Formuler un problème d'ordonnancement juste-à-temps grâce à des inégalités de non-chevauchement

Anne-Elisabeth FALQ

encadrée par Pierre Fouilhoux et Safia Kedad-Sidhoum

18 Novembre 2020, en ligne avec le GSCOP de Grenoble

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

Outline

1. Introduction
   Scheduling around a common due date
   Known results about UCDDP and CDDP
   How to encode schedules?

2. A formulation for UCDDP using natural variables

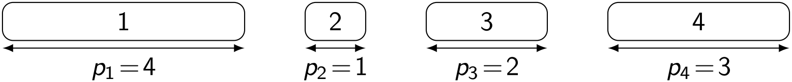
3. How to manage this kind of formulations in practice

4. How to extend this formulation

5. Conclusion

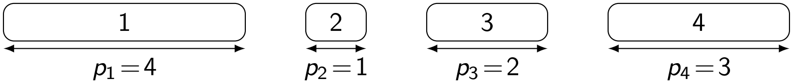
Scheduling around a common due-date on a single machine

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

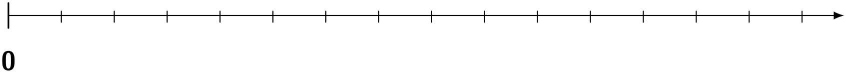


Scheduling around a common due-date on a single machine

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

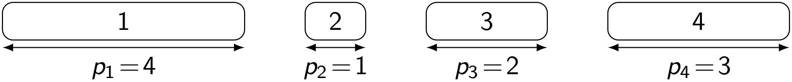


A schedule :

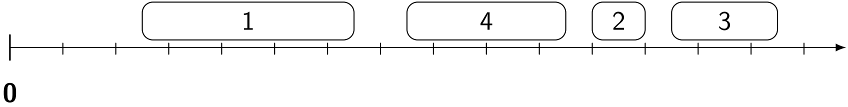


Scheduling around a common due-date on a single machine

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

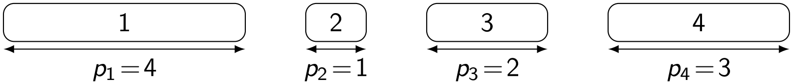


A schedule :



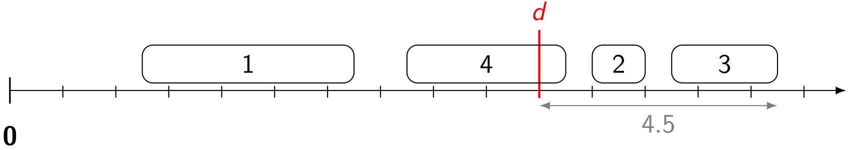
Scheduling around a common due-date on a single machine

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



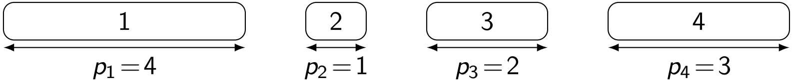
• common due date : $d = 10$

A schedule :



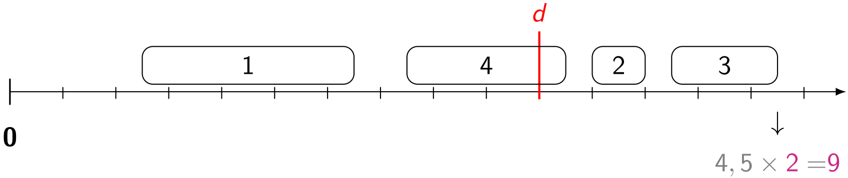
Scheduling around a common due-date on a single machine

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



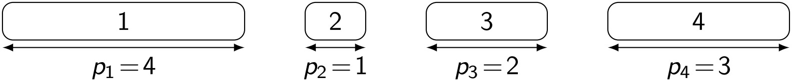
- common due date : $d = 10$
- unit tardiness penalties : $\beta_1 = \beta_4 = 5$, $\beta_2 = \beta_3 = 2$

A schedule :



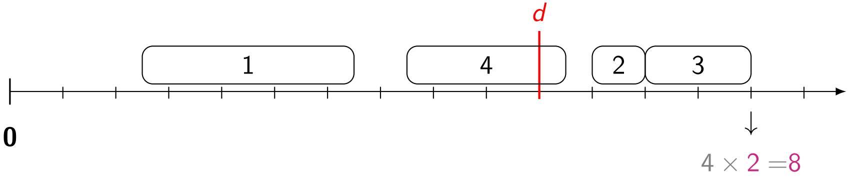
Scheduling around a common due-date on a single machine

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



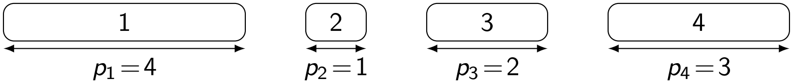
- common due date : $d = 10$
- unit tardiness penalties : $\beta_1 = \beta_4 = 5$, $\beta_2 = \beta_3 = 2$

A schedule :



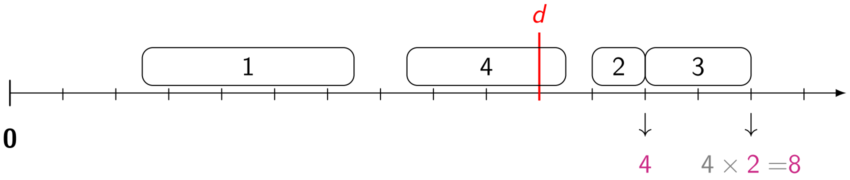
Scheduling around a common due-date on a single machine

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



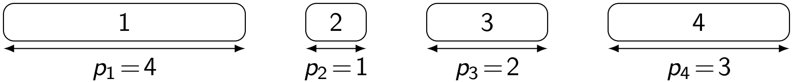
- common due date : $d = 10$
- unit tardiness penalties : $\beta_1 = \beta_4 = 5$, $\beta_2 = \beta_3 = 2$

A schedule :



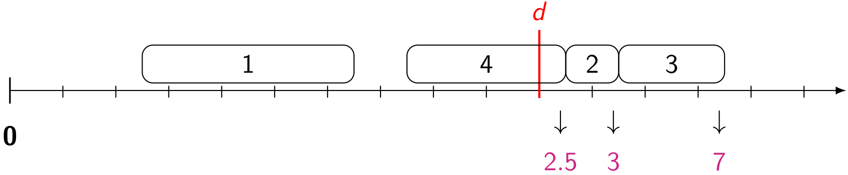
Scheduling around a common due-date on a single machine

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



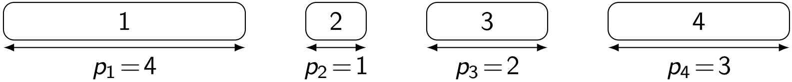
- common due date : $d = 10$
- unit tardiness penalties : $\beta_1 = \beta_4 = 5$, $\beta_2 = \beta_3 = 2$

A schedule :



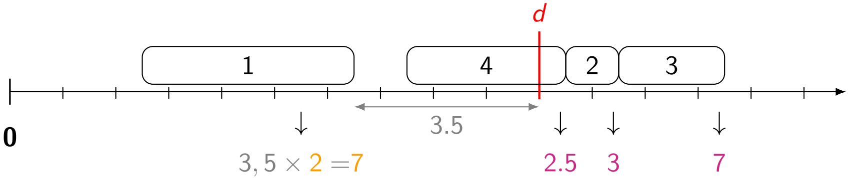
Scheduling around a common due-date on a single machine

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



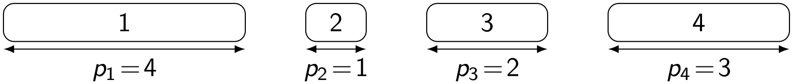
- common due date : $d = 10$
- unit tardiness penalties : $\beta_1 = \beta_4 = 5$, $\beta_2 = \beta_3 = 2$
- unit earliness penalties : $\forall j \in J$, $\alpha_j = 2$

A schedule :



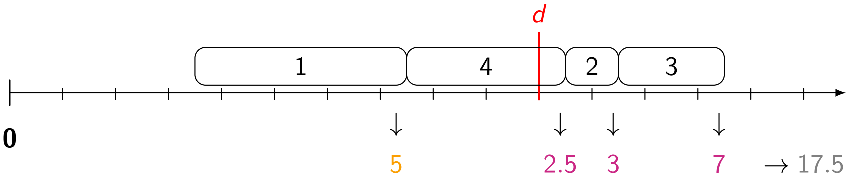
Scheduling around a common due-date on a single machine

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



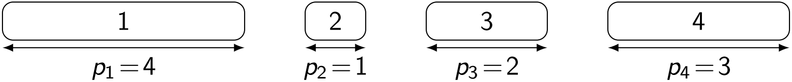
- common due date : $d=10$
- unit tardiness penalties : $\beta_1=\beta_4=5$, $\beta_2=\beta_3=2$
- unit earliness penalties : $\forall j\in J$, $\alpha_j=2$

A schedule :



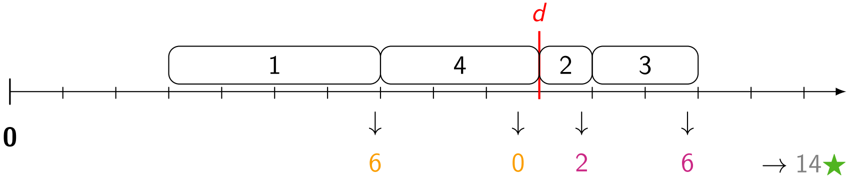
Scheduling around a common due-date on a single machine

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



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

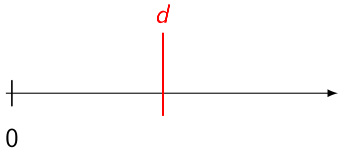
An instance =

- 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 =

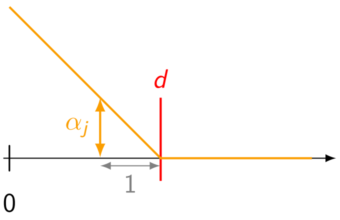
- a set of tasks J
- their processing times $(p_j)_{j \in J} \in \mathbb{N}^J$
- an **unrestrictive** common due-date $d \geqslant \sum_{j \in J} p_j$



The Unrestrictive Common Due Date Problem (UCDDP)

An instance =

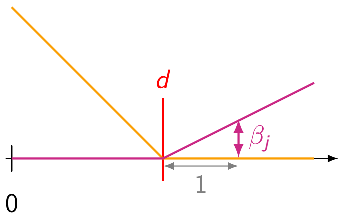
- a set of tasks J
- their processing times $(p_j)_{j \in J} \in \mathbb{N}^J$
- an **unrestrictive** common due-date $d \geqslant \sum_{j \in J} p_j$
- their unit earliness penalties $(\alpha_j)_{j \in J}$



The Unrestrictive Common Due Date Problem (UCDDP)

An instance =

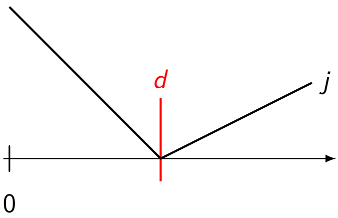
- a set of tasks J
- their processing times $(p_j)_{j \in J} \in \mathbb{N}^J$
- an **unrestrictive** common due-date $d \geqslant \sum_{j \in J} p_j$
- their unit earliness penalties $(\alpha_j)_{j \in J}$
- their unit tardiness penalties $(\beta_j)_{j \in J}$



The Unrestrictive Common Due Date Problem (UCDDP)

An instance =

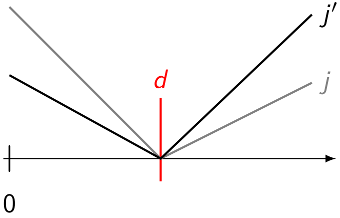
- a set of tasks J
- their processing times $(p_j)_{j \in J} \in \mathbb{N}^J$
- an **unrestrictive** common due-date $d \geqslant \sum_{j \in J} p_j$
- their unit earliness penalties $(\alpha_j)_{j \in J}$
- their unit tardiness penalties $(\beta_j)_{j \in J}$



The Unrestrictive Common Due Date Problem (UCDDP)

An instance =

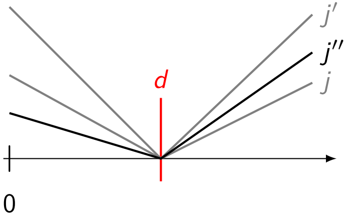
- a set of tasks J
- their processing times $(p_j)_{j\in J} \in \mathbb{N}^J$
- an **unrestrictive** common due-date $d \geqslant \sum_{j\in J} p_j$
- their unit earliness penalties $(\alpha_j)_{j\in J}$
- their unit tardiness penalties $(\beta_j)_{j\in J}$



