time lags clearly makes the problem intractable despite of its apparent simplicity. This illustrates the deep modifications of the problem structure brought by the maximum time lags. To confirm this, we exhibit in the following section a polynomial algorithm, able to solve the insertion problem when only minimum time lags are considered and when activity durations are strictly positive.

6 RCAIP with minimum time lags only

We suppose in this section that all time lags are non-negative and that $p_i \geq 1, \forall A_i \in \mathcal{A}$. We still assume that f represents a feasible solution to (P$_{-x}$) of makespan $\Delta_{0,n+1}$ corresponding to the length of the longest path between 0 and $n+1$ in $\mathcal{G}$. Because of the absence of negative time lags, the graph $\mathcal{G}$ is no more fully connected. By convention, when there is no path from i to j, we have $\Delta_{i,j} = -\infty$.

We already know that the insertion problem is polynomial for the standard RCPSP as stated in [2]. The standard RCPSP corresponds to the case where all minimum time lags are such that $l_{ij} = p_i$. it follows that in the case where all minimum time lags are such that $l_{ij} \geq p_i$, we can trivially show that the problem is polynomially solvable by transforming the RCPSP instance with minimum time lags into a standard RCPSP instance as follows. For each $(A_i, A_j) \in E$, create an activity A_{ij} of duration $l_{ij} - p_i$ and zero resource demand and augment E with the precedence constraints (A_i, A_{ij}),

(A_{ij}, A_j). Then, set all arc values to the duration of the origin activity. Since the initial flow is trivially valid for the so-generated instance, the polynomial insertion algorithm for the standard RCPSP can be applied. Thus, the problem remains open only in the case there is at least one minimum time lag such that $0 \leq l_{ij} \leq p_i$. We show that in this case the insertion problem (P_x) remains polynomially solvable. We now consider the optimization variant, aiming at finding a minimal makespan feasible insertion (so we have $l_{(n+1)0} = -\infty$).

6.1 Makespan after insertion

Given, a feasible insertion (α, β), the makespan $\Delta_{0,n+1}(\alpha, \beta)$ after insertion in (α, β) is either equal to the makespan before insertion $\Delta_{0,n+1}$ or to the length of the longest path from A_0 to A_{n+1} in $\mathcal{G}(\alpha, \beta)$ traversing node A_x. This corresponds to three posssible sets of $(0, n+1)$ paths.

- The set of paths traversing an arc (A_i, A_x) with $(A_i \in \alpha)$ and no arc (A_x, A_j) with $(A_j \in \beta)$. The longest path of this set has a length equal to

$$\mathcal{M}_1(\alpha, \beta) = \max_{A_i \in \alpha}(\Delta_{0,i} + p_i) + \Delta_{x,n+1}$$

- The set of paths traversing an arc (A_x, A_j) with $(A_j \in \beta)$ and no arc (A_i, A_x) with $(A_i \in \alpha)$. The longest path of this set has a length equal to

$$\mathcal{M}_2(\alpha, \beta) = \Delta_{0,x} + p_x + \max_{A_j \in \beta}(\Delta_{j,n+1})$$

- The set of paths traversing an arc (A_i, A_x) with $(A_i \in \alpha)$ and an arc (A_x, A_j) with $(A_j \in \beta)$. The longest path of this set has a length equal to

$$\mathcal{M}_1(\alpha, \beta) = \max_{A_i \in \alpha}(\Delta_{0,i} + p_i) + p_x + \max_{A_j \in \beta}(\Delta_{j,n+1})$$

Figure 7 illustrates the 3 types of $(0, n+1)$ paths.

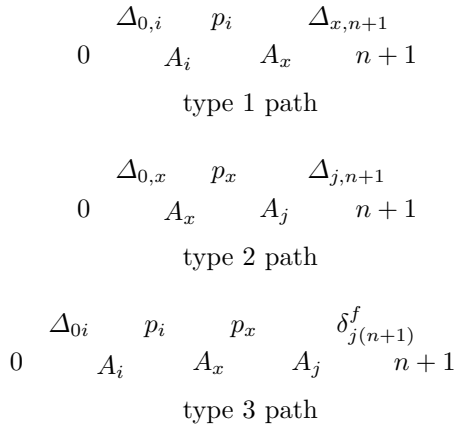

Fig. 7 The 3 types of $(0, n+1)$ paths generated by the insertion

It follows that the makespan of a feasible insertion (α, β) is equal to

$$\Delta_{0,n+1}(\alpha,\beta) = \max\left(\Delta_{0,n+1}, \mathcal{M}_1(\alpha,\beta), \mathcal{M}_2(\alpha,\beta), \mathcal{M}_3(\alpha,\beta)\right) \tag{9}$$

6.2 Feasibility conditions and feasible insertion

Below, we derive a necessary and sufficient condition for the existence of a feasible insertion and we define a feasible insertion in the case the condition is verified.

