Inégalités linéaires de dominance pour l'ordonnancement juste-à-temps avec date d'échéance commune non restrictive

Anne-Elisabeth FALQ[1]
Pierre Fouilhoux[1] and Safia Kedad-Sidhoum[2]

1 : Sorbonne Université, CNRS, LIP6, Paris
2 : CNAM, CEDRIC, Paris

ROADEF, Montpellier, février 2020

Outline

1. Introduction
2. Dominance properties
3. Dominance inequalities
4. Conclusion

Outline

The Unrestritive Common Due Date Problem (UCDDP)

An instance =

- a set of tasks J

The Unrestritive Common Due Date Problem (UCDDP)

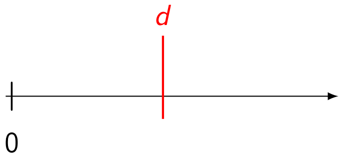
An instance =

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

The Unrestritive Common Due Date Problem (UCDDP)

An instance =

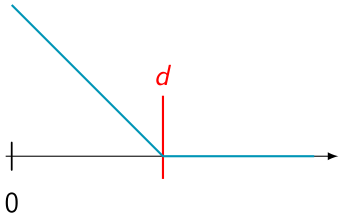
- a set of tasks J
- their processing times $(p_j)_{j \in J}$
- an **unrestrictive** common due-date $d \geqslant \sum p_j$



The Unrestritive Common Due Date Problem (UCDDP)

An instance =

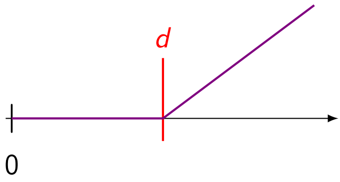
- a set of tasks J
- their processing times $(p_j)_{j \in J}$
- an **unrestrictive** common due-date $d \geqslant \sum p_j$
- their unit earliness penalties $(\alpha_j)_{j \in J}$



The Unrestritive Common Due Date Problem (UCDDP)

An instance =

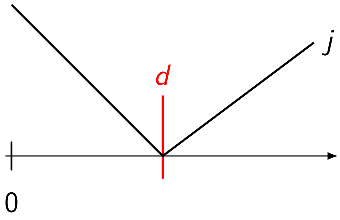
- a set of tasks J
- their processing times $(p_j)_{j \in J}$
- an **unrestrictive** common due-date $d \geqslant \sum p_j$
- their unit earliness penalties $(\alpha_j)_{j \in J}$
- their unit tardiness penalties $(\beta_j)_{j \in J}$



The Unrestritive Common Due Date Problem (UCDDP)

An instance =

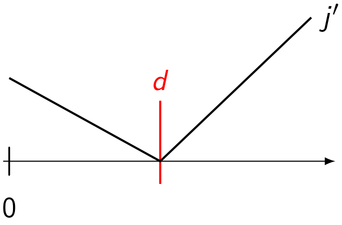
- a set of tasks J
- their processing times $(p_j)_{j \in J}$
- an **unrestrictive** common due-date $d \geqslant \sum p_j$
- their unit earliness penalties $(\alpha_j)_{j \in J}$
- their unit tardiness penalties $(\beta_j)_{j \in J}$



The Unrestritive Common Due Date Problem (UCDDP)

An instance =

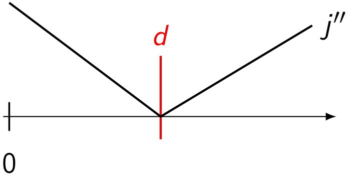
- a set of tasks J
- their processing times $(p_j)_{j \in J}$
- an **unrestrictive** common due-date $d \geqslant \sum p_j$
- their unit earliness penalties $(\alpha_j)_{j \in J}$
- their unit tardiness penalties $(\beta_j)_{j \in J}$



The Unrestritive Common Due Date Problem (UCDDP)

An instance =

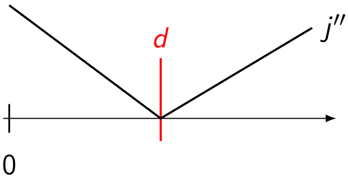
- a set of tasks J
- their processing times $(p_j)_{j \in J}$
- an **unrestrictive** common due-date $d \geqslant \sum p_j$
- their unit earliness penalties $(\alpha_j)_{j \in J}$
- their unit tardiness penalties $(\beta_j)_{j \in J}$



The Unrestritive Common Due Date Problem (UCDDP)

An instance =

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



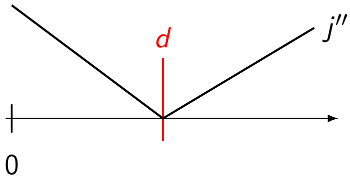
A solution schedule =

- a family of pairwise disjoint processing intervals

The Unrestritive Common Due Date Problem (UCDDP)

An instance =

- a set of tasks J
- their processing times $(p_j)_{j \in J}$
- an **unrestrictive** common due-date $d \geqslant \sum p_j$
- 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 = $\min \sum_{j \in J} \alpha_j E_j + \beta_j T_j$

Definition of dominance

Let T be a solution subset.

Definition of dominance

Let T be a solution subset.

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

Definition of dominance

Let T be a solution subset.

- 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

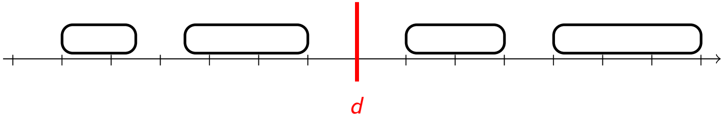
Definition of dominance

Let T be a solution subset.

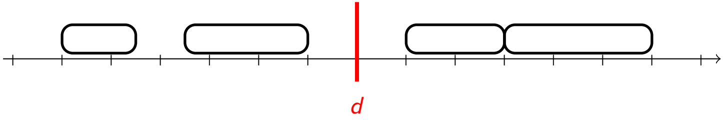
- T is a dominant set if it contains at least one optimal solution
- T is a strictly dominant set if it contains all the optimal solutions

In both cases, the searching space can be reduced to T, other solutions can be discarded.

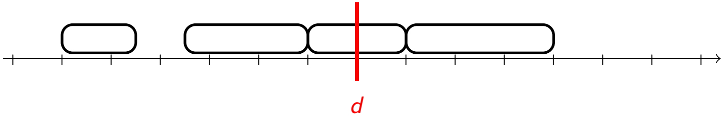
Dominance properties for the UCDDP



Dominance properties for the UCDDP

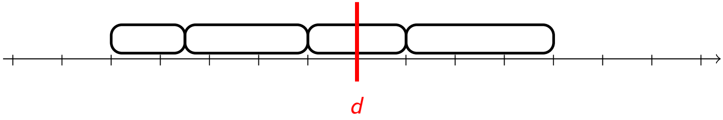


Dominance properties for the UCDDP



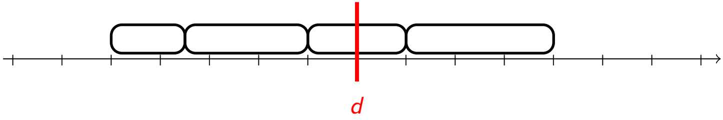
Hall and Posner, 1991, Operations research

Dominance properties for the UCDDP



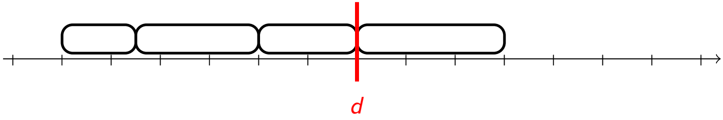
Dominance properties for the UCDDP

- blocks *(= schedules without idle time)* are strictly dominant



Dominance properties for the UCDDP

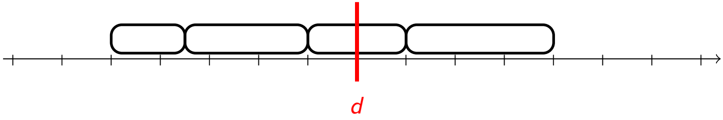
- blocks *(= schedules without idle time)* are strictly dominant



Hall and Posner, 1991, Operations research

Dominance properties for the UCDDP

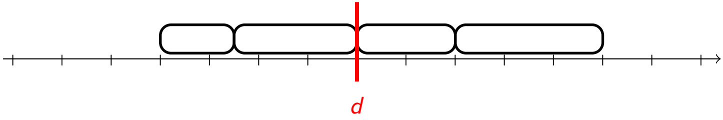
- blocks *(= schedules without idle time)* are strictly dominant



Hall and Posner, 1991, Operations research

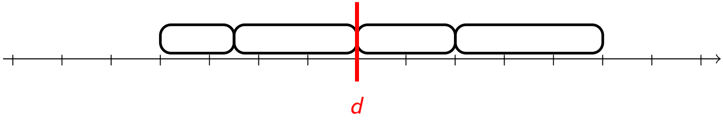
Dominance properties for the UCDDP

- blocks *(= schedules without idle time)* are strictly dominant



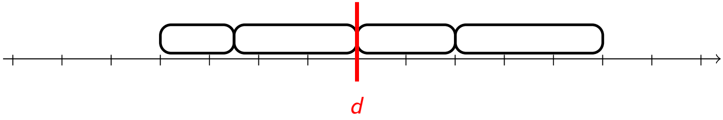
Dominance properties for the UCDDP

- blocks *(= schedules without idle time)* are strictly dominant
- schedules with one on-time task are dominant



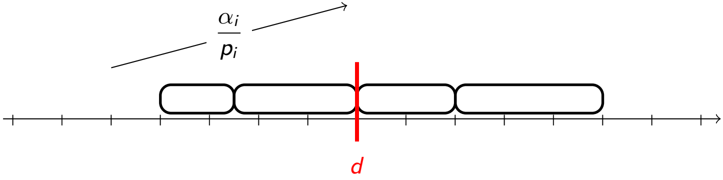
Dominance properties for the UCDDP

- blocks *(= schedules without idle time)* are strictly dominant
- schedules with one on-time task are dominant
- V-shaped schedules are strictly dominant