The Unrestrictive Common Due Date Problem (UCDDP)

An instance =

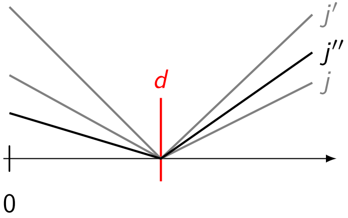
- a set of tasks J
- their processing times $(p_j)_{j \in J} \in \mathbb{N}^J$
- an **unrestrictive** common due-date $d \geqslant \sum_{j \in J} p_j$
- their unit earliness penalties $(\alpha_j)_{j \in J}$
- their unit tardiness penalties $(\beta_j)_{j \in J}$



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 $(\beta_j)_{j \in J}$

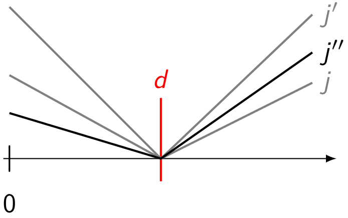


A solution schedule = a family of pairwise disjoint processing intervals

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 $(\beta_j)_{j \in J}$



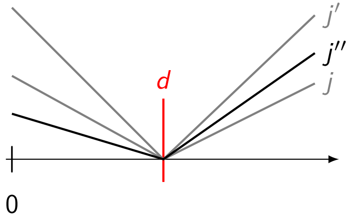
A solution schedule = a family of pairwise disjoint processing intervals

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

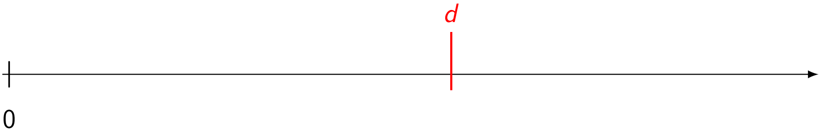
Kanet, 1981, Naval Research Logistics Quaterly
Hall and Posner, 1991, Operations research
Hoogeveen and van de Velde, 1991, European Journal of Operational research
Sourd, 2009, Informs Journal on Computing

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

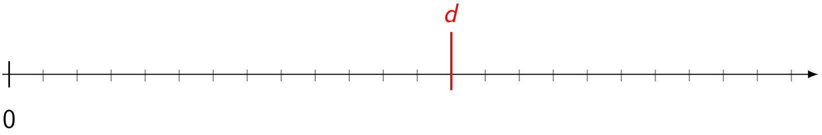
Kanet, 1981, Naval Research Logistics Quaterly
Hall and Posner, 1991, Operations research
Hoogeveen and van de Velde, 1991, European Journal of Operational research
Sourd, 2009, Informs Journal on Computing

Time-indexed variables



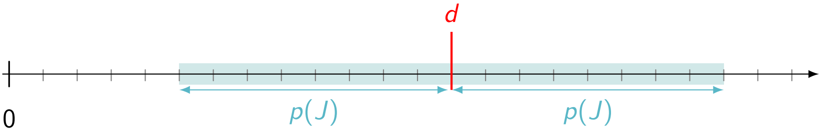
First let us discretize the time horizon .

Time-indexed variables



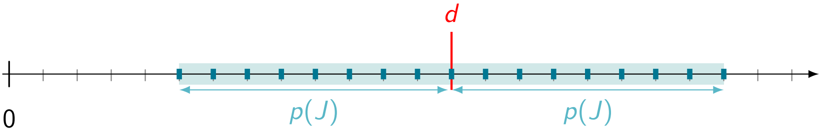
First let us discretize the time horizon .

Time-indexed variables



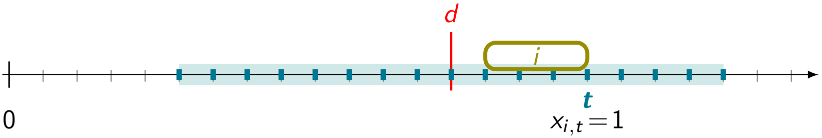
First let us discretize the time horizon .

Time-indexed variables



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

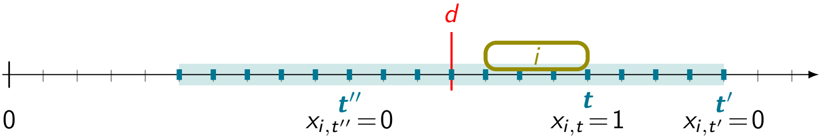
Time-indexed variables



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

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

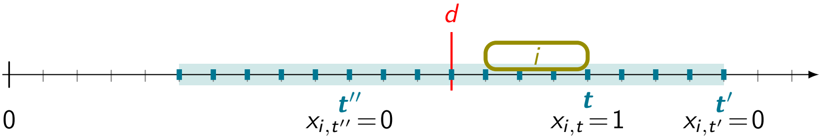
Time-indexed variables



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

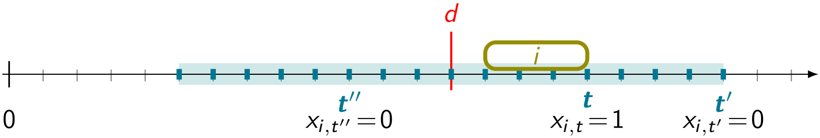


First let us discretize the time horizon as $\mathcal{T} = [d - p(J), d + p(J)]$.

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 $\mathcal{T}=[d-p(J), d+p(J)]$.

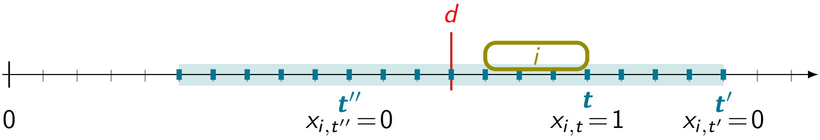
Variables: $\forall i \in J,\ \forall t \in \mathcal{T}$: $x_{i,t} = \begin{cases} 1 \text{ if } i \text{ completes at } t \\ 0 \text{ otherwise} \end{cases}$

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



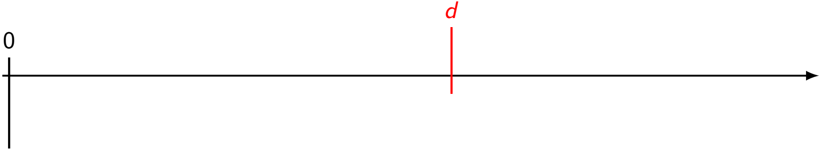
First let us discretize the time horizon as $\mathcal{T} = [d - p(J), d + p(J)]$.

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

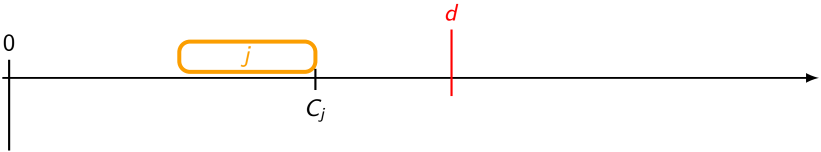
- $+$ easy to formulate as a **MIP**
- $+$ good relaxation value
- $-$ $2n\, p(J)$ binary variables $=$ a pseudo polynomial number
- $-$ $n + n\, p(J)$ inequalities $=$ a pseudo polynomial number

Completion time variables



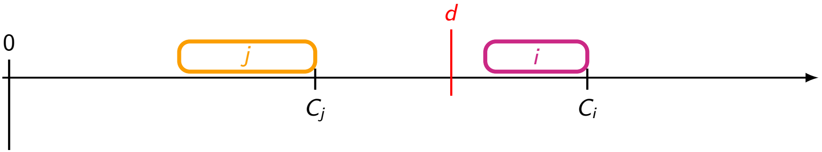
Variables: $\forall j \in J,\ C_j \in \mathbb{R}_+$ is the time when task j completes

Completion time variables



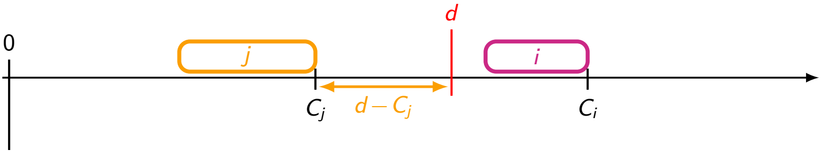
Variables: $\forall j \in J,\ C_j \in \mathbb{R}_+$ is the time when task j completes

Completion time variables



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

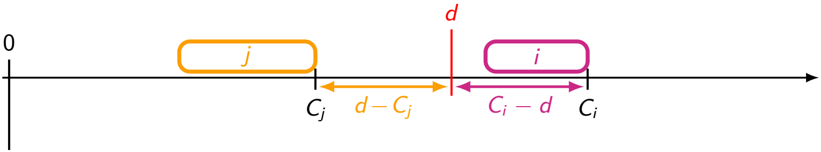
Completion time variables



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

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

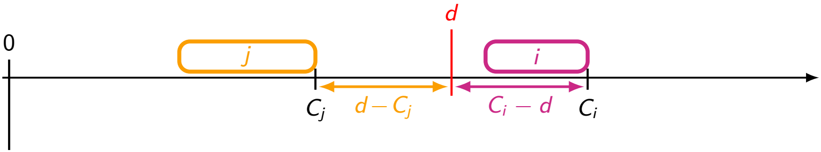
Completion time variables



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

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

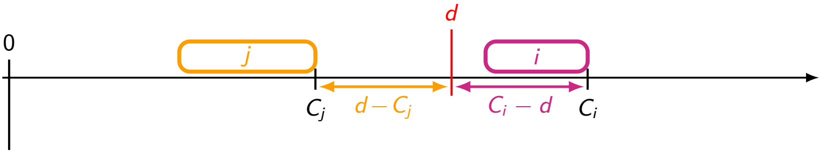
Completion time variables



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

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

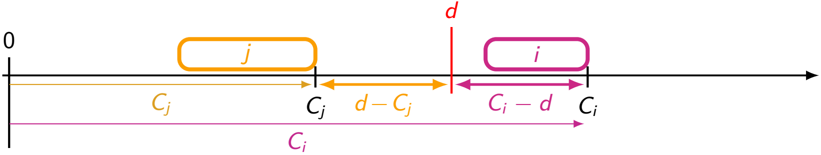
Completion time variables



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!

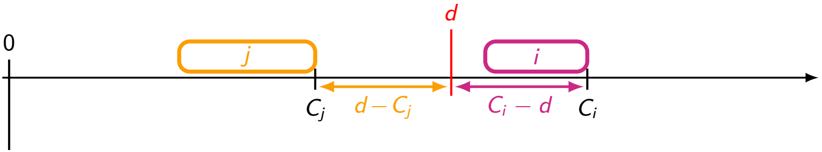
Completion time variables



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!

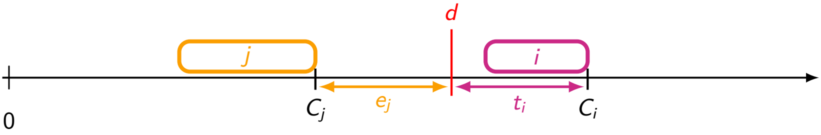
Completion time variables



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

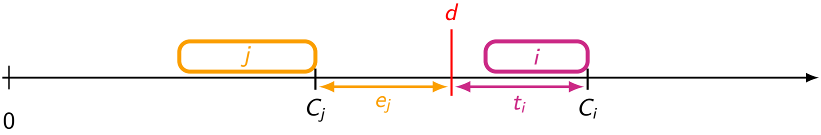
Objective function: $\sum_{j \in J} \alpha_j \, [d - C_j]^+ + \beta_j \, [C_j - d]^+$ not linear!

