Resource leveling for scheduling problems: some complexity and approximation results

Luca Brunod Indrigo [1, 2] Pascale Bendotti [1, 2]
Philippe Chrétienne [2] Bruno Escoffier [2]

[1]EDF R&D

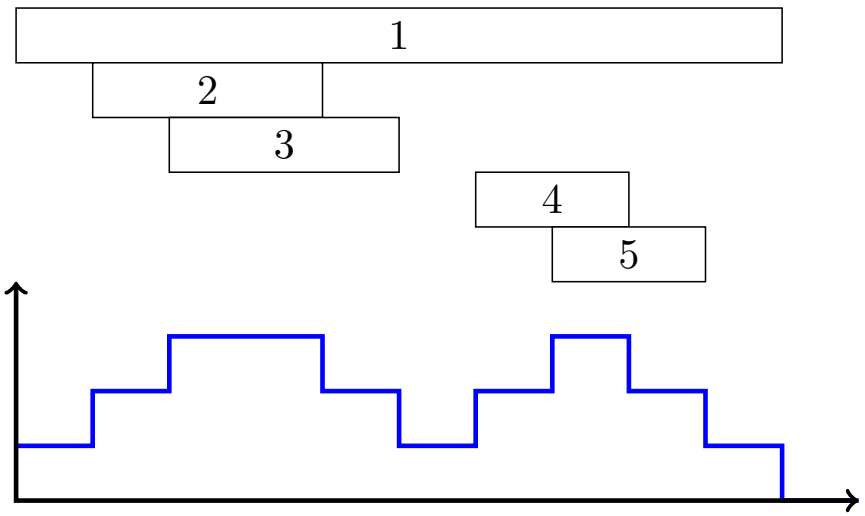
[2]Sorbonne Université, CNRS, LIP6 UMR 7606

15/04/2026

Scheduling Seminar, schedulingseminar.com

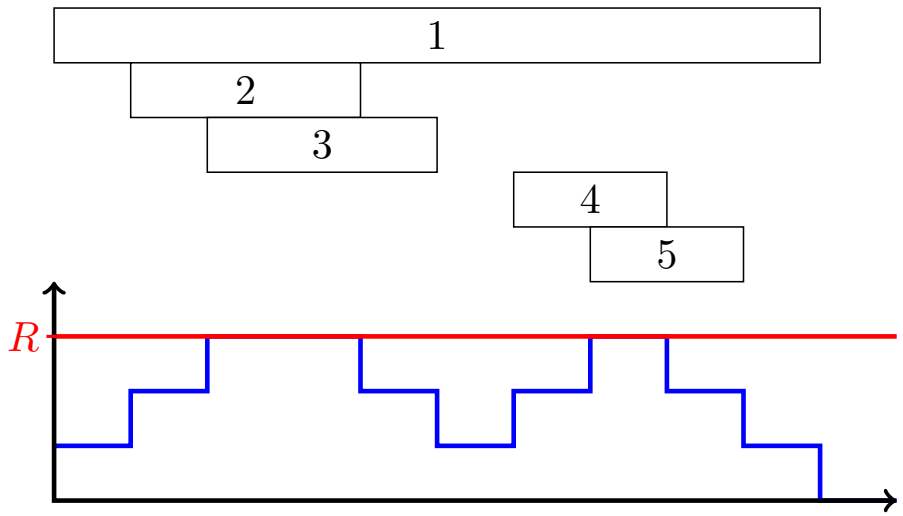
Resource leveling

- Many scheduling problems involve **resources**
- Resources represent workers, equipments, processors...



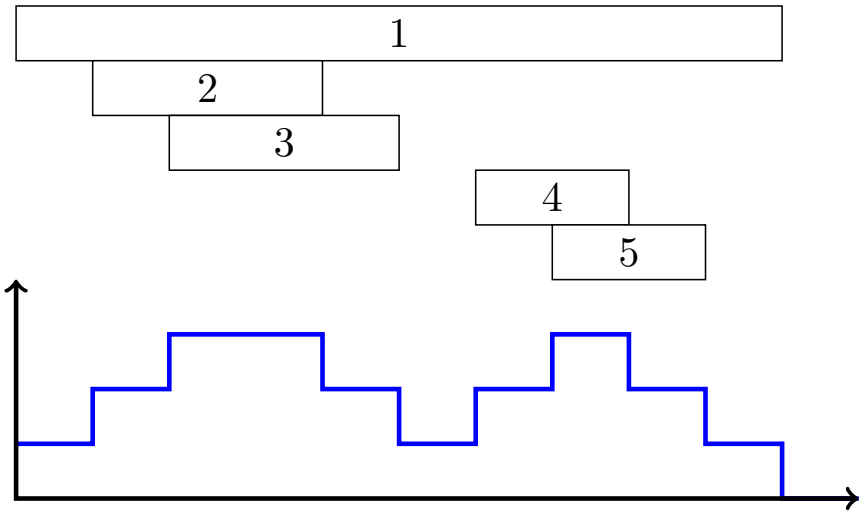
Resource leveling

- Many scheduling problems involve **resources**
- Resources represent workers, equipments, processors...
- Resource capacity usually imposed as **hard constraint**
- ex: machine scheduling, RCPSP



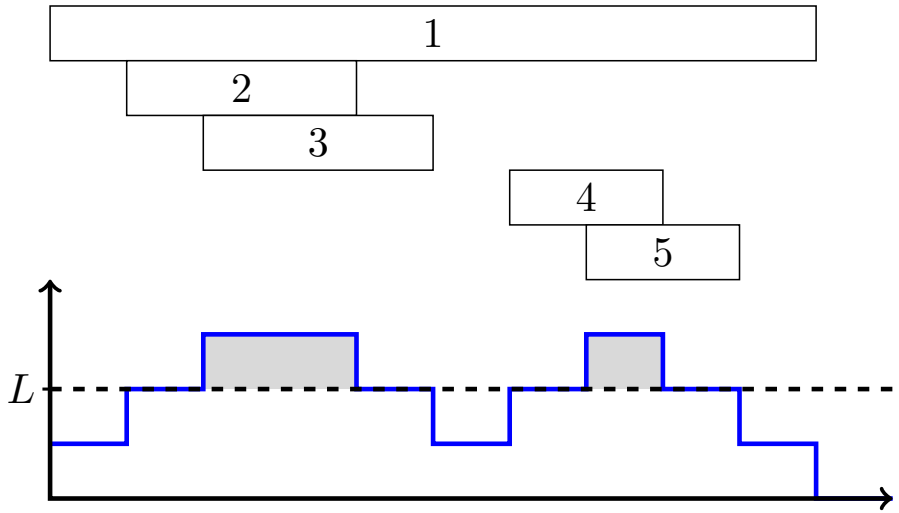
Resource leveling

- In resource leveling, **no hard constraints** on resources
- Objective function penalizes irregularities in resource use



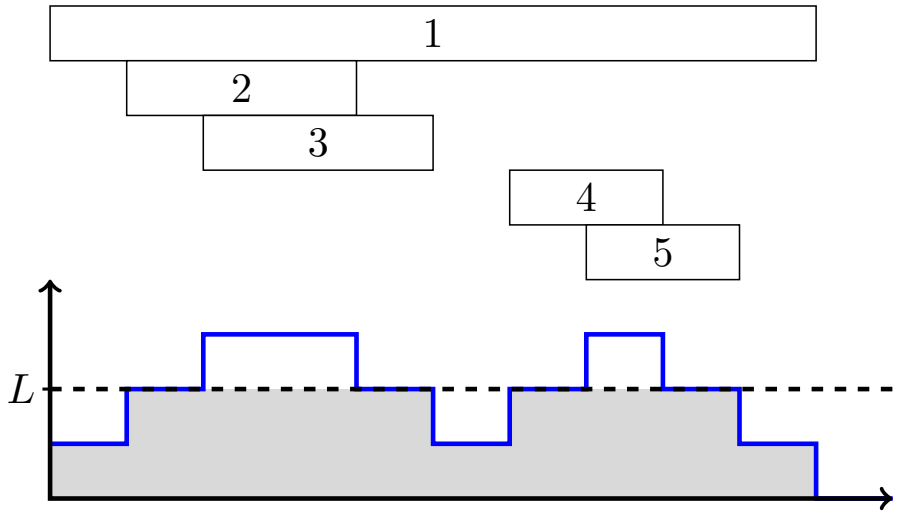
Resource leveling

- In resource leveling, **no hard constraints** on resources
- Objective function penalizes irregularities in resource use
- Total overload cost: **minimizing** the resource use **exceeding a level** L



Resource leveling

- In resource leveling, **no hard constraints** on resources
- Objective function penalizes irregularities in resource use
- Total overload cost: **minimizing** the resource use **exceeding a level** L
- Equivalent to **maximizing** the resource use that **fits under level** L



Resource leveling

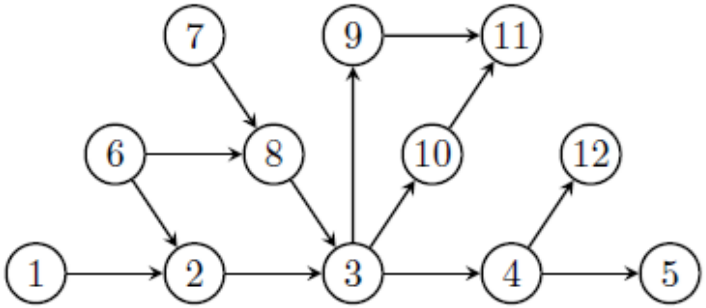
Problems under consideration

- A set of jobs J, with processing times
- A (target) resource level L
- Objective: try not to exceed this level
  $\rightarrow$ ideally no more than L jobs in parallel
  $\rightarrow$ Max the load that fits under this level L: function F
- Under a makespan deadline M
- and various constraints or cases, such as:
  - Precedence constraints
  - Unit processing times
  - ...

Resource leveling

Example

- 12 jobs with processing times $p_i = 1$, with precedence constraints
- Resource level $L = 2$
- Makespan deadline M.

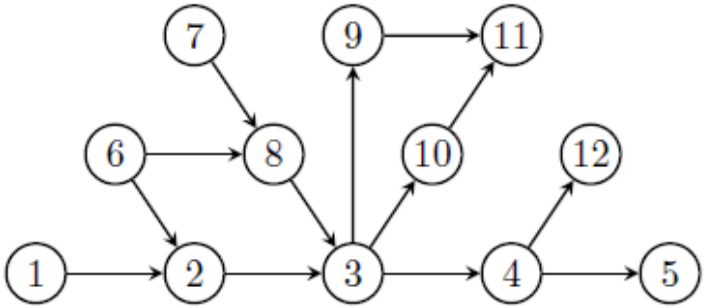


Precedence graph G

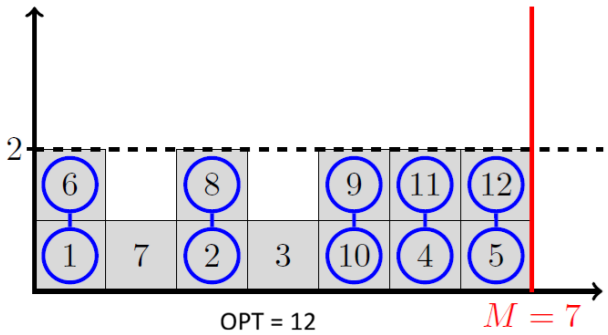
Resource leveling

Example

- 12 jobs with processing times $p_i = 1$, with precedence constraints
- Resource level $L = 2$
- Makespan deadline M.



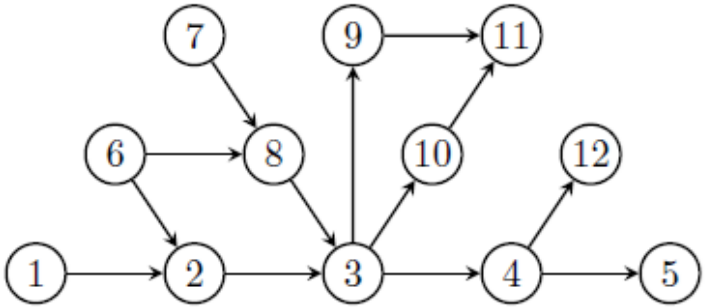
Precedence graph G



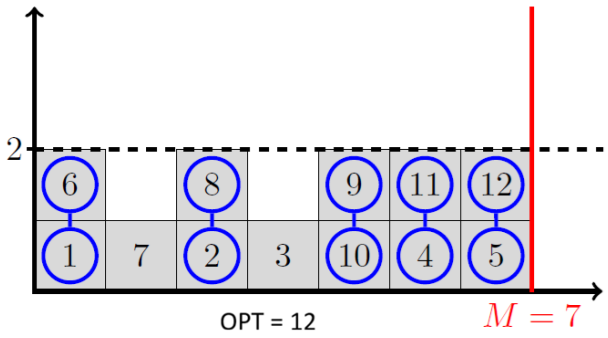
Resource leveling

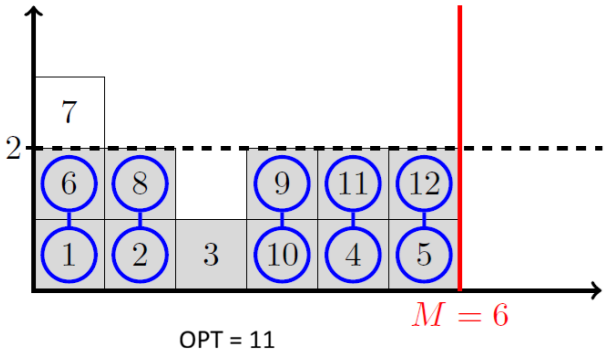
Example

- 12 jobs with processing times $p_i = 1$, with precedence constraints
- Resource level $L = 2$
- Makespan deadline M.



Precedence graph G

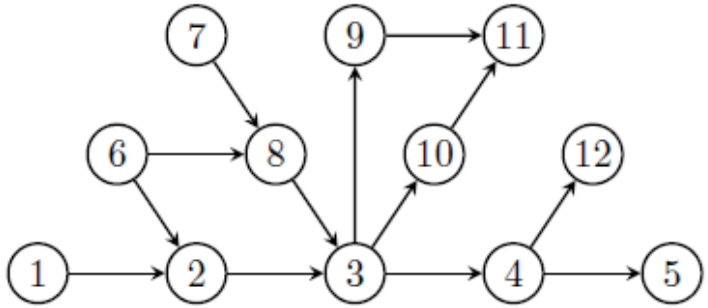




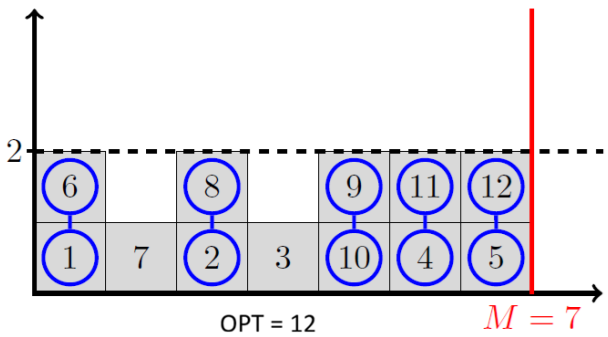
Resource leveling

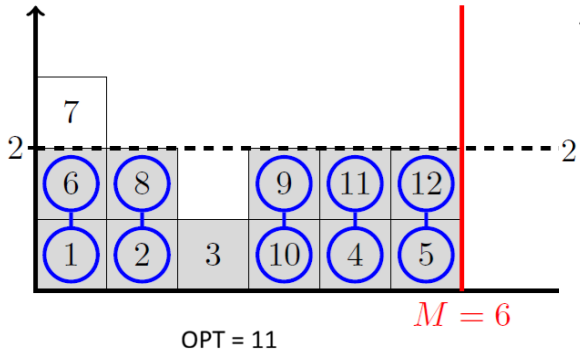
Example

- 12 jobs with processing times $p_i = 1$, with precedence constraints
- Resource level $L = 2$
- Makespan deadline M.



Precedence graph G





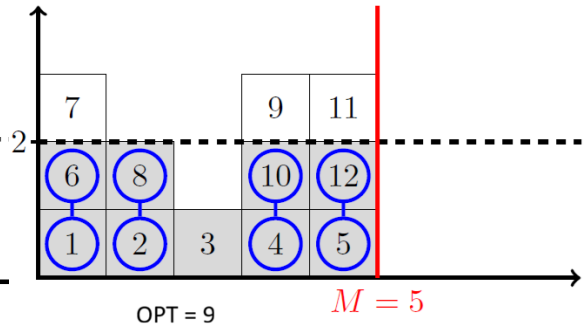


Table of complexity results

		$C_{max} \leq M$	$prec, C_{max} \leq M$		r_i, d_i	
			any	*in-tree*	no *pmtn*	*pmtn*
$L = 1$	$\emptyset$	in P	in P		strongly NP-hard	in P
$L = 2$	$\emptyset$	pseudo polynomial	strongly NP-hard		strongly NP-hard	in P
	$p_i = 1$	in P	in P		in P	
$L \in \mathbb{N}$	$\emptyset$	strongly NP-hard				in P
	$p_i = 1$	in P	strongly NP-hard	in P	in P	

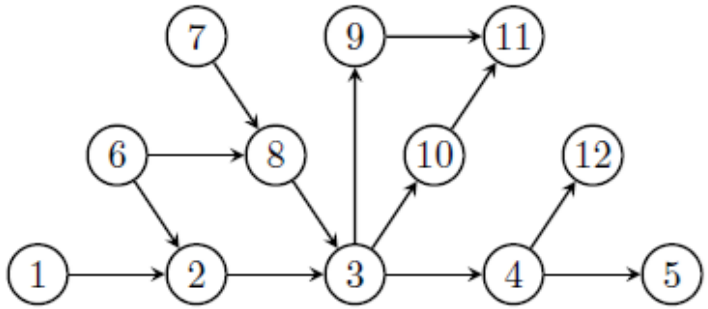
Complexity of resource leveling problems [1]

[1] Bendotti, Brunod Indrigo, Chrétienne, Escoffier, Journal of Scheduling 2024.

Resource leveling

$L2|prec, p_i = 1, C_{max} \leq M|F$

- n jobs with processing times $p_i = 1$, with precedence constraints
- Resource level $L = 2$
- Makespan deadline M.
- Maximize F.

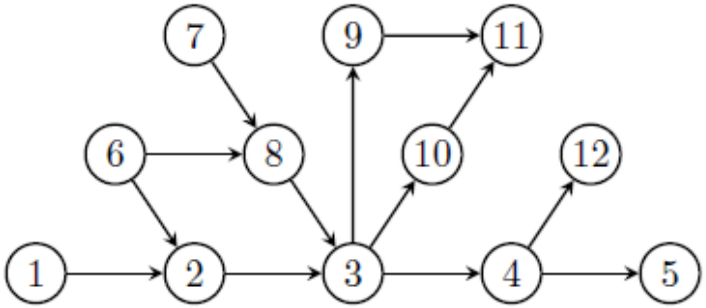


Precedence graph G

Resource leveling

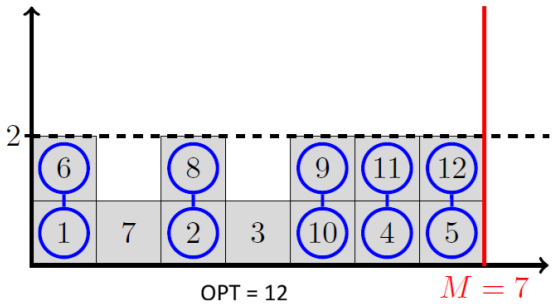
$L2|prec, p_i = 1, C_{max} \leq M|F$

- n jobs with processing times $p_i = 1$, with precedence constraints
- Resource level $L = 2$
- Makespan deadline M.
- Maximize F.



Precedence graph G

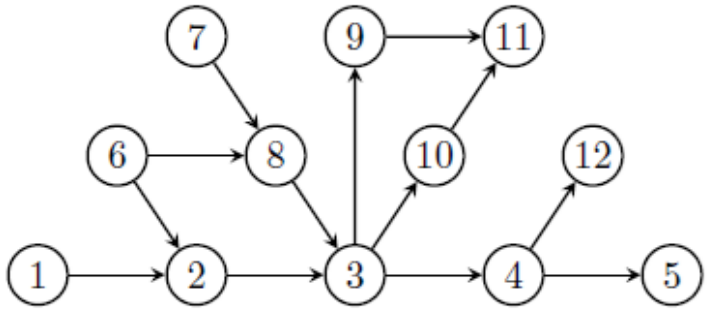
- Q1: can I know if everything can fit or not?
  $\rightarrow OPT = \sum p_i = n$?



