A fixed-parameter algorithm for a unit-execution-time unit-communication-time tasks scheduling problem with a limited number of identical processors

Alix Munier-Kordon, Ning Tang

RAIRO-Oper. Res. 56 (2022) 3777–3788
https://doi.org/10.1051/ro/2022174

RAIRO Operations Research
www.rairo-ro.org

A FIXED-PARAMETER ALGORITHM FOR A UNIT-EXECUTION-TIME UNIT-COMMUNICATION-TIME TASKS SCHEDULING PROBLEM WITH A LIMITED NUMBER OF IDENTICAL PROCESSORS

Alix Munier Kordon* and Ning Tang

Abstract. This paper considers the minimization of the maximum lateness for a set of dependent tasks with unit duration, unit communication delays release times and due dates. The number of processors is limited, and each task requires one processor for its execution. A time window built from an upper bound of the minimum maximum lateness is associated to each task. The parameter considered is the pathwidth of the associated interval graph. A fixed-parameter algorithm based on a dynamic programming approach is developed to solve this optimization problem. This is, as far as we know, the first fixed-parameter algorithm for a scheduling problem with communication delays and a limited number of processors.

Mathematics Subject Classification. 90B35, 68Q27.

Received November 16, 2021. Accepted October 4, 2022.

1. Introduction

Scheduling problems with communication delays have been intensively studied since 1990s because of the importance of practical applications. Several surveys are dedicated to this class of problems known to be mostly NP-hard [4, 9, 11, 24].

This paper considers a scheduling problem with communication delays defined as follows: a set $\mathcal{T} = \{1, 2, \ldots, n\}$ of n tasks is to be executed on m identical machines (processors). Each machine can process at most one task at a time and each task is to be processed once. Tasks have a unit execution time and are partially ordered by a precedence graph $\mathcal{G} = (\mathcal{T}, \mathcal{A})$. Let σ be a feasible schedule; for any task $i \in \mathcal{T}$, t_i^σ denotes the starting time of the task i following σ. For any arc $(i, j) \in \mathcal{A}$, the task i must finish its execution before the task j starts executing, *i.e.* $t_i^\sigma + 1 \leq t_j^\sigma$. If the tasks i and j are assigned to different processors, a unit communication delay must be added after the execution of the task i to send data to the task j, thus $t_i^\sigma + 2 \leq t_j^\sigma$. Moreover, we assume that release dates $r_i \in \mathbb{N}$ and due dates $d_i \in \mathbb{N}$ are given. Then, the inequality $r_i \leq t_i^\sigma$ holds for each task $i \in \mathcal{T}$.

The maximum lateness of a feasible schedule σ is defined as $L(\sigma) = \max_{i \in \mathcal{T}}(t_i^\sigma + 1 - d_i)$; the minimization of the maximum lateness is denoted by $L_{\max}$. The problem considered in this paper is designated by $P|r_i, \text{prec}, p_i = 1, c_{ij} = 1|L_{\max}$ using standard notations [13]. The minimization of the maximum lateness

Keywords. Scheduling, communication delays, minimization of the maximum lateness, fixed-parameter algorithm.

Sorbonne Université, CNRS, LIP6, F-75005 Paris, France.
*Corresponding author: Alix.Munier@lip6.fr

includes the minimization of the makespan $C_{\max}$; Indeed, the minimization of the maximum lateness for an instance with $d_i = 0$ for each task $i \in \mathcal{T}$ corresponds to minimize the makespan.

Rayward-Smith [18] was the first author to study the problem $P|\text{prec}, p_i = 1, c_{ij} = 1|C_{\max}$ and to establish its NP-hardness. Later, Hoogeveen *et al.* [15] proved that the decision problem $P|\text{prec}, p_i = 1, c_{ij} = 1|C_{\max} \leq 3$ is polynomially solvable while $P|\text{prec}, p_i = 1, c_{ij} = 1|C_{\max} \leq 4$ is NP-complete.

Several authors developed algorithms for solving scheduling problems with similar constraints. Veltman provided in [23] an exact dynamic programming algorithm of time complexity $\mathcal{O}(2^{w(G)} \times n^{2w(G)})$ for $P|\text{prec}, p_i = 1, c_{ij} = 1|C_{\max}$ where $w(G)$ is the width of the precedence graph G, *i.e.* the size of the largest antichain. Zinder *et al.* [26] have developed and tested an exact branch-and-bound algorithm for the problem $P|\text{prec}, p_i = 1, c_{ij} = 1|L_{\max}$. For more general problems, several authors considered integer linear programming formulations (ILP in short). Davidović *et al.* [6] tackled scheduling problems for a fixed network of processors; communications are proportional to both the amount of exchanged data between pairs of dependent tasks and the distance between processors in the multiprocessor architecture. These authors developed two formulations and compared them experimentally. Later, Ait El Cadi *et al.* [1] improved this approach by reducing the size of the linear program (number of variables and constraints) and adding cuts; they compared positively to the previous works. Venugopalan and Sinnen [25] provided a new ILP formulation for $P|\text{prec}, c_{ij}|C_{\max}$ and comparisons with Davidović *et al.* [6] for several classes of graphs and fixed number of processors.

Fixed-parameter algorithms for NP-complete problems allow to obtain polynomial-time algorithms when some parameters are fixed [5, 8]. More precisely, a fixed-parameter algorithm solves any instance of a problem of size n in time $f(k) \times \text{poly}(n)$, where f is allowed to be a computable superpolynomial function and k the associated parameter.

Mnich and van Bevern [16] surveyed recently main results on parameterized complexity for scheduling problems and identified 15 challenging open problems. For the scheduling problem with usual precedence constraints, many researchers consider the width $w(G)$ of the precedence graph as a parameter, leading usually to negative results. Du *et al.* [10] proved that $P2|\text{chains}|C_{\max}$ is strongly NP-hard for unbounded width. Günther *et al.* [14] proved that $P2|\text{chains}, w(G) \leq 3|C_{\max}$ is weakly NP-hard. Bodlaender and Fellows [3] proved that $P|\text{prec}, p_i = 1|C_{\max}$ is W[2]-hard parameterized by the width and the number of machines. More recently, van Bevern *et al.* [22] proved that $P2|\text{prec}, p_j \in \{1, 2\}|C_{\max}$ is W[2]-hard parameterized by the width $w(G)$.

Let us suppose that an upper bound $\bar{L}$ of the maximum lateness is fixed. We develop in this paper a fixed-parameter algorithm for $P|r_i, \text{prec}, p_i = 1, c_{ij} = 1|L_{\max}$ in time complexity $\mathcal{O}(n^2 + n \times pw(\bar{L}) \times 2^{3pw(\bar{L})})$. Let us consider the interval graph $\mathcal{I}(\bar{L}) = (\mathcal{T}, E(\bar{L}))$ associated to the time windows $(r_i, d_i + \bar{L})$, $i \in \mathcal{T}$. An edge $e = (i, j) \in E(\bar{L})$ if the intersection $(r_i, d_i + \bar{L}) \cap (r_j, d_j + \bar{L}) \neq \emptyset$. The parameter $pw(\bar{L})$ is the pathwidth of the interval graph $\mathcal{I}(\bar{L})$ [2]; It corresponds to the maximum number of intersecting time windows $(r_i, d_i + \bar{L})$, $i \in \mathcal{T}$ minus 1. Our algorithm is as far as we know the first fixed-parameter algorithm for solving a scheduling problem with communication delays and a bounded number of processors.

