

Filtering for Continuous Multi-Resources *cumulative* Constraint with Resource Consumption - Production

Emmanuel Poder¹ - Nicolas Beldiceanu²

¹ Emmanuel.Poder@laposte.net

² LINA UMR CNRS 6241, École des Mines de Nantes, France
Nicolas.Beldiceanu@emn.fr

Outline

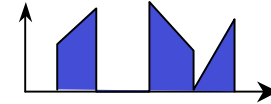
- Context and contributions
- Task model and piecewise linear cumulative *cumulatives_pwl*
- Filtering a task according to a minimum cumulated profile
 - Minimum cumulated resource profile
 - Filtering of the **resource** assignment
 - Filtering of the **temporal** attributes
 - Filtering of the **heights** of the sub-tasks
- Conclusion and perspectives

Context and contributions

Context: cumulative scheduling and producer/consumer model

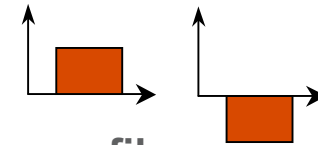
- (Poder, PhD 2002) (Poder *et al.*, EJOR 2004):

- Task made of a sequence of **positive** trapezoids ; **one single** resource
- **Compulsory part and Minimum cumulated profile**
- Implemented in CHIP (*cumulative_trapeze* constraint)
- -> Main applications: cumulative scheduling, producer/consumer model



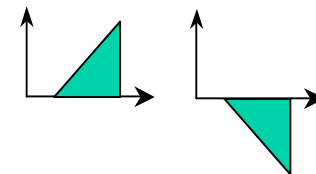
- (Beldiceanu and Carlsson, CP - 2002):

- Positive or negative **rectangle** task ; **Multi-resources**
- Notions of **minimum and maximum cumulated resource profiles**
- Implemented in SICStus



- (Sourd and Rogerie, EJOR- 2004): Continuous filling and emptying storage systems

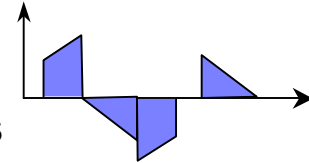
- Positive or negative triangular task
- Minimum and maximum levels
- Implemented in ILOG scheduler



Context and contributions

- (Beldiceanu and Poder , CPAIOR - 2007):

- Task made of a sequence of **positive** or **negative** trapezoids
- **Multi-resources**
- **Minimum and maximum cumulated resource profiles** build using a sweep algorithm



Contribution to ICAPS - 2008:

Filtering algorithm for the task model presented in CPAIOR - 2007

i.e. given a resource and a task, how to filter

- the **temporal attributes** of such a task
- the **resource assignment** of such a task

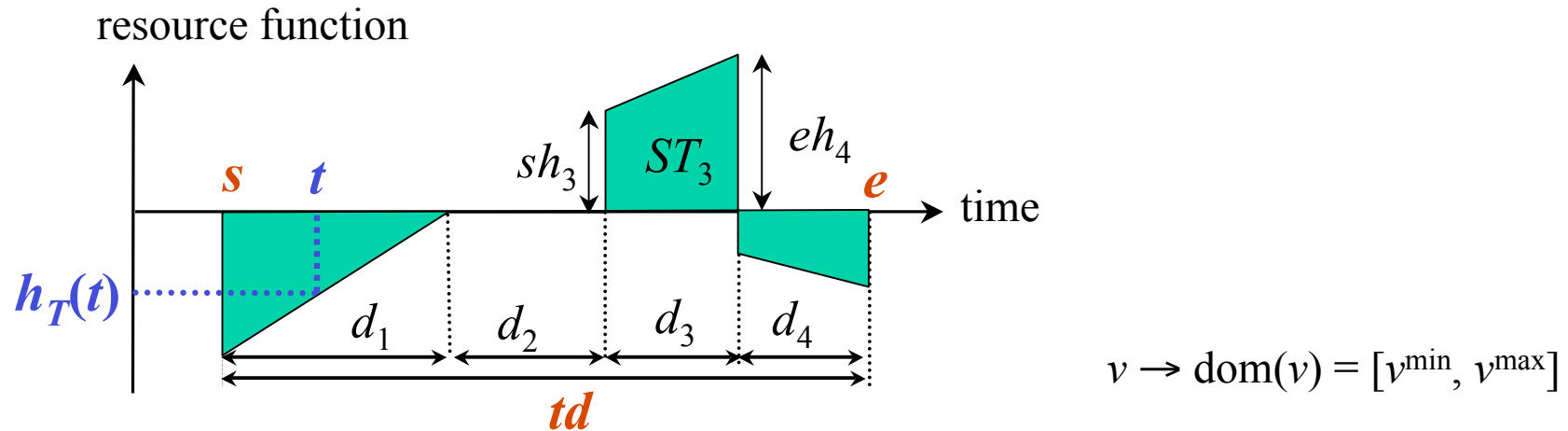
according to the minimum cumulated profile of the resource

Task model and piecewise linear cumulative *cumulatives_pwl*

Definition (Cf CPAIOR07):

A task $T = (s, td, e, Seq, a)$ is defined by:

- a **start** s , an **end** e , a total **duration** td and a set a of **possible resource assignments**
- a **positive or negative piecewise linear** resource function h_T represented by a **sequence** Seq of p consecutive **trapezoid** sub-tasks ST_1, ST_2, \dots, ST_p where ST_i has a start height sh_i , an end height eh_i and a duration d_i .