Earliness–Tardiness variables



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

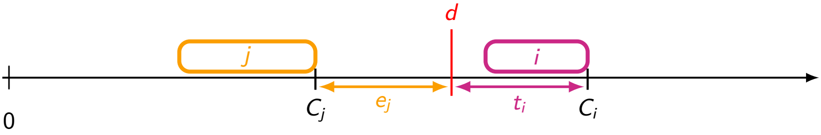
Earliness–Tardiness variables



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

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

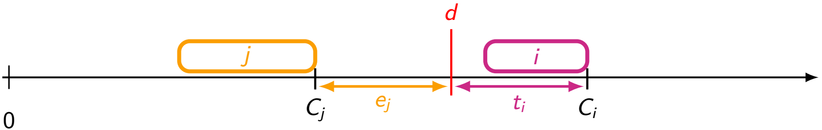
Earliness–Tardiness variables



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*

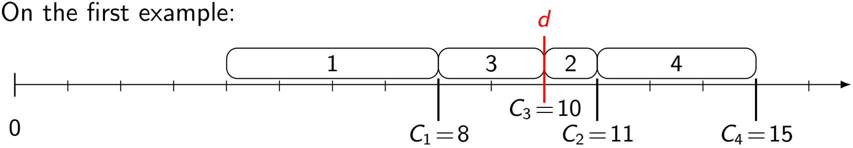
Earliness–Tardiness variables



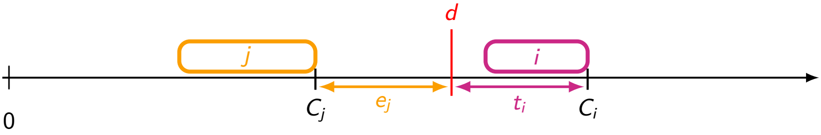
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

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:



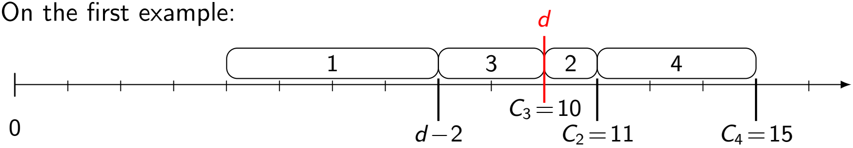
Earliness–Tardiness variables



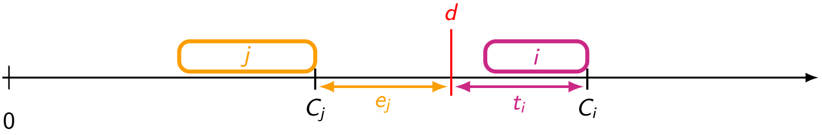
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

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

On the first example:



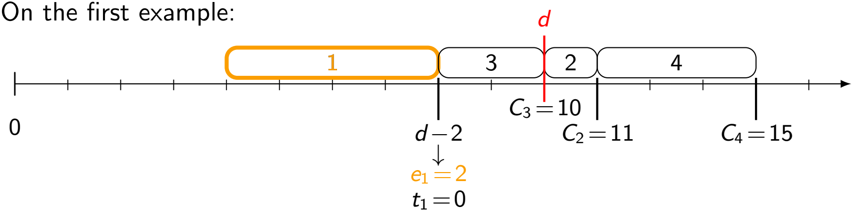
Earliness–Tardiness variables



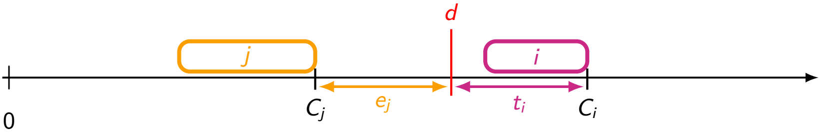
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

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:



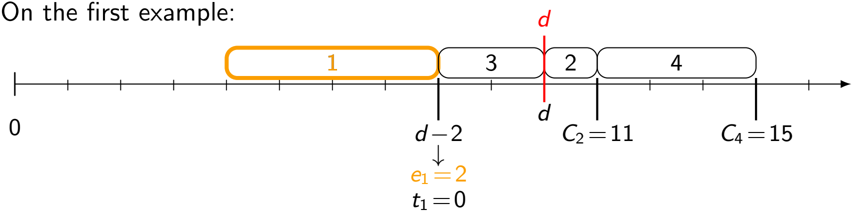
Earliness–Tardiness variables



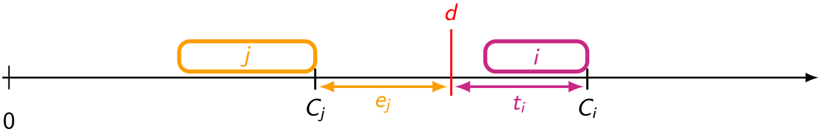
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

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:



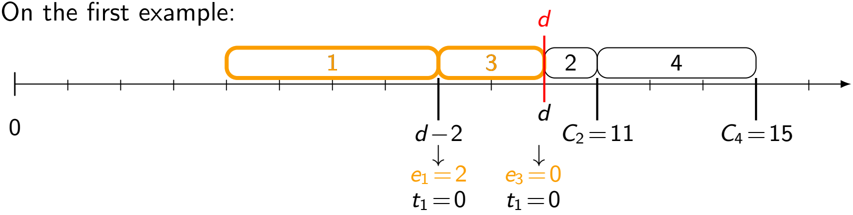
Earliness–Tardiness variables



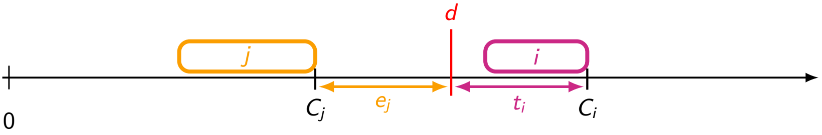
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

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:



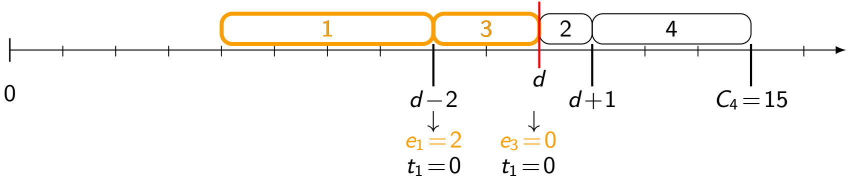
Earliness–Tardiness variables



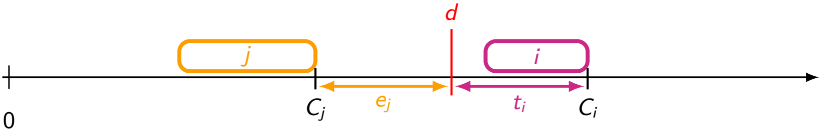
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

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:



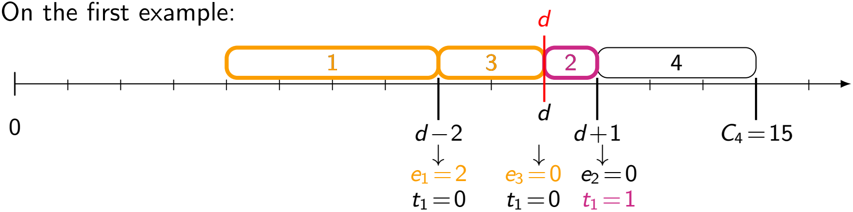
Earliness–Tardiness variables



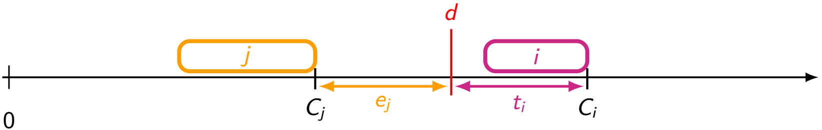
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

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:



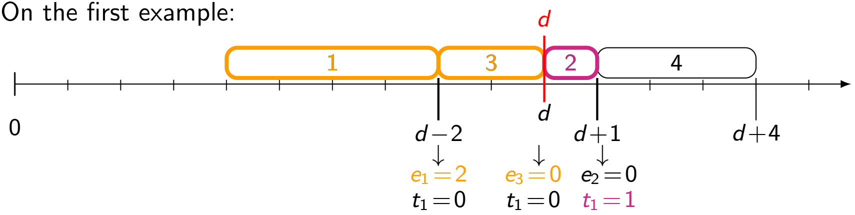
Earliness–Tardiness variables



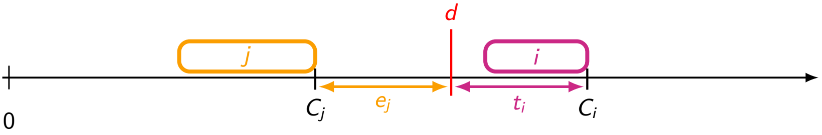
Variables: $\forall j \in J,\ e_j = [d - C_j]^+$ and $t_j = [C_j - d]^+$

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:



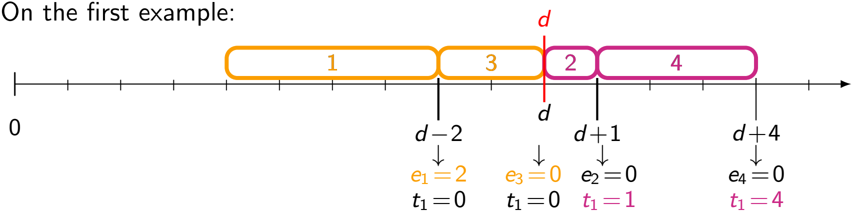
Earliness–Tardiness variables



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:



Outline

1. Introduction

2. A formulation for UCDDP using natural variables
   - Describing the solution set for (e, t) variables
   - Non-overlapping inequalities
   - Validity

3. How to manage this kind of formulations in practice

4. How to extend this formulation

5. Conclusion

What is the goal?

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

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

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

What is the goal?

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

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

What is the goal?

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

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

What is the goal?

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

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

What is the goal?

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

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

What is the goal?

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

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

$\rightarrow$ $\mathscr{S}$ 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: $\boxed{\text{UCDDP} \iff \min\limits_{(e,t)\in\text{int}(\text{extr}\, P)} g_{\alpha,\beta}(e,t)}$

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 $\delta_j = \begin{cases} 1 \text{ if } j \text{ is early} \\ 0 \text{ if } j \text{ is tardy} \end{cases}$

$$\begin{aligned} \forall j \in J,\ e_j &\geqslant 0 && (e.0) \\ e_j &\leqslant M\,\delta_j && (e.1) \end{aligned} \qquad \begin{aligned} \forall j \in J,\ t_j &\geqslant 0 && (t.1) \\ t_j &\leqslant M\,(1-\delta_j) && (t.2) \end{aligned} \qquad \text{where } \boldsymbol{M} = \sum_{j\in J} p_j$$

[**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 $\delta_j = \begin{cases} 1 \text{ if } j \text{ is early} \\ 0 \text{ if } j \text{ is tardy} \end{cases}$

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

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 $\delta_j = \begin{cases} 1 \text{ if } j \text{ is early} \\ 0 \text{ if } j \text{ is tardy} \end{cases}$

$$\forall j \in J,\ e_j \geqslant 0 \quad (e.0) \qquad \forall j \in J,\ t_j \geqslant 0 \quad (t.1)$$

$$e_j \leqslant M\,\delta_j \quad (e.1) \qquad t_j \leqslant M\,(1-\delta_j) \quad (t.2) \qquad \text{where } \boldsymbol{M} = \sum_{j\in J} p_j$$

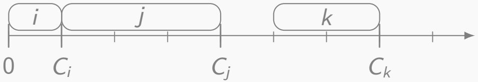
[**non-overlapping**] processing intervals are pairwise disjoints

- 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

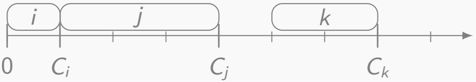


$$\forall S \subseteq J, \sum_{j \in S} p_j C_j \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

- scheduling problem without due-date

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

Queyranne's non-overlapping inequalities

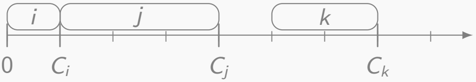


$$\forall S \subseteq J, \sum_{j \in S} p_j C_j \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

- 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

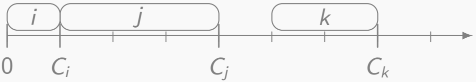


$$\forall S \subseteq J, \sum_{j \in S} p_j C_j \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

- 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 $1 \,|\, - \,|\, \min \sum \omega_j C_j$

Queyranne's non-overlapping inequalities

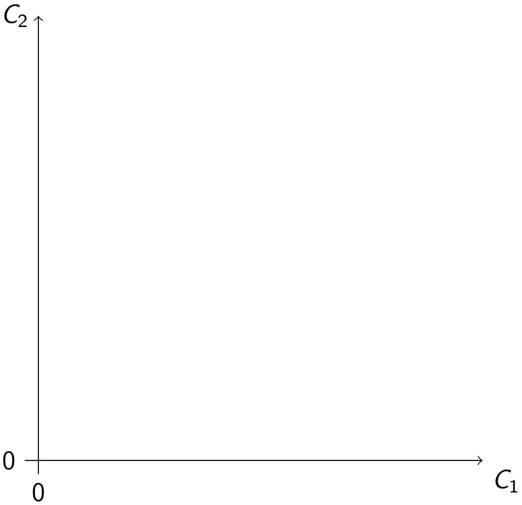


$$\forall S \subseteq J, \sum_{j \in S} p_j C_j \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