The pathwidth was identified recently by several authors as an important parameter for several classes of scheduling problems. De Weerdt *et al.* [7] provided an exact fixed-parameter algorithm in the slack and the pathwidth for a sequencing problem with rejection, set-up times and penalties. Munier Kordon [17] developed a fixed-parameter algorithm in the pathwidth for the basic scheduling problem $P|r_i, \text{prec}, p_i = 1|C_{\max}$ to handle both precedence constraints and resource limitations. A similar approach was developed by Tang and Munier Kordon [21] who presented a fixed-parameter algorithm in the pathwidth in time complexity $\mathcal{O}(n^3 + n \times pw(\bar{C}) \times 2^{4pw(\bar{C})})$ for the scheduling problem with communication delays and unlimited number of processors $\bar{P}|r_i, \text{prec}, p_i = 1, c_{ij} = 1|C_{\max}$. The value $\bar{C}$ is here an upper bound of the minimum makespan. This approach was extended in this paper by considering a more complex criteria, namely the maximum lateness, and a limited number of parallel identical processors. Moreover, the structure of the algorithm was improved to limit the enumeration to active schedules with a slightly better worst-case time complexity. A schedule σ is active if there is not another feasible schedule σ' such that, for each task $i \in \mathcal{T}$, $t_i^{\sigma'} \leq t_i^{\sigma}$ with at least one of these inequalities is strict [19].

This paper is organized as follows: Section 2 presents additional notations. In order to limit the combinatorial explosion of the method, we identify a structural property of the set of tasks schedulable at each time instant and a characterization of active schedules [19]. Our algorithm is presented in Section 3, and its correctness established in Section 4. Section 5 is devoted to its complexity. Section 6 is our conclusion.

2. Problem definition and dominance properties

This section is devoted to some theoretical lemmas that will be considered for the correctness of our algorithm. A small example is also provided. The scheduling problem considered is described in Section 2.1, while a small example is presented in Section 2.2. Section 2.3 presents some important notations and a structural property of feasible schedules. Lastly, Section 2.3 presents a dominance property of active schedules that will be considered below.

2.1. Problem definition

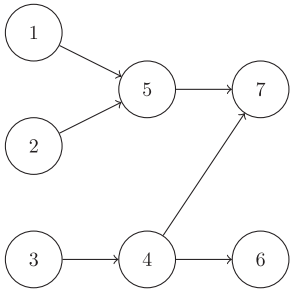
For each task $i \in \mathcal{T}$, let $\Gamma^+(i)$ (resp. $\Gamma^-(i)$) be the set of direct successors (resp. predecessors) of i, *i.e.* $\Gamma^+(i) = \{j \in \mathcal{T}, (i,j) \in \mathcal{A}\}$ and $\Gamma^-(i) = \{j \in \mathcal{T}, (j,i) \in \mathcal{A}\}$.

We observe that a feasible schedule σ is completely defined by the starting times vector $t^\sigma \in \mathbb{N}^n$. Indeed, for any arc $e = (i,j) \in \mathcal{A}$, we note x_{ij}^σ the communication delay between the tasks i and j; we set $x_{ij}^\sigma = 0$ if the execution of the task j starts right after the task i. These two tasks are necessarily executed by a same processor and the communication delay is removed. Otherwise, a communication delay is required between the completion time of the task i and the starting time of the task j and thus $x_{ij}^\sigma = 1$. We then set $x_{ij}^\sigma = \min\{t_j^\sigma - t_i^\sigma - 1, 1\}$ for each arc $e = (i,j) \in \mathcal{A}$.

The problem considered is expressed below. A time-indexed formulation should be considered to transform it into an integer linear program [20] for modelling the resource constraints. We set $d_{\max} = \max_{i\in\mathcal{T}} d_i$ (resp. $r_{\max} = \max_{i\in\mathcal{T}} r_i$) the maximum due date (resp. release time); we also suppose that an upper bound of the maximum lateness $\bar{L}$ is fixed. We then observe that $\bar{C} = \min(r_{\max} + 2n, d_{\max} + \bar{L})$ is an upper bound of the makespan of any active feasible schedule which maximum lateness is bounded by $\bar{L}$.

$$
\begin{cases}
\text{minimize } L & \\
t \in \mathbb{N}^n; L \in \mathbb{Z}; \forall e = (i,j) \in \mathcal{A}, x_{ij} \in \{0,1\} & (1)\\
\forall i \in \mathcal{T}, L \geq t_i + 1 - d_i \text{ and } t_i \geq r_i & (2)\\
\forall e = (i,j) \in \mathcal{A}, x_{ij} = \min\{t_j - t_i - 1, 1\} & (3)\\
\forall e = (i,j) \in \mathcal{A}, t_i < t_j & (4)\\
\forall i \in \mathcal{T}, \displaystyle\sum_{j\in\Gamma^+(i)} x_{ij} \geq |\Gamma^+(i)| - 1 & (5)\\
\forall i \in \mathcal{T}, \displaystyle\sum_{j\in\Gamma^-(i)} x_{ji} \geq |\Gamma^-(i)| - 1 & (6)\\
\forall \alpha \in \{0,\ldots,\bar{C}\}, |\{i \in \mathcal{T}, t_i = \alpha\}| \leq m. & (7)
\end{cases}
$$

Since communications delays and length of the tasks are unitary, starting times can be reduced to integer values; Inequalities (2) come from the definition of the maximum lateness and the release dates. Communication delays are defined from the starting time of the tasks (3). Inequalities (4)–(6) express the communication delay constraints: any task i has at most one successor (resp. predecessor) performed at its completion time (resp. just before its starting time) on the same processor. Inequalities (7) express the limitation on the number of processors.



$i \in \mathcal{T}$	1	2	3	4	5	6	7
r_i	0	0	0	1	1	2	2
d_i	1	1	1	2	2	3	3

FIGURE 1. An instance of $P|r_i, \text{prec}, p_i = 1, c_{ij} = 1|L_{\max}$ with $m = 2$ machines.

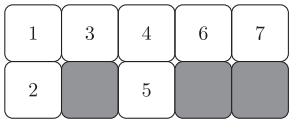


FIGURE 2. A feasible schedule σ of maximum lateness $L(\sigma) = 2$ associated to the example given in Figure 1.

2.2. Example

Let us consider an instance of our scheduling problem defined by 7 tasks of unit length. The precedence graph, release dates and due dates are reported by Figure 1. The number of machines is fixed to 2. A feasible schedule σ of maximum lateness $L(\sigma) = 2$ is given by Figure 2.

2.3. Time windows, pathwidth, and a structural property

Let us consider that an upper bound of the maximum lateness $\bar{L}$ is fixed. This value can be easily computed by extending the classical Graham priority list algorithm [12]. Then for any optimal feasible schedule σ, each task $i \in \mathcal{T}$ has to be completed in the time window $[r_i, d_i + \bar{L}]$, which is equivalent to $t_i^\sigma \geq r_i$ and $t_i^\sigma \leq d_i + \bar{L} - 1$.