Difficulties and interest of the model:

- All attributes of a task may be **not fixed**
- **Multi**-resources
- Sequence of trapezoid sub-tasks with **variable** durations where their **sum** is equal to the duration of the task
 - ⇒ *no need to define relations of precedence between sub-tasks **outside***
- Trapezoid sub-tasks with **positive** heights and trapezoid sub-tasks with **negatives** heights in a same task

Task model and piecewise linear cumulative *cumulatives_pwl*

Given :

- **Tasks**, a collection of Tasks T_1, T_2, \dots, T_n where $T_i = (s_i, td_i, e_i, Seq_i, a_i)$ (start, total duration, end, Sequence of sub-tasks, assignment)
- **Resources**, a set of integers C_1, C_2, \dots, C_q where C_k is:
 - The capacity of the resource k if Constraint = ' \leq '
 - The minimum level to reach of the resource k if Constraint = ' \geq '
- **Constraint**, one of the constraint ' \leq ' or ' \geq '

The constraint *cumulatives_pwl*(Tasks, Resources, Constraint) **holds** iff:

$$1. \forall i = 1..n, s_i + td_i = e_i \qquad 2. \forall i = 1..n, \sum_{j=1}^{p_i} d_j = td_i$$

$$3. \text{Case } \leq : (\forall k = 1..q)(\forall t \in R) \sum_{i/a_i=\{k\}} h_{T_i}(t) \leq C_k$$

$$\text{Case } \geq : (\forall k = 1..q)(\forall t \in R/\exists i, t \in [s_i, e_i[) \sum_{i/a_i=\{k\}} h_{T_i}(t) \geq C_k$$

Filtering a task according to a minimum cumulated profile

B - Notion of minimum cumulated profile

Definition: The **minimum cumulated profile** $mcrP(r)$ of a resource r is such

that its resource function verifies for any t :
$$h_{mcrP(r)}(t) = \sum_{T_i / r \in a_i} h_{mP(T_i)}(t)$$

where $mP(T_i)$ is the **minimum profile** of the task T_i and is build from the

compulsory part $CP(T_i^+)$ of the positive sub-tasks T_i^+ of T_i and of the envelope

$Env(T_i^-)$ of the negative sub-tasks T_i^- of T_i :
$$h_{mP(T_i)}(t) = h_{CP(T_i^+)}(t) + h_{Env(T_i^-)}(t)$$

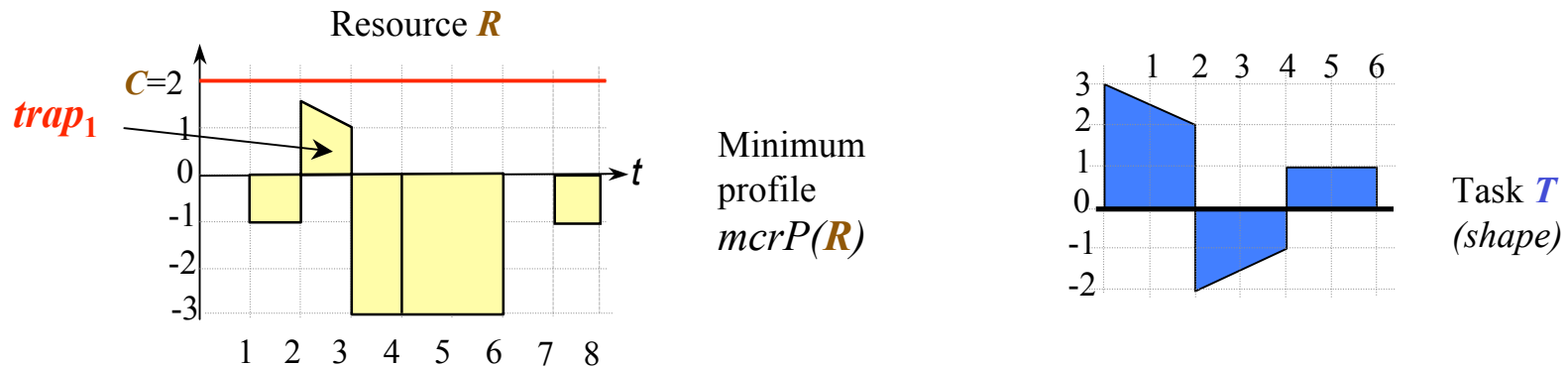
Remarks:

- For any t , we can't have both $h_{CP(T_i^+)}(t) \neq 0$ and $h_{Env(T_i^-)}(t) \neq 0$
- If T_i is assigned to the resource r then **both** $CP(T_i^+)$ and $Env(T_i^-)$ are taken into account within $mcrP(r)$. *Else*, if T_i is not yet assigned to the resource r then **only** $Env(T_i^-)$ are taken into account within $mcrP(r)$

Filtering a task according to a minimum cumulated profile

From now on, we consider:

- A resource R with maximum capacity C
- A fixed trapezoid $trap_1$ of the minimum cumulated profile $mcrP(R)$ of the resource R .
- A task T that is or may be assigned to the resource R and that is or may be executed in parallel with $trap_1$



We are interested in:

- A - Filtering the resource assignment of the task T
- B - Filtering its temporal attributes (start, duration, end and duration of its sub-tasks)
- C - Filtering the heights of its sub-tasks
according to the minimum resource profile of the resource R

Filtering a task according to a minimum cumulated profile

B - Filtering of the resource assignment

We distinguish **two cases**

- T has a **known contribution** already registered within $mcrP(R)$. *If this contribution is absolutely required to avoid a capacity overflow then:*
 - We **assign** T to the resource R
 - We filter the latest start and the earliest end of T so as to **they include** $trap_1$
- There exists **no registered contribution** of T within $mcrP(R)$. *If the assignment of T to the resource R would lead to a capacity overflow then we **remove** R from the resource assignment variable of T*