Resource leveling

$L2|prec, p_i = 1, C_{max} \leq M|F$

- n jobs with processing times $p_i = 1$, with precedence constraints
- Resource level $L = 2$
- Makespan deadline M.
- Maximize F.

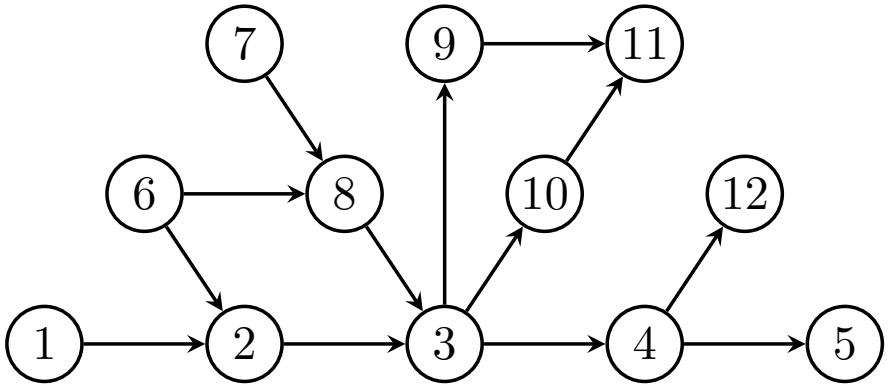


Precedence graph G

- Q1: can I know if everything can fit or not?
  $\rightarrow OPT = \sum p_i = n$?

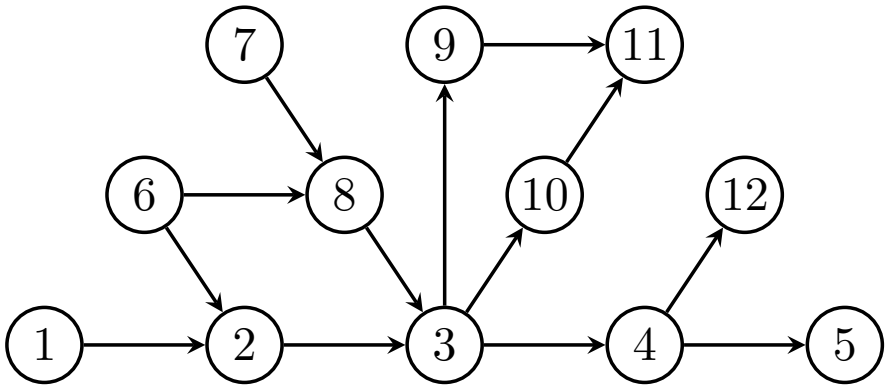
Equivalent to the well-known $P_2|prec, p_i = 1|C_{max}$ problem . . .
. . . solvable in polynomial time (Fujii, Kasami, and Ninomiya (1969)).

Solving Q1, or $P2|prec, p_i = 1|C_{max}$

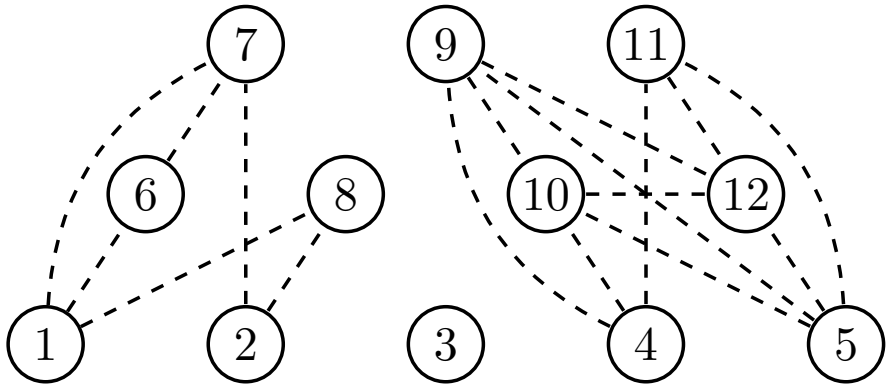


Precedence graph G

Solving Q1, or $P2|prec, p_i = 1|C_{max}$

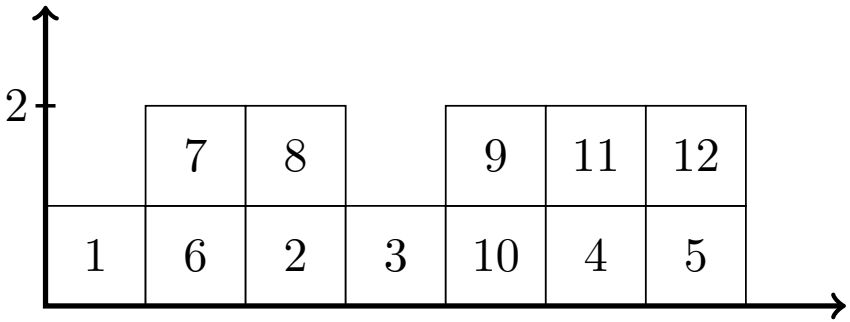


Precedence graph G

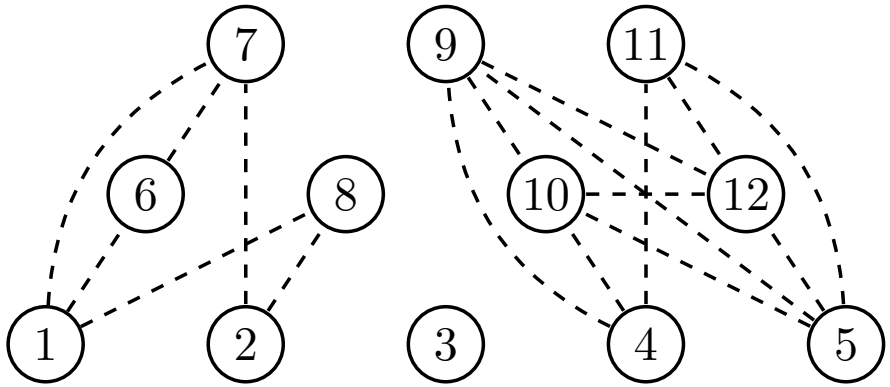


Independence graph $\widetilde{G}$

Solving Q1, or $P2|prec, p_i = 1|C_{max}$

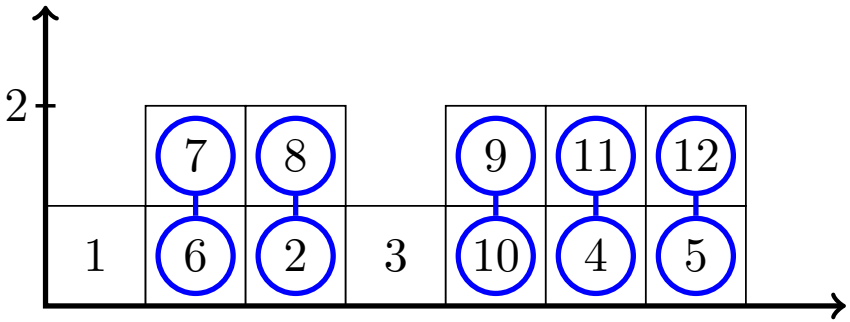


Feasible schedule

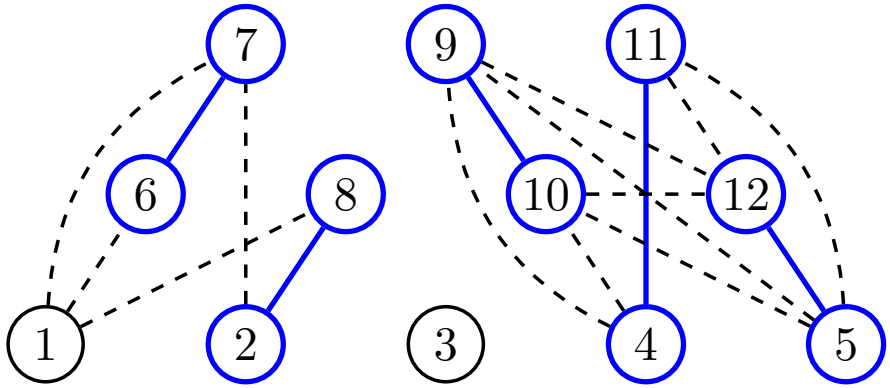


Independence graph $\widetilde{G}$

Solving Q1, or $P2|prec, p_i = 1|C_{max}$

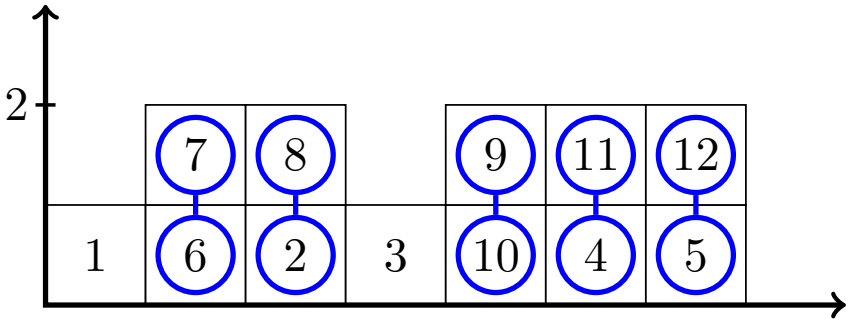


Feasible schedule

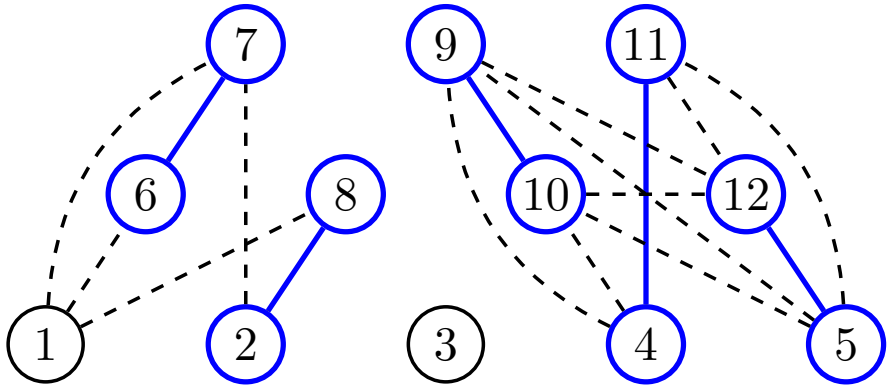


Independence graph $\widetilde{G}$

Solving Q1, or $P2|prec, p_i = 1|C_{max}$



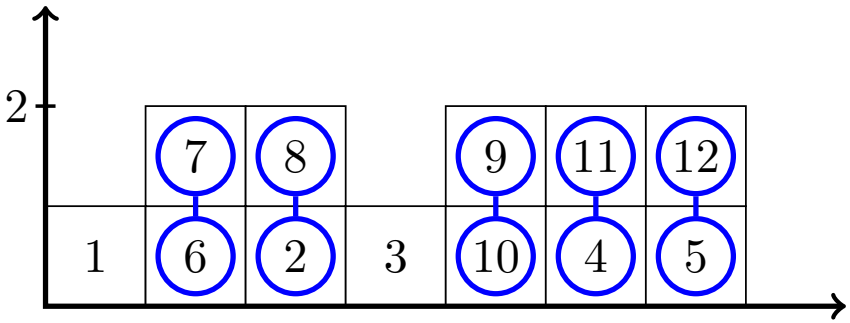
Feasible schedule



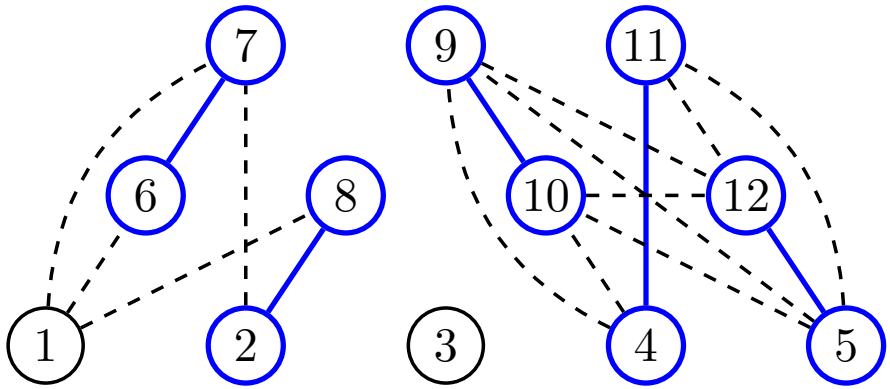
Independence graph $\widetilde{G}$

Schedule of makespan $|J| - m \Rightarrow$ matching of size m in $\widetilde{G}$

Solving Q1, or $P2|prec, p_i = 1|C_{max}$



Feasible schedule

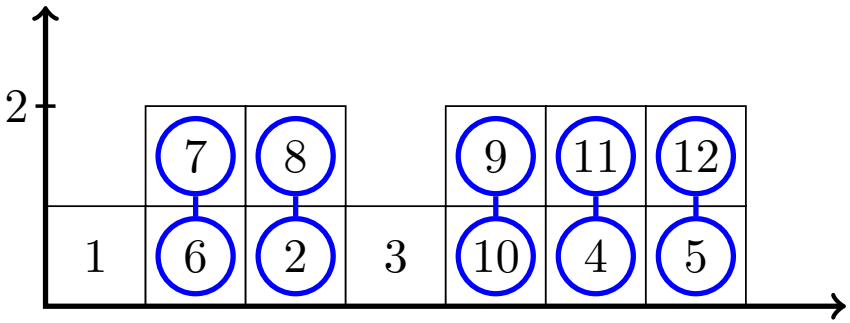


Independence graph $\widetilde{G}$

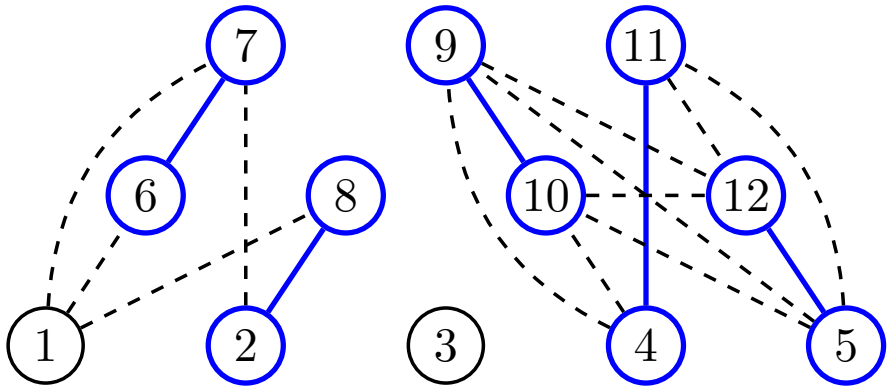
Theorem (Fujii1969)

Schedule of makespan $|J| - m \Leftrightarrow$ *matching of size* m *in* $\widetilde{G}$

Solving Q1, or $P2|prec, p_i = 1|C_{max}$



Feasible schedule



Independence graph $\widetilde{G}$

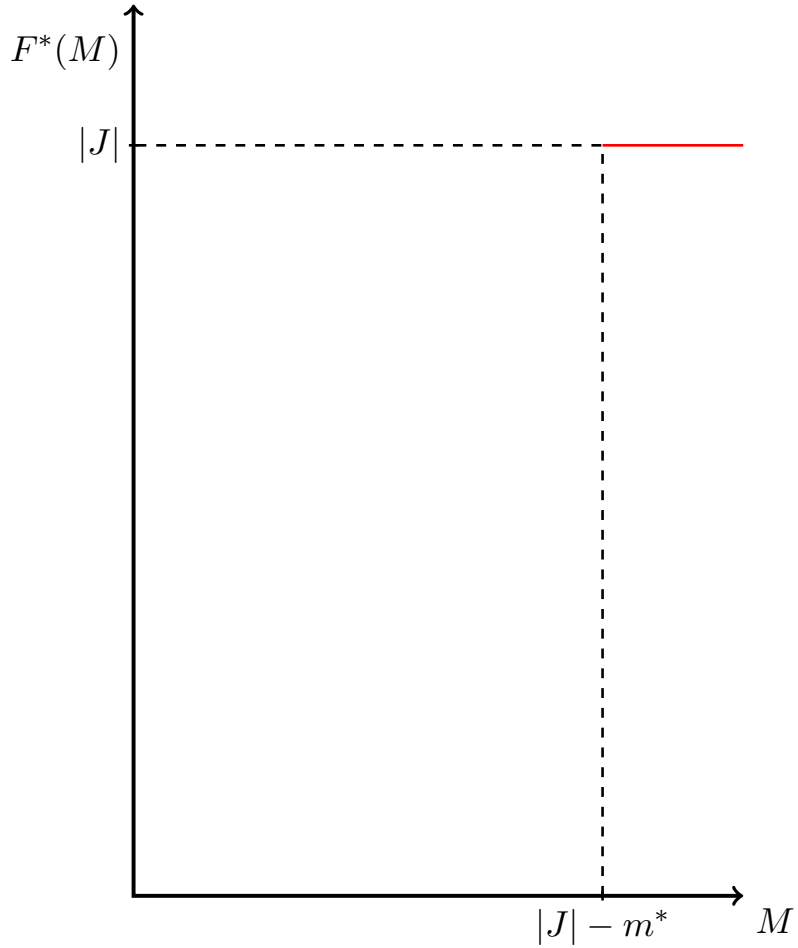
Theorem (Fujii1969)

Schedule of makespan $|J| - m \Leftrightarrow$ *matching of size* m *in* $\widetilde{G}$

m^* size of a max matching in $\widetilde{G} \Leftrightarrow$ minimum makespan is $|J| - m^*$

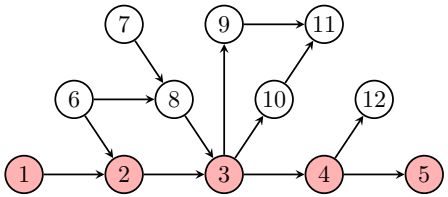
Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

- Q1: when does $OPT = |J|$?
  $\rightarrow$ When $M \geq |J| - m^*$
  using [Fujii et al, 1969]

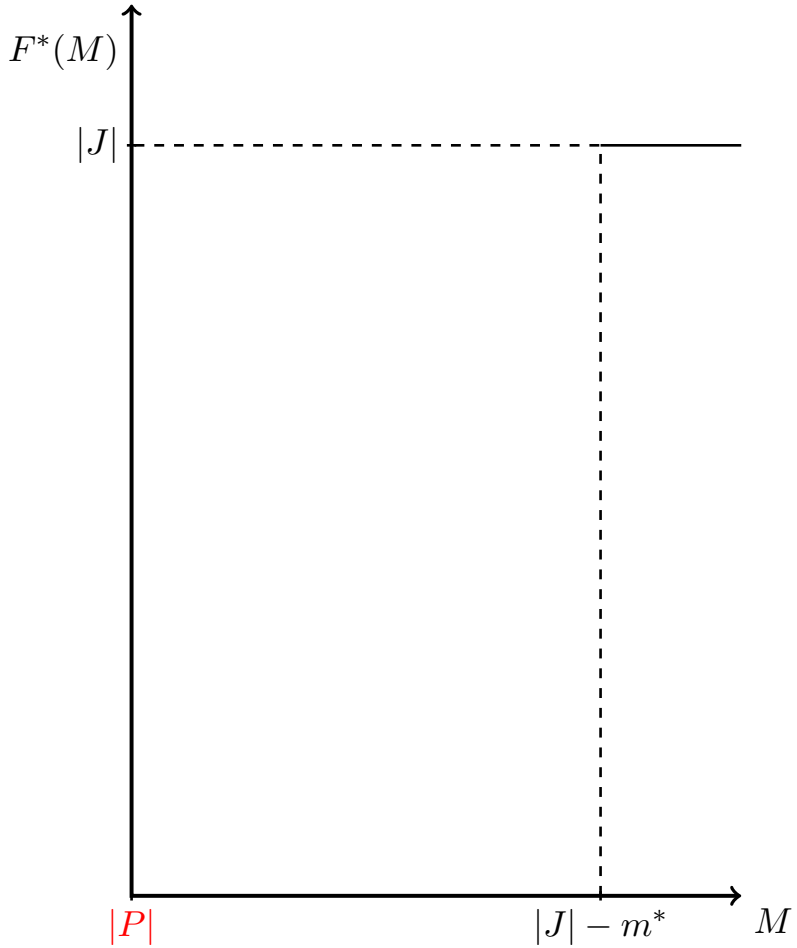


Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

- Q1: when does $OPT = |J|$?
  $\rightarrow$ When $M \geq |J| - m^*$
  using [Fujii et al, 1969]
- Q2
  P **critical path** in G
  No feasible schedules for
  $M < |P|$