Let us suppose that the tasks are numbered following increasing release times, that is $r_1 \leq r_2 \ldots \leq r_n$. We also suppose that the minimum value for a release date $r_{\min} = r_1 = 0$. We can consider that release dates are compatible with respect to precedence, that is, if $(i, j) \in \mathcal{A}$, $r_i + 1 \leq r_j$.

Next lemma bounds the maximum value of the release dates and $\bar{C}$:

Lemma 2.1. *We can suppose that* $r_{\max} = \max_{i \in \mathcal{T}} r_i \leq 2(n-1)$ *and* $\bar{C} \leq 4n - 2$ *without loss of generality.*

Proof. Let suppose by contradiction that $r_n > 2(n-1)$, and let $i^\star$ be the smallest value $i \in \{1, \ldots, n\}$ with $r_i > 2(i-1)$; since $r_1 = 0$, we get that $i^\star > 1$. For every task $j \in \{1, \ldots, i^\star - 1\}$, $r_j \leq 2(j-1)$.

The biggest values for the release date of tasks j in $\{1, \ldots, i^\star - 1\}$ is $r'_j = 2(j-1)$ and the most constrained instance is a path $1 \to 2 \ldots \to i^\star - 1$. In this case, the only active feasible schedule σ is $t_j^\sigma = 2(j-1)$. Thus, any active feasible schedule of tasks $\{1, \ldots, i^\star - 1\}$ would end at time $2(i^\star - 2) + 1$ or before. Since $r_{i^\star} \geq 2(i^\star - 1) + 1$, there will be at least two idle time slots at time $2(i^\star - 2) + 1$ and $2(i^\star - 1)$ before the beginning of tasks in $\{i^\star, i^\star + 1, \ldots n\}$. Since release times are compatible with respect to precedence, we can treat separately the two sets of tasks $\{1, \ldots, i^\star - 1\}$ and $\{i^\star, \ldots, n\}$. The second scheduling problem considers tasks $\{i^\star, \ldots, n\}$ with release times $\tilde{r}_j = r_j - r_{i^\star}$ and due dates $\tilde{d}_j = d_j - r_{i^\star}$. Thus, the first part of the lemma is proved.

Now, since $\bar{C} = \min(r_{\max} + 2n, d_{\max} + \bar{L})$, $\bar{C} \leq 4n - 2$ and the lemma is proved. □

For any value $\alpha \in \{0, \ldots, \bar{C}\}$, we set $X_\alpha = \{i \in \mathcal{T}, (\alpha, \alpha + 1) \cap (r_i, d_i + \bar{L}) \neq \emptyset\}$ the set of tasks that are schedulable at time α considering only the time windows. We also set $Z_\alpha = \{i \in \mathcal{T}, d_i + \bar{L} \leq \alpha + 1\}$ as the set

of tasks that must be completed at time $\alpha+1$. The pathwidth $pw(\bar{L})$ is the maximum size of X_α minus 1, *i.e.* $pw(\bar{L}) = \max_{\alpha\in\{0,\ldots,\bar{C}\}} |X_\alpha| - 1$.

Since $\bar{L}$ is an upper bound of the minimum maximum lateness, our optimization problem is solvable even if we restrict our algorithm to the determination of feasible schedules σ with $L(\sigma) \leq \bar{L}$.

Now, let suppose that σ is a feasible schedule with $L(\sigma) \leq \bar{L}$. For every integer $\alpha \in \{0,\ldots,\bar{C}-1\}$, we set $\mathcal{T}^\sigma_\alpha = \{i \in \mathcal{T}, t^\sigma_i = \alpha\}$ as the set of tasks performed at time α by σ. The following lemma will be considered further to reduce the size of the tasks sets built at each step of our algorithm.

Lemma 2.2. *Let σ be a feasible schedule of $\mathcal{G}$ of maximum lateness $L(\sigma)$ bounded by $\bar{L}$. For any $\alpha \in \{0,\ldots,\bar{C}-1\}$, $\bigcup_{\beta=0}^{\alpha} \mathcal{T}^\sigma_\beta - Z_\alpha \subseteq X_\alpha \cap X_{\alpha+1}$.*

Proof. Since σ is feasible, for any $\alpha \in \{0,\ldots,\bar{C}-1\}$, $\mathcal{T}^\sigma_\alpha \subseteq X_\alpha$ and thus, $\forall i \in \bigcup_{\beta=0}^{\alpha} \mathcal{T}^\sigma_\beta$, $r_i \leq \alpha$. Moreover, each task $i \notin Z_\alpha$ satisfies $d_i + \bar{L} \geq \alpha + 2$.

Thus, for any task $i \in \bigcup_{\beta=0}^{\alpha} \mathcal{T}^\sigma_\beta - Z_\alpha$, $[\alpha, \alpha+2] \subseteq [r_i, d_i + \bar{L}]$. Therefore $\bigcup_{\beta=0}^{\alpha} \mathcal{T}^\sigma_\beta - Z_\alpha \subseteq X_\alpha \cap X_{\alpha+1}$, and the lemma is proved. □

For the example given by Figure 1 and the upper bound $\bar{L} = 2$, we get $\bar{C} = \min(2+14, 3+2) = 5$, $X_0 = \{1,2,3\}$, $X_1 = \{1,2,3,4,5\}$, $X_2 = \{1,2,3,4,5,6,7\}$, $X_3 = \{4,5,6,7\}$, $X_4 = \{6,7\}$ and $X_5 = \emptyset$. Similarly, $Z_0 = Z_1 = \emptyset$, $Z_2 = \{1,2,3\}$, $Z_3 = \{1,2,3,4,5\}$ and $Z_4 = \{1,2,3,4,5,6,7\} = Z_5$.

Now, let us consider the feasible schedule given by Figure 2. For $\alpha = 3$, $\bigcup_{\beta=0}^{3} \mathcal{T}^\sigma_\beta - Z_3 = \{1,2,3,4,5,6\} - \{1,2,3,4,5\} = \{6\}$. Since $X_3 \cap X_4 = \{6,7\}$, we get that $\bigcup_{\beta=0}^{3} \mathcal{T}^\sigma_\beta - Z_3 \subseteq X_3 \cap X_4$.

2.4. A general dominance property of active schedules

Let us consider that σ is a feasible schedule of maximum lateness bounded by $\bar{L}$. For every integer $\alpha \in \{-1,\ldots,\bar{C}-1\}$, we set $W_\alpha = \bigcup_{\beta=0}^{\alpha} \mathcal{T}^\sigma_\beta$ and $B_\alpha = \mathcal{T}^\sigma_\alpha$. The set W_α contains all the tasks that are executed in time $[0, \alpha+1)$, and B_α contains all the tasks that are executed at time α. Notice that $W_{-1} = B_{-1} = \emptyset$.

For a fixed value $\alpha \in \{-1,\ldots,\bar{C}-2\}$, we note $\mathcal{S}(W_\alpha, B_\alpha)$ to be the set of tasks from $X_{\alpha+1} - W_\alpha$ that are schedulable at time $\alpha+1$ following W_α and B_α. Formally, $\mathcal{S}(W_\alpha, B_\alpha) = \{i \in X_{\alpha+1} - W_\alpha, \Gamma^-(i) \subseteq W_\alpha \text{ and } |\Gamma^-(i) \cap B_\alpha| \leq 1\}$.

