Parameterized Complexity and New Efficient Enumerative Schemes for RCPSP

Maher Mallem[1], Claire Hanen[2,3], Alix Munier-Kordon[2]

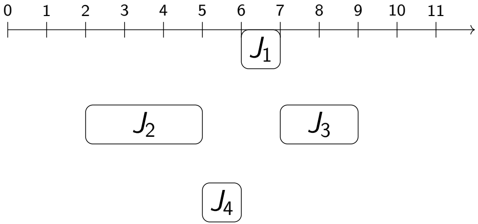
[1]Inria, CNRS, ENS de Lyon, Université Claude Bernard Lyon 1, LIP, UMR 5668, 69342, Lyon cedex 07, France,
[2]Sorbonne Université, CNRS, LIP6, F-75005 Paris, France,
[3]Université Paris Nanterre, F-92000 Nanterre, France,

18 November 2025

Outline

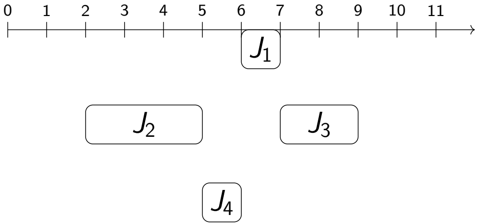
Outline

Scheduling



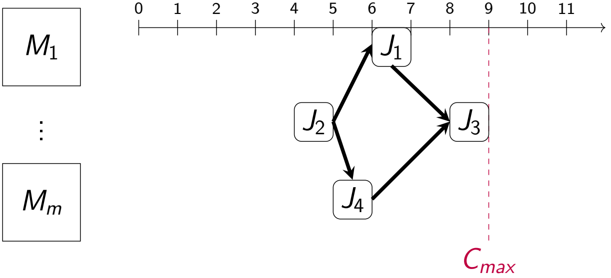
$\mathcal{J} = \{J_1, \ldots, J_n\}$ set of n jobs of processing time $p_1, \ldots, p_n$

Scheduling



Notation: machines | job properties | optimization criterion

Scheduling



$$P|prec, p_j = 1|C_{max}$$

Resource-Constrained Project Scheduling Problem (RCPSP)

- INPUT:
  - a set $\mathcal{J}$ of n jobs J_j with processing time p_j,
  - a precedence graph *prec* on $\mathcal{J}$,
  - a set $\mathcal{R}$ renewable resources,
  - for each resource $\rho \in \mathcal{R}$ the available amount $\mathcal{R}_\rho$,
  - for each job $j \in \mathcal{J}$ the amount $res_{j,\rho} \leq \mathcal{R}_\rho$ of resource $\rho \in \mathcal{R}$ used during its execution.

Resource-Constrained Project Scheduling Problem (RCPSP)

- INPUT:
  - a set $\mathcal{J}$ of n jobs J_j with processing time p_j,
  - a precedence graph *prec* on $\mathcal{J}$,
  - a set $\mathcal{R}$ renewable resources,
  - for each resource $\rho \in \mathcal{R}$ the available amount $\mathcal{R}_\rho$,
  - for each job $j \in \mathcal{J}$ the amount $res_{j,\rho} \leq \mathcal{R}_\rho$ of resource $\rho \in \mathcal{R}$ used during its execution.
- OUTPUT: a schedule $\tau : \mathcal{J} \to \mathbb{N}_0$ such that:
  - for every relation $i \to j$ in *prec* job i finishes before job j starts,
  - at any time t at most $\mathcal{R}_\rho$ units of each resource ρ are used,
  - makespan $C_{max} = max_{j \in \mathcal{J}}(\tau(j) + p_j)$ is minimized.

Resource-Constrained Project Scheduling Problem (RCPSP)

- INPUT:
  - a set $\mathcal{J}$ of n jobs J_j with processing time p_j,
  - a precedence graph *prec* on $\mathcal{J}$,
  - a set $\mathcal{R}$ renewable resources,
  - for each resource $\rho \in \mathcal{R}$ the available amount $\mathcal{R}_\rho$,
  - for each job $j \in \mathcal{J}$ the amount $res_{j,\rho} \leq \mathcal{R}_\rho$ of resource $\rho \in \mathcal{R}$ used during its execution.
- OUTPUT: a schedule $\tau : \mathcal{J} \to \mathbb{N}_0$ such that:
  - for every relation $i \to j$ in *prec* job i finishes before job j starts,
  - at any time t at most $\mathcal{R}_\rho$ units of each resource ρ are used,
  - makespan $C_{max} = max_{j \in \mathcal{J}}(\tau(j) + p_j)$ is minimized.

- $|\mathcal{R}| = 1$, unit p_j and $res_{j,\rho}$: $P|prec, p_j = 1|C_{max}$.

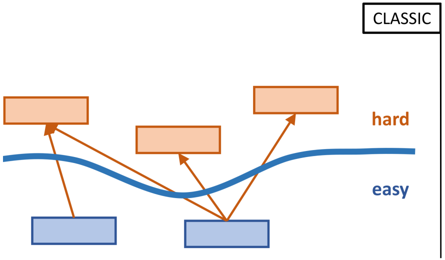
Resource-Constrained Project Scheduling Problem (RCPSP)

- INPUT:
  - a set $\mathcal{J}$ of n jobs J_j with processing time p_j,
  - a precedence graph *prec* on $\mathcal{J}$,
  - a set $\mathcal{R}$ renewable resources,
  - for each resource $\rho \in \mathcal{R}$ the available amount $\mathcal{R}_\rho$,
  - for each job $j \in \mathcal{J}$ the amount $res_{j,\rho} \leq \mathcal{R}_\rho$ of resource $\rho \in \mathcal{R}$ used during its execution.
- OUTPUT: a schedule $\tau : \mathcal{J} \rightarrow \mathbb{N}_0$ such that:
  - for every relation $i \rightarrow j$ in *prec* job i finishes before job j starts,
  - at any time t at most $\mathcal{R}_\rho$ units of each resource ρ are used,
  - makespan $C_{max} = max_{j \in \mathcal{J}}(\tau(j) + p_j)$ is minimized.

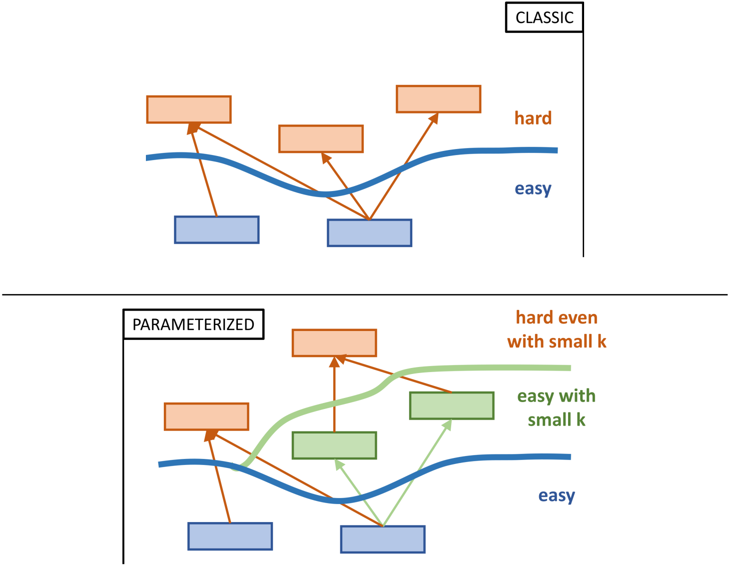
- $|\mathcal{R}| = 1$, unit p_j and $res_{j,\rho}$: $P|prec, p_j = 1|C_{max}$.

- $P|prec, p_j = 1|C_{max}$ is NP-complete [Ullman, 1975].