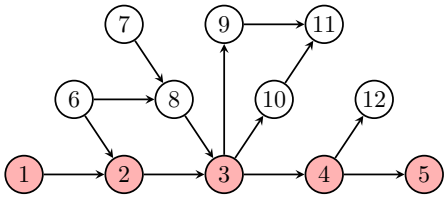


Critical path in precedence graph G



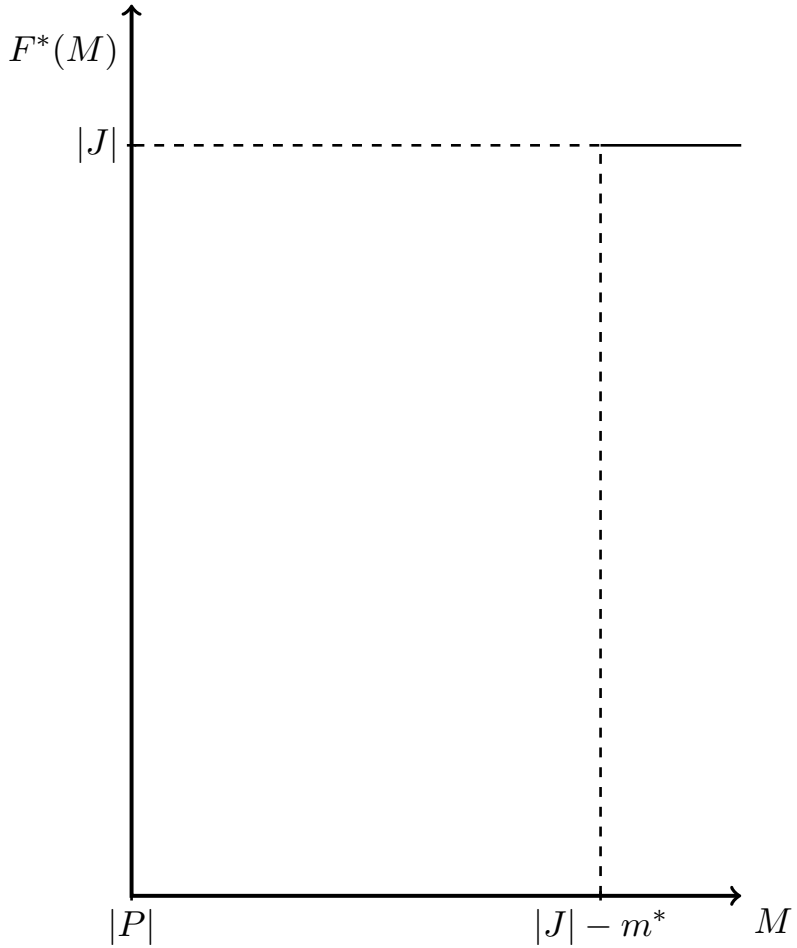
Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

- Q1: when does $OPT = |J|$?
  $\rightarrow$ When $M \geq |J| - m^*$
  using [Fujii et al, 1969]
- Q2
  P **critical path** in G
  No feasible schedules for
  $M < |P|$



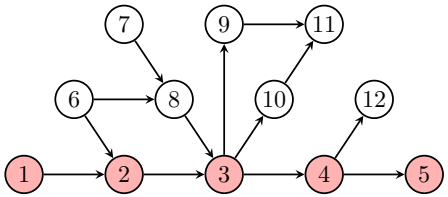
Critical path in precedence graph G

$\rightarrow$ Q2: can we solve the problem for $M = |P|$?



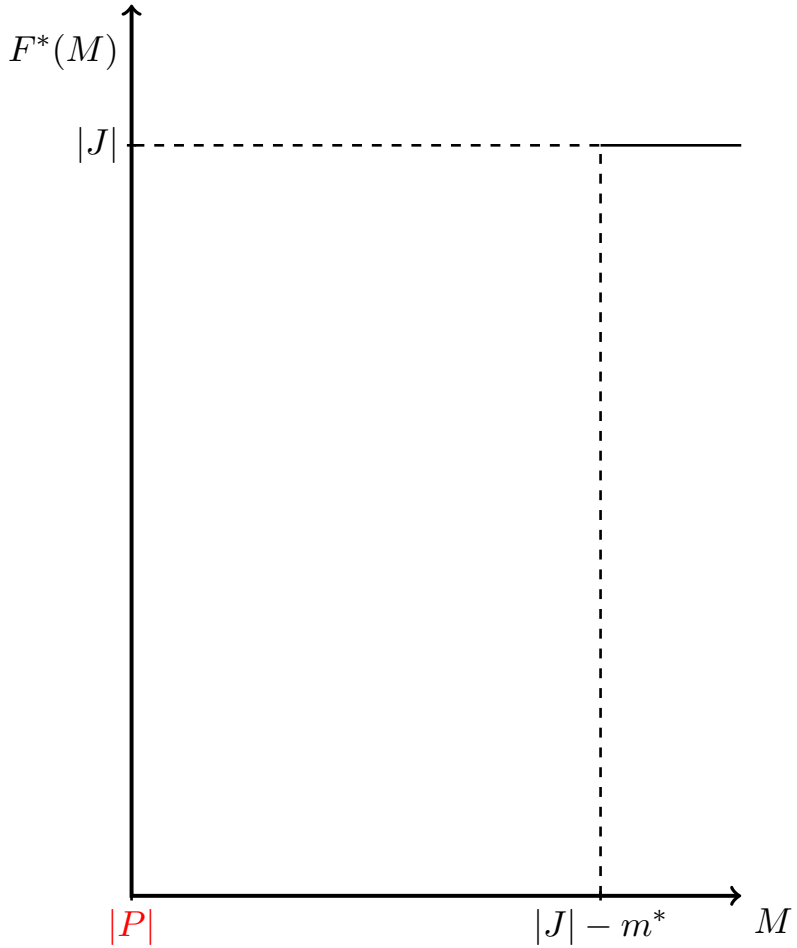
Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

- Q1: when does $OPT = |J|$?
  $\rightarrow$ When $M \geq |J| - m^*$
  using [Fujii et al, 1969]
- Q2
  P **critical path** in G
  No feasible schedules for
  $M < |P|$

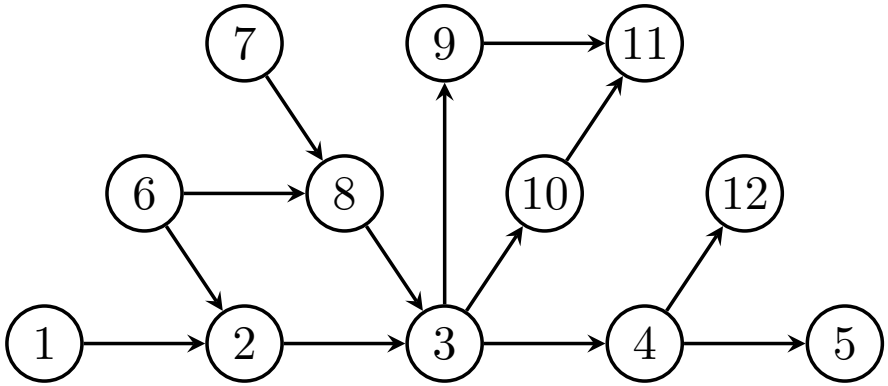


Critical path in precedence graph G

$\rightarrow$ Q2: can we solve the problem for $M = |P|$?

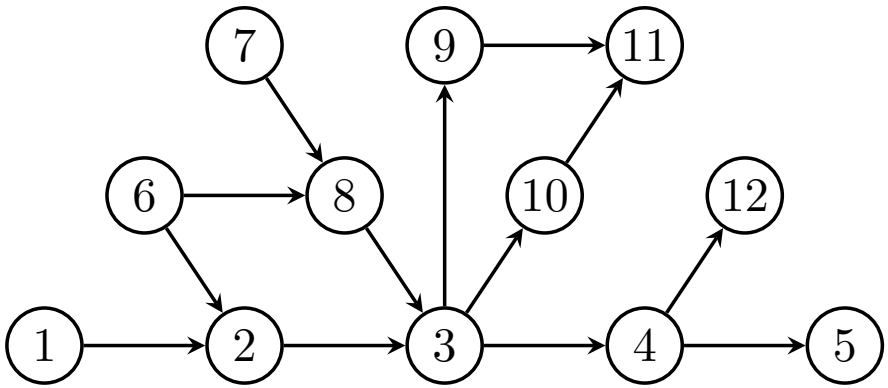


Solving Q2: when $M =$ size of a critical path

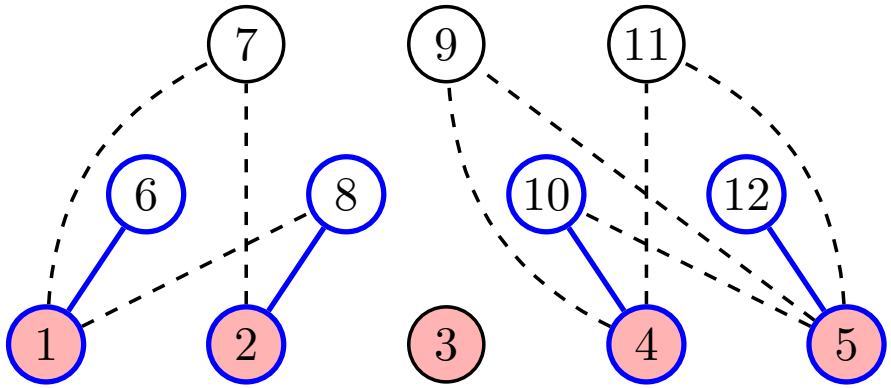


Precedence graph G

Solving Q2: when M = size of a critical path

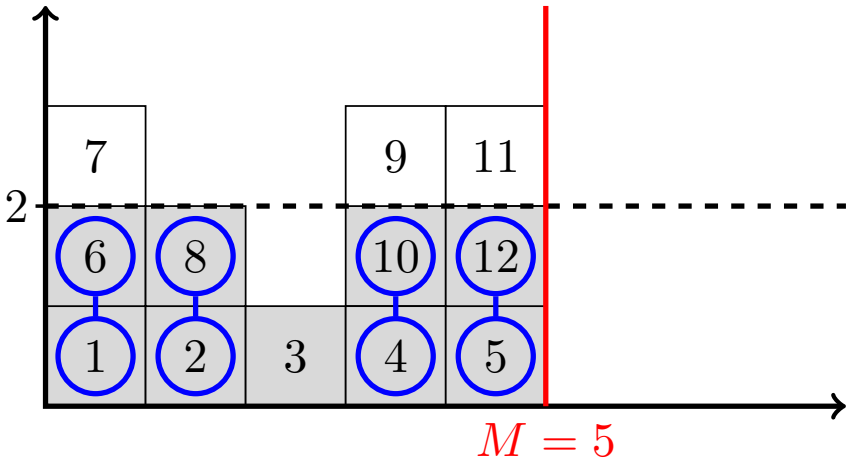


Precedence graph G

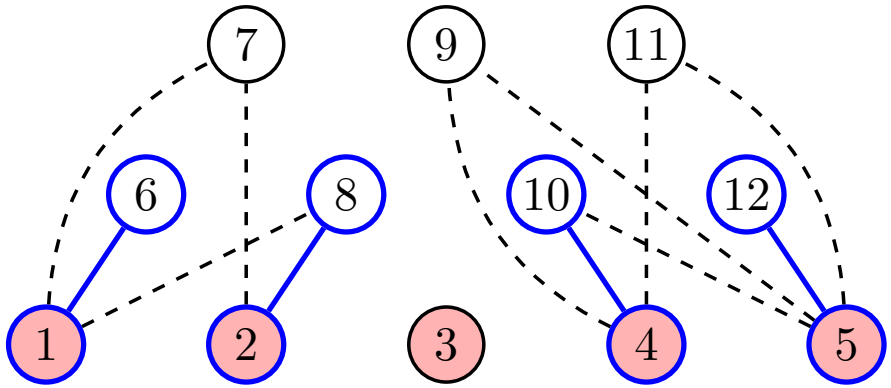


Independence bipartite graph $\widetilde{G}_P$

Solving Q2: when $M =$ size of a critical path

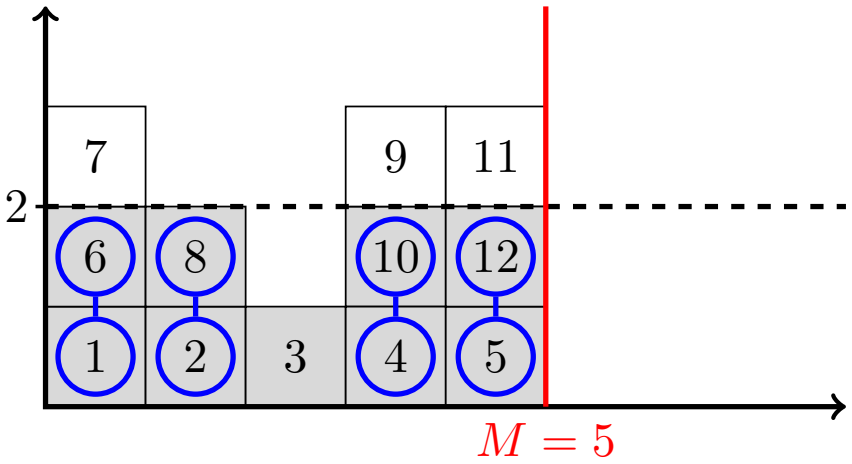


Feasible schedule

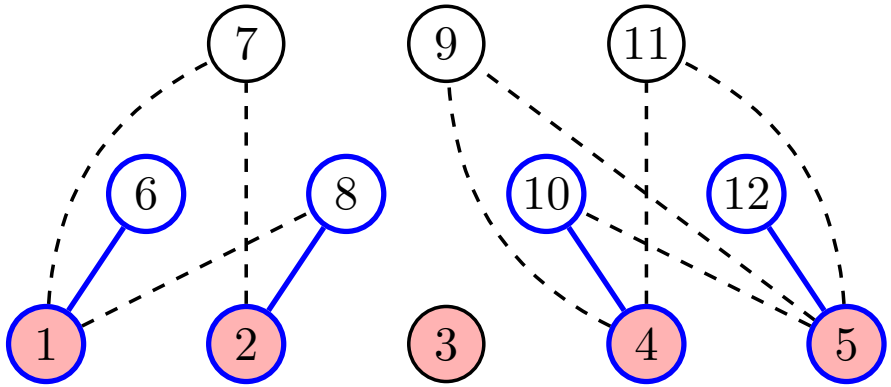


Independence bipartite graph $\widetilde{G}_P$

Solving Q2: when M = size of a critical path



Feasible schedule



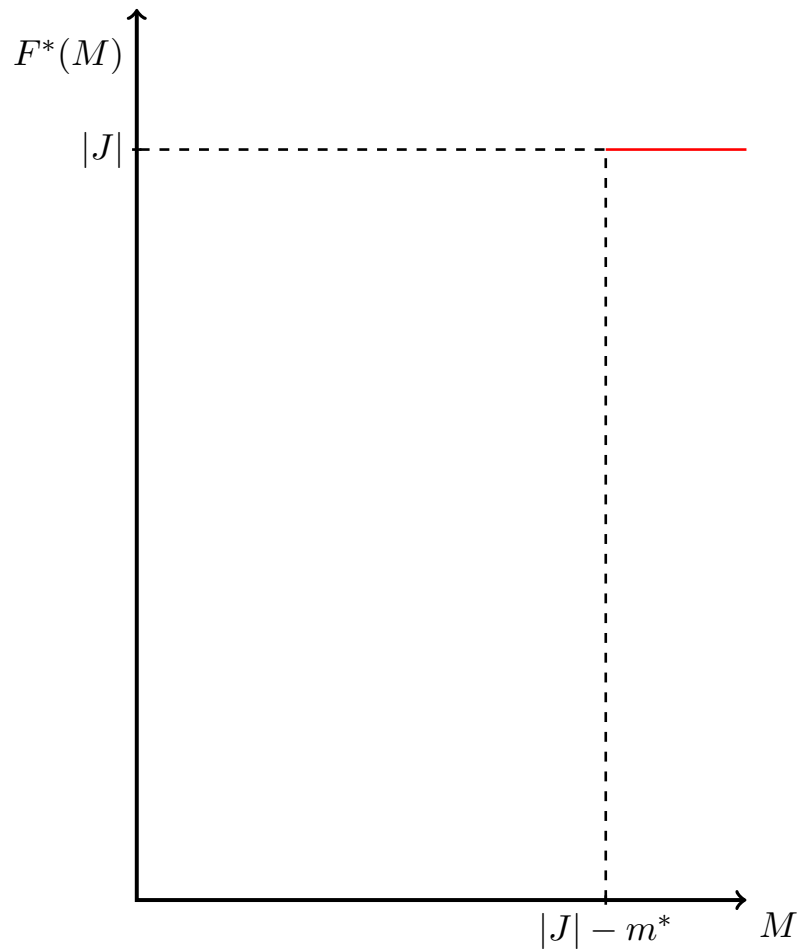
Independence bipartite graph $\widetilde{G}_P$

Theorem ("Fujii-like")

$OPT = |P| + m_P^*$ where P = critical path and m_P^* = size of a max matching in $\widetilde{G}_P$.