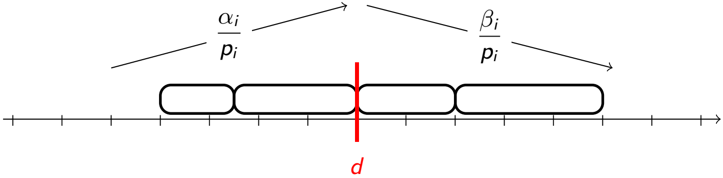
Dominance properties for the UCDDP

- blocks *(= schedules without idle time)* are strictly dominant
- schedules with one on-time task are dominant
- V-shaped schedules are strictly dominant



Dominance properties for the UCDDP

- blocks *(= schedules without idle time)* are strictly dominant
- schedules with one on-time task are dominant
- V-shaped schedules are strictly dominant



Main results for the UCDDP

- Complexity:

- Solving approach:

Main results for the UCDDP

- Complexity:
  - **unitary case**: $\forall j \in J, \alpha_j = \beta_j = \text{cst} \quad \Rightarrow$ P

- Solving approach:

Kanet, 1981, Naval Research Logistics Quaterly

Main results for the UCDDP

- Complexity:
  - **unitary case**: $\forall j \in J, \alpha_j = \beta_j = \text{cst} \quad \Rightarrow$ P
  - **symmetric case**: $\forall j \in J, \alpha_j = \beta_j \quad \Rightarrow$ NP-hard

- Solving approach:

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

Main results for the UCDDP

- Complexity:
  - **unitary case**: $\forall j \in J, \alpha_j = \beta_j = \text{cst} \quad \Rightarrow$ P
  - **symmetric case**: $\forall j \in J, \alpha_j = \beta_j \quad \Rightarrow$ NP-hard (in a weak sense)

- Solving approach:
  - dynamic programming *in case of symmetric penalties*

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

Main results for the UCDDP

- Complexity:
  - **unitary case**: $\forall j \in J, \alpha_j = \beta_j = \text{cst} \quad \Rightarrow$ P
  - **symmetric case**: $\forall j \in J, \alpha_j = \beta_j \quad \Rightarrow$ NP-hard (in a weak sense)
  - **general case**: arbitrary coeficients $\alpha, \beta \quad \Rightarrow$ NP-hard

- Solving approach:
  - dynamic programming *in case of symmetric penalties*

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

Main results for the UCDDP

- Complexity:
  - **unitary case**: $\forall j \in J, \alpha_j = \beta_j = \text{cst} \quad \Rightarrow$ P
  - **symmetric case**: $\forall j \in J, \alpha_j = \beta_j \quad \Rightarrow$ NP-hard (in a weak sense)
  - **general case**: arbitrary coeficients $\alpha, \beta \quad \Rightarrow$ NP-hard

- Solving approach:
  - dynamic programming *in case of symmetric penalties*
  - heuristic

Kanet, 1981, Naval Research Logistics Quaterly
Hall and Posner, 1991, Operations research
Biskup and Feldmann, 2001, Computers and Operations Research

Main results for the UCDDP

- Complexity:
  - **unitary case**: $\forall j \in J, \alpha_j = \beta_j = \text{cst} \quad \Rightarrow$ P
  - **symmetric case**: $\forall j \in J, \alpha_j = \beta_j \quad \Rightarrow$ NP-hard (in a weak sense)
  - **general case**: arbitrary coeficients $\alpha, \beta \quad \Rightarrow$ NP-hard
- Solving approach:
  - dynamic programming *in case of symmetric penalties*
  - heuristic
  - branch-and-bound algorithm *solve up to 1000-task instances within 1400 s*

Kanet, 1981, Naval Research Logistics Quaterly
Hall and Posner, 1991, Operations research
Biskup and Feldmann, 2001, Computers and Operations Research
Sourd, 2009, Informs Journal on Computing

Main results for the UCDDP

- Complexity:
  - **unitary case**: $\forall j \in J, \alpha_j = \beta_j = \text{cst}$ $\Rightarrow$ P
  - **symmetric case**: $\forall j \in J, \alpha_j = \beta_j$ $\Rightarrow$ NP-hard (in a weak sense)
  - **general case**: arbitrary coeficients α, β $\Rightarrow$ NP-hard

- Solving approach:
  - dynamic programming *in case of symmetric penalties*
  - heuristic
  - branch-and-bound algorithm *solve up to 1000-task instances within 1400 s*
  - MIP formulations *(one using natural variables, one compact)*

Kanet, 1981, Naval Research Logistics Quaterly
Hall and Posner, 1991, Operations research
Biskup and Feldmann, 2001, Computers and Operations Research
Sourd, 2009, Informs Journal on Computing
Falq, Fouilhoux, Kedad-Sidhoum 2019, https://arxiv.org/abs/1901.06880

Main results for the UCDDP

- Complexity:
  - **unitary case**: $\forall j \in J, \alpha_j = \beta_j = \text{cst}$ $\Rightarrow$ P
  - **symmetric case**: $\forall j \in J, \alpha_j = \beta_j$ $\Rightarrow$ NP-hard (in a weak sense)
  - **general case**: arbitrary coeficients α, β $\Rightarrow$ NP-hard

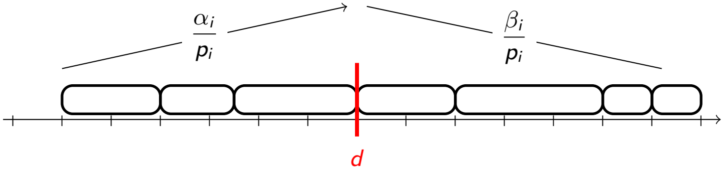
- Solving approach:
  - dynamic programming *in case of symmetric penalties*
  - heuristic
  - branch-and-bound algorithm *solve up to 1000-task instances within 1400 s*
  - MIP formulations *(one using natural variables, one compact)*

- This work: translate dominance properties in a MIP context

Kanet, 1981, Naval Research Logistics Quaterly
Hall and Posner, 1991, Operations research
Biskup and Feldmann, 2001, Computers and Operations Research
Sourd, 2009, Informs Journal on Computing
Falq, Fouilhoux, Kedad-Sidhoum 2019, `https://arxiv.org/abs/1901.06880`

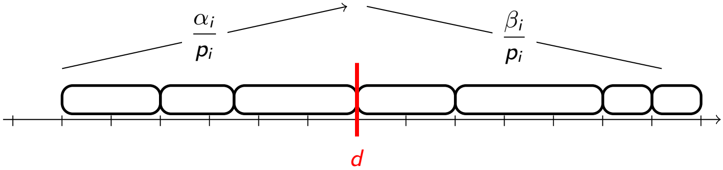
Outline

Structural dominance properties



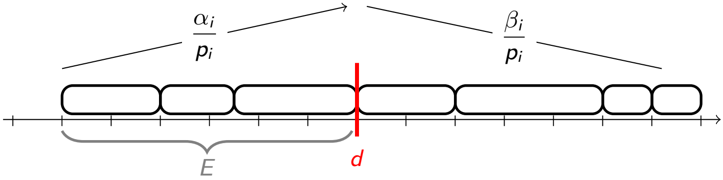
- consider only the V-shaped d-blocks since they are dominant

Structural dominance properties



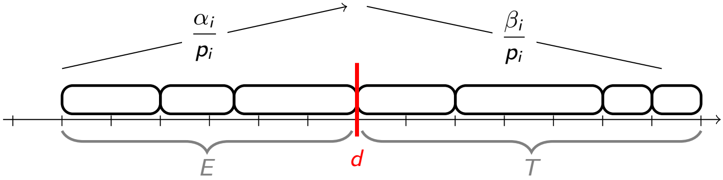
- consider only the V-shaped d-blocks since they are dominant
- define the early/tardy partition of a V-shaped d-block

Structural dominance properties



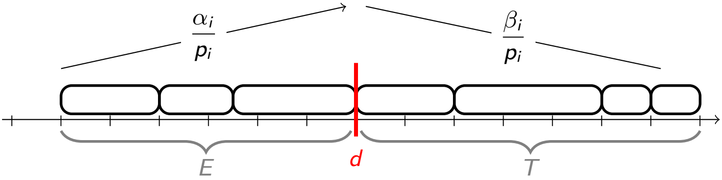
- consider only the V-shaped d-blocks since they are dominant
- define the early/tardy partition of a V-shaped d-block

Structural dominance properties



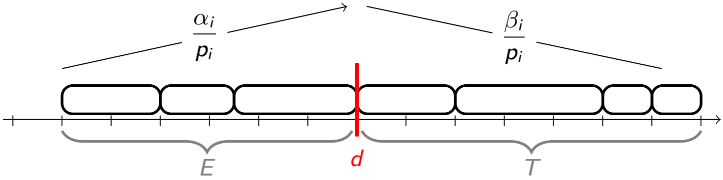
- consider only the V-shaped d-blocks since they are dominant
- define the early/tardy partition of a V-shaped d-block

Structural dominance properties



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

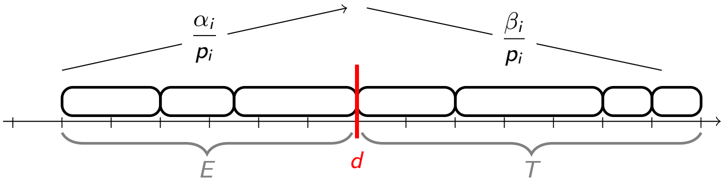
Structural dominance properties



- consider only the V-shaped d-blocks since they are dominant
- define the early/tardy partition of a V-shaped d-block
- V-shaped d-blocks having the same early/tardy partition (E, T) have the same penalty $f(E, T)$
- formulate UCDDP as a partition problem $\min_{(E,T) \in \vec{\mathcal{P}}_2^*(J)} f(E, T)$