Filtering a task according to a minimum cumulated profile

C - Filtering of the **temporal** attributes of a task

Two phases:

1. We browse all sub-tasks of T and, for each such sub-task, we compute the interval of non feasible values for its starts and duration according to $trap_1$
2. We propagate back these non-feasible values the others temporal attributes using the following linear constraints (**holes within the origin/end of sub-tasks matter**) :
 - (1) *The sum of the durations of all the sub-tasks is equal to the total duration of the task,*
 - (2) *The start plus the total duration of the task is equal to its end,*
 - (3) *The start of a sub-task plus its duration is equal to the start of the next sub-task.*

Filtering a task according to a minimum cumulated profile

C - Filtering of the **temporal** attributes of a task (phase 1)

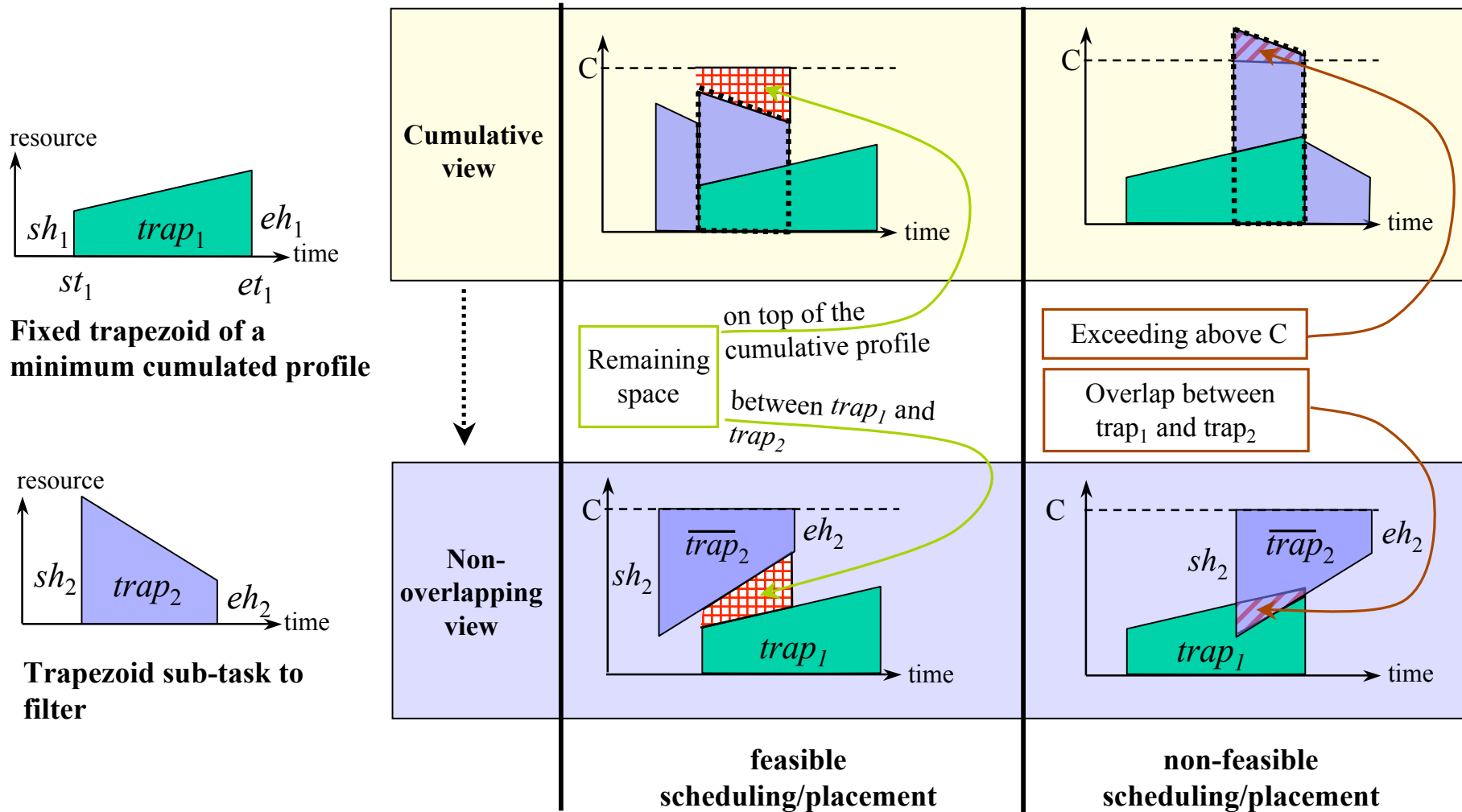
In the following:

- $trap_2$ denotes the sub-task of T we want to filter according to $trap_1$
- assumes that the contribution of T within $trap_1$ has been removed before the filtering

To filter the attributes of $trap_2$, we first **transform the problem** of scheduling $trap_2$ according to $trap_1$ without exceeding C in a particular **non-overlapping problem**

Filtering a task according to a minimum cumulated profile

C-1. **transforming** a *cumulative scheduling problem* into a particular *non-overlapping problem*

