Generalizing the Edge-Finder Rule for the Cumulative Constraint

Vincent Gingras, Claude-Guy Quimper

Abstract

We present two novel filtering algorithms for the **Cumulative constraint** based on a new energetic **relaxation**. We introduce a **generalization** of the **Overload Check** and **Edge-Finder** rules based on a function computing the earliest completion time for a set of tasks. Depending on the **relaxation** used to compute this function, one obtains different levels of filtering. We present two algorithms that enforce these rules. The algorithms utilize a **novel data structure** that we call **Profile** and that encodes the resource utilization over time. Experiments show that these algorithms are **competitive** with the state-of-the-art algorithms, by doing **a greater filtering** and **having a faster runtime**.

Cumulative Scheduling Problem

- **Definition :** A set of tasks need to be executed, without interruption, on a cumulative resource of capacity $C \in \mathbb{Z}^+$
- Properties of a non-preemptive task $i \in \mathcal{I} = \{1, \dots, n\}$
  - earliest starting time : $\text{est}_i \in \mathbb{Z}$
  - processing time : $p_i \in \mathbb{Z}^+$
  - energy : $e_i = p_i h_i$
  - latest starting time : $\text{lst}_i = \text{lct}_i - p_i$
  - latest completion time : $\text{lct}_i \in \mathbb{Z}$
  - resource consumption value : $h_i \in \mathbb{Z}^+$
  - earliest completion time : $\text{ect}_i = \text{est}_i + p_i$
- Generalized properties to a set of tasks Ω :

$$\text{est}_\Omega = \min_{i \in \Omega} \text{est}_i \qquad \text{lct}_\Omega = \max_{i \in \Omega} \text{lct}_i \qquad e_\Omega = \sum_{i \in \Omega} e_i$$

- Cumulative constraint :

$$\forall i \in \mathcal{I} \, \text{dom}(S_i) = [\text{est}_i, \text{lst}_i]$$

$$\textsc{Cumulative}([S_1, \dots, S_n], C) \iff \forall t : \sum_{i \in \mathcal{I}, S_i \le t < S_i + p_i} h_i \le C$$

Overload Check

- If the energy consumption required by a set of tasks Ω exceeds the capacity over $[\text{est}_\Omega, \text{lct}_\Omega)$, then the test fails.

$$\exists \Omega \subseteq \mathcal{I} : C(\text{lct}_\Omega - \text{est}_\Omega) < e_\Omega \Rightarrow \text{fail}$$

- [Fahimi et al., 2014] run the Overload Check in $\mathcal{O}(n)$ time

Edge-Finder

1) **Detection Phase**
   - Detects "ends before end" ($\lessdot$) temporal relation
   - [Vilím, 2009] runs the Detection Phase in $\mathcal{O}(n \log n)$ time
   - If a task $i \notin \Omega$ cannot be executed along the tasks in Ω without having any of them missing their deadline, then $\Omega \lessdot i$

$$e_{\Omega \cup \{i\}} > C(\text{lct}_\Omega - \text{est}_{\Omega \cup \{i\}}) \Rightarrow \Omega \lessdot i$$

2) **Adjustment phase**
   - Given a precedence $\Omega \lessdot i$, adjusts the lower bound of S_i
   - [Vilím, 2009] runs the Adjustment Phase in $\mathcal{O}(kn \log n)$ time, where k is the number of distinct heights

$$\Omega \lessdot i \Rightarrow \text{est}_i \ge \max_{\Omega' \subseteq \Omega} \left\{ \text{est}_{\Omega'} + \left\lceil \frac{e_{\Omega'} - (C - h_i)(\text{lct}_{\Omega'} - \text{est}_{\Omega'})}{h_i} \right\rceil \right\}$$

Fully-Elastic Relaxation

- Revolves around the elasticity of a task [Baptiste, Le Pape, Nuijten, 2001]
- The resource consumption of a fully elastic task can fluctuate over time
- Fully-Elastic computation of ect_Ω [Vilím, 2009]

$$\text{ect}^{\text{F}}_\Omega = \left\lceil \frac{\max\{C \, \text{est}_{\Omega'} + e_{\Omega'} \mid \Omega' \subseteq \Omega\}}{C} \right\rceil$$

Generalization of known filtering rules

- Overload Check

$$\exists \Omega \subseteq \mathcal{I} : \text{ect}_\Omega > \text{lct}_\Omega \implies \text{fail}$$

- Edge-Finder Detection

$$\forall \Omega \subset \mathcal{I}, \forall i \in \mathcal{I} \setminus \Omega : \text{ect}_{\Omega \cup \{i\}} > \text{lct}_\Omega \Rightarrow \Omega \lessdot i$$