Let γ denote the set of non-dummy activities linked with A_x by a synchronisation constraint.

$$\gamma = \{A_i \in \mathcal{A} | \Delta_{i,x} = 0 \text{ and } \Delta_{x,i} = 0\}$$

Note that $A_{n+1} \notin \gamma$. Indeed, A_{n+1} cannot be synchronized with A_x since $l_{x(n+1)} = p_x \geq 1$. A_0 can only be synchronized with A_x if $l_{0x} = 0$. By convention, although we may have $l_{0x} = l_{x0} = 0$, dummy activity A_0 does not belong to γ. Consequently, for any activity $A_i \in \gamma$ we cannot include A_i in the set of possible predecessors α (otherwise a cycle of length p_a would be issued) nor in the set of possible successors β (otherwise a cycle of length p_x would be issued). Furthermore, in any feasible schedule respecting the precedence constraints, the synchronized activities must start exactly at the same time. Consequently, there exists a solution to the RCAIP only if the following inequalities hold.

$$\sum_{A_i \in \gamma} b_{i,k} + b_{x,k} \leq B_k \quad \forall R_k \in \mathcal{R} \tag{10}$$

In fact, (10) is a necessary and sufficient feasibility condition for the RCAIP. Indeed, insertion (α_0, β_0) defined hereafter is feasible if and only if (10) hold, which will be shown through Lemma 1.

Let α_0 denote the set of activities A_i which are not synchronized with A_x and such that there is a non negative length path from A_i to A_x.

$$\alpha_0 = \{A_i \in V \setminus \gamma | \Delta_{i,x} \geq 0\}$$

Let β_0 denote the complement set of α_0 in V minus the activities synchronized with A_x.

$$\beta_0 = V \setminus (\gamma \cup \alpha_0)$$

Lemma 1 *The RCAIP is feasible and (α_0, β_0) is a feasible insertion if and only if γ satisfies constraints (10).*

Proof First we show that (α_0, β_0) verifies capacity condition (7) if and only if γ satisfies constraints (10). Consider insertion $(\alpha^0, \beta^0) = (\{A_0\}, V \setminus \{A_0\})$. Due to flow conservation of A_0 we have $Q_k(\alpha^0, \beta^0) = B_k$. Consider the activities of $V \setminus \{A_0\}$ such that $\Delta_{i,x} \geq 0$. Assume these activities $\{A_{i_1}, \ldots, A_{i_w}\}$ are sorted in decreasing order of $\Delta_{i_w,x}$, $v = 1, \ldots, w$. Consider now insertions (α^v, β^v), $v = 1, \ldots, w$, such that $(\alpha^v, \beta^v) = (\alpha^{v-1} \cup \{A_{i_t}\}, \beta^{v-1} \setminus \{A_{i_v}\})$. In other words, (α^v, β^v) is obtained by moving A_{i_v} from β^{v-1} to α^{v-1}. Since $\Delta_{i_v,x} \geq \Delta_{i_{v-1},x}$, all the flow received by A_{i_v} is received only from $\{A_0, \ldots, A_{i_{v-1}}\} = \alpha^{v-1}$ and all the flow sent from A_{i_v} is sent to $V \setminus \{A_0, \ldots, A_{i_{v-1}}\} = \beta^{v-1}$. Consequently $Q_k(\alpha^v, \beta^v) = Q_k(\alpha^{v-1}, \beta^{v-1}) = B_k$, for all $R_k \in \mathcal{R}$ and for all $v = 1, \ldots, w$.

Observe now that (α_0, β_0) is obtained from (α^w, β^w) by removing the activities synchronized with A_x (set γ) from the set of posible predecessors α^w. Since $\Delta_{i,x} = 0$ for any activity $A_i \in \gamma$ we have $\Delta_{i,x} \leq \Delta_{i_v,x}$ for all $v = 1, \ldots, w$ and so any activity belonging to γ can only send flow to activities of $V \setminus \{A_0, \ldots, A_{i_w}\} = \beta^w$. Hence, we have

$$Q_k(\alpha_0, \beta_0) = B_k - \sum_{i \in \gamma} b_{i,k}, \forall R_k \in \mathcal{R}$$

Consequenly if γ verifies (10), capacity condition (7) is verified by (α_0, β_0).

Second, we show the length of the cyles in $\mathcal{C}_1(\alpha_0, \beta_0)$, $\mathcal{C}_2(\alpha_0, \beta_0)$ and $\mathcal{C}_3(\alpha_0, \beta_0)$ are non positive. From the definition of α_0, there is no path in $\mathcal{G}$ from x to any activity $A_i \in \alpha_0$ which yields $\Delta_{x,i} = -\infty$ and $\mathcal{L}_1(\alpha_0, \beta_0) = -\infty$. From the definition of β_0, there is no path in $\mathcal{G}$ from any activity $A_j \in \beta_0$ and x which yields $\Delta_{j,x} = -\infty$, and $\mathcal{L}_2(\alpha_0, \beta_0) = -\infty$. There is also no path in $\mathcal{G}$ from any activity $A_j \in \beta_0$ and any activity $A_i \in \alpha_0$, otherwise we would have $\Delta_{j,x} \geq 0$. Consequently $\Delta_{j,i} = -\infty$ and $\mathcal{L}_3(\alpha_0, \beta_0) = -\infty$. □

Condition (10) for the existence of a feasible insertion can be checked in $O(nm)$. Feasible insertion (α_0, β_0) can be obtained in $O(|E \cup F|)$ since checking if $\Delta_{i,x} \geq 0$ can be done by a simple depth-first search algorithm in $\mathcal{G}$.

