Dominances en programmation linéaire :
ordonnancement autour d'une date d'échéance commune

soutenance de thèse d'**Anne-Elisabeth FALQ**

encadrée par Pierre Fouilhoux et Safia Kedad-Sidhoum

2 Novembre 2020, LIP6, Paris

slides et tapuscrit disponibles sur
http://perso.eleves.ens-rennes.fr/~afalq494/recherche-these.html

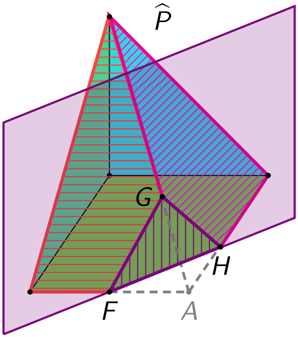
Comment est née cette thèse? (only slide in french)

2016/2017 cours de M2 à l'UPMC, donné par Pierre et Safia

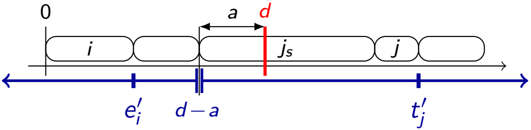
Comment est née cette thèse? (only slide in french)

2016/2017 cours de M2 à l'UPMC, donné par Pierre et Safia

PLNE et polyèdres



Ordonnancement juste-à-temps

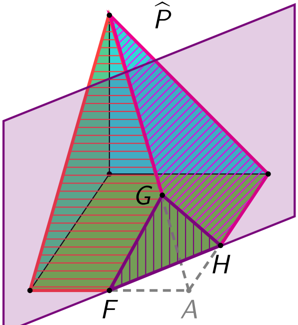


Comment est née cette thèse? (only slide in french)

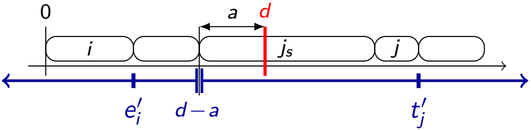
2016/2017 cours de M2 à l'UPMC, donné par Pierre et Safia

été 2017 stage au LIP6, équipe RO, encadré par Pierre et Safia

PLNE et polyèdres



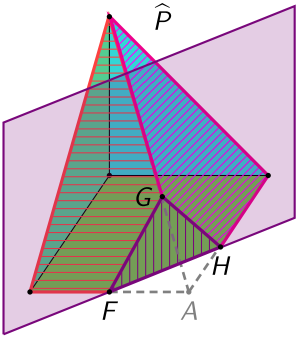
Ordonnancement juste-à-temps



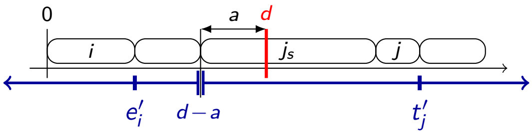
Comment est née cette thèse? (only slide in french)

2016/2017 cours de M2 à l'UPMC, donné par Pierre et Safia
été 2017 stage au LIP6, équipe RO, encadré par Pierre et Safia
depuis thèse au LIP6, équipe RO, encadrée par Pierre et Safia

PLNE et polyèdres



Ordonnancement juste-à-temps

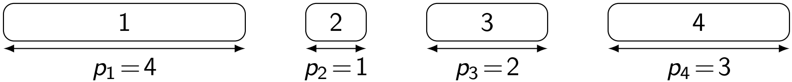


Outline

1. Introduction
   - Scheduling around a common due date
   - Known results about UCDDP and CDDP
   - How to encode schedules?
2. Focus 1: A formulation for UCDDP using natural variables
3. Interlude : a first attempt to eliminate dominated solutions
4. Focus 2: dominance inequality
5. Conclusion and perspectives

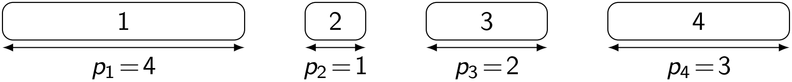
Scheduling around a common due-date on a single machine

An instance = • a set of tasks : $J = \{1, 2, 3, 4\}$

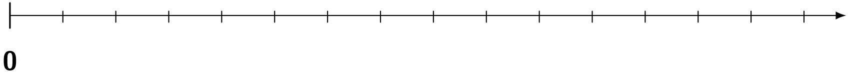


Scheduling around a common due-date on a single machine

An instance = • a set of tasks : $J = \{1, 2, 3, 4\}$

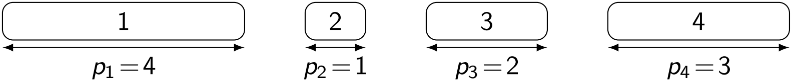


A schedule :

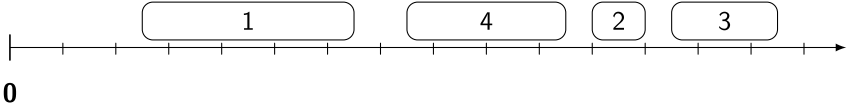


Scheduling around a common due-date on a single machine

An instance = • a set of tasks : $J = \{1, 2, 3, 4\}$

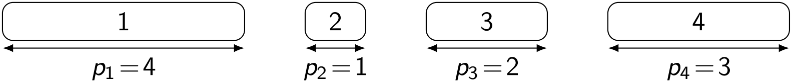


A schedule :



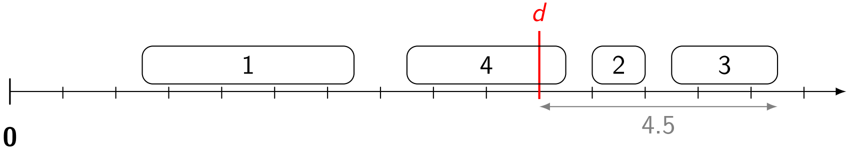
Scheduling around a common due-date on a single machine

An instance = • a set of tasks : $J = \{1, 2, 3, 4\}$



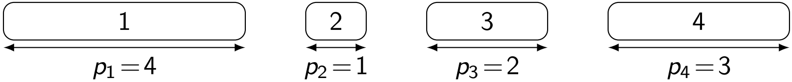
• common due date : $d = 10$

A schedule :



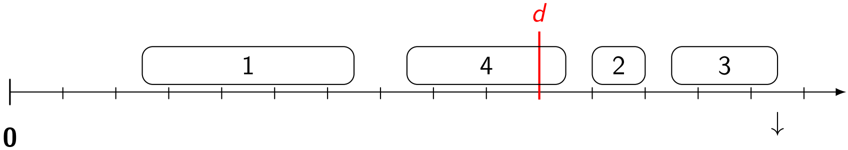
Scheduling around a common due-date on a single machine

An instance = • a set of tasks : $J = \{1, 2, 3, 4\}$



- common due date : $d = 10$
- unit tardiness penalties : $\beta_1 = \beta_4 = 5, \quad \beta_2 = \beta_3 = 2$

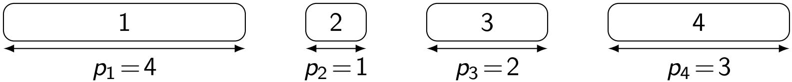
A schedule :



$4,5 \times 2 = 9$

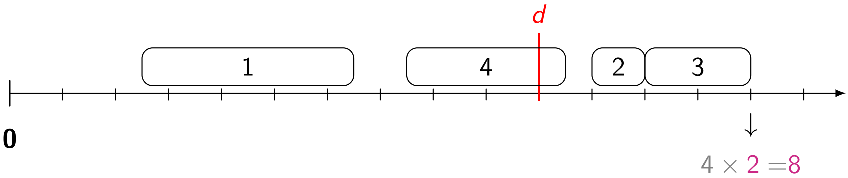
Scheduling around a common due-date on a single machine

An instance = • a set of tasks : $J = \{1, 2, 3, 4\}$



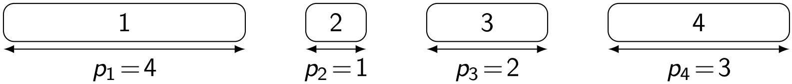
- common due date : $d = 10$
- unit tardiness penalties : $\beta_1 = \beta_4 = 5$, $\beta_2 = \beta_3 = 2$

A schedule :



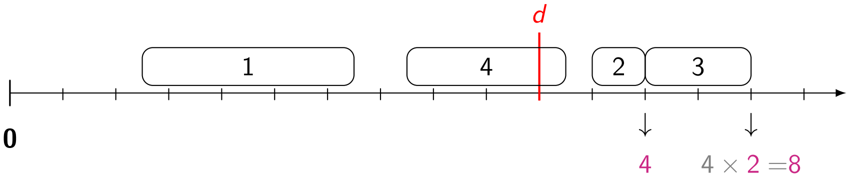
Scheduling around a common due-date on a single machine

An instance = • a set of tasks : $J = \{1, 2, 3, 4\}$



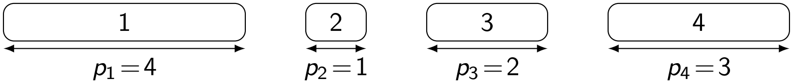
- common due date : $d = 10$
- unit tardiness penalties : $\beta_1 = \beta_4 = 5$, $\beta_2 = \beta_3 = 2$

A schedule :



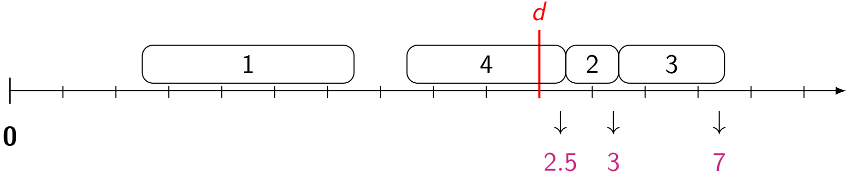
Scheduling around a common due-date on a single machine

An instance = • a set of tasks : $J = \{1, 2, 3, 4\}$



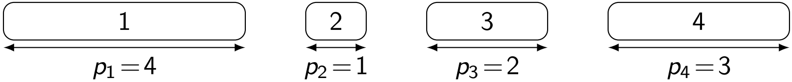
- common due date : $d = 10$
- unit tardiness penalties : $\beta_1 = \beta_4 = 5$, $\beta_2 = \beta_3 = 2$

A schedule :



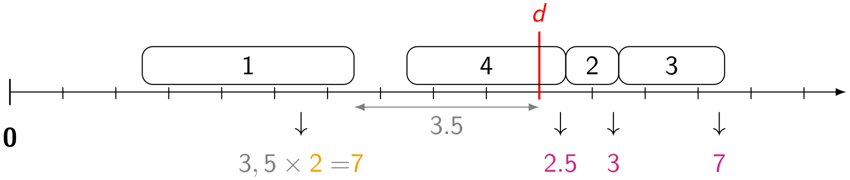
Scheduling around a common due-date on a single machine

An instance = • a set of tasks : $J = \{1, 2, 3, 4\}$



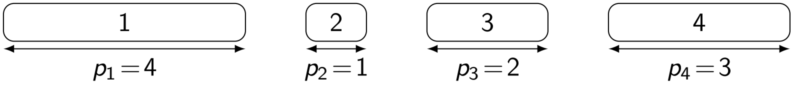
- common due date : $d = 10$
- unit tardiness penalties : $\beta_1 = \beta_4 = 5$, $\beta_2 = \beta_3 = 2$
- unit earliness penalties : $\forall j \in J$, $\alpha_j = 2$

A schedule :



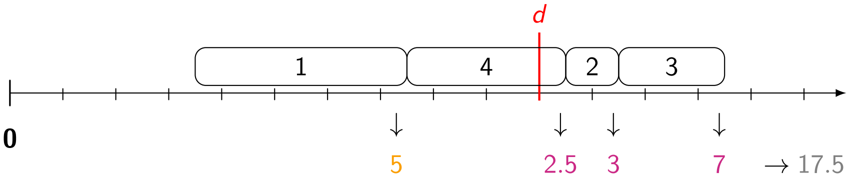
Scheduling around a common due-date on a single machine

An instance = • a set of tasks : $J = \{1, 2, 3, 4\}$



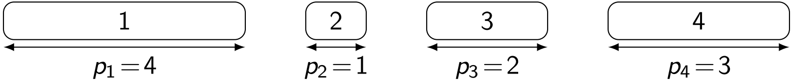
- common due date : $d = 10$
- unit tardiness penalties : $\beta_1 = \beta_4 = 5, \quad \beta_2 = \beta_3 = 2$
- unit earliness penalties : $\forall j \in J,\ \alpha_j = 2$

A schedule :



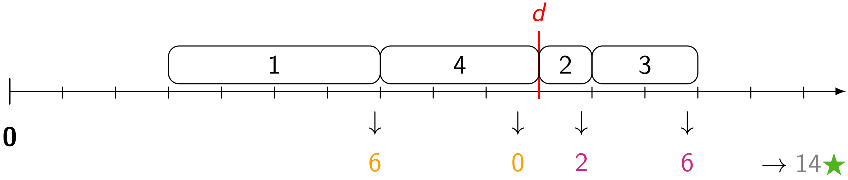
Scheduling around a common due-date on a single machine

An instance = • a set of tasks : $J = \{1, 2, 3, 4\}$



- common due date : $d = 10$
- unit tardiness penalties : $\beta_1 = \beta_4 = 5, \quad \beta_2 = \beta_3 = 2$
- unit earliness penalties : $\forall j \in J,\ \alpha_j = 2$

A schedule :



The Unrestrictive Common Due Date Problem (UCDDP)

An instance =

- a set of tasks J

The Unrestrictive Common Due Date Problem (UCDDP)

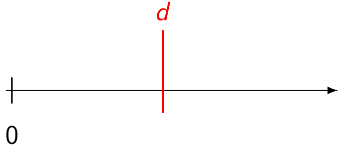
An instance =

- a set of tasks J
- their processing times $(p_j)_{j \in J} \in \mathbb{N}^J$

The Unrestrictive Common Due Date Problem (UCDDP)

An instance =

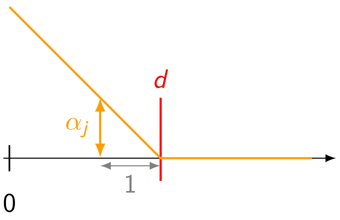
- a set of tasks J
- their processing times $(p_j)_{j \in J} \in \mathbb{N}^J$
- an **unrestrictive** common due-date $d \geqslant \sum_{j \in J} p_j$



The Unrestrictive Common Due Date Problem (UCDDP)

An instance =

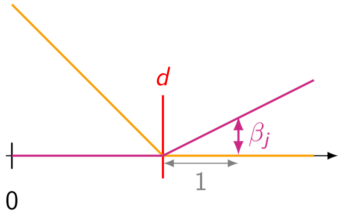
- a set of tasks J
- their processing times $(p_j)_{j \in J} \in \mathbb{N}^J$
- an **unrestrictive** common due-date $d \geqslant \sum_{j \in J} p_j$
- their unit earliness penalties $(\alpha_j)_{j \in J}$



The Unrestrictive Common Due Date Problem (UCDDP)

An instance =

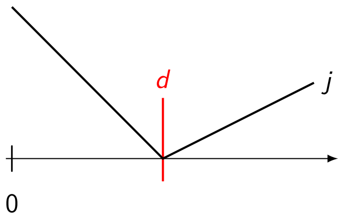
- a set of tasks J
- their processing times $(p_j)_{j \in J} \in \mathbb{N}^J$
- an **unrestrictive** common due-date $d \geqslant \sum_{j \in J} p_j$
- their unit earliness penalties $(\alpha_j)_{j \in J}$
- their unit tardiness penalties $(\beta_j)_{j \in J}$



The Unrestrictive Common Due Date Problem (UCDDP)

An instance =

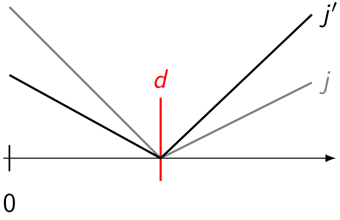
- a set of tasks J
- their processing times $(p_j)_{j\in J} \in \mathbb{N}^J$
- an **unrestrictive** common due-date $d \geqslant \sum_{j\in J} p_j$
- their unit earliness penalties $(\alpha_j)_{j\in J}$
- their unit tardiness penalties $(\beta_j)_{j\in J}$



The Unrestrictive Common Due Date Problem (UCDDP)

An instance =

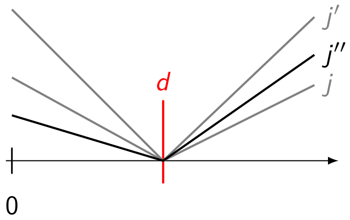
- a set of tasks J
- their processing times $(p_j)_{j\in J} \in \mathbb{N}^J$
- an **unrestrictive** common due-date $d \geqslant \sum_{j\in J} p_j$
- their unit earliness penalties $(\alpha_j)_{j\in J}$
- their unit tardiness penalties $(\beta_j)_{j\in J}$



The Unrestrictive Common Due Date Problem (UCDDP)

An instance =

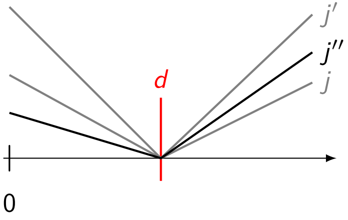
- a set of tasks J
- their processing times $(p_j)_{j\in J} \in \mathbb{N}^J$
- an **unrestrictive** common due-date $d \geqslant \sum_{j\in J} p_j$
- their unit earliness penalties $(\alpha_j)_{j\in J}$
- their unit tardiness penalties $(\beta_j)_{j\in J}$



The Unrestrictive Common Due Date Problem (UCDDP)

An instance =

- a set of tasks J
- their processing times $(p_j)_{j \in J} \in \mathbb{N}^J$
- an **unrestrictive** common due-date $d \geqslant \sum_{j \in J} p_j$
- their unit earliness penalties $(\alpha_j)_{j \in J}$
- their unit tardiness penalties $(\beta_j)_{j \in J}$

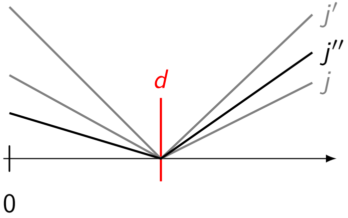


A solution schedule = a family of pairwise disjoint processing intervals

The Unrestrictive Common Due Date Problem (UCDDP)

An instance =

- a set of tasks J
- their processing times $(p_j)_{j \in J} \in \mathbb{N}^J$
- an **unrestrictive** common due-date $d \geqslant \sum_{j \in J} p_j$
- their unit earliness penalties $(\alpha_j)_{j \in J}$
- their unit tardiness penalties $(\beta_j)_{j \in J}$



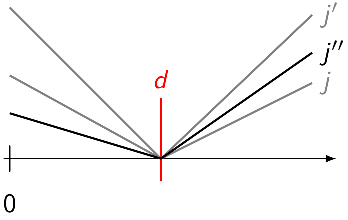
A solution schedule = a family of pairwise disjoint processing intervals

The objective = $\min \sum_{j \in J} \alpha_j E_j + \beta_j T_j$ = minimize the sum of earliness and tardiness penalties

The ~~Unrestrictive~~ Common Due Date Problem (CDDP)

An instance =

- a set of tasks J
- their processing times $(p_j)_{j \in J} \in \mathbb{N}^J$
- a ~~unrestrictive~~ common due-date d
- their unit earliness penalties $(\alpha_j)_{j \in J}$
- their unit tardiness penalties $(\beta_j)_{j \in J}$



A solution schedule = a family of pairwise disjoint processing intervals

The objective = $\min \sum_{j \in J} \alpha_j E_j + \beta_j T_j$ = minimize the sum of earliness and tardiness penalties

What are dominance properties?

Let T be a solution subset of an arbitrary optimization problem.

What are dominance properties?

Let T be a solution subset of an arbitrary optimization problem.

- T is a dominant set if it contains at least one optimal solution

What are dominance properties?

Let T be a solution subset of an arbitrary optimization problem.

- T is a dominant set if it contains at least one optimal solution
- T is a strictly dominant set if it contains all the optimal solutions

What are dominance properties?

Let T be a solution subset of an arbitrary optimization problem.

- T is a dominant set if it contains at least one optimal solution
- T is a strictly dominant set if it contains all the optimal solutions

In both cases,

$\rightarrow$ the searching space can be reduced to T

$\rightarrow$ other solutions can be discarded

Dominance properties and complexity

	unrestrictive case $d \geqslant \sum_{j \in J} p_j$
dominance properties	

Dominance properties and complexity

	unrestrictive case $d \geqslant \sum_{j \in J} p_j$
dominance properties	• without idle time □□□

Dominance properties and complexity

	unrestrictive case $d \geqslant \sum_{j \in J} p_j$
dominance properties	• without idle time • one on time task

Dominance properties and complexity

	unrestrictive case $d \geqslant \sum_{j \in J} p_j$
dominance properties	• without idle time • one on time task
complexity	$\forall j \in J,\ \alpha_j = \beta_j = \omega \rightarrow$ P

Kanet, 1981, Naval Research Logistics Quaterly

Dominance properties and complexity

	unrestrictive case $d \geqslant \sum_{j \in J} p_j$
dominance properties	• without idle time • one on time task
complexity	$\forall j \in J,\ \alpha_j = \beta_j = \omega \rightarrow$ P $\forall j \in J,\ \alpha_j = \beta_j \rightarrow$ NP-hard

Kanet, 1981, Naval Research Logistics Quaterly
Hall and Posner, 1991, Operations research

Dominance properties and complexity

	unrestrictive case $d \geqslant \sum_{j \in J} p_j$
dominance properties	• without idle time  • one on time task  (+ "V-shaped")
complexity	$\forall j \in J,\ \alpha_j = \beta_j = \omega \rightarrow$ P $\forall j \in J,\ \alpha_j = \beta_j \rightarrow$ weakly NP-hard

Kanet, 1981, Naval Research Logistics Quaterly
Hall and Posner, 1991, Operations research

Dominance properties and complexity

	unrestrictive case $d \geqslant \sum_{j \in J} p_j$	general case
dominance properties	• without idle time  • one on time task  (+ "V-shaped")	
complexity	$\forall j \in J,\ \alpha_j = \beta_j = \omega \rightarrow$ P $\forall j \in J,\ \alpha_j = \beta_j \rightarrow$ weakly NP-hard	

Kanet, 1981, Naval Research Logistics Quaterly
Hall and Posner, 1991, Operations research

Dominance properties and complexity

	unrestrictive case $d \geqslant \sum_{j \in J} p_j$	general case
dominance properties	• without idle time • one on time task (+ "V-shaped")	• without idle time • one on time task **or** beginning at 0 ~~V-shaped~~
complexity	$\forall j \in J,\ \alpha_j = \beta_j = \omega \rightarrow$ P $\forall j \in J,\ \alpha_j = \beta_j \rightarrow$ weakly NP-hard	

Kanet, 1981, Naval Research Logistics Quaterly
Hall and Posner, 1991, Operations research
Hoogeveen and van de Velde, 1991, European Journal of Operational research

Dominance properties and complexity

	unrestrictive case $d \geqslant \sum_{j\in J} p_j$	general case
dominance properties	• without idle time • one on time task (+ "V-shaped")	• without idle time • one on time task **or** beginning at 0 ~~V-shaped~~
complexity	$\forall j \in J,\ \alpha_j = \beta_j = \omega \rightarrow$ P $\forall j \in J,\ \alpha_j = \beta_j \rightarrow$ weakly NP-hard	$\forall j \in J,\ \alpha_j = \beta_j = \omega \rightarrow$ NP-hard

Kanet, 1981, Naval Research Logistics Quaterly
Hall and Posner, 1991, Operations research
Hoogeveen and van de Velde, 1991, European Journal of Operational research