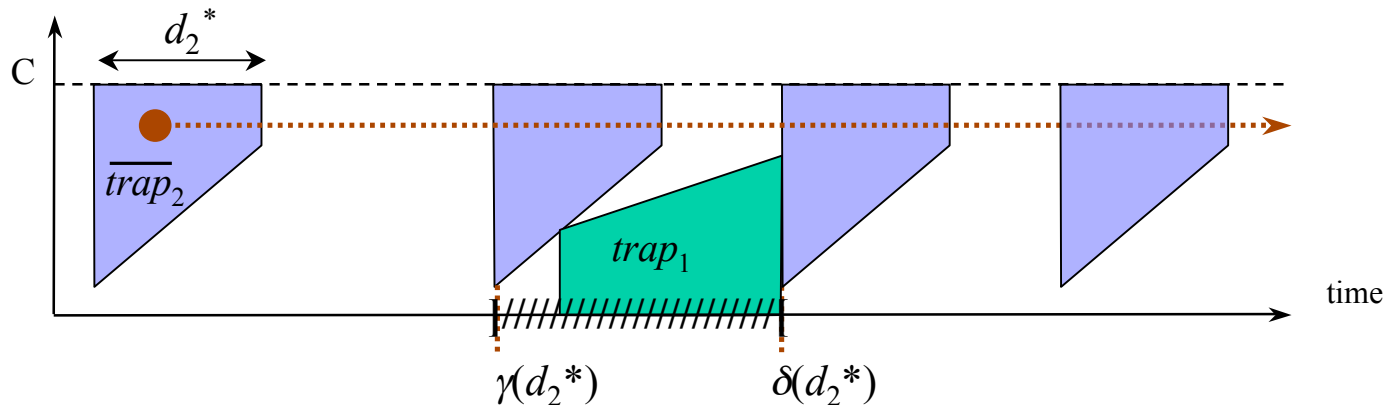


Filtering a task according to a minimum cumulated profile

C-2. Computing the interval of non feasible starts of $trap_2$

- Case when **the duration** of $trap_2$ **is fixed** to d_2^* :

We translate $\overline{trap_2}$ according to $trap_1$ and C



Last feasible start for $trap_2$ before $trap_1$
i.e. the **earliest possible overlapping**

First feasible start for $trap_2$ after $trap_1$
i.e. the **latest possible overlapping**

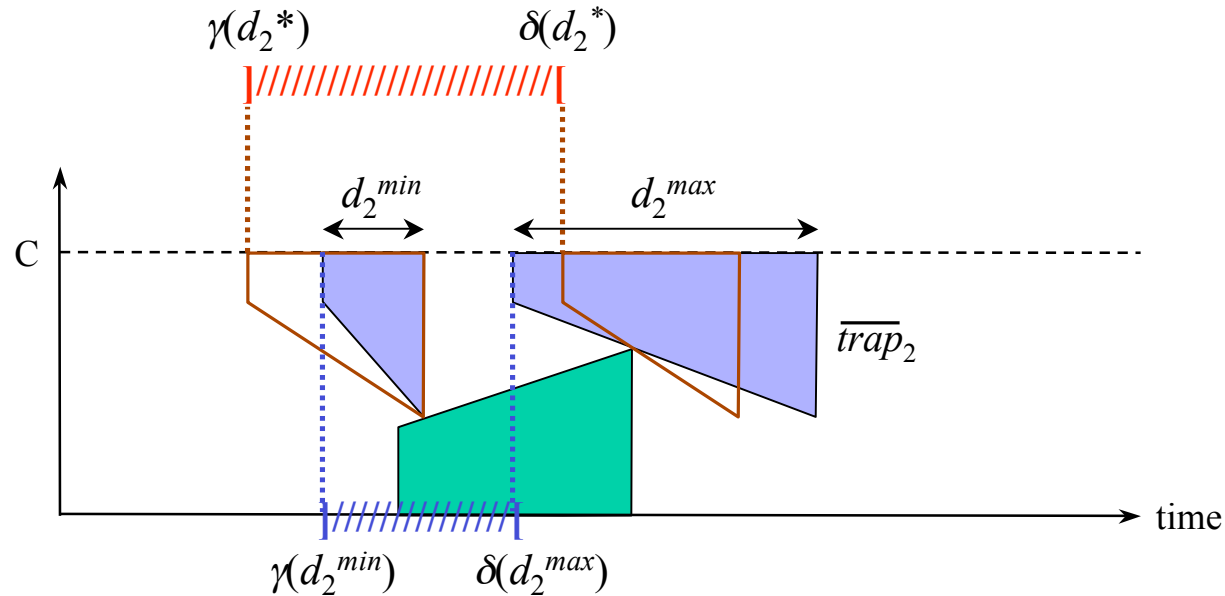
The non feasible starts of $trap_2$ is the interval $] \gamma(d_2^*), \delta(d_2^*) [$

Remark : five cases for the computation of $\gamma(d_2^*)$ and five cases for the computation of $\delta(d_2^*)$

Filtering a task according to a minimum cumulated profile

C-2. Computing the interval of non feasible starts of $trap_2$

- Case when **the duration** of $trap_2$ is **not fixed** but $d_2 \in [d_2^{min}, d_2^{max}]$:



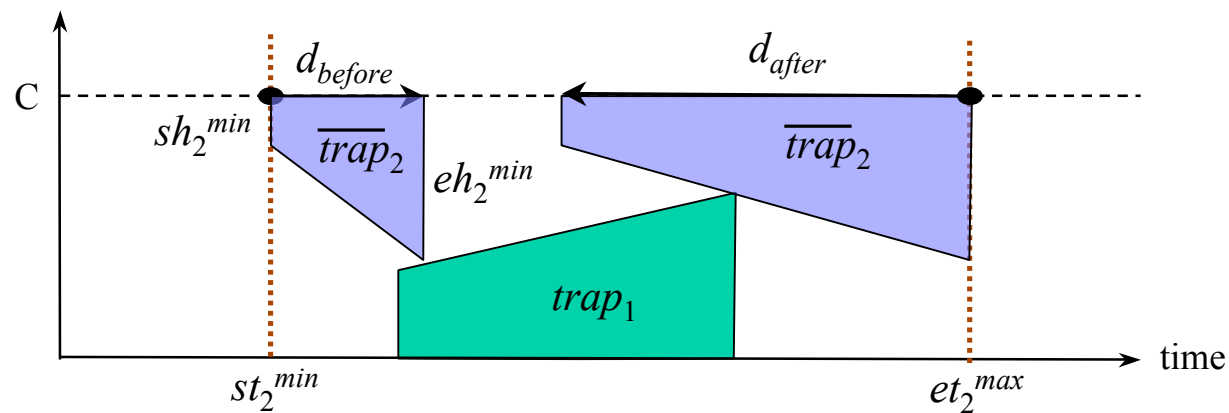
The non feasible starts of $trap_2$ is the interval $[\gamma(d_2^{min}), \delta(d_2^{max})]$

Remark : The **earliest start** st_2^{min} and the **latest end** et_2^{max} can **also** be taken into account for the computation of this interval (**more pruning**).

Filtering a task according to a minimum cumulated profile

C-3. Computing the interval of non feasible durations of $trap_2$

1. We fix the heights sh_2 and eh_2 of \overline{trap}_2 at their minimum respective value sh_2^{\min} and eh_2^{\min}
2. We place \overline{trap}_2 with $d_2=d_2^{\min}$, if possible, before $trap_1$ at its earliest start st_2^{\min} and stretch its duration as much as possible (i.e., without overlapping) -> we obtain a duration d_{before}
3. We place \overline{trap}_2 with $d_2=d_2^{\min}$, if possible, after $trap_1$ at its latest end et_2^{\max} and stretch its duration as much as possible (i.e., without overlapping) -> we obtain a duration d_{after}

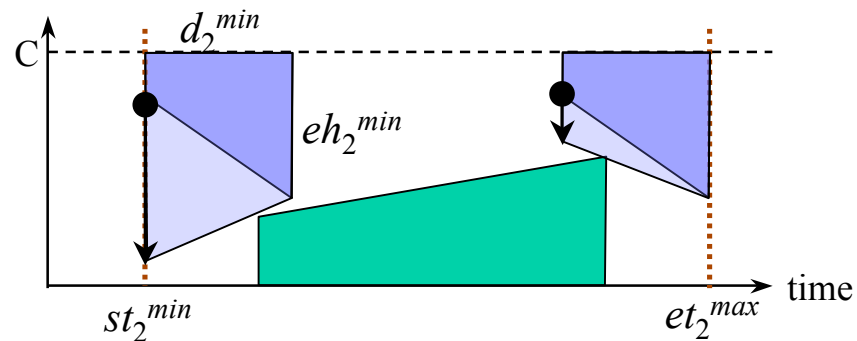


The non feasible durations of $trap_2$ is $] \max(d_{before}, d_{after}), +\infty [$

Filtering a task according to a minimum cumulated profile

D. Computing the interval of non feasible heights of $trap_2$

- Non feasible values for the **start height** sh_2 of $trap_2$:
 1. We fix the duration d_2 and the end height eh_2 of \overline{trap}_2 at their minimum respective value d_2^{\min} and eh_2^{\min}
 2. We place \overline{trap}_2 , if possible, before $trap_1$ at its earliest start st_2^{\min} and stretch its start height as much as possible (i.e., without overlapping) -> we obtain the value sh_{before}
 3. We place \overline{trap}_2 , if possible, after $trap_1$ at its latest end et_2^{\max} and stretch its start height as much as possible (i.e., without overlapping) -> we obtain the value sh_{after}



The non feasible values for sh_2 is $]\max(sh_{before}, sh_{after}), +\infty [$

- Non feasible values for the end height eh_2 of $trap_2$: Similar

Conclusion and Perspectives

Results:

- Efficient algorithm for pruning all tasks attributes

Perspectives:

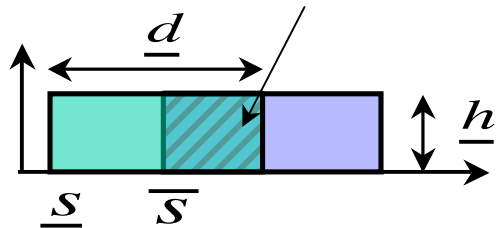
- Generalize edge finding to this task model

QUESTION: *is it possible to have a **polynomial** algorithm which gives the **exact** minimum intersection of a task (with **non-fixed** attributes) with a fixed interval ?*

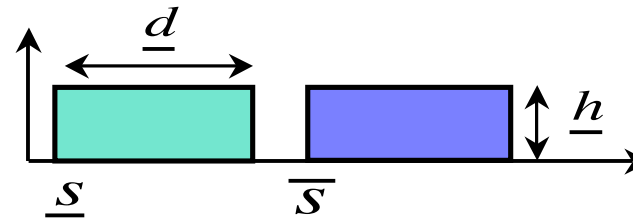
Compulsory Part, Envelope, minimum cumulated profile

Definition: the *compulsory part* (*CP*) of a positive task is the intersection of all its feasible instances (A. Lahrichi, 1979)