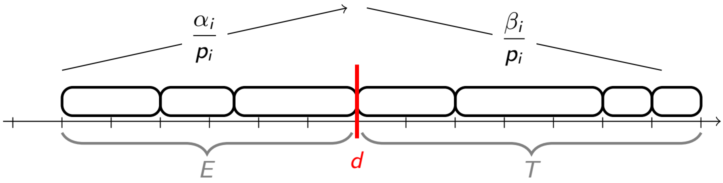
where $\vec{\mathcal{P}}_2^*(J) = \{(E, T) \mid \{E, T\} \text{ is a partition of } J \text{ and } E \neq \emptyset\}$

Structural dominance properties



- consider only the V-shaped d-blocks since they are dominant
- introduce notations for any $u \in J$,

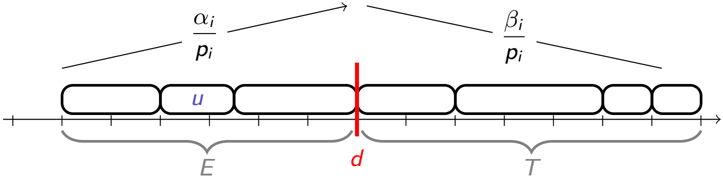
Structural dominance properties



- consider only the V-shaped d-blocks since they are dominant
- introduce notations for any $u \in J$,

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

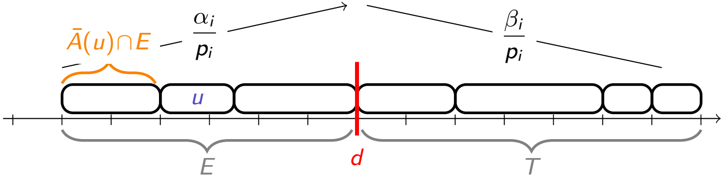
Structural dominance properties



- consider only the V-shaped d-blocks since they are dominant
- introduce notations for any $u \in J$,

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

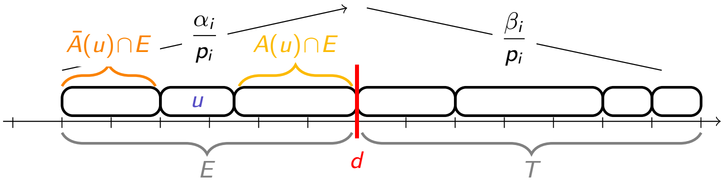
Structural dominance properties



- consider only the V-shaped d-blocks since they are dominant
- introduce notations for any $u \in J$,

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

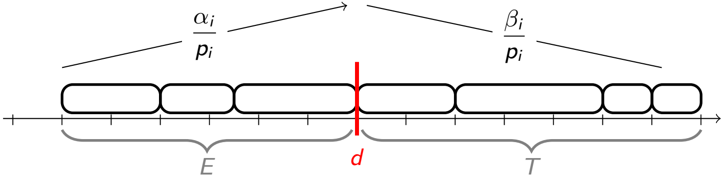
Structural dominance properties



- consider only the V-shaped d-blocks since they are dominant
- introduce notations for any $u \in J$,

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

Structural dominance properties

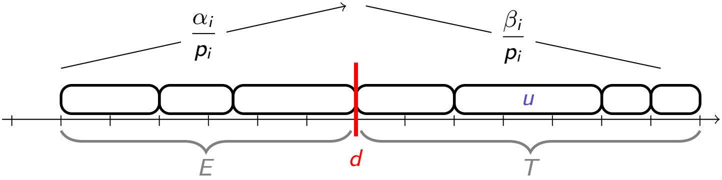


- consider only the V-shaped d-blocks since they are dominant
- introduce notations for any $u \in J$,

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

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

Structural dominance properties

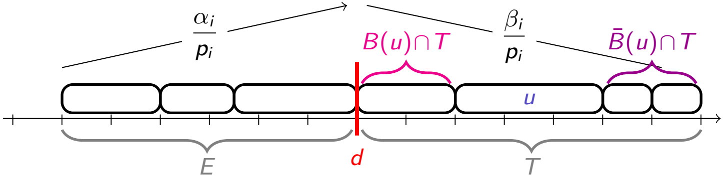


- consider only the V-shaped d-blocks since they are dominant
- introduce notations for any $u \in J$,

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

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

Structural dominance properties



- consider only the V-shaped d-blocks since they are dominant
- introduce notations for any $u \in J$,

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

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

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

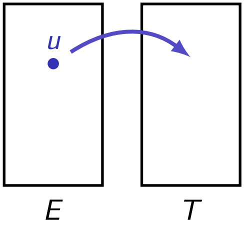
Remark: If a solution is dominated by one of its neighbors, then it is not an optimal solution.

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

Our approach:

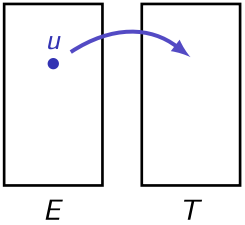
- define a neighborhood based on operations
- translate the associate dominance property by constraints

Insert and swap operations on partitions

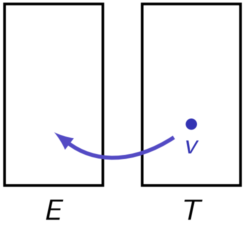


insert *of an early task*

Insert and swap operations on partitions

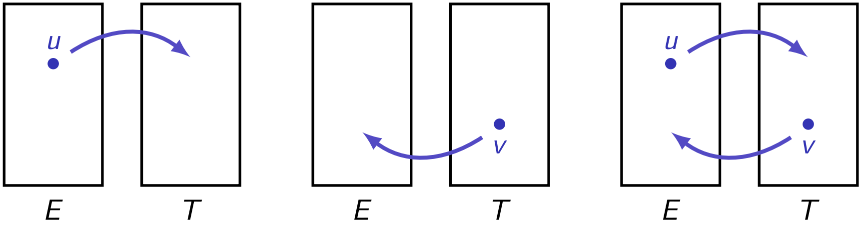


insert *of an early task*



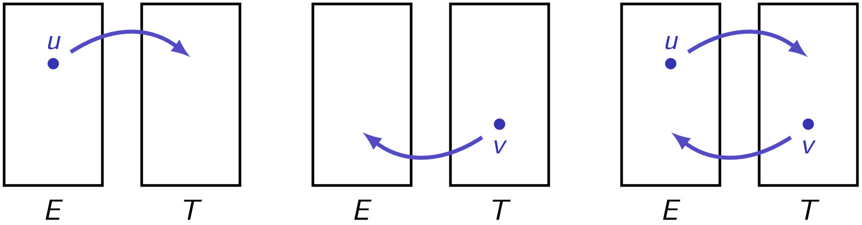
insert *of a tardy task*

Insert and swap operations on partitions



insert** of an early task* | ***insert** of a tardy task* | ***swap

Insert and swap operations on partitions



insert of an early task *insert of a tardy task* *swap*

A partition (E, T) is said:

- insert non-dominated if $\begin{cases} \forall v \in T,\ f(E,T) \leqslant f(E \cup \{v\},\ T_{\setminus\{v\}}) \\ \forall u \in E,\ f(E,T) \leqslant f(E_{\setminus\{u\}},\ T \cup \{u\}) \end{cases}$
- swap non-dominated if $\forall (u,v) \in E \times T,\ f(E,T) \leqslant f(E_{\setminus\{u\}} \cup \{v\},\ T_{\setminus\{v\}} \cup \{u\})$

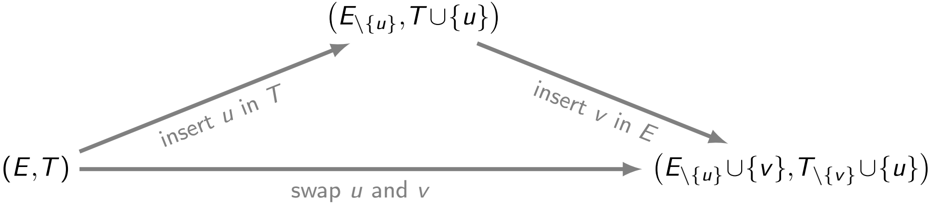
Compare insert dominance and swap dominance

(E, T)

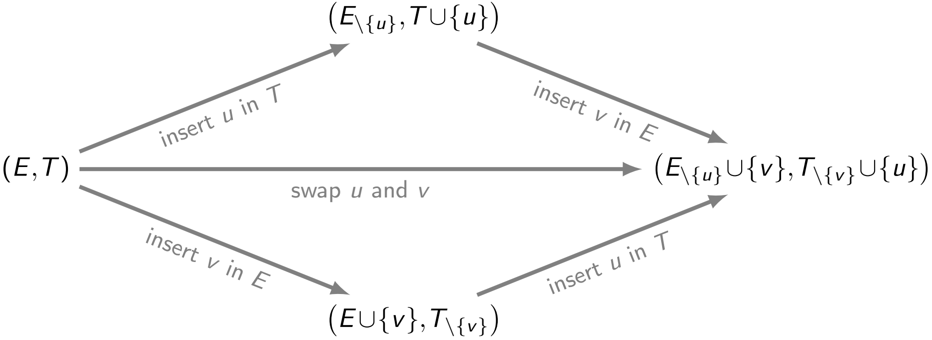
Compare insert dominance and swap dominance

$$(E, T) \xrightarrow{\text{swap } u \text{ and } v} \left(E_{\setminus\{u\}} \cup \{v\}, T_{\setminus\{v\}} \cup \{u\}\right)$$