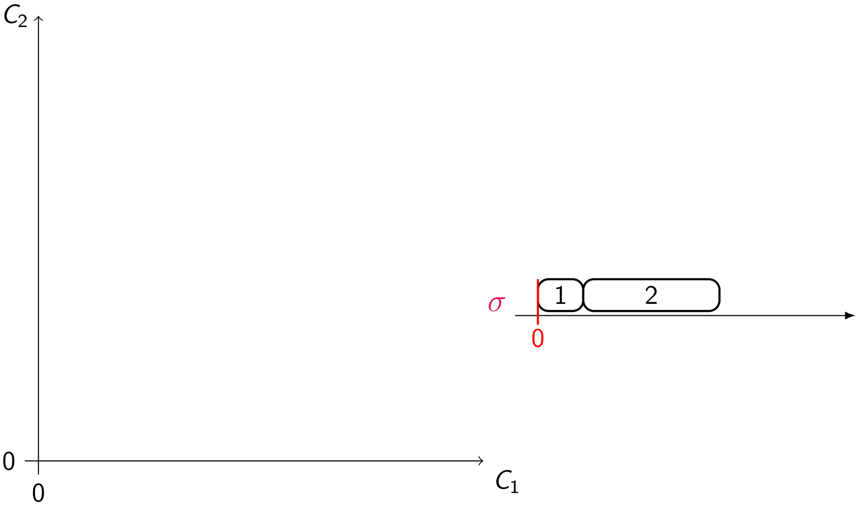
- scheduling problem without due-date
- encoding schedules by their completion times $(C_j)_{j \in J}$
- these inequalities describe the convex hull of such vectors C
- all extreme points of the polyhedron encode feasible schedules

Queyranne, 1993, Mathematical Programming

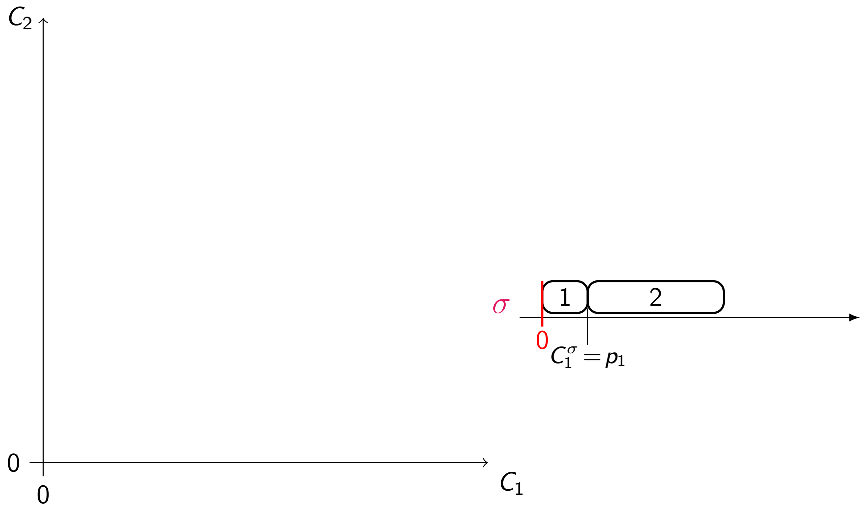
The set of vectors (C_1, C_2) encoding a 2-task schedule



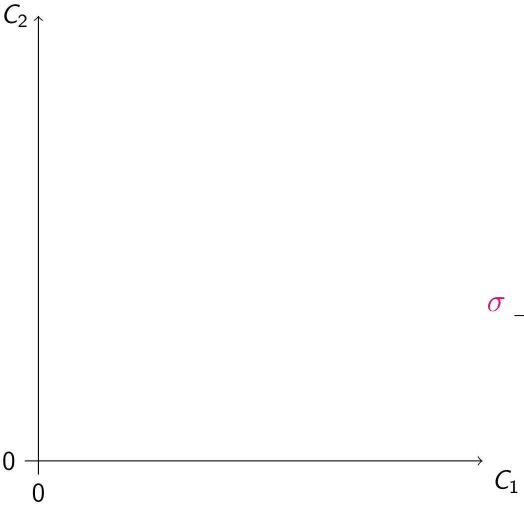
The set of vectors (C_1, C_2) encoding a 2-task schedule

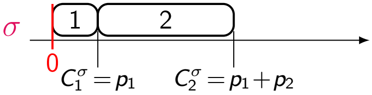


The set of vectors (C_1, C_2) encoding a 2-task schedule



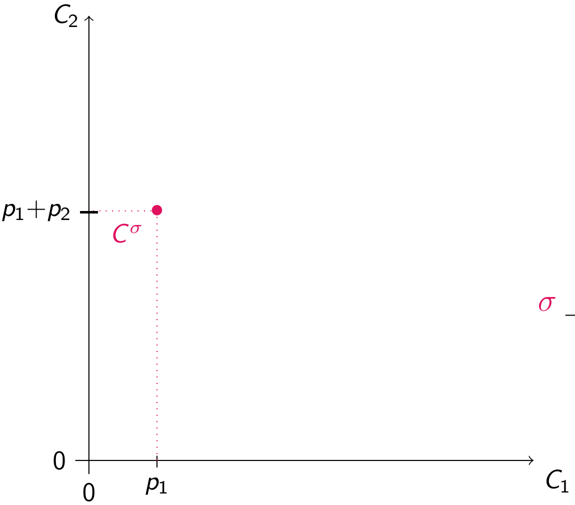
The set of vectors (C_1, C_2) encoding a 2-task schedule

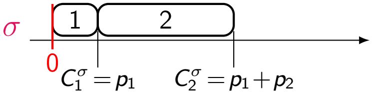




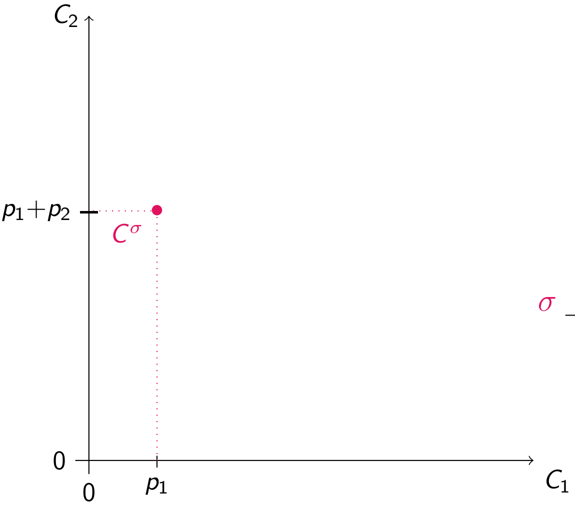
Queyranne, 1993, Mathematical Programming

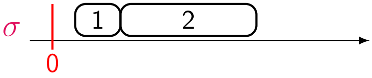
The set of vectors (C_1, C_2) encoding a 2-task schedule





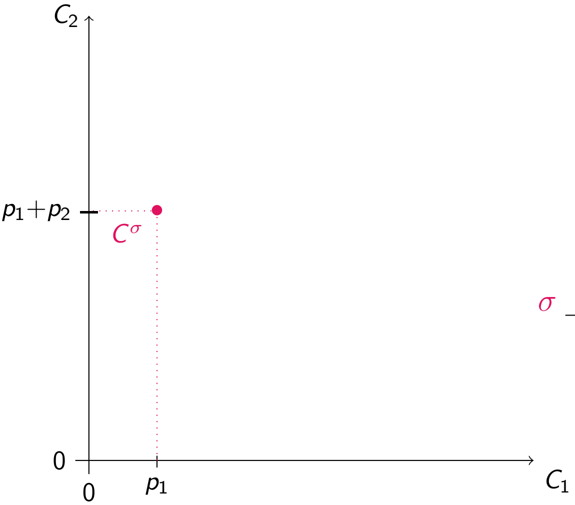
The set of vectors (C_1, C_2) encoding a 2-task schedule

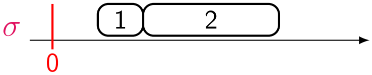




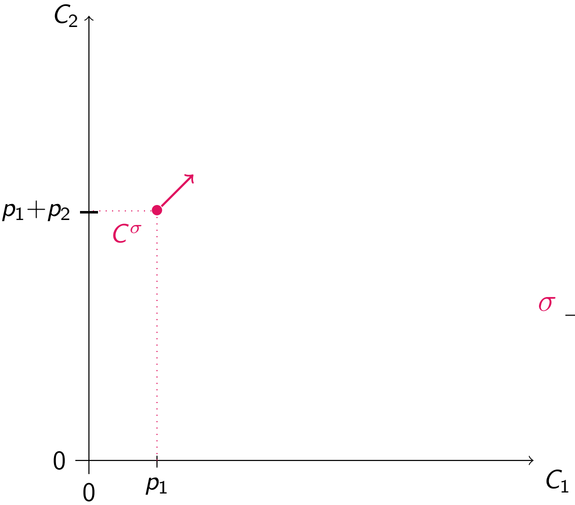
Queyranne, 1993, Mathematical Programming

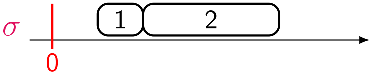
The set of vectors (C_1, C_2) encoding a 2-task schedule



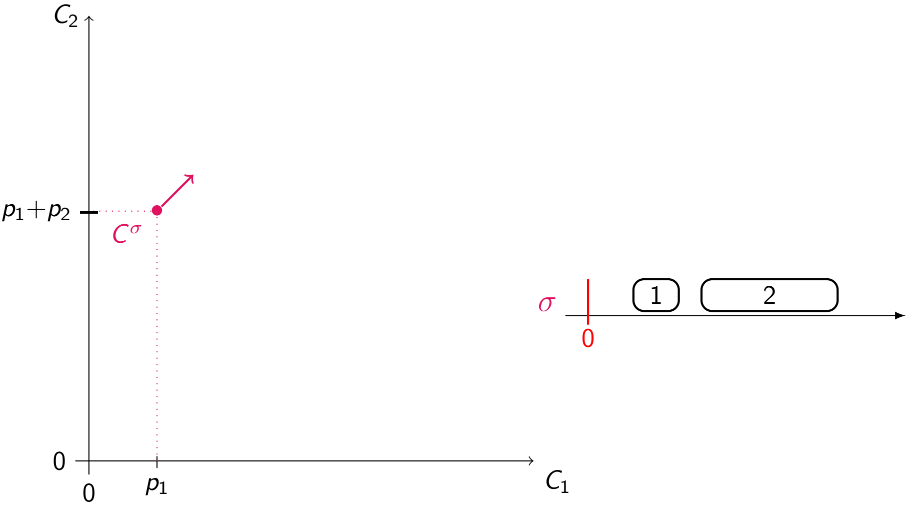


The set of vectors (C_1, C_2) encoding a 2-task schedule

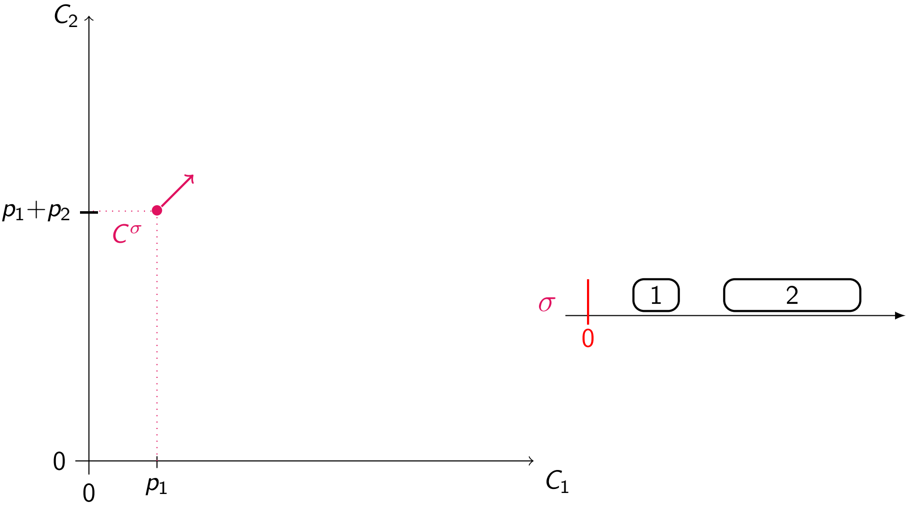




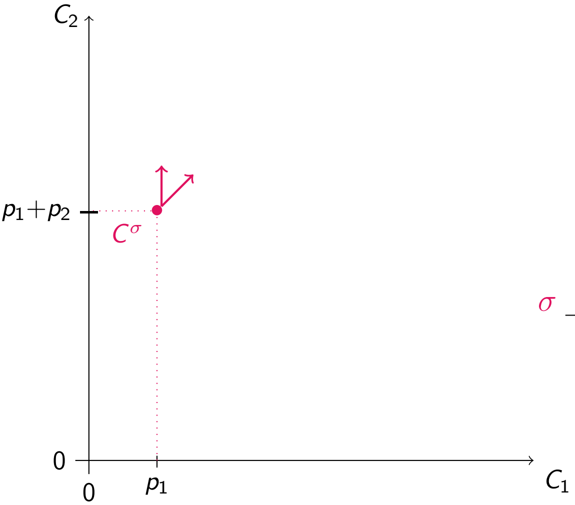
The set of vectors (C_1, C_2) encoding a 2-task schedule