Motivation



Motivation

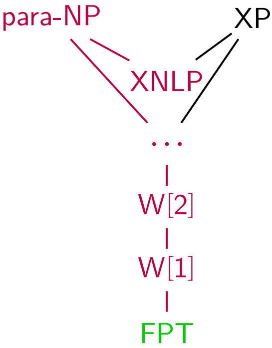

CLASSIC
hard
easy
PARAMETERIZED
hard even with small k
easy with small k
easy

Going beyond NP-hardness

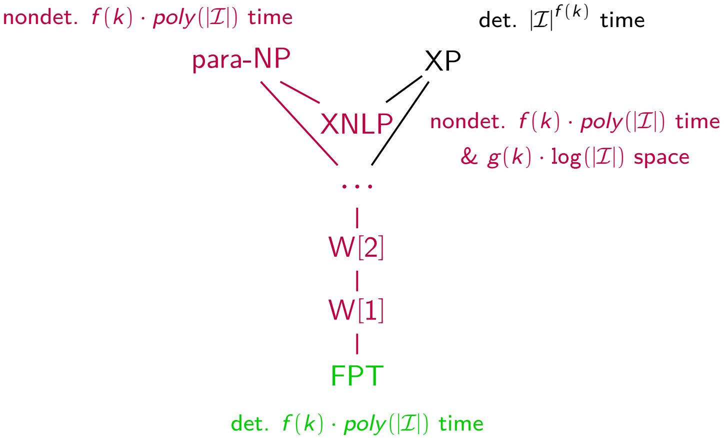
- Problem $\mathcal{P}$, instance $\mathcal{I}$.
- P: deterministic time $poly(|\mathcal{I}|)$.
- NP: nondeterministic time $poly(|\mathcal{I}|)$.

Going beyond NP-hardness

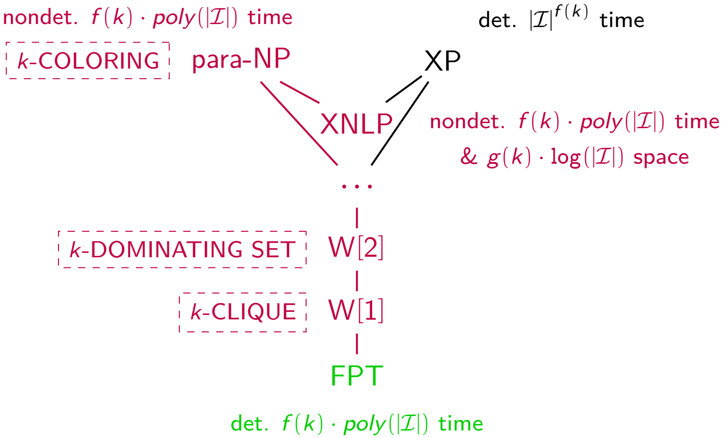
- Problem $\mathcal{P}$, instance $\mathcal{I}$.
  - Parameter $\bar{k}$, instance $\mathcal{I}$ with parameter value $\leq k$.
- P: deterministic time $poly(|\mathcal{I}|)$.
  - FPT: deterministic $f(k) \cdot poly(|\mathcal{I}|)$ time.
- NP: nondeterministic time $poly(|\mathcal{I}|)$.
  - para-NP: nondeterministic $f(k) \cdot poly(|\mathcal{I}|)$ time.


para-NP
XP
XNLP
...
W[2]
W[1]
FPT

Parameterized complexity classes



Parameterized complexity classes



Example of parameterized results

- Parameters
  - w: width of directed acyclic graph *prec*,
  - λ: maximum allowed lag.

Example of parameterized results

- Parameters
  - w: width of directed acyclic graph *prec*,
  - λ: maximum allowed lag.

- w: $P|prec, p_j = 1|C_{max}$ is XNLP-complete [Bodlaender et al., 2022].
- λ: $P|prec, p_j = 1|C_{max}$ is para-NP-complete [Lenstra et al., 1977].
- $w + \lambda$: RCPSP is in FPT [van Bevern et al., 2016].

Example of parameterized results

- Parameters
  - w: width of directed acyclic graph *prec*,
  - λ: maximum allowed lag.

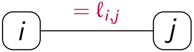
- w: $P|prec, p_j = 1|C_{max}$ is XNLP-complete [Bodlaender et al., 2022].
- λ: $P|prec, p_j = 1|C_{max}$ is para-NP-complete [Lenstra et al., 1977].
- $w + \lambda$: RCPSP is in FPT [van Bevern et al., 2016].

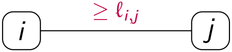
Objective

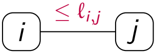
Apply this framework to other scheduling problems/parameters.

Outline

Precedence delays

- Given a feasible schedule and $(i \xrightarrow{\ell_{i,j}} j)$:
  - Exact delay: 

  - Minimum delay: 

  - Maximum delay: 

Precedence delays

- Given a feasible schedule and $(i \xrightarrow{\ell_{i,j}} j)$:
  - Exact delay: $\boxed{i} \overset{= \ell_{i,j}}{\longrightarrow} \boxed{j}$
  - Minimum delay: $\boxed{i} \overset{\geq \ell_{i,j}}{\longrightarrow} \boxed{j}$
  - Maximum delay: $\boxed{i} \overset{\leq \ell_{i,j}}{\longrightarrow} \boxed{j}$
- Notation: $prec(\ell^X)$ with $X \in \{ex, min, max\}$.

Precedence delays

- Given a feasible schedule and $(i \xrightarrow{\ell_{i,j}} j)$:
  - Exact delay: $i \xrightarrow{= \ell_{i,j}} j$
  - Minimum delay: $i \xrightarrow{\geq \ell_{i,j}} j$
  - Maximum delay: $i \xrightarrow{\leq \ell_{i,j}} j$
- Notation: $prec(\ell^X)$ with $X \in \{ex, min, max\}$.
- Parameter $\ell_{max} = max_{(i,j) \in prec}(\ell_{i,j})$.

Hardness with bounded precedence delays

- $1|chains(\ell^{ex}), \ell_{i,j} = \ell, p_j = 1|C_{max}$ parameterized by ℓ_{max} is W[1]-hard.

Hardness with bounded precedence delays

- $1|chains(\ell^{ex}), \ell_{i,j} = \ell, p_j = 1|C_{max}$ parameterized by ℓ_{max} is W[1]-hard.
  - $\leftrightarrow$ UNARY BIN PACKING parameterized by the number of bins [Jansen et al., 2013].

Hardness with bounded precedence delays

- $1|chains(\ell^{ex}), \ell_{i,j} = \ell, p_j = 1|C_{max}$ parameterized by ℓ_{max} is W[1]-hard.
  - $\leftrightarrow$ Unary bin packing parameterized by the number of bins [Jansen et al., 2013].
- $1|prec(\ell^{max}), \ell_{i,j} = \ell, p_j = 1|C_{max}$ parameterized by ℓ_{max} is W[t]-hard for all $t \in \mathbb{N}$.

Hardness with bounded precedence delays

- $1|chains(\ell^{ex}), \ell_{i,j} = \ell, p_j = 1|C_{max}$ parameterized by ℓ_{max} is W[1]-hard.
  - $\leftrightarrow$ Unary bin packing parameterized by the number of bins [Jansen et al., 2013].
- $1|prec(\ell^{max}), \ell_{i,j} = \ell, p_j = 1|C_{max}$ parameterized by ℓ_{max} is W[t]-hard for all $t \in \mathbb{N}$.
  - $\leftrightarrow$ k-Directed bandwidth [Bodlaender, 2021].

Hardness with bounded precedence delays

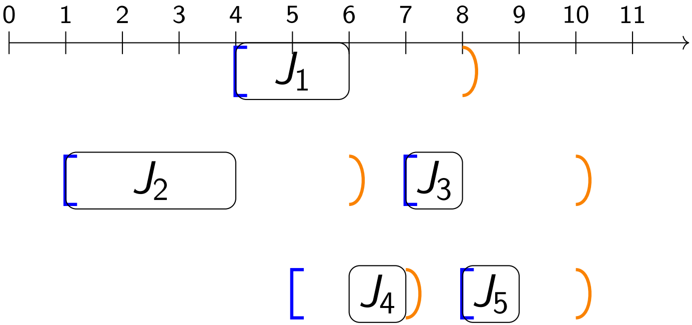
- $1|chains(\ell^{ex}), \ell_{i,j} = \ell, p_j = 1|C_{max}$ parameterized by ℓ_{max} is W[1]-hard.
  - $\leftrightarrow$ UNARY BIN PACKING parameterized by the number of bins [Jansen et al., 2013].
- $1|prec(\ell^{max}), \ell_{i,j} = \ell, p_j = 1|C_{max}$ parameterized by ℓ_{max} is W[t]-hard for all $t \in \mathbb{N}$.
  - $\leftrightarrow$ k-DIRECTED BANDWIDTH [Bodlaender, 2021].
- $1|prec(\ell^{min}), \ell_{i,j} = \ell, p_j = 1|C_{max}$ parameterized by $\ell_{max} + w$ is XNLP-complete.