Now, we observe that a set of tasks B can be scheduled at time $\alpha+1$ following W_α and B_α if $B \subseteq \mathcal{S}(W_\alpha, B_\alpha)$ with $|B| \leq m$ and there is no couple of tasks $(i,j) \in B^2$ with a same predecessor in B_α (*i.e.* for each couple of tasks $(i,j) \in B^2$, $\Gamma^-(i) \cap \Gamma^-(j) \cap B_\alpha = \emptyset$). We set then $C(W_\alpha, B_\alpha)$ to be the set of all the subsets of $\mathcal{S}(W_\alpha, B_\alpha)$ that fulfills all these conditions.

Lastly, we may reduce our study to active schedules without loss of generality; then we set $A(W_\alpha, B_\alpha)$ to be the set of the elements from $C(W_\alpha, B_\alpha)$ that are maximum by inclusion.

Lemma 2.3. *Let us consider that σ is an active feasible schedule of maximum lateness bounded by $\bar{L}$. For any value $\alpha \in \{-1,\ldots,\bar{C}-2\}$, $\mathcal{T}^\sigma_{\alpha+1} \in A(W_\alpha, B_\alpha)$.*

Proof. Let us suppose by contradiction that, for a fixed value $\alpha \in \{-1,\ldots,\bar{C}-2\}$, $\mathcal{T}^\sigma_{\alpha+1} \notin A(W_\alpha, B_\alpha)$. Since tasks from $\mathcal{T}^\sigma_{\alpha+1}$ are all schedulable together at time $\alpha+1$, $\mathcal{T}^\sigma_{\alpha+1} \in C(W_\alpha, B_\alpha)$, and thus $\mathcal{T}^\sigma_{\alpha+1} \in C(W_\alpha, B_\alpha) - A(W_\alpha, B_\alpha)$. The consequence is that $\mathcal{T}^\sigma_{\alpha+1}$ is not maximum for the inclusion in $C(W_\alpha, B_\alpha)$, and thus σ is not active, a contradiction. □

3. Description of the algorithm

This section is dedicated to the description of our algorithm. Section 3.1 describes the multistage graph $S(\mathcal{G})$ built, while Section 3.2 is devoted to the algorithm.

3.1. Description of the multistage graph

Our algorithm builds an associated multistage graph $S(\mathcal{G}) = (V, E)$ described as follows:
Nodes of $S(\mathcal{G})$. The set of nodes V is partitioned into $\bar{C}+1$ stages. For any value $\alpha \in \{-1, \dots, \bar{C}-1\}$, N_α is the set of nodes at stage α. A node $p \in N_\alpha$ is a triple $(Y(p), B(p), L(p))$, where $Y(p) \subseteq X_\alpha \cap X_{\alpha+1} - Z_\alpha$, $B(p) \subseteq Y(p) \cup Z_\alpha \subseteq \mathcal{T}$ and $L(p) \in \mathbb{Z} \cup \{+\infty\}$. The node p is associated to the feasible schedules $\sigma(p)$ of tasks from $W(p) = Y(p) \cup Z_\alpha$ ending at time $\alpha+1$ with tasks from $B(p)$ scheduled at time α. Thus, we also get $B(p) \subseteq X_\alpha$ and $|B(p)| \leq m$. Lastly, $L(p)$ is the minimum maximum lateness among all the feasible schedules associated to p. Moreover, $N_{-1} = \{s\}$ with $W(s) = B(s) = \emptyset$ and $L(s) = -\infty$.
Arcs of $S(\mathcal{G})$. For each value $\alpha \in \{0, \dots, \bar{C}-2\}$ and $(p, q) \in N_\alpha \times N_{\alpha+1}$, there is an arc $(p, q) \in E$ if the following conditions are fulfilled:

(A.1) Tasks from $W(q) = Y(q) \cup Z_{\alpha+1}$ are completed at time $\alpha+2$ with tasks from $B(q)$ executed at time $\alpha+1$ and those from $B(p)$ at time α. Thus, $W(p) \cup B(q) = W(q)$ and since tasks are executed once, $W(p) \cap B(q) = \emptyset$;
(A.2) Any task $i \in B(q)$ must be schedulable at time $\alpha+1$ and all the schedule considered are active, thus by Lemma 2.3, $B(q) \in A(W(p), B(p))$;
(A.3) The node s is a source of $S(\mathcal{G})$, thus for any node $p \in N_0$, $(s, p) \in E$.

Maximum Lateness of a node of $S(\mathcal{G})$. For any node $q \in N_\alpha$ with $\alpha \in \{0, \dots, \bar{C}-1\}$, let $\ell(q) = \alpha+1-\min_{i \in B(q)} d_i$ be the maximum lateness of the tasks from $B(q)$. Recall that these tasks are executed at time α.

For any value $\alpha \in \{-1, \dots, \bar{C}-1\}$ and $q \in N_\alpha$, $L(q)$ is the minimum maximum lateness of a schedule of tasks from $W(q) = Y(q) \cup Z_\alpha$ with $B(q) \subseteq W(q)$ scheduled at time α. L is defined as follows:

(1) by convention, $L(s) = -\infty$;
(2) for any value $\alpha \in \{0, \dots, \bar{C}-1\}$ and $q \in N_\alpha$,

$$L(q) = \max\left(\ell(q), \min_{p \in \Gamma^-(q)} L(p)\right).$$

Here, $\Gamma^-(q)$ is the set of the immediate predecessors of q in $S(\mathcal{G})$. Next lemma provides a technical property on L that will be considered below to evaluate the maximum lateness of a node of $S(\mathcal{G})$.

Lemma 3.1. *Let us consider a node* $q \in N_\alpha$ *with* $\alpha \in \{1, \dots, \bar{C}-1\}$ *and its predecessors* $\Gamma^-(q) = \{p_1, p_2, \dots, p_k\} \subseteq N_{\alpha-1}$. *Let us also consider the sequence of integers defined as* $L_0(q) = +\infty$ *and for* $i \in \{1, \dots, k\}$,

$$L_i(q) = \max\left(\ell(q), \min_{j \in \{1,\dots,i\}} L(p_j)\right).$$

Then, for any integer $i \in \{1, \dots, k\}$,

$$L_i(q) = \max(\ell(q), \min(L(p_i), L_{i-1}(q))).$$

Proof. For $i = 1$, by definition,

$$L_1(q) = \max(\ell(q), L(p_1)) = \max(\ell(q), \min(L(p_1), L_0(q))).$$

Let us consider now $i \in \{2, \dots, k\}$ and let $M_i = \min(L(p_i), L_{i-1}(q))$. Two cases are considered:

– If $\ell(q) \geq M_i$, let us suppose first that $M_i = L(p_i)$, then $\min_{j \in \{1,\dots,i\}} L(p_j) \leq \ell(q)$ and by definition $L_i(q) = \max(\ell(q), \min_{j \in \{1,\dots,i\}} L(p_j)) = \ell(q)$; Now, if we suppose that $M_i = L_{i-1}(q)$, then $L_{i-1}(q) \leq \ell(q)$ and thus $L_{i-1}(q) = \ell(q)$. Since $L(p_i) \geq L_{i-1}(q)$, we get $L(p_i) \geq \min_{j \in \{1,\dots,i-1\}} L(p_j)$ and thus $L_i(q) = L_{i-1}(q) = \ell(q)$.
– Else, $\ell(q) < M_i$ and thus $\min(L(p_i), L_{i-1}(q)) > \ell(q)$. Since $L_{i-1}(q) > \ell(q)$, $L_{i-1}(q) = \min_{j \in \{1,\dots,i-1\}} L(p_j)$; thus we obtain $\min_{j \in \{1,\dots,i\}} L(p_j) = \min(L_{i-1}(q), L(p_i)) = M_i$, and the lemma is proved.