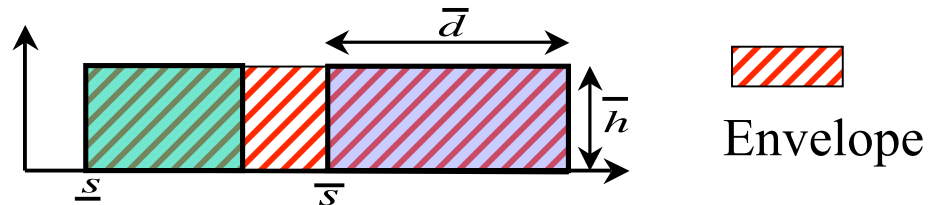
Non empty Compulsory Part



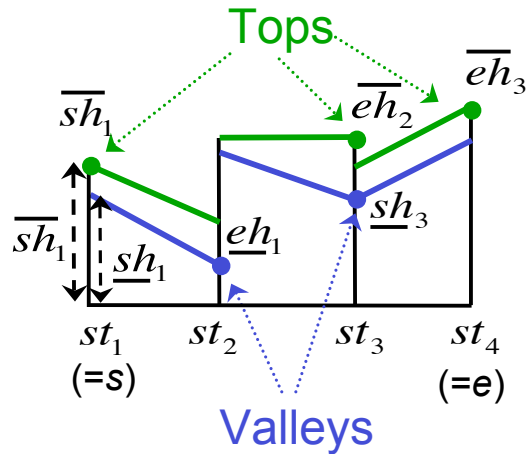
Empty Compulsory Part



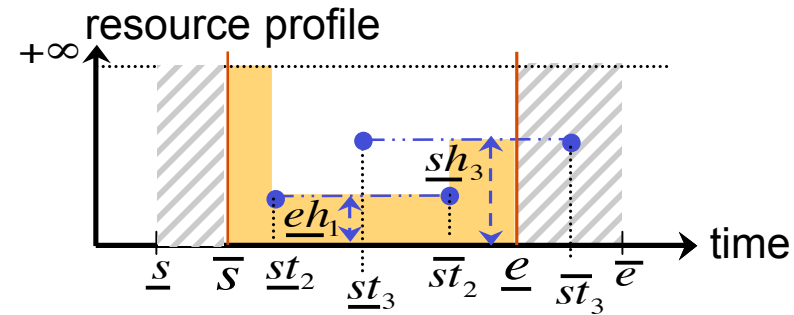
Definition: The *enveloppe* (*Env*) of a positive task is the union of all its feasible instances.



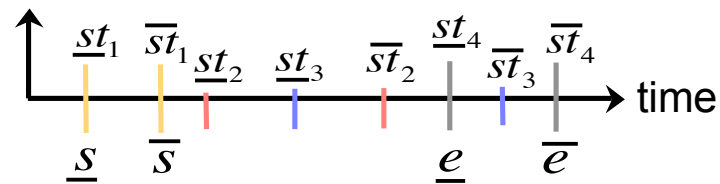
Compulsory Part and Envelope of a positive task



(A) Task T

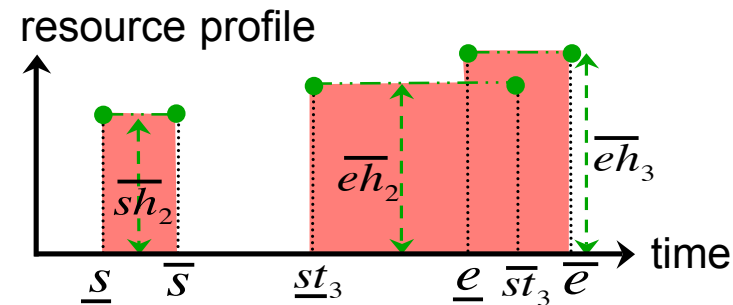


(C) Level Valley Task on $[\bar{s}, \bar{e}[$



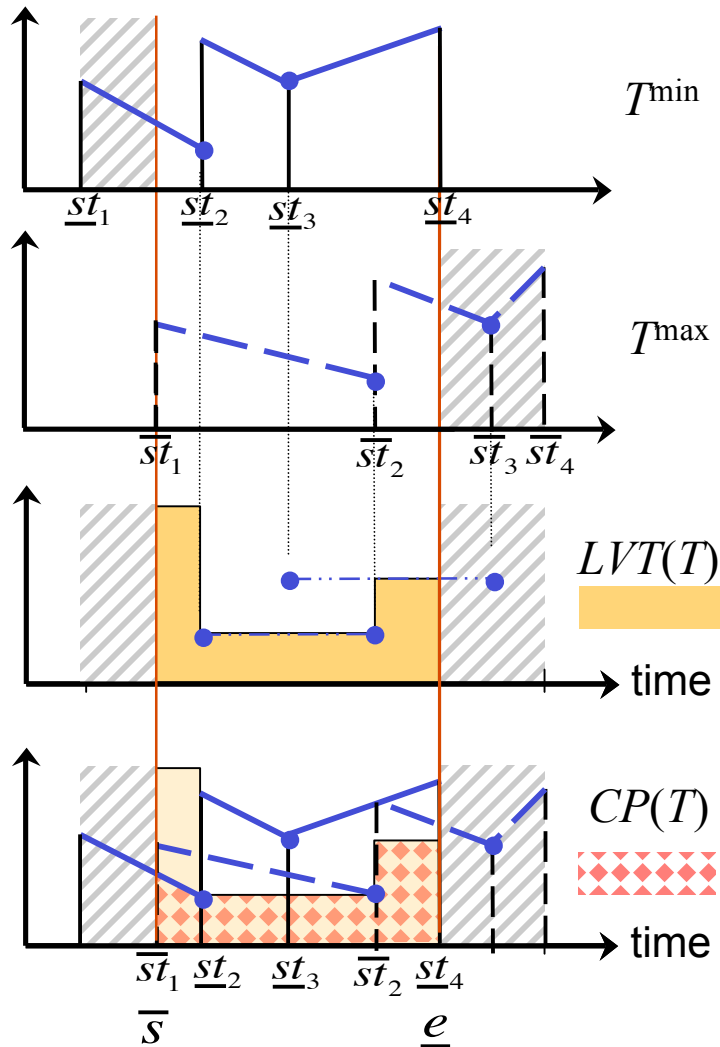
(B) Earliest and latest starts of each st_i