Dominance properties and complexity

	unrestrictive case $d \geqslant \sum_{j \in J} p_j$	general case
dominance properties	• without idle time • one on time task (+ "V-shaped")	• without idle time • one on time task **or** beginning at 0 ~~V-shaped~~
complexity	$\forall j \in J,\ \alpha_j = \beta_j = \omega \rightarrow$ P $\forall j \in J,\ \alpha_j = \beta_j \rightarrow$ weakly NP-hard	$\forall j \in J,\ \alpha_j = \beta_j = \omega \rightarrow$ NP-hard $\forall j \in J,\ \alpha_j = \beta_j \rightarrow$ weakly NP-hard

Kanet, 1981, Naval Research Logistics Quaterly
Hall and Posner, 1991, Operations research
Hoogeveen and van de Velde, 1991, European Journal of Operational research

Dominance properties and complexity

	unrestrictive case $d \geqslant \sum_{j \in J} p_j$	general case
dominance properties	• without idle time • one on time task (+ "V-shaped")	• without idle time • one on time task **or** beginning at 0 ~~V-shaped~~
complexity	$\forall j \in J,\ \alpha_j = \beta_j = \omega \rightarrow$ P $\forall j \in J,\ \alpha_j = \beta_j \rightarrow$ weakly NP-hard	$\forall j \in J,\ \alpha_j = \beta_j = \omega \rightarrow$ NP-hard $\forall j \in J,\ \alpha_j = \beta_j \rightarrow$ weakly NP-hard arbitrary $\alpha_j, \beta_j \rightarrow$ Branch-and-Bound can solve up to 1000-task instances

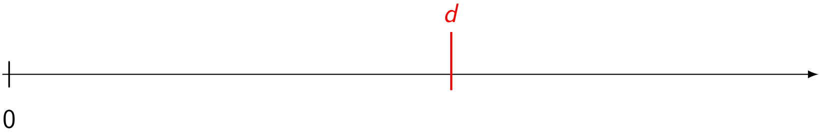
Kanet, 1981, Naval Research Logistics Quaterly
Hall and Posner, 1991, Operations research
Hoogeveen and van de Velde, 1991, European Journal of Operational research
Sourd, 2009, Informs Journal on Computing

Dominance properties and complexity

	unrestrictive case $d \geqslant \sum_j p_j$	general case
dominance properties	• without idle time • one on time task (+ "V-shaped")	• without idle time • one on time task **or** beginning at 0 ~~V-shaped~~
complexity	$\forall j \in J,\ \alpha_j = \beta_j = \omega \rightarrow$ P $\forall j \in J,\ \alpha_j = \beta_j \rightarrow$ weakly NP-hard arbitrary $\alpha_j, \beta_j \rightarrow$ NP-hard	$\forall j \in J,\ \alpha_j = \beta_j = \omega \rightarrow$ NP-hard $\forall j \in J,\ \alpha_j = \beta_j \rightarrow$ weakly NP-hard arbitrary $\alpha_j, \beta_j \rightarrow$ Branch-and-Bound can solve up to 1000-task instances

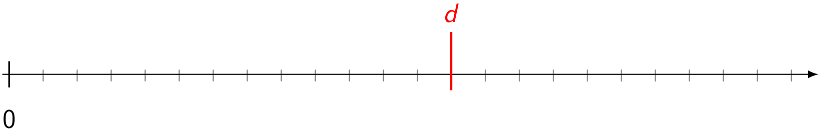
Kanet, 1981, Naval Research Logistics Quaterly
Hall and Posner, 1991, Operations research
Hoogeveen and van de Velde, 1991, European Journal of Operational research
Sourd, 2009, Informs Journal on Computing

Time-indexed variables



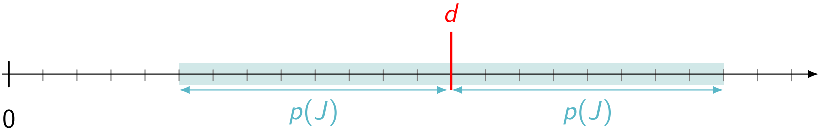
First let us discretize the time horizon .

Time-indexed variables



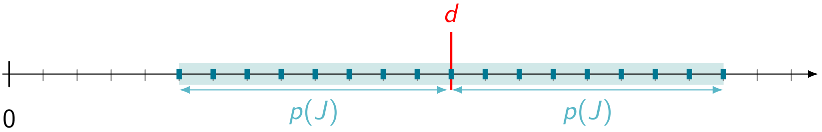
First let us discretize the time horizon .

Time-indexed variables



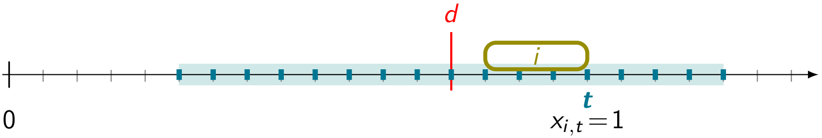
First let us discretize the time horizon .

Time-indexed variables



First let us discretize the time horizon as $\mathcal{T} = [d - p(J), d + p(J)]$.

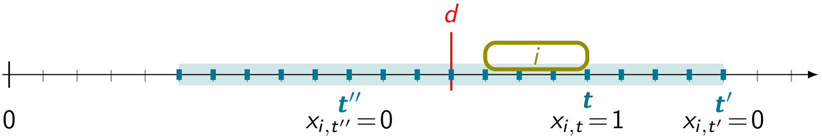
Time-indexed variables



First let us discretize the time horizon as $\mathcal{T} = [d - p(J), d + p(J)]$.

Variables: $\forall i \in J$, $\forall t \in \mathcal{T}$: $x_{i,t} = \begin{cases} 1 \text{ if } i \text{ completes at } t \\ 0 \text{ otherwise} \end{cases}$

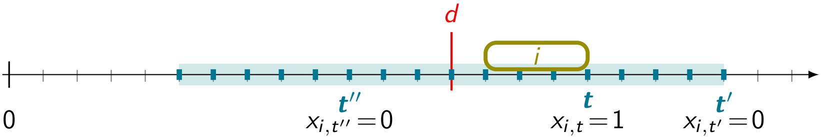
Time-indexed variables



First let us discretize the time horizon as $\mathcal{T}=[d-p(J), d+p(J)]$.

Variables: $\forall i \in J$, $\forall t \in \mathcal{T}$: $x_{i,t} = \begin{cases} 1 \text{ if } i \text{ completes at } t \\ 0 \text{ otherwise} \end{cases}$

Time-indexed variables

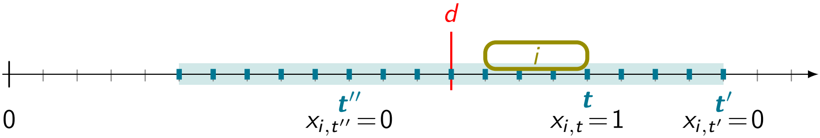


First let us discretize the time horizon as $\mathcal{T} = [d - p(J), d + p(J)]$.

Variables: $\forall i \in J, \forall t \in \mathcal{T}$: $x_{i,t} = \begin{cases} 1 \text{ if } i \text{ completes at } t \\ 0 \text{ otherwise} \end{cases}$

Objective function: $\sum_{j \in J} \sum_{t \in \mathcal{T}} c_{i,t}\, x_{i,t}$ where $c_{i,t}$ are pre-computed from the instance

Time-indexed variables



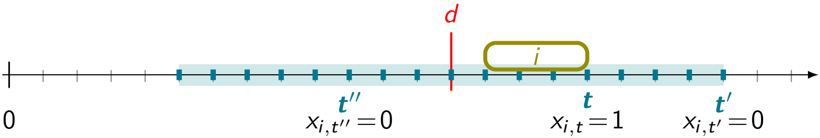
First let us discretize the time horizon as $\mathcal{T}=[d-p(J), d+p(J)]$.

Variables: $\forall i \in J, \forall t \in \mathcal{T}$: $x_{i,t} = \begin{cases} 1 \text{ if } i \text{ completes at } t \\ 0 \text{ otherwise} \end{cases}$

Objective function: $\sum_{j\in J}\sum_{t\in\mathcal{T}} c_{i,t}\, x_{i,t}$ where $c_{i,t}$ are pre-computed from the instance

Constraints:
- $\forall i \in J, \sum_{t\in\mathcal{T}} x_{i,t} = 1$ task i is placed
- $\forall t \in \mathcal{T}, \sum_{i\in J}\sum_{\substack{s\in\mathcal{T}\\ s\in[t,t+p_i[}} x_{i,s} \leqslant 1$ at most 1 task is in progress at t
- $\forall i \in J, \forall t \in \mathcal{T}, x_{i,t} \in \mathbb{Z}$ integrity constraint

Time-indexed variables



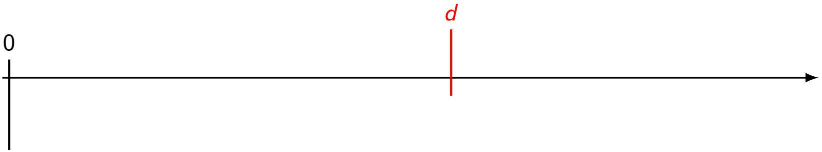
First let us discretize the time horizon as $\mathcal{T} = [d - p(J), d + p(J)]$.

Variables: $\forall i \in J,\ \forall t \in \mathcal{T}$: $x_{i,t} = \begin{cases} 1 \text{ if } i \text{ completes at } t \\ 0 \text{ otherwise} \end{cases}$

Objective function: $\sum_{j \in J} \sum_{t \in \mathcal{T}} c_{i,t}\, x_{i,t}$ where $c_{i,t}$ are pre-computed from the instance

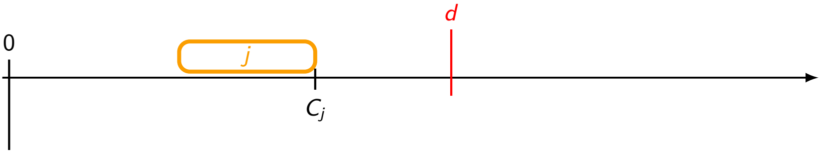
- \+ easy to formulate as a **MIP**
- \+ good relaxation value
- − $2n\,p(J)$ binary variables = a pseudo polynomial number
- − $n + n\,p(J)$ inequalities = a pseudo polynomial number

Completion time variables



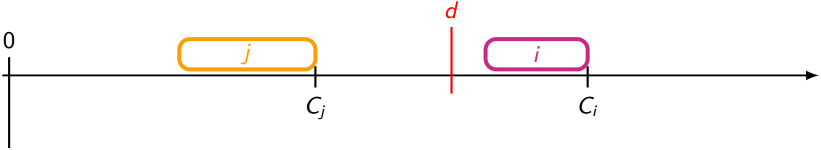
Variables: $\forall j \in J,\ C_j \in \mathbb{R}_+$ is the time when task j completes

Completion time variables



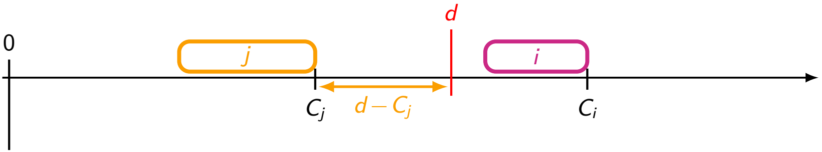
Variables: $\forall j \in J,\ C_j \in \mathbb{R}_+$ is the time when task j completes

Completion time variables



Variables: $\forall j \in J,\ C_j \in \mathbb{R}_+$ is the time when task j completes

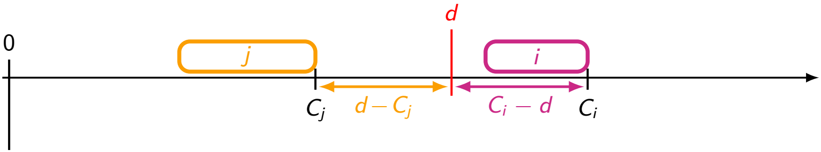
Completion time variables



Variables: $\forall j \in J,\ C_j \in \mathbb{R}_+$ is the time when task j completes

Objective function: $\sum_{j \in J} \alpha_j\,[d - C_j]^+ + \beta_j\,[C_j - d]^+$

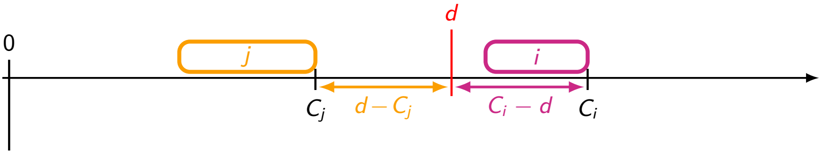
Completion time variables



Variables: $\forall j \in J,\ C_j \in \mathbb{R}_+$ is the time when task j completes

Objective function: $\sum_{j \in J} \alpha_j \, [d - C_j]^+ + \beta_j \, [C_j - d]^+$

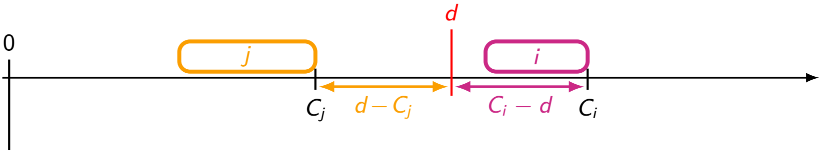
Completion time variables



Variables: $\forall j \in J,\ C_j \in \mathbb{R}_+$ is the time when task j completes

Objective function: $\sum_{j \in J} \alpha_j \, [d - C_j]^+ + \beta_j \, [C_j - d]^+$

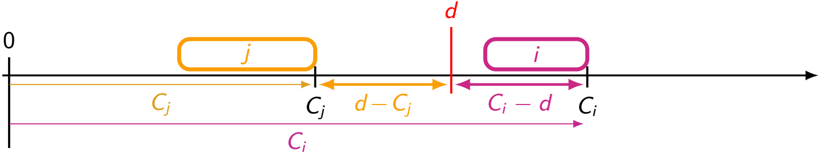
Completion time variables



Variables: $\forall j \in J,\ C_j \in \mathbb{R}_+$ is the time when task j completes

Objective function: $\sum_{j \in J} \alpha_j \, [d - C_j]^+ + \beta_j \, [C_j - d]^+$ not linear!

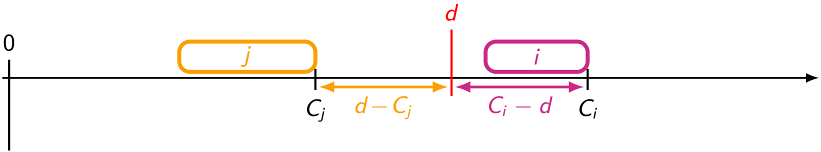
Completion time variables



Variables: $\forall j \in J,\ C_j \in \mathbb{R}_+$ is the time when task j completes

Objective function: $\sum_{j \in J} \alpha_j \, [d - C_j]^+ + \beta_j \, [C_j - d]^+$ not linear!

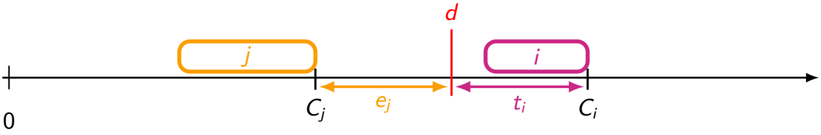
Completion time variables



Variables: $\forall j \in J,\ C_j \in \mathbb{R}_+$ is the time when task j completes

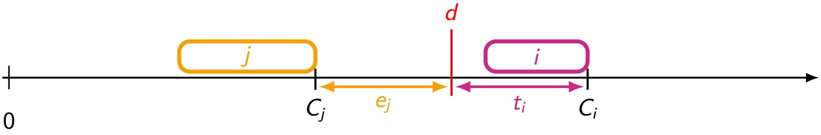
Objective function: $\sum_{j \in J} \alpha_j\,[d - C_j]^+ + \beta_j\,[C_j - d]^+$ not linear!

Earliness–Tardiness variables



Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

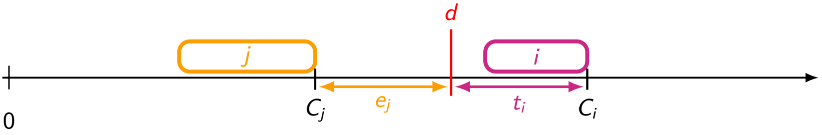
Earliness–Tardiness variables



Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

Objective function: $\sum_{j \in J} \alpha_j\, e_j + \beta_j\, t_j$

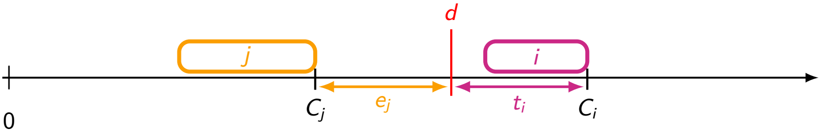
Earliness–Tardiness variables



Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

Objective function: $\sum_{j \in J} \alpha_j\, e_j + \beta_j\, t_j \leftarrow$ *linear function of variables e and t*

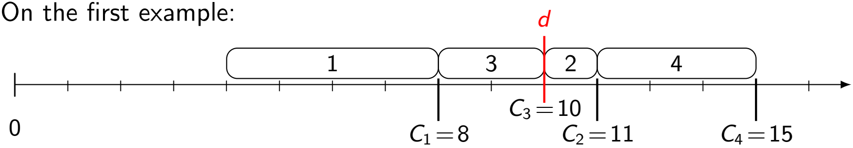
Earliness–Tardiness variables



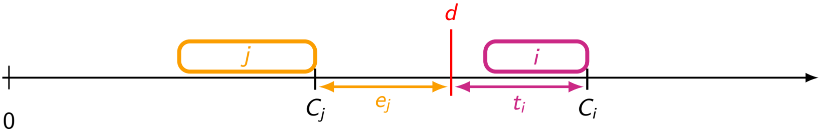
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

Objective function: $\sum_{j \in J} \alpha_j\, e_j + \beta_j\, t_j \leftarrow$ *linear function of variables e and t*

On the first example:



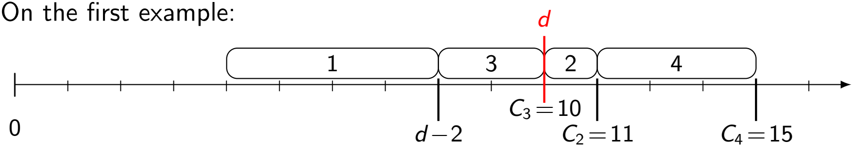
Earliness–Tardiness variables



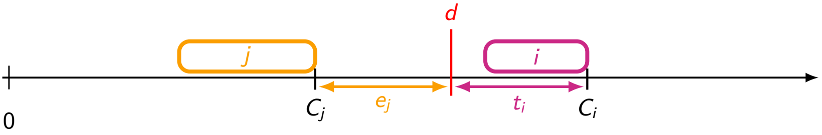
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

Objective function: $\sum_{j \in J} \alpha_j\, e_j + \beta_j\, t_j$ ← *linear function of variables* e *and* t

On the first example:



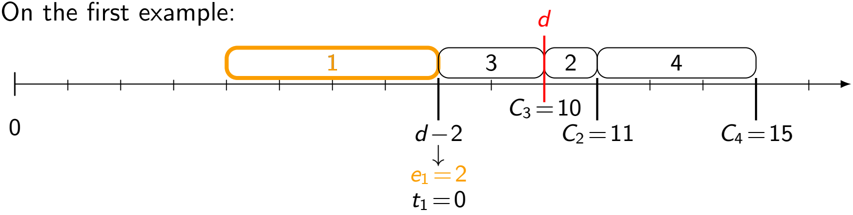
Earliness–Tardiness variables



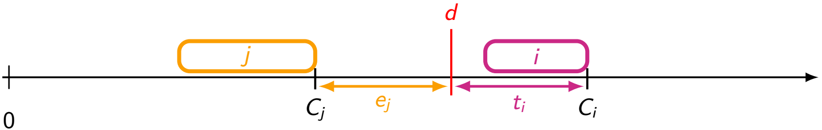
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

Objective function: $\sum_{j \in J} \alpha_j\, e_j + \beta_j\, t_j$ ← *linear function of variables* e *and* t

On the first example:



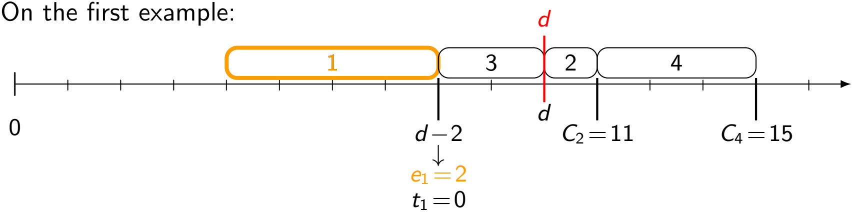
Earliness–Tardiness variables



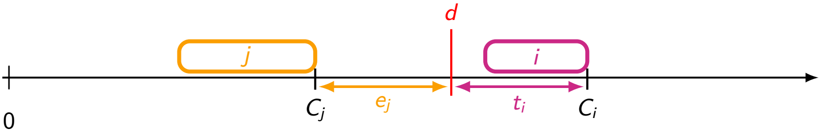
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

Objective function: $\sum_{j \in J} \alpha_j\, e_j + \beta_j\, t_j \leftarrow$ *linear function of variables e and t*

On the first example:



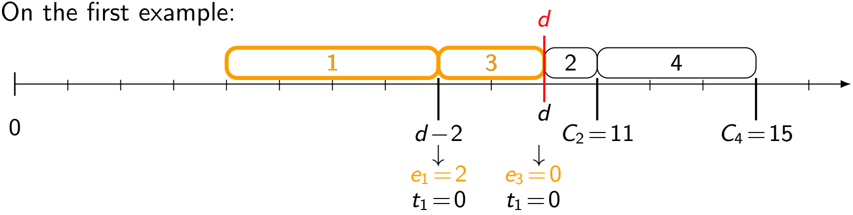
Earliness–Tardiness variables



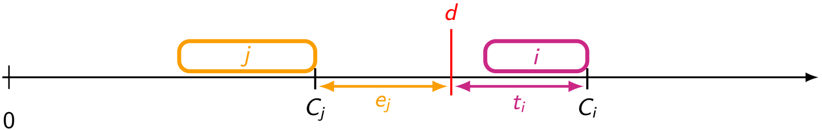
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

Objective function: $\sum_{j \in J} \alpha_j\, e_j + \beta_j\, t_j$ ← *linear function of variables* e *and* t

On the first example:



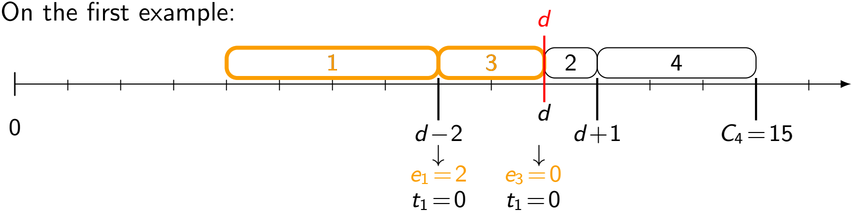
Earliness–Tardiness variables



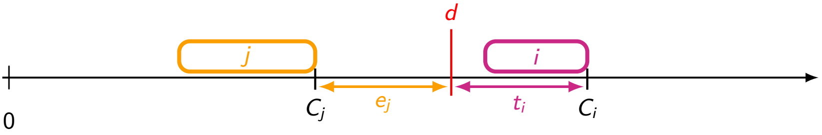
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

Objective function: $\sum_{j \in J} \alpha_j\, e_j + \beta_j\, t_j$ ← *linear function of variables e and t*

On the first example:



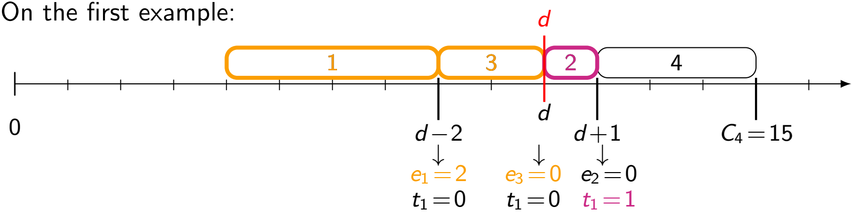
Earliness–Tardiness variables



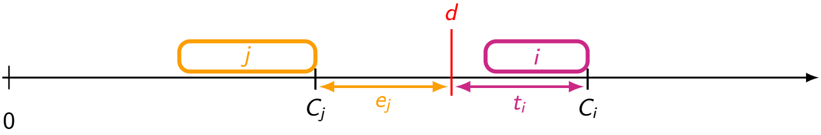
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

Objective function: $\sum_{j \in J} \alpha_j\, e_j + \beta_j\, t_j \leftarrow$ *linear function of variables e and t*

On the first example:



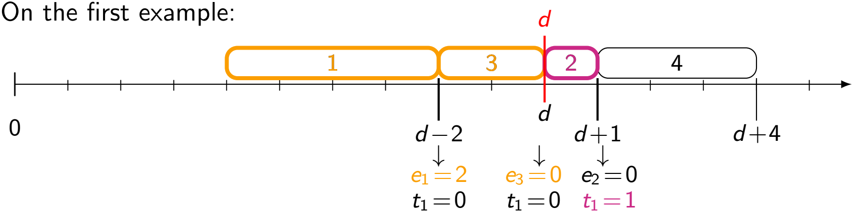
Earliness–Tardiness variables



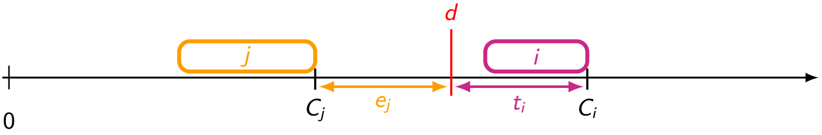
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

Objective function: $\sum_{j \in J} \alpha_j\, e_j + \beta_j\, t_j \leftarrow$ *linear function of variables* e *and* t

On the first example:



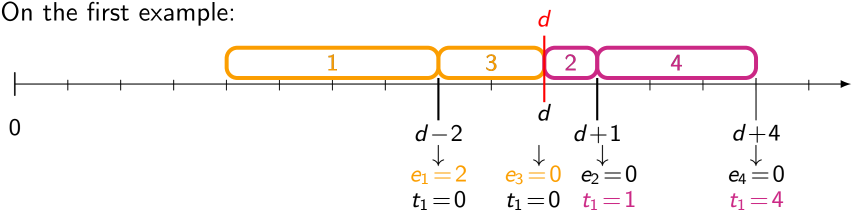
Earliness–Tardiness variables



Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

Objective function: $\sum_{j \in J} \alpha_j\, e_j + \beta_j\, t_j \leftarrow$ *linear function of variables e and t*

On the first example:



Thesis guideline and presentation outline

Questions of the thesis:

- How can we use linear programming to formulate scheduling problems for an exact solving?