Hardness with bounded precedence delays

- $1|chains(\ell^{ex}), \ell_{i,j} = \ell, p_j = 1|C_{max}$ parameterized by ℓ_{max} is W[1]-hard.
  - $\leftrightarrow$ UNARY BIN PACKING parameterized by the number of bins [Jansen et al., 2013].
- $1|prec(\ell^{max}), \ell_{i,j} = \ell, p_j = 1|C_{max}$ parameterized by ℓ_{max} is W[t]-hard for all $t \in \mathbb{N}$.
  - $\leftrightarrow$ k-DIRECTED BANDWIDTH [Bodlaender, 2021].
- $1|prec(\ell^{min}), \ell_{i,j} = \ell, p_j = 1|C_{max}$ parameterized by $\ell_{max} + w$ is XNLP-complete.
  - $\leftarrow$ $P|prec, p_j = 1|C_{max}$ parameterized by $m + w$ [Bodlaender et al., 2022].

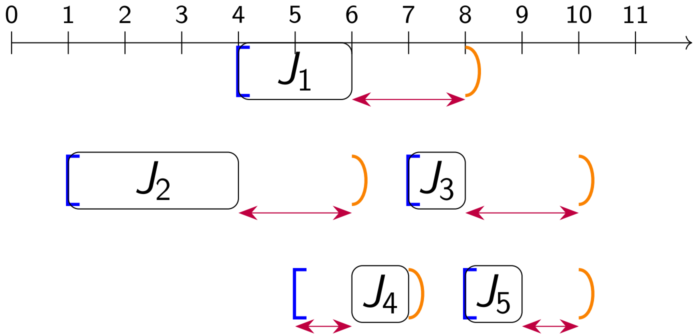
Outline

Time windows, slack σ and pathwidth pw



Release dates r_j, deadlines d_j.

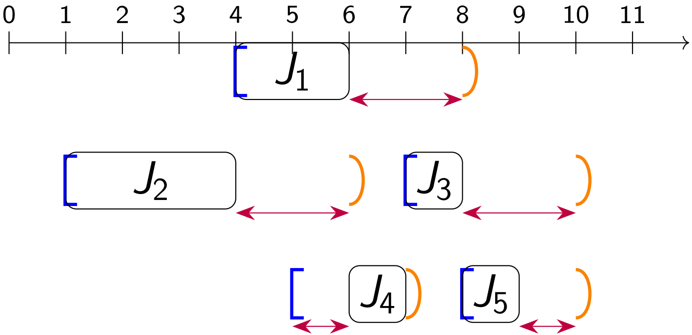
Time windows, slack σ and pathwidth *pw*



Release dates r_j, deadlines d_j.

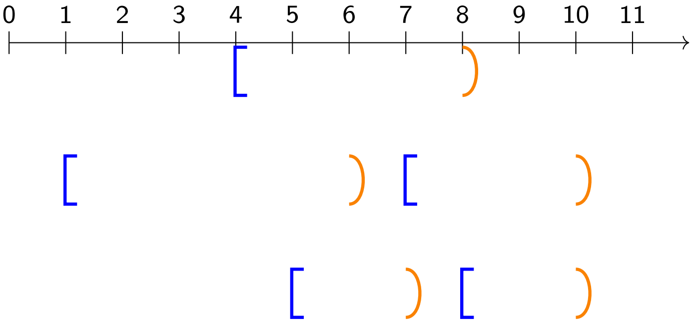
Time windows, slack σ and pathwidth pw

$\sigma = 2$



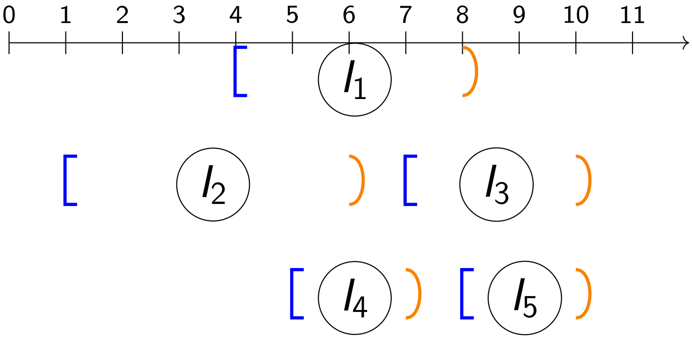
Release dates r_j, deadlines d_j.

Time windows, slack σ and pathwidth *pw*



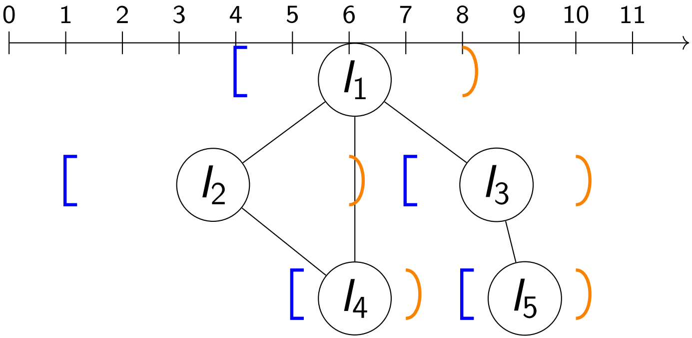
Release dates r_j, deadlines d_j.

Time windows, slack σ and pathwidth pw



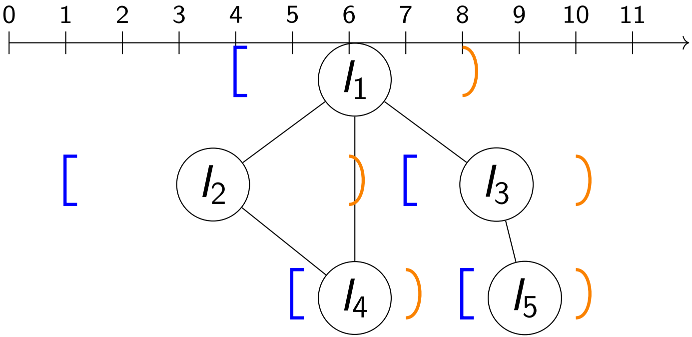
Release dates r_j, deadlines d_j.

Time windows, slack σ and pathwidth pw



Release dates r_j, deadlines d_j.

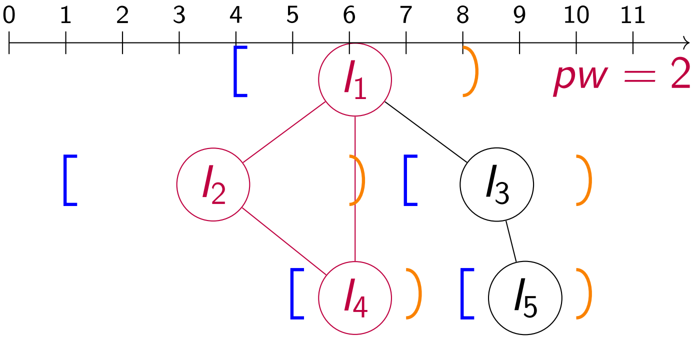
Time windows, slack σ and pathwidth *pw*



[Bodlaender and Möhring, 1993] For any interval graph G:

$$\omega(G) = \chi(G) = tw(G) + 1 = pw(G) + 1.$$

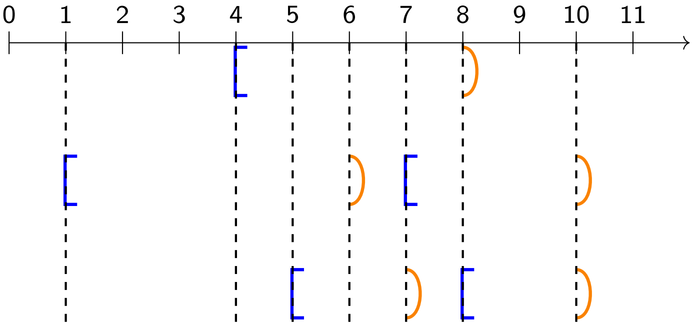
Time windows, slack σ and pathwidth *pw*



[Bodlaender and Möhring, 1993] For any interval graph G:

$$\omega(G) = \chi(G) = tw(G) + 1 = pw(G) + 1.$$

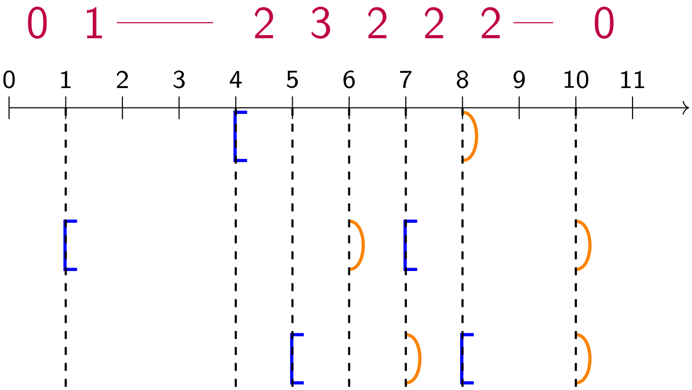
Time windows, slack σ and pathwidth *pw*