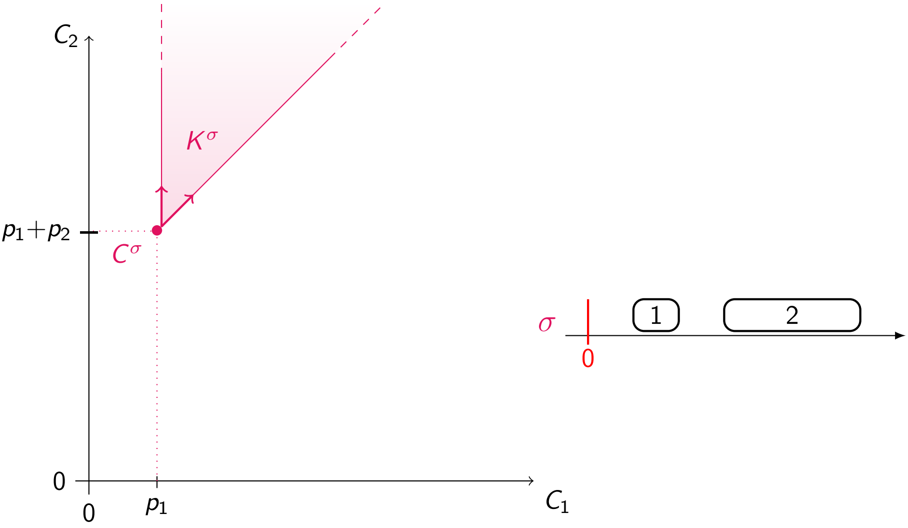
The set of vectors (C_1, C_2) encoding a 2-task schedule



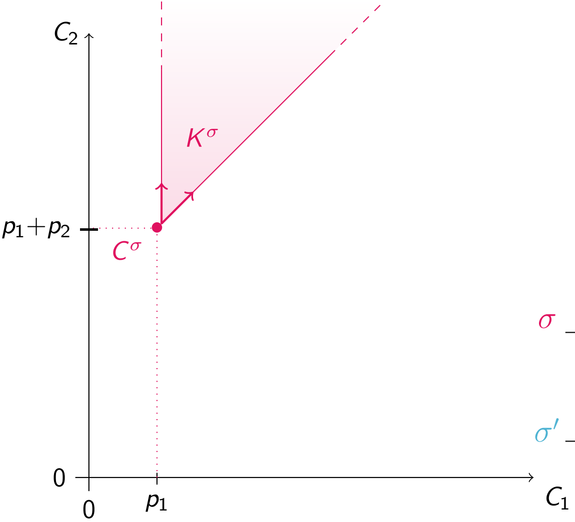
The set of vectors (C_1, C_2) encoding a 2-task schedule

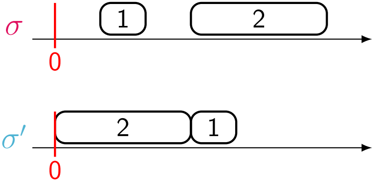


The set of vectors (C_1, C_2) encoding a 2-task schedule

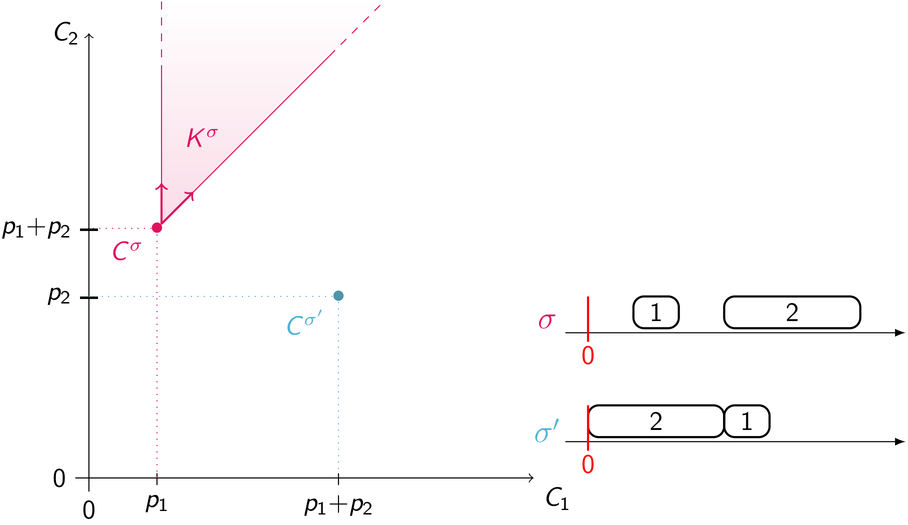


The set of vectors (C_1, C_2) encoding a 2-task schedule

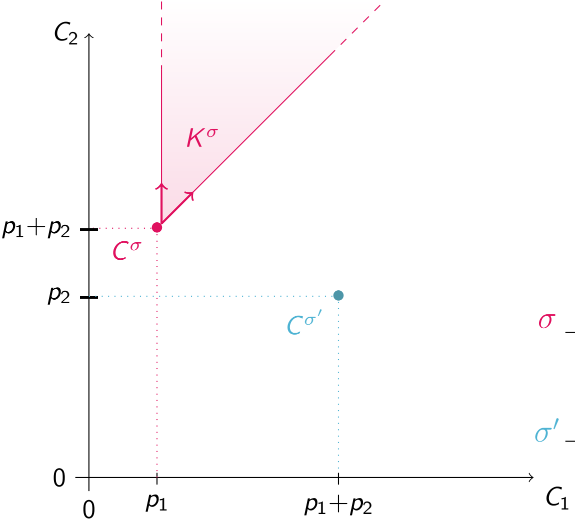


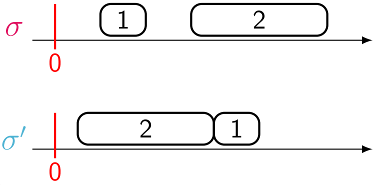


The set of vectors (C_1, C_2) encoding a 2-task schedule



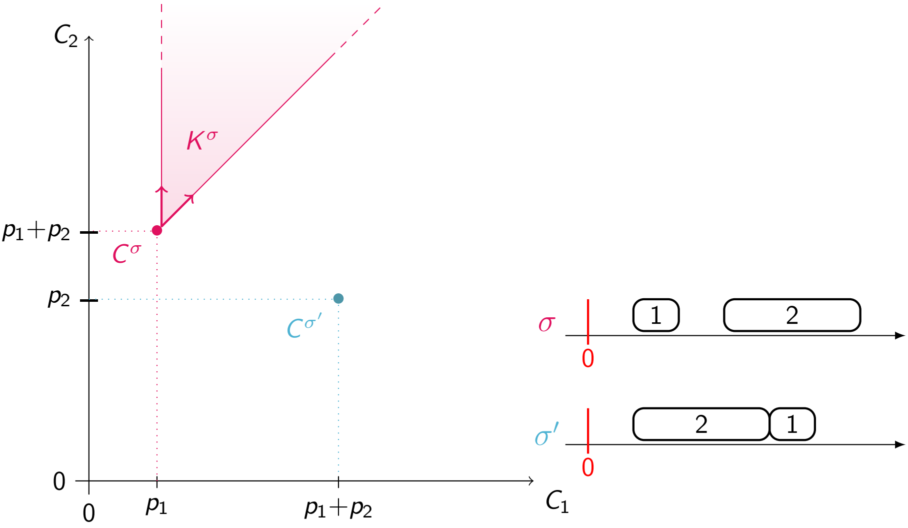
The set of vectors (C_1, C_2) encoding a 2-task schedule





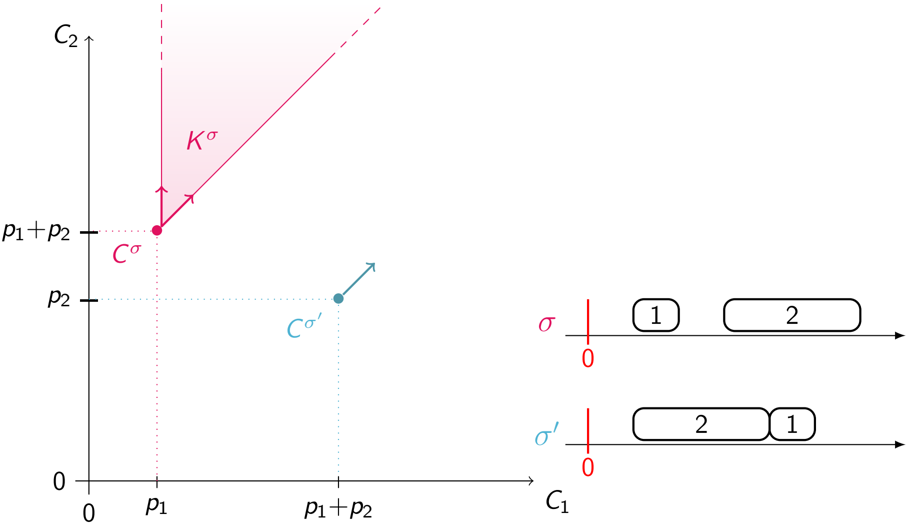
Queyranne, 1993, Mathematical Programming

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

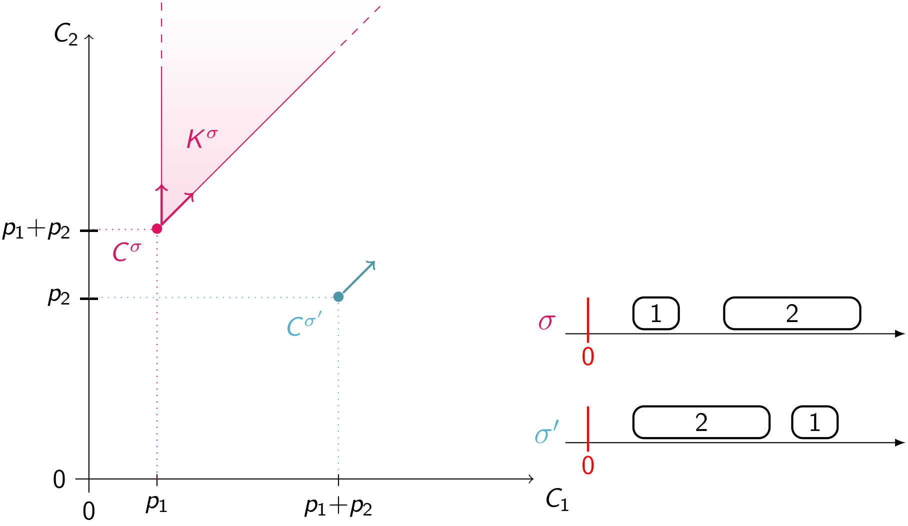


Queyranne, 1993, Mathematical Programming

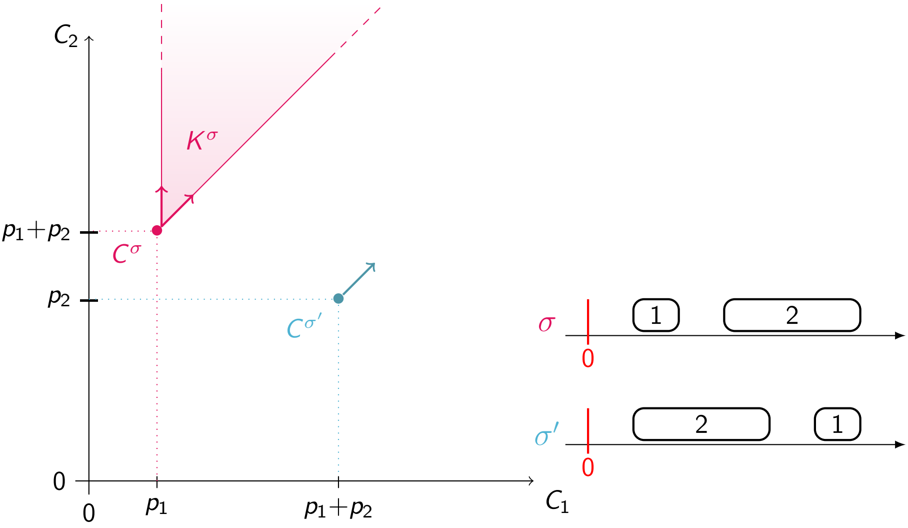
The set of vectors (C_1, C_2) encoding a 2-task schedule