Thesis guideline and presentation outline

Questions of the thesis:

- How can we use linear programming to formulate scheduling problems for an exact solving?
- In particular for the both problems UCDDP and CDDP?

Thesis guideline and presentation outline

Questions of the thesis:

- How can we use linear programming to formulate scheduling problems for an exact solving?
- In particular for the both problems UCDDP and CDDP?

Outline:

Focus 1: MIP formulations using natural variables for these problems
parts of Chapter 1 and 2

Thesis guideline and presentation outline

Questions of the thesis:

- How can we use linear programming to formulate scheduling problems for an exact solving?
- In particular for the both problems UCDDP and CDDP?

Outline:

Focus 1: MIP formulations using natural variables for these problems
parts of Chapter 1 and 2

Interlude: First attempt to reinforce a compact formulation for UCDDP?
parts of Chapter 3 and 4

Thesis guideline and presentation outline

Questions of the thesis:

- How can we use linear programming to formulate scheduling problems for an exact solving?
- In particular for the both problems UCDDP and CDDP?

Outline:

Focus 1: MIP formulations using natural variables for these problems
parts of Chapter 1 and 2

Interlude: First attempt to reinforce a compact formulation for UCDDP?
parts of Chapter 3 and 4

Focus 2: Dominance inequalities to reinforce such a formulation
part of Chapter 6

Outline

1. Introduction
2. Focus 1: A formulation for UCDDP using natural variables
   Describing the solution set for (e, t) variables
   How to extend this formulation
   How to manage this kind of formulations in practice
3. Interlude : a first attempt to eliminate dominated solutions
4. Focus 2: dominance inequality
5. Conclusion and perspectives

What is the goal?

We already have: $\text{UCDDP} \iff \min_{(e,t)\in\mathscr{S}} g_{\alpha,\beta}(e,t)$

where : $\rightarrow$ $g_{\alpha,\beta}$ is linear $\left(g_{\alpha,\beta} = (e,t) \mapsto \sum_{j\in J} \alpha_j e_j + \beta_j t_j\right)$

$\rightarrow$ $\mathscr{S}$ is the set of (e, t) vectors that describe a schedule

What is the goal?

We already have: $\text{UCDDP} \iff \min_{(e,t)\in\mathscr{S}} g_{\alpha,\beta}(e,t)$

where : $\rightarrow$ $g_{\alpha,\beta}$ is linear $\left(g_{\alpha,\beta} = (e,t) \mapsto \sum_{j\in J} \alpha_j e_j + \beta_j t_j\right)$

$\rightarrow$ $\mathscr{S}$ is the set of (e,t) vectors that describe a schedule

We want to describe $\mathscr{S}$ with:

- linear inequalities

in order to obtain: $\text{UCDDP} \iff \min_{(e,t)\in\mathscr{S}} g_{\alpha,\beta}(e,t)$

What is the goal?

We already have: $\text{UCDDP} \iff \min_{(e,t)\in\mathscr{S}} g_{\alpha,\beta}(e,t)$

where : $\rightarrow$ $g_{\alpha,\beta}$ is linear $\left(g_{\alpha,\beta} = (e,t) \mapsto \sum_{j\in J} \alpha_j e_j + \beta_j t_j\right)$

$\rightarrow$ $\mathscr{S}$ is the set of (e,t) vectors that describe a schedule

We want to describe $\mathscr{S}$ with:

- linear inequalities that define a polyhedron P

in order to obtain: $\text{UCDDP} \iff \min_{(e,t)\in P} g_{\alpha,\beta}(e,t)$

What is the goal?

We already have: $\text{UCDDP} \iff \min_{(e,t)\in\mathscr{S}} g_{\alpha,\beta}(e,t)$

where : $\rightarrow$ $g_{\alpha,\beta}$ is linear $\left(g_{\alpha,\beta} = (e,t) \mapsto \sum_{j\in J} \alpha_j e_j + \beta_j t_j\right)$

$\rightarrow$ $\mathscr{S}$ is the set of (e, t) vectors that describe a schedule

We want to describe $\mathscr{S}$ with:

- linear inequalities that define a polyhedron P
  if $\mathcal{P} \neq \mathcal{NP}$, it cannot be sufficient (LP-solving $\in \mathcal{P}$ and UCDDP $\in \mathcal{NP}$)

in order to obtain: $\text{UCDDP} \iff \min_{(e,t)\in P} g_{\alpha,\beta}(e,t)$

What is the goal?

We already have: $\text{UCDDP} \iff \min_{(e,t)\in\mathscr{S}} g_{\alpha,\beta}(e,t)$

where : $\rightarrow$ $g_{\alpha,\beta}$ is linear $\left(g_{\alpha,\beta} = (e,t) \mapsto \sum_{j\in J} \alpha_j e_j + \beta_j t_j\right)$

$\rightarrow$ $\mathscr{S}$ is the set of (e,t) vectors that describe a schedule

We want to describe $\mathscr{S}$ with:

- linear inequalities that define a polyhedron P
  if $\mathcal{P} \neq \mathcal{NP}$, it cannot be sufficient (LP-solving $\in \mathcal{P}$ and UCDDP $\in \mathcal{NP}$)
- integrity constraints

in order to obtain: $\text{UCDDP} \iff \min_{(e,t)\in\text{int}(P)} g_{\alpha,\beta}(e,t)$

What is the goal?

We already have: $\text{UCDDP} \iff \min_{(e,t)\in\mathscr{S}} g_{\alpha,\beta}(e,t)$

where : $\rightarrow$ $g_{\alpha,\beta}$ is linear $\left(g_{\alpha,\beta} = (e,t) \mapsto \sum_{j\in J} \alpha_j e_j + \beta_j t_j\right)$

$\rightarrow$ $\mathscr{S}$ is the set of (e,t) vectors that describe a schedule

We want to describe $\mathscr{S}$ with:

- linear inequalities that define a polyhedron P
  if $\mathcal{P} \neq \mathcal{NP}$, it cannot be sufficient (LP-solving $\in \mathcal{P}$ and UCDDP $\in \mathcal{NP}$)
- integrity constraints
- extremality constraints

in order to obtain: $\text{UCDDP} \iff \min_{(e,t)\in\text{int}(\text{extr } P)} g_{\alpha,\beta}(e,t)$

What do we need for describing the solution set?

An instance =

- a set of tasks J
- the processing times of these tasks $(p_j)_{j \in J}$
- an unrestrictive common due-date $d \geqslant \sum p_j$
- a unitary earliness penalty for each task $(\alpha_j)_{j \in J}$
- a unitary tardiness penalty for each task $(\beta_j)_{j \in J}$

What do we need for describing the solution set?

An instance =

- a set of tasks J
- the processing times of these tasks $(p_j)_{j \in J}$
- an unrestrictive common due-date $d \geqslant \sum p_j$
- a unitary earliness penalty for each task $(\alpha_j)_{j \in J}$
- a unitary tardiness penalty for each task $(\beta_j)_{j \in J}$

What do we need for describing the solution set?

An instance =

- a set of tasks J
- the processing times of these tasks $(p_j)_{j \in J}$
- an unrestrictive common due-date $d \geqslant \sum p_j$
- a unitary earliness penalty for each task $(\alpha_j)_{j \in J}$
- a unitary tardiness penalty for each task $(\beta_j)_{j \in J}$

How to describe the solution set?

To encode a feasible schedule, a vector (e, t) must satisfy :

[**consistancy**] e_j and t_j are not simultaneously strictly positive

[**non-overlapping**] processing intervals are pairwise disjoints

[**positivity**] processing intervals don't begin before time 0

How to describe the solution set?

To encode a feasible schedule, a vector (e, t) must satisfy :

[**consistancy**] e_j and t_j are not simultaneously strictly positive

[**non-overlapping**] processing intervals are pairwise disjoints

[**positivity**] processing intervals don't begin before time 0

How to describe the solution set?

To encode a feasible schedule, a vector (e, t) must satisfy :

[**consistancy**] e_j and t_j are not simultaneously strictly positive

[**non-overlapping**] processing intervals are pairwise disjoints

How to describe the solution set?

To encode a feasible schedule, a vector (e, t) must satisfy :

[**consistancy**] e_j and t_j are not simultaneously strictly positive

- Adding disjunctive variables $(\delta_j)_{j \in J}$ such that $\delta_j = \begin{cases} 1 \text{ if } j \text{ is early} \\ 0 \text{ if } j \text{ is tardy} \end{cases}$

$$\forall j \in J,\ e_j \geqslant 0 \quad (e.0) \qquad e_j \leqslant M\,\delta_j \quad (e.1)$$

$$\forall j \in J,\ t_j \geqslant 0 \quad (t.1) \qquad t_j \leqslant M\,(1-\delta_j) \quad (t.2)$$

where $\boldsymbol{M} = \sum_{j \in J} p_j$

[**non-overlapping**] processing intervals are pairwise disjoints

How to describe the solution set?

To encode a feasible schedule, a vector (e, t) must satisfy :

[**consistancy**] e_j and t_j are not simultaneously strictly positive

- Adding disjunctive variables $(\delta_j)_{j \in J}$ such that $\delta_j = \begin{cases} 1 \text{ if } j \text{ is early} \\ 0 \text{ if } j \text{ is tardy} \end{cases}$

$$\forall j \in J,\ e_j \geqslant 0 \quad (e.0) \qquad e_j \leqslant M\,\delta_j \quad (e.1)$$

$$\forall j \in J,\ t_j \geqslant 0 \quad (t.1) \qquad t_j \leqslant M\,(1-\delta_j) \quad (t.2)$$

where $\boldsymbol{M} = \sum_{j \in J} p_j$

[**non-overlapping**] processing intervals are pairwise disjoints

- decomposing non-overlapping for a schedule  :

the early tasks	the tardy tasks
• are entirely processed before d	• are entirely processsed after d

How to describe the solution set?

To encode a feasible schedule, a vector (e, t) must satisfy :

[**consistancy**] e_j and t_j are not simultaneously strictly positive

- Adding disjunctive variables $(\delta_j)_{j \in J}$ such that $\delta_j = \begin{cases} 1 \text{ if } j \text{ is early} \\ 0 \text{ if } j \text{ is tardy} \end{cases}$

$$\forall j \in J,\ e_j \geqslant 0 \quad (e.0) \qquad e_j \leqslant M\,\delta_j \quad (e.1)$$

$$\forall j \in J,\ t_j \geqslant 0 \quad (t.1) \qquad t_j \leqslant M\,(1-\delta_j) \quad (t.2)$$

where $\boldsymbol{M} = \sum_{j \in J} p_j$

[**non-overlapping**] processing intervals are pairwise disjoints

- decomposing non-overlapping for a schedule :

the early tasks		the tardy tasks
• are entirely processed before d • do not overlap each other	⟷	• are entirely processsed after d

How to describe the solution set?

To encode a feasible schedule, a vector (e, t) must satisfy :

[**consistancy**] e_j and t_j are not simultaneously strictly positive

- Adding disjunctive variables $(\delta_j)_{j\in J}$ such that $\delta_j = \begin{cases} 1 \text{ if } j \text{ is early} \\ 0 \text{ if } j \text{ is tardy} \end{cases}$

$$\forall j \in J,\ e_j \geqslant 0 \quad (e.0) \qquad e_j \leqslant M\,\delta_j \quad (e.1)$$

$$\forall j \in J,\ t_j \geqslant 0 \quad (t.1) \qquad t_j \leqslant M\,(1-\delta_j) \quad (t.2)$$

where $\boldsymbol{M} = \sum_{j\in J} p_j$

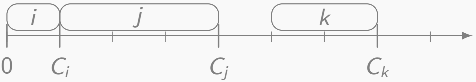
[**non-overlapping**] processing intervals are pairwise disjoints

- decomposing non-overlapping for a schedule :

the early tasks	the tardy tasks
• are entirely processed before d	• are entirely processsed after d
• do not overlap each other	• do not overlap each other

Non-overlapping inequalities for $1 \mid - \mid \min \sum \omega_j C_j$

Queyranne's non-overlapping inequalities

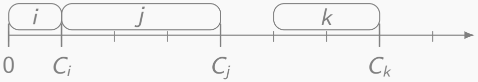


$$\forall S \subset J, \sum_{j \in S} p_j C_j \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

- scheduling problem without due-date

Non-overlapping inequalities for $1 | - | \min \sum \omega_j C_j$

Queyranne's non-overlapping inequalities

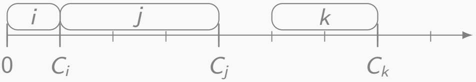


$$\forall S \subset J, \sum_{j \in S} p_j C_j \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

- scheduling problem without due-date
- encoding schedules by their completion times $(C_j)_{j \in J}$

Non-overlapping inequalities for $1 \mid - \mid \min \sum \omega_j C_j$

Queyranne's non-overlapping inequalities

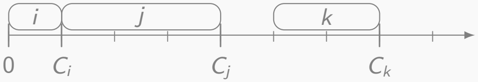


$$\forall S \subset J, \sum_{j \in S} p_j C_j \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

- scheduling problem without due-date
- encoding schedules by their completion times $(C_j)_{j \in J}$
- these inequalities describe the convex hull of such vectors C

Non-overlapping inequalities for $1 \mid - \mid \min \sum \omega_j C_j$

Queyranne's non-overlapping inequalities

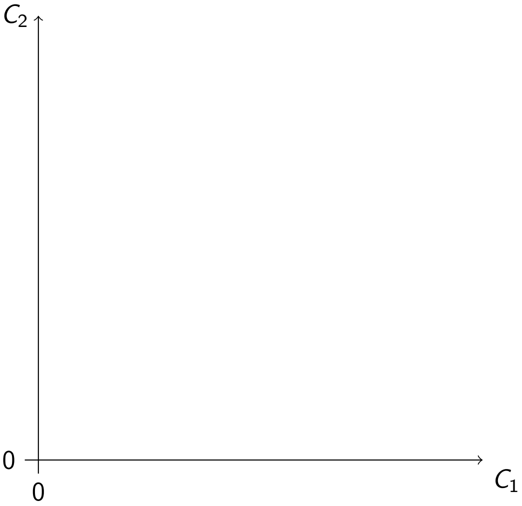


$$\forall S \subset J, \sum_{j \in S} p_j C_j \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

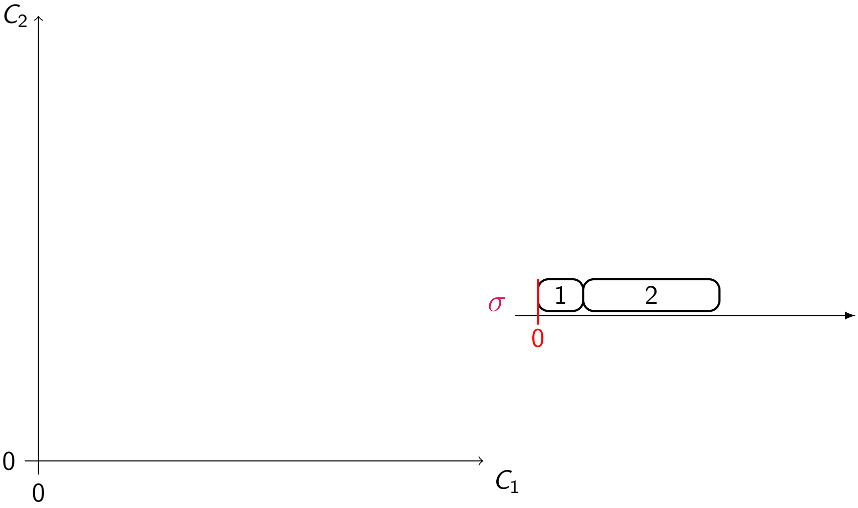
- scheduling problem without due-date
- encoding schedules by their completion times $(C_j)_{j \in J}$
- these inequalities describe the convex hull of such vectors C
- all extreme points of the polyhedron encode feasible schedules

Queyranne, 1993, Mathematical Programming

The set of vectors (C_1, C_2) encoding a 2-task schedule

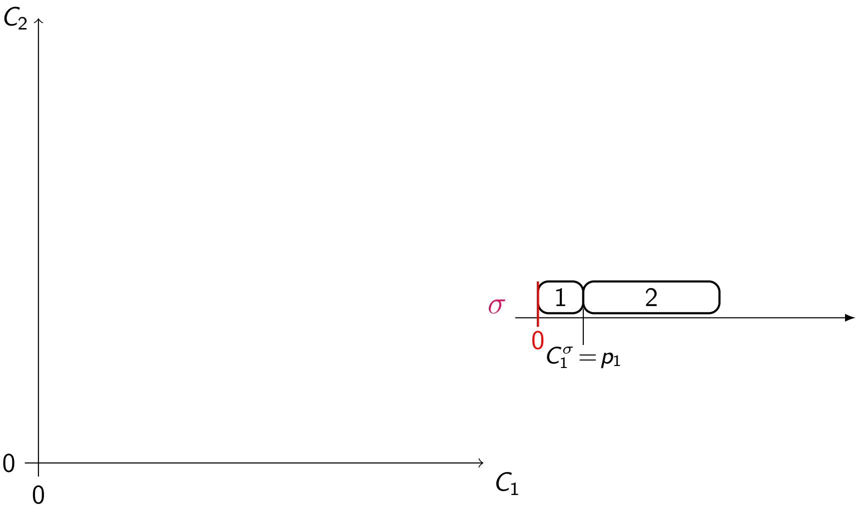


The set of vectors (C_1, C_2) encoding a 2-task schedule

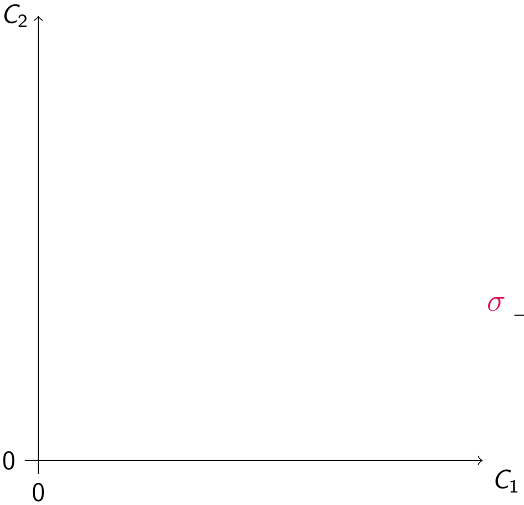


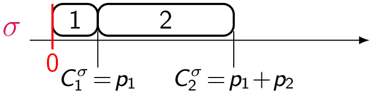
Queyranne, 1993, Mathematical Programming

The set of vectors (C_1, C_2) encoding a 2-task schedule



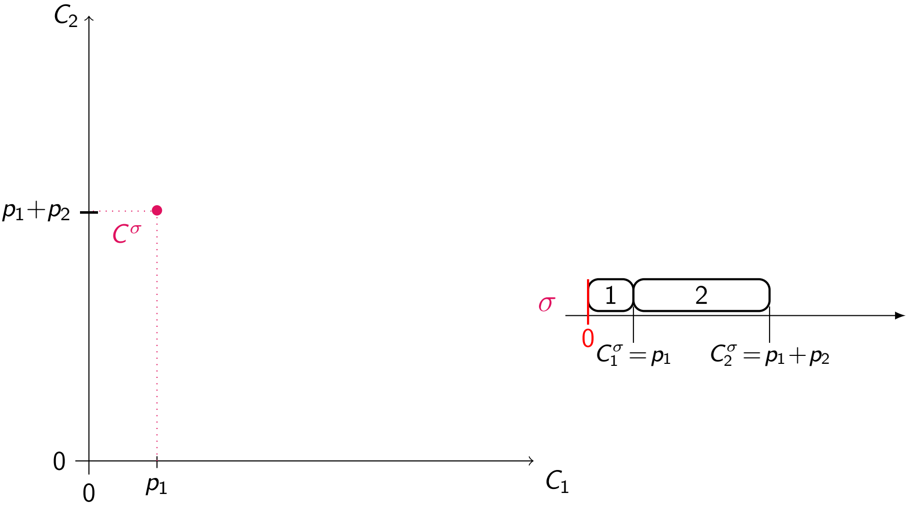
The set of vectors (C_1, C_2) encoding a 2-task schedule