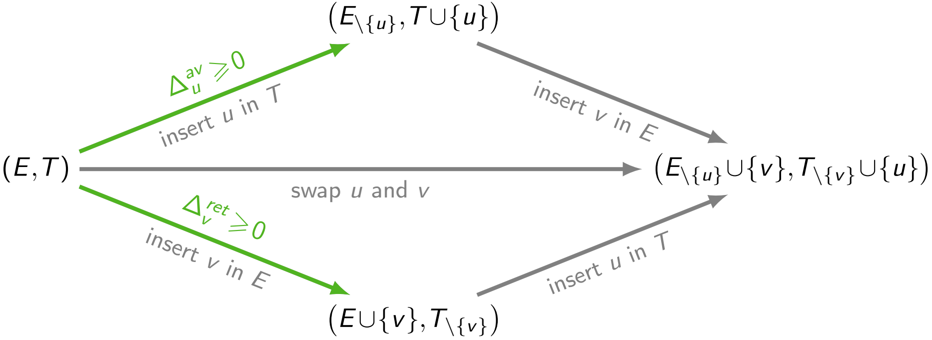
Compare insert dominance and swap dominance



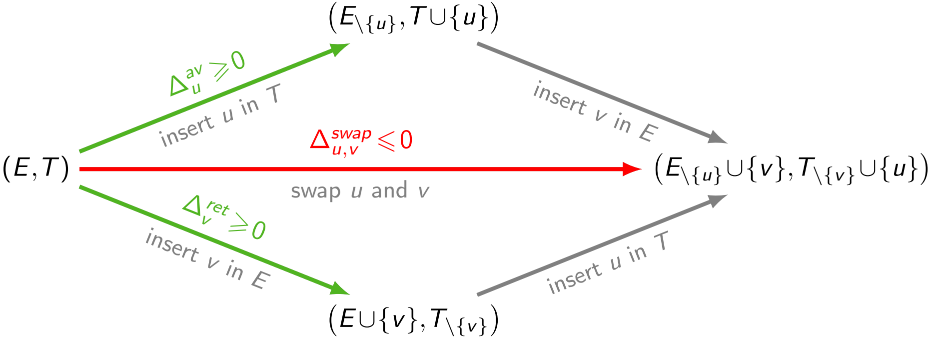
Compare insert dominance and swap dominance



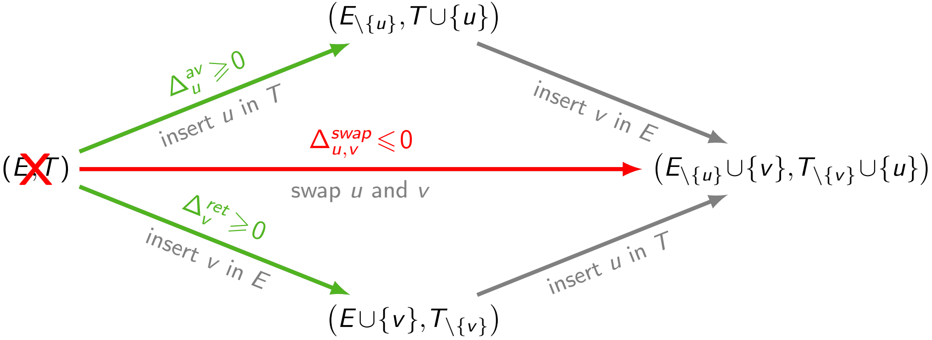
Compare insert dominance and swap dominance



Compare insert dominance and swap dominance



Compare insert dominance and swap dominance



An example on schedules

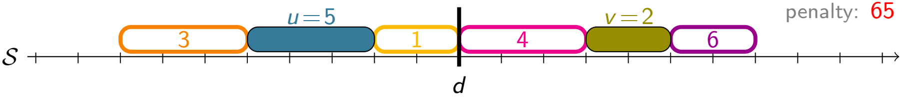
Instance :

α_j	4	3	3	-	5	-
β_j	-	3	-	6	3	1

An example on schedules

Instance :

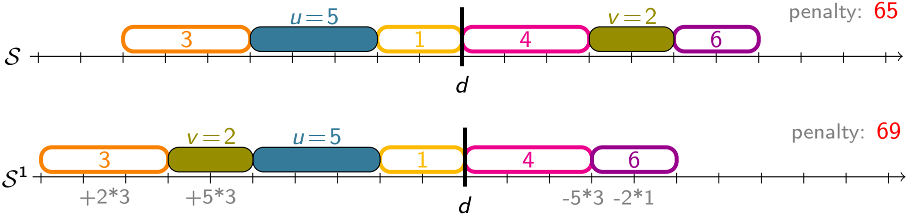
α_j	4	3	3	-	5	-
β_j	-	3	-	6	3	1



An example on schedules

Instance :

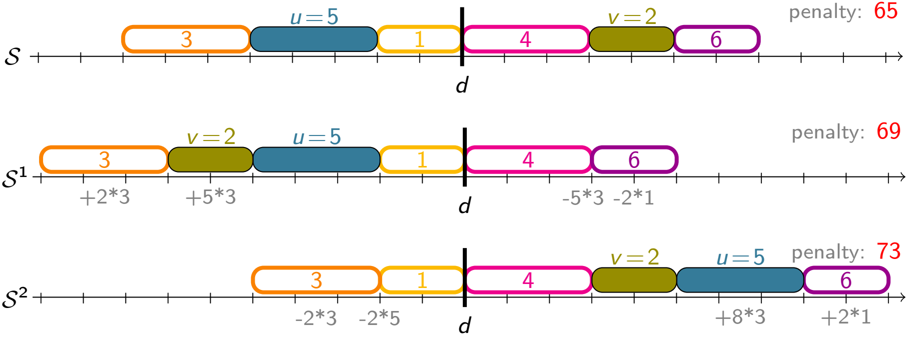
α_j	4	3	3	-	5	-
β_j	-	3	-	6	3	1



An example on schedules

Instance :

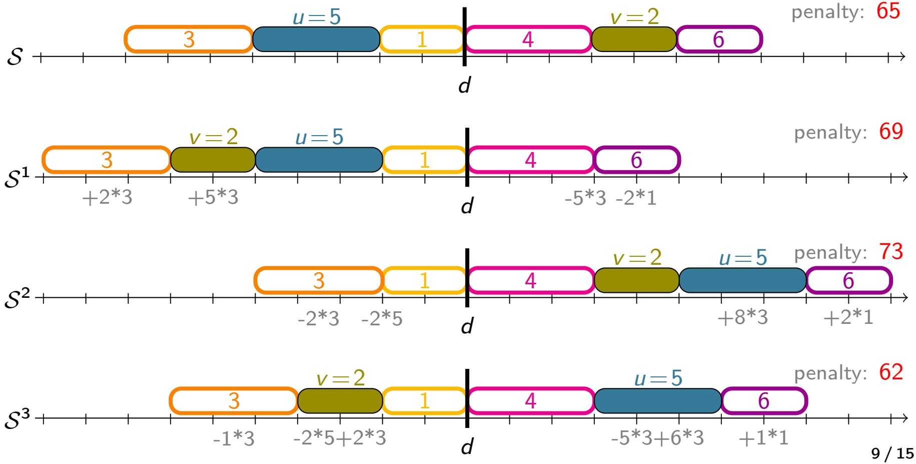
α_j	4	3	3	-	5	-
β_j	-	3	-	6	3	1



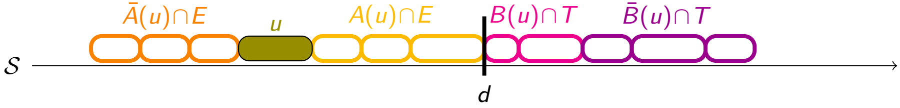
An example on schedules

Instance :

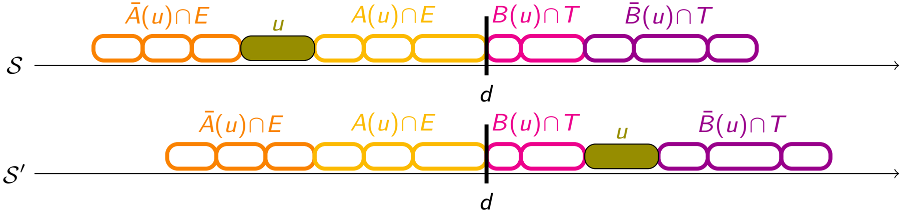
α_j	4	3	3	-	5	-
β_j	-	3	-	6	3	1



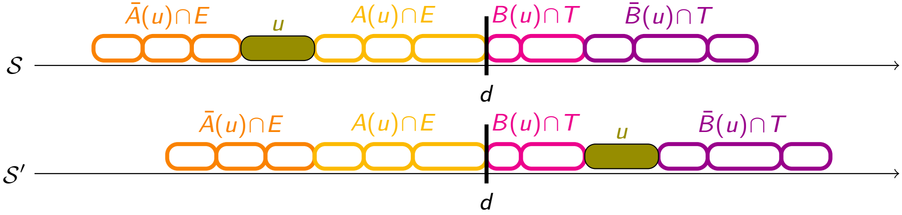
Cost variation induced by the insertion of an early task u



Cost variation induced by the insertion of an early task u

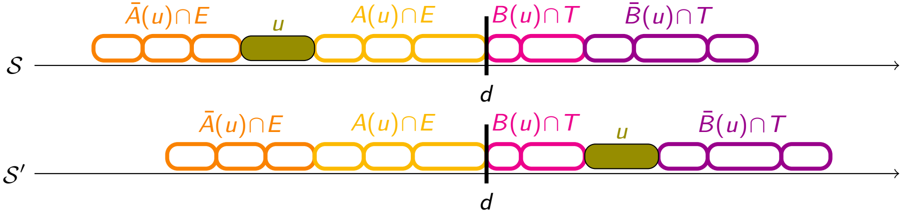


Cost variation induced by the insertion of an early task u



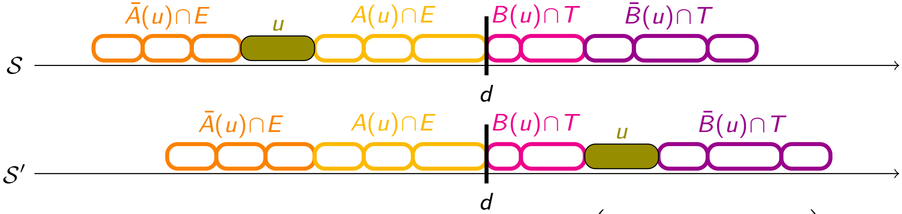
$\text{cost}(\mathcal{S}') = \text{cost}(\mathcal{S})$