[Bodlaender and Möhring, 1993] For any interval graph G:

$$\omega(G) = \chi(G) = tw(G) + 1 = pw(G) + 1.$$

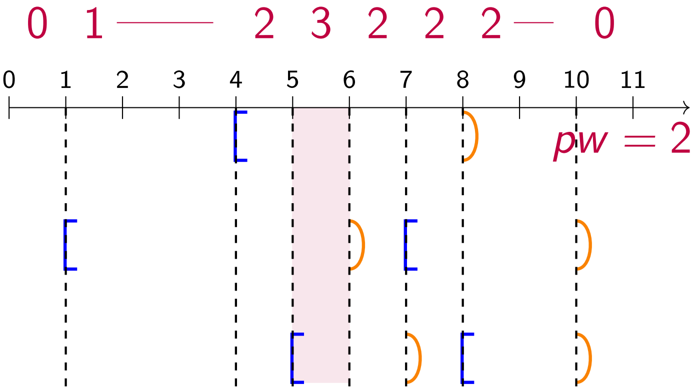
Time windows, slack σ and pathwidth *pw*



[Bodlaender and Möhring, 1993] For any interval graph G:

$$\omega(G) = \chi(G) = tw(G) + 1 = pw(G) + 1.$$

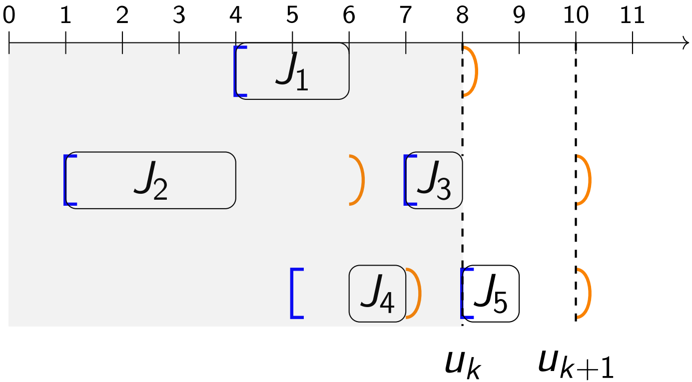
Time windows, slack σ and pathwidth *pw*



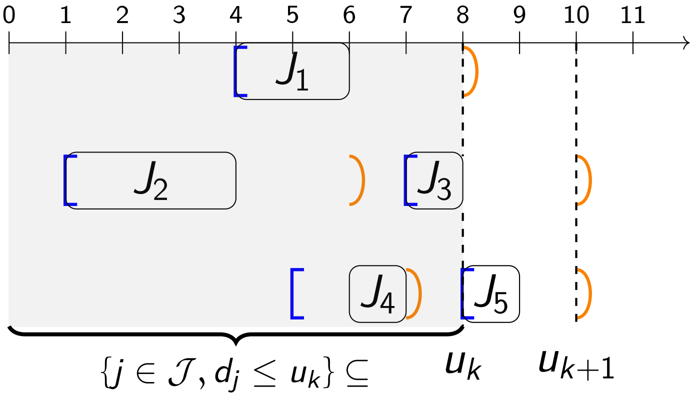
[Bodlaender and Möhring, 1993] For any interval graph G:

$$\omega(G) = \chi(G) = tw(G) + 1 = pw(G) + 1.$$

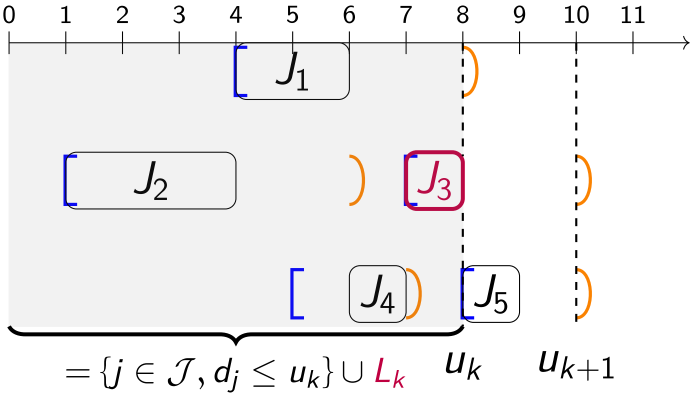
Time windows, slack σ and pathwidth *pw*



Time windows, slack σ and pathwidth *pw*

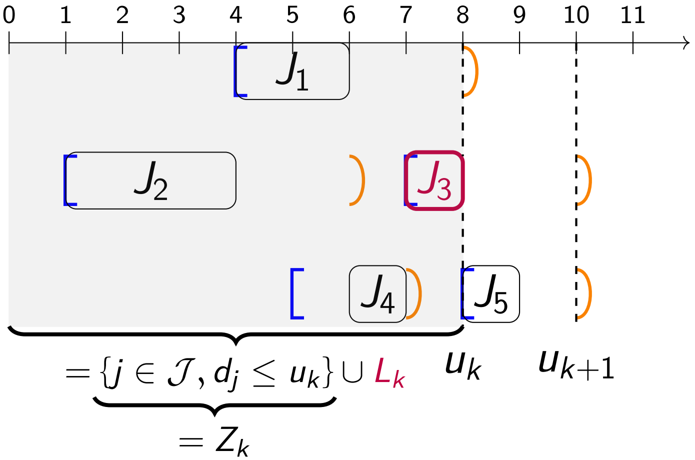


Time windows, slack σ and pathwidth *pw*



- $J_j \in L_k \implies [u_k, u_{k+1}) \subseteq [r_j, d_j)$: at most $pw + 1$ such jobs.

Time windows, slack σ and pathwidth *pw*



- $J_j \in L_k \implies [u_k, u_{k+1}) \subseteq [r_j, d_j)$: at most $pw + 1$ such jobs.

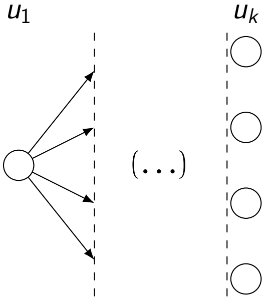
Dynamic programming with pathwidth pw

Dynamic programming state (k, L_k)

- u_k = associated release date/deadline value.
- $Z_k \cup L_k$ = set of jobs starting before u_k.
- State value = minimum makespan among the associated partial schedules.

Dynamic programming state (k, L_k)

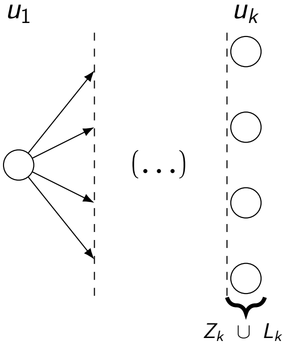
- u_k = associated release date/deadline value.
- $Z_k \cup L_k$ = set of jobs starting before u_k.
- State value = minimum makespan among the associated partial schedules.



Dynamic programming with pathwidth pw

Dynamic programming state (k, L_k)

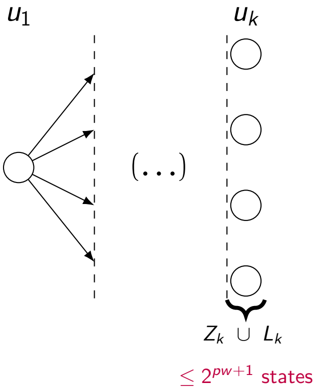
- u_k = associated release date/deadline value.
- $Z_k \cup L_k$ = set of jobs starting before u_k.
- State value = minimum makespan among the associated partial schedules.



Dynamic programming with pathwidth pw

Dynamic programming state (k, L_k)

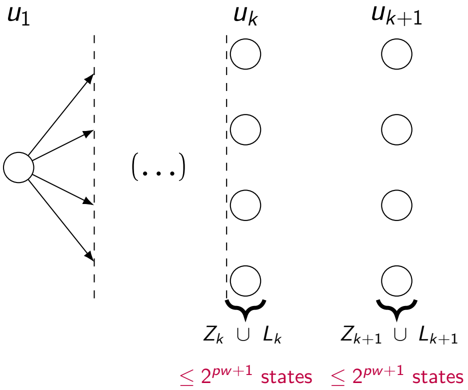
- u_k = associated release date/deadline value.
- $Z_k \cup L_k$ = set of jobs starting before u_k.
- State value = minimum makespan among the associated partial schedules.