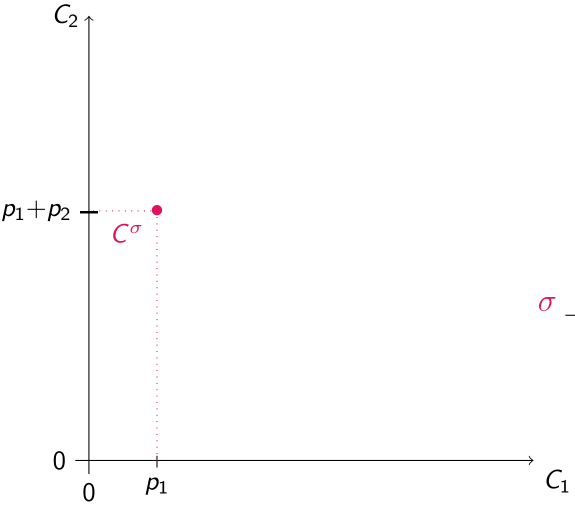


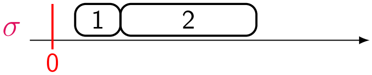
Queyranne, 1993, Mathematical Programming

The set of vectors (C_1, C_2) encoding a 2-task schedule

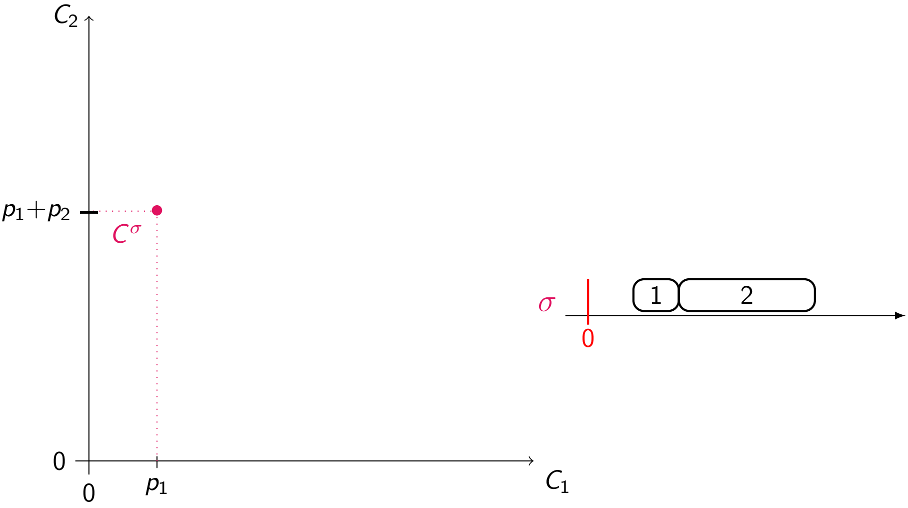


The set of vectors (C_1, C_2) encoding a 2-task schedule

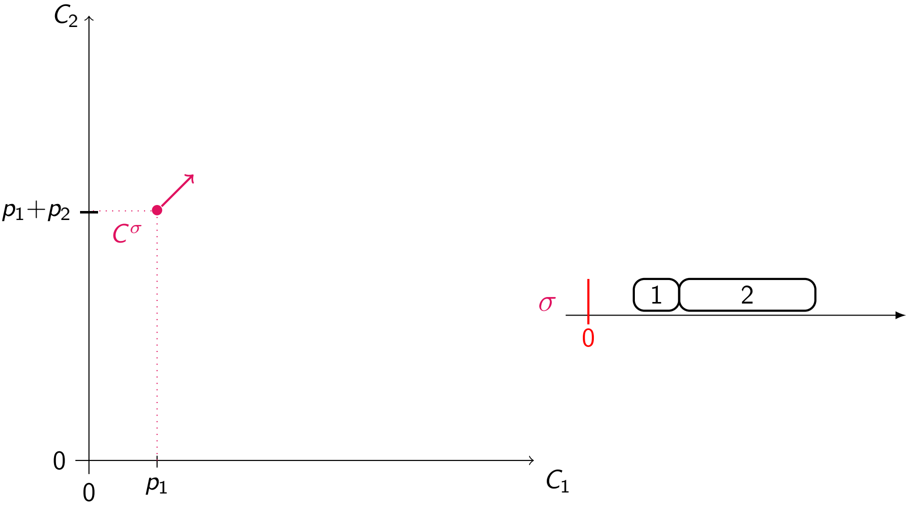




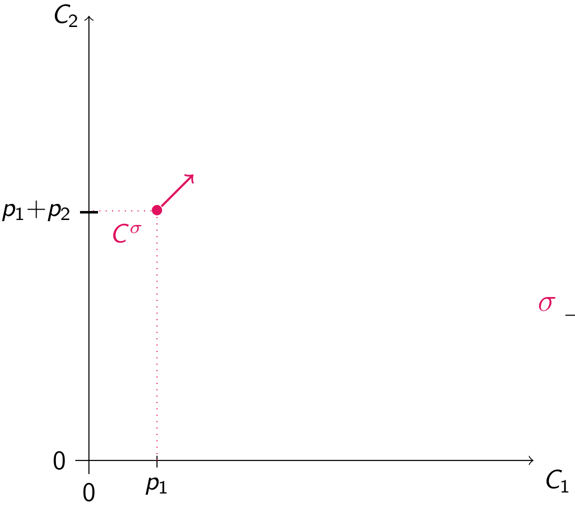
The set of vectors (C_1, C_2) encoding a 2-task schedule

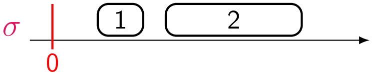


The set of vectors (C_1, C_2) encoding a 2-task schedule

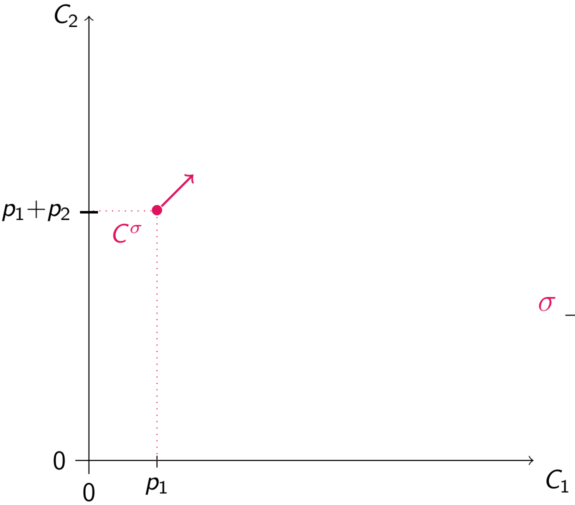


The set of vectors (C_1, C_2) encoding a 2-task schedule

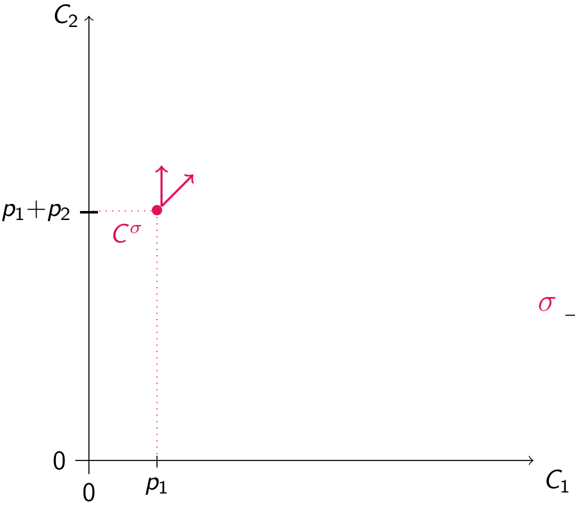




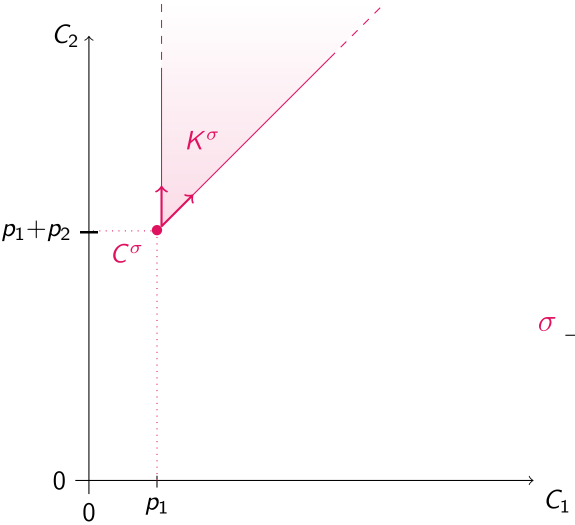
The set of vectors (C_1, C_2) encoding a 2-task schedule



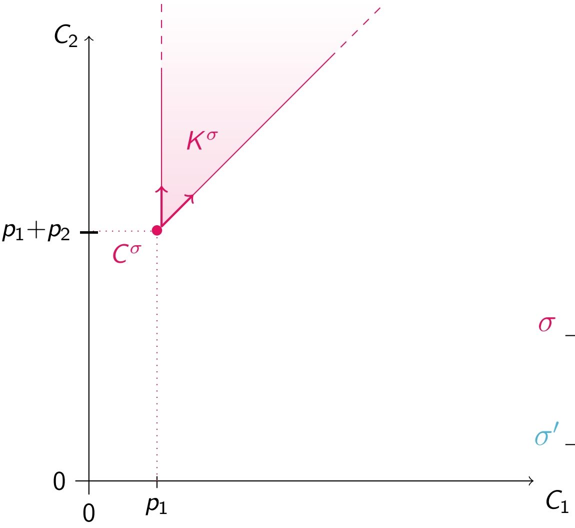
The set of vectors (C_1, C_2) encoding a 2-task schedule

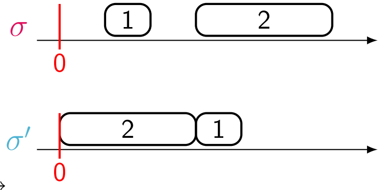


The set of vectors (C_1, C_2) encoding a 2-task schedule

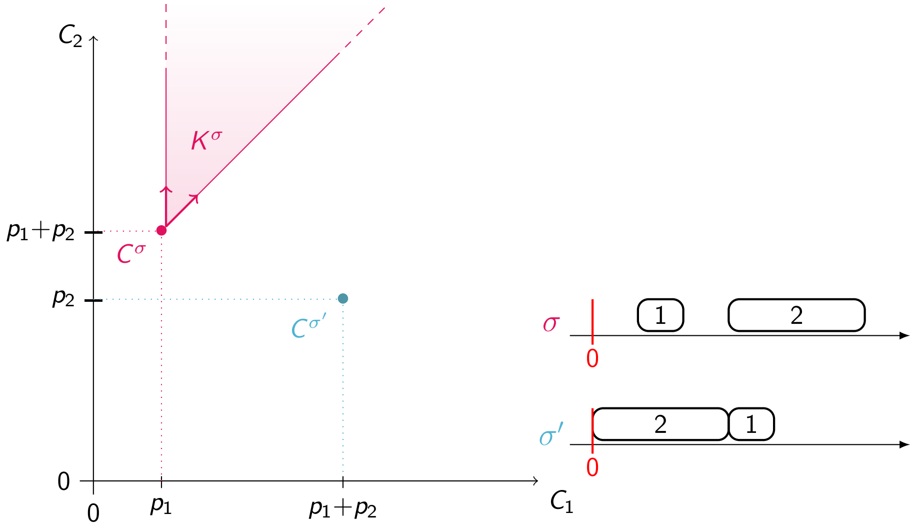


The set of vectors (C_1, C_2) encoding a 2-task schedule

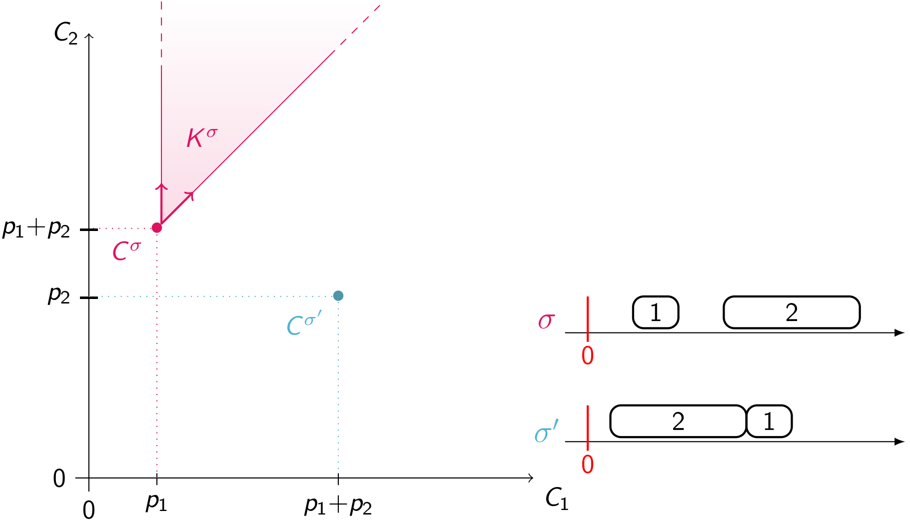




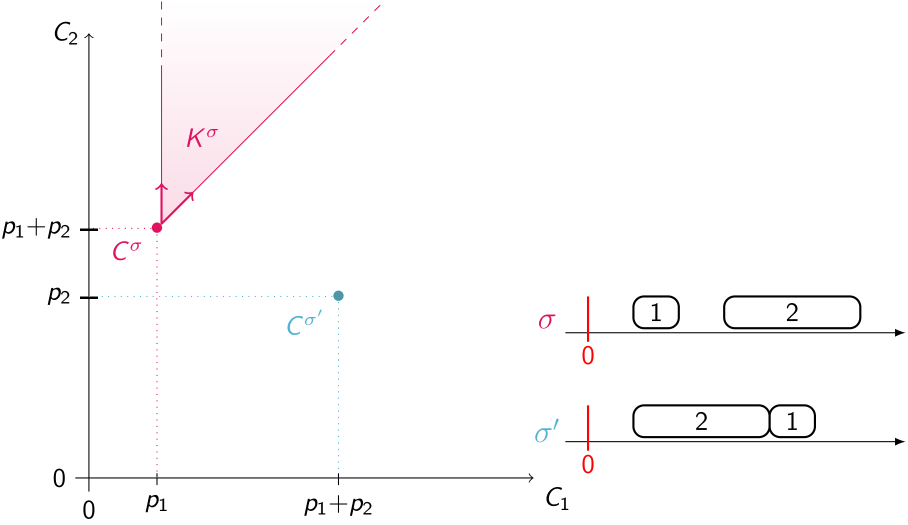
The set of vectors (C_1, C_2) encoding a 2-task schedule



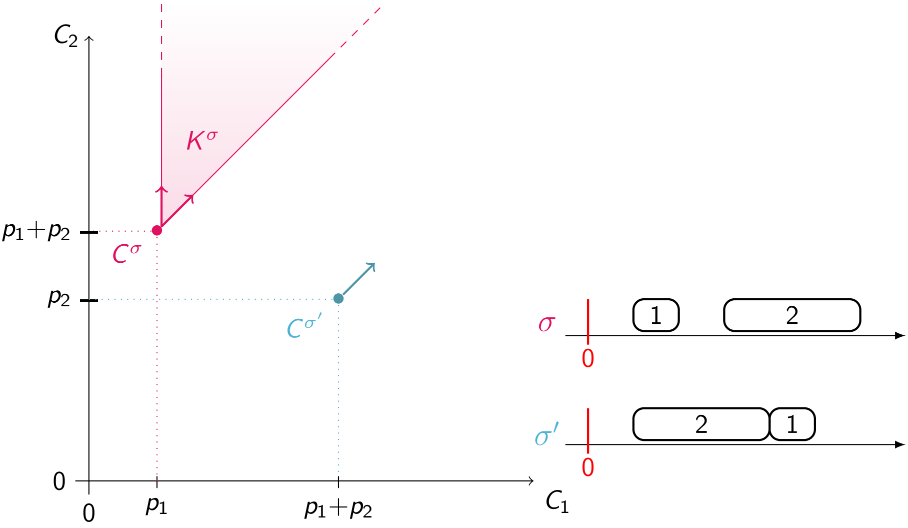
The set of vectors (C_1, C_2) encoding a 2-task schedule



The set of vectors (C_1, C_2) encoding a 2-task schedule

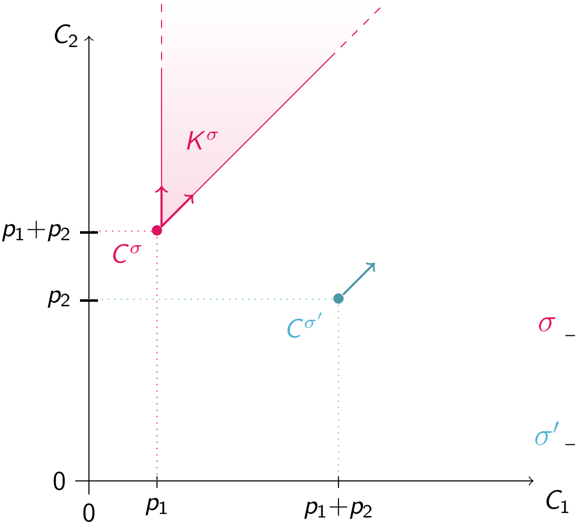


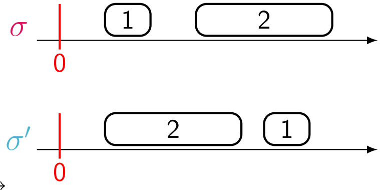
The set of vectors (C_1, C_2) encoding a 2-task schedule



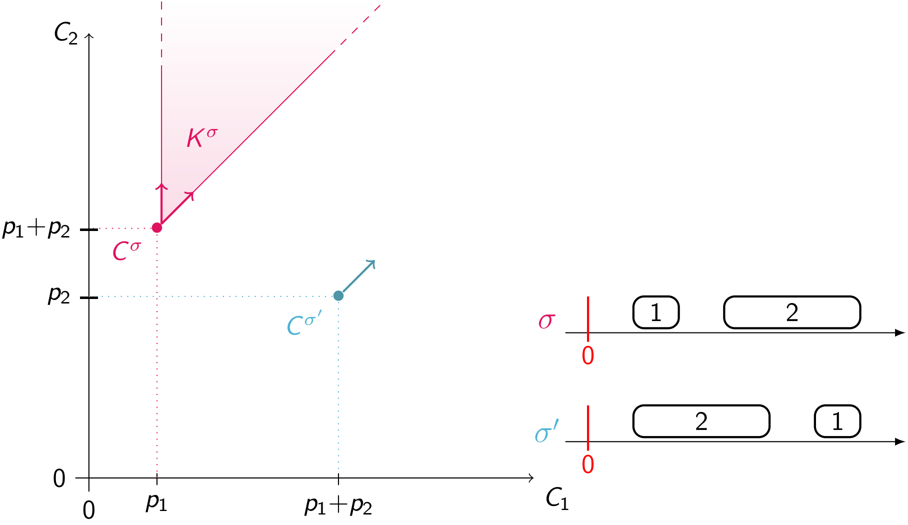
Queyranne, 1993, Mathematical Programming

The set of vectors (C_1, C_2) encoding a 2-task schedule

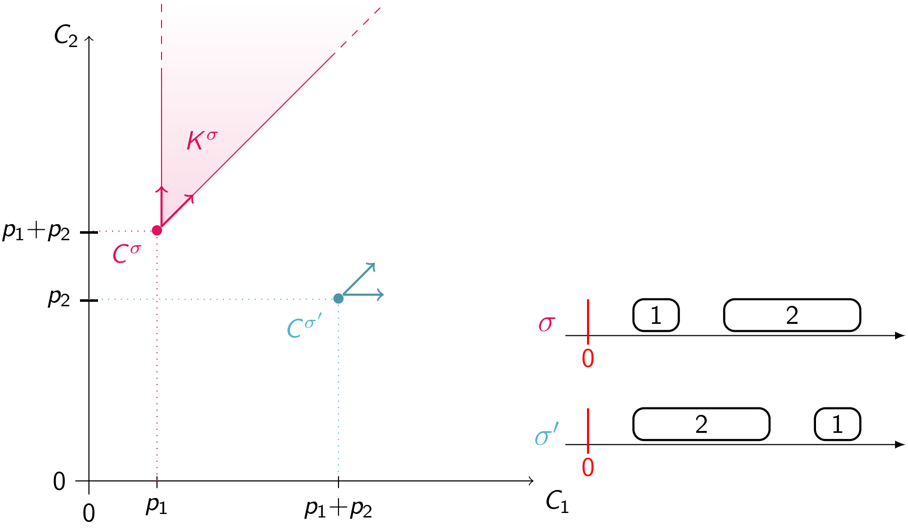




The set of vectors (C_1, C_2) encoding a 2-task schedule

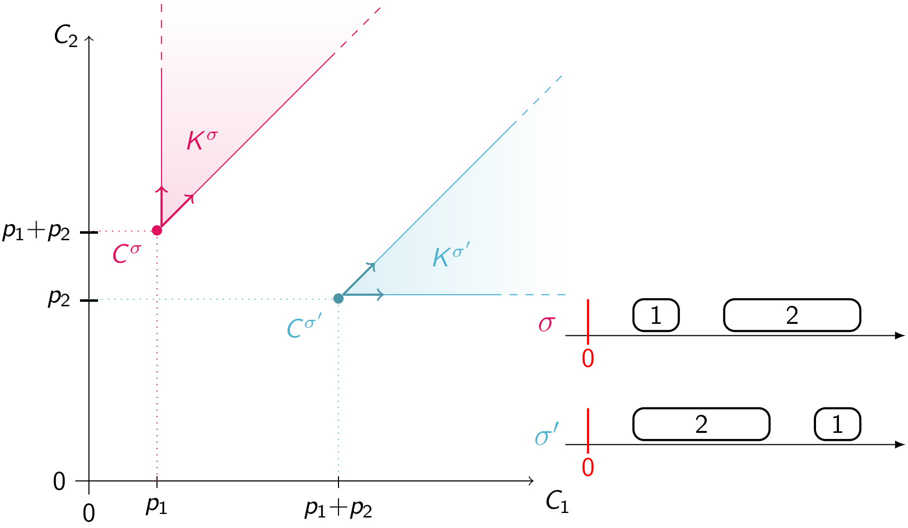


The set of vectors (C_1, C_2) encoding a 2-task schedule



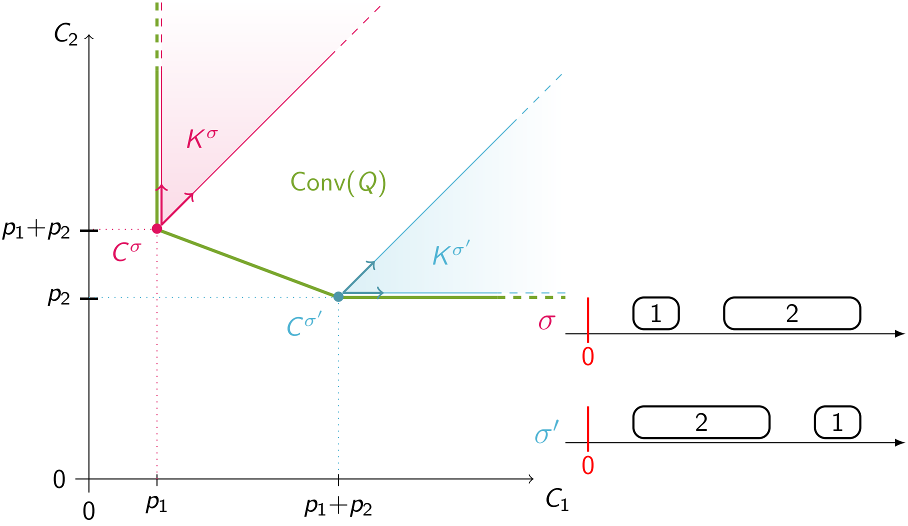
Queyranne, 1993, Mathematical Programming