□

3.2. Description of the algorithm

Algorithm 1 builds iteratively the multistage graph $S(\mathcal{G}) = (V, E)$. For any set of tasks $X \subseteq \mathcal{T}$, let $\mathcal{P}(X)$ be the set of all subsets of X including the empty set. This algorithm returns the minimum value of the maximum lateness if it is upper bounded by $\bar{L}$, $+\infty$ otherwise.

The algorithm is composed by three main sections. Lines 3–5 correspond to the initialization step. Lines 6 and 7 build all the possible nodes lines 8–18 build the arcs and delete all the non connected nodes. The evaluation of $L(q)$ at line 14 follows Lemma 3.1.

Algorithm 1: Minimum maximum lateness L_{opt} if $L_{\text{opt}} \leq \bar{L}$, $+\infty$ otherwise.

1 **Input:** A precedence graph $\mathcal{G} = (\mathcal{T}, \mathcal{A})$, release dates r and due dates d
2 **Output:** L_{opt} if $L_{\text{opt}} \leq \bar{L}$, $+\infty$ otherwise
3 **for** $\alpha \in \{0, 1, \ldots, \bar{C}\}$ **do**
4 Calculate X_α and Z_α
5 $N_{-1} = \{s = (\emptyset, \emptyset, -\infty)\}$, $V = N_{-1}$, $E = \emptyset$, $L_{\text{opt}} = +\infty$
6 **for** $\alpha \in \{0, \ldots, \bar{C} - 1\}$ **do**
7 $N_\alpha = \{p = (Y, B, L), Y \in \mathcal{P}(X_\alpha \cap X_{\alpha+1} - Z_\alpha), B \in \mathcal{P}(X_\alpha \cap (Y \cup Z_\alpha)), |B| \leq m \text{ and } L = +\infty\}$
8 **for** $\alpha \in \{-1, \ldots, \bar{C} - 2\}$ **do**
9 **for** $p \in N_\alpha$ **do**
10 **if** $Y(p) \cup Z_\alpha \neq \mathcal{T}$ **then**
11 **for** $B \in A(Y(p) \cup Z_\alpha, B(p))$ **do**
12 Find $q \in N_{\alpha+1}$ such that $(Y(p) \cup Z_\alpha \cup B, B) = (Y(q) \cup Z_{\alpha+1}, B(q))$
13 $\ell(q) = \alpha + 2 - \min_{i \in B(q)} d_i$
14 $L(q) = \max(\ell(q), \min(L(p), L(q)))$
15 $E = E \cup \{(p, q)\}$
16 **else**
17 $L_{\text{opt}} = \min(L_{\text{opt}}, L(p))$
18 $N_{\alpha+1} = \{q \in N_{\alpha+1}, \Gamma^-(q) \neq \emptyset\}$, $V = V \cup N_{\alpha+1}$
19 **return** L_{opt}

Figure 3 presents the graph $S(\mathcal{G})$ built by Algorithm 1 associated to the example shown by Figure 1 and the upper bound $\bar{L} = 2$. Algorithm 1 returns the optimal value $L_{\text{opt}} = 1$.

We observe that the schedule presented by Figure 2 is associated to the path $s \to p_0 \to p_1 \to p_2 \to p_3 \to p_4$ with $p_0 = (\{1, 2\}, \{1, 2\}, 0)$, $p_1 = (\{1, 2, 3\}, \{3\}, 1)$, $p_2 = (\{4, 5\}, \{4, 5\}, 1)$, $p_3 = (\{6\}, \{6\}, 1)$ and $p_4 = (\emptyset, \{7\}, 2)$. The maximum lateness of the path is $L(p_4) = 2$.

Conversely, the path $s \to q_0 \to q_1 \to q_2 \to q_3$ with $q_0 = (\{1, 3\}, \{1, 3\}, 0)$, $q_1 = (\{1, 2, 3, 4\}, \{2, 4\}, 0)$, $q_2 = (\{4, 5, 6\}, \{5, 6\}, 1)$ and $q_3 = (\{6, 7\}, \{7\}, 1)$ corresponds to the active feasible schedule σ of maximum lateness $L(\sigma) = 1$ presented by Figure 4.

4. Correctness of Algorithm 1

This section is devoted to the proof of the correctness of Algorithm 1. Lemma 4.1 shows that the evaluation of the maximum lateness $L(q)$ for each node $q \in N$ is correct with respect to the definition of Section 3.1. Lemma 4.2 shows that any active feasible schedule is associated to a path of $S(\mathcal{G})$ from s to a node without successor, while Lemma 4.4 proves that extremum paths of $S(\mathcal{G})$ are associated to feasible schedules. Our main theorem follows.

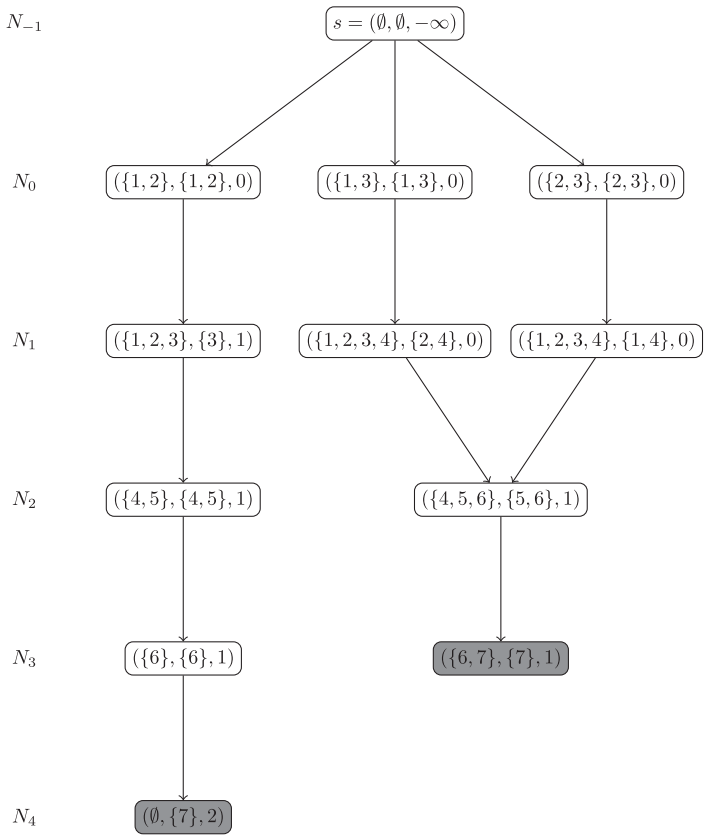


FIGURE 3. The multistage auxiliary graph $S(\mathcal{G}) = (V, E)$ associated with the example given in Figure 1. Each node $p \in N_\alpha$ is designated by the triple $(Y(p), B(p), L(p))$. The nodes p filled in gray are associated to a set of feasible schedules which minimum maximum lateness is $L(p)$.