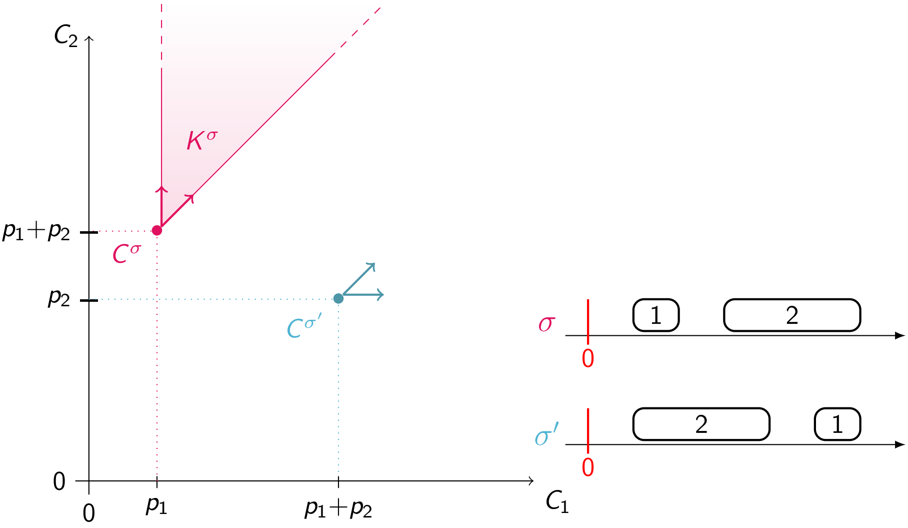
The set of vectors (C_1, C_2) encoding a 2-task schedule



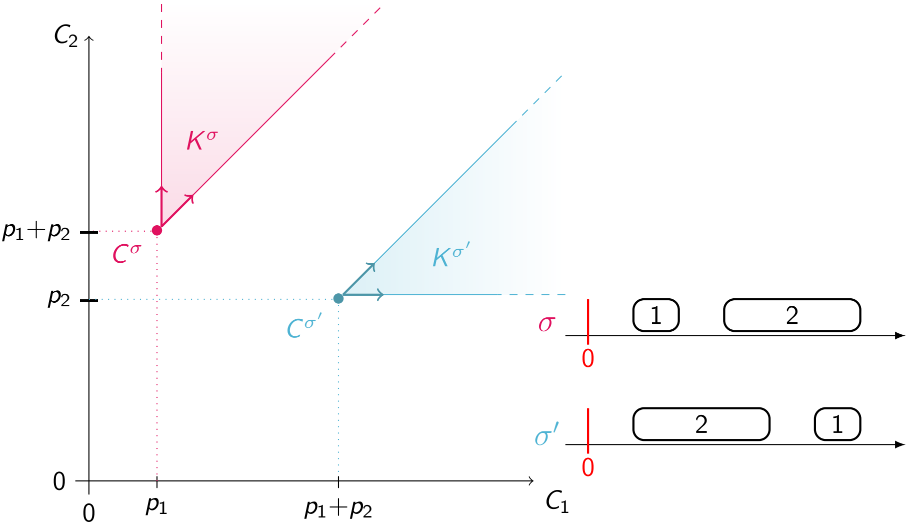
The set of vectors (C_1, C_2) encoding a 2-task schedule



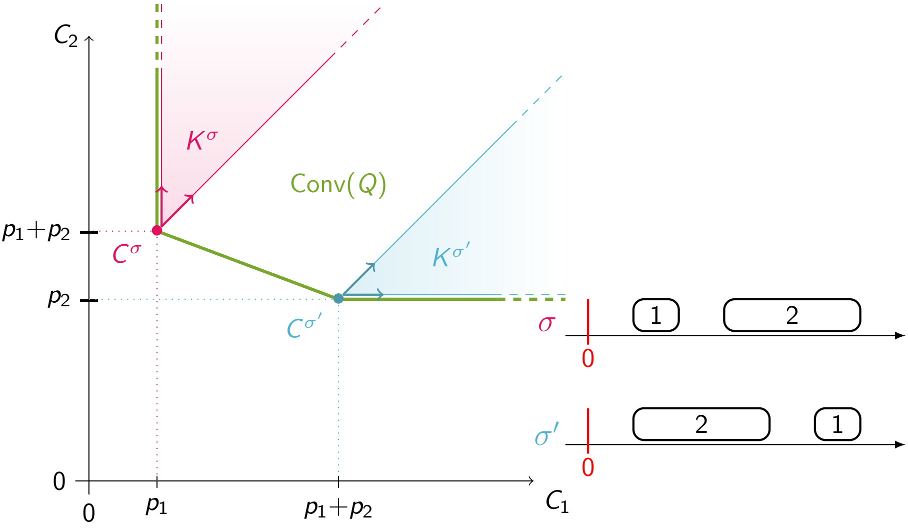
The set of vectors (C_1, C_2) encoding a 2-task schedule



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

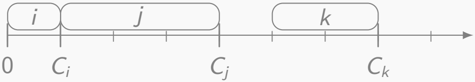


The set of vectors (C_1, C_2) encoding a 2-task schedule



Non-overlapping inequalities for UCDDP

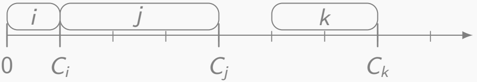
Queyranne's non-overlapping inequalities



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

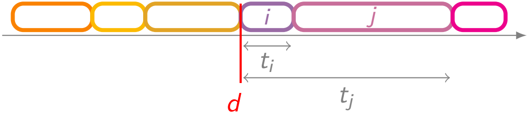
Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



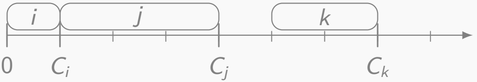
$$\forall S \subseteq J,\ \underbrace{\sum_{i \in S} p_i C_i}_{p * C(S)} \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j \in S} p_j^2 + \big(\sum_{j \in S} p_j\big)^2\Big]$$

A first idea : {



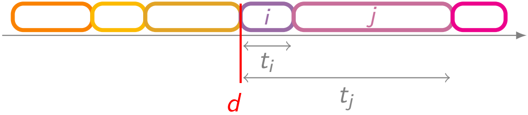
Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



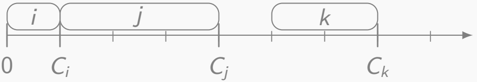
$$\forall S \subseteq J,\ \underbrace{\sum_{i\in S} p_i C_i}_{p*C(S)} \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j\in S} p_j^2 + \big(\sum_{j\in S} p_j\big)^2\Big]$$

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



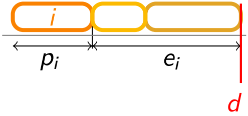
Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



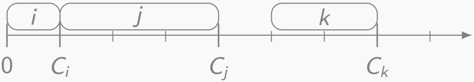
$$\forall S \subseteq J,\ \underbrace{\textstyle\sum_{i\in S} p_i C_i}_{p*C(S)} \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j\in S} p_j^2 + \big(\sum_{j\in S} p_j\big)^2\Big]$$

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



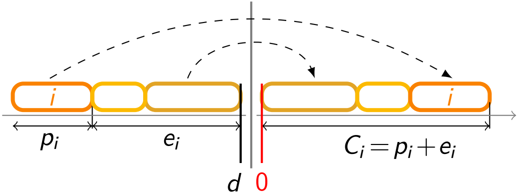
Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



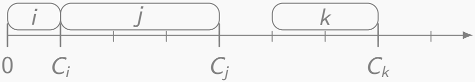
$$\forall S \subseteq J,\ \underbrace{\sum_{i\in S} p_i C_i}_{p*C(S)} \geqslant g(S) \quad \text{where } g(S) = \frac{1}{2}\Big[\sum_{j\in S} p_j^2 + \big(\sum_{j\in S} p_j\big)^2\Big]$$

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



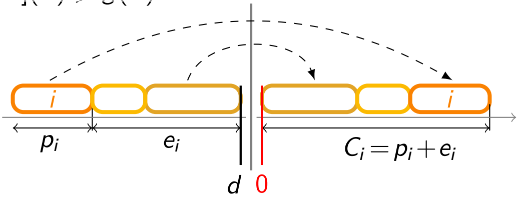
Non-overlapping inequalities for UCDDP

Queyranne's non-overlapping inequalities



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

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



Non-overlapping inequalities for UCDDP

$\forall S \subseteq J,\ p * t(S) \geqslant g(S)$ Are these inequalities valid?

Non-overlapping inequalities for UCDDP

$\forall S \subseteq J,\ p * t(S) \geqslant g(S)$ Are these inequalities valid?

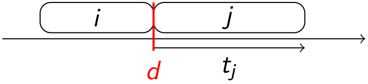
For $S = \{i, j\}$ with $t_i = 0$, $p * t(S) \geqslant g(S) \Leftrightarrow p_i t_i + p_j t_j > p_i^2 + p_j^2 + p_i p_j \Leftrightarrow t_j > p_j$

Non-overlapping inequalities for UCDDP

$\forall S \subseteq J,\ p * t(S) \geqslant g(S)$ Are these inequalities valid?

For $S = \{i, j\}$ with $t_i = 0$, $p * t(S) \geqslant g(S) \Leftrightarrow p_i t_i + p_j t_j > p_i^2 + p_j^2 + p_i p_j \Leftrightarrow t_j > p_j$

This inequality is not valid for

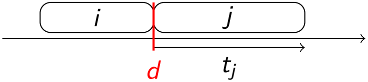


Non-overlapping inequalities for UCDDP

$\forall S \subseteq J,\ p * t(S) \geqslant g(S)$ Are these inequalities valid?

For $S = \{i, j\}$ with $t_i = 0$, $p * t(S) \geqslant g(S) \Leftrightarrow p_i t_i + p_j t_j > p_i^2 + p_j^2 + p_i p_j \Leftrightarrow t_j > p_j$

This inequality is not valid for



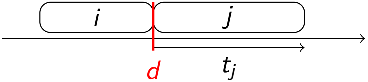
$\hookrightarrow$ So we consider: $\begin{cases} \forall S \subseteq J,\ p * [p+e](S \cap E) \geqslant g(S \cap E) \\ \forall S \subseteq J,\ p * t(S \cap T) \geqslant g(S \cap T) \end{cases}$

Non-overlapping inequalities for UCDDP

$\forall S \subseteq J,\ p * t(S) \geqslant g(S)$ Are these inequalities valid?

For $S = \{i, j\}$ with $t_i = 0$, $p * t(S) \geqslant g(S) \Leftrightarrow p_i t_i + p_j t_j > p_i^2 + p_j^2 + p_i p_j \Leftrightarrow t_j > p_j$

This inequality is not valid for



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

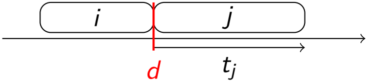
Using δ_j variables $E = \{j \in J \mid \delta_j = 1\}$ and $T = \{j \in J \mid \delta_j = 0\}$

Non-overlapping inequalities for UCDDP

$\forall S \subseteq J,\ p * t(S) \geqslant g(S)$ Are these inequalities valid?

For $S = \{i, j\}$ with $t_i = 0$, $p * t(S) \geqslant g(S) \Leftrightarrow p_i t_i + p_j t_j > p_i^2 + p_j^2 + p_i p_j \Leftrightarrow t_j > p_j$

This inequality is not valid for



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

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

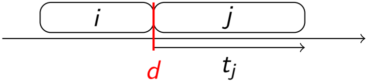
- the right-hand side term is no more a constant

Non-overlapping inequalities for UCDDP

$\forall S \subseteq J,\ p * t(S) \geqslant g(S)$ Are these inequalities valid?