The set of vectors (C_1, C_2) encoding a 2-task schedule



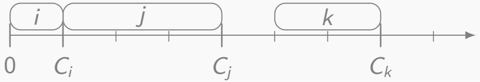
Queyranne, 1993, Mathematical Programming

The set of vectors (C_1, C_2) encoding a 2-task schedule



Non-overlapping inequalities for UCDDP

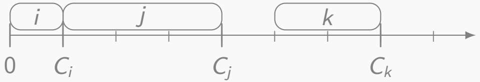
Queyranne's non-overlapping inequalities



$$\forall S \subseteq J, \underbrace{\sum_{i \in S} p_i C_i}_{p*C(S)} \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

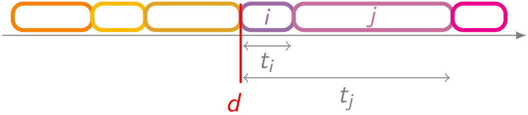
Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



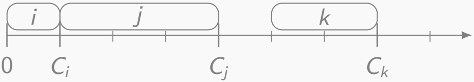
$$\forall S \subseteq J, \underbrace{\sum_{i\in S} p_i C_i}_{p*C(S)} \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j\in S} p_j^2 + \big(\sum_{j\in S} p_j\big)^2\Big]$$

A first idea : {



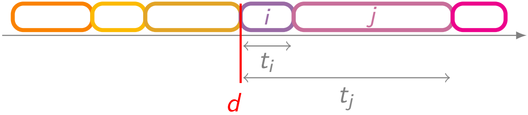
Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



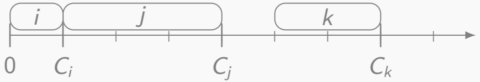
$$\forall S \subseteq J,\ \underbrace{\textstyle\sum_{i\in S} p_i C_i}_{p*C(S)} \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j\in S} p_j^2 + \big(\sum_{j\in S} p_j\big)^2\Big]$$

A first idea : $\begin{cases} \forall S \subseteq J,\ p * t(S) \geqslant g(S) \\ \end{cases}$



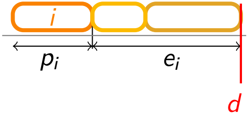
Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



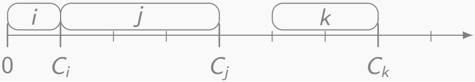
$$\forall S \subseteq J, \underbrace{\sum_{i \in S} p_i C_i}_{p * C(S)} \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

A first idea : $\begin{cases} \forall S \subseteq J,\ p * t(S) \geqslant g(S) \\ \end{cases}$



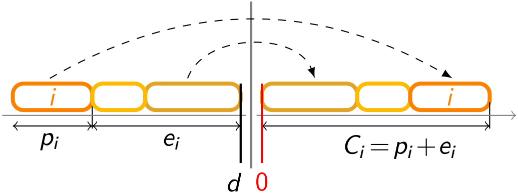
Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



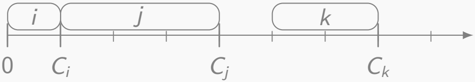
$\forall S \subseteq J, \underbrace{\sum_{i \in S} p_i C_i}_{p*C(S)} \geqslant g(S)$ where $g(S) = \frac{1}{2}\left[\sum_{j \in S} p_j^2 + (\sum_{j \in S} p_j)^2\right]$

A first idea : $\begin{cases} \forall S \subseteq J,\ p * t(S) \geqslant g(S) \\ \end{cases}$



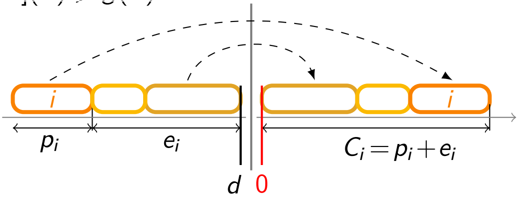
Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



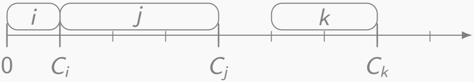
$$\forall S \subseteq J, \underbrace{\sum_{i \in S} p_i C_i}_{p*C(S)} \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

A first idea :
$$\begin{cases} \forall S \subseteq J,\ p * t(S) \geqslant g(S) \\ \forall S \subseteq J,\ p * [p + e](S) \geqslant g(S) \end{cases}$$



Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



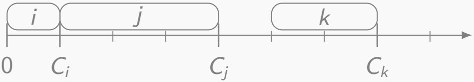
$$\forall S \subseteq J,\ \underbrace{\sum_{i\in S} p_i C_i}_{p*C(S)} \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j\in S} p_j^2 + \big(\sum_{j\in S} p_j\big)^2\Big]$$

A first idea :

$$\begin{cases} \forall S \subseteq J,\ p * t(S \cap T) \geqslant g(S \cap T) \\ \forall S \subseteq J,\ p * [p + e](S \cap E) \geqslant g(S \cap E) \end{cases}$$

Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



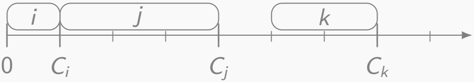
$$\forall S \subseteq J, \underbrace{\sum_{i\in S} p_i C_i}_{p*C(S)} \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j\in S} p_j^2 + \big(\sum_{j\in S} p_j\big)^2\Big]$$

A first idea :
$$\begin{cases} \forall S \subseteq J,\ p * t(S\cap T) \geqslant g(S\cap T) \\ \forall S \subseteq J,\ p * [p+e](S\cap E) \geqslant g(S\cap E) \end{cases}$$

Using δ_j variables $E = \{j \in J \mid \delta_j = 1\}$ and $T = \{j \in J \mid \delta_j = 0\}$

Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



$\forall S \subseteq J, \underbrace{\sum_{i \in S} p_i C_i}_{p*C(S)} \geqslant g(S)$ where $g(S) = \frac{1}{2}\left[\sum_{j \in S} p_j^2 + (\sum_{j \in S} p_j)^2\right]$

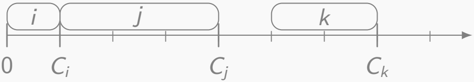
A first idea : $\begin{cases} \forall S \subseteq J,\ p * t(S \cap T) \geqslant g(S \cap T) \\ \forall S \subseteq J,\ p * [p + e](S \cap E) \geqslant g(S \cap E) \end{cases}$

Using δ_j variables $E = \{j \in J \mid \delta_j = 1\}$ and $T = \{j \in J \mid \delta_j = 0\}$

- the right-hand side term is no more a constant

Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



$$\forall S \subseteq J, \underbrace{\sum_{i \in S} p_i C_i}_{p*C(S)} \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

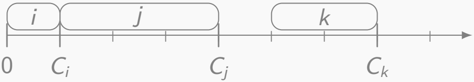
A first idea :
$$\begin{cases} \forall S \subseteq J,\ p * t(S \cap T) \geqslant g(S \cap T) \\ \forall S \subseteq J,\ p * [p + e](S \cap E) \geqslant g(S \cap E) \end{cases}$$

Using δ_j variables $E = \{j \in J \mid \delta_j = 1\}$ and $T = \{j \in J \mid \delta_j = 0\}$

- the right-hand side term is no more a constant
- variables δ appear on both side to express the intersection

Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



$$\forall S \subseteq J, \underbrace{\sum_{i \in S} p_i C_i}_{p * C(S)} \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

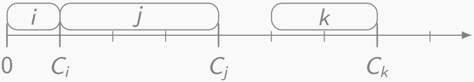
A first idea :
$$\begin{cases} \forall S \subseteq J,\ p * t(S \cap T) \geqslant g(S \cap T) \\ \forall S \subseteq J,\ p * [p + e](S \cap E) \geqslant g(S \cap E) \end{cases}$$

Using δ_j variables $E = \{j \in J \mid \delta_j = 1\}$ and $T = \{j \in J \mid \delta_j = 0\}$

- the right-hand side term is no more a constant
- variables δ appear on both side to express the intersection
- products $\delta_i \delta_j$ appear

Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



$$\forall S \subseteq J, \underbrace{\sum_{i \in S} p_i C_i}_{p * C(S)} \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

A first idea :
$$\begin{cases} \forall S \subseteq J,\ p * t(S \cap T) \geqslant g(S \cap T) \\ \forall S \subseteq J,\ p * [p + e](S \cap E) \geqslant g(S \cap E) \end{cases}$$

Using δ_j variables $E = \{j \in J \mid \delta_j = 1\}$ and $T = \{j \in J \mid \delta_j = 0\}$

- the right-hand side term is no more a constant
- variables δ appear on both side to express the intersection
- products $\delta_i\, \delta_j$ appear
  $\hookrightarrow$ linearisation variables are needed

Formulation F^3 for UCDDP

$$\forall (i,j) \in J^{<},\ \begin{aligned} X_{i,j} &\geqslant 0 && (x.1)\\ X_{i,j} &\leqslant \delta_i + \delta_j && (x.2)\\ X_{i,j} &\geqslant \delta_i - \delta_j && (x.3)\\ X_{i,j} &\geqslant 2 - \delta_i - \delta_j && (x.4) \end{aligned}$$

Formulation F^3 for UCDDP

$$\forall (i,j) \in J^<, \quad \begin{aligned} X_{i,j} &\geqslant 0 && (x.1)\\ X_{i,j} &\leqslant \delta_i + \delta_j && (x.2)\\ X_{i,j} &\geqslant \delta_i - \delta_j && (x.3)\\ X_{i,j} &\geqslant 2 - \delta_i - \delta_j && (x.4) \end{aligned}$$

$$\forall S \in \mathcal{P}(J), \quad \sum_{i \in S} p_i\, e_i \geqslant \sum_{(i,j) \in S^<} p_i\, p_j\, \frac{\delta_i + \delta_j - X_{i,j}}{2} \qquad (S1)$$

$$\sum_{i \in S} p_i\, t_i \geqslant \sum_{(i,j) \in S^<} p_i\, p_j\, \frac{2 - (\delta_i + \delta_j) - X_{i,j}}{2} + \sum_{i \in S} p_i^2 (1 - \delta_i) \qquad (S2)$$

Formulation F^3 for UCDDP

$$P^3 = \left\{ (e,t,\delta,X) \;\middle|\; \begin{array}{ll}
\forall j \in J, 0 \leqslant \delta_j \leqslant 1 & (\delta) \\
\forall j \in J,\ e_j \geqslant 0 \quad (e.0) \qquad e_j \leqslant M\,\delta_j \quad (e.1) & \forall j \in J,\ t_j \geqslant 0 \quad (t.1) \qquad t_j \leqslant M\,(1-\delta_j) \quad (t.2) \\
\forall (i,j) \in J^<,\ X_{i,j} \geqslant 0 & (x.1) \\
\qquad X_{i,j} \leqslant \delta_i + \delta_j & (x.2) \\
\qquad X_{i,j} \geqslant \delta_i - \delta_j & (x.3) \\
\qquad X_{i,j} \geqslant 2 - \delta_i - \delta_j & (x.4) \\
\forall S \in \mathcal{P}(J),\ \sum_{i \in S} p_i\, e_i \geqslant \sum_{(i,j) \in S^<} p_i\, p_j\, \frac{\delta_i + \delta_j - X_{i,j}}{2} & (S1) \\
\qquad \sum_{i \in S} p_i\, t_i \geqslant \sum_{(i,j) \in S^<} p_i\, p_j\, \frac{2-(\delta_i+\delta_j) - X_{i,j}}{2} + \sum_{i \in S} p_i^2 (1-\delta_i) & (S2)
\end{array} \right\}$$

Formulation F^3 for UCDDP

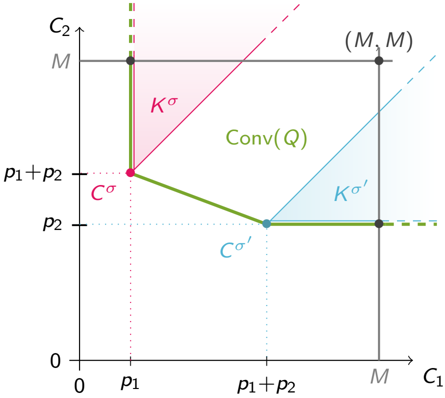
$$F^3 : \min \left\{ \sum_{j\in J} \alpha_j\, e_j + \beta_j\, t_j \;\middle|\; (e,t,\delta,X) \in \mathrm{extr}(P^3) \text{ and } \delta \in \{0,1\}^J \right\}$$

where:

$$P^3 = \left\{ (e,t,\delta,X) \;\middle|\; \begin{array}{lll}
\forall j\in J, 0 \leqslant \delta_j \leqslant 1 & & (\delta) \\
\forall j \in J,\ e_j \geqslant 0 & & (e.0) \\
 e_j \leqslant M\,\delta_j & & (e.1) \\
\forall j \in J,\ t_j \geqslant 0 & & (t.1) \\
 t_j \leqslant M\,(1-\delta_j) & & (t.2) \\
\forall (i,j) \in J^<,\ X_{i,j} \geqslant 0 & & (x.1) \\
\phantom{\forall (i,j) \in J^<,\ } X_{i,j} \leqslant \delta_i + \delta_j & & (x.2) \\
\phantom{\forall (i,j) \in J^<,\ } X_{i,j} \geqslant \delta_i - \delta_j & & (x.3) \\
\phantom{\forall (i,j) \in J^<,\ } X_{i,j} \geqslant 2 - \delta_i - \delta_j & & (x.4) \\
\forall S \in \mathcal{P}(J),\ \sum_{i\in S} p_i\, e_i \geqslant \sum_{(i,j)\in S^<} p_i\, p_j\, \frac{\delta_i+\delta_j-X_{i,j}}{2} & & (S1) \\
\phantom{\forall S \in \mathcal{P}(J),\ } \sum_{i\in S} p_i\, t_i \geqslant \sum_{(i,j)\in S^<} p_i\, p_j\, \frac{2-(\delta_i+\delta_j)-X_{i,j}}{2} + \sum_{i\in S} p_i^2 (1-\delta_i) & & (S2)
\end{array} \right\}$$

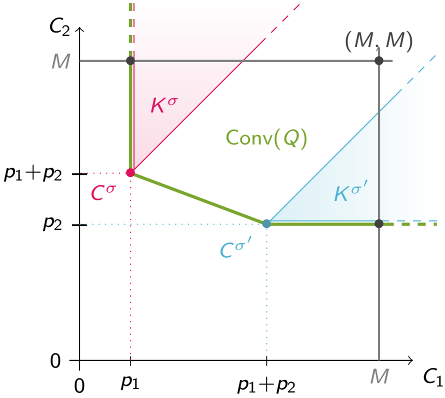
Validity of F^3

- validity proof
  - is not based on a geometrical proof
  - must be compatible with additional inequalities



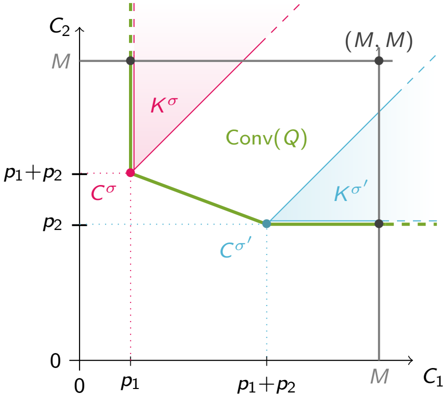
Validity of F^3

- validity proof
  - is not based on a geometrical proof
  - must be compatible with additional inequalities
- first lemma:
  a point satisfying non-overlapping ineq. that corresponds to a schedule with an overlapp is the middle of two others



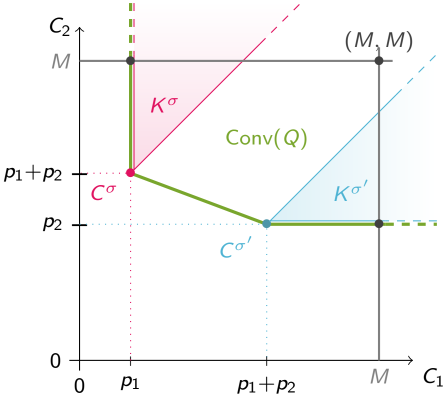
Validity of F^3

- validity proof
  - is not based on a geometrical proof
  - must be compatible with additional inequalities
- first lemma:
  a point satisfying non-overlapping ineq. that corresponds to a schedule with an overlapp is the middle of two others
  $\hookrightarrow$ is not extreme



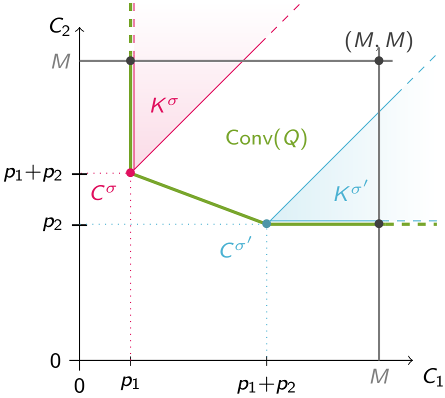
Validity of F^3

- validity proof
  - is not based on a geometrical proof
  - must be compatible with additional inequalities
- first lemma:
  a point satisfying non-overlapping ineq. that corresponds to a schedule with an overlapp is the middle of two others
  ↪ is not extreme
- second lemma:
  a point satisfying non-overlapping ineq. that corresponds to a schedule with a late task is larger that another



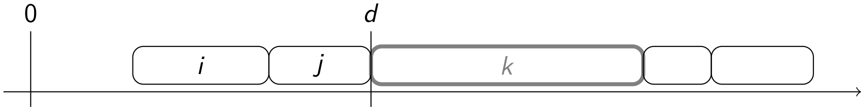
Validity of F^3

- validity proof
  - is not based on a geometrical proof
  - must be compatible with additional inequalities
- first lemma:
  a point satisfying non-overlapping ineq. that corresponds to a schedule with an overlapp is the middle of two others
  ↪ is not extreme
- second lemma:
  a point satisfying non-overlapping ineq. that corresponds to a schedule with a late task is larger that another
  ↪ is not minimal



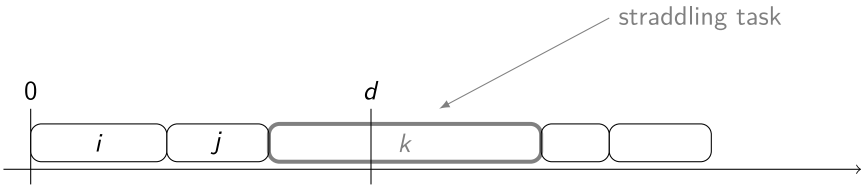
How to adapt the formulation for the general case ?

- unrestrictive case: d-blocks are dominant



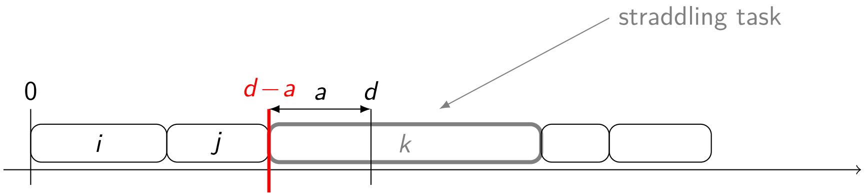
How to adapt the formulation for the general case ?

- unrestrictive case: d-blocks are dominant
- **general case**: d-or-left-blocks are dominant



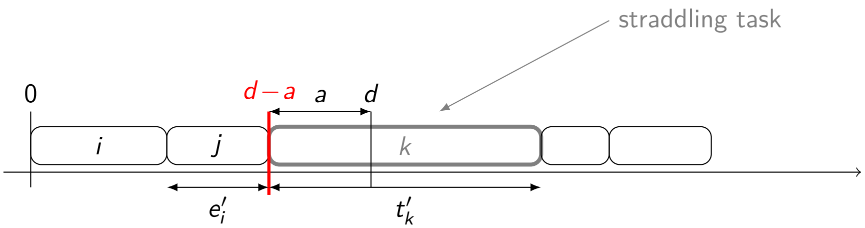
How to adapt the formulation for the general case ?

- unrestrictive case: d-blocks are dominant
- **general case**: d-or-left-blocks are dominant
  ↪ new variable a for a **new reference point**: $d - a$



How to adapt the formulation for the general case ?

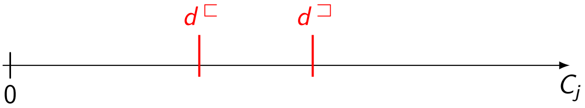
- unrestrictive case: d-blocks are dominant
- **general case**: d-or-left-blocks are dominant
  ↪ new variable a for a **new reference point**: $d-a$



Interest of a flexible reference point?

Common due window problem:

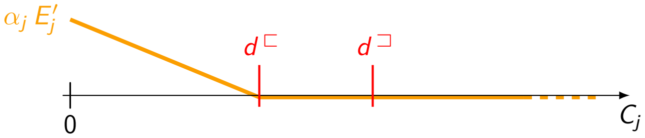
→ a due window $[d^{\sqsubset}, d^{\sqsupset}]$ instead of a due date d



Interest of a flexible reference point?

Common due window problem:

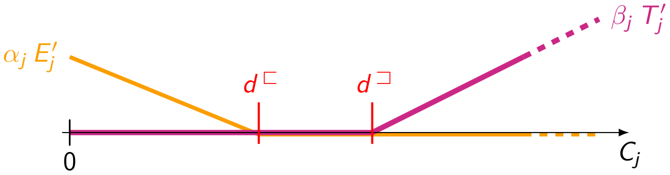
→ a due window $[d^{\sqsubset}, d^{\sqsupset}]$ instead of a due date d



Interest of a flexible reference point?

Common due window problem:

→ a due window $[d^{\sqsubset}, d^{\sqsupset}]$ instead of a due date d



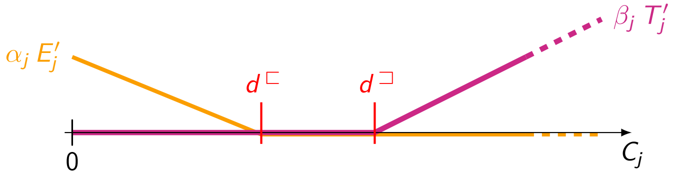
→ block with a task completing at $d^{\sqsubset}$ or $d^{\sqsupset}$ are not dominant

↪ a straddling task can occur over $d^{\sqsubset}$ (resp. over $d^{\sqsupset}$)

Interest of a flexible reference point?

Common due window problem:

→ a due window $[d^{\sqsubset}, d^{\sqsupset}]$ instead of a due date d



→ block with a task completing at $d^{\sqsubset}$ or $d^{\sqsupset}$ are not dominant

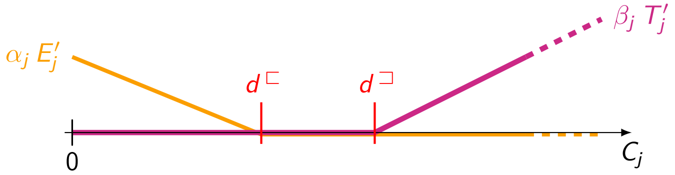
↪ a straddling task can occur over $d^{\sqsubset}$ (resp. over $d^{\sqsupset}$)

↪ two half-axes with flexible reference point

Interest of a flexible reference point?