6.3 Dominant insertions

Let (α, β) denotes a feasible insertion. Let $\mu(\alpha)$ denotes the subset of α such that

$$\mu(\alpha) = \{A_i \in \alpha | \Delta_{0,i} + p_i = \max_{A_a \in \alpha} (\Delta_{0,a} + p_a)\},$$

and $\nu(\beta)$ denotes the subset of β such that

$$\nu(\beta) = \{A_i \in \beta | \Delta_{i,n+1} = \max_{A_b \in \beta} (\Delta_{b,n+1})\}.$$

$\mu(\alpha)$ is the set of activities of α yielding the longest path of length $\mathcal{M}_1(\alpha, \beta)$. $\nu(\beta)$ is the set of activities of β yielding the longest path of length $\mathcal{M}_2(\alpha, \beta)$. $\{\mu(\alpha), \nu(\beta)\}$ is the pair of activity subsets yielding the longest path of length $\mathcal{M}_3(\alpha, \beta)$. Thus, due to the expression of the makespan after insertion (9), to obtain an insertion (α', β') of makespan lower than $\Delta_{0,n+1}(\alpha, \beta)$, we must have $\mu(\alpha) \cap \alpha' = \emptyset$ or $\nu(\beta) \cap \beta' = \emptyset$. For any $A_i \in \mu(\alpha)$ and $A_j \in \nu(\beta)$, we conclude that (α, β) is dominant compared to any other insertion (α', β') verifying A_i belongs to the set of possible predecessors α' and A_j belongs to the set of possible successors β'. Lemma 2 below shows that from (α_0, β_0), we can derive an insertion with even stronger dominance properties.

Given an activity set e, consider the intermediate insertion problem $P_x(e)$ aiming at finding the optimal insertion (α, β) such that $\beta \cap e \neq \emptyset$, i.e. such that at least one possible successor belongs to e. Suppose $\alpha_0 = \{i_1, \ldots, i_{|\alpha_0|}\}$, where activities i_q are sorted in increasing $\Delta_{0,i} + p_i$. Let α_0^* denote the subset $\{i_1, \ldots, i_{|\alpha_0^*|}\}$ of α_0 such that $Q(\alpha_0^*, \beta_0) \geq b_{xk}$ for all $R_k \in \mathcal{R}$ and such that $|\alpha_0^*|$ is minimal.

Lemma 2 *(α_0^*, β_0) is the optimal solution of $P_x(\nu(\beta_0))$.*

Proof We already know that (α_0, β_0) dominates any insertion (α, β) such that $\mu(\alpha_0) \cap \alpha \neq \emptyset$ and $\nu(\beta_0) \cap \beta \neq \emptyset$. By construction, (α_0^*, β_0) dominates any insertion (α, β) such that $\alpha \subseteq \alpha_0$ and $\nu(\beta_0) \cap \beta \neq \emptyset$. The case $\alpha \cap \beta_0 \neq \emptyset$ remains to be checked. Consider an insertion (α, β) with $\nu(\beta_0) \cap \beta \neq \emptyset$, $\alpha \cap \beta_0 \neq \emptyset$ and $\Delta_{0,n+1}(\alpha, \beta) < \Delta_{0,n+1}(\alpha_0^*, \beta_0)$. Let $e = \alpha \cap \beta_0$ denote the set of activities of α that belong also to β_0. Suppose in addition that $e = \{i_1, \ldots, i_{|e|}\}$ is sorted in increasing $\Delta_{0,i_v} + p_{i_v}$ and that $\Delta_{0,i_{|e|}} + p_{i_{|e|}}$ is minimal. Then, there is no flow sent from $i_{|e|}$ to i_v, $v < |e|$. There is no flow sent from $i_{|e|}$ to any activity of $\alpha \setminus e$ neither, otherwise we would have $\Delta_{i_{|e|},x} \geq 0$ and thus $i_{|e|} \in \alpha_0$. Hence, all the flow sent by $i_{|e|}$ is received by β_0 and we have $Q_k(\{i_{|e|}\}, \beta_0) = b_{i_{|e|},k}$, for all $R_k \in \mathcal{R}$. Let $P(i_{|e|})$ denote the set of activities sending flow to $i_{|e|}$ (its resource predecessors). Since $i_{|e|} \in \alpha$, $P(i_{|e|})$ can be included in α without generating any cycle. Also, $i_{|e|}$ can be moved from α to β. without generating any cycle since $i_{|e|} \in \beta_0$ and there is no path from $i_{|e|}$ to i_v, $v < |e|$. By performing both operations, the insertion capacity cannot decrease and the insertion is still feasible. Furthermore, All activities A_j in $P(i_{|e|})$ are such that $\Delta_{0,j} + p_j < \Delta_{0,j} + p_j$ since $p_j \geq 1$ for all j, which contradicts the minimality of $\Delta_{0,i_{|e|}} + p_{i_{|e|}}$ □

Lemma 2 shows that (α_0^*, β_0) dominates any other insertion (α, β) such that at least one activity in $\nu(\beta_0)$ belongs to the set of possible successors β. Thus, we may now consider only insertions such that $\nu(\beta_0) \cap \beta = \emptyset$. In set $\nu(\beta_0)$ some activities are possible resource predecessors while other activities are not. Let $\nu'(\beta)$ denote the subset of $\nu(\beta)$ such that $\Delta_{x,i} = -\infty$, for all $A_i \in \nu'(\beta)$. All activities in $\nu'(\beta_0)$ are possible resource predecessors for A_x. We define a series of insertions $(\alpha_t, \beta_t)_{t \geq 0}$ starting from (α_0, β_0) and performing recurrent modifications as follows:

$$(\alpha_{t+1}, \beta_{t+1}) = \big(\alpha_t \cup \nu'(\beta_t), \beta_t \setminus \nu(\beta_t)\big)$$

Lemma 3 *$(\alpha_{t+1}, \beta_{t+1})$ is feasible if and only if (α_t, β_t) is feasible and $Q_k(\alpha_{t+1}, \beta_{t+1}) \geq b_{x,k}$, for all $R_k \in \mathcal{R}$.*