For $S = \{i, j\}$ with $t_i = 0$, $p * t(S) \geqslant g(S) \Leftrightarrow p_i t_i + p_j t_j > p_i^2 + p_j^2 + p_i p_j \Leftrightarrow t_j > p_j$

This inequality is not valid for



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

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

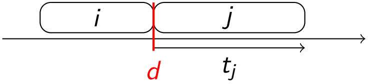
- the right-hand side term is no more a constant
- variables δ appear on both side to express the intersection

Non-overlapping inequalities for UCDDP

$\forall S \subseteq J,\ p * t(S) \geqslant g(S)$ Are these inequalities valid?

For $S = \{i, j\}$ with $t_i = 0$, $p * t(S) \geqslant g(S) \Leftrightarrow p_i t_i + p_j t_j > p_i^2 + p_j^2 + p_i p_j \Leftrightarrow t_j > p_j$

This inequality is not valid for



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

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

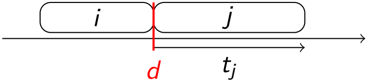
- the right-hand side term is no more a constant
- variables δ appear on both side to express the intersection
- products $\delta_i \delta_j$ appear

Non-overlapping inequalities for UCDDP

$\forall S \subseteq J,\ p * t(S) \geqslant g(S)$ Are these inequalities valid?

For $S = \{i, j\}$ with $t_i = 0$, $p * t(S) \geqslant g(S) \Leftrightarrow p_i t_i + p_j t_j > p_i^2 + p_j^2 + p_i p_j \Leftrightarrow t_j > p_j$

This inequality is not valid for



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

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

Falq, Fouilhoux, Kedad-Sidhoum, 2020, Discrete Applied Maths *(to appear)*

Formulation F^3 for UCDDP

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

Formulation F^3 for UCDDP

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

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

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

Formulation F^3 for UCDDP

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

Formulation F^3 for UCDDP

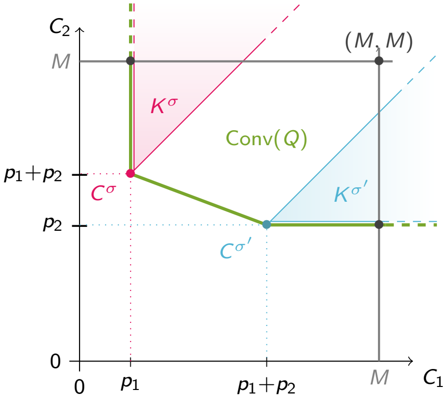
$$F^3 : \min \left\{ \sum_{j\in J} \alpha_j\, e_j + \beta_j\, t_j \;\middle|\; (e,t,\delta,X) \in \text{extr}(P^3) \;\text{ and }\; \delta \in \{0,1\}^J \right\}$$

where:

$$P^3 = \left\{ (e,t,\delta,X) \;\middle|\; \begin{array}{l}
\forall j\in J, 0 \leqslant \delta_j \leqslant 1 \quad (\delta) \\[6pt]
\forall j \in J,\; e_j \geqslant 0 \quad (e.0) \qquad \forall j\in J,\; t_j \geqslant 0 \quad (t.1) \\
 e_j \leqslant M\,\delta_j \quad (e.1) \qquad t_j \leqslant M\,(1-\delta_j) \quad (t.2) \\[6pt]
\forall (i,j)\in J^<,\; X_{i,j} \geqslant 0 \quad (x.1) \\
\phantom{\forall (i,j)\in J^<,\;} X_{i,j} \leqslant \delta_i + \delta_j \quad (x.2) \\
\phantom{\forall (i,j)\in J^<,\;} X_{i,j} \geqslant \delta_i - \delta_j \quad (x.3) \\
\phantom{\forall (i,j)\in J^<,\;} X_{i,j} \geqslant 2 - \delta_i - \delta_j \quad (x.4) \\[6pt]
\forall S \in \mathcal{P}(J),\; \sum_{i\in S} p_i\, e_i \geqslant \sum_{(i,j)\in S^<} p_i\, p_j\, \frac{\delta_i+\delta_j-X_{i,j}}{2} \quad (S1) \\[6pt]
\phantom{\forall S \in \mathcal{P}(J),\;} \sum_{i\in S} p_i\, t_i \geqslant \sum_{(i,j)\in S^<} p_i\, p_j\, \frac{2-(\delta_i+\delta_j)-X_{i,j}}{2} + \sum_{i\in S} p_i^2 (1-\delta_i) \quad (S2)
\end{array} \right\}$$

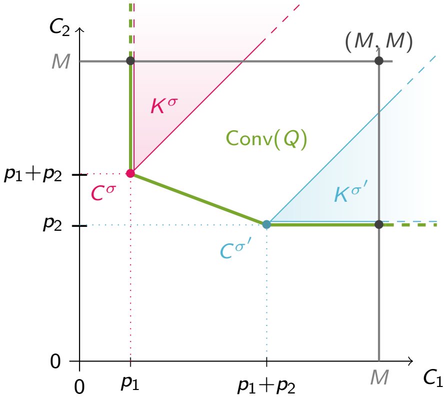
Validity of F^3

- 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
- we provide 2 key lemmas to use non-overlapping inequalities combined with additional inequalities



Two key lemmas for non-overlapping inequalities

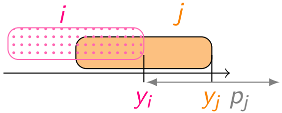
Let $y \in \mathbb{R}^J$ satisfying $\forall S \subseteq J,\ \sum_{j \in S} p_j y_j \geqslant g(S)\ \ (Q)$.

Two key lemmas for non-overlapping inequalities

Let $y \in \mathbb{R}^J$ satisfying $\forall S \subseteq J, \sum_{j \in S} p_j y_j \geqslant g(S)$ (Q).

Lemma 1

If there exists $(i, j) \in J^2$ s.t. $i \neq j$ and $y_i \leqslant y_j < y_i + p_j$,

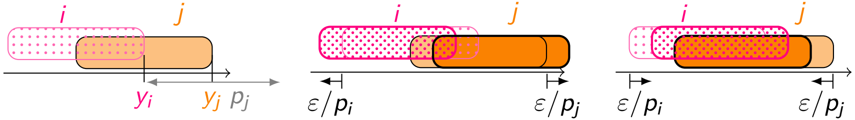


Two key lemmas for non-overlapping inequalities

Let $y \in \mathbb{R}^J$ satisfying $\forall S \subseteq J, \sum_{j \in S} p_j y_j \geqslant g(S)$ (Q).

Lemma 1

If there exists $(i,j) \in J^2$ s.t. $i \neq j$ and $y_i \leqslant y_j < y_i + p_j$,



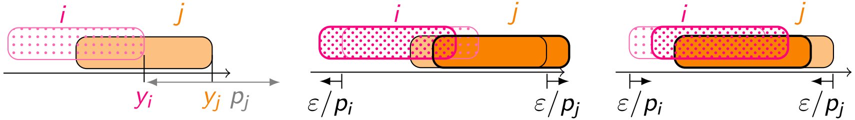
Two key lemmas for non-overlapping inequalities

Let $y \in \mathbb{R}^J$ satisfying $\forall S \subseteq J,\ \sum_{j \in S} p_j y_j \geqslant g(S)\ \ (Q)$.

Lemma 1

If there exists $(i,j) \in J^2$ s.t. $i \neq j$ and $y_i \leqslant y_j < y_i + p_j$,

then there exists $\varepsilon \in \mathbb{R}_+^*$ s.t. $\begin{cases} y^{+-} = y + \frac{\varepsilon}{p_i}\mathbb{I}_i - \frac{\varepsilon}{p_j}\mathbb{I}_j \\ y^{-+} = y - \frac{\varepsilon}{p_i}\mathbb{I}_i + \frac{\varepsilon}{p_j}\mathbb{I}_j \end{cases}$ also satisfy ineq. (Q).



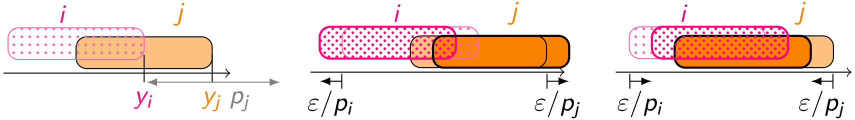
Two key lemmas for non-overlapping inequalities

Let $y \in \mathbb{R}^J$ satisfying $\forall S \subseteq J, \sum_{j \in S} p_j y_j \geqslant g(S)$ (Q).

Lemma 1 to use with extremality of y

If there exists $(i,j) \in J^2$ s.t. $i \neq j$ and $y_i \leqslant y_j < y_i + p_j$,

then there exists $\varepsilon \in \mathbb{R}_+^*$ s.t. $\begin{cases} y^{+-} = y + \frac{\varepsilon}{p_i}\mathbb{I}_i - \frac{\varepsilon}{p_j}\mathbb{I}_j \\ y^{-+} = y - \frac{\varepsilon}{p_i}\mathbb{I}_i + \frac{\varepsilon}{p_j}\mathbb{I}_j \end{cases}$ also satisfy ineq. (Q).



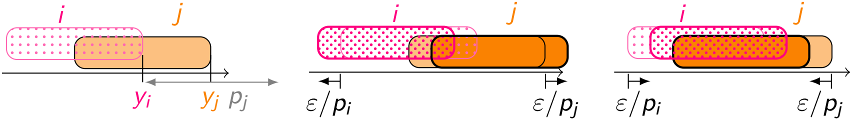
Two key lemmas for non-overlapping inequalities

Let $y \in \mathbb{R}^J$ satisfying $\forall S \subseteq J,\ \sum_{j \in S} p_j y_j \geqslant g(S)$ (Q).

Lemma 1 to use with extremality of y

If there exists $(i,j) \in J^2$ s.t. $i \neq j$ and $y_i \leqslant y_j < y_i + p_j$,

then there exists $\varepsilon \in \mathbb{R}_+^*$ s.t. $\begin{cases} y^{+-} = y + \frac{\varepsilon}{p_i}\mathbb{I}_i - \frac{\varepsilon}{p_j}\mathbb{I}_j \\ y^{-+} = y - \frac{\varepsilon}{p_i}\mathbb{I}_i + \frac{\varepsilon}{p_j}\mathbb{I}_j \end{cases}$ also satisfy ineq. (Q).



Lemma 2

If there exists $(i,j) \in J^2$ s.t. $i \neq j$, $y_j < y_i + p_j$ and $y_j \geqslant p_j$,

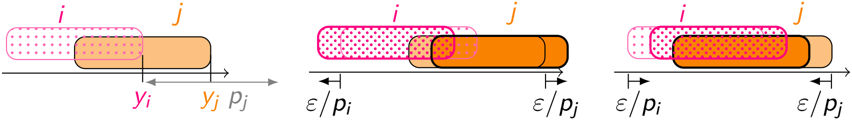
Two key lemmas for non-overlapping inequalities

Let $y \in \mathbb{R}^J$ satisfying $\forall S \subseteq J, \sum_{j \in S} p_j y_j \geqslant g(S)$ (Q).

Lemma 1 to use with extremality of y

If there exists $(i,j) \in J^2$ s.t. $i \neq j$ and $y_i \leqslant y_j < y_i + p_j$,

then there exists $\varepsilon \in \mathbb{R}_+^*$ s.t. $\begin{cases} y^{+-} = y + \frac{\varepsilon}{p_i}\mathbb{I}_i - \frac{\varepsilon}{p_j}\mathbb{I}_j \\ y^{-+} = y - \frac{\varepsilon}{p_i}\mathbb{I}_i + \frac{\varepsilon}{p_j}\mathbb{I}_j \end{cases}$ also satisfy ineq. (Q).



Lemma 2

If there exists $(i,j) \in J^2$ s.t. $i \neq j$, $y_j < y_i + p_j$ and $y_j \geqslant p_j$,

then there exists $\varepsilon \in \mathbb{R}_+^*$ s.t. $y - \frac{\varepsilon}{p_j}\mathbb{I}_j$ also satisfies ineq. (Q).

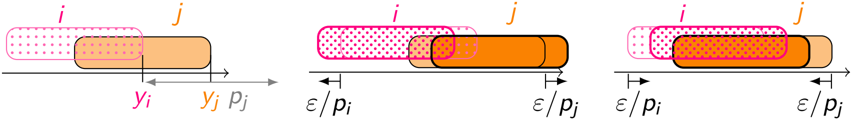
Two key lemmas for non-overlapping inequalities

Let $y \in \mathbb{R}^J$ satisfying $\forall S \subseteq J, \sum_{j\in S} p_j y_j \geqslant g(S)$ (Q).