Common due window problem:

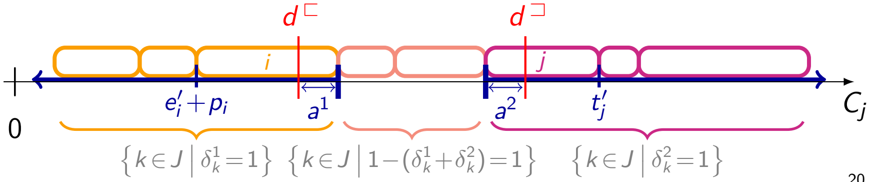
→ a due window $[d^{\sqsubset}, d^{\sqsupset}]$ instead of a due date d



→ block with a task completing at $d^{\sqsubset}$ or $d^{\sqsupset}$ are not dominant

↪ a straddling task can occur over $d^{\sqsubset}$ (resp. over $d^{\sqsupset}$)

↪ two half-axes with flexible reference point



What are the particularities of our formulations?

The two proposed formulations are linear formulations with:

What are the particularities of our formulations?

The two proposed formulations are linear formulations with:

- integer variables

What are the particularities of our formulations?

The two proposed formulations are linear formulations with:

- integer variables

  $\hookrightarrow$ Branch-and-Bound algorithm

What are the particularities of our formulations?

The two proposed formulations are linear formulations with:

- integer variables

  $\hookrightarrow$ Branch-and-Bound algorithm $\rightarrow$ branching on δ (and γ) variables

What are the particularities of our formulations?

The two proposed formulations are linear formulations with:

- integer variables

  $\hookrightarrow$ Branch-and-Bound algorithm $\rightarrow$ branching on δ (and γ) variables

- exponential number of inequalities

What are the particularities of our formulations?

The two proposed formulations are linear formulations with:

- integer variables
  $\hookrightarrow$ Branch-and-Bound algorithm $\rightarrow$ branching on δ (and γ) variables
- exponential number of inequalities
  $\hookrightarrow$ Branch-and-Cut algorithm

What are the particularities of our formulations?

The two proposed formulations are linear formulations with:

- integer variables

  $\hookrightarrow$ Branch-and-Bound algorithm $\rightarrow$ branching on δ (and γ) variables

- exponential number of inequalities

  $\hookrightarrow$ Branch-and-Cut algorithm $\rightarrow$ polynomial separation algorithm

What are the particularities of our formulations?

The two proposed formulations are linear formulations with:

- integer variables
  $\hookrightarrow$ Branch-and-Bound algorithm $\rightarrow$ branching on δ (and γ) variables
- exponential number of inequalities
  $\hookrightarrow$ Branch-and-Cut algorithm $\rightarrow$ polynomial separation algorithm
- extremality constraints

What are the particularities of our formulations?

The two proposed formulations are linear formulations with:

- integer variables
  $\hookrightarrow$ Branch-and-Bound algorithm $\rightarrow$ branching on δ (and γ) variables
- exponential number of inequalities
  $\hookrightarrow$ Branch-and-Cut algorithm $\rightarrow$ polynomial separation algorithm
- extremality constraints
  $\hookrightarrow$ ensuring the solutions extremality in spite of the branching scheme

How to ensure extremality during a Branch-and-Bound?

each node is solved by an extreme point $\Longrightarrow$ the solution of the Branch-and-Bound is an extreme point

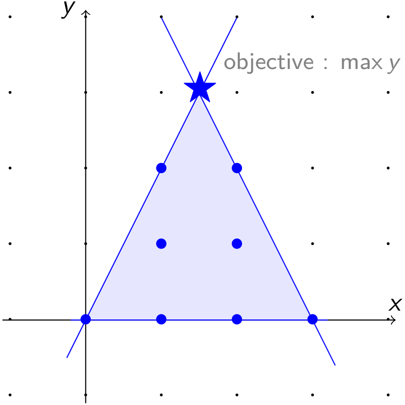
How to ensure extremality during a Branch-and-Bound?

each node is solved by an extreme point $\not\Longrightarrow$ the solution of the Branch-and-Bound is an extreme point

How to ensure extremality during a Branch-and-Bound?

each node is solved by an extreme point $\not\Longrightarrow$ the solution of the Branch-and-Bound is an extreme point

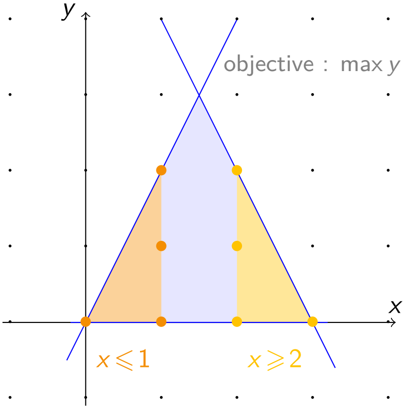
Counter-example:



How to ensure extremality during a Branch-and-Bound?

each node is solved by an extreme point $\not\Longrightarrow$ the solution of the Branch-and-Bound is an extreme point

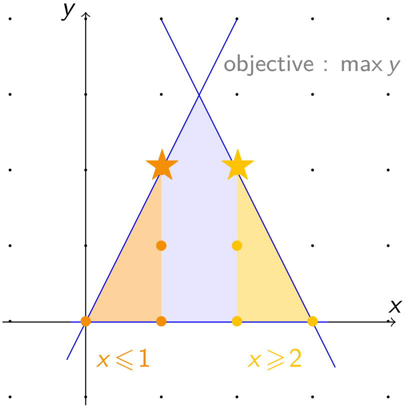
Counter-example:



How to ensure extremality during a Branch-and-Bound?

each node is solved by an extreme point $\not\Longrightarrow$ the solution of the Branch-and-Bound is an extreme point

Counter-example:



Conclusion on natural variable formulations

This kind of formulation with natural variables and non-overlapping inequalities allows to:

$\rightarrow$ formulate both UCDDP and CDDP

Conclusion on natural variable formulations

This kind of formulation with natural variables and non-overlapping inequalities allows to:

- $\rightarrow$ formulate both UCDDP and CDDP
- $\rightarrow$ formulate other similar problems
  (e.g. common due window, multi-machine common due date...)

Conclusion on natural variable formulations

This kind of formulation with natural variables and non-overlapping inequalities allows to:

- → formulate both UCDDP and CDDP
- → formulate other similar problems
  (e.g. common due window, multi-machine common due date...)
- → solve UCDDP instances up to size 40 within one hour

Conclusion on natural variable formulations

This kind of formulation with natural variables and non-overlapping inequalities allows to:

- $\rightarrow$ formulate both UCDDP and CDDP
- $\rightarrow$ formulate other similar problems
  (e.g. common due window, multi-machine common due date...)
- $\rightarrow$ solve UCDDP instances up to size 40 within one hour
- $\rightarrow$ solve CDDP instances up to size 20 or 30 within one hour

Conclusion on natural variable formulations

This kind of formulation with natural variables and non-overlapping inequalities allows to:

- → formulate both UCDDP and CDDP
- → formulate other similar problems
  (e.g. common due window, multi-machine common due date...)
- → solve UCDDP instances up to size 40 within one hour
- → solve CDDP instances up to size 20 or 30 within one hour

Why it is not so efficient?

Conclusion on natural variable formulations

This kind of formulation with natural variables and non-overlapping inequalities allows to:

- → formulate both UCDDP and CDDP
- → formulate other similar problems
  (e.g. common due window, multi-machine common due date...)
- → solve UCDDP instances up to size 40 within one hour
- → solve CDDP instances up to size 20 or 30 within one hour

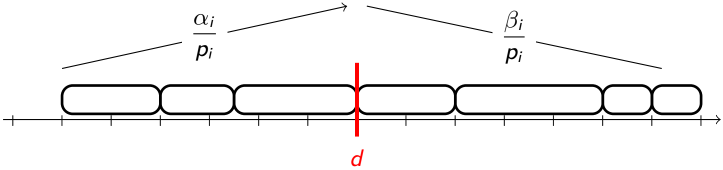
Why it is not so efficient? poor linear relaxation value

Outline

1. Introduction
2. Focus 1: A formulation for UCDDP using natural variables
3. Interlude : a first attempt to eliminate dominated solutions
4. Focus 2: dominance inequality
5. Conclusion and perspectives

A compact MIP formulation for UCDDP...

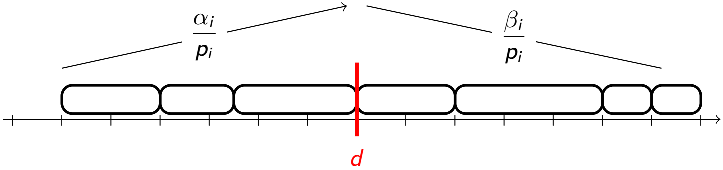
... using the V-shaped dominance property



- consider only the V-shaped d-blocks since they are dominant

A compact MIP formulation for UCDDP...

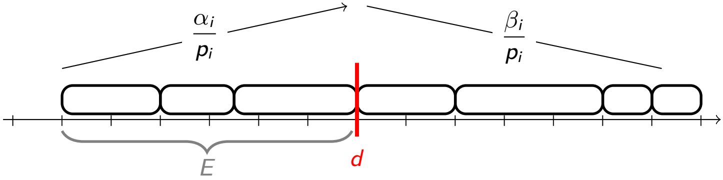
... using the V-shaped dominance property



- consider only the V-shaped d-blocks since they are dominant
- define the early/tardy partition of a V-shaped d-block

A compact MIP formulation for UCDDP...

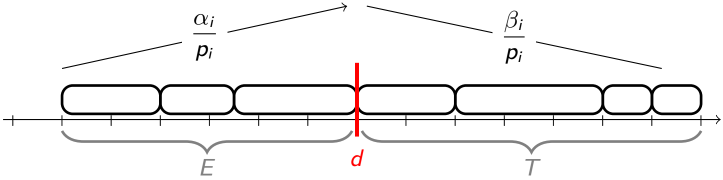
... using the V-shaped dominance property



- consider only the V-shaped d-blocks since they are dominant
- define the early/tardy partition of a V-shaped d-block

A compact MIP formulation for UCDDP...

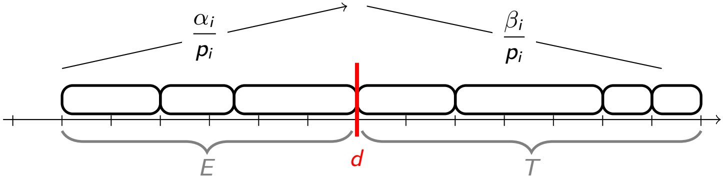
... using the V-shaped dominance property



- consider only the V-shaped d-blocks since they are dominant
- define the early/tardy partition of a V-shaped d-block

A compact MIP formulation for UCDDP...

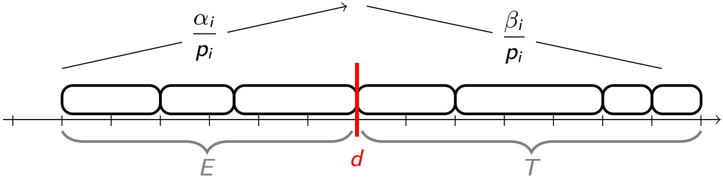
... using the V-shaped dominance property



- consider only the V-shaped d-blocks since they are dominant
- define the early/tardy partition of a V-shaped d-block
- same early/tardy partition $(E, T) \Leftrightarrow$ same penalty $f(E, T)$

A compact MIP formulation for UCDDP...

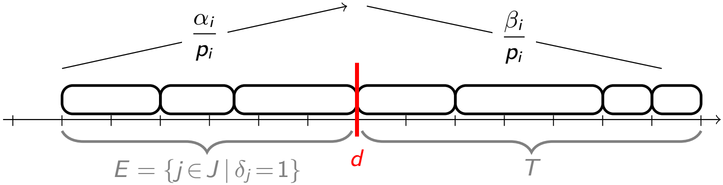
... using the V-shaped dominance property



- consider only the V-shaped d-blocks since they are dominant
- define the early/tardy partition of a V-shaped d-block
- same early/tardy partition $(E, T) \Leftrightarrow$ same penalty $f(E, T)$
- formulate UCDDP as a partition problem $\min_{\{E,T\} \text{ bi-partition of } J} f(E, T)$

A compact MIP formulation for UCDDP...

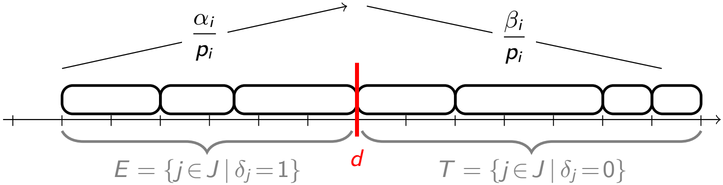
... using the V-shaped dominance property and δ and X variables.



- consider only the V-shaped d-blocks since they are dominant
- define the early/tardy partition of a V-shaped d-block
- same early/tardy partition $(E, T) \Leftrightarrow$ same penalty $f(E, T)$
- formulate UCDDP as a partition problem $\min_{\{E,T\} \text{ bi-partition of } J} f(E, T)$

A compact MIP formulation for UCDDP...

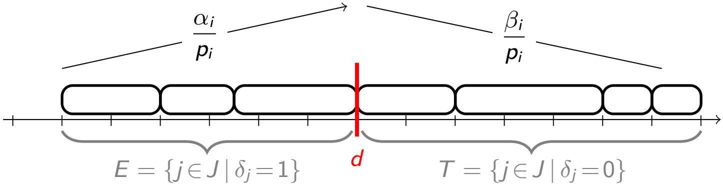
... using the V-shaped dominance property and δ and X variables.



- consider only the V-shaped d-blocks since they are dominant
- define the early/tardy partition of a V-shaped d-block
- same early/tardy partition $(E, T) \Leftrightarrow$ same penalty $f(E, T)$
- formulate UCDDP as a partition problem $\min_{\{E,T\} \text{ bi-partition of } J} f(E, T)$

A compact MIP formulation for UCDDP...

... using the V-shaped dominance property and δ and X variables.



- consider only the V-shaped d-blocks since they are dominant
- define the early/tardy partition of a V-shaped d-block
- same early/tardy partition $(E, T) \Leftrightarrow$ same penalty $f(E, T)$
- formulate UCDDP as a partition problem $\min_{\{E,T\} \text{ bi-partition of } J} f(E, T)$
- formulate UCDDP as a **MIP** $F^2 : \min_{(\delta,X) s.t. (X.1-X.4)} h_{\alpha,\beta}(\delta, X)$

  where $h_{\alpha,\beta}$ is a linear function depending on α and β

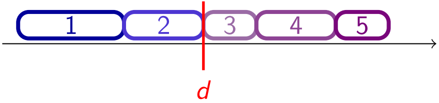
Link between F^2 and the Cut Polytope

in Formulation F^2 | in the complete graph K_n

Link between F^2 and the Cut Polytope

in Formulation F^2 — in the complete graph K_n

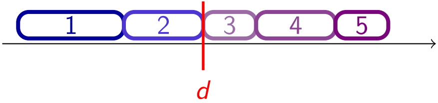
δ variables $\longleftrightarrow$ vertices

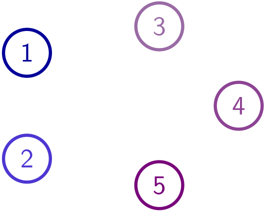


Link between F^2 and the Cut Polytope

in Formulation F^2 in the complete graph K_n

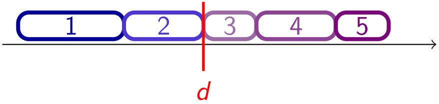
δ variables $\longleftrightarrow$ vertices

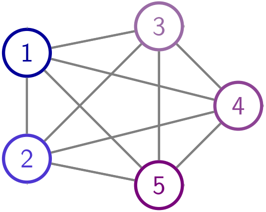




Link between F^2 and the Cut Polytope

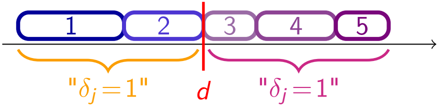
in Formulation F^2		in the complete graph K_n
δ variables	$\longleftrightarrow$	vertices
X variables	$\longleftrightarrow$	edges

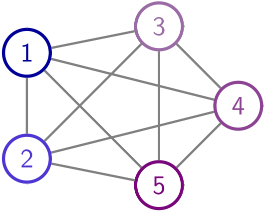




Link between F^2 and the Cut Polytope

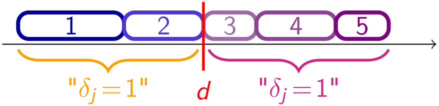
in Formulation F^2		in the complete graph K_n
δ variables	$\longleftrightarrow$	vertices
X variables	$\longleftrightarrow$	edges
$(\{"\delta_j = 1"\}, \{"\delta_j = 0"\})$	$\longleftrightarrow$	a vertices bipartition

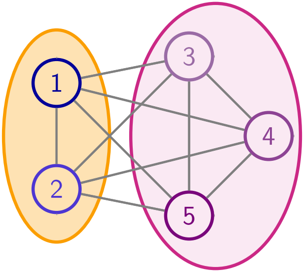




Link between F^2 and the Cut Polytope

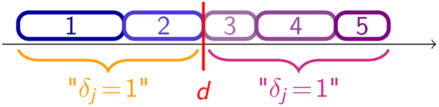
in Formulation F^2		in the complete graph K_n
δ variables	$\longleftrightarrow$	vertices
X variables	$\longleftrightarrow$	edges
$(\{"\delta_j = 1"\}, \{"\delta_j = 0"\})$	$\longleftrightarrow$	a vertices bipartition

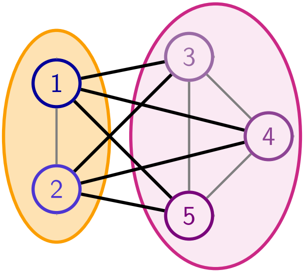




Link between F^2 and the Cut Polytope

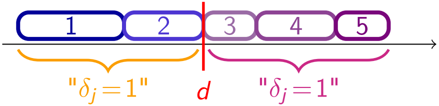
in Formulation F^2		in the complete graph K_n
δ variables	$\longleftrightarrow$	vertices
X variables	$\longleftrightarrow$	edges
$(\{``\delta_j = 1"\}, \{``\delta_j = 0"\})$	$\longleftrightarrow$	a vertices bipartition
$\{``X_{i,j} = 1"\}$	$\longleftrightarrow$	a **cut** in K_n

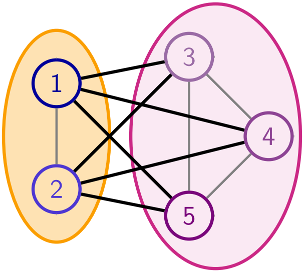




Link between F^2 and the Cut Polytope

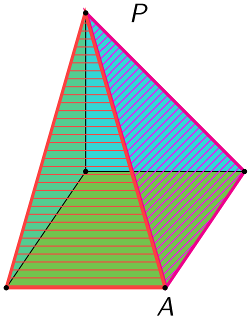
in Formulation F^2		in the complete graph K_n
δ variables	$\longleftrightarrow$	vertices
X variables	$\longleftrightarrow$	edges
$(\{"\delta_j=1"\},\{"\delta_j=0"\})$	$\longleftrightarrow$	a vertices bipartition
$\{"X_{i,j}=1"\}$	$\longleftrightarrow$	a **cut** in K_n
$(\delta, X)\in P^2$	$\longleftrightarrow$	$X \in \mathrm{CUT}_n$ the cut polytope for K_n





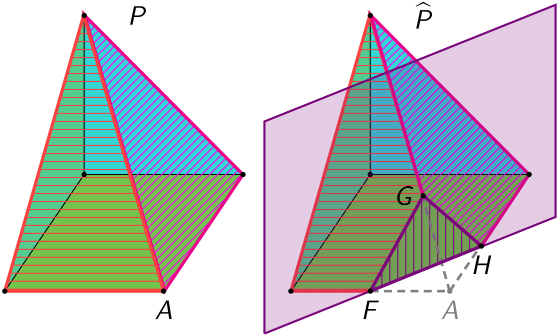
Idea

Eliminate an extreme point corresponding to a "bad" solution according to the objective value,



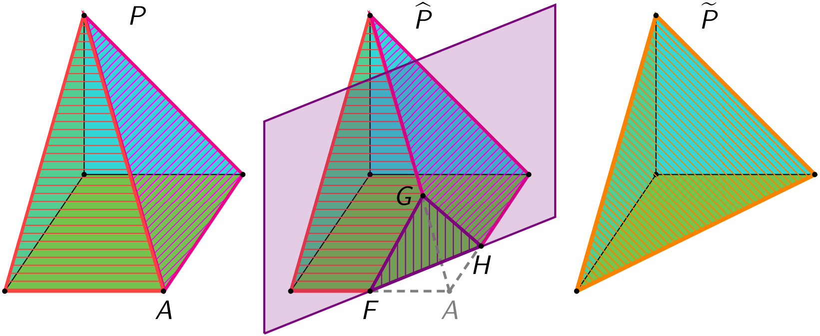
Idea

Eliminate an extreme point corresponding to a "bad" solution according to the objective value,



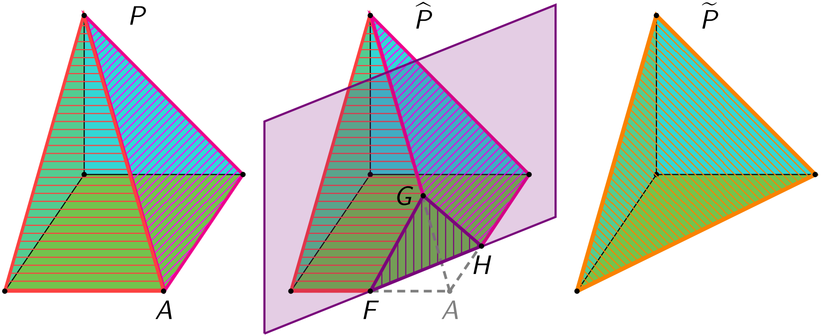
Idea

Eliminate an extreme point corresponding to a "bad" solution according to the objective value, using **facet defining** inequalities to avoid the apparition of new extreme points



Idea

Eliminate an extreme point corresponding to a "bad" solution according to the objective value, using **facet defining** inequalities to avoid the apparition of new extreme points

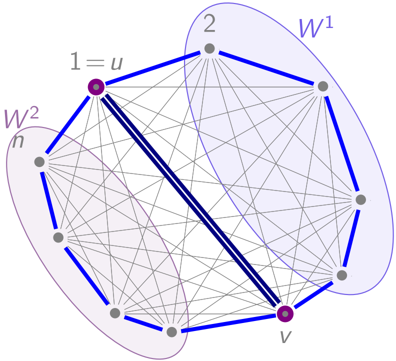


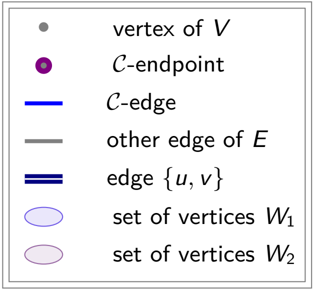
Application to P^2, elimination of $(\delta, X) = (0, 0)$,

$$\widetilde{P}^n_{\delta,X} = \text{conv}\left\{(\delta, X) \in \{0,1\}^J \times \{0,1\}^{J^<} \,\middle|\, (X.1 - X.4) \text{ and } \delta \neq 0\right\}$$

Example of a new facet defining inequality family

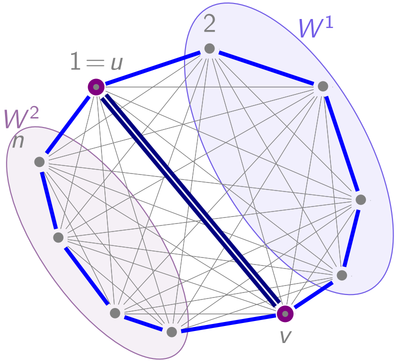
for $\mathcal{C}$ an hamiltonian cycle in K_n, $\underline{\delta_u+\delta_v} - \underline{X_{u,v}} + \underline{X(\mathcal{C})} \geqslant 2$

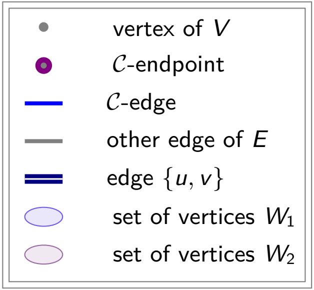




Example of a new facet defining inequality family

for $\mathcal{C}$ an hamiltonian cycle in K_n, $\underline{\delta_u + \delta_v} - \underline{X_{u,v}} + \underline{X(\mathcal{C})} \geqslant 2$





Too many and too various inequalities appear → change of strategy

Outline

Neighborhood based dominance properties : generic idea

Remark: If a solution is dominated by one of its neighbors,
then it is not an optimal solution.