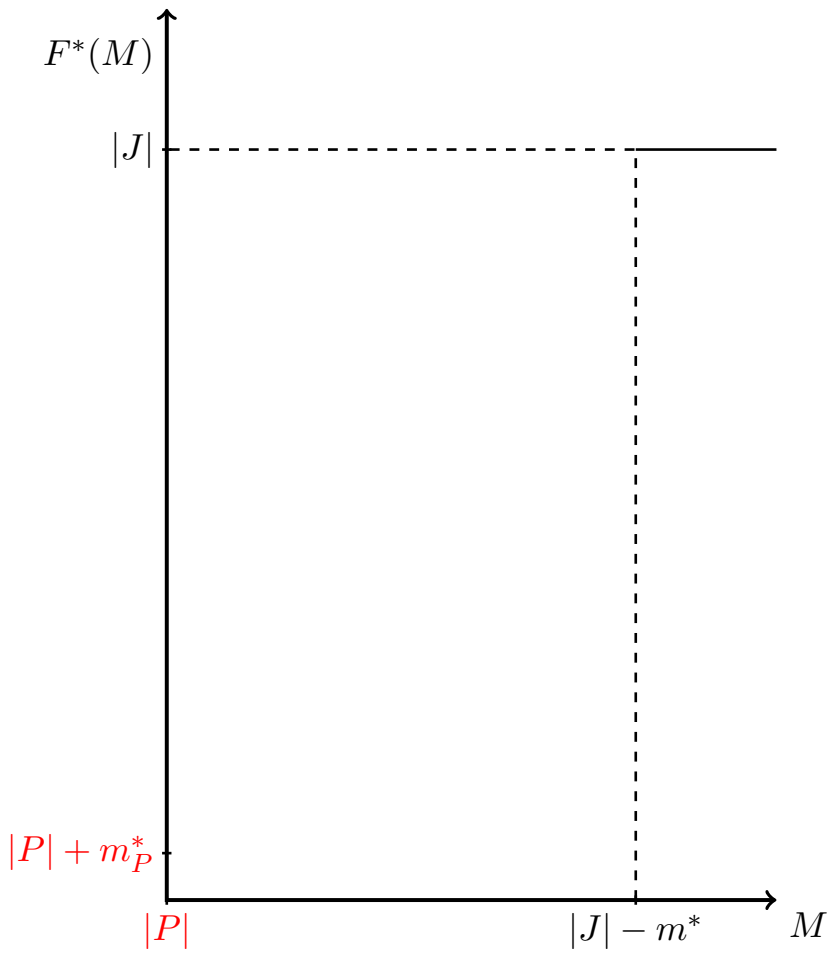
Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

- $M \geq |J| - m^*$: $OPT = |J|$
  (Q1,Fujii et al. 1969)



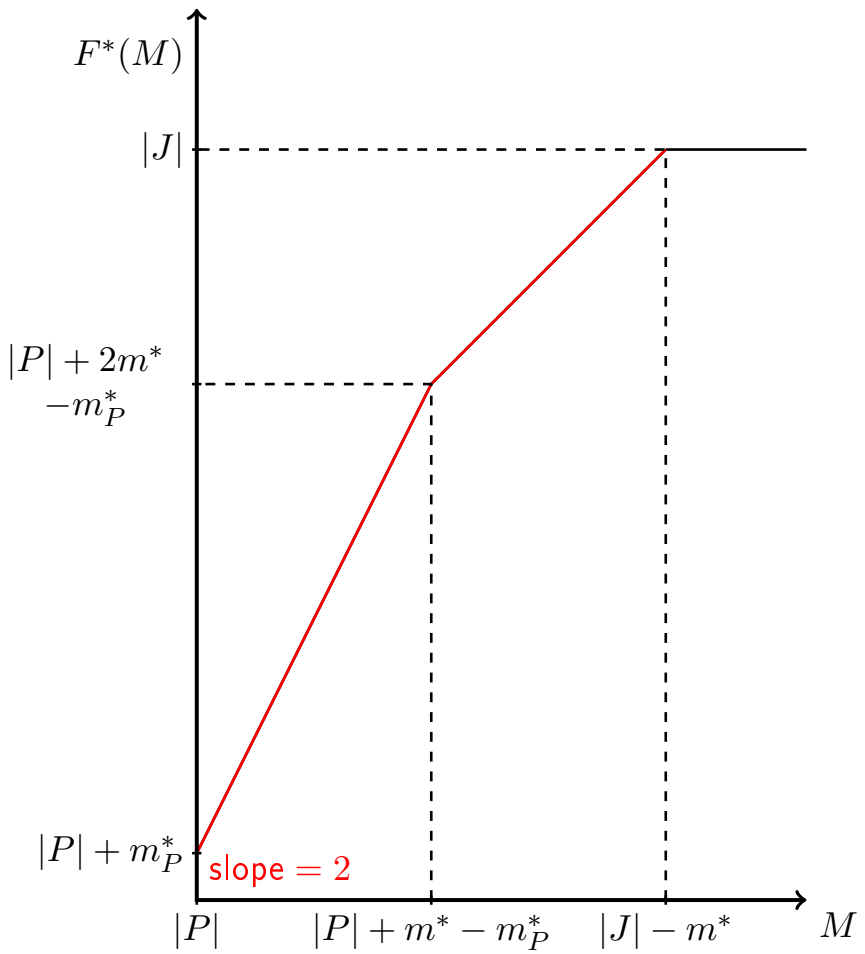
Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

- $M \geq |J| - m^*$: $OPT = |J|$ (Q1,Fujii et al. 1969)
- $M = |P|$: $OPT = |P| + m_P^*$ (Q2, previous slide) m_P^* the size of a max matching in $\widetilde{G}_P$



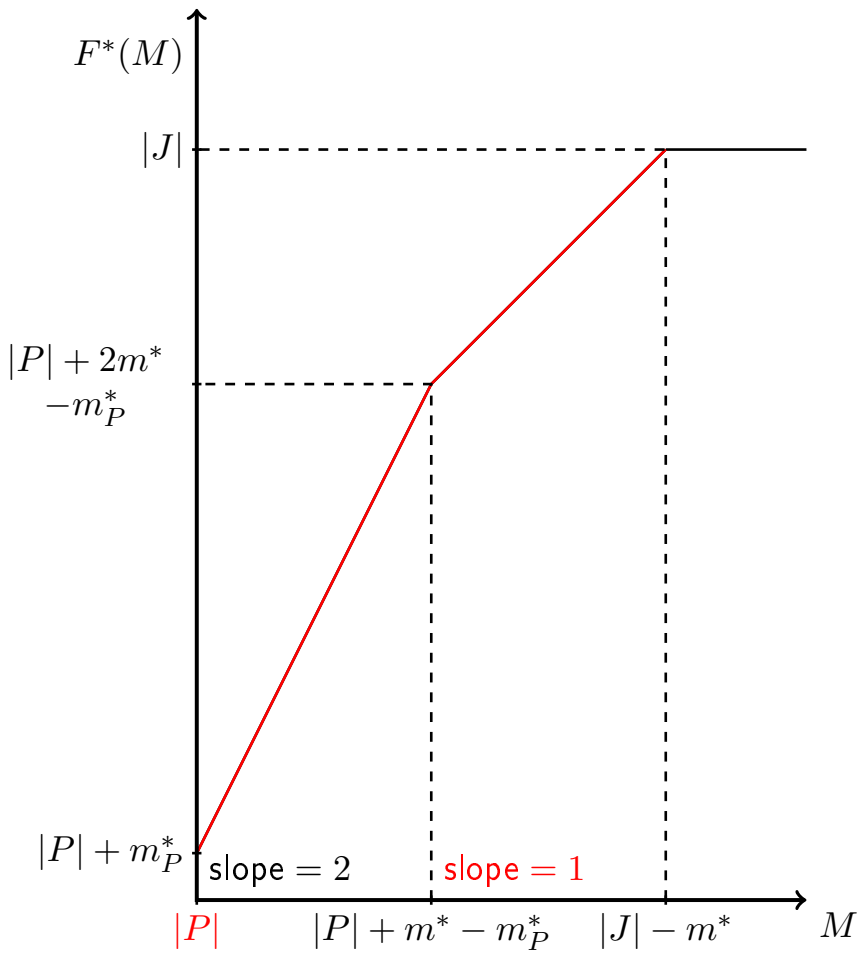
Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

- $M \geq |J| - m^*$: $OPT = |J|$ (Q1,Fujii et al. 1969)
- $M = |P|$: $OPT = |P| + m_P^*$ (Q2, previous slide) m_P^* the size of a max matching in $\widetilde{G}_P$
- $|P| \leq M \leq |P| + m^* - m_P^*$: optimal objective function has slope 2

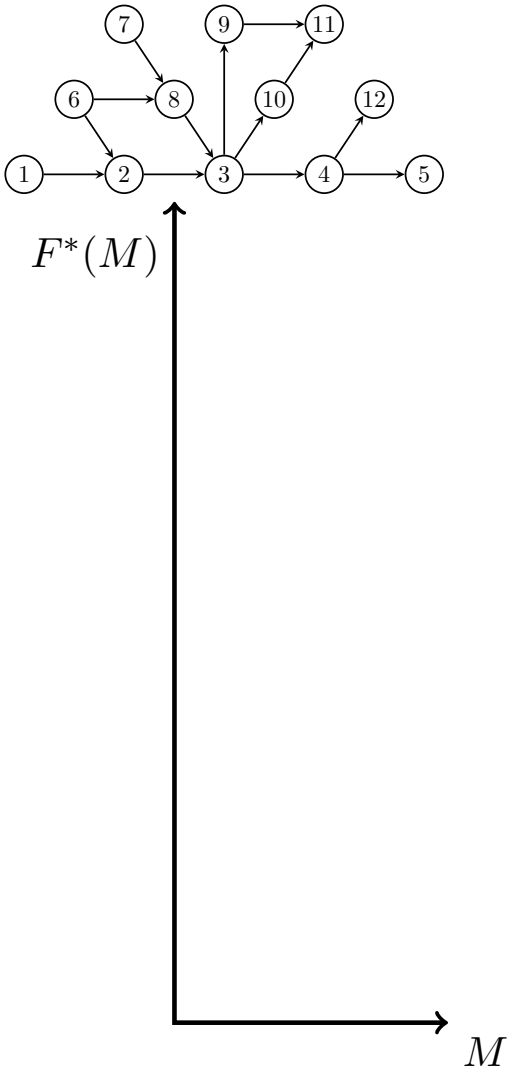


Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

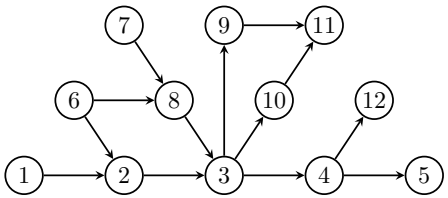
- $M \geq |J| - m^*$: $OPT = |J|$ (Q1,Fujii et al. 1969)
- $M = |P|$: $OPT = |P| + m_P^*$ (Q2, previous slide) m_P^* the size of a max matching in $\widetilde{G}_P$
- $|P| \leq M \leq |P| + m^* - m_P^*$: optimal objective function has slope 2
- $|P| + m^* - m_P^* \leq M \leq |J| - m^*$: optimal objective function has slope 1

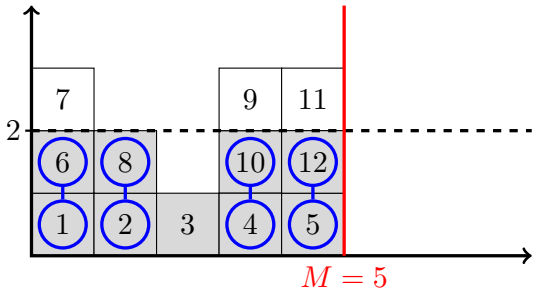


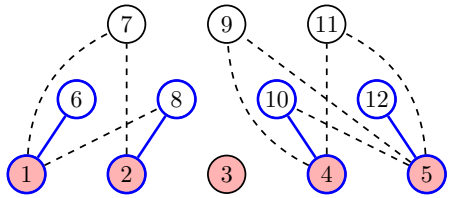
Example



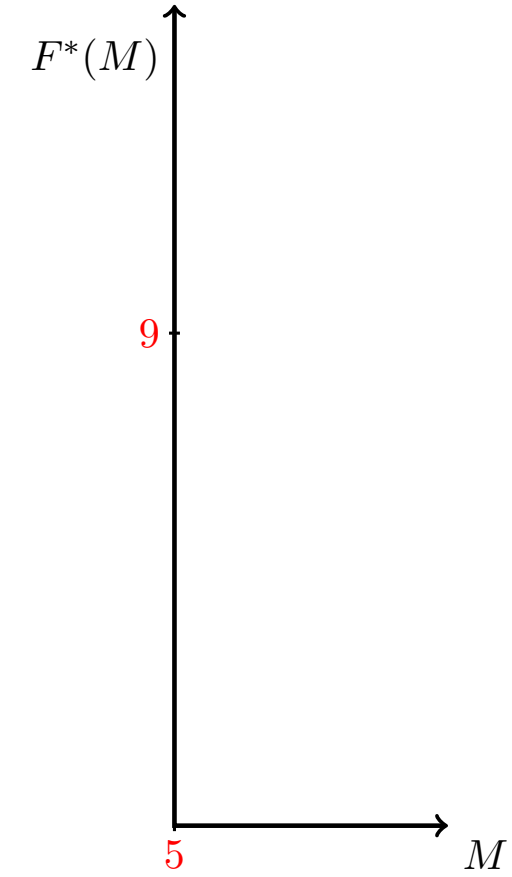
Example



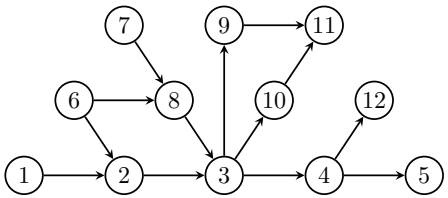


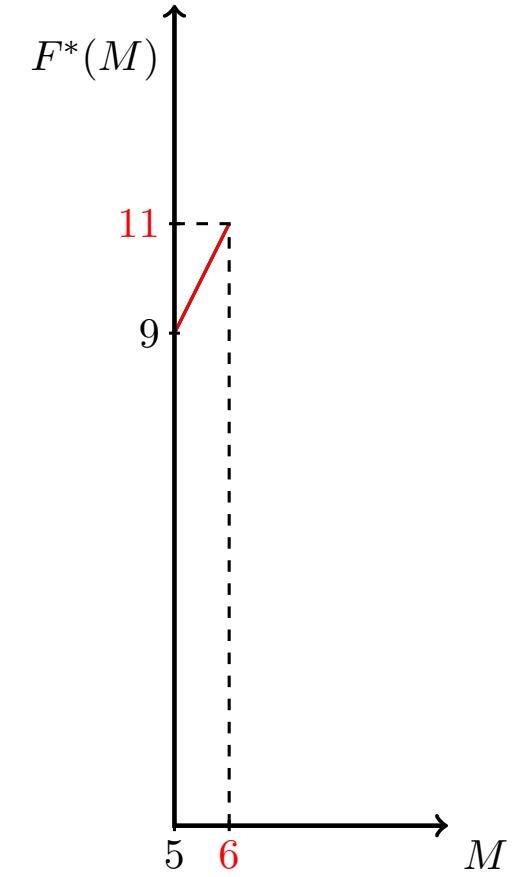


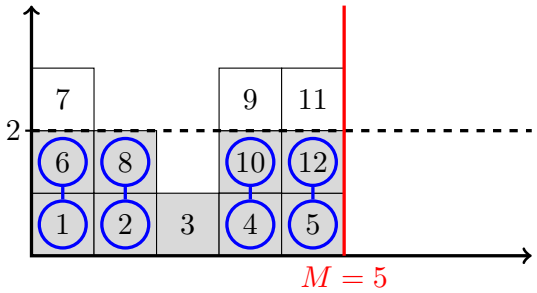
max matching in $\widetilde{G}_P$

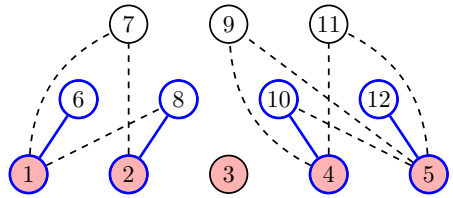


Example

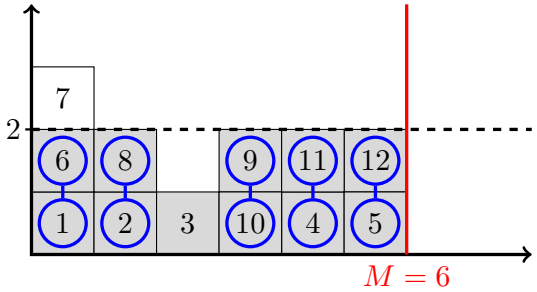


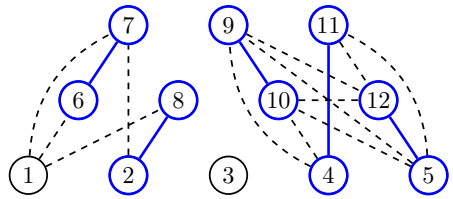






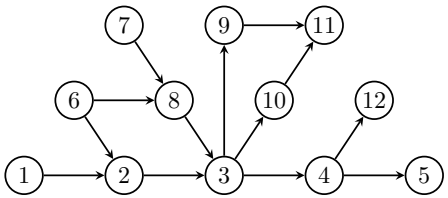
max matching in $\widetilde{G}_P$

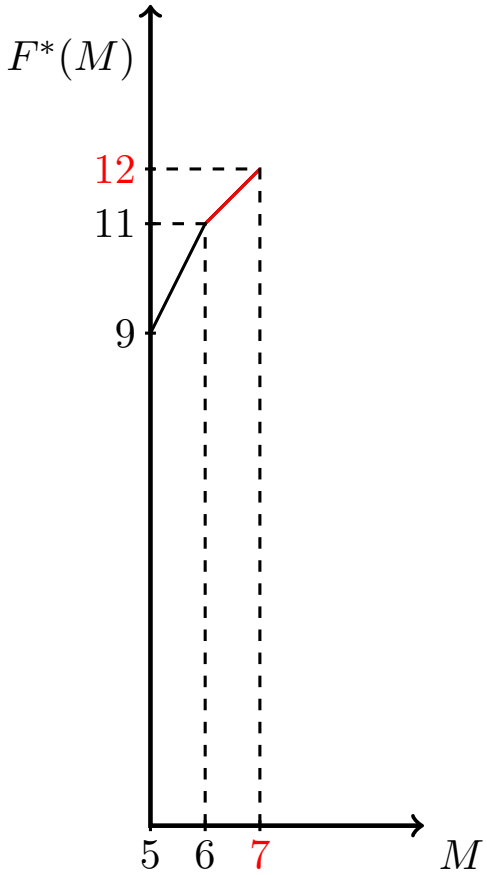


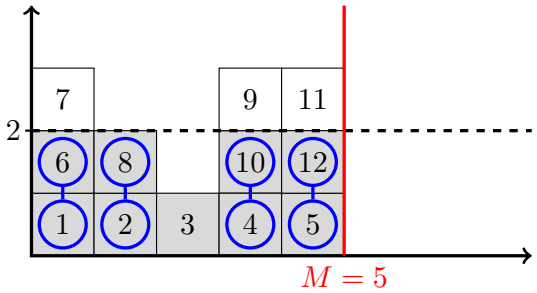


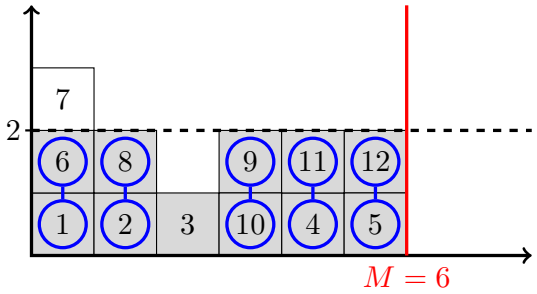
max matching in $\widetilde{G}$

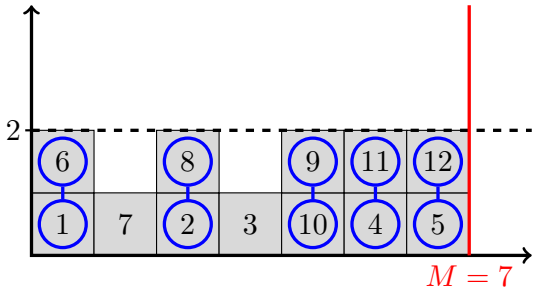
Example

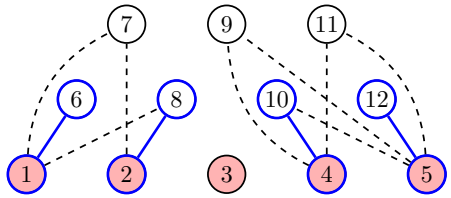




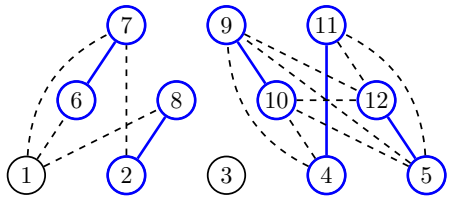






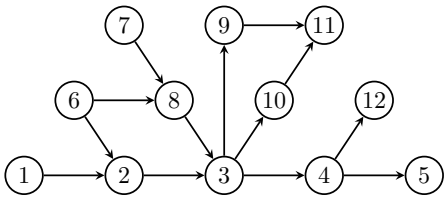


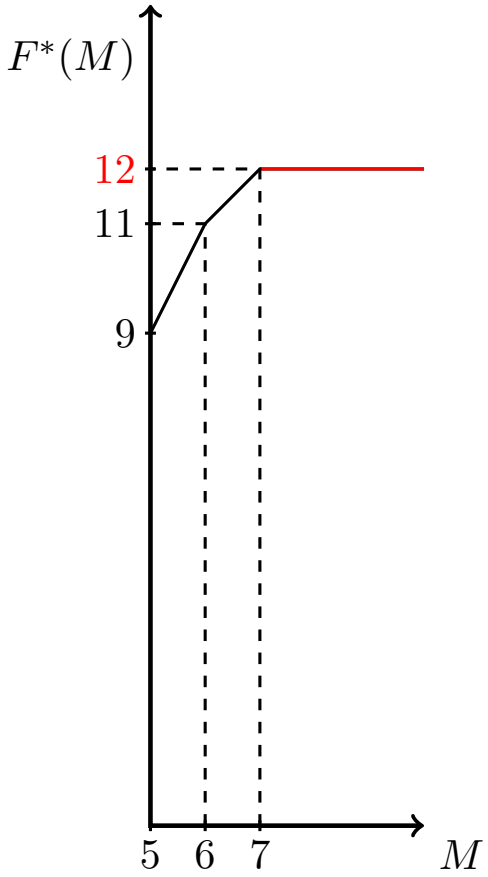
max matching in $\widetilde{G}_P$

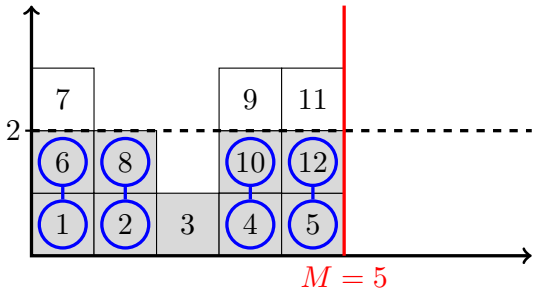


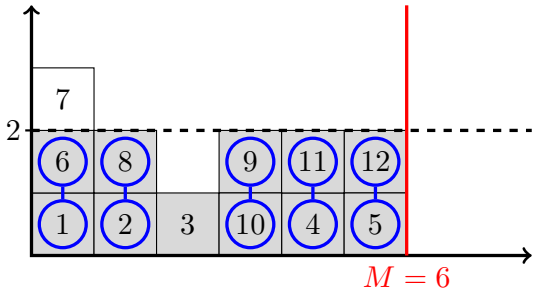
max matching in $\widetilde{G}$

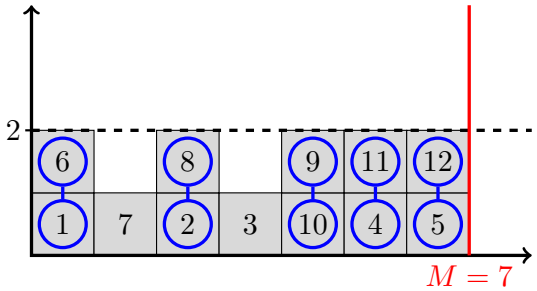
Example

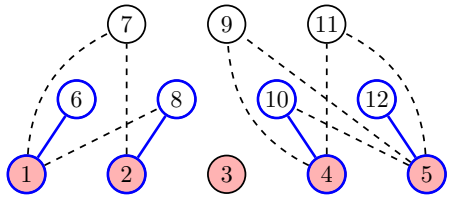




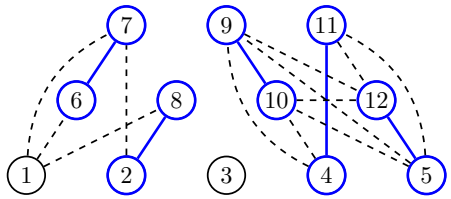








max matching in $\widetilde{G}_P$



max matching in $\widetilde{G}$

Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

Theorem

$L2|prec, p_i = 1, C_{max} \leq M|F$ is polynomial-time solvable

Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

Theorem

$L2|prec, p_i = 1, C_{max} \leq M|F$ *is polynomial-time solvable*

Question

What about non unit processing time?

Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

Theorem

$L2|prec, p_i = 1, C_{max} \leq M|F$ *is polynomial-time solvable*

Question

What about non unit processing time?

The problem becomes strongly NP-hard

Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

Theorem

$L2|prec, p_i = 1, C_{max} \leq M|F$ *is polynomial-time solvable*

Question

What about non unit processing time?

The problem becomes strongly NP-hard

Question

What about $L = 3$ (with $p_i = 1$)?

Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

Theorem

$L2|prec, p_i = 1, C_{max} \leq M|F$ *is polynomial-time solvable*

Question

What about non unit processing time?

The problem becomes strongly NP-hard

Question

What about $L = 3$ (with $p_i = 1$)?

It is an open question!

Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

Theorem

$L2|prec, p_i = 1, C_{max} \leq M|F$ *is polynomial-time solvable*

Question

What about non unit processing time?

The problem becomes strongly NP-hard

Question

What about $L = 3$ (with $p_i = 1$)?

It is an open question!
$L3|prec, p_i = 1, C_{max} \leq M|F$ is strongly related to machine scheduling problem $P3|prec, p_i = 1|C_{max}$...

Solving $L2|prec, p_i = 1, C_{max} \leq M|F$

Theorem

$L2|prec, p_i = 1, C_{max} \leq M|F$ *is polynomial-time solvable*

Question

What about non unit processing time?

The problem becomes strongly NP-hard

Question

What about $L = 3$ (with $p_i = 1$)?

It is an open question!
$L3|prec, p_i = 1, C_{max} \leq M|F$ is strongly related to machine scheduling problem $P3|prec, p_i = 1|C_{max}$...
which is of **unknown complexity**!

Table of complexity results

		$C_{max} \leq M$	$prec, C_{max} \leq M$		r_i, d_i	
			$\emptyset$	*in-tree*	no *pmtn*	*pmtn*
$L = 1$	$\emptyset$	in P	in P		strongly NP-hard	in P
$L = 2$	$\emptyset$	pseudo polynomial	strongly NP-hard		strongly NP-hard	in P
	$p_i = 1$	in P	in P		in P	
$L \in \mathbb{N}$	$\emptyset$	strongly NP-hard				in P
	$p_i = 1$	in P	strongly NP-hard	in P	in P	

Complexity of resource leveling problems [2]

[2] Bendotti, Brunod Indrigo, Chrétienne, Escoffier, Journal of Scheduling 2024.

Approximation results

Polynomial cases are restrictive

[3] Complexity $f(\epsilon)p(n)$. A PTAS can be deduced from [Györgyi,Kis, EJOR 2020]

[4] if $NP \not\subseteq \bigcap_{\epsilon>0} BPTIME(2^{n^\epsilon})$

[5] Bendotti, Brunod Indrigo, Chrétienne, Escoffier, Theoretical Computer Science 2025.

Approximation results

Polynomial cases are restrictive $\rightarrow$ improved tractability of more realistic yet NP-hard cases through approximation results

[3] Complexity $f(\epsilon)p(n)$. A PTAS can be deduced from [Györgyi,Kis, EJOR 2020]

[4] if $NP \nsubseteq \bigcap_{\epsilon>0} BPTIME(2^{n^\epsilon})$

[5] Bendotti, Brunod Indrigo, Chrétienne, Escoffier, Theoretical Computer Science 2025.

Approximation results

Polynomial cases are restrictive $\rightarrow$ improved tractability of more realistic yet NP-hard cases through approximation results

	$C_{max} \leq M$	$C_{max} \leq M, prec$	
		$\emptyset$	*in-tree*
$L = 2$	(str.) NP-hard	str. NP-hard	
$L = 3$		str. NP-hard	
$L \in \mathbb{N}$		str. NP-hard	str. NP-hard

Approximation of resource leveling problems[5]

[3] Complexity $f(\epsilon)p(n)$. A PTAS can be deduced from [Györgyi,Kis, EJOR 2020]

[4] if $NP \nsubseteq \bigcap_{\epsilon>0} BPTIME(2^{n^\epsilon})$

[5] Bendotti, Brunod Indrigo, Chrétienne, Escoffier, Theoretical Computer Science 2025.

Approximation results

Polynomial cases are restrictive $\rightarrow$ improved tractability of more realistic yet NP-hard cases through approximation results

	$C_{max} \leq M$	$C_{max} \leq M, prec$	
		$\emptyset$	*in-tree*
$L = 2$	(str.) NP-hard	str. NP-hard ratio $\frac{2}{3}$	
$L = 3$	EPTAS[3] ratio $\frac{7}{8}$ in $O(\lvert J\rvert \log(\lvert J\rvert))$	str. NP-hard ratio $\frac{3}{5}$	
$L \in \mathbb{N}$		str. NP-hard No PTAS[4]	str. NP-hard ratio $\frac{1}{2}$

Approximation of resource leveling problems[5]

[3] Complexity $f(\epsilon)p(n)$. A PTAS can be deduced from [Györgyi,Kis, EJOR 2020]

[4] if $NP \nsubseteq \bigcap_{\epsilon>0} BPTIME(2^{n^\epsilon})$

[5] Bendotti, Brunod Indrigo, Chrétienne, Escoffier, Theoretical Computer Science 2025.

Inapproximability

Proposition

$L|prec, C_{max} \leq M|F$ admits no PTAS under a classical complexity assumption.

Inapproximability

Proposition

$L|bipartite, p_i = 1, C_{max} \leq 3|F$ admits no PTAS under a classical complexity assumption.

- Holds even if $prec = bipartite$, $p_i = 1$ and $M = 3$

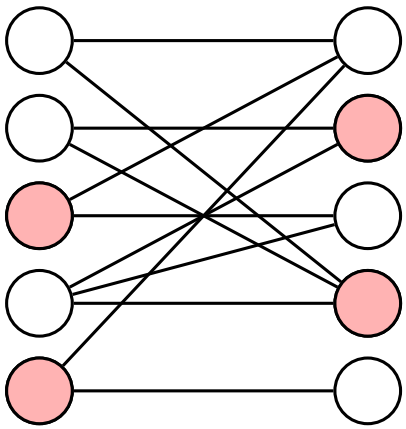
Inapproximability

Proposition

$L|bipartite, p_i = 1, C_{max} \leq 3|F$ admits no PTAS under a classical complexity assumption.

- Holds even if $prec = bipartite$, $p_i = 1$ and $M = 3$
- Reduction from a Balanced Bipartite Independent Set Problem (**Khot2006**)

Inapproximability

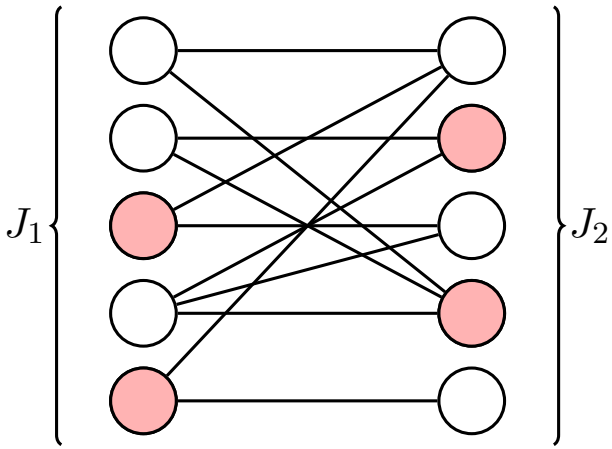


Independent Set

Find a set S, $|S| \geq k$, of pairwise non adjacent vertices

- Hard problem (hard to approximate as well)
- Easy in bipartite graph ...

Inapproximability



Independent Set

Find a set S, $|S| \geq k$, of pairwise non adjacent vertices

- Hard problem (hard to approximate as well)
- Easy in bipartite graph ...
- ... Balanced version is hard: find an IS S with $|S \cap J_1| \geq k$ and $|S \cap J_2| \geq k$.
  No PTAS under some complexity assumption ((**Khot2006**)).

Inapproximability

Proposition

$L|bipartite, p_i = 1, C_{max} \leq 3|F$ admits no PTAS under a classical complexity assumption.

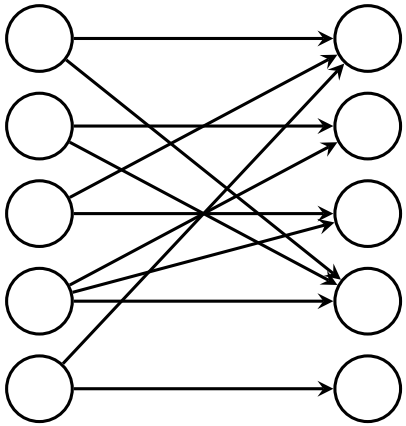
- Holds even if $prec = bipartite$, $p_i = 1$ and $M = 3$
- Reduction from a Balanced Bipartite Independent Set Problem (**Khot2006**)

Inapproximability

Proposition

$L|bipartite, p_i = 1, C_{max} \leq 3|F$ admits no PTAS under a classical complexity assumption.

- Holds even if $prec = bipartite$, $p_i = 1$ and $M = 3$
- Reduction from a Balanced Bipartite Independent Set Problem (**Khot2006**)

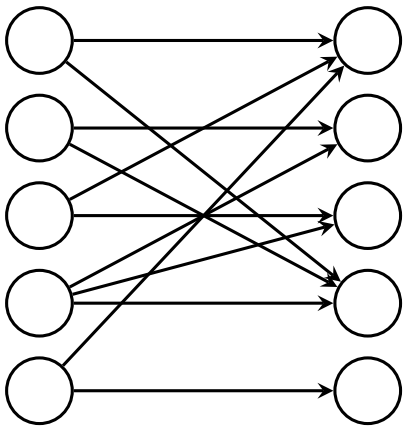


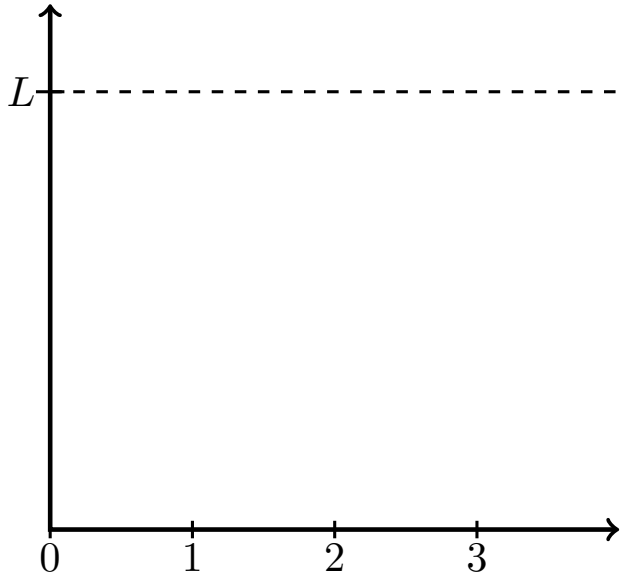
Inapproximability

Proposition

$L|bipartite, p_i = 1, C_{max} \leq 3|F$ admits no PTAS under a classical complexity assumption.

- Holds even if $prec = bipartite$, $p_i = 1$ and $M = 3$
- Reduction from a Balanced Bipartite Independent Set Problem (**Khot2006**)



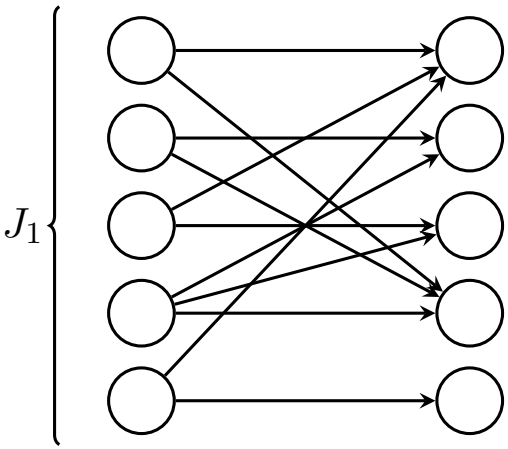


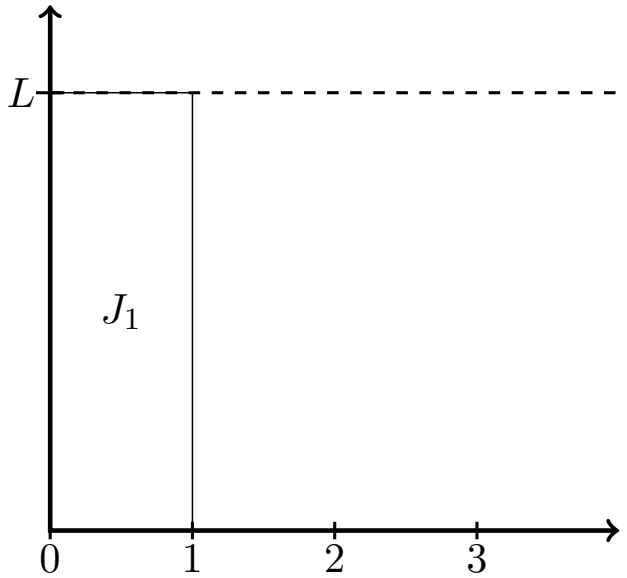
Inapproximability

Proposition

$L|bipartite, p_i = 1, C_{max} \leq 3|F$ admits no PTAS under a classical complexity assumption.

- Holds even if $prec = bipartite$, $p_i = 1$ and $M = 3$
- Reduction from a Balanced Bipartite Independent Set Problem (**Khot2006**)



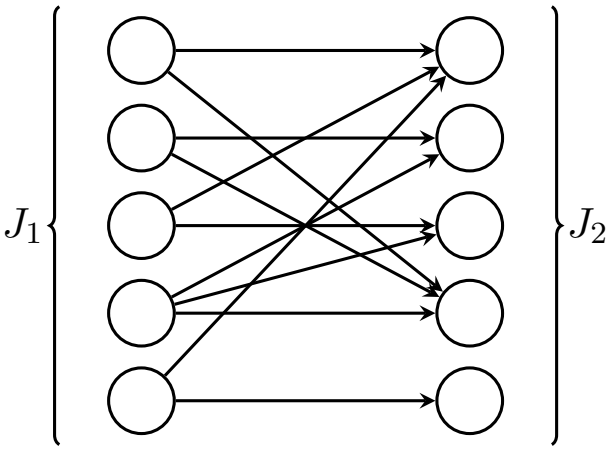


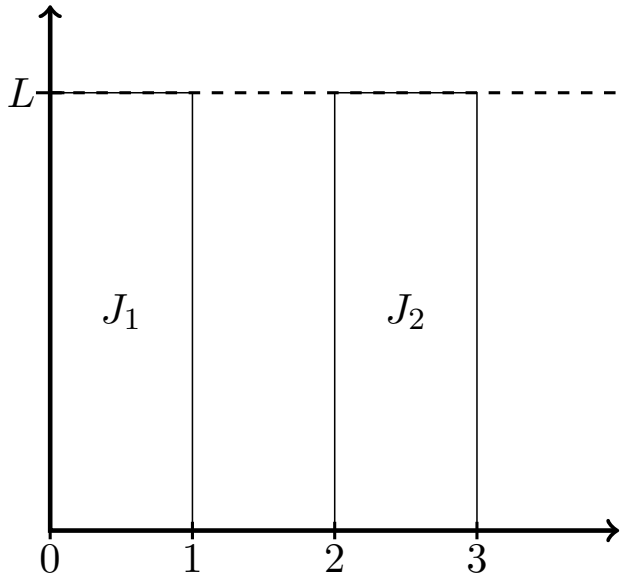
Inapproximability

Proposition

$L|bipartite, p_i = 1, C_{max} \leq 3|F$ admits no PTAS under a classical complexity assumption.

- Holds even if $prec = bipartite$, $p_i = 1$ and $M = 3$
- Reduction from a Balanced Bipartite Independent Set Problem (**Khot2006**)



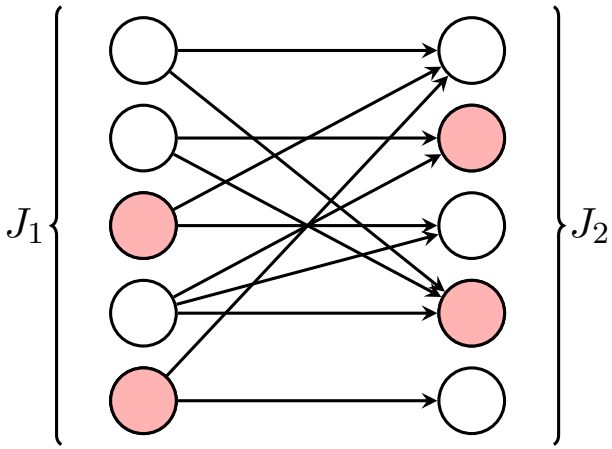


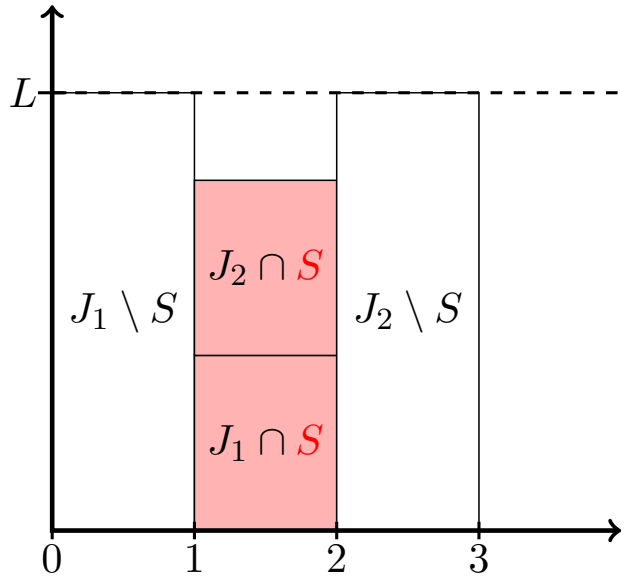
Inapproximability

Proposition

$L|bipartite, p_i = 1, C_{max} \leq 3|F$ admits no PTAS under a classical complexity assumption.