-> Give T^{min} and T^{max}

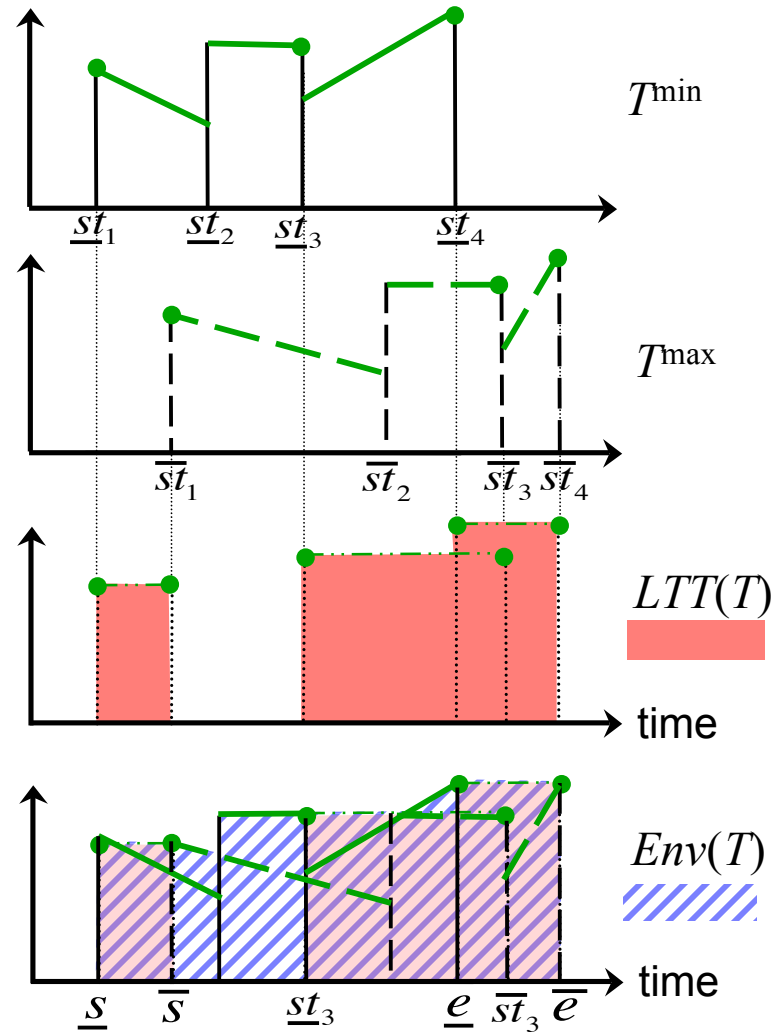


(D) Level Top Task on $[\underline{s}, \bar{e}]$

Compulsory Part and Envelope of a positive task



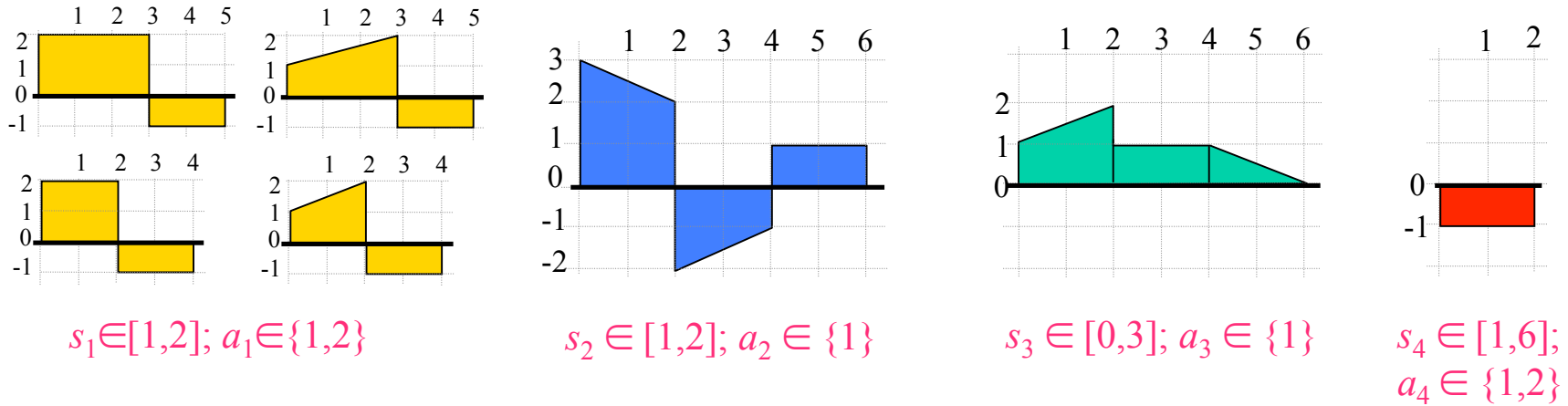
$$CP(T) = T^{\min} \cap T^{\max} \cap LVT(T)$$