Proof Since (α_t, β_t) is feasible, we have to show that no positive length cycle is generated by the operation performed on (α_t, β_t) to compute $(\alpha_{t+1}, \beta_{t+1})$. Obviously, removing $\nu(\beta_t)$ from β_t cannot increase any cycle length. Since (α_t, β_t) is feasible, no cycle can be added in $\mathcal{C}_2(\alpha_{t+1}, \beta_{t+1})$. Since $\Delta_{x,i} = -\infty$, for all $A_i \in \nu'(\beta_{t+1})$, no cycle is added in $\mathcal{C}_1(\alpha_{t+1}, \beta_{t+1})$. Since $\Delta_{i,n+1} = \max_{A_b \in \beta_t}(\Delta_{b,n+1})$, for all $A_i \in \nu'(\beta_q)$, there is no path from $A_j \in \beta_t \setminus \nu(\beta_t)$ to A_i and no cycle is added in $\mathcal{C}_3(\alpha_{t+1}, \beta_{t+1})$. Hence $(\alpha_{t+1}, \beta_{t+1})$ remains feasible if its resource capacity remains sufficient. □

If α_t is feasible, suppose $\alpha_t = \{i_1, \ldots, i_{|\alpha_t|}\}$ where activities i_v are sorted in increasing $\Delta_{0,i_t} + p_{i_v}$. Let α_t^* denote the subset $\{i_1, \ldots, i_{|\alpha_t^*|}\}$ of α_t such that $Q(\alpha_t^*, \beta_t) \geq b_{x,k}$ for all $R_k \in \mathcal{R}$ and such that $|\alpha_t^*|$ is minimal.

Note that since set $\nu(\beta_t)$ contains at least one activity, there are at most n terms in the series until β_t becomes empty. Suppose insertions $(\alpha_t, \beta_t)_{0 \leq t < \rho}$ are feasible while $(\alpha_\rho, \beta_\rho)$ is infeasible.

Lemma 4 *Each feasible insertion (α_t^*, β_t) is an optimal solution of $P_x(\nu(\beta_t))$ and all solutions (α, β) such that $\beta \subseteq \beta_\rho$ are dominated by one solution (α_t^*, β_t), $t < \rho$.*

Proof If (α_t, β_t) is feasible, the same arguments as for lemma 2 can be used to show that (α_t^*, β_t) is the optimal solution of $P_x(\nu(\beta_t))$. If $(\alpha_\rho, \beta_\rho)$ is not feasible, we have a resource $R_k \in \mathcal{R}$ such that $Q_k(\alpha_\rho, \beta_\rho) < b_{x,k}$ (from lemma 3). Hence the flow sent

from α_ρ to β_ρ is not sufficient to insert A_x. Consequently, some activities of α_ρ should appear as possible successors in β in any feasible solution such (α, β) that $\beta \subseteq \beta_\rho$. Any possible successor in α_ρ belongs also to a set $\nu(\beta_t)$, $t < \rho$ and so (α, β) is dominated by (α_t, β_t) □

Theorem 2 *The series* $(\alpha_t^*, \beta_t)_{0 \leq t < \rho}$ *is dominant.*

Proof The property follows from lemma 4. Any activity A_i belongs to one of the following sets α_0, γ, $\nu(\beta_t)_{0 \leq t < \rho}$ or β_ρ. If A_i belongs to α_0 or to γ, A_i is not a possible successor of A_x. If A_i belongs to $\nu(\beta_t)$ for some $t \in \{0, \ldots, \rho - 1\}$, the optimal solution (α, β) with $A_i \in \beta$ is included in the series. If Ai is in β_ρ then any insertion such that $A_i \in \beta$ is dominated by one insertion of the series. □

6.4 Polynomial insertion algorithm

We propose Algorithm 2 (OptInsert) generating the series of dominant insertions in polynomial time. Set γ of synchonized activities is computed at step 1. The feasibility condition is checked at step 1. It problem (P_x) is feasible, the initial insertion (α_0, β_0) is computed at steps 3-4. Loop 6-17 generates insertion series (α_t, β_t) starting from (α_0, β_0). Loop 8-11 compute set α_t^*. Steps 12-16 compute $(\alpha_{t+1}, \beta_{t+1})$ from (α_t, β_t), ν and ν'. Finally, Step 18 retrieves the optimal insertion and Step 19 compute the optimal flow. The algorithm can be implemented easily in $O(n^2 m)$ time.