FIGURE 4. The active feasible schedule σ of maximum lateness $L(\sigma) = 1$ associated to the path $s \to q_0 \to q_1 \to q_2 \to q_3$ with $q_0 = (\{1,3\},\{1,3\},0)$, $q_1 = (\{1,2,3,4\},\{2,4\},0)$, $q_2 = (\{4,5,6\},\{5,6\},1)$ and $q_3 = (\{6,7\},\{7\},1)$.

Lemma 4.1. *For any node $q \in N_\alpha$ with $\alpha \in \{-1, \ldots, \bar{C} - 1\}$, the value $L(q)$ computed by Algorithm 1 follows the definition of the minimum maximum lateness of a node (see, Sect. 3.1).*

Proof. $L(s)$ is set to $-\infty$ and is not modified. Let us consider a node $q \in N_\alpha$ of $S(\mathcal{G})$ with $\alpha \in \{0, \ldots, \bar{C} - 1\}$ and the value $\ell(q) = \alpha + 1 - \min_{i \in B(q)} d_i$. If $\alpha = 0$, $L(q) = \ell(q)$ is correctly set at line 5 of Algorithm 1.

Now, if $\alpha > 0$, $L(q) = +\infty$ at the initialization of the node q. Since q belongs to $S(\mathcal{G})$, then $\Gamma^-(q) \neq \emptyset$, thus the value $L(q)$ is adjusted once for each predecessor of q. Let $\Gamma^-(q) = \{p_1, \ldots, p_k\} \subseteq N_{\alpha-1}$ be the predecessors of q numbered following the inner loop lines 9–17. Let us denote by $L_i(q)$ the value of $L(q)$ associated to p_i for $i \in \{1, \ldots, k\}$. Following line 14, $L_i(q) = \max(\ell(q), \min(L(p_i), L_{i-1}(q))$. Thus, by Lemma 3.1,

$L_i(q) = \max(\ell(q), \min_{j\in\{1,\dots,i\}} L(p_j))$ and $L_k(p)$ is the minimum maximum lateness of q, which proves the lemma. □

Lemma 4.2. *Any active feasible schedule σ of maximum lateness bounded by $\bar{L}$ is associated to a path $\nu(\sigma)$ of $S(\mathcal{G})$ ending at a node $p \in N_\alpha$ with $\alpha \in \{0,\dots,\bar{C}-1\}$ and $W(p) = Y(p) \cup Z_\alpha = \mathcal{T}$. Moreover, the maximum lateness $L(\sigma) = \max_{q\in\nu(\sigma)} L(q)$.*

Proof. Let us consider an active feasible schedule of maximum lateness bounded by $\bar{L}$. Let us denote by $C(\sigma)$ the length of the schedule σ, *i.e.* $C(\sigma) = \max_{i\in\mathcal{T}}(t_i^\sigma + 1)$. Clearly, since σ is active, $C(\sigma) \leq \bar{C}$. For $\alpha \in \{0,\dots,C(\sigma)-1\}$, we set $W_\alpha = \bigcup_{\beta=0}^{\alpha} \mathcal{T}_\beta^\sigma$ and $B_\alpha = \mathcal{T}_\alpha^\sigma$.

The set $A(\emptyset,\emptyset)$ contains the maximum sets of tasks schedulable at time 0. Since σ is a feasible active schedule, there exists $q_0 \in N_0$ such that $W(q_0) = Y(q_0) \cup Z_0 = B(q_0) = W_0 = B_0$. Moreover, $(s, q_0) \in A$ and thus A.3 is verified.

Moreover, since σ is feasible, for every value $\alpha \in \{0,\dots,C(\sigma)-1\}$, $\mathcal{T}_\alpha^\sigma \subseteq X_\alpha$. According to Lemma 2.2, $\bigcup_{\beta=0}^{\alpha} \mathcal{T}_\beta^\sigma - Z_\alpha \subseteq X_\alpha \cap X_{\alpha+1}$. So the node $q_\alpha = (I(q_\alpha), B(q_\alpha), +\infty)$ has been built at stage α in the loop of lines 6 and 7 of Algorithm 1.

We prove that for every value $\alpha \in \{0,\dots,C(\sigma)-2\}$, $(q_\alpha, q_{\alpha+1}) \in A$.

- $Y(q_{\alpha+1}) \cup Z_{\alpha+1} = W(q_{\alpha+1}) = \bigcup_{\beta=0}^{\alpha+1} \mathcal{T}_\beta^\sigma = \bigcup_{\beta=0}^{\alpha} \mathcal{T}_\beta^\sigma \cup \mathcal{T}_{\alpha+1}^\sigma = W(q_\alpha) \cup B(q_{\alpha+1}) = Y(q_\alpha) \cup Z_\alpha \cup B(q_{\alpha+1})$. Moreover, $W(q_\alpha) \cap B(q_{\alpha+1}) = \bigcup_{\beta=0}^{\alpha} \mathcal{T}_\beta^\sigma \cap \mathcal{T}_{\alpha+1}^\sigma = \emptyset$. Thus, A.1 is verified.
- Since σ is a feasible active schedule, A.2 is verified.

We conclude that $(s, q_0, q_1, \dots, q_{C(\sigma)-1})$ is a path of $S(\mathcal{G})$. Moreover, $Y(q_{C(\sigma)-1}) \cup Z_{C(\sigma)-1} = W(q_{C(\sigma)-1}) = \mathcal{T}$ since the schedule σ ends at time $C(\sigma)$, thus $p = q_{C(\sigma)-1}$ is an ending node.

Lastly, by Lemma 4.1, each value $L(q_\alpha)$ computed by Algorithm 1 for $\alpha \in \{0,\dots,C(\sigma)-1\}$ is the minimum maximum lateness of the sub-schedule associated to q_α; the maximum lateness of the schedule σ is thus $L(\sigma) = \max_{q\in\nu(\sigma)} L(q)$, which concludes the proof. □

Lemma 4.3. *Let $(s, p_0, p_1, \dots, p_{C-1})$ be a path of $S(\mathcal{G})$ with $Y(p_{C-1}) \cup Z_{C-1} = W(p_{C-1}) = \mathcal{T}$. For each task $i \in \mathcal{T}$, there exists a unique value $\alpha \in \{0,\dots,C-1\}$ such that $i \in B(p_\alpha)$.*

Proof. According to the definition of $S(\mathcal{G})$, $W(p_0) \subseteq W(p_1) \subseteq \cdots \subseteq W(p_{C-1})$. Moreover, by assumption, $W(p_{C-1}) = \mathcal{T}$. Thus, for each task $i \in \mathcal{T}$, there is a unique $\alpha \in \{0,\dots,C-1\}$ with $i \in W(p_\alpha)$ and $i \notin W(p_{\alpha-1})$. Since $W(p_{\alpha-1}) \cup B(p_\alpha) = W(p_\alpha)$, we get $i \in B(p_\alpha)$. □

Lemma 4.4. *Each node $p \in N_\alpha$ such that $\alpha \in \{0,\dots,\bar{C}-1\}$ and $Y(p) \cup Z_\alpha = W(p) = \mathcal{T}$ is associated to an active feasible schedule σ of maximum lateness $L(\sigma) = L(p)$.*

Proof. Let us consider a node $p \in N_{C-1}$ with $W(p) = \mathcal{T}$ and $C \in \{0,\dots,\bar{C}-1\}$. We build iteratively a sequence of nodes $p_{-1}, p_0, p_1, \dots, p_{C-1}$ of $S(\mathcal{G})$ as follows:

(1) $p_{C-1} = p$;
(2) for each $k \in \{1,\dots C-1\}$, p_{k-1} is a predecessor of p_k in $S(\mathcal{G})$ such that $L(p_{k-1})$ is minimum;
(3) $p_{-1} = s$.