Dynamic programming with pathwidth pw

Dynamic programming state (k, L_k)

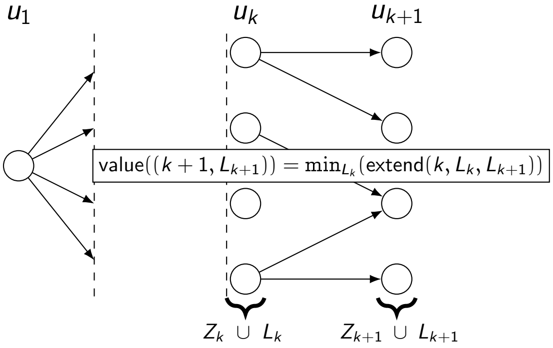
- u_k = associated release date/deadline value.
- $Z_k \cup L_k$ = set of jobs starting before u_k.
- State value = minimum makespan among the associated partial schedules.



Dynamic programming with pathwidth *pw*

Dynamic programming state (k, L_k)

- u_k = associated release date/deadline value.
- $Z_k \cup L_k$ = set of jobs starting before u_k.
- State value = minimum makespan among the associated partial schedules.

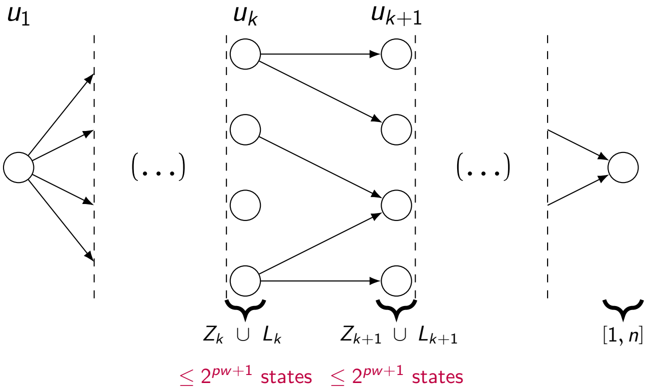


$\leq 2^{pw+1}$ states $\leq 2^{pw+1}$ states

Dynamic programming with pathwidth *pw*

Dynamic programming state (k, L_k)

- u_k = associated release date/deadline value.
- $Z_k \cup L_k$ = set of jobs starting before u_k.
- State value = minimum makespan among the associated partial schedules.



Some results with slack σ and pathwidth *pw*

- $1|prec, r_j, d_j|C_{max}$ param. by pw is in FPT.
  - $\mathcal{O}((pw+1)^2 4^{pw} n + n\log(n))$ time [Hanen et al., 2022].
- $P|prec, p_j = 1, r_j, d_j|C_{max}$ param. by pw is in FPT.
  - $\mathcal{O}(16^{pw} n^4)$ time [Munier Kordon, 2021].
  - $\mathcal{O}(pw 2^{pw} mn^3)$ time [Shi et al., 2026].

Some results with slack σ and pathwidth *pw*

- $1|prec, r_j, d_j|C_{max}$ param. by pw is in FPT.
  - $\mathcal{O}((pw+1)^2 4^{pw} n + n\log(n))$ time [Hanen et al., 2022].
- $P|prec, p_j = 1, r_j, d_j|C_{max}$ param. by pw is in FPT.
  - $\mathcal{O}(16^{pw} n^4)$ time [Munier Kordon, 2021].
  - $\mathcal{O}(pw2^{pw} mn^3)$ time [Shi et al., 2026].
- $P|\mathcal{M}_j(type), prec, r_j, d_j|C_{max}$ param. by $(pw+\sigma)$ is in FPT.
  - $\mathcal{O}([2\sigma(pw+1)]^{pw^2} n^4)$ time [Hanen and Munier Kordon, 2023]

Some results with slack σ and pathwidth pw

- $1|prec, r_j, d_j|C_{max}$ param. by pw is in FPT.
  - $\mathcal{O}((pw+1)^2 4^{pw} n + n\log(n))$ time [Hanen et al., 2022].
- $P|prec, p_j = 1, r_j, d_j|C_{max}$ param. by pw is in FPT.
  - $\mathcal{O}(16^{pw} n^4)$ time [Munier Kordon, 2021].
  - $\mathcal{O}(pw 2^{pw} m n^3)$ time [Shi et al., 2026].
- $P|\mathcal{M}_j(type), prec, r_j, d_j|C_{max}$ param. by $(pw+\sigma)$ is in FPT.
  - $\mathcal{O}([2\sigma(pw+1)]^{pw^2} n^4)$ time [Hanen and Munier Kordon, 2023]

Question

- What about problems with precedence delays?

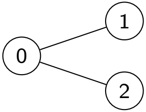
$1|chains(\ell^{ex}), p_j = 1, r_j, d_j|C_{max}$ with $pw \leq 2$ is NP-complete.

- Reduction from 3-COLORING: $G = (V, E)$.

$1|chains(\ell^{ex}), p_j = 1, r_j, d_j|C_{max}$ with $pw \leq 2$ is NP-complete.

- Reduction from 3-COLORING: $G = (V, E)$.

Example:



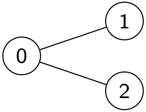
$\longrightarrow$

Scheduling instance with one machine, $3|V| + 2|E|$ jobs and exact delays.

$1|chains(\ell^{ex}), p_j = 1, r_j, d_j|C_{max}$ with $pw \leq 2$ is NP-complete.

- Reduction from 3-COLORING: $G = (V, E)$.

Example:

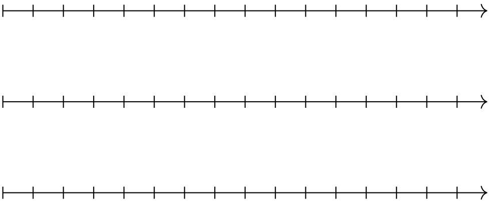


$\longrightarrow$ Scheduling instance with one machine, $3|V| + 2|E|$ jobs and exact delays.

Vertex chain $\mathcal{C}_0$

Vertex chain $\mathcal{C}_1$

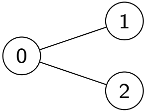
Vertex chain $\mathcal{C}_2$



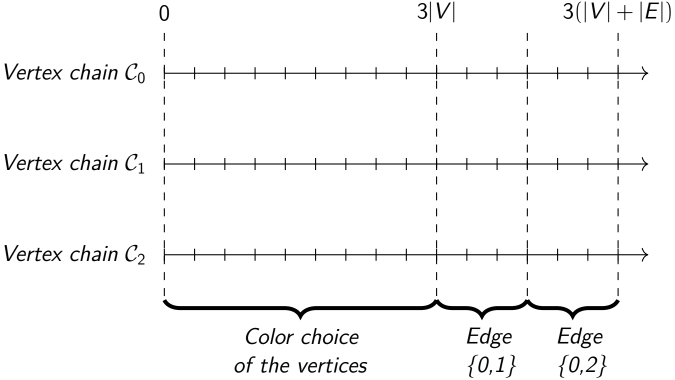
$1|chains(\ell^{ex}), p_j = 1, r_j, d_j|C_{max}$ with $pw \leq 2$ is NP-complete.

- Reduction from 3-COLORING: $G = (V, E)$.

Example:



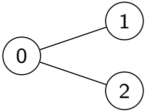
$\longrightarrow$ Scheduling instance with one machine, $3|V| + 2|E|$ jobs and exact delays.



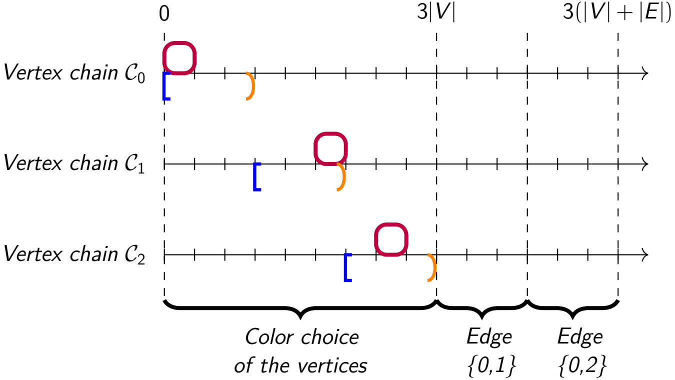
$1|chains(\ell^{ex}), p_j = 1, r_j, d_j|C_{max}$ with $pw \leq 2$ is NP-complete.