Algorithm 2 OptInsert(x) optimal insertion algorithm

1: $\gamma \leftarrow \{A_i \in \mathcal{A} | \Delta_{i,x} = 0 \text{ and } \Delta_{x,i} = 0\}$;
2: **if** $\sum_{A_i \in \gamma} b_{i,k} + b_{x,k} \leq B_k \quad \forall R_k \in \mathcal{R}$ **then**
3: $\alpha_0 \leftarrow \{A_i \in V \setminus \gamma | \Delta_{i,x} \geq 0\}$
4: $\beta_0 \leftarrow \{V \setminus (\gamma \cup \alpha_0)\}$
5: $t \leftarrow 0$
6: **while** $Q_k(\alpha_t, \beta_t) \geq b_{i,k}$ for all $R_j \in \mathcal{R}$ **do**
7: $\alpha \leftarrow \alpha_t$
8: **while** $Q_k(\alpha, \beta_t) \geq b_{i,k}$ for all $R_j \in \mathcal{R}$ **do**
9: $\alpha_t^* \leftarrow \alpha$
10: $\alpha \leftarrow \alpha \setminus \{A_i \in \alpha_t^* | \Delta_{0,i} + p_i = \max_{A_a \in \alpha_t^*}(\Delta_{0,a} + p_a)\}$
11: **end while**
12: $\nu \leftarrow \{A_i \in \beta_\rho | \Delta_{i,n+1} = \max_{A_b \in \beta_\rho}(\Delta_{b,n+1})\}$;
13: $\nu' \leftarrow \{A_i \in \nu | \Delta_{x,i} = -\infty\}$;
14: $\alpha_{t+1} \leftarrow \alpha_t \cup \nu'$
15: $\beta_{t+1} \leftarrow \beta_{t+1} \setminus \nu$
16: $t \leftarrow t + 1$
17: **end while**
18: $(\alpha^*, \beta^*) = \arg \min_{\tau = 0, \ldots, t-1} \Delta_{0,n+1}(\alpha_t^*, \beta_t)$
19: $f \leftarrow$generateFlowFromInsertion(α, β)
20: **else**
21: The problem is infeasible;
22: **end if**

6.5 Illustrative example

Finally we give the complete behaviour of the algorithm on an illustrative example.

To illustrate the feasibility condition, let us consider a RCPSP example with 10 activities and two resources such that $B_1 = 7$ and $B_2 = 4$. The processing times and the resource demands are indicated on Table 2. Minimum time lags are displayed on Figure 8. We assume that all activities, except Activity A_5 that has to be inserted, are already scheduled as shown on the Gantt diagram of Figure 10. The flow network f corresponding to this partial schedule is displayed on Figure 9 and gives a makespan $C_{\max} = 11$. The arc weight L_{ij} and flow values $f_{i,j,k}$ (between parenthesis) are displayed near each arc. In the resource flow network, A_5 is isolated. 6, 1, 1, 2, 5, 3, 5, 3, 2, 4

Table 2 Processing times and resource demands

i	1	2	3	4	5	6	7	8	9	10
p_i	6	1	1	2	5	3	5	3	2	4
$(b_{i,1}, b_{i,2})$	$(2,1)$	$(1,0)$	$(3,1)$	$(2,0)$	$(1,2)$	$(2,1)$	$(2,0)$	$(1,1)$	$(1,2)$	$(1,1)$

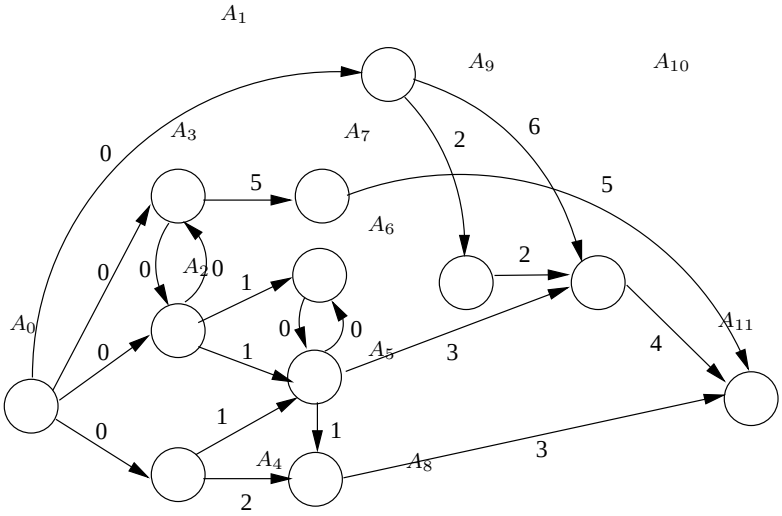

Fig. 8 Minimum time lags

The values of $\Delta_{0,i} + p_i$ and $\Delta_{i,n+1}$, for all activities A_i, needed to check the feasibility or compute the makespan of any insertion can be computed using the Bellmann-Ford's algorithm. They are indicated in Table 3.

Sets γ, α_0 and β_0 are defined as follows. $\gamma = \{6\}$ since A_6 is the only activity which is synchronized with activity 5 ($S_5 = S_6$). Then we find $\alpha_0 = \{A_0, A_2, A_3, A_4\}$ and $\beta_0 = \{A_1, A_7, A_8, A_9, A_{10}, A_{11}\}$ (see the cut denoted $\rho = 0$ on Figure 9). This