- Holds even if $prec = bipartite$, $p_i = 1$ and $M = 3$
- Reduction from a Balanced Bipartite Independent Set Problem (**Khot2006**)





Approximation results

	$C_{max} \leq M$	$C_{max} \leq M, prec$	
		$\emptyset$	$in\text{-}tree$
$L = 2$	str. NP-hard	str. NP-hard ratio $\frac{2}{3}$	
$L = 3$	EPTAS $ratio \frac{7}{8}$ in $O(\lvert J\rvert \log(\lvert J\rvert))$	str. NP-hard ratio $\frac{3}{5}$	
$L \in \mathbb{N}$		str. NP-hard No PTAS[6]	str. NP-hard ratio $\frac{1}{2}$

Approximation of resource leveling problems[7]

[6] if $NP \not\subseteq \bigcap_{\epsilon>0} BPTIME(2^{n^\epsilon})$

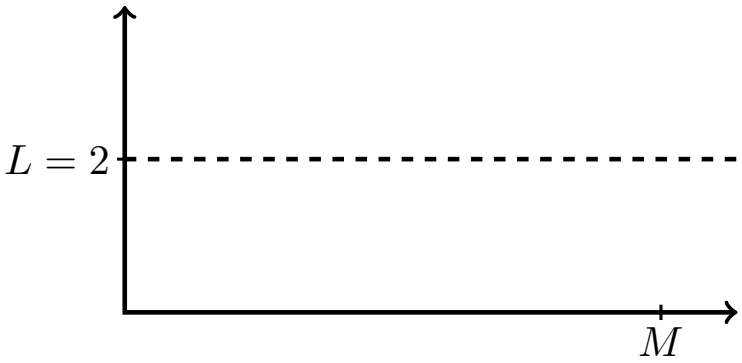
[7] Bendotti, Brunod Indrigo, Chrétienne, Escoffier, Theoretical Computer Science 2025.

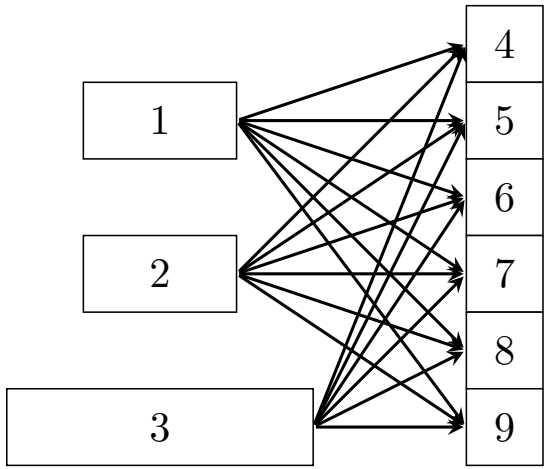
$L|prec, C_{max} \leq M|F$: A list algorithm

A feasible solution for $L|prec, C_{max} \leq M|F$ can be built using a simple list algorithm

$L|prec, C_{max} \leq M|F$: A list algorithm

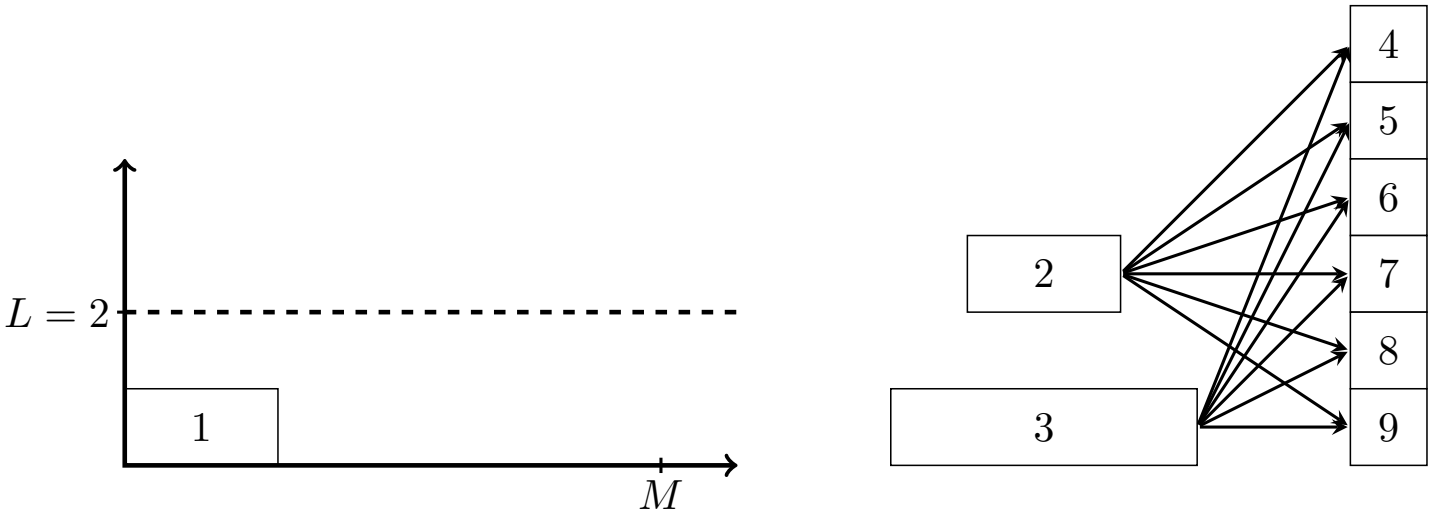
A feasible solution for $L|prec, C_{max} \leq M|F$ can be built using a simple list algorithm





$L|prec, C_{max} \leq M|F$: A list algorithm

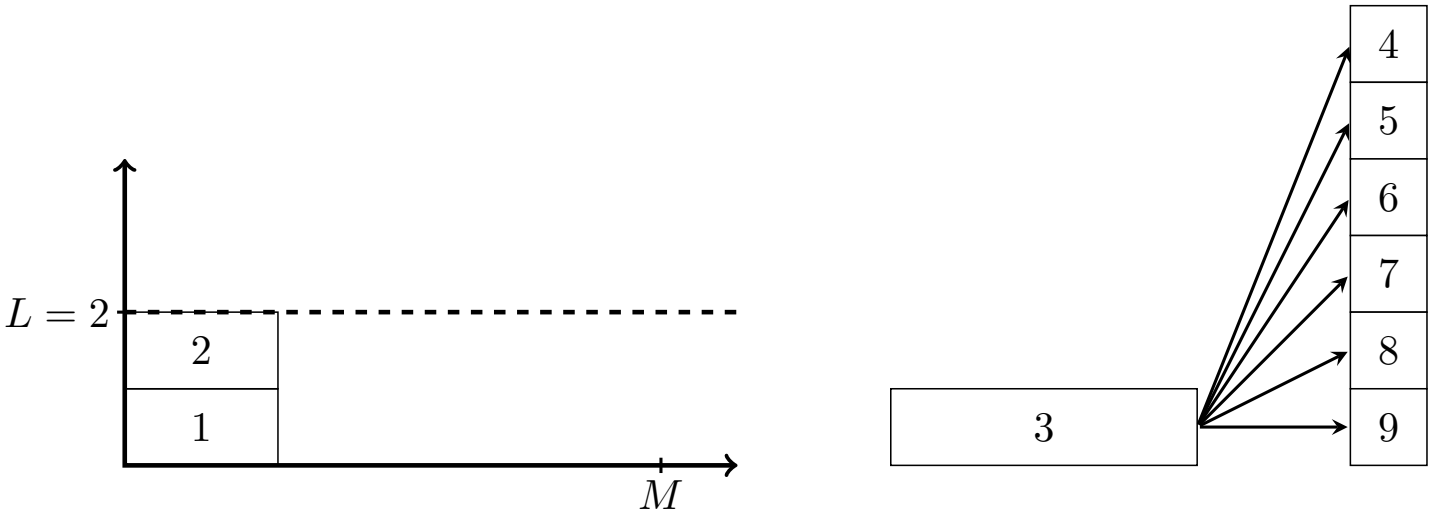
A feasible solution for $L|prec, C_{max} \leq M|F$ can be built using a simple list algorithm



- Jobs scheduled in a topological order

$L|prec, C_{max} \leq M|F$: A list algorithm

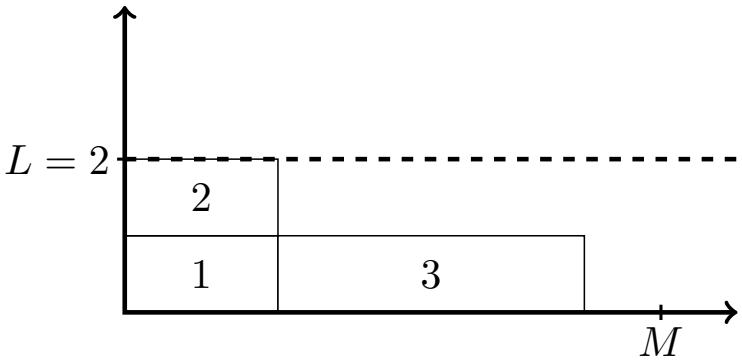
A feasible solution for $L|prec, C_{max} \leq M|F$ can be built using a simple list algorithm



- Jobs scheduled in a topological order

$L|prec, C_{max} \leq M|F$: A list algorithm

A feasible solution for $L|prec, C_{max} \leq M|F$ can be built using a simple list algorithm

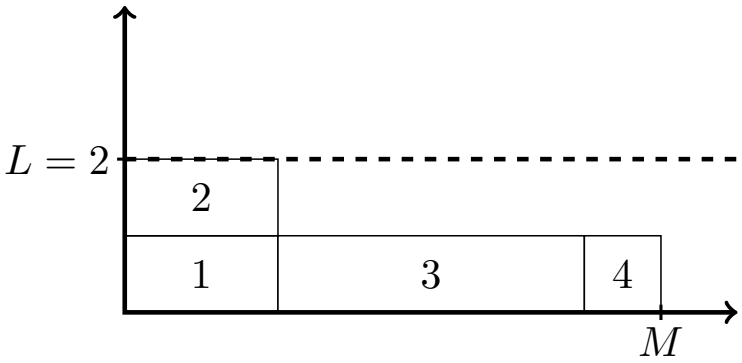


4
5
6
7
8
9

- Jobs scheduled in a topological order
- Trying not to exceed L

$L|prec, C_{max} \leq M|F$: A list algorithm

A feasible solution for $L|prec, C_{max} \leq M|F$ can be built using a simple list algorithm

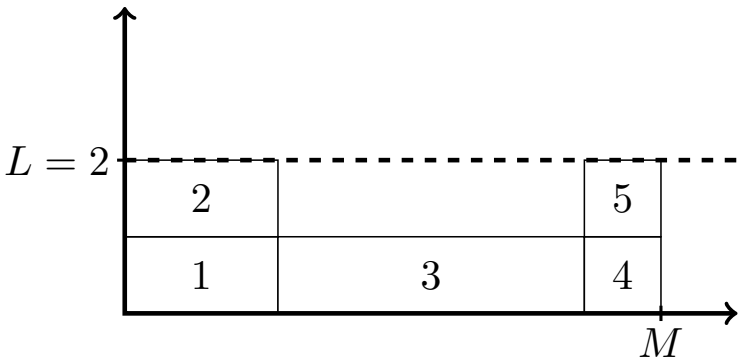


5
6
7
8
9

- Jobs scheduled in a topological order
- Trying not to exceed L

$L|prec, C_{max} \leq M|F$: A list algorithm

A feasible solution for $L|prec, C_{max} \leq M|F$ can be built using a simple list algorithm

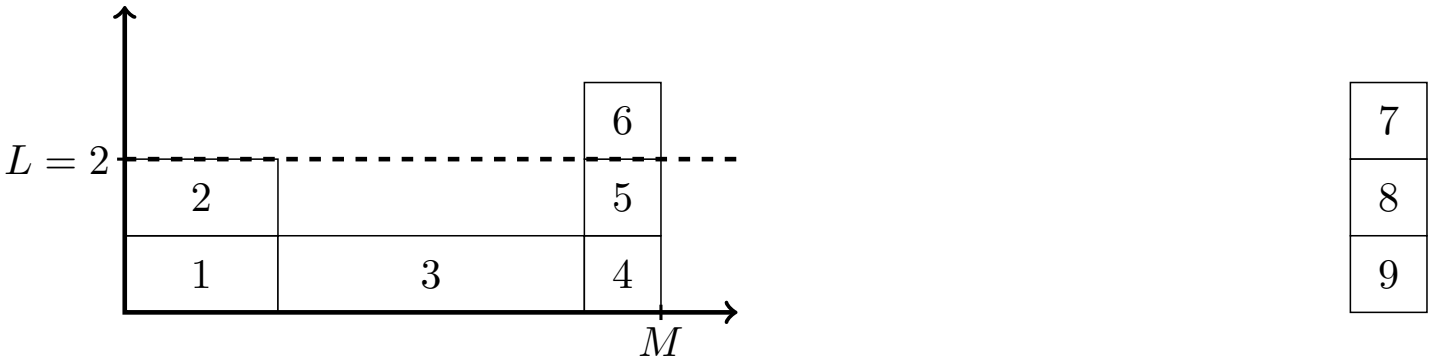


6
7
8
9

- Jobs scheduled in a topological order
- Trying not to exceed L

$L|prec, C_{max} \leq M|F$: A list algorithm

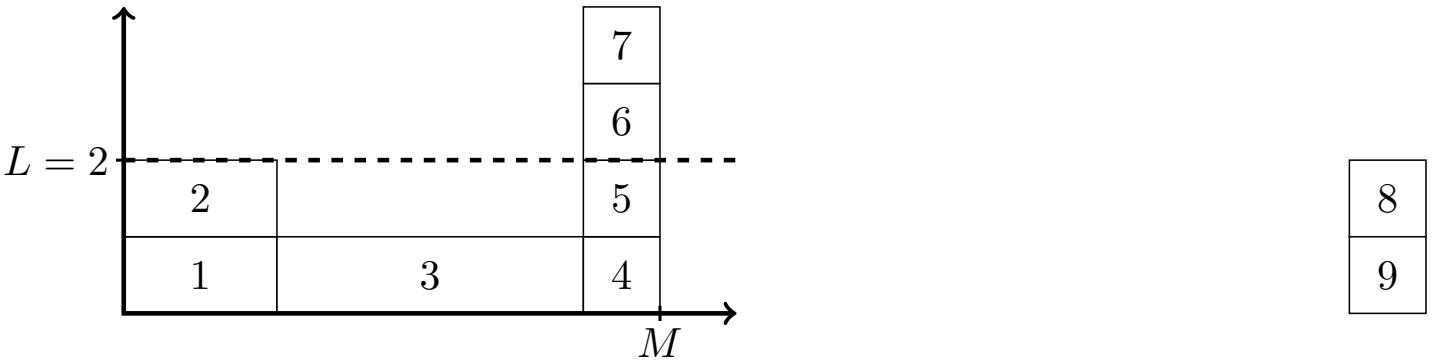
A feasible solution for $L|prec, C_{max} \leq M|F$ can be built using a simple list algorithm



- Jobs scheduled in a topological order
- Trying not to exceed L
- Exceeding L when forced by a critical path

$L|prec, C_{max} \leq M|F$: A list algorithm

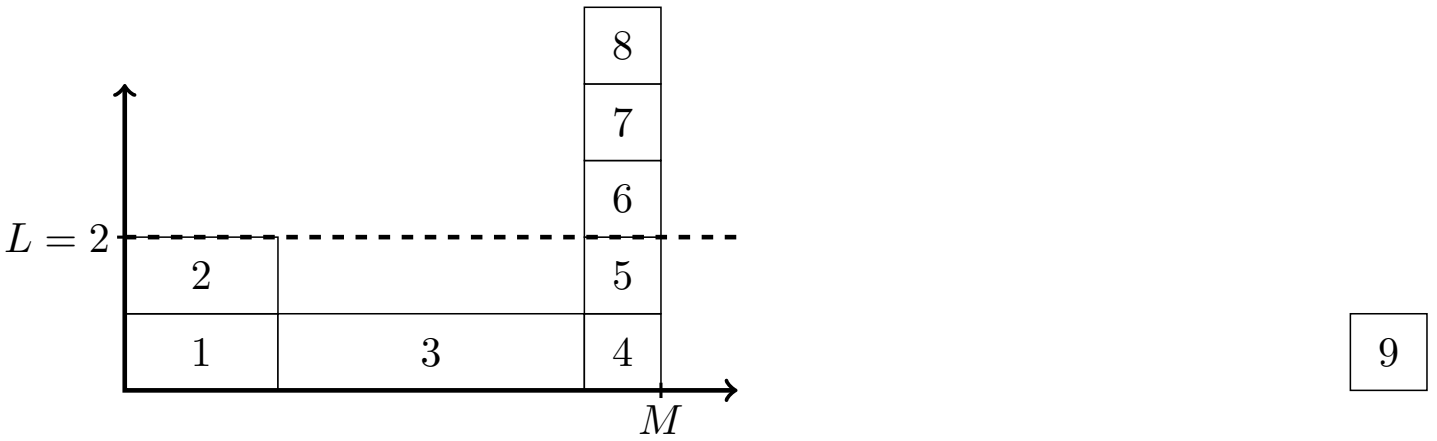
A feasible solution for $L|prec, C_{max} \leq M|F$ can be built using a simple list algorithm



- Jobs scheduled in a topological order
- Trying not to exceed L
- Exceeding L when forced by a critical path

$L|prec, C_{max} \leq M|F$: A list algorithm

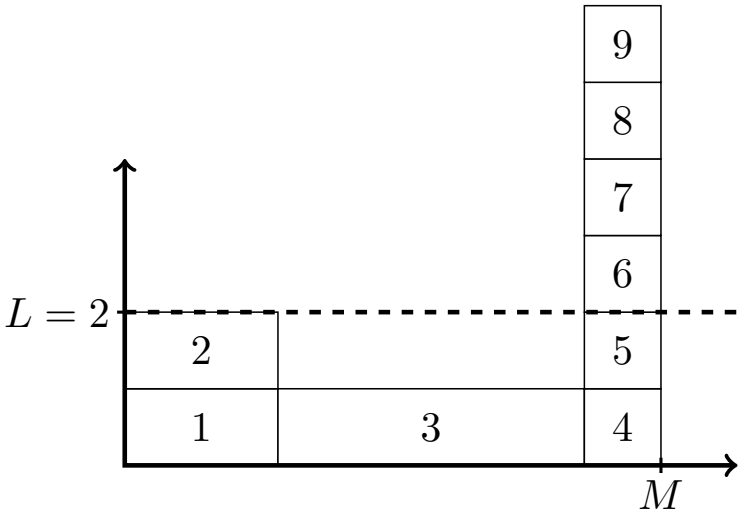
A feasible solution for $L|prec, C_{max} \leq M|F$ can be built using a simple list algorithm



- Jobs scheduled in a topological order
- Trying not to exceed L
- Exceeding L when forced by a critical path

$L|prec, C_{max} \leq M|F$: A list algorithm

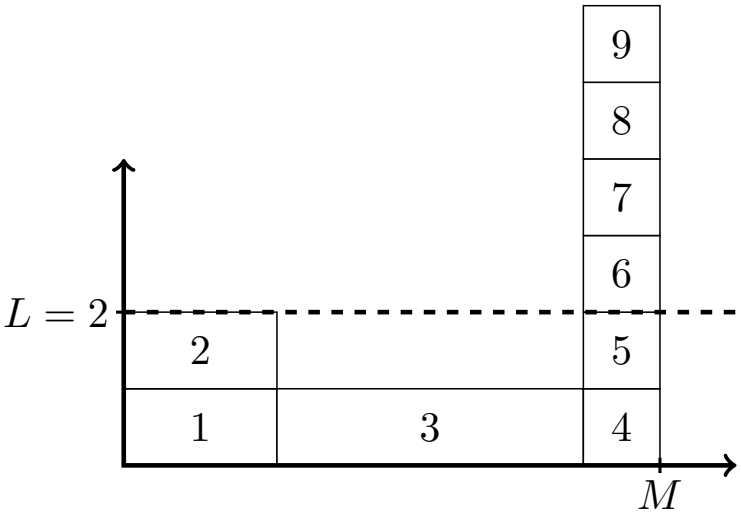
A feasible solution for $L|prec, C_{max} \leq M|F$ can be built using a simple list algorithm