This sequence is defined since each node of N_α with $\alpha \in \{0,\dots,\bar{C}-1\}$ has at least one predecessor (or it will be deleted at line 18). Moreover, $p_k \in N_k$ for $k \in \{-1,\dots,C-1\}$.

Now, by Lemma 4.3, for each task $i \in \mathcal{T}$, there exists a unique value $\alpha \in \{0,\dots,C-1\}$ with $i \in B(p_\alpha)$. Thus, a starting time can be defined for i by setting $t_i^\sigma = \alpha$. We prove in the following that these starting times define an active feasible schedule σ.

We first observe that for each value $\alpha \in \{-1,\dots,C-1\}$, $B(p_\alpha) \subseteq X_\alpha$ and $|B(p_\alpha)| \leq m$. Thus, the maximum lateness of each task is bounded by $\bar{L}$; the constraints (2) and (7) of the problem definition are fulfilled.

Now, let consider a task $i \in B(p_\alpha)$ with $\alpha \in \{0, \dots, C-1\}$. By condition A.2, i is schedulable at time α. Thus, all its predecessors are belonging to $W(p_{\alpha-1})$ and the condition (4) of the problem definition is verified.

Moreover, $B(p_\alpha) \in A(W(p_{\alpha-1}), B(p_{\alpha-1}))$. Thus, there is at least one predecessor j of i scheduled at time $\alpha - 1$ and j has no other successor scheduled at time α. Thus, conditions (5) and (6) of the problem definition are validated.

Lastly, elements from $A(W(p_\alpha), B(p_\alpha))$ for $\alpha \in \{0, \dots, C-1\}$ are maximum by inclusion; this condition guarantees that σ is an active schedule.

Now, by definition of the sequence p_k, $L(p_k) = \max(\ell(p_k), L(p_{k-1}))$ with $\ell(p_k) = k + 1 - \min_{i \in B(p_k)} d_i$. Thus, $L(p_0) \leq L(p_1) \dots \leq L(p_{C-1})$. By Lemma 4.1, the maximum lateness of the schedule σ is $L(\sigma) = L(p_{C-1}) = L(p)$ and the lemma is proved. □

Theorem 4.5 (Correctness of Algorithm 1)**.** *Algorithm 1 returns the minimum maximum lateness $L(\sigma) \leq \bar{L}$ of a feasible schedule σ if it exists, $+\infty$ if there is no feasible schedule of maximum lateness bounded by $\bar{L}$.*

Proof. Let us suppose first that Algorithm 1 returns a value $L^\star$. Then, let a node $p \in N$ such that $L^\star = L(p) = \min\{L(p), p \in N, W(p) = \mathcal{T}\}$. By Lemma 4.4, p is associated to an active feasible schedule of maximum lateness $L(p)$, thus the minimum maximum lateness of our instance $L_{\text{opt}} \leq L^\star$. Now, let us suppose by contradiction that $L^\star > L_{\text{opt}}$. Thus, there exists an active feasible schedule σ such that $L(\sigma) = L_{\text{opt}}$ and σ is not associated to a path of $S(\mathcal{G})$, which contradicts Lemma 4.2, and thus $L^\star = L_{\text{opt}}$.

Now, let us suppose that there is no node $p \in N$ such that $W(p) = \mathcal{T}$; in this case, Algorithm 1 returns $+\infty$. By Lemma 4.2, there is no active feasible schedule and the theorem is proved. □

5. Complexity analysis

We study in this section the complexity of Algorithm 1 to conclude that our scheduling problem is fixed-parameter tractable in the pathwidth.

Lemma 5.1. *Let us denote by n the number of tasks and $pw(\bar{L})$ the pathwidth associated to the upper bound $\bar{L}$ of the maximum lateness. For any value $\alpha \in \{0, \dots, \bar{C}-1\}$, the number of elements of N_α belongs to $\mathcal{O}(2^{2pw(\bar{L})})$.*

Proof. By Algorithm 1, the number of nodes in N_α for $\alpha \in \{0, \dots, \bar{C}-1\}$ is bounded by $2^{|X_\alpha|} \times 2^{|X_{\alpha+1}|}$. The values $|X_\alpha|$ and $|X_{\alpha+1}|$ are both bounded by $pw(\bar{L}) + 1$, thus the lemma holds. □

Lemma 5.2. *The time complexity of the inner loop of Algorithm 1 (lines 9–17) for a fixed node $p \in N_\alpha$ and $\alpha \in \{-1, \dots, \bar{C}-2\}$ is $\mathcal{O}(pw(\bar{L}) \times 2^{pw(\bar{L})})$.*

Proof. Let us suppose that $Y(p) \cup Z_\alpha = W(p) \neq \mathcal{T}$. By definition, $A(W(p), B(p)) \subseteq \mathcal{P}(X_{\alpha+1})$ and thus $|A(W(p), B(p))| \leq 2^{|X_{\alpha+1}|}$.

Searching for a node q in $N_{\alpha+1}$ can be done in time $\mathcal{O}(\log |N_{\alpha+1}|)$; By Lemma 5.1, $\mathcal{O}(\log |N_{\alpha+1}|) \subseteq \mathcal{O}(pw(\bar{L}))$, thus the overall time of the inner loop is $\mathcal{O}(pw(\bar{L}) \times 2^{pw(\bar{L})})$, and the lemma is proved. □

Theorem 5.3 (Complexity of Algorithm 1)**.** *The time complexity of Algorithm 1 is $\mathcal{O}(n^2 + n \times pw(\bar{L}) \times 2^{3pw(\bar{L})})$, where $pw(\bar{L})$ is the pathwidth of the interval graph associated to the time windows $(r_i, d_i + \bar{L})$, $i \in \mathcal{T}$. The space complexity of this algorithm is $\mathcal{O}(n \times 2^{4pw(\bar{L})})$.*

Proof. The time complexity of the computation of the sets X_α and Z_α for $\alpha \in \{0, \dots, \bar{C}\}$ (lines 3 and 4) is $\mathcal{O}(n^2)$ since $\bar{C}$ is bounded by $4n - 2$ following Lemma 2.1.

The time complexity for building V at lines 7 and 8 is $\mathcal{O}(n \times 2^{2pw(\bar{L})})$ by Lemmas 5.1 and 2.1. Following Lemma 5.2, the whole complexity of building arcs of $S(\mathcal{G})$ in lines 9–20 is $\mathcal{O}(n \times 2^{2pw(\bar{L})} \times pw(\bar{L}) \times 2^{pw(\bar{L})})$.