- The function ect_Ω is NP-Hard to compute, so a relaxation is necessary
- The known Overload Check and Edge-Finder rules are based on the Fully-Elastic relaxation

Horizontally-Elastic Relaxation

- We introduce a stronger relaxation that restricts the elasticity of a task
- At any time t a task can consume between 0 and h_i units of resource
- Horizontally-Elastic computation of ect_Ω is given by

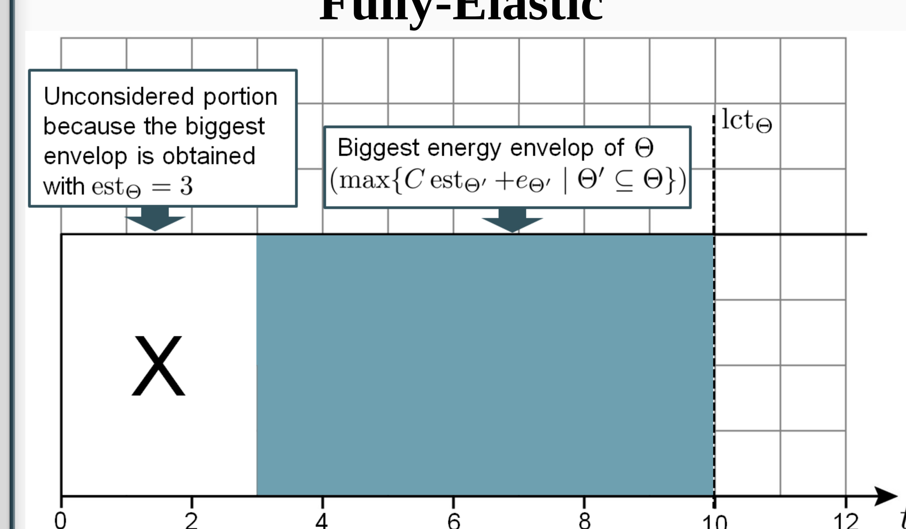
$$\begin{aligned}
h_{\text{max}}(t) &= \min\left(\sum_{i \in \Omega \mid \text{est}_i \le t < \text{lct}_i} h_i, C\right) \\
h_{\text{req}}(t) &= \sum_{i \in \Omega \mid \text{est}_i \le t < \text{ect}_i} h_i \\
h_{\text{cons}}(t) &= \min(h_{\text{req}}(t) + ov(t-1), h_{\text{max}}(t)) \\
ov(t) &= ov(t-1) + h_{\text{req}}(t) - h_{\text{cons}}(t) \, , \; ov(\min_{i \in \Omega} \text{est}_i) = 0 \\
\text{ect}^{\text{H}} &= \max\{t \mid h_{\text{cons}}(t) > 0\} + 1
\end{aligned}$$

Examples

- **Edge-Finder Detection**

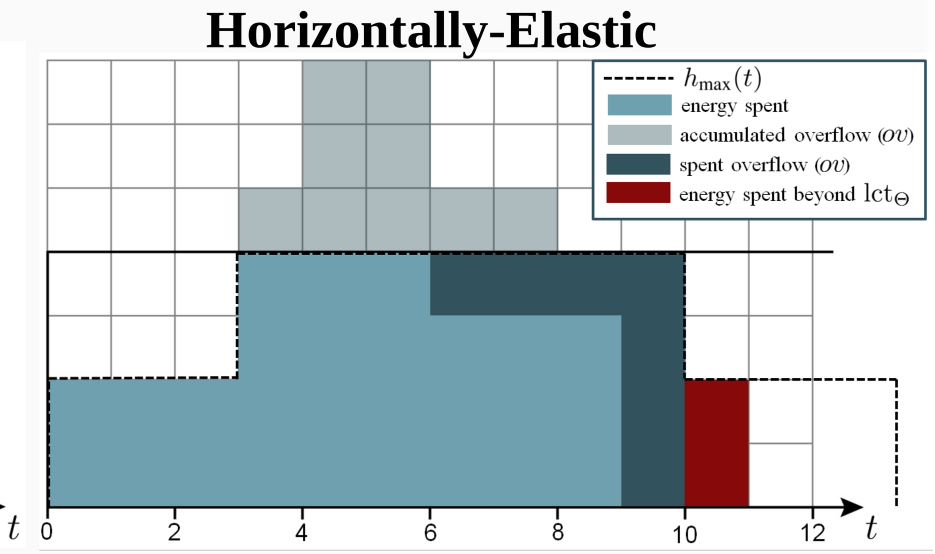
$$\{\langle \text{est}_i, \text{lct}_i, p_i, h_i \rangle\} = \{x : \langle 0,8,4,1 \rangle, y : \langle 0,8,4,1 \rangle, z : \langle 3,9,6,3 \rangle \\ w : \langle 4,8,2,1 \rangle, v : \langle 4,8,2,1 \rangle, u \langle 4,20,3,2 \rangle\}$$