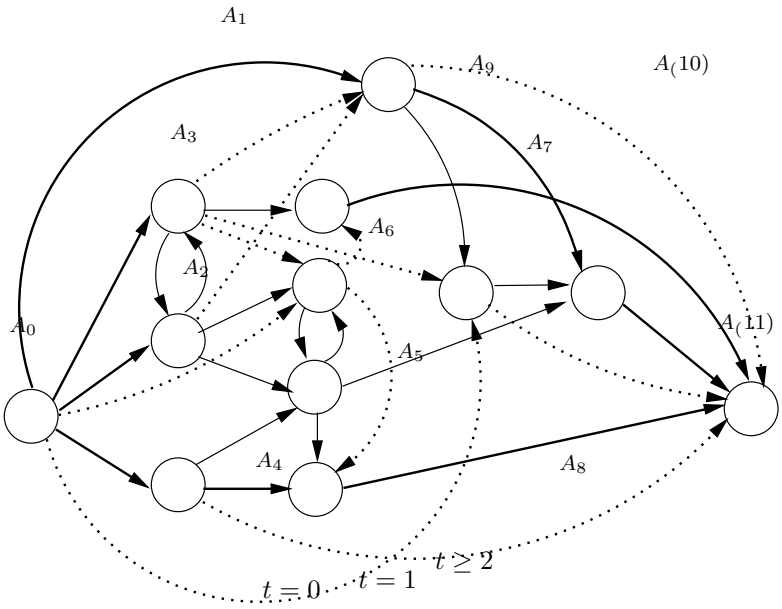

Fig. 9 Resource flow network

Table 3 Values of $\Delta_{0,i} + p_i$ and $\Delta_{i,n+1}$

i	0	1	2	3	4	5	6	7	8	9	10	11
$\Delta_{0,i} + p_i$	0	7	1	1	2	6	4	10	7	5	11	11
$\Delta_{i,n+1}$	11	10	11	11	9	8	8	5	3	6	4	0

insertion (α_0, β_0) is resource-feasible since, $B_1 - b_{6,1} = 5 \geq b_{5,1} = 1$ and $B_2 - b_{6,2} = 3 \geq b_{5,2} = 2$. We may check in Table 3 that $\mu(\alpha_0, \beta_0) = \{A_4\}$ and $\nu(\alpha_0, \beta_0) = \{A_1\}$ with $\Delta_{0,4} + p_4 = 2$ and $\Delta_{1,11} = 10$. The insertion yields a makespan

$$\Delta_{0,n+1}(\alpha, \beta) = \max\left(\Delta_{0,n+1}, \mathcal{M}_1(\alpha_0, \beta_0), \mathcal{M}_2(\alpha_0, \beta_0), \mathcal{M}_3(\alpha_0, \beta_0)\right) = 17.$$

since we have $\Delta_{0,n+1} = 11$, $\mathcal{M}_1(\alpha_0, \beta_0) = \Delta_{0,4} + p_4 + \Delta_{5,11} = 10$, $\mathcal{M}_2(\alpha_0, \beta_0) = \Delta_{0,5} + p_5 + \Delta_{1,11} = 16$ and $\mathcal{M}_3(\alpha_0, \beta_0) = \Delta_{0,4} + p_4 + p_5 + \Delta_{1,11} = 17$. (α_0, β_0) is an optimal insertion among all insertion such that A_4 is a resource predecessor and A_1 is a resource successor. We now give the dominant insertions computed by Algorithm 2. For $t = 0$, dominant insertion (α_0^*, β_0) is computed by removing as many activities i in (α_0) of largest $\Delta_{0,i} + p_i$ as possible while keeping the insertion capacity sufficient. Here we can remonce activities $\{2, 3, 4\}$ yielding $\alpha_t^* = \{0\}$ and $\Delta_{0,n+1}(\alpha_0^*, \beta_0) = 16$. The next dominant insertion is computed by removing $\nu(\alpha_0, \beta_0)$ from β_0 to obtain β_1 and adding $\nu'(\alpha_0, \beta_0)$ in α_0 to obtain α_1. Here since $\Delta_{1,5} = -\infty$ we can insert A_1 into α_0. The process is iterated for $t = 1$ until there remain enough capacity in the insertion. Table 4 displays the 5 dominant insertion for the illustrative example problem. Insertions (α_t, β_t) aer displayed as cuts of the resource flow network in Figure 9.

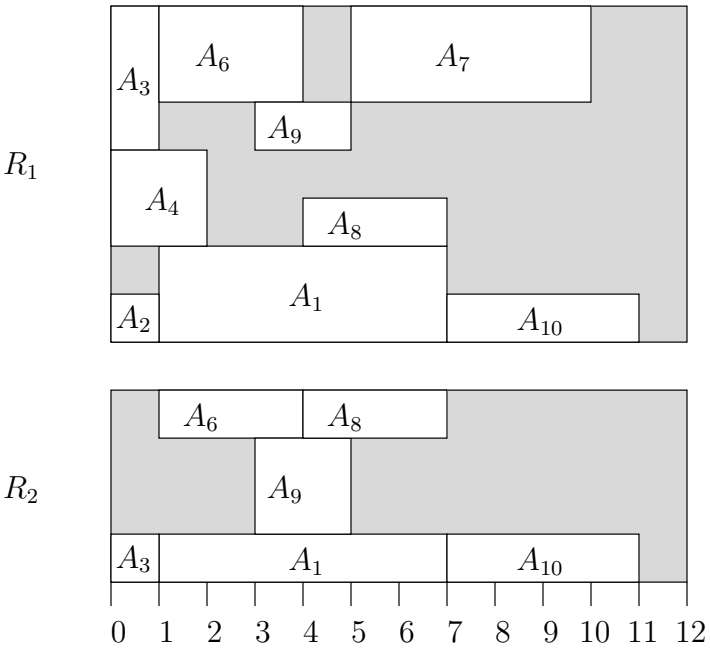

Fig. 10 Partial solution not including Activity A_5

The best insertion is obtained for $\alpha_1^* = \{A_0, A_2, A_3\}$ and $\beta_1 = \{A_7, A_8, A_9, A_10, A_11\}$ yielding a makespan of 12. Figure 11 displays the solution obtained after insertion A_5 into (α_1^*, β_1).