Cost variation induced by the insertion of an early task u



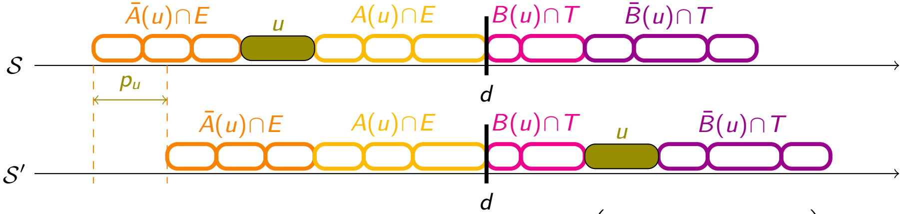
$$\text{cost}(\mathcal{S}') = \text{cost}(\mathcal{S}) - \alpha_u \, p\big(\underline{A(u) \cap E}\big)$$

Cost variation induced by the insertion of an early task u



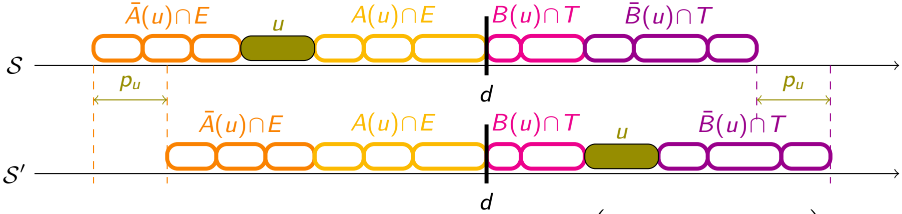
$$\text{cost}(\mathcal{S}') = \text{cost}(\mathcal{S}) - \alpha_u\, p\big(\underline{A(u)\cap E}\big) + \beta_u \Big(p\big(\underline{B(v)\cap T}\big) + p_u\Big)$$

Cost variation induced by the insertion of an early task u



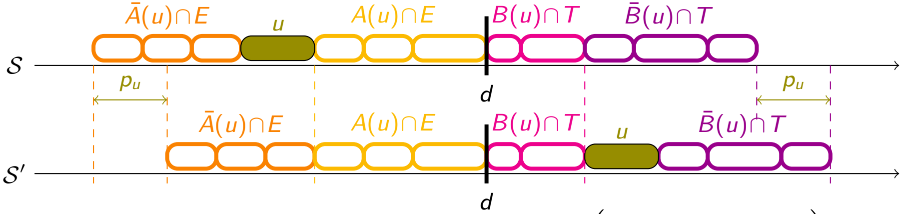
$$\text{cost}(\mathcal{S}') = \text{cost}(\mathcal{S}) - \alpha_u\, p\big(A(u)\cap E\big) + \beta_u \Big(p\big(B(v)\cap T\big) + p_u\Big) - p_u\, \alpha\big(\bar{A}(u)\cap E\big)$$

Cost variation induced by the insertion of an early task u



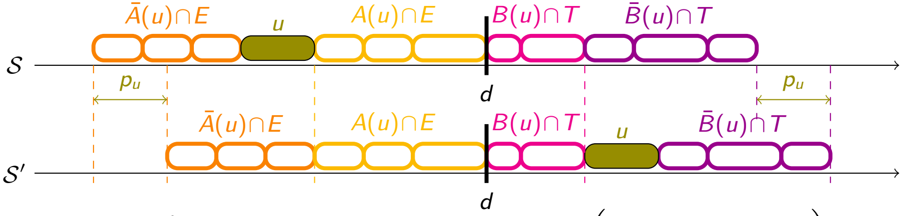
$$\text{cost}(\mathcal{S}') = \text{cost}(\mathcal{S}) - \alpha_u\, p\big(\underline{A(u) \cap E}\big) + \beta_u \Big(p\big(\underline{B(v) \cap T}\big) + p_u\Big)$$
$$- p_u\, \alpha\big(\underline{\bar{A}(u) \cap E}\big) + p_u\, \beta\big(\underline{\bar{B}(v) \cap T}\big)$$

Cost variation induced by the insertion of an early task u



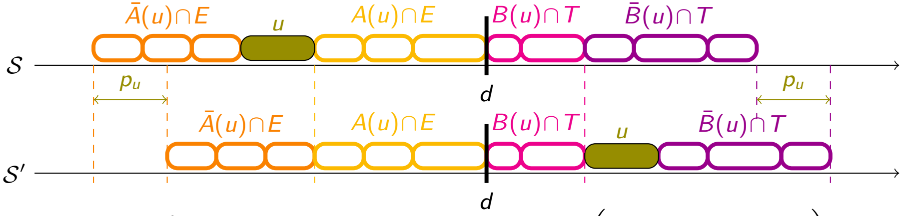
$$\text{cost}(\mathcal{S}') = \text{cost}(\mathcal{S}) - \alpha_u\, p\big(A(u)\cap E\big) + \beta_u \Big(p\big(B(v)\cap T\big) + p_u\Big)$$
$$- p_u\, \alpha\big(\bar{A}(u)\cap E\big) + p_u\, \beta\big(\bar{B}(v)\cap T\big)$$

Cost variation induced by the insertion of an early task u



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

Cost variation induced by the insertion of an early task u

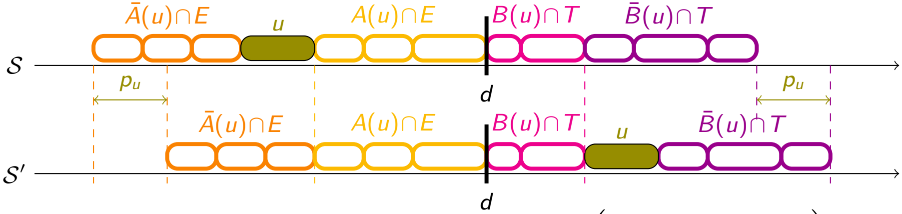


$$\Delta_u^{early}(E,T) = -\alpha_u\, p\big(A(u)\cap E\big) + \beta_u \Big(p\big(B(v)\cap T\big) + p_u\Big) - p_u\, \alpha\big(\bar{A}(u)\cap E\big) + p_u\, \beta\big(\bar{B}(v)\cap T\big)$$

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

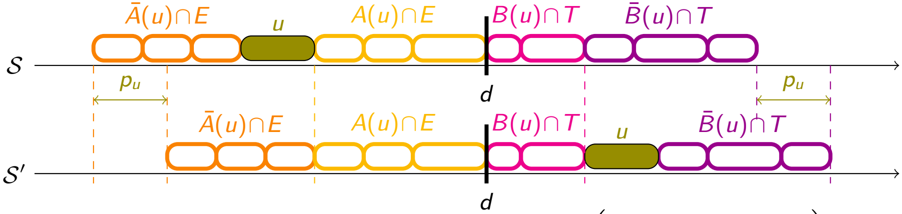


$$\Delta_u^{early}(E,T) = -\alpha_u\, p\big(A(u)\cap E\big) + \beta_u \Big(p\big(B(v)\cap T\big) + p_u\Big) - p_u\, \alpha\big(\bar{A}(u)\cap E\big) + p_u\, \beta\big(\bar{B}(v)\cap T\big)$$

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

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



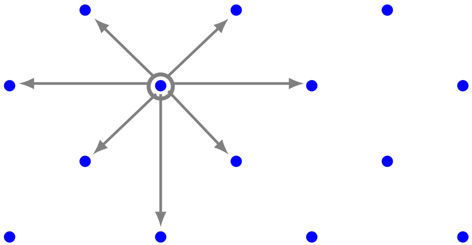
$$\Delta_u^{early}(E,T) = -\alpha_u\, p\big(A(u)\cap E\big) + \beta_u \Big(p\big(B(v)\cap T\big) + p_u\Big) - p_u\,\alpha\big(\bar{A}(u)\cap E\big) + p_u\,\beta\big(\bar{B}(v)\cap T\big)$$

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 $(u,v)\in E\times T$

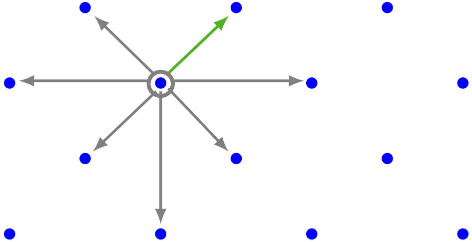
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

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

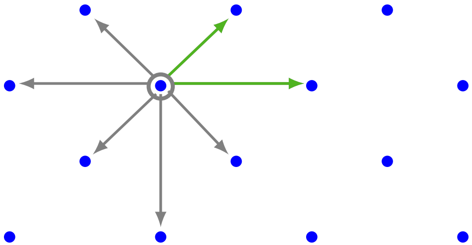
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

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

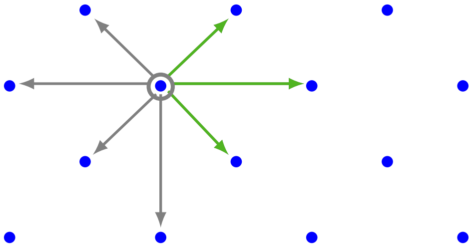
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

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

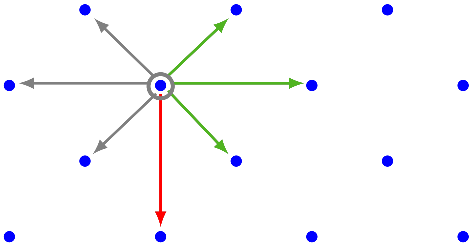
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

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

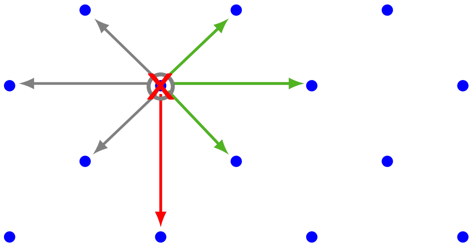
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