Fully-Elastic



No precedence detected

Horizontally-Elastic



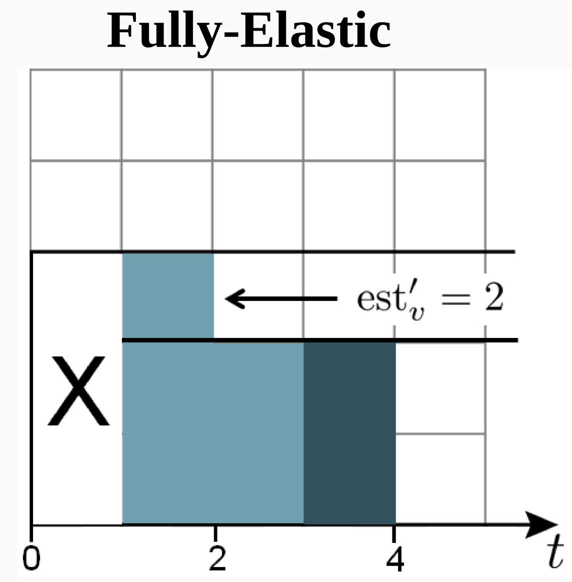
Precedence $\{x, y, z, v, w\} \lessdot u$ is detected

- **Edge-Finder Adjustment**

$$\{\langle \text{est}_i, \text{lct}_i, p_i, h_i \rangle\} = \{x : \langle 0,4,2,1 \rangle, y : \langle 1,4,1,3 \rangle, z : \langle 2,4,1,1 \rangle \\ w : \langle 2,4,1,3 \rangle, v : \langle 1,10,3,1 \rangle\}$$