Table 4 Dominant insertions

$t=0$	$\alpha_0=\{A_0,A_2,A_3,A_4\}\quad \beta_0=\{A_1,A_7,A_8,A_9,A_{10},A_{11}\}$ $\alpha_0^*=\{A_0\}\quad \Delta_{0,n+1}(\alpha_0^*,\beta_0)=16$ $\nu(\alpha_0,\beta_0)=\{A_1\}\quad \nu'(\alpha_0,\beta_0)=\{A_1\}$
$t=1$	$\alpha_1=\{A_0,A_2,A_3,A_4,A_1\}\quad \beta_1=\{A_7,A_8,A_9,A_{10},A_{11}\}$ $\alpha_1^*=\{A_0,A_2,A_3\}\quad \Delta_{0,n+1}(\alpha_1^*,\beta_1)=12$ $\nu(\alpha_1,\beta_1)=\{A_9\}\quad \nu'(\alpha_1,\beta_1)=\{A_9\}$
$t=2$	$\alpha_2=\{A_0,A_2,A_3,A_4,A_1,A_9\}\quad \beta_2=\{A_7,A_8,A_{10},A_{11}\}$ $\alpha_2^*=\{A_0,A_2,A_3,A_4,A_9\}\quad \Delta_{0,n+1}(\alpha_2^*,\beta_2)=15$ $\nu(\alpha_2,\beta_2)=\{A_7\}\quad \nu'(\alpha_2,\beta_2)=\emptyset$
$t=3$	$\alpha_3=\{A_0,A_2,A_3,A_4,A_1,A_9\}\quad \beta_3=\{A_8,A_{10},A_{11}\}$ $\alpha_3^*=\{A_0,A_2,A_3,A_4,A_9\}\quad \Delta_{0,n+1}(\alpha_3^*,\beta_3)=14$ $\nu(\alpha_3,\beta_3)=\{A_10\}\quad \nu'(\alpha_3,\beta_3)=\emptyset$
$t=4$	$\alpha_4=\{A_0,A_2,A_3,A_4,A_1,A_9\}\quad \beta_4=\{A_8,A_{11}\}$ $\alpha_4^*=\{A_0,A_2,A_3,A_4,A_9\}\quad \Delta_{0,n+1}(\alpha_4^*,\beta_4)=13$ $\nu(\alpha_4,\beta_4)=\{A_8\}\quad \nu'(\alpha_4,\beta_4)=\emptyset$
$t=5$	$\alpha_5=\{A_0,A_2,A_3,A_4,A_1,A_9\}\quad \beta_5=\{A_{11}\}$ $\alpha_5^*=\{A_0,A_2,A_3,A_4,A_9\}\quad \Delta_{0,n+1}(\alpha_5^*,\beta_5)=13$ $\nu(\alpha_5,\beta_5)=\{A_{11}\}\quad \nu'(\alpha_5,\beta_5)=\emptyset$

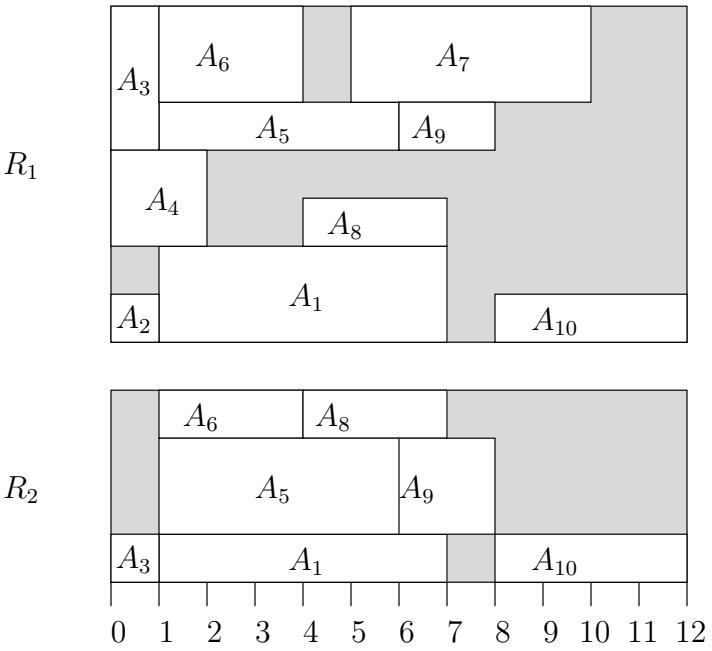

Fig. 11 Complete solution after A_5 insertion

7 Conclusion

Whereas the insertion problem is polynomially solvable for the standard RCPSP (where only precedence constraints are taken into account), we showed that the introduction of minimum and maximum time lags makes the problem NP-hard. Nevertheless, when only minimum time lags are considered and when activity durations are strictly positive, the problem turns back polynomially solvable and we proposed an algorithm to solve it.

Further research may consist in designing efficent heuristics and/or branch and bound methods to solve the RCAIP with minimum and maximum time lags. A possible heuristic is to use the proposed polynomial algorithm as a basic search framework, while considering maximal time lag violation as a criterion to derive more insertion solutions. Another way of tackling insertion problems in presence of maximum time lags, would be to define another structure of the partial schedule to turn the RCAIP into a polynomially solvable problem. This remains a critical issue for designing efficient local search methods for the RCPSP/max based on individual activity reinsertions.