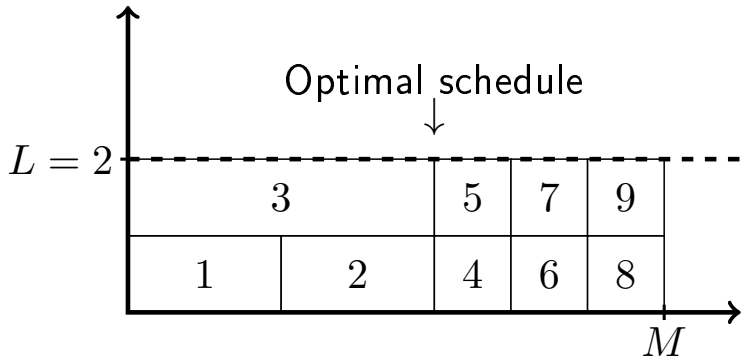


- Jobs scheduled in a topological order
- Trying not to exceed L
- Exceeding L when forced by a critical path

$L|prec, C_{max} \leq M|F$: A list algorithm

A feasible solution for $L|prec, C_{max} \leq M|F$ can be built using a simple list algorithm





- Jobs scheduled in a topological order
- Trying not to exceed L
- Exceeding L when forced by a critical path
- Not optimal

Approximation of $L2|prec, C_{max} \leq M|F$

Lemma

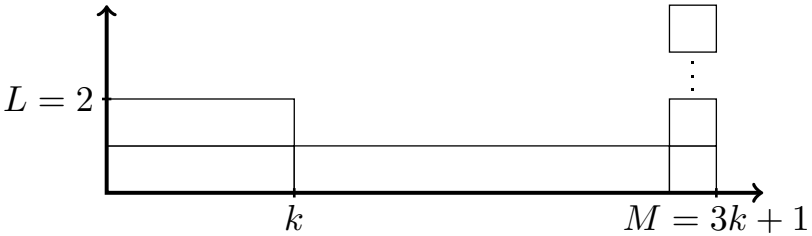
The list algorithm has approximation ratio ***at most*** $\frac{2}{3}$ *for* $L2|prec, C_{max} \leq M|F$

Approximation of $L2|prec, C_{max} \leq M|F$

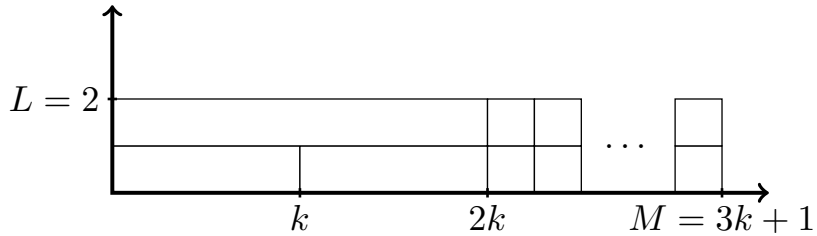
Lemma

The list algorithm has approximation ratio ***at most*** $\frac{2}{3}$ *for* $L2|prec, C_{max} \leq M|F$

Idea of proof:



List algorithm schedule



Optimal schedule

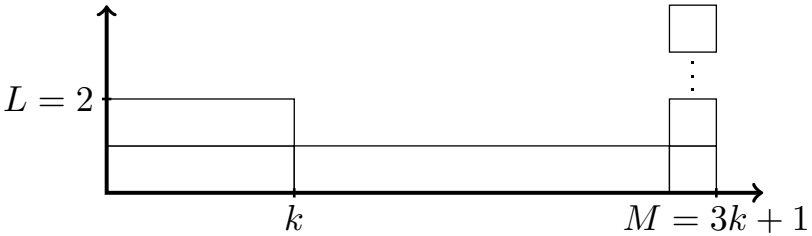
- For $k \in \mathbb{N}$, there is an instance with approximation ratio $\frac{2k+1}{3k+1}$

Approximation of $L2|prec, C_{max} \leq M|F$

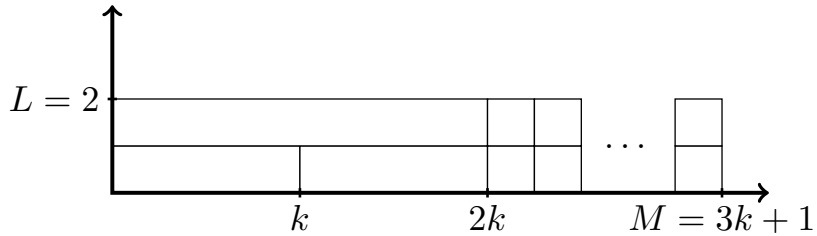
Lemma

The list algorithm has approximation ratio ***at most*** $\frac{2}{3}$ *for* $L2|prec, C_{max} \leq M|F$

Idea of proof:



List algorithm schedule



Optimal schedule

- For $k \in \mathbb{N}$, there is an instance with approximation ratio $\frac{2k+1}{3k+1}$
- The ratio can be arbitrarily close to $\frac{2}{3}$ for large k

Approximation of $L2|prec, C_{max} \leq M|F$

Lemma

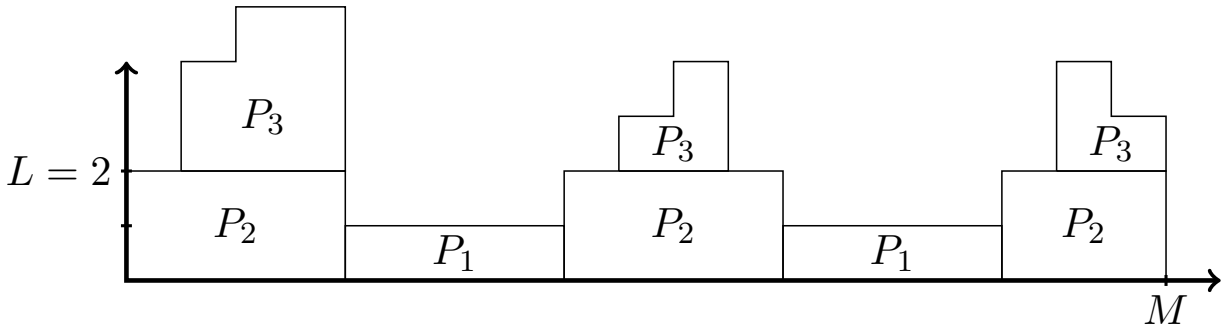
The list algorithm has approximation ratio ***at least*** $\frac{2}{3}$ *for* $L2|prec, C_{max} \leq M|F$

Approximation of $L2|prec, C_{max} \leq M|F$

Lemma

The list algorithm has approximation ratio ***at least*** $\frac{2}{3}$ *for* $L2|prec, C_{max} \leq M|F$

Idea of proof:

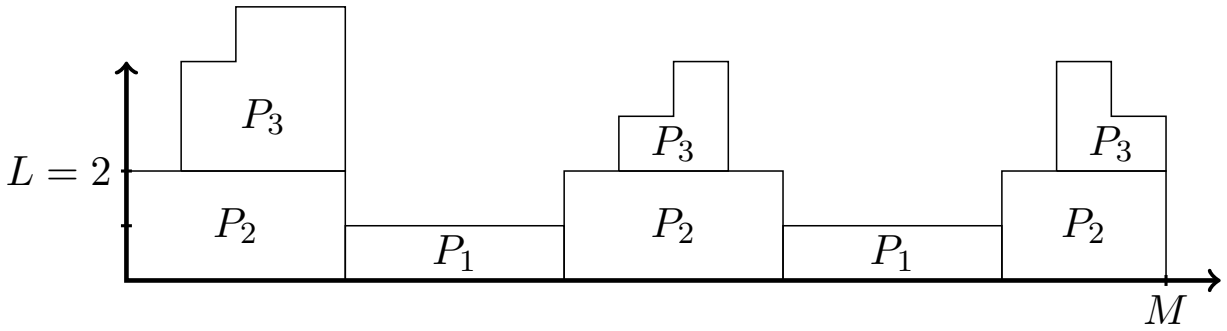


Approximation of $L2|prec, C_{max} \leq M|F$

Lemma

The list algorithm has approximation ratio ***at least*** $\frac{2}{3}$ *for* $L2|prec, C_{max} \leq M|F$

Idea of proof:



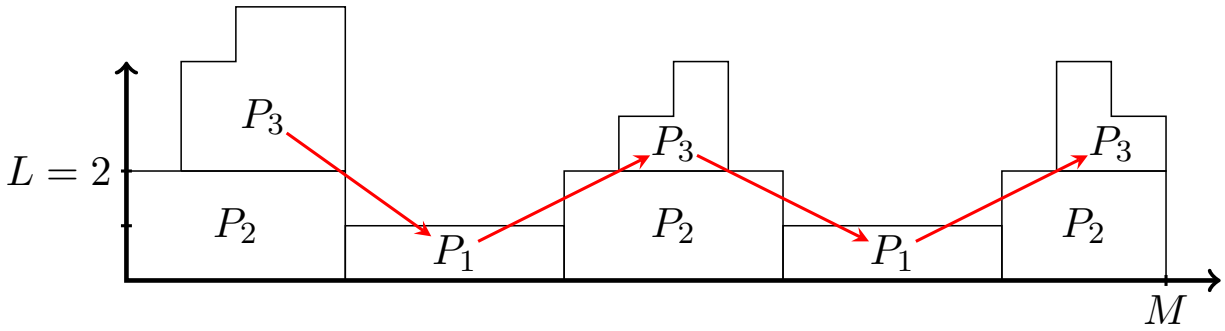
- $F(x) = P_1 + P_2$

Approximation of $L2|prec, C_{max} \leq M|F$

Lemma

The list algorithm has approximation ratio ***at least*** $\frac{2}{3}$ *for* $L2|prec, C_{max} \leq M|F$

Idea of proof:



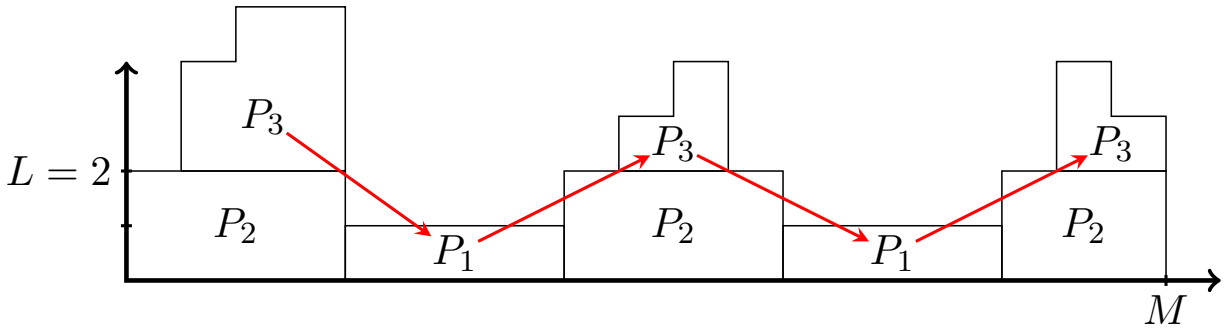
- $F(x) = P_1 + P_2$
- Impossible to schedule P_1 and P_3 in parallel due to precedence constraints $\Rightarrow F^* \leq F(x) + P_2$

Approximation of $L2|prec, C_{max} \leq M|F$

Lemma

The list algorithm has approximation ratio ***at least*** $\frac{2}{3}$ *for* $L2|prec, C_{max} \leq M|F$

Idea of proof:



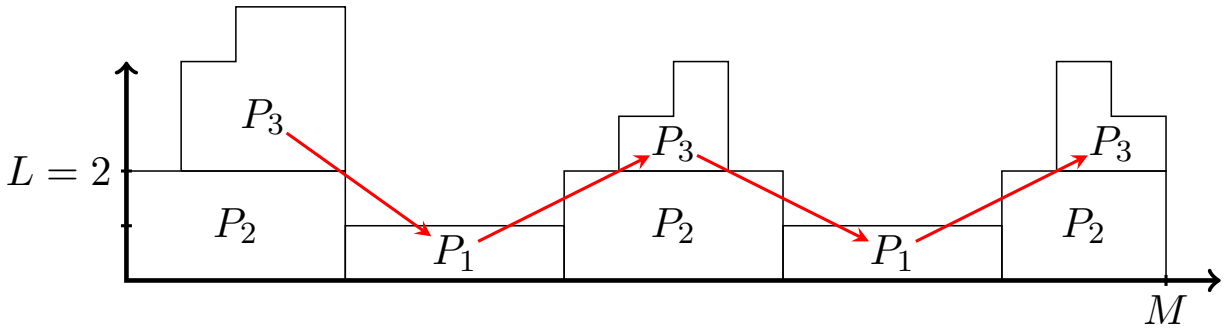
- $F(x) = P_1 + P_2$
- Impossible to schedule P_1 and P_3 in parallel due to precedence constraints $\Rightarrow F^* \leq F(x) + P_2$
- Natural bound: $F^* \leq 2M = P_2 + 2P_1$

Approximation of $L2|prec, C_{max} \leq M|F$

Lemma

The list algorithm has approximation ratio ***at least*** $\frac{2}{3}$ *for* $L2|prec, C_{max} \leq M|F$

Idea of proof:



- $F(x) = P_1 + P_2$
- Impossible to schedule P_1 and P_3 in parallel due to precedence constraints $\Rightarrow F^* \leq F(x) + P_2$
- Natural bound: $F^* \leq 2M = P_2 + 2P_1$
- Summing up to: $2F^* \leq 3F(x)$

Approximation of $L2|prec, C_{max} \leq M|F$

Proposition

The list algorithm has (tight) approximation ratio $\frac{2}{3}$ *for* $L2|prec, C_{max} \leq M|F$

Approximation of $L2|prec, C_{max} \leq M|F$

Proposition

The list algorithm has (tight) approximation ratio $\frac{2}{3}$ *for* $L2|prec, C_{max} \leq M|F$

Question

What about $L = 3$?

Approximation of $L2|prec, C_{max} \leq M|F$

Proposition

The list algorithm has (tight) approximation ratio $\frac{2}{3}$ *for* $L2|prec, C_{max} \leq M|F$

Question

What about $L = 3$?

Proposition

The list algorithm has (tight) approximation ratio $\frac{3}{5}$ *for* $L3|prec, C_{max} \leq M|F$

Approximation of $L3|prec, C_{max} \leq M|F$

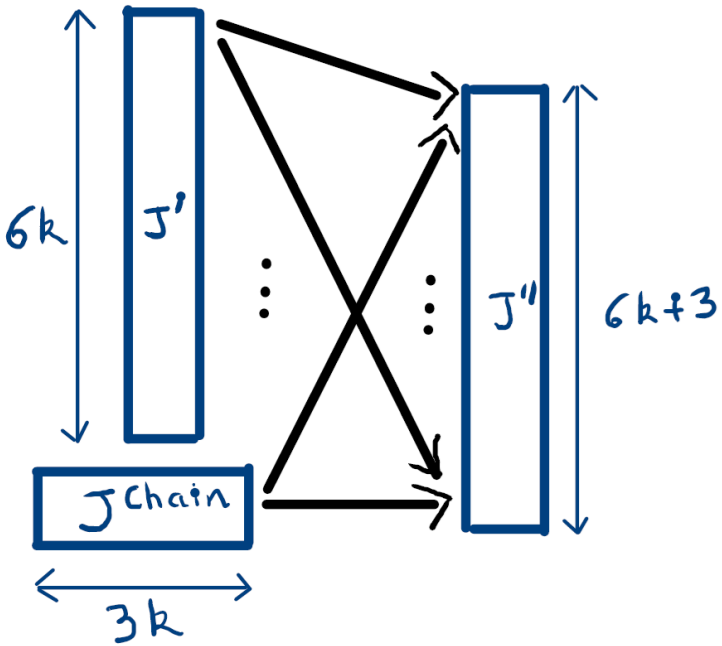
Lemma

The list algorithm has approximation ratio ***at most*** $\frac{3}{5}$ *for* $L3|prec, C_{max} \leq M|F$

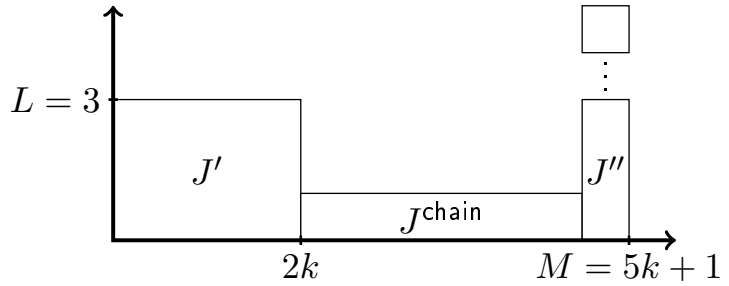
Approximation of $L3|prec, C_{max} \leq M|F$

Lemma

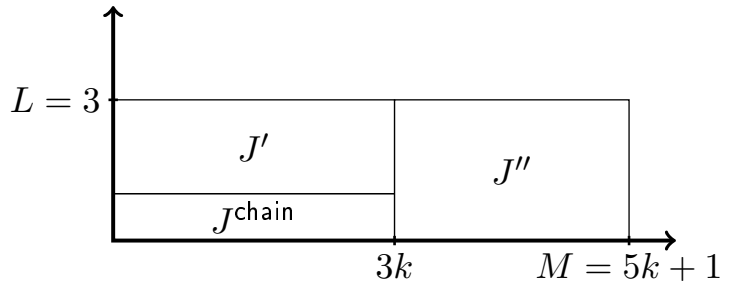
The list algorithm has approximation ratio ***at most*** $\frac{3}{5}$ *for* $L3|prec, C_{max} \leq M|F$



Instance



List algorithm schedule



Optimal schedule

Approximation of $L3|prec, C_{max} \leq M|F$

Lemma

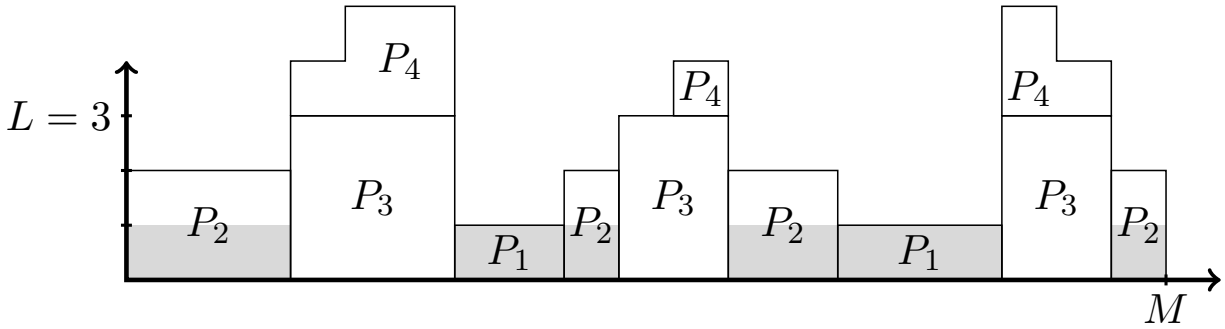
The list algorithm has approximation ratio ***at least*** $\frac{3}{5}$ *for* $L2|prec, C_{max} \leq M|F$

Approximation of $L3|prec, C_{max} \leq M|F$

Lemma

The list algorithm has approximation ratio ***at least*** $\frac{3}{5}$ *for* $L2|prec, C_{max} \leq M|F$

Proof also based on volume of groups of tasks (but more involved!)



Conclusion

		$C_{max} \leq M$	$prec, C_{max} \leq M$		r_i, d_i	
			$\emptyset$	*in-tree*	no *pmtn*	*pmtn*
$L = 1$	$\emptyset$	in P	in P		strongly NP-hard	in P
$L = 2$	$\emptyset$	pseudo polynomial	strongly NP-hard		strongly NP-hard	in P
	$p_i = 1$	in P	in P		in P	
$L \in \mathbb{N}$	$\emptyset$	strongly NP-hard				in P
	$p_i = 1$	in P	strongly NP-hard	in P	in P	

Complexity of resource leveling problems [2]

	$C_{max} \leq M$	$C_{max} \leq M, prec$	
		$\emptyset$	*in-tree*
$L = 2$	str. NP-hard	str. NP-hard $\frac{2}{3}$ ratio	
$L = 3$	EPTAS $\frac{7}{8}$ ratio in $O(\lvert J\rvert \log(\lvert J\rvert))$	str. NP-hard $\frac{3}{5}$ ratio	
$L \in \mathbb{N}$		str. NP-hard No PTAS[3]	str. NP-hard $\frac{1}{2}$ ratio

Approximation of resource leveling problems[4]

Conclusion

		$C_{max} \leq M$	$prec, C_{max} \leq M$		r_i, d_i	
			$\emptyset$	*in-tree*	no *pmtn*	*pmtn*
$L = 1$	$\emptyset$	in P	in P		strongly NP-hard	in P
$L = 2$	$\emptyset$	pseudo polynomial	strongly NP-hard		strongly NP-hard	in P
	$p_i = 1$	in P	in P		in P	
$L \in \mathbb{N}$	$\emptyset$	strongly NP-hard				in P
	$p_i = 1$	in P	strongly NP-hard	in P	in P	

Complexity of resource leveling problems [2]

	$C_{max} \leq M$	$C_{max} \leq M, prec$	
		$\emptyset$	*in-tree*
$L = 2$	str. NP-hard	str. NP-hard $\frac{2}{3}$ ratio	
$L = 3$	EPTAS $\frac{7}{8}$ ratio in $O(\lvert J\rvert \log(\lvert J\rvert))$	str. NP-hard $\frac{3}{5}$ ratio	
$L \in \mathbb{N}$		str. NP-hard No PTAS[3]	str. NP-hard $\frac{1}{2}$ ratio

Approximation of resource leveling problems[4]

Some open questions

- Complexity of $L3|prec, p_i = 1, C_{max} \leq M|F$
- Is $L|prec, C_{max} \leq M|F$ approximable within constant ratio?
- Improved ratios (PTAS?) for $L = 2, 3, \ldots$?