The overall complexity of the algorithm is thus $\mathcal{O}(n^2 + n \times pw(\bar{L}) \times 2^{3pw(\bar{L})})$, and the first part of the theorem holds.

Let us consider now the evaluation of the space complexity. By Lemma 5.1, the number of nodes $|N_\alpha|$ with $\alpha \in \{0, \ldots, \bar{C} - 1\}$ belongs to $\mathcal{O}(2^{2pw(\bar{L})})$. Thus, the total number of vertices of $S(\mathcal{G})$, $|V| = 1 + \sum_{\alpha=0}^{\bar{C}-1} |N_\alpha|$ is in $\mathcal{O}(\bar{C} \times 2^{2pw(\bar{L})})$. By Lemma 2.1, we deduce that $|V|$ belongs to $\mathcal{O}(n \times 2^{2pw(\bar{L})})$. Moreover, the number of arcs $|E|$ is bounded by $|N_0| + \sum_{\alpha=1}^{\bar{C}-2} |N_\alpha| \times |N_{\alpha+1}|$, which is in $\mathcal{O}(n \times 2^{4pw(\bar{L})})$ achieving the proof. □

6. Conclusion and perspectives

We proved in this paper that the scheduling problem $P|r_i, \text{prec}, p_i = 1, c_{ij} = 1|L_{\max}$ is fixed-parameter tractable in the pathwidth $pw(\bar{L})$ of the interval graph $\mathcal{I}(\bar{L})$ associated with the intervals $(r_i, d_i + \bar{L})$, $i \in \mathcal{T}$. We extended previous approaches [17,21] to tackle both communications delay, a limited number of machines, and to optimize the maximum lateness. We also limit our enumeration to active schedules, which allows to decrease the worst-case complexity of the method.

We believe that this work opens up many perspectives. From a theoretical point of view, many fundamental questions remain open as the existence of a fixed-parameter algorithm in the width, or the possible extension of this work to scheduling problems with large communication delays. From a practical point of view, our algorithm defines an original exploration scheme probably well suited to general scheduling problems. Similarly to branch-and-bound methods, dominance properties allow to reduce the size of the generated multistage graph. It would then be interesting to test this new class of algorithms to compare their performance with those from the literature.

Acknowledgements. We are very grateful to the two reviewers for their helpful recommendations.

References

[1] A. Ait El Cadi, R. Ben Atitallah, S. Hanafi, N. Mladenovic and A. Artiba, New MIP model for multiprocessor scheduling problem with communication delays. *Optim. Lett.* **11** (2017) 1091–1107.

[2] H.L. Bodlaender, A tourist guide through treewidth. *Acta Cybern.* **11** (1992) 1–21.

[3] H.L. Bodlaender and M.R. Fellows, W[2]-hardness of precedence constrained k-processor scheduling. *Oper. Res. Lett.* **18** (1995) 93–97.

[4] P. Chrétienne and C. Picouleau, Scheduling with communication delays: a survey, in Scheduling Theory and its Applications. John Wiley & Sons, New York (1995) 65–90.

[5] M. Cygan, F.V. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk and S. Saurabh, Parameterized Algorithms, 1st edition. Springer Publishing Company, Incorporated (2015).

[6] T. Davidović, L. Liberti, N. Maculan and N. Mladenovic, Towards the Optimal Solution of the Multiprocessor Scheduling Problem with Communication Delays. MISTA Conference (2007).

[7] M. de Weerdt, R. Baart and L. He, Single-machine scheduling with release times, deadlines, setup times, and rejection. *Eur. J. Oper. Res.* **291** (2021) 629–639.

[8] R.G. Downey and M.R. Fellows, Fundamentals of Parameterized Complexity. Springer, London (2013).

[9] M. Drozdowski, Scheduling for Parallel Processing. Springer (2009).

[10] J. Du, J.Y.-T. Leung and G.H. Young, Scheduling chain-structured tasks to minimize makespan and mean flow time. *Inf. Comput.* **92** (1991) 219–236.

[11] R. Giroudeau and J.-C. Koenig, Scheduling with communication delays, in Multiprocessor Scheduling, edited by E. Levner. IntechOpen, Rijeka (2007).

[12] R.L. Graham, Bounds for certain multiprocessing anomalies. *Bell Syst. Tech. J.* **45** (1966) 1563–1581.

[13] R.L. Graham, E.L. Lawler, J.K. Lenstra and A.H.G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: a survey, in Discrete Optimization II. *Annals of Discrete Mathematics*, edited by P.L. Hammer, E.L. Johnson and B.H. Korte. Vol. 5. Elsevier (1979) 287–326.

[14] E. Günther, F.G. König and N. Megow, Scheduling and packing malleable and parallel tasks with precedence constraints of bounded width. *J. Comb. Optim.* **27** (2014) 164–181.

[15] H. Hoogeveen, J.K. Lenstra and B. Veltman, Three, four, five, six, or the complexity of scheduling with communication delays. *Oper. Res. Lett.* **16** (1994) 129–137.

[16] M. Mnich and R. Van Bevern, Parameterized complexity of machine scheduling: 15 open problems. *Comput. Oper. Res.* **100** (2018) 254–261.

[17] A. Munier Kordon, A fixed-parameter algorithm for scheduling unit dependent tasks on parallel machines with time windows. *Discrete Appl. Math.* **290** (2021) 1–6.

[18] V.J. Rayward-Smith, Uet scheduling with unit interprocessor communication delays. *Discrete Appl. Math.* **18** (1987) 55–71.
[19] L. Schrage, Solving resource-constrained network problems by implicit enumeration – nonpreemptive case. *Oper. Res.* **18** (1970) 263–278.
[20] J.P. Sousa and L.A. Wolsey, A time indexed formulation of non-preemptive single machine scheduling problems. *Math. Program.* **54** (1992) 353–367.
[21] N. Tang and A.M. Kordon, A fixed-parameter algorithm for scheduling unit dependent tasks with unit communication delays, in European Conference on Parallel Processing. *Lecture Notes in Computer Science*. Vol. 12820. Springer (2021) 105–119.
[22] R. van Bevern, R. Bredereck, L. Bulteau, C. Komusiewicz, N. Talmon and G.J. Woeginger, Precedence-constrained scheduling problems parameterized by partial order width, in International Conference on Discrete Optimization and Operations Research. Springer International Publishing (2016) 105–120.
[23] B. Veltman, *Multiprocessor scheduling with communication delays*, Ph.D. thesis. Eindhoven University of Technology (1993).
[24] B. Veltman, B.J. Lageweg and J.K. Lenstra, Multiprocessor scheduling with communication delays. *Parallel Comput.* **16** (1990) 173–182.
[25] S. Venugopalan and O. Sinnen, Ilp formulations for optimal task scheduling with communication delays on parallel systems. *IEEE Trans. Parallel Distrib. Syst.* **26** (2015) 142–151.
[26] Y. Zinder, B. Su, G. Singh and R. Sorli, Scheduling uet-uct tasks: Branch-and-bound search in the priority space. *Optim. Eng.* **11** (2010) 627–646.