- Reduction from 3-COLORING: $G = (V, E)$.

Example:



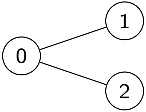
$\longrightarrow$ Scheduling instance with one machine, $3|V| + 2|E|$ jobs and exact delays.



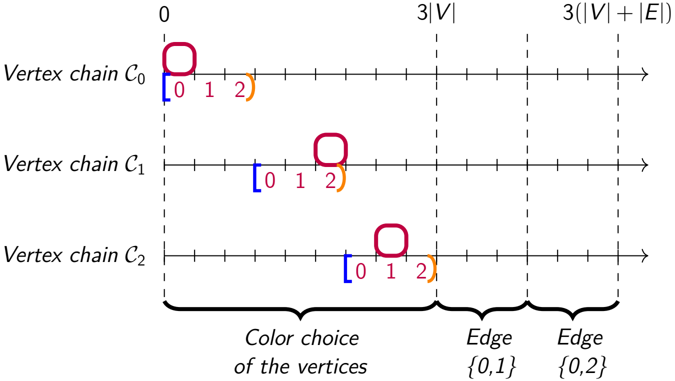
$1|chains(\ell^{ex}), p_j = 1, r_j, d_j|C_{max}$ with $pw \leq 2$ is NP-complete.

- Reduction from 3-COLORING: $G = (V, E)$.

Example:



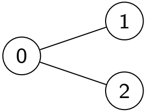
$\longrightarrow$ Scheduling instance with one machine, $3|V| + 2|E|$ jobs and exact delays.



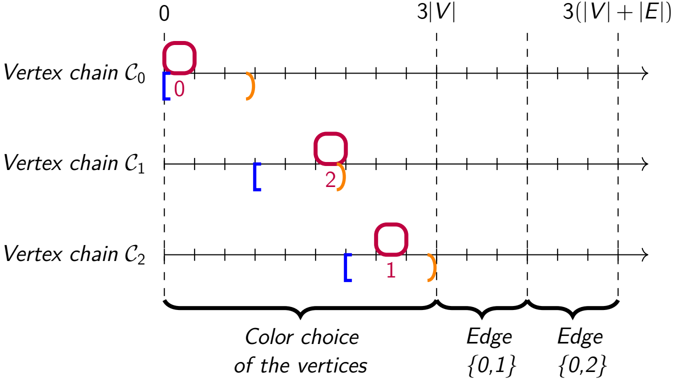
$1|chains(\ell^{ex}), p_j = 1, r_j, d_j|C_{max}$ with $pw \leq 2$ is NP-complete.

- Reduction from 3-COLORING: $G = (V, E)$.

Example:

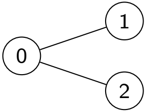


$\longrightarrow$ Scheduling instance with one machine, $3|V| + 2|E|$ jobs and exact delays.

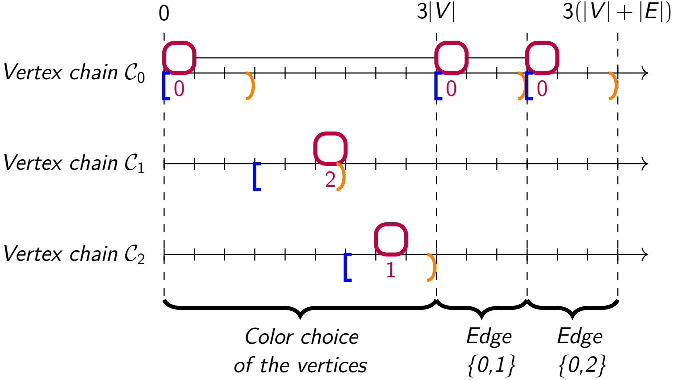


- Reduction from 3-COLORING: $G = (V, E)$.

Example:



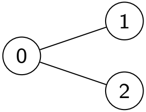
$\longrightarrow$ Scheduling instance with one machine, $3|V| + 2|E|$ jobs and exact delays.



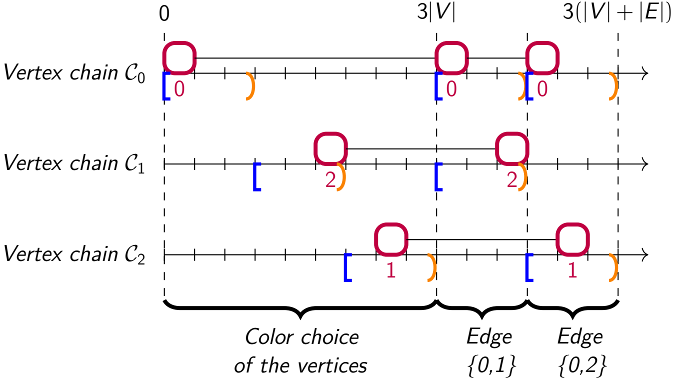
$1|chains(\ell^{ex}), p_j = 1, r_j, d_j|C_{max}$ with $pw \leq 2$ is NP-complete.

- Reduction from 3-COLORING: $G = (V, E)$.

Example:



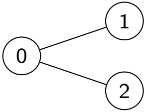
$\longrightarrow$ Scheduling instance with one machine, $3|V| + 2|E|$ jobs and exact delays.



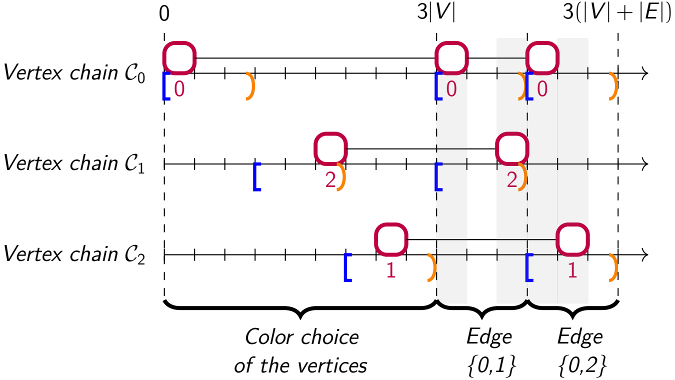
$1|chains(\ell^{ex}), p_j = 1, r_j, d_j|C_{max}$ with $pw \le 2$ is NP-complete.

- Reduction from 3-COLORING: $G = (V, E)$.

Example:



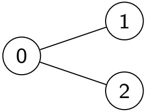
$\longrightarrow$ Scheduling instance with one machine, $3|V| + 2|E|$ jobs and exact delays.



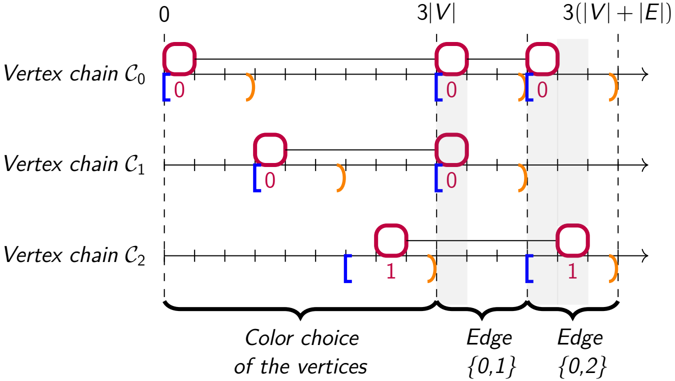
$1|chains(\ell^{ex}), p_j = 1, r_j, d_j|C_{max}$ with $pw \leq 2$ is NP-complete.

- Reduction from 3-COLORING: $G = (V, E)$.

Example:



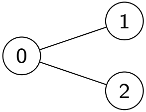
$\longrightarrow$ Scheduling instance with one machine, $3|V| + 2|E|$ jobs and exact delays.



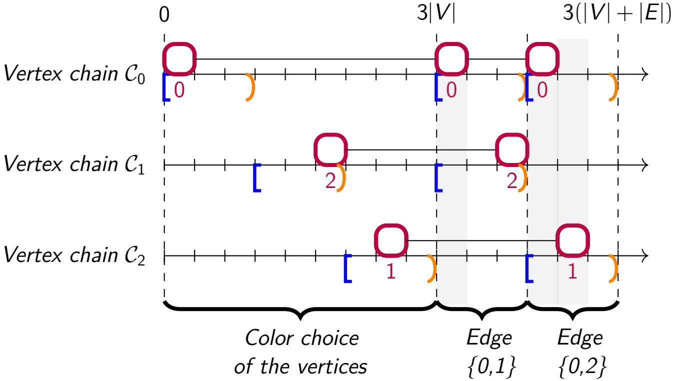
$1|chains(\ell^{ex}), p_j = 1, r_j, d_j|C_{max}$ with $pw \leq 2$ is NP-complete.