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

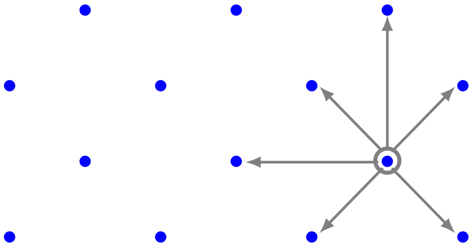
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

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

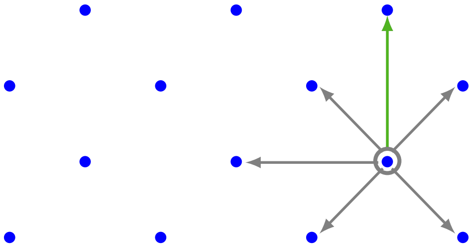
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

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

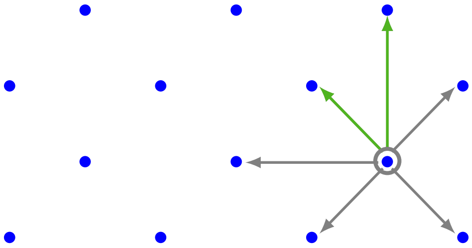
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

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

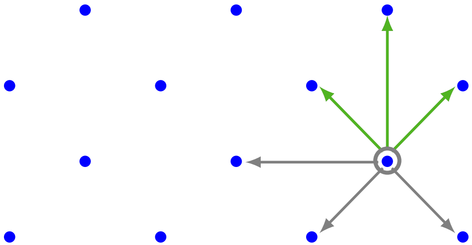
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

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

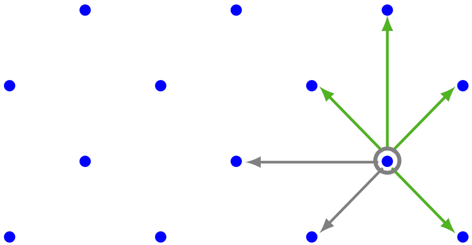
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

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

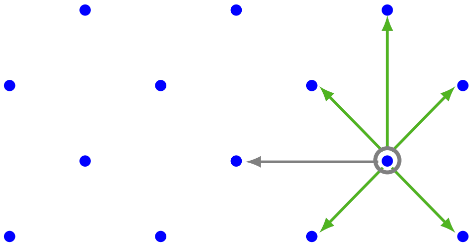
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

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

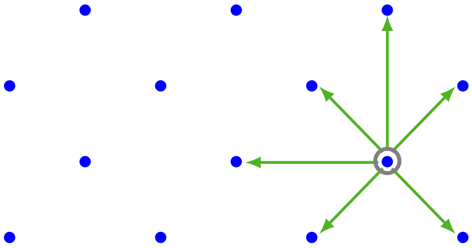
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

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

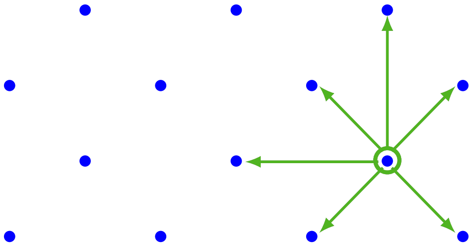
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

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

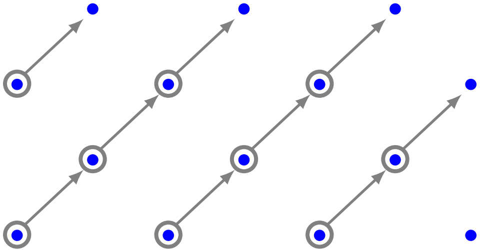
Neighborhood: solution-centered vs operation-centered point of view



Solution-centered

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

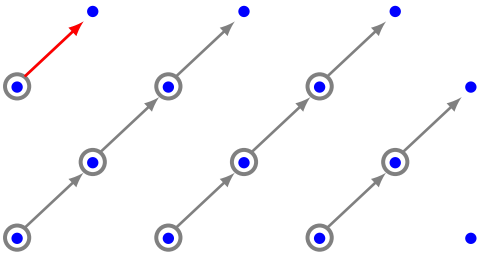
Neighborhood: solution-centered vs operation-centered point of view



Operation-centered

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

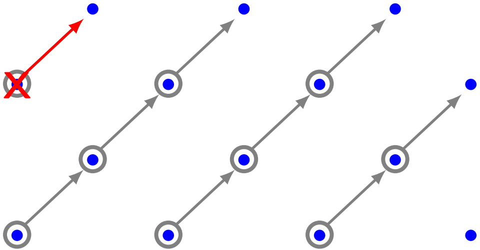
Neighborhood: solution-centered vs operation-centered point of view



Operation-centered

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

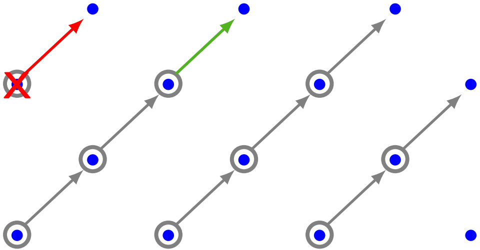
Neighborhood: solution-centered vs operation-centered point of view



Operation-centered

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

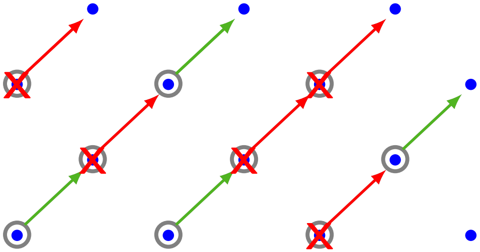
Neighborhood: solution-centered vs operation-centered point of view



Operation-centered

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

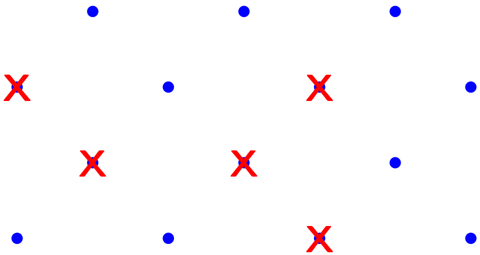
Neighborhood: solution-centered vs operation-centered point of view



Operation-centered

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

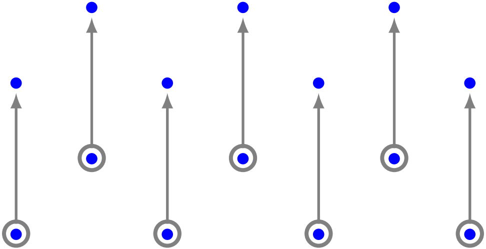
Neighborhood: solution-centered vs operation-centered point of view



Operation-centered

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

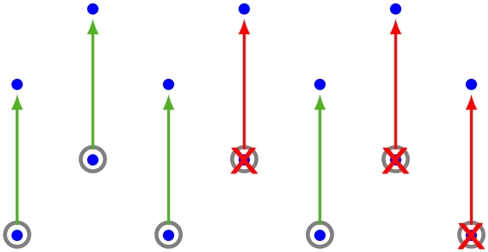
Neighborhood: solution-centered vs operation-centered point of view



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*

Outline

Linear formulation

- describe a partition (E, T)
  - express *"task i is early"* is needed
  - → introduce a binary variable δ_j for each task $j \in J$ $\delta_j = 1$ iff $j \in E$

Linear formulation

- describe a partition (E, T)
  - express *"task i is early"* is needed
  - → introduce a binary variable δ_j for each task $j \in J$ $\delta_j = 1$ iff $j \in E$
- express f as a linear function
  - express "both tasks i and j are early" is needed
  - → introduce a variable $X_{i,j}$ for each couple of tasks (i, j) with $i < j$
  - → introduce 4 inequalities for each (i, j) $X_{i,j} = 1$ iff $\delta_i \neq \delta_j$

Linear formulation

- describe a partition (E, T)
  - express *"task i is early"* is needed
  - $\rightarrow$ introduce a binary variable δ_j for each task $j \in J$ $\delta_j = 1$ iff $j \in E$
- express f as a linear function
  - express "both tasks i and j are early" is needed
  - $\rightarrow$ introduce a variable $X_{i,j}$ for each couple of tasks (i, j) with $i < j$
  - $\rightarrow$ introduce 4 inequalities for each (i, j) $X_{i,j} = 1$ iff $\delta_i \neq \delta_j$

$=$ **a compact linear formulation**

Linear formulation

- describe a partition (E, T)
  - express *"task i is early"* is needed
  - $\rightarrow$ introduce a binary variable δ_j for each task $j \in J$ $\delta_j = 1$ iff $j \in E$
- express f as a linear function
  - express "both tasks i and j are early" is needed
  - $\rightarrow$ introduce a variable $X_{i,j}$ for each couple of tasks (i, j) with $i < j$
  - $\rightarrow$ introduce 4 inequalities for each (i, j) $X_{i,j} = 1$ iff $\delta_i \neq \delta_j$

$=$ **a compact linear formulation**

- translate dominance constraints

Linear formulation

- describe a partition (E,T)
  - express *"task i is early"* is needed

  → introduce a binary variable δ_j for each task $j \in J$ $\delta_j = 1$ iff $j \in E$
- express f as a linear function
  - express "both tasks i and j are early" is needed

  → introduce a variable $X_{i,j}$ for each couple of tasks (i,j) with $i < j$

  → introduce 4 inequalities for each (i,j) $X_{i,j} = 1$ iff $\delta_i \neq \delta_j$