Lemma 1 to use with extremality of y

If there exists $(i,j) \in J^2$ s.t. $i \neq j$ and $y_i \leqslant y_j < y_i + p_j$,

then there exists $\varepsilon \in \mathbb{R}_+^*$ s.t. $\begin{cases} y^{+-} = y + \frac{\varepsilon}{p_i}\mathbb{I}_i - \frac{\varepsilon}{p_j}\mathbb{I}_j \\ y^{-+} = y - \frac{\varepsilon}{p_i}\mathbb{I}_i + \frac{\varepsilon}{p_j}\mathbb{I}_j \end{cases}$ also satisfy ineq. (Q).



Lemma 2 to use with minimality of y

If there exists $(i,j) \in J^2$ s.t. $i \neq j$, $y_j < y_i + p_j$ and $y_j \geqslant p_j$,

then there exists $\varepsilon \in \mathbb{R}_+^*$ s.t. $y - \frac{\varepsilon}{p_j}\mathbb{I}_j$ also satisfies ineq. (Q).

Outline

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 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 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 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 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 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 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

Inequalities to separate

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

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

2 families of inequalities

$\hookrightarrow$ 2 independent separation problems

but also 2 similar separation problems

Inequalities to separate

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

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

2 families of inequalities

$\hookrightarrow$ 2 independent separation problems

but also 2 similar separation problems

separation of $(S1)$ inequalities	$\longrightarrow$	maximization of Γ_1

Inequalities to separate

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

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

2 families of inequalities

$\hookrightarrow$ 2 independent separation problems

but also 2 similar separation problems

separation of $(S1)$ inequalities $\longrightarrow$ maximization of Γ_1

separation of $(S2)$ inequalities $\longrightarrow$ maximization of Γ_2

Inequalities to separate

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

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

2 families of inequalities

↪ 2 independent separation problems
but also 2 similar separation problems

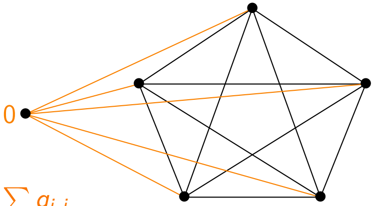
separation of $(S1)$ inequalities → maximization of Γ_1 →

separation of $(S2)$ inequalities → maximization of Γ_2 →

$$\max_{S\subset[1..n]} \Big(\sum_{(i,j)\in S^<} \overset{\in \mathbb{R}_+^*}{q_{i,j}} + \sum_{i\in S} \overset{\in \mathbb{R}}{c_i} \Big)$$

Reduction to a min-cut problem

$$\boxed{\max_{S\subset[1..n]} \sum_{(i,j)\in S^<} \overset{\in\mathbb{R}_+^*}{q_{i,j}} + \sum_{i\in S} \overset{\in\mathbb{R}}{c_i}}$$



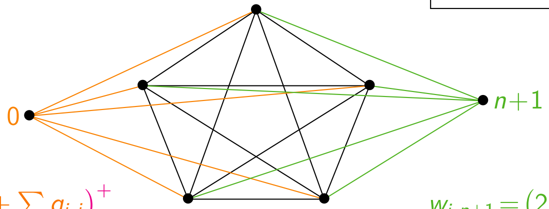
$\forall j\in[1..n]$
$w_{0,j} = 2c_i + \sum_{i\neq j} q_{i,j}$

$$\forall (i,j)\in[1..n],\ w_{i,j} = q_{i,j}$$

Picard et Ratliff, 1975, Networks

Reduction to a min-cut problem

$$\boxed{\max_{S\subset[1..n]} \sum_{(i,j)\in S^<} \overset{\in\mathbb{R}_+^*}{q_{i,j}} + \sum_{i\in S} \overset{\in\mathbb{R}}{c_i}}$$



$\forall j\in[1..n]$

$w_{0,j} = \left(2c_j + \sum_{i\neq j} q_{i,j}\right)^+$

$\forall i\in[1..n]$

$w_{i,n+1} = \left(2c_j + \sum_{i\neq j} q_{i,j}\right)^-$

$\forall (i,j)\in[1..n],\ w_{i,j} = q_{i,j}$

Picard et Ratliff, 1975, Networks

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

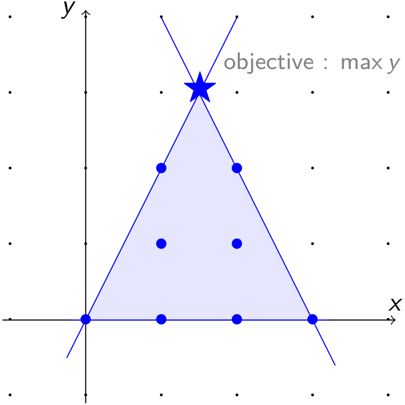
How to ensure extremality during a Branch-and-Bound?

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

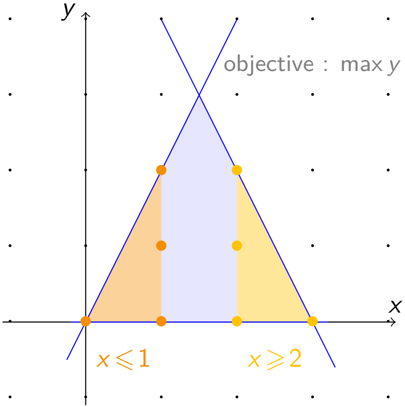
Counter-example:



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

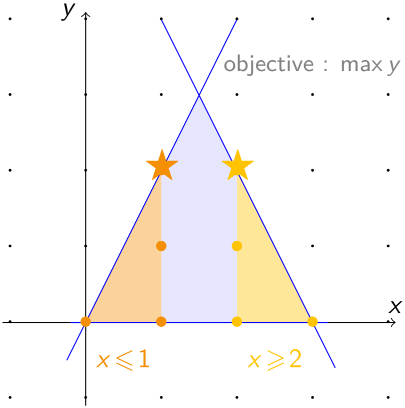
Counter-example:



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

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

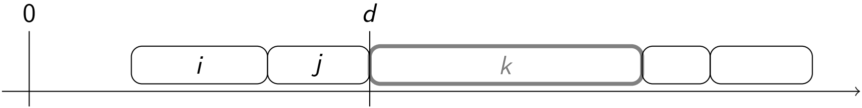
Counter-example:



Outline

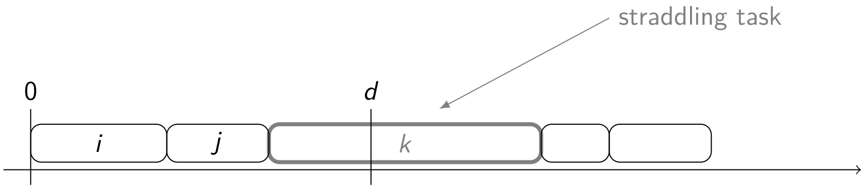
How to adapt the formulation for the general case ?

- unrestrictive case: d-blocks are dominant



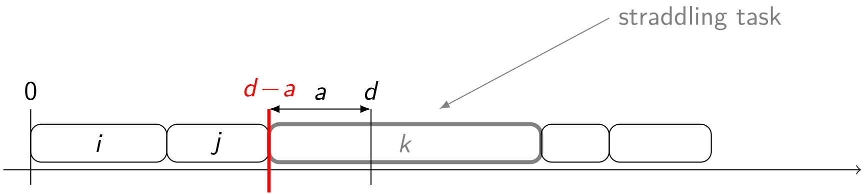
How to adapt the formulation for the general case ?

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



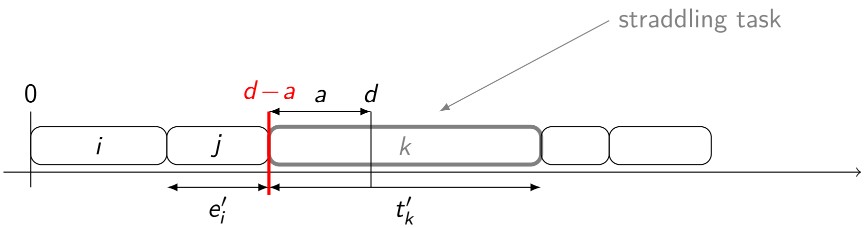
How to adapt the formulation for the general case ?

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



How to adapt the formulation for the general case ?

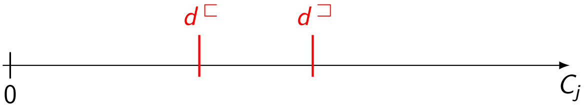
- unrestrictive case: d-blocks are dominant
- **general case**: d-or-left-blocks are dominant
  $\hookrightarrow$ new variable a for a **new reference point**: $d-a$



Interest of a flexible reference point?

Common due window problem:

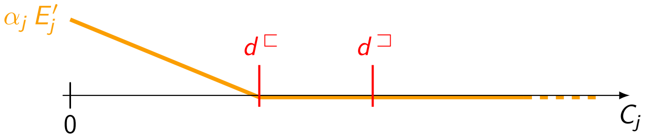
$\rightarrow$ a due window $[d^{\sqsubset}, d^{\sqsupset}]$ instead of a due date d



Interest of a flexible reference point?

Common due window problem:

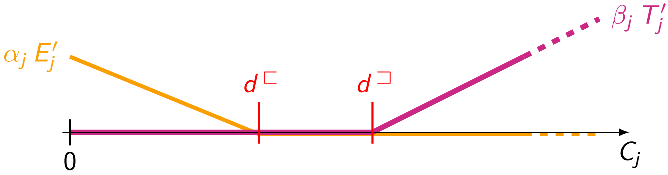
$\rightarrow$ a due window $[d^{\sqsubset}, d^{\sqsupset}]$ instead of a due date d



Interest of a flexible reference point?

Common due window problem:

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

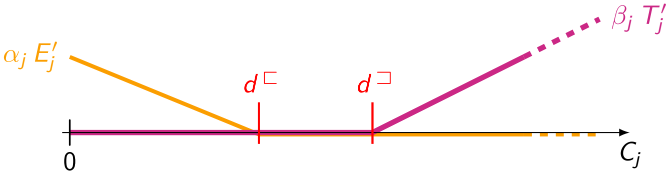


$\rightarrow$ block with a task completing at $d^{\sqsubset}$ or $d^{\sqsupset}$ are not dominant
  $\hookrightarrow$ a straddling task can occur over $d^{\sqsubset}$ (resp. over $d^{\sqsupset}$)

Interest of a flexible reference point?

Common due window problem:

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

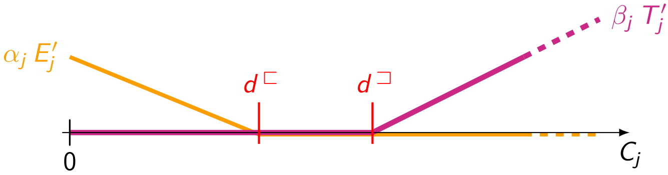


$\rightarrow$ block with a task completing at $d^{\sqsubset}$ or $d^{\sqsupset}$ are not dominant
  $\hookrightarrow$ a straddling task can occur over $d^{\sqsubset}$ (resp. over $d^{\sqsupset}$)
    $\hookrightarrow$ two half-axes with flexible reference point

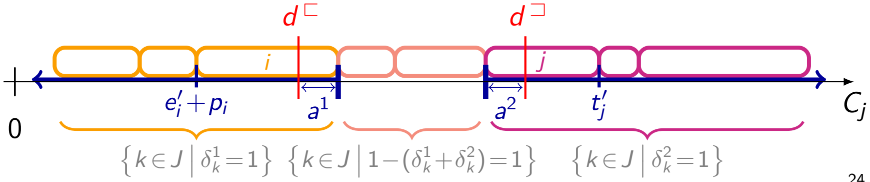
Interest of a flexible reference point?

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



Outline

1. Introduction
2. A formulation for UCDDP using natural variables
3. How to manage this kind of formulations in practice
4. How to extend this formulation
5. Conclusion

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
- $\rightarrow$ 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
- $\rightarrow$ 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:

- $\rightarrow$ formulate both UCDDP and CDDP
- $\rightarrow$ solve UCDDP instances up to size 40 within one hour
- $\rightarrow$ solve CDDP instances up to size 20 or 30 within one hour
- $\rightarrow$ 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:

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