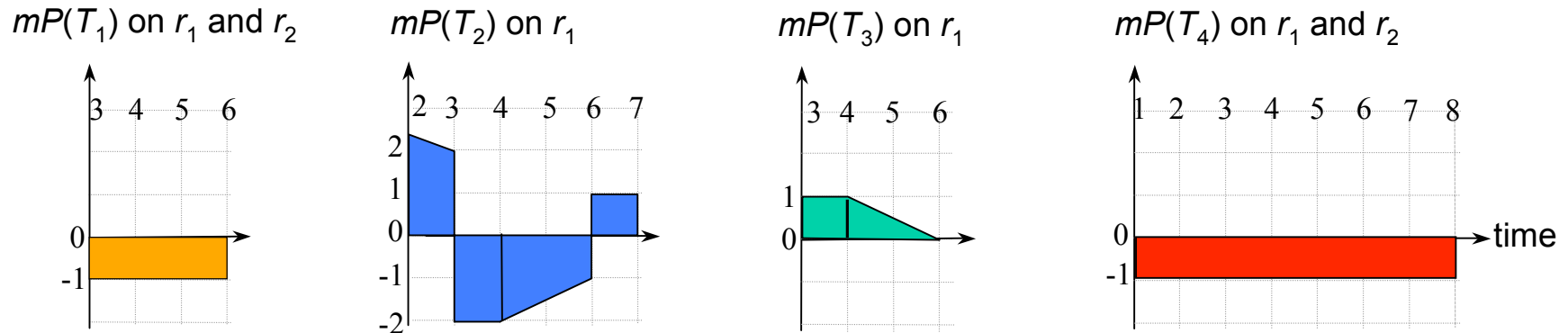
$$Env(T) = T^{\min} \cup T^{\max} \cup LTT(T)$$

Minimum and maximum profiles of a task

Tasks and their feasible sub-sequences:

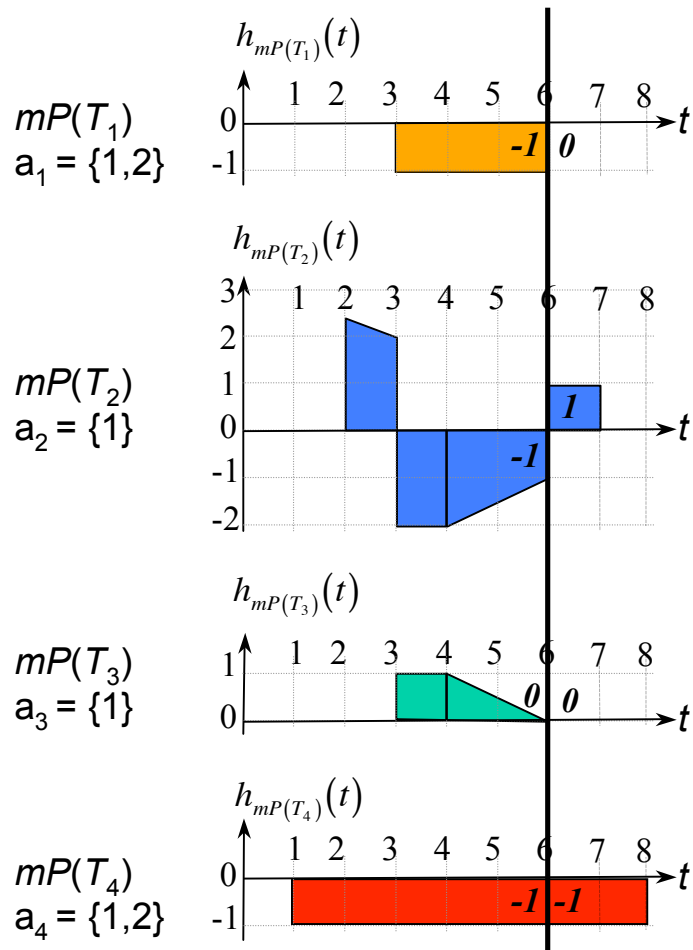


Minimum profiles mP of the four tasks on the resources r_1 et r_2 :



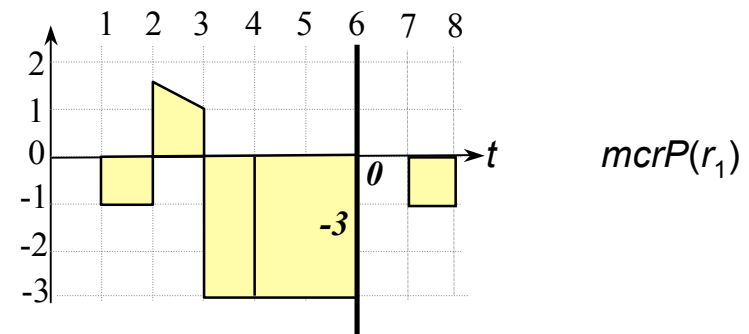
Minimum and maximum cumulated resource profiles

Minimum Profiles of tasks T_1, T_2, T_3 and T_4 on resources 1 and 2:



Minimum cumulated profiles of resources 1 and 2:

$$h_{mcrP(r_1)}(t) = h_{mP(T_1)}(t) + h_{mP(T_2)}(t) + h_{mP(T_3)}(t) + h_{mP(T_4)}(t)$$



$$h_{mcrP(r_2)}(t) = h_{mP(T_1)}(t) + h_{mP(T_4)}(t)$$