= **a compact linear formulation**

- translate dominance constraints

$$\Delta_u^{early}(E,T) \geqslant 0 \text{ if } u \in E$$

Linear formulation

- describe a partition (E, T)
  - express *"task i is early"* is needed
  - $\rightarrow$ introduce a binary variable δ_j for each task $j \in J$ $\delta_j = 1$ iff $j \in E$
- express f as a linear function
  - express "both tasks i and j are early" is needed
  - $\rightarrow$ introduce a variable $X_{i,j}$ for each couple of tasks (i, j) with $i < j$
  - $\rightarrow$ introduce 4 inequalities for each (i, j) $X_{i,j} = 1$ iff $\delta_i \neq \delta_j$

$=$ **a compact linear formulation**

- translate dominance constraints

$$\boxed{\Delta_u^{early}(E, T) \geqslant 0 \text{ if } u \in E}$$

ex: $-\alpha_u \, p\big(\underline{A(u) \cap E}\big) \quad \longrightarrow$

Linear formulation

- describe a partition (E,T)
  - express *"task i is early"* is needed

  $\rightarrow$ introduce a binary variable δ_j for each task $j \in J$ $\delta_j = 1$ iff $j \in E$
- express f as a linear function
  - express "both tasks i and j are early" is needed

  $\rightarrow$ introduce a variable $X_{i,j}$ for each couple of tasks (i,j) with $i < j$

  $\rightarrow$ introduce 4 inequalities for each (i,j) $X_{i,j} = 1$ iff $\delta_i \neq \delta_j$

$=$ **a compact linear formulation**

- translate dominance constraints

$$\Delta_u^{early}(E,T) \geqslant 0 \text{ if } u \in E$$

$$\text{ex: } -\alpha_u \, p\big(A(u) \cap E\big) \longrightarrow -\alpha_u \sum_{i \in A(u)} p_i \delta_i$$

Linear formulation

- describe a partition (E,T)
  - express *"task i is early"* is needed
  - $\rightarrow$ introduce a binary variable δ_j for each task $j \in J$ $\delta_j = 1$ iff $j \in E$
- express f as a linear function
  - express "both tasks i and j are early" is needed
  - $\rightarrow$ introduce a variable $X_{i,j}$ for each couple of tasks (i,j) with $i < j$
  - $\rightarrow$ introduce 4 inequalities for each (i,j) $X_{i,j} = 1$ iff $\delta_i \neq \delta_j$

= a compact linear formulation

- translate dominance constraints

$$\boxed{\Delta_u^{early}(E,T) \geqslant 0 \text{ if } u \in E} \longrightarrow \boxed{\Delta_u^{early}(\delta) \geqslant 0 \text{ if } u \in E}$$

$$\text{ex: } -\alpha_u\, p\big(A(u) \cap E\big) \longrightarrow -\alpha_u \sum_{i \in A(u)} p_i \delta_i$$

Linear formulation

- describe a partition (E, T)
  - express *"task i is early"* is needed
  - $\rightarrow$ introduce a binary variable δ_j for each task $j \in J$ $\delta_j = 1$ iff $j \in E$
- express f as a linear function
  - express "both tasks i and j are early" is needed
  - $\rightarrow$ introduce a variable $X_{i,j}$ for each couple of tasks (i, j) with $i < j$
  - $\rightarrow$ introduce 4 inequalities for each (i, j) $X_{i,j} = 1$ iff $\delta_i \neq \delta_j$

= a compact linear formulation

- translate dominance constraints

$$\boxed{\Delta_u^{early}(E, T) \geqslant 0 \text{ if } u \in E} \longrightarrow \boxed{\Delta_u^{early}(\delta) \geqslant -M(1 - \delta_u)}$$

$$\text{ex: } -\alpha_u\, p\big(\underline{A(u) \cap E}\big) \longrightarrow -\alpha_u \sum_{i \in A(u)} p_i \delta_i$$

Experimental results

20	6
50	1835
60	1/10

Framework:

- machine
  RAM 144 Go
  1 core at 3.47 Ghz
- PL Solver
  Cplex 12.6.3
- time limit
  3600s

Benchmark:

- by Biskup&Feldmann
- $n \in \{10, 20, 30, 40, 50, 60\}$
- 10 instances for each n
- p, α and β integers
- $p_i \in [1, 20]$

Experimental results

20	6
50	1835
60	1/10

insert ineq. →

20	7
50	32
100	7/10

Framework:

- machine
  RAM 144 Go
  1 core at 3.47 Ghz
- PL Solver
  Cplex 12.6.3
- time limit
  3600s

Benchmark:

- by Biskup&Feldmann
- $n \in \{10, 20, 30, 40, 50, 60\}$
- 10 instances for each n
- p, α and β integers
- $p_i \in [1, 20]$

Experimental results

20	7
50	32
100	323
120	868

swap ineq.

20	6
50	1835
60	1/10

insert ineq.

20	7
50	32
100	7/10

Framework:

- machine
  RAM 144 Go
  1 core at 3.47 Ghz
- PL Solver
  Cplex 12.6.3
- time limit
  3600s

Benchmark:

- by Biskup&Feldmann
- $n \in \{10, 20, 30, 40, 50, 60\}$
- 10 instances for each n
- p, α and β integers
- $p_i \in [1, 20]$

Experimental results

20	7
50	32
100	323
120	868

insert ineq. →

20	7
50	15
100	75
150	865
180	9/10

swap ineq. ↑

20	6
50	1835
60	1/10

insert ineq. →

20	7
50	32
100	7/10

swap ineq. ↑

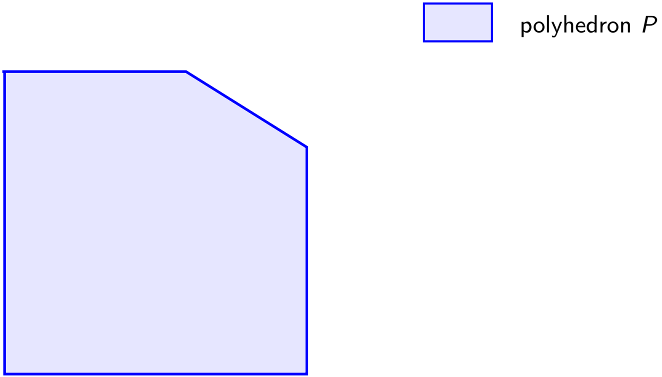
Framework:

- machine
  RAM 144 Go
  1 core at 3.47 Ghz
- PL Solver
  Cplex 12.6.3
- time limit
  3600s

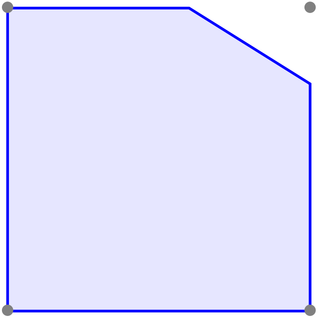
Benchmark:

- by Biskup&Feldmann
- $n \in \{10, 20, 30, 40, 50, 60\}$
- 10 instances for each n
- p, α and β integers
- $p_i \in [1, 20]$