Neighborhood based dominance properties : generic idea

Remark: If a solution is dominated by one of its neighbors, then it is not an optimal solution.

Consequence: The set of solutions **non-dominated** in their neighborhood is a strictly dominant set.

Neighborhood based dominance properties : generic idea

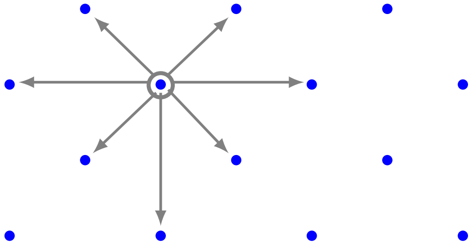
Remark: If a solution is dominated by one of its neighbors, then it is not an optimal solution.

Consequence: The set of solutions **non-dominated** in their neighborhood is a strictly dominant set.

Our approach:

- define a neighborhood based on operations
- translate the associate dominance property by linear inequalities

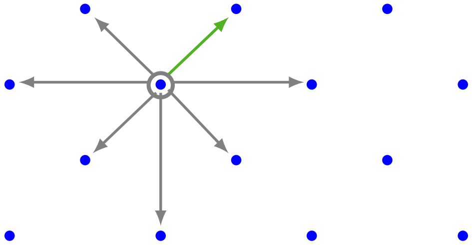
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

*= consider **all** the neighbors of **one** given solution*

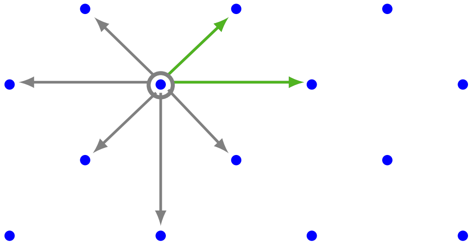
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

$= \textit{consider } \boldsymbol{all} \textit{ the neighbors of } \boldsymbol{one} \textit{ given solution}$

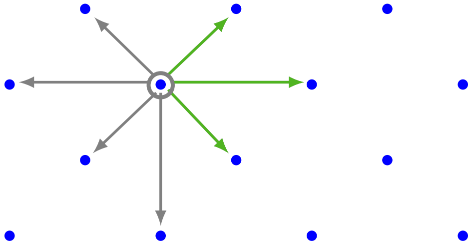
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

= *consider **all** the neighbors of **one** given solution*

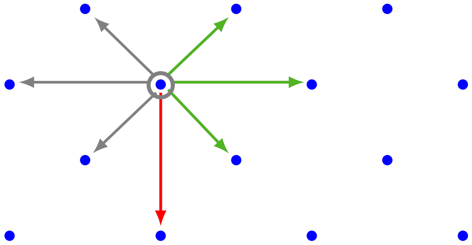
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

= *consider **all** the neighbors of **one** given solution*

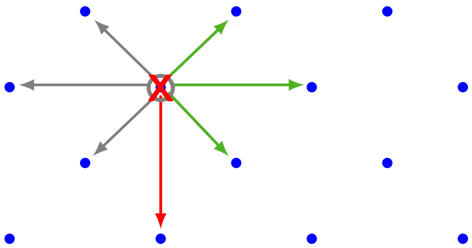
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

= *consider* ***all*** *the neighbors of* ***one*** *given solution*

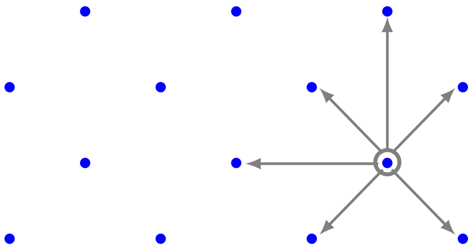
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

= *consider* ***all*** *the neighbors of* ***one*** *given solution*

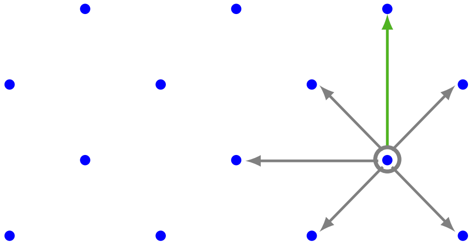
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

*= consider **all** the neighbors of **one** given solution*

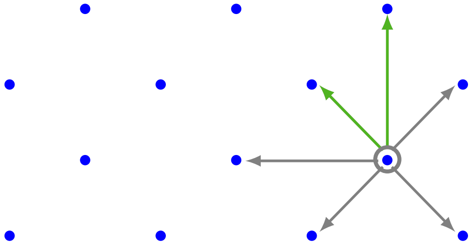
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

= *consider* ***all*** *the neighbors of* ***one*** *given solution*

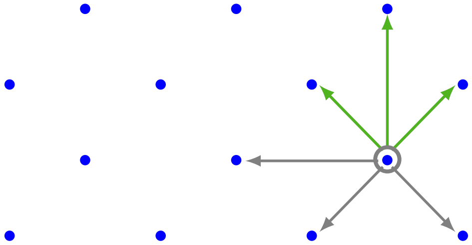
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

*= consider **all** the neighbors of **one** given solution*

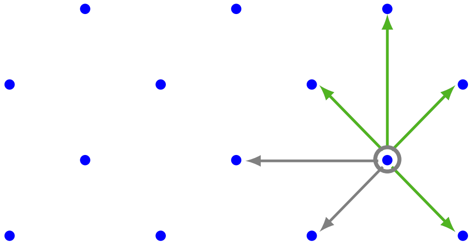
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

*= consider **all** the neighbors of **one** given solution*

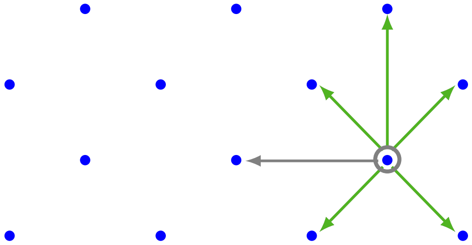
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

$= \textit{consider } \textbf{\textit{all}} \textit{ the neighbors of } \textbf{\textit{one}} \textit{ given solution}$

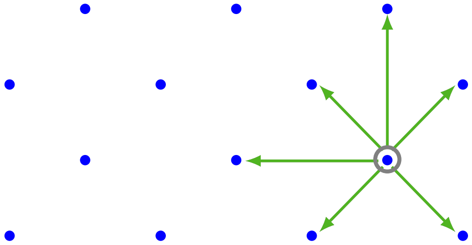
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

*= consider **all** the neighbors of **one** given solution*

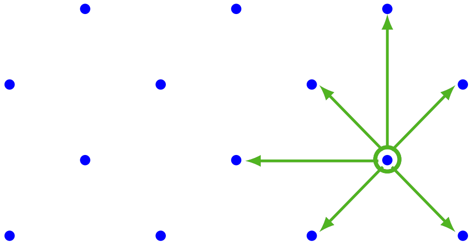
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

*= consider **all** the neighbors of **one** given solution*

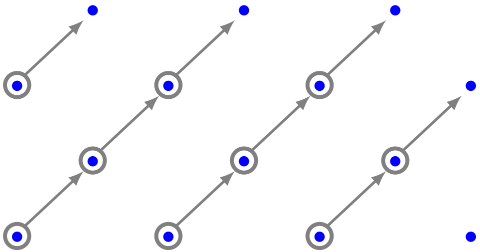
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

*= consider **all** the neighbors of **one** given solution*

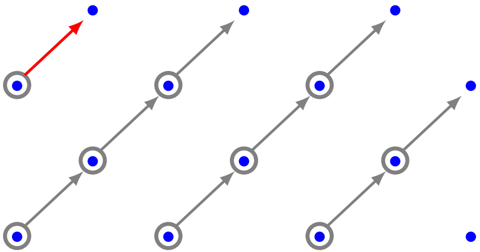
Neighborhood: solution-centered vs operation-centered point of view



Operation-centered

= *consider* ***one*** *given type of neighbor for* ***all*** *the solutions*

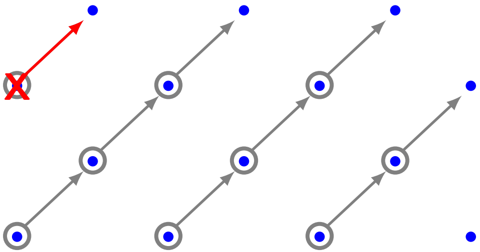
Neighborhood: solution-centered vs operation-centered point of view



Operation-centered

$= \textit{consider } \textbf{\textit{one}} \textit{ given type of neighbor for } \textbf{\textit{all}} \textit{ the solutions}$

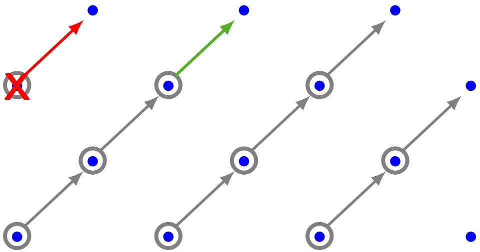
Neighborhood: solution-centered vs operation-centered point of view



Operation-centered

$= \textit{consider } \textbf{\textit{one}} \textit{ given type of neighbor for } \textbf{\textit{all}} \textit{ the solutions}$

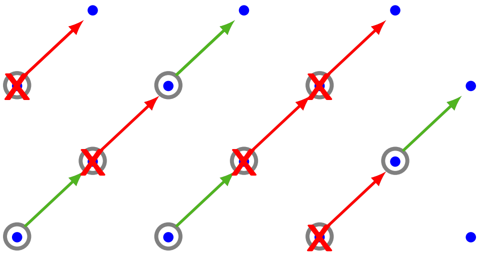
Neighborhood: solution-centered vs operation-centered point of view



Operation-centered

= *consider* ***one*** *given type of neighbor for* ***all*** *the solutions*

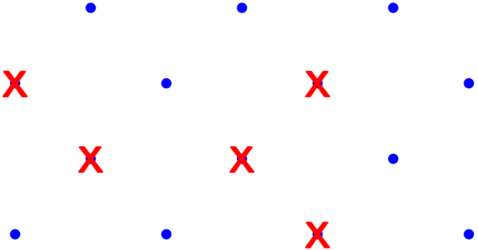
Neighborhood: solution-centered vs operation-centered point of view



Operation-centered

= *consider* ***one*** *given type of neighbor for* ***all*** *the solutions*

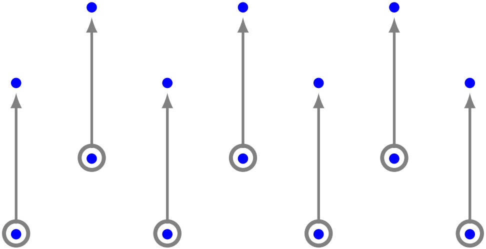
Neighborhood: solution-centered vs operation-centered point of view



Operation-centered

= *consider* ***one*** *given type of neighbor for* ***all*** *the solutions*

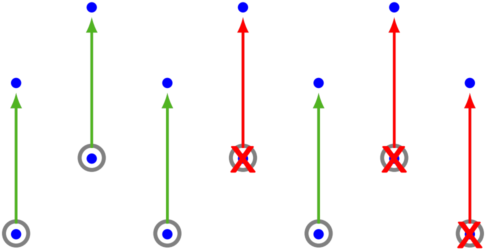
Neighborhood: solution-centered vs operation-centered point of view



Operation-centered

= *consider* ***one*** *given type of neighbor for* ***all*** *the solutions*

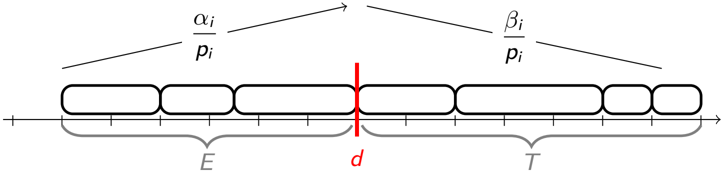
Neighborhood: solution-centered vs operation-centered point of view



Operation-centered

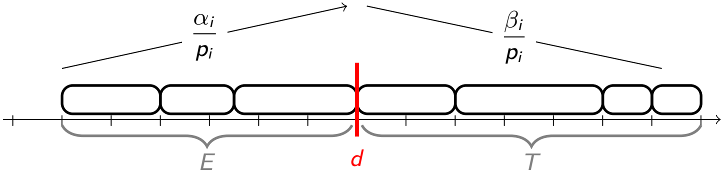
= *consider **one** given type of neighbor for **all** the solutions*

Structure of a V-shaped d-block



- introduce notations for any $u \in J$,

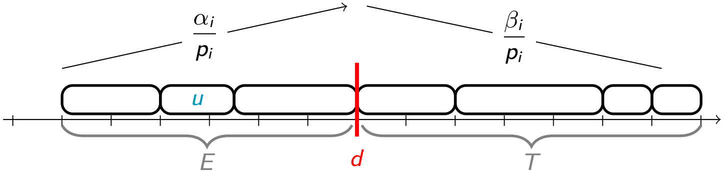
Structure of a V-shaped d-block



- introduce notations for any $u \in J$,

$$A(u) = \left\{ i \in J \mid \frac{\alpha_i}{p_i} > \frac{\alpha_u}{p_u} \right\} \quad \text{and} \quad \bar{A}(u) = \left\{ i \in J \mid i \neq u,\ \frac{\alpha_i}{p_i} \leqslant \frac{\alpha_u}{p_u} \right\}$$

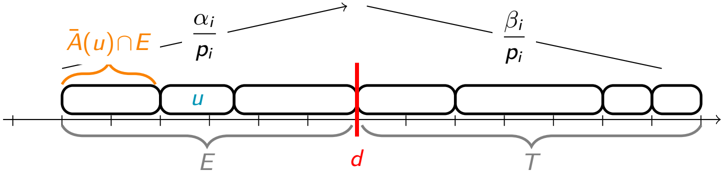
Structure of a V-shaped d-block



- introduce notations for any $u \in J$,

$$A(u) = \left\{ i \in J \mid \frac{\alpha_i}{p_i} > \frac{\alpha_u}{p_u} \right\} \quad \text{and} \quad \bar{A}(u) = \left\{ i \in J \mid i \neq u,\ \frac{\alpha_i}{p_i} \leqslant \frac{\alpha_u}{p_u} \right\}$$

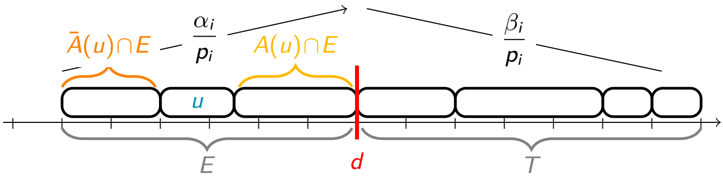
Structure of a V-shaped d-block



- introduce notations for any $u \in J$,

$$A(u) = \left\{ i \in J \mid \frac{\alpha_i}{p_i} > \frac{\alpha_u}{p_u} \right\} \quad \text{and} \quad \bar{A}(u) = \left\{ i \in J \mid i \neq u,\ \frac{\alpha_i}{p_i} \leqslant \frac{\alpha_u}{p_u} \right\}$$

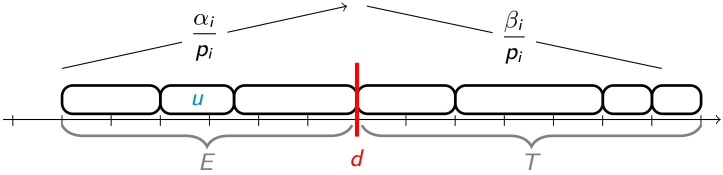
Structure of a V-shaped d-block



- introduce notations for any $u \in J$,

$$A(u) = \left\{ i \in J \mid \frac{\alpha_i}{p_i} > \frac{\alpha_u}{p_u} \right\} \quad \text{and} \quad \bar{A}(u) = \left\{ i \in J \mid i \neq u,\ \frac{\alpha_i}{p_i} \leqslant \frac{\alpha_u}{p_u} \right\}$$

Structure of a V-shaped d-block

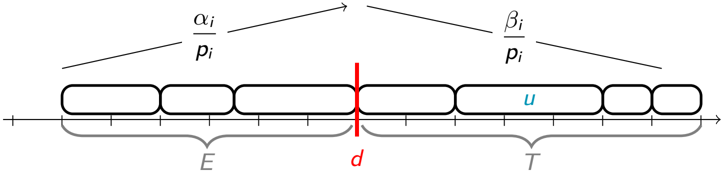


- introduce notations for any $u \in J$,

$$A(u) = \left\{ i \in J \mid \frac{\alpha_i}{p_i} > \frac{\alpha_u}{p_u} \right\} \quad \text{and} \quad \bar{A}(u) = \left\{ i \in J \mid i \neq u,\ \frac{\alpha_i}{p_i} \leqslant \frac{\alpha_u}{p_u} \right\}$$

$$B(u) = \left\{ i \in J \mid \frac{\beta_i}{p_i} > \frac{\beta_u}{p_u} \right\} \quad \text{and} \quad \bar{B}(u) = \left\{ i \in J \mid i \neq u,\ \frac{\beta_i}{p_i} \leqslant \frac{\beta_u}{p_u} \right\}$$

Structure of a V-shaped d-block

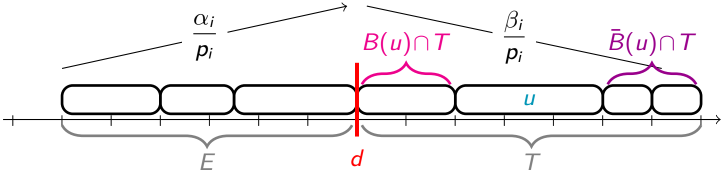


- introduce notations for any $u \in J$,

$$A(u) = \left\{ i \in J \mid \frac{\alpha_i}{p_i} > \frac{\alpha_u}{p_u} \right\} \quad \text{and} \quad \bar{A}(u) = \left\{ i \in J \mid i \neq u, \frac{\alpha_i}{p_i} \leqslant \frac{\alpha_u}{p_u} \right\}$$

$$B(u) = \left\{ i \in J \mid \frac{\beta_i}{p_i} > \frac{\beta_u}{p_u} \right\} \quad \text{and} \quad \bar{B}(u) = \left\{ i \in J \mid i \neq u, \frac{\beta_i}{p_i} \leqslant \frac{\beta_u}{p_u} \right\}$$

Structure of a V-shaped d-block

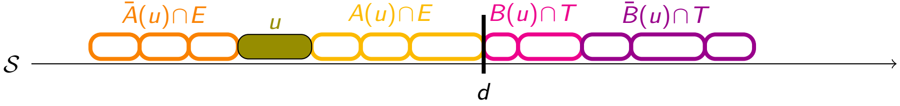


- introduce notations for any $u \in J$,

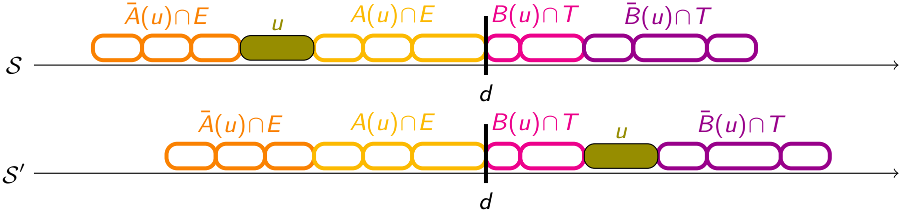
$$A(u) = \left\{ i \in J \mid \frac{\alpha_i}{p_i} > \frac{\alpha_u}{p_u} \right\} \quad \text{and} \quad \bar{A}(u) = \left\{ i \in J \mid i \neq u,\ \frac{\alpha_i}{p_i} \leqslant \frac{\alpha_u}{p_u} \right\}$$

$$B(u) = \left\{ i \in J \mid \frac{\beta_i}{p_i} > \frac{\beta_u}{p_u} \right\} \quad \text{and} \quad \bar{B}(u) = \left\{ i \in J \mid i \neq u,\ \frac{\beta_i}{p_i} \leqslant \frac{\beta_u}{p_u} \right\}$$

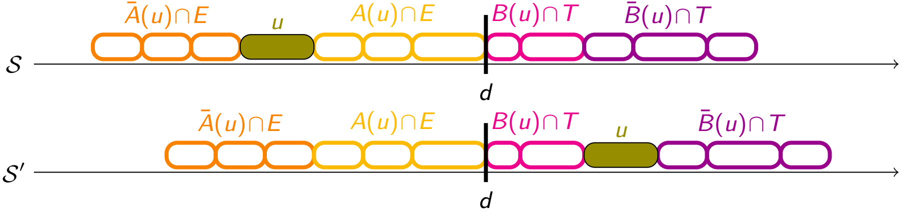
Cost variation induced by the insertion of an early task u



Cost variation induced by the insertion of an early task u

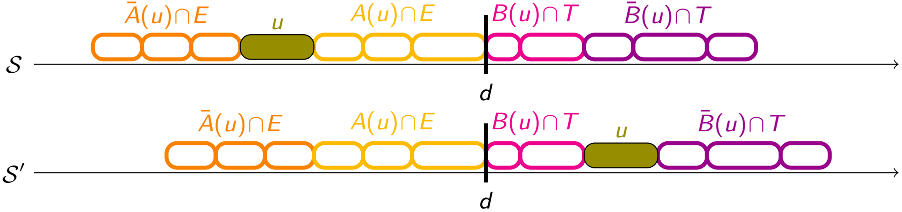


Cost variation induced by the insertion of an early task u



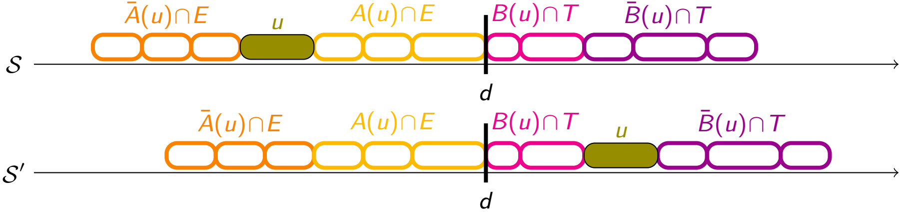
$\text{cost}(\mathcal{S}') = \text{cost}(\mathcal{S})$

Cost variation induced by the insertion of an early task u



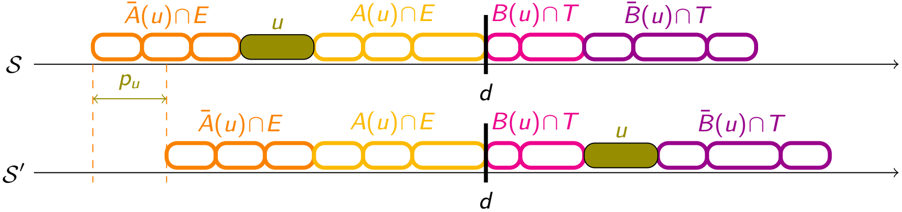
$$\text{cost}(\mathcal{S}') = \text{cost}(\mathcal{S}) - \alpha_u \, p\big(\underline{A(u) \cap E}\big)$$