Conclusion

Resource Leveling?

$$L|prec, C_{max} \leq M|F$$

Conclusion

Resource Leveling?

$$L|prec, C_{max} \leq M|F$$

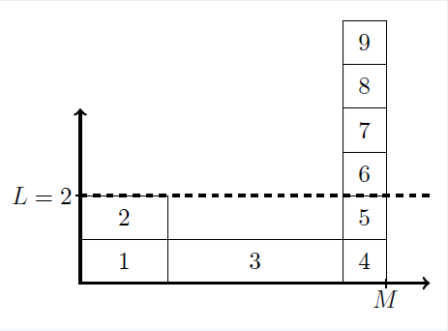
- Function F only penalizes the amount of resource overload

Conclusion

Resource Leveling?

$$L|prec, C_{max} \leq M|F$$

- Function F only penalizes the amount of resource overload
- Nothing prevents variations or high peaks in resource use

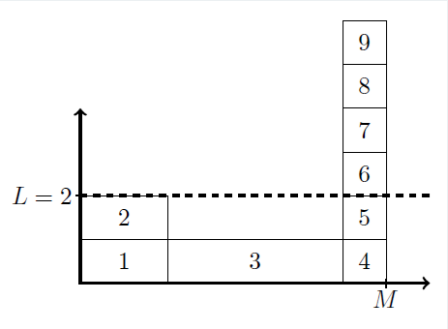


Conclusion

Resource Leveling?

$$L|prec, C_{max} \leq M|?$$

- Function F only penalizes the amount of resource overload
- Nothing prevents variations or high peaks in resource use



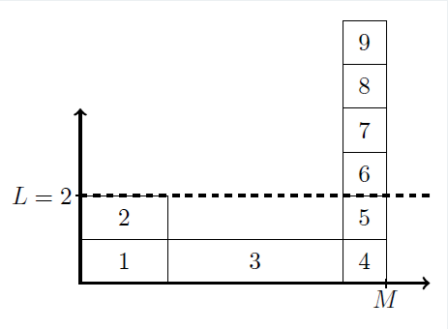
- Other objective functions penalize such behaviors (e.g., resource investment cost, squared changes in resource request)

Conclusion

Resource Leveling?

$$L|prec, C_{max} \leq M|?$$

- Function F only penalizes the amount of resource overload
- Nothing prevents variations or high peaks in resource use



- Other objective functions penalize such behaviors (e.g., resource investment cost, squared changes in resource request)
- Bi-objective problems could be considered to combine resource leveling with another objective (e.g., maximum resource use)


Thank you

$L|C_{max} \leq M|F$: Efficient PTAS

Configuration LP

- Configuration: description per machine/resource unit, with job *types*
  $\rightarrow$ *Config c_1: 3 jobs of proc. time 2 and 4 jobs of proc. time 4.*
  $\rightarrow$ Set $\mathcal{C} = \{c_1, \ldots, c_p\}$ of possible config. per resource unit.

$L|C_{max} \leq M|F$: Efficient PTAS

Configuration LP

- Configuration: description per machine/resource unit, with job *types*
  $\rightarrow$ *Config c_1: 3 jobs of proc. time 2 and 4 jobs of proc. time 4.*
  $\rightarrow$ Set $\mathcal{C} = \{c_1, \ldots, c_p\}$ of possible config. per resource unit.
- Solution $= x_i, i = 1 \ldots p$: number of config c_i used.
  $\rightarrow \sum x_i = L$
- ILP: (integer) variables x_i

$L|C_{max} \leq M|F$: Efficient PTAS

Configuration LP

- Configuration: description per machine/resource unit, with job *types*
  $\rightarrow$ *Config c_1: 3 jobs of proc. time 2 and 4 jobs of proc. time 4.*
  $\rightarrow$ Set $\mathcal{C} = \{c_1, \ldots, c_p\}$ of possible config. per resource unit.
- Solution $= x_i, i = 1 \ldots p$: number of config c_i used.
  $\rightarrow \sum x_i = L$
- ILP: (integer) variables x_i

To work we need to bound p:

- Bound the number of different processing times
- Bound the number of jobs per machine

$L|C_{max} \leq M|F$: Efficient PTAS

Configuration LP

- Configuration: description per machine/resource unit, with job *types*
  $\rightarrow$ *Config* c_1*: 3 jobs of proc. time 2 and 4 jobs of proc. time 4.*
  $\rightarrow$ Set $\mathcal{C} = \{c_1, \ldots, c_p\}$ of possible config. per resource unit.
- Solution $= x_i, i = 1 \ldots p$: number of config c_i used.
  $\rightarrow \sum x_i = L$
- ILP: (integer) variables x_i

To work we need to bound p:

- Bound the number of different processing times
- Bound the number of jobs per machine

as a function of ϵ for $(1 - \epsilon)$-approximation algorithm.

$L|C_{max} \leq M|F$: Efficient PTAS

To work we need to bound p:

- Bound the number of different processing times
- Bound the number of jobs per machine

as a function of ϵ for $(1-\epsilon)$-approximation algorithm.

$L|C_{max} \leq M|F$: Efficient PTAS

To work we need to bound p:

- Bound the number of different processing times
- Bound the number of jobs per machine

as a function of ϵ for $(1-\epsilon)$-approximation algorithm.

For $L|C_{max} \leq M|F$:

- Reformulate the problem $\rightarrow$ "resource unit view"

$L|C_{max} \leq M|F$: Efficient PTAS

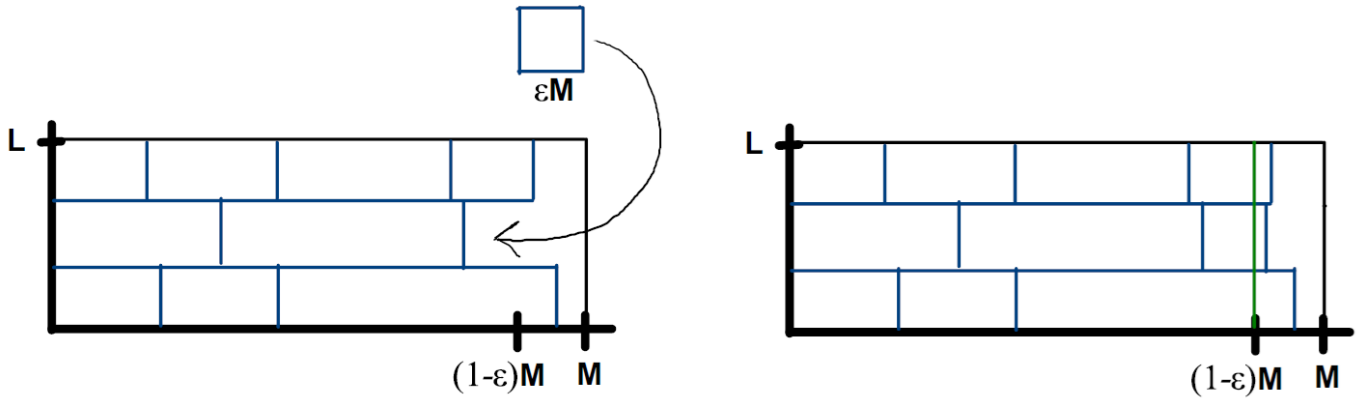
To work we need to bound p:

- Bound the number of different processing times
- Bound the number of jobs per machine

as a function of ϵ for $(1-\epsilon)$-approximation algorithm.

For $L|C_{max} \leq M|F$:

- Reformulate the problem $\rightarrow$ "resource unit view"
- Small jobs, $p_i \leq \epsilon M$: can be removed and dealt with later
  - Either there is a spot of $\epsilon M \rightarrow$ put the job.
  - Or ... $F(S) \geq (1-\epsilon)ML \geq (1-\epsilon)OPT$.



$L|C_{max} \leq M|F$: Efficient PTAS

To work we need to bound p:

- Bound the number of different processing times
- Bound the number of jobs per machine

as a function of ϵ for $(1-\epsilon)$-approximation algorithm.

For $L|C_{max} \leq M|F$:

- Reformulate the problem $\rightarrow$ "resource unit view"
- Small jobs, $p_i \leq \epsilon M$: can be removed and dealt with later
  - Either there is a spot of $\epsilon M \rightarrow$ put the job.
  - Or ... $F(S) \geq (1-\epsilon)ML \geq (1-\epsilon)OPT$.
- Now $p_i \geq \epsilon M \rightarrow$ so at most $1/\epsilon$ jobs per resource unit.

$L|C_{max} \leq M|F$: Efficient PTAS

To work we need to bound p:

- Bound the number of different processing times
- Bound the number of jobs per machine

as a function of ϵ for $(1-\epsilon)$-approximation algorithm.

For $L|C_{max} \leq M|F$:

- Reformulate the problem $\rightarrow$ "resource unit view"
- Small jobs, $p_i \leq \epsilon M$: can be removed and dealt with later
  - Either there is a spot of $\epsilon M \rightarrow$ put the job.
  - Or ... $F(S) \geq (1-\epsilon)ML \geq (1-\epsilon)OPT$.
- Now $p_i \geq \epsilon M \rightarrow$ so at most $1/\epsilon$ jobs per resource unit.
- Round the p_i:
  $(1-\epsilon)^k M \leq p_i < (1-\epsilon)^{k-1} M \rightarrow \tilde{p}_i = (1-\epsilon)^k M$.

$L|C_{max} \leq M|F$: Efficient PTAS

To work we need to bound p:

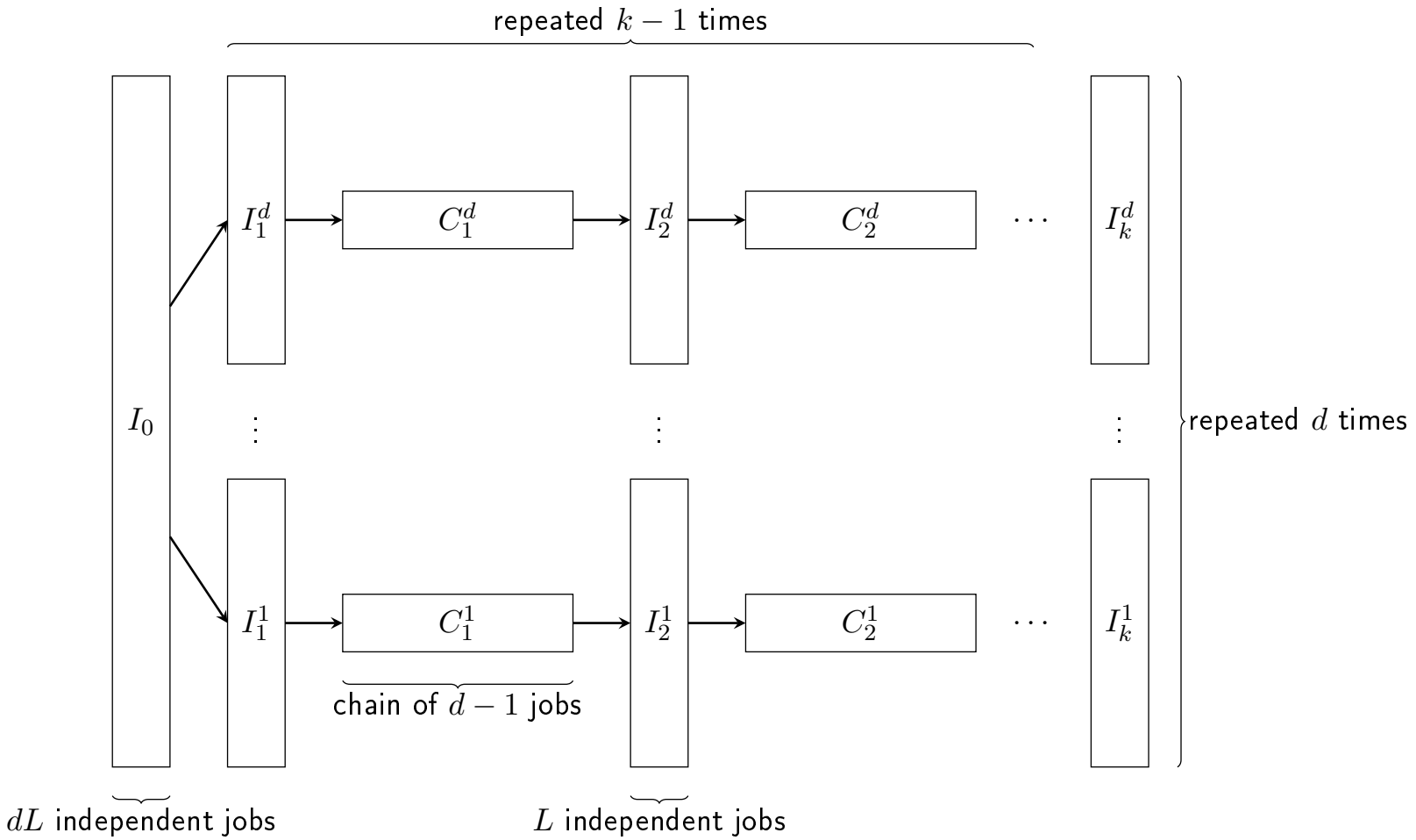
- Bound the number of different processing times
- Bound the number of jobs per machine

as a function of ϵ for $(1-\epsilon)$-approximation algorithm.

For $L|C_{max} \leq M|F$:

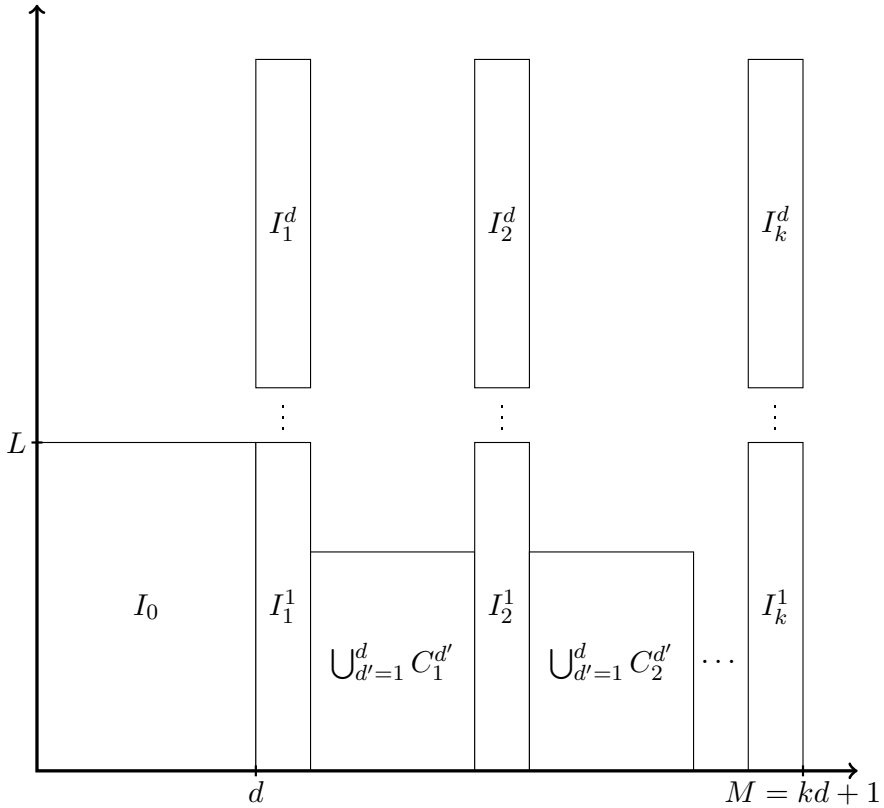
- Reformulate the problem $\rightarrow$ "resource unit view"
- Small jobs, $p_i \leq \epsilon M$: can be removed and dealt with later
  - Either there is a spot of $\epsilon M \rightarrow$ put the job.
  - Or ... $F(S) \geq (1-\epsilon)ML \geq (1-\epsilon)OPT$.
- Now $p_i \geq \epsilon M \rightarrow$ so at most $1/\epsilon$ jobs per resource unit.
- Round the p_i:
  $(1-\epsilon)^k M \leq p_i < (1-\epsilon)^{k-1} M \rightarrow \tilde{p}_i = (1-\epsilon)^k M$.
  $\rightarrow$ factor $(1-\epsilon)$ in the rouding
  $\rightarrow \sim \log_{1-\epsilon}(\epsilon)$ different processing times.

Bad list algorithm approximation ratio for large L

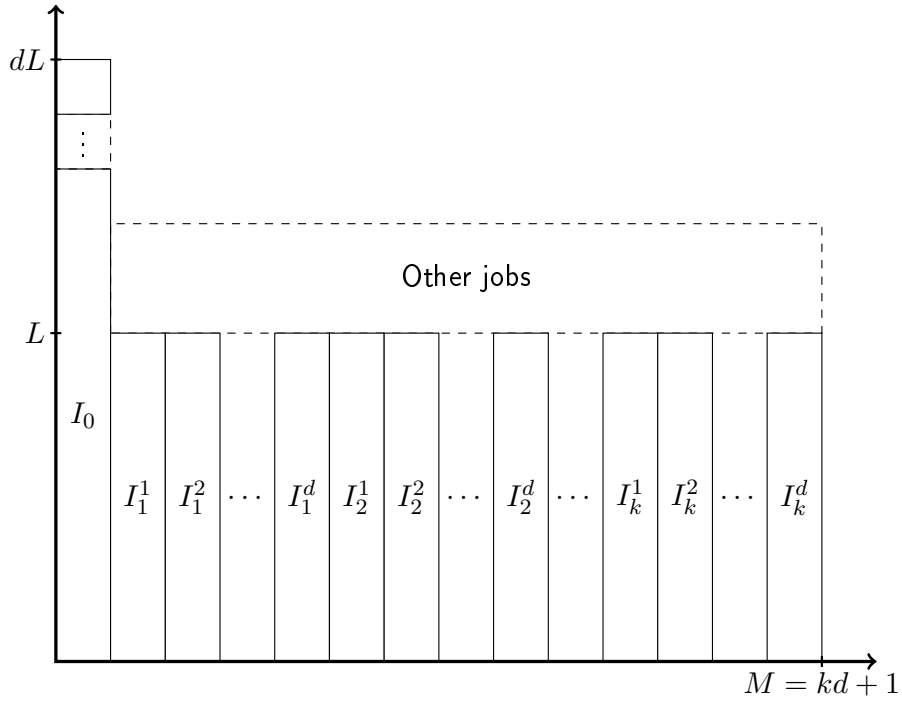


Instance leading to arbitrarily small approximation ratio

Bad list algorithm approximation ratio for large L



List algorithm schedule



Optimal schedule

Resource leveling (main) criteria

u_t = resource consumption at time step t

- Total overload cost (=F)
- Resource Investment cost: Minimize $\max_t u_t$
- Square resource used: Min $\sum_t u_t^2$ (or its variant $\sum_t (u_t - L, 0)^2$.
- Absolute variations: $\sum_t |u_{t+1} - u_t|$
- Squared absolute variations: $\sum_t (u_{t+1} - u - t)^2$

Usually considered for problems with multiple resources (RCPSP)

Some related works

- P. Györgyi, T. Kis, A common approximation framework for early work, late work, and resource leveling problems, EJOR (2020). $\rightarrow$ A PTAS for $L|C_{max} \leq M|F$.
- S. Hartmann, D. Briskorn. An updated survey of variants and extensions of the resource-constrained project scheduling problem, EJOR (2022). $\rightarrow$ Review on RCPSP, including resource leveling.