- Reduction from 3-COLORING: $G = (V, E)$.

Example:



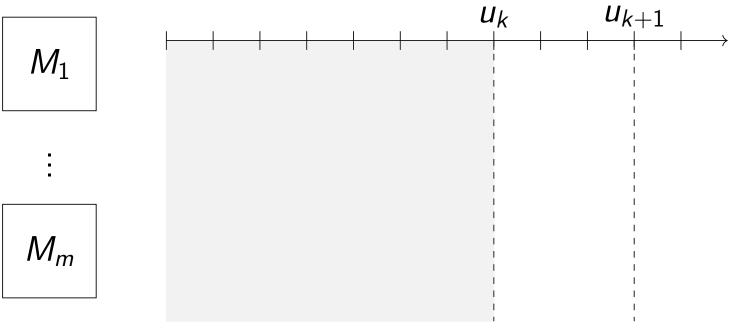
$\longrightarrow$ Scheduling instance with one machine, $3|V| + 2|E|$ jobs and exact delays.



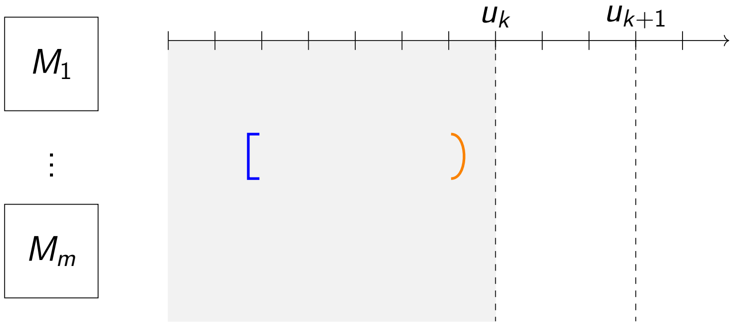
Outline

1. Introduction
2. Precedence delays
3. Adding time windows
4. **Pairing parameters**
5. Conclusion

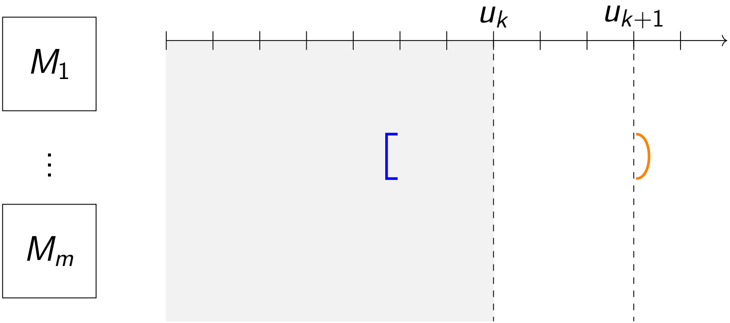
$P|prec(\ell^{min}), p_j = 1, r_j, d_j|C_{max}$ is in FPT w.r.t. $(pw + \ell_{max})$.



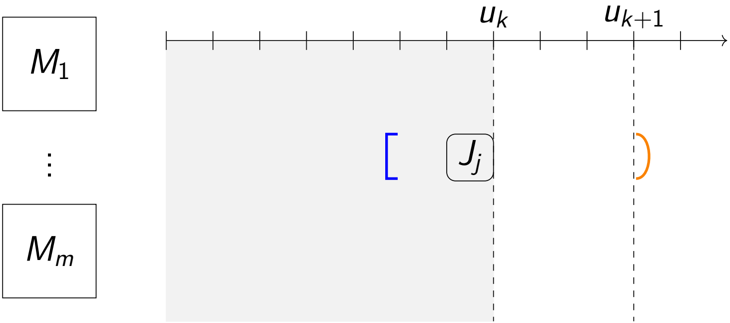
$P|prec(\ell^{min}), p_j = 1, r_j, d_j|C_{max}$ is in FPT w.r.t. $(pw + \ell_{max})$.



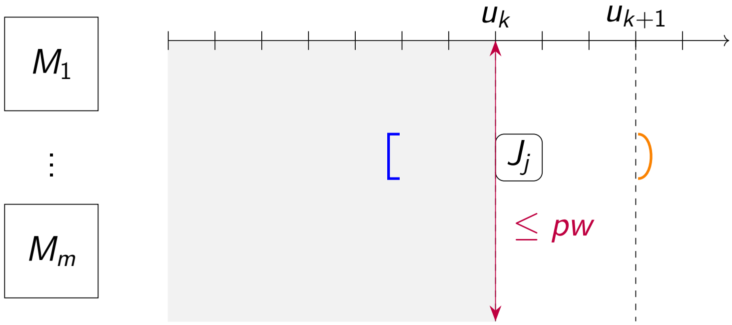
$P|prec(\ell^{min}), p_j = 1, r_j, d_j|C_{max}$ is in FPT w.r.t. $(pw + \ell_{max})$.



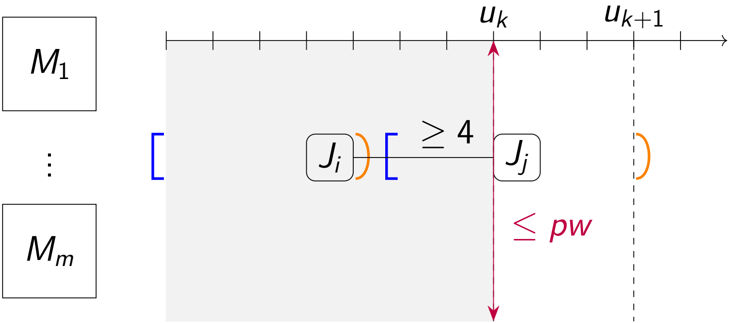
$P|prec(\ell^{min}), p_j = 1, r_j, d_j|C_{max}$ is in FPT w.r.t. $(pw + \ell_{max})$.



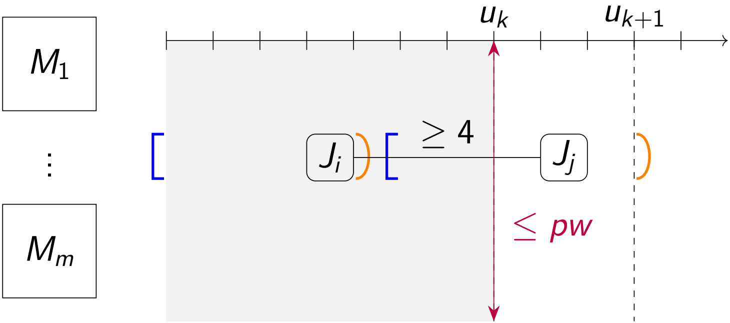
$P|prec(\ell^{min}), p_j = 1, r_j, d_j|C_{max}$ is in FPT w.r.t. $(pw + \ell_{max})$.



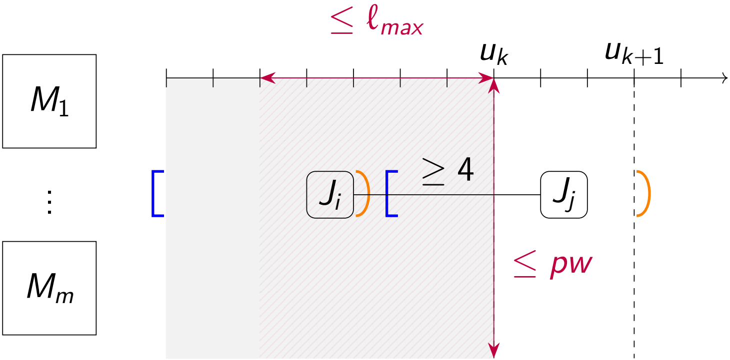
$P|prec(\ell^{min}), p_j = 1, r_j, d_j|C_{max}$ is in FPT w.r.t. $(pw + \ell_{max})$.