References

1. Artigues C., Michelon P. and Reusser S. Insertion techniques for static and dynamic resource-constrained project scheduling. European Journal of Operational Research 2003; **149** (2):249-267.
2. Artigues C., Roubellat F. A polynomial activity insertion algorithm in a multi-resource schedule with cumulative constraints and multiple modes. European Journal of Operational Research 2000; **127** (2):297-316.
3. Bartusch M., Möhring R.H. and Radermacher F.J. Scheduling project networks with resource constraints and time windows. Annals of Operations Research 1988; **16**:201-240.
4. Brucker P. and Neyer J. Tabu-search for the multi-mode job-shop problem. OR Spektrum 1998; **20**:21-28.
5. Duron C., Proth, J.M., Wardi, Y. Insertion of a random task in a schedule: a real-time approach, European Journal of Operational Research 2005, **164** (1):52-63.
6. Fortemps Ph. and Hapke M. On the Disjunctive Graph for Project Scheduling. Foundations of Computing and Decision Sciences 1997; **22**:195-209.
7. Garey M. R. and Johnson D.S. Computers and intractability. A guide to the theory of NP-completeness. Freeman, 1979.
8. Gröflin H. and Klinkert A. Feasible insertions in job shop scheduling, short cycles and stable sets. European Journal of Operational Research 2007; **177** (2):763-785.
9. Kis T. and Hertz, A. A lower bound for the job insertion problem. Discrete Applied Mathematics 2003; **128** (2-3):395419.
10. Klinkert, A., Gröflin, H. and Pham-Dinh, N. Feasible Job Insertions in the Multi-Processor-Task Job Shop. European Journal of Operational Research 2008, **185** (3): 1308-1318.
11. Leus R. and Herroelen W. Stability and resource allocation in project planning. IIE Transactions 2004; **36** (7):1-16.
12. Neumann K., Schwindt C. and Zimmermann J. Project scheduling with time windows and scarce resources. Springer, 2003.
13. Vaessens R.J.M. Generalized job shop scheduling: complexity and local search. Ph.D. thesis, Eindhoven University of Technology, Rotterdam, 1995.
14. Vieira G.E., Herrmann J.W., Lin E. Rescheduling manufacturing systems: A framework of strategies, policies, and methods. Journal of Scheduling 2003, **6** (1): 39-62.
15. Vonder, S., Demeulemeester, E., Herroelen, W. A classification of predictive-reactive project scheduling procedures. Journal of Scheduling 2007, **10** (3): 195-207.

Summary of abbreviations and notations

RCPSP	resource-constrained project scheduling problem
RCPSP/Max	RCPSP with minimum and maximum time lags
RCAIP	resource-constrained activity insertion problem
n	number of activities
$G(V, E, l)$	activity-on-node graph
$V = \{A_0, \ldots, A_{n+1}\}$	set of activities
E	set of precedence relations (A_i, A_j)
p_i	duration of activity A_i
l_{ij}	time lag for $(A_i, A_j) \in E$
m	number of resources
$\mathcal{R} = \{R_1, \ldots, R_m\}$	set of resources
B_k	number of available units for resource R_k
$b_{i,k}$	number of units of R_k required by A_i
S_i	start time of A_i
$\mathcal{A}_t$	set of activities in process at time t
(P)	short notation for the RCPSP/max problem
$(\mathrm{P}{-x})$	short notation for the RCPSP/max where $b_{xk} = 0$, $\forall R_k \in \mathcal{R}$
$(\mathrm{P}x)$	short notation for the RCAIP where A_x has to be inserted
UB	upper bound on the makespan ($l_{0,n+1} = -UB$)
$\delta_{i,j}$	longest path from A_i to A_j in $G(V, E, l)$
$f_{i,j,k}$	number of resource R_k units transferred from A_i to A_j
$\mathcal{G}(f)$ or $\mathcal{G}$	graph induced by flow f
$F(f)$ or F	set of arcs (precedence constraints) induced by flow f
$L_{ij}(f)$ or L_{ij}	weight of arc (A_i, A_j) induced by flow f
$\Delta_{i,j}(f)$ or $\Delta_{i,j}$	longest path from A_i to A_j in $\mathcal{G}(f)$
$q_{i,j,k}$	part of flow $f_{i,j,k}$ rerouted to the inserted activity
(α, β)	ordered pair of set of activities representing an insertion
α	set of possible resource predecessors
β	set of possible resource successors
$Q_k(\alpha, \beta)$	amount of R_k units available for insertion in (α, β)
$\mathcal{G}(\alpha, \beta)$	graph issued from the insertion of A_x in (α, β)
$F(\alpha, \beta)$	set of arcs issued from the insertion of A_x in (α, β)
$L_{i,j}(\alpha, \beta)$	weight of arc (A_i, A_j) after insertion of A_x in (α, β)
$\Delta_{i,j}(\alpha, \beta)$	longest path from A_i to A_j in $\mathcal{G}(\alpha, \beta)$
$\mathcal{C}_q(\alpha, \beta)$	set of type q cycles in $\mathcal{G}(\alpha, \beta)$ $(q = 1, 2, 3)$
$\mathcal{L}_q(\alpha, \beta)$	length of the longest cycle in $\mathcal{G}(\alpha, \beta)$ $(q = 1, 2, 3)$
$\mathcal{M}_q(\alpha, \beta)$	length of the longest path of type q in $\mathcal{G}(\alpha, \beta)$ $(q = 1, 2, 3)$
γ	set of non-dummy activities linked with A_x by a synchronisation constraint
$\mu(\alpha)$	set of activities $i \in \alpha$ of largest $\Delta_{0,i} + p_i$
$\nu(\alpha)$	set of activities $i \in \beta$ of largest $\Delta_{i,n+1}$
$\nu'(\alpha)$	subset activities $i \in \nu$ such that $\Delta_{x,i} = -\infty$