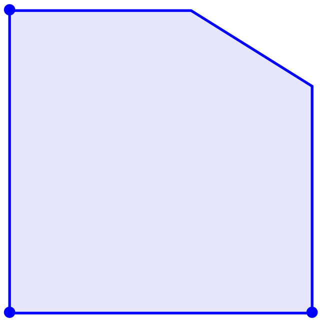
Unusual inequalities

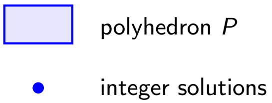


Unusual inequalities

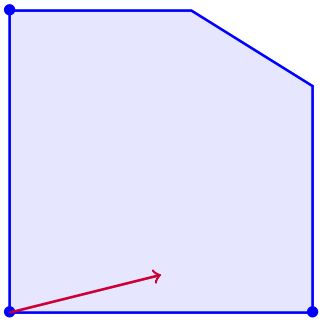


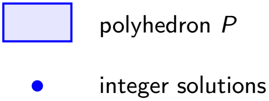
Unusual inequalities



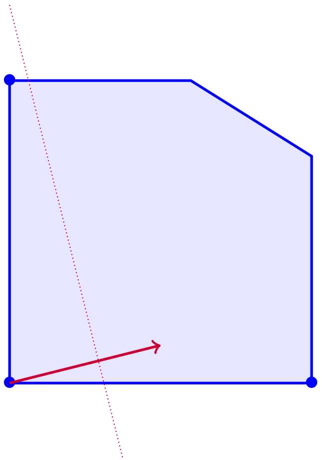


Unusual inequalities





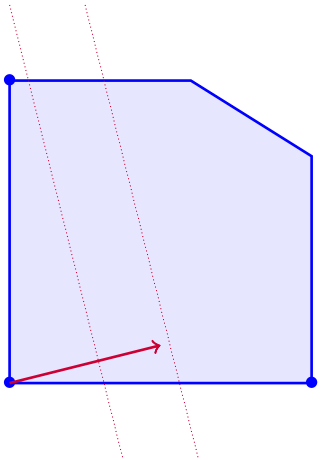
Unusual inequalities

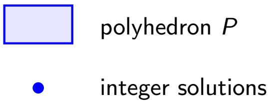


polyhedron P

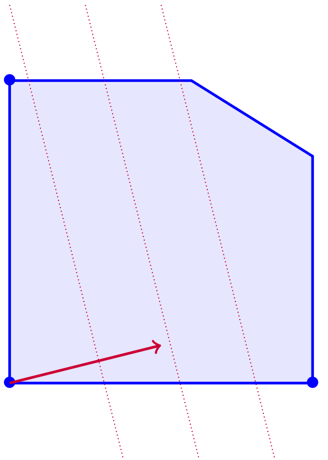
integer solutions

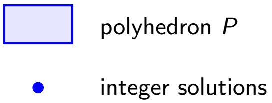
Unusual inequalities



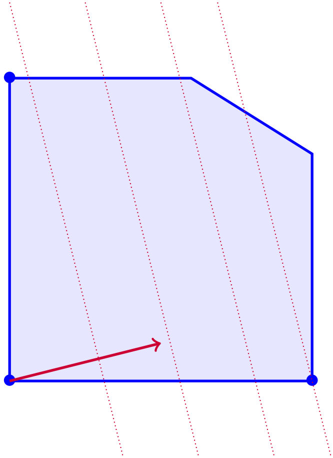


Unusual inequalities





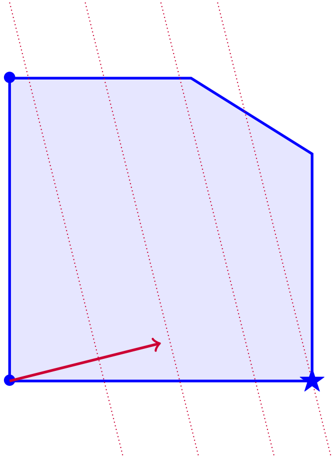
Unusual inequalities



polyhedron P

integer solutions

Unusual inequalities

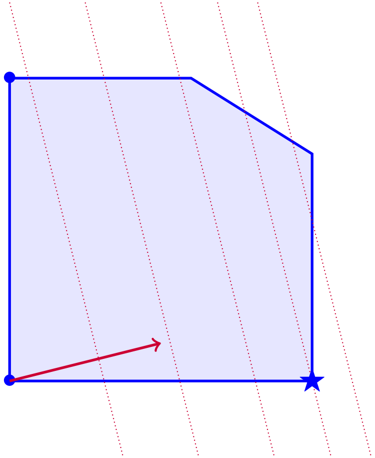


polyhedron P

integer solutions

best integer solution

Unusual inequalities

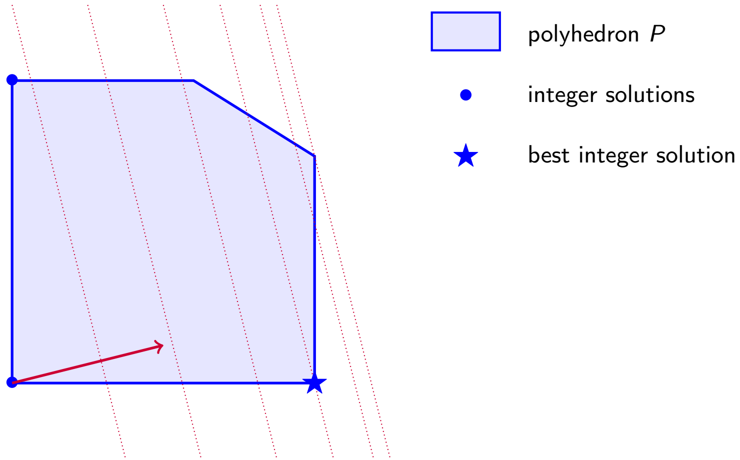


polyhedron P

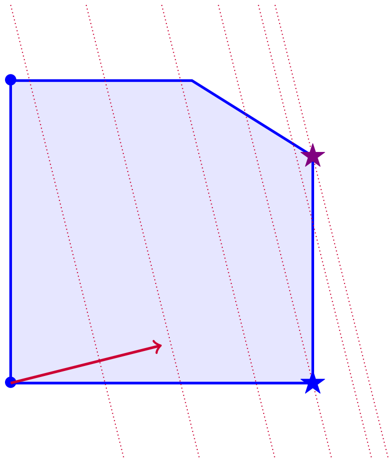
integer solutions

best integer solution

Unusual inequalities

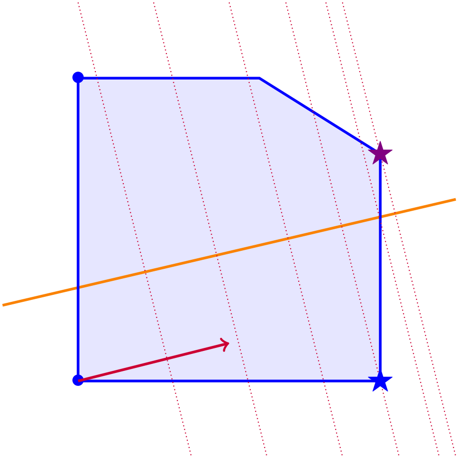


Unusual inequalities



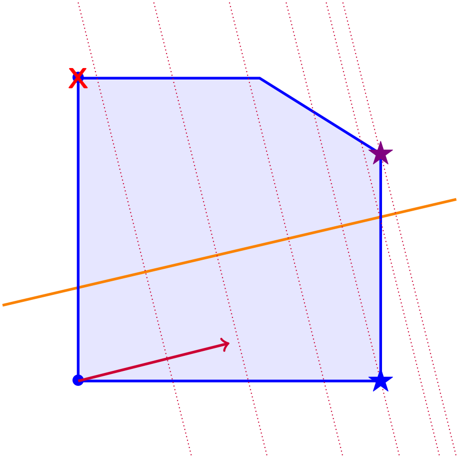
- polyhedron P
- integer solutions
- best integer solution
- best fractionnal solution in P

Unusual inequalities



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

Unusual inequalities



polyhedron P

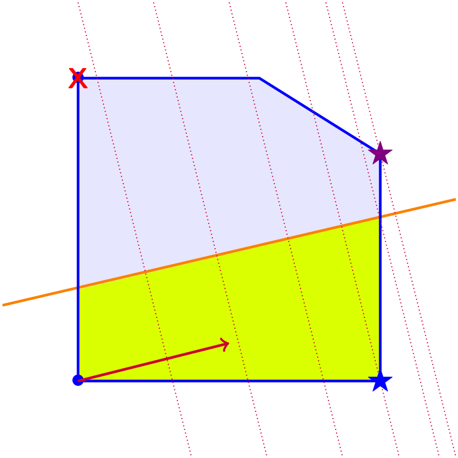
integer solutions

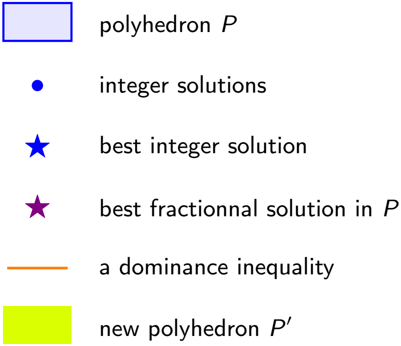
best integer solution

best fractionnal solution in P

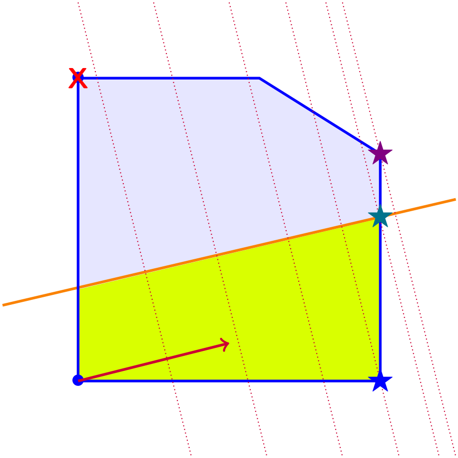
a dominance inequality

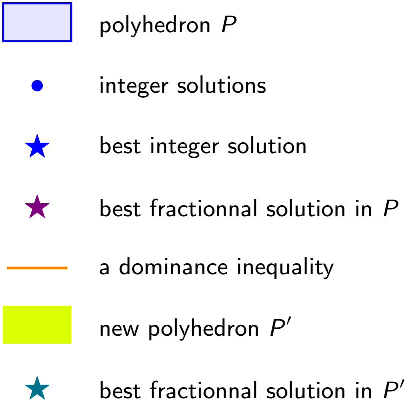
Unusual inequalities





Unusual inequalities





Outline

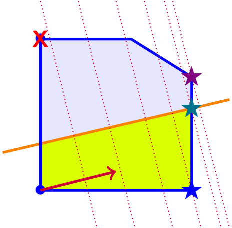
We propose linear inequalities

We propose linear inequalities

- which improves the linear formulation

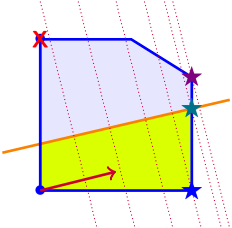
We propose linear inequalities

- which improves the linear formulation
- in a non classical way

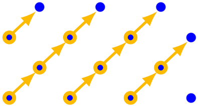


We propose linear inequalities

- which improves the linear formulation
- in a non classical way

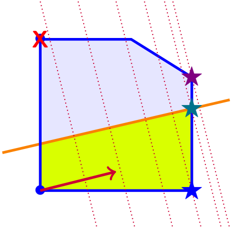


- by translating neigborhood-based dominance properties

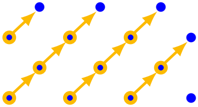


We propose linear inequalities

- which improves the linear formulation
- in a non classical way



- by translating neigborhood-based dominance properties



Future work:

- applying the dominance inequalities principle to other combinatorial problems

References I

D. Biskup and M. Feldmann.
Benchmarks for scheduling on a single machine against restrictive and unrestrictive common due dates.
Computers and operations research, Vol 28:787–801, 2001.

A. Falq, P. Fouilhoux, and S. Kedad-Sidhoum.
Mixed integer formulations using natural variables for single machine scheduling around a common due date.
CoRR, abs/1901.06880, 2019.

N. G. Hall and M. E. Posner.
Earliness-tardiness scheduling problems, 1: Weighted deviation of completion times about a common due date.
Operations Research, Vol 39:836–846, Sep-Oct 1991.

J. A. Hoogeveen and S. van de Velde.
Scheduling around a small common due date.
European Journal of Operational Research, Vol 55:237–242, 1991.

References II

A. Jouglet and J. Carlier.

Dominance rules in combinatorial optimization problems.

European Journal of Operational Research, 212(3):433–444, 2011.

J. J. Kanet.

Minimizing the average deviation of job completion times about a common due date.

Naval Research Logistics Quarterly, Vol 28:643–651, Dec 1981.

F. Sourd.

New exact algorithms for one-machine earliness-tardiness scheduling.

INFORMS Journal on Computing, 21(1):167–175, 2009.