Cost variation induced by the insertion of an early task u



$$\text{cost}(\mathcal{S}') = \text{cost}(\mathcal{S}) - \alpha_u \, p\big(\underline{A(u) \cap E}\big) + \beta_u \Big(p\big(\underline{B(u) \cap T}\big) + p_u\Big)$$

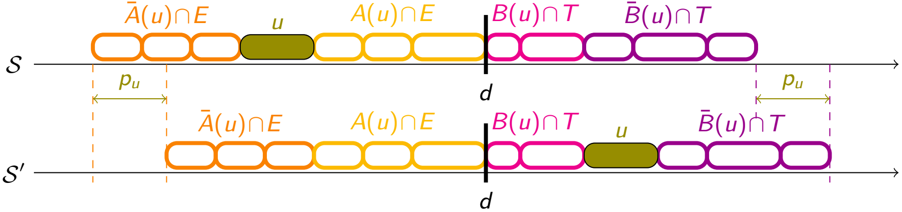
Cost variation induced by the insertion of an early task u



$$\text{cost}(\mathcal{S}') = \text{cost}(\mathcal{S}) - \alpha_u\, p\big(A(u) \cap E\big) + \beta_u \Big(p\big(B(u) \cap T\big) + p_u\Big)$$

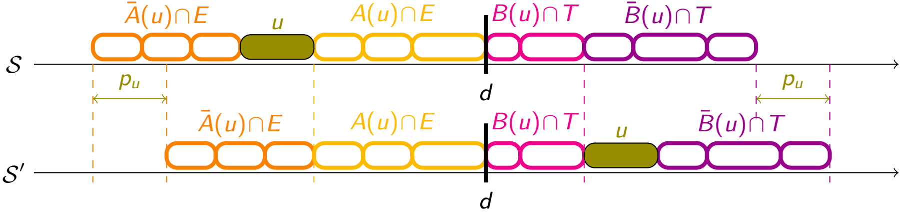
$$- p_u\, \alpha\big(\bar{A}(u) \cap E\big)$$

Cost variation induced by the insertion of an early task u



$$\mathrm{cost}(\mathcal{S}') = \mathrm{cost}(\mathcal{S}) - \alpha_u\, p\big(A(u)\cap E\big) + \beta_u \Big(p\big(B(u)\cap T\big) + p_u\Big)$$
$$- p_u\, \alpha\big(\bar{A}(u)\cap E\big) + p_u\, \beta\big(\bar{B}(v)\cap T\big)$$

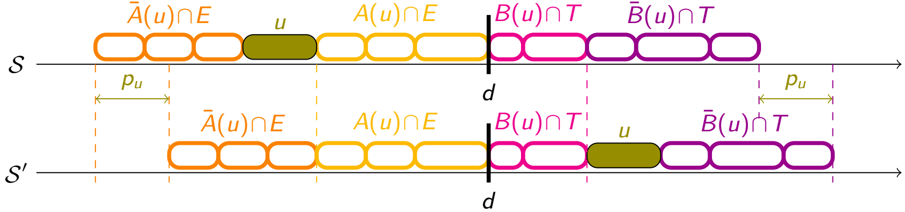
Cost variation induced by the insertion of an early task u



$$\text{cost}(\mathcal{S}') = \text{cost}(\mathcal{S}) - \alpha_u\, p\big(\underline{A(u)\cap E}\big) + \beta_u \Big(p\big(\underline{B(u)\cap T}\big) + p_u\Big)$$

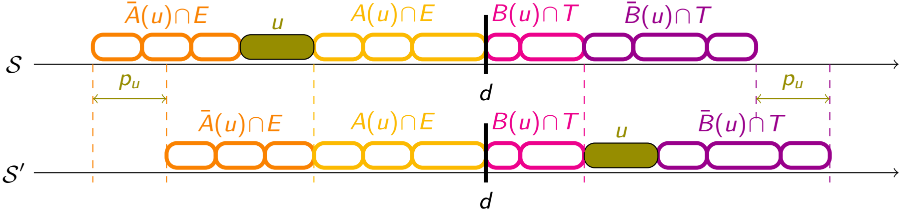
$$- p_u\, \alpha\big(\underline{\bar{A}(u)\cap E}\big) + p_u\, \beta\big(\underline{\bar{B}(v)\cap T}\big)$$

Cost variation induced by the insertion of an early task u



$$\Delta_u^{early}(E,T) = -\alpha_u\, p\big(A(u)\cap E\big) + \beta_u \Big(p\big(B(u)\cap T\big) + p_u\Big)$$
$$- p_u\, \alpha\big(\bar{A}(u)\cap E\big) + p_u\, \beta\big(\bar{B}(v)\cap T\big)$$

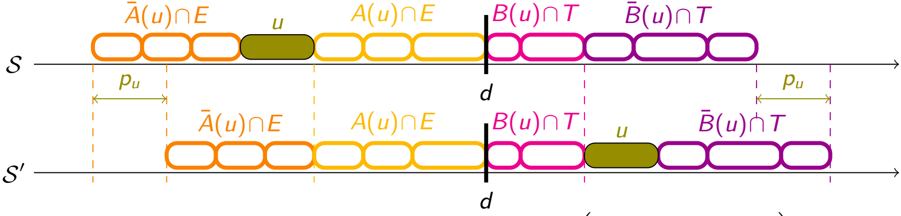
Cost variation induced by the insertion of an early task u



$$\Delta_u^{early}(E,T) = -\alpha_u\, p\big(A(u)\cap E\big) + \beta_u \Big(p\big(B(u)\cap T\big) + p_u\Big) - p_u\, \alpha\big(\bar{A}(u)\cap E\big) + p_u\, \beta\big(\bar{B}(v)\cap T\big)$$

dominance constraint
for the early-insert of $u \in J$ $\boxed{\Delta_u^{early}(E,T) \geqslant 0 \text{ if } u \in E}$

Cost variation induced by the insertion of an early task u

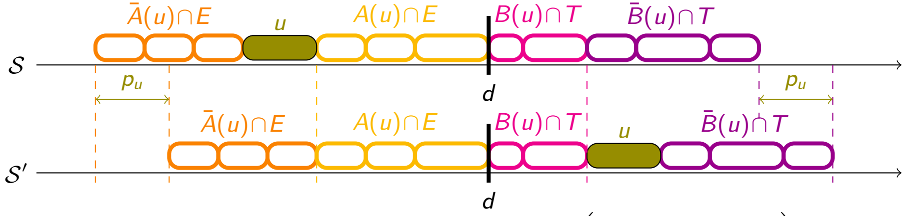


$$\Delta_u^{early}(E,T) = -\alpha_u\, p\big(\underline{A(u)\cap E}\big) + \beta_u \Big(p\big(\underline{B(u)\cap T}\big) + p_u\Big)$$
$$- p_u\, \alpha\big(\underline{\bar{A}(u)\cap E}\big) + p_u\, \beta\big(\underline{\bar{B}(v)\cap T}\big)$$

dominance constraint for the early-insert of $u \in J$ $\boxed{\Delta_u^{early}(E,T) \geqslant 0 \text{ if } u \in E}$

Similarly: dominance constraint for the tardy-insert of $v \in J$ $\boxed{\Delta_v^{tardy}(E,T) \geqslant 0 \text{ if } v \in T}$

Cost variation induced by the insertion of an early task u



$$\Delta_u^{early}(E,T) = -\alpha_u\, p\big(A(u) \cap E\big) + \beta_u \Big(p\big(B(u) \cap T\big) + p_u\Big)$$
$$- p_u\, \alpha\big(\bar{A}(u) \cap E\big) + p_u\, \beta\big(\bar{B}(v) \cap T\big)$$

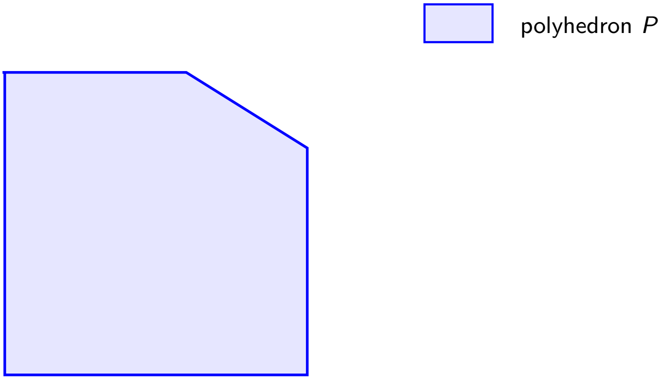
dominance constraint for the early-insert of $u \in J$: $\Delta_u^{early}(E,T) \geqslant 0$ if $u \in E$

Similarly:

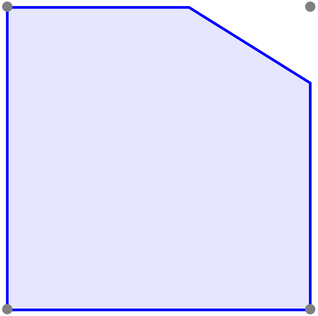
dominance constraint for the tardy-insert of $v \in J$: $\Delta_v^{tardy}(E,T) \geqslant 0$ if $v \in T$

dominance constraint for the swap of $u \in J$ and $v \in J$: $\Delta_{u,v}^{swap}(E,T) \geqslant 0$ if $(u,v) \in E \times T$

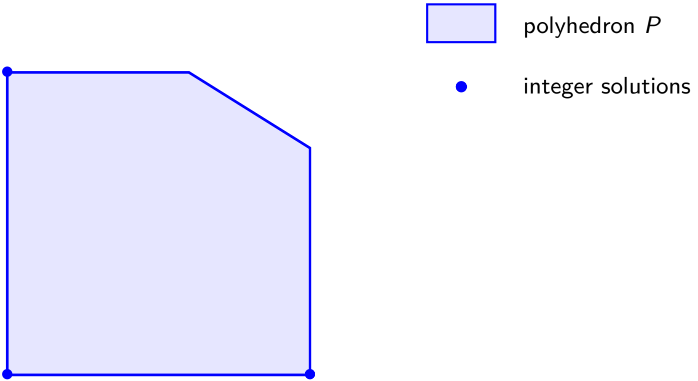
Unusual inequalities



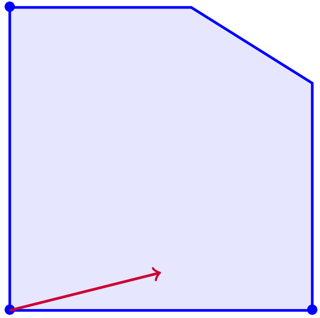
Unusual inequalities

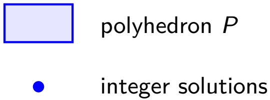


Unusual inequalities

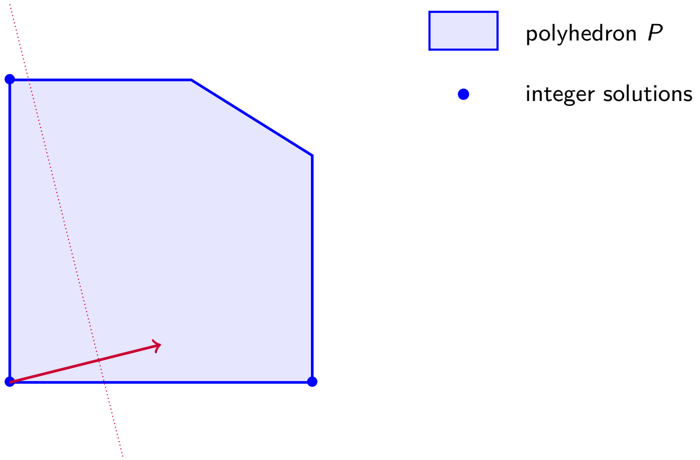


Unusual inequalities

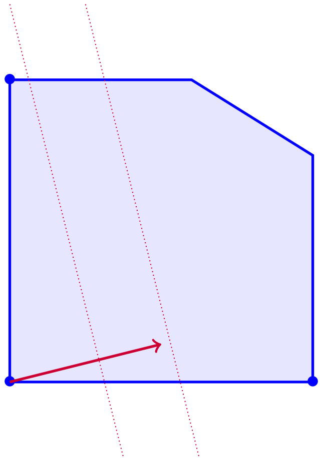


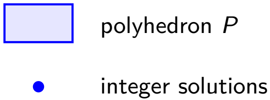


Unusual inequalities

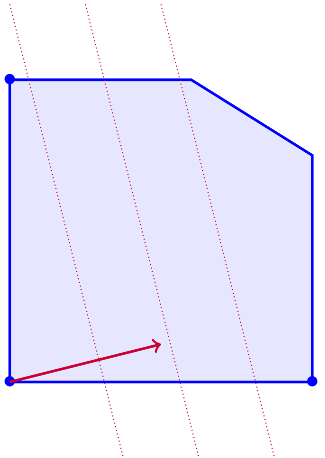


Unusual inequalities





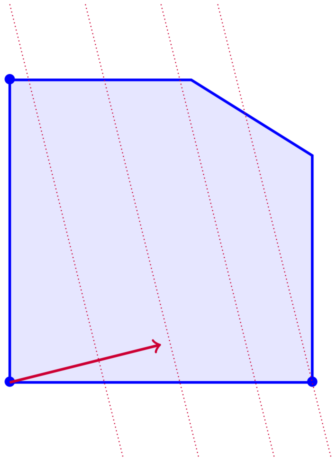
Unusual inequalities

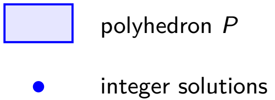


polyhedron P

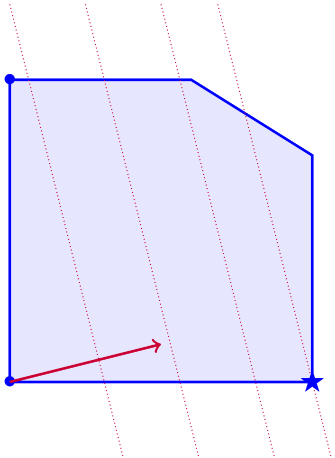
integer solutions

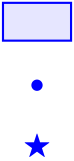
Unusual inequalities



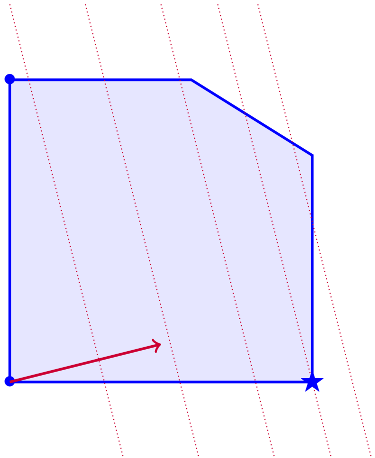


Unusual inequalities





Unusual inequalities

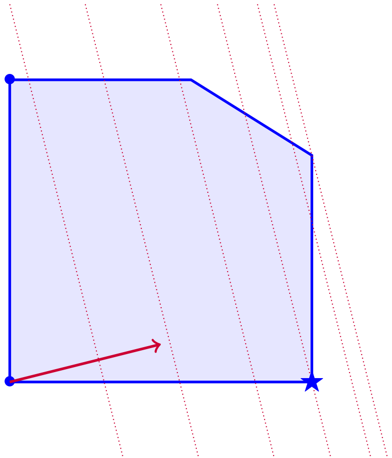


polyhedron P

integer solutions

best integer solution

Unusual inequalities

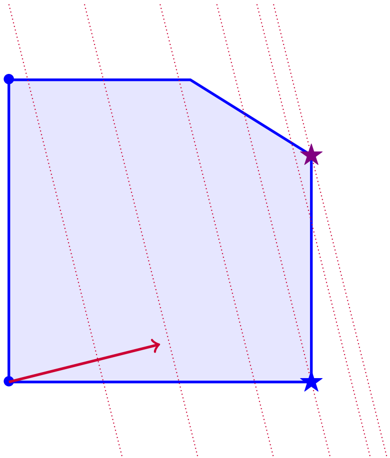


polyhedron P

integer solutions

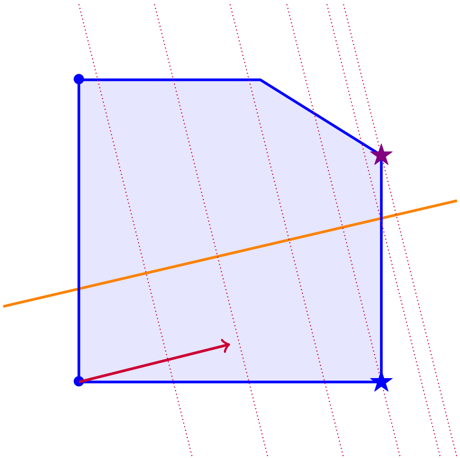
best integer solution

Unusual inequalities



- polyhedron P
- integer solutions
- best integer solution
- best fractionnal solution in P

Unusual inequalities



polyhedron P

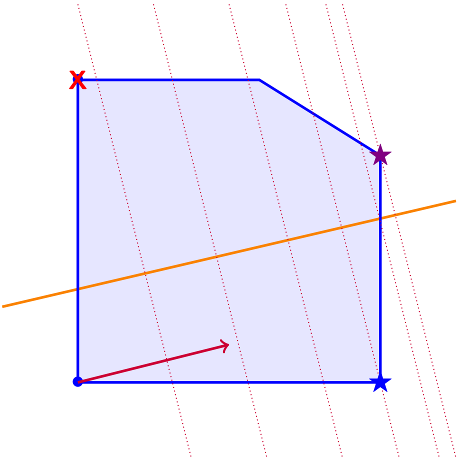
integer solutions

best integer solution

best fractionnal solution in P

a dominance inequality

Unusual inequalities



polyhedron P

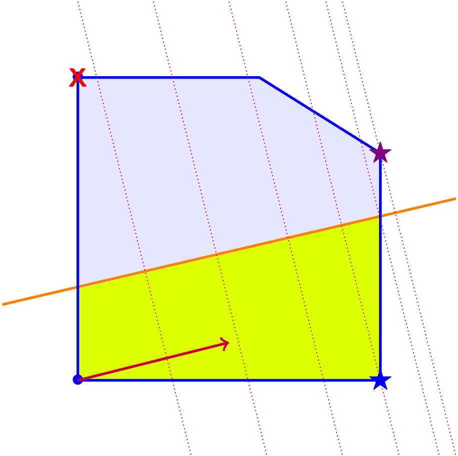
integer solutions

best integer solution

best fractionnal solution in P

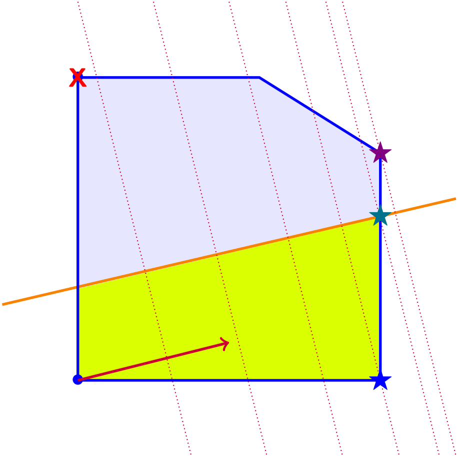
a dominance inequality

Unusual inequalities



- polyhedron P
- integer solutions
- best integer solution
- best fractionnal solution in P
- a dominance inequality
- new polyhedron P'

Unusual inequalities



- polyhedron P
- integer solutions
- best integer solution
- best fractionnal solution in P
- a dominance inequality
- new polyhedron P'
- best fractionnal solution in P'

Conclusion on dominance inequalities

Insert and swap inequalities allow to:

→ reinforce F^3 in a different way as usual strengthen inequalities
no improvement of the linear relaxation value

Conclusion on dominance inequalities

Insert and swap inequalities allow to:

- → reinforce F^3 in a different way as usual strengthen inequalities
  no improvement of the linear relaxation value
- → solve UCDDP up to size 180 within one hour
  (instead of size 60 without them)

Conclusion on dominance inequalities

Insert and swap inequalities allow to:

- → reinforce F^3 in a different way as usual strengthen inequalities
  no improvement of the linear relaxation value
- → solve UCDDP up to size 180 within one hour
  (instead of size 60 without them)
- → illustrate the dominance inequality concept

Outline

1. Introduction
2. Focus 1: A formulation for UCDDP using natural variables
3. Interlude : a first attempt to eliminate dominated solutions
4. Focus 2: dominance inequality
5. Conclusion and perspectives

Done during the thesis

- providing formulation F^3 for UCDDP, F^4 for CDDP
- establishing two lemmas about non-overlapping inequalities
- proving validity of F^3 and F^4
- finding a separation algorithm for non-overlapping inequalities in F^3 and F^4
- providing another formulation F^2 for UCDDP
- implementing and testing F^2, F^3 and F^4
    - ↪ writing a journal paper submitted to DAM, *currently accepted*

- proposing a framework to "transpose" facet defining inequalities
- testing formulation F^2 when known facet defining inequalities are added
- using PORTA on small-dimensional "non-trivial cuts" polytopes
- identifying some family of facet defining inequalities for arbitrary dimension

- proposing a new kind of inequality to improve some MIP formulations solving
- providing dominance inequalities for F^2 based on insert and swap operations
- implementing and testing the impact of these insert and swap inequalities on F^2
- deriving a heuristic algorithm for UCDDP from insert and swap operations
    - ↪ writing a journal paper submitted to EJOR, *currently to review*

+ listening and speaking in seminars or conferences, attending summer school, teaching, writing final report...

Done during the thesis

- providing formulation F^3 for UCDDP, F^4 for CDDP
- establishing two lemmas about non-overlapping inequalities
- proving validity of F^3 and F^4
- finding a separation algorithm for non-overlapping inequalities in F^3 and F^4
- providing another formulation F^2 for UCDDP
- implementing and testing F^2, F^3 and F^4
  - $\hookrightarrow$ writing a journal paper submitted to DAM, *currently accepted*

- proposing a framework to "transpose" facet defining inequalities
- testing formulation F^2 when known facet defining inequalities are added
- using PORTA on small-dimensional "non-trivial cuts" polytopes
- identifying some family of facet defining inequalities for arbitrary dimension

- proposing a new kind of inequality to improve some MIP formulations solving
- providing dominance inequalities for F^2 based on insert and swap operations
- implementing and testing the impact of these insert and swap inequalities on F^2
- deriving a heuristic algorithm for UCDDP from insert and swap operations
  - $\hookrightarrow$ writing a journal paper submitted to EJOR, *currently to review*

\+ listening and speaking in seminars or conferences, attending summer school, teaching, writing final report...

Generic frameworks (tools to take away)

- About natural variables and non-overlapping inequality formulations:
  - → a methodology to formulate some scheduling problem
  - → a scheme of validity proof for such formulations
  - → two key lemmas about non-overlapping inequalities

Generic frameworks (tools to take away)

- About natural variables and non-overlapping inequality formulations:
  - $\rightarrow$ a methodology to formulate some scheduling problem
  - $\rightarrow$ a scheme of validity proof for such formulations
  - $\rightarrow$ two key lemmas about non-overlapping inequalities
- About facet defining inequalities:
  - $\rightarrow$ a ready-to-use property to transpose facet defining inequalities

Generic frameworks (tools to take away)

- About natural variables and non-overlapping inequality formulations:
  - → a methodology to formulate some scheduling problem
  - → a scheme of validity proof for such formulations
  - → two key lemmas about non-overlapping inequalities
- About facet defining inequalities:
  - → a ready-to-use property to transpose facet defining inequalities
- About dominance inequalities:
  - → a theoretical framework
  - → a recipe to obtain a dominance inequality from a given operation

Perspectives

About dominance inequalities:

→ How do insert and swap inequalities improve formulation F^2?

→ Can we provide dominance inequalities useful for other combinatorial problems?

The end

Thank you for your attention