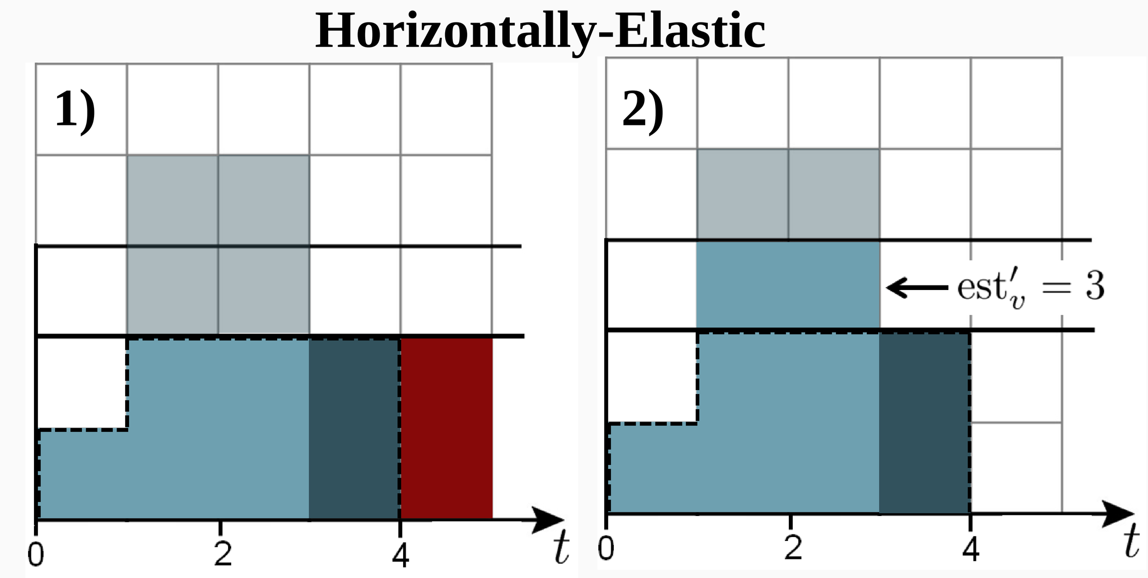
$$\{x, y, z, w\} \lessdot v$$

Fully-Elastic



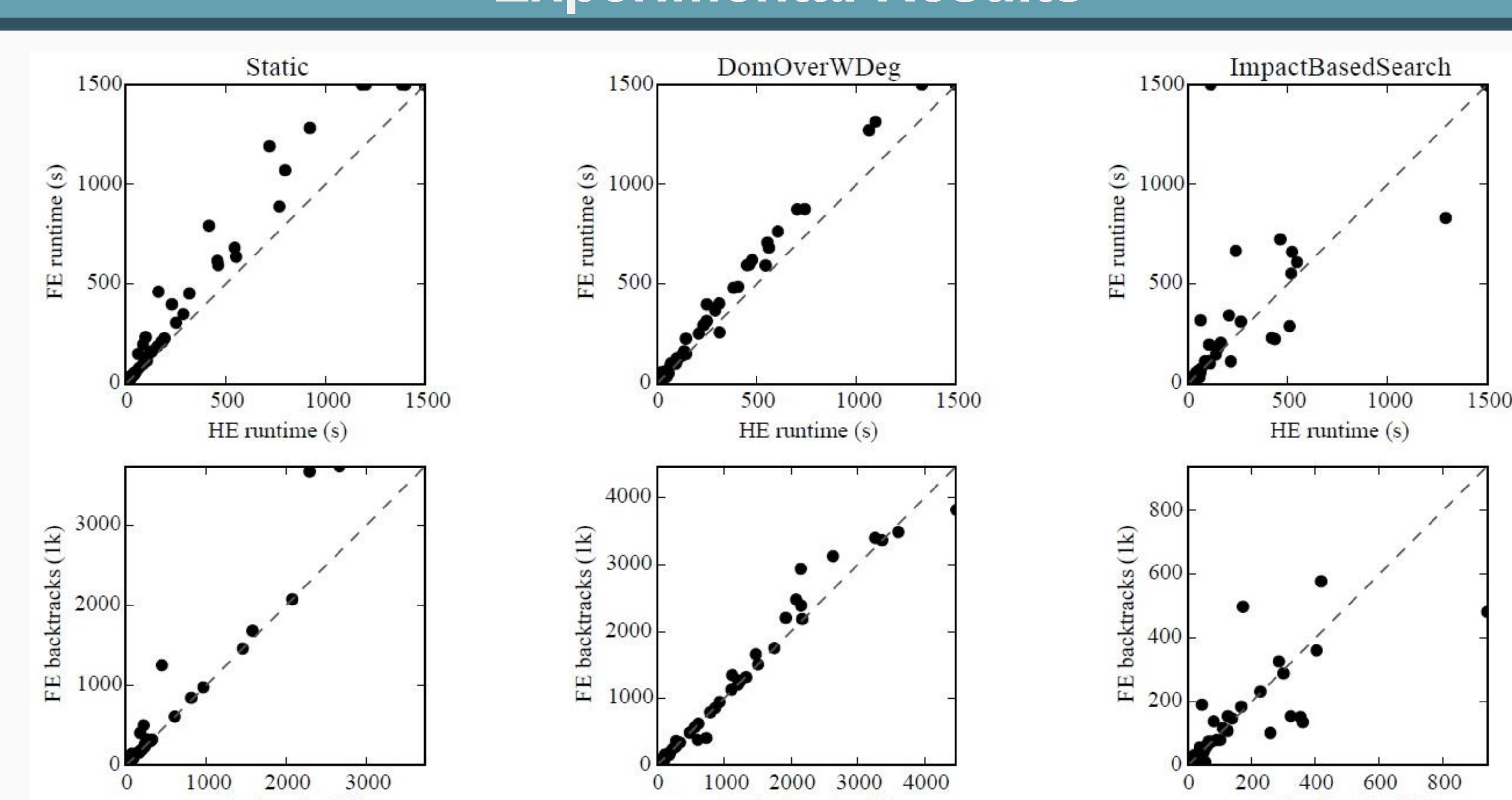
The maximum update is $\text{est}'_v = 2$

Horizontally-Elastic



A scheduling on the resource of capacity $C - h_v$ reveals that 2 units of energy must be spent on the upper part of the resource. A second scheduling then compute the new value of est'_v.

Experimental Results



Conclusion

1. We generalized the Overload Check and Edge-Finder rules (Cumulative)
2. We introduced a strong relaxation to compute ect_Ω
3. We presented a data structure to efficiently compute $\text{ect}^{\text{H}}_\Omega$
4. We presented algorithms enforcing the Overload Check and Edge-Finder rules using our relaxation in $\mathcal{O}(n^2)$ time and $\mathcal{O}(kn^2 + n^2)$ time respectively
5. Experimental results demonstrated the effectiveness of the method

UNIVERSITÉ LAVAL

Vincent Gingras, M.Sc. Student
vincent.gingras.5@ulaval.ca