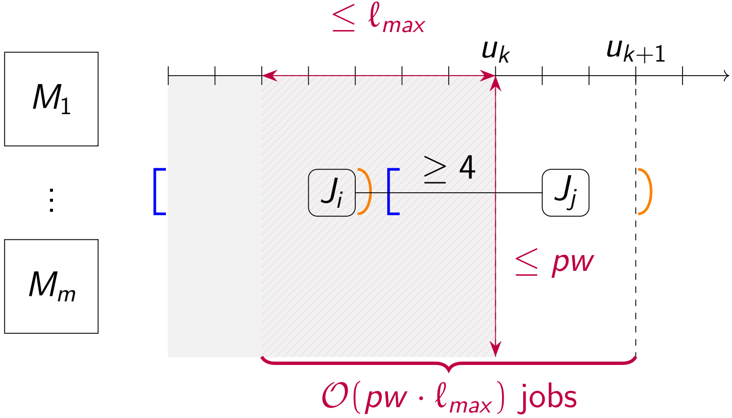
$P|prec(\ell^{min}), p_j = 1, r_j, d_j|C_{max}$ is in FPT w.r.t. $(pw + \ell_{max})$.



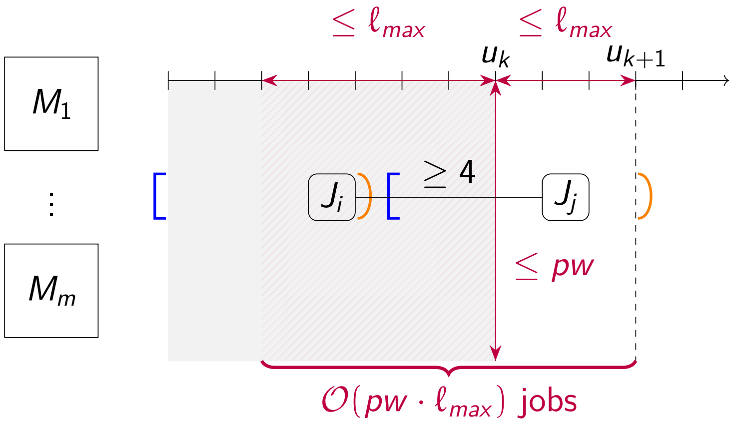
$P|prec(\ell^{min}), p_j = 1, r_j, d_j|C_{max}$ is in FPT w.r.t. $(pw + \ell_{max})$.



$P|prec(\ell^{min}), p_j = 1, r_j, d_j|C_{max}$ is in FPT w.r.t. $(pw + \ell_{max})$.



$P|prec(\ell^{min}), p_j = 1, r_j, d_j|C_{max}$ is in FPT w.r.t. $(pw + \ell_{max})$.



Outline

Conclusion

Parameter	$1\|prec(\ell^{min}), r_j, d_j\|C_{max}$	$P\|prec(\ell^{min}), p_j = 1, r_j, d_j\|C_{max}$
ℓ_{max}	XNLP-hard	
$pw + \sigma$	para-NP-complete	
$\sigma + \ell_{max}$	in FPT	para-NP-complete
$pw + \ell_{max}$	open	in FPT

Conclusion

Parameter	$1\|prec(\ell^{min}), r_j, d_j\|C_{max}$	$P\|prec(\ell^{min}), p_j = 1, r_j, d_j\|C_{max}$
ℓ_{max}	XNLP-hard	
$pw + \sigma$	para-NP-complete	
$\sigma + \ell_{max}$	in FPT	para-NP-complete
$pw + \ell_{max}$	open	in FPT

Future work

- Improve the existing algorithms.
    - Dependency on the parameter (fine-grained complexity).
    - Kernelization.

Conclusion

Parameter	$1 \vert prec(\ell^{min}), r_j, d_j \vert C_{max}$	$P \vert prec(\ell^{min}), p_j = 1, r_j, d_j \vert C_{max}$
ℓ_{max}	XNLP-hard	
$pw + \sigma$	para-NP-complete	
$\sigma + \ell_{max}$	in FPT	para-NP-complete
$pw + \ell_{max}$	open	in FPT

Future work

- Improve the existing algorithms.
  - Dependency on the parameter (fine-grained complexity).
  - Kernelization.
- Investigate other parameters/scheduling settings.
  - Postdoc: periodic time windows.

Postdoc: daily time windows

LIP, ENS de Lyon Nov. 2024 - Apr. 2026

- *Topic:* Scheduling radio-astronomical observations.

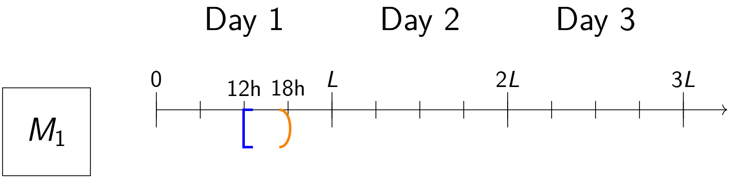
Postdoc: daily time windows

LIP, ENS de Lyon Nov. 2024 - Apr. 2026

- *Topic:* Scheduling radio-astronomical observations.
  - Partnership with the ECLAT laboratory (HPC & AI for astronomy & astrophysics).
  - Access to NenuFAR radio-telescope usage data.

LIP, ENS de Lyon Nov. 2024 - Apr. 2026

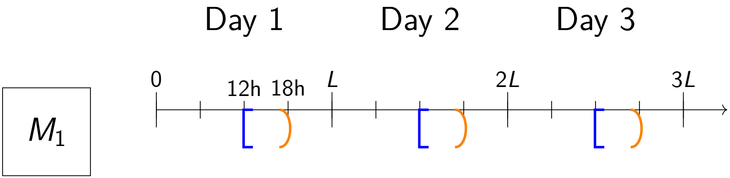
- *Topic:* Scheduling radio-astronomical observations.
  - Partnership with the ECLAT laboratory (HPC & AI for astronomy & astrophysics).
  - Access to NenuFAR radio-telescope usage data.



Postdoc: daily time windows

LIP, ENS de Lyon Nov. 2024 - Apr. 2026

- *Topic:* Scheduling radio-astronomical observations.
  - Partnership with the ECLAT laboratory (HPC & AI for astronomy & astrophysics).
  - Access to NenuFAR radio-telescope usage data.



References I

Bodlaender, H. L. (2021).
Parameterized complexity of bandwidth of caterpillars and weighted path emulation.
In *Graph-Theoretic Concepts in Computer Science: 47th International Workshop, WG 2021, Warsaw, Poland, June 23–25, 2021, Revised Selected Papers 47*, pages 15–27. Springer.

Bodlaender, H. L., Groenland, C., Nederlof, J., and Swennenhuis, C. M. (2022).
Parameterized problems complete for nondeterministic FPT time and logarithmic space.
In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 193–204. IEEE.

Bodlaender, H. L. and Möhring, R. H. (1993).
The pathwidth and treewidth of cographs.
SIAM Journal on Discrete Mathematics, 6(2):181–188.

Hanen, C., Mallem, M., and Munier-Kordon, A. (2022).
Parameterized complexity of single-machine scheduling with precedence, release dates and deadlines.
In *15th Workshop on Models and Algorithms for Planning and Scheduling Problems (MAPSP)*.

References II

Hanen, C. and Munier Kordon, A. (2023).
Fixed-parameter tractability of scheduling dependent typed tasks subject to release times and deadlines.
Journal of Scheduling, pages 1–15.

Jansen, K., Kratsch, S., Marx, D., and Schlotter, I. (2013).
Bin packing with fixed number of bins revisited.
Journal of Computer and System Sciences, 79(1):39–49.

Lenstra, J., Rinnooy Kan, A., and Brucker, P. (1977).
Complexity of machine scheduling problems.
Ann. of Discrete Math., 1:343–362.

Munier Kordon, A. (2021).
A fixed-parameter algorithm for scheduling unit dependent tasks on parallel machines with time windows.
Discrete Applied Mathematics, 290:1–6.

References III

Shi, F., Zhu, Y., Wu, G., Liu, J., and Wang, J. (2026).
Improved parameterized algorithms for scheduling with precedence constraints and time windows.
In Fomin, F. V. and Xiao, M., editors, *Computing and Combinatorics (COCOON 2025)*, pages 253–267, Singapore. Springer Nature Singapore.

Ullman, J. D. (1975).
NP-complete scheduling problems.
Journal of Computer and System sciences.

van Bevern, R., Bredereck, R., Bulteau, L., Komusiewicz, C., Talmon, N., and Woeginger, G. J. (2016).
Precedence-constrained scheduling problems parameterized by partial order width.
In Kochetov, Y., Khachay, M., Beresnev, V., Nurminski, E., and Pardalos, P., editors, *Discrete Optimization and Operations Research*, pages 105–120, Cham. Springer International Publishing.