The SoftCumulative Constraint with Quadratic Penalty

Yanick Ouellet Claude-Guy Quimper

Department of Computer Science and Software Engineering
Université Laval, Québec (QC), Canada

Abstract

The cumulative constraint greatly contributes to the success of constraint programming at solving scheduling problems. SoftCumulative, a version of the cumulative constraint where overloading the resource incurs a penalty is, however, less studied. We introduce a checker and a filtering algorithm for SoftCumulative, which are inspired by the energetic reasoning rule for the cumulative. Both algorithms can be used with a classic linear penalty function, but also with a quadratic penalty function, where the penalty of overloading the resource increases quadratically with the amount of the overload. We show that these algorithms are more general than existing algorithms and outperform a decomposition of SoftCumulative in practice.

Motivation

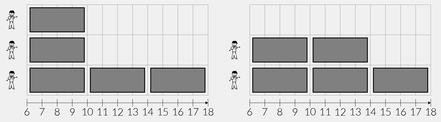
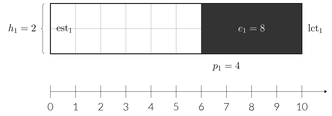


Figure 1. Motivation

Tasks

- 5 surgeries of 4 hours each to complete in the day
- Possible to call additional surgeons if required
- We want to call as few additional surgeons as possible

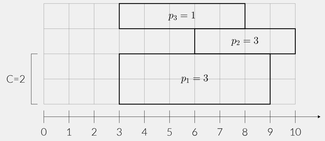
Task definition



Notation

- est: Earliest Starting Time
- lct: Lastest Completion Time
- p: Processing time
- h: Height
- $e = p \cdot h$: Energy

Cumulative constraint



Definition

- cumulative$([S_1, \ldots, S_n], [p_1, \ldots, p_n], [h_1, \ldots, h_n], C)$
- $S_i \in \{\text{est}_i, \ldots, \text{lct}_i - p_i\}$
- The sum of the height of the tasks in execution at a given time point t is at most C

SoftCumulative constraint

Definition

$$\texttt{SoftCumulative}(S, p, h, C, f(x), Z) \overset{\text{def}}{\iff} Z \geq \sum_{t \in T} f(\max(0, \sum_{\substack{i \in \mathcal{I}: \\ S_i \leq t < S_i + p_i}} h_i - C))$$

Additional parameters

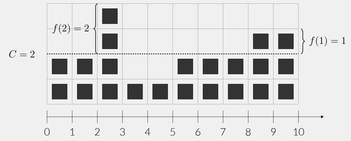
- $f(x)$: Cost function
- Z: Overcost variable

State of the art

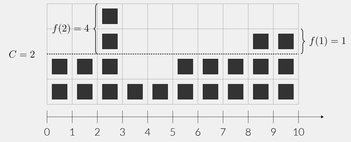
- Linear cost function $f(x) = x$
- Time-Tabling and Edge-Finding [De Clerc et al. 2010]

Exemple

Linear cost: $Z \geq 2 + 1 + 1 = 4$



Quadratic cost: $Z \geq 4 + 1 + 1 = 6$

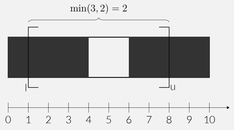


Objective function

Minimize the overcost Z

Minimum intersection

Minimum intersection

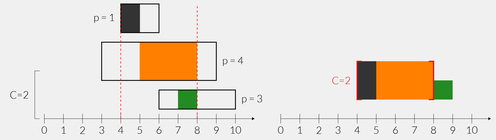


- $LS(1, 8) = 3$
- $RS(1, 8) = 2$
- $MI(1, 8) = \min(LS(1, 8), RS(1, 8)) = 2$

Energetic reasoning

Energetic reasoning

- Slack$(l, u) = C \cdot (u - l) - \sum_{i \in \mathcal{I}} MI(i, l, u)$
- If negative Slack, no solution for the cumulative
- Sufficient to check $O(n^2)$ intervals



Our contribution

Contributions

1. Use of a generic cost function (quadratic is of particular interest)
2. Adaptation of the energetic reasoning from the cumulative
   1. Checker algorithm
   2. Filtering algorithm
3. How to generate explanations to use with lazy clause generation
4. Empirical comparison against the decomposition

Adapting the energetic reasoning

Cumulative case

- Too much energy in a single interval means a failure

SoftCumulative case

- Possible to exceed the capacity
- But not by too much, nor in too many intervals
- We need a way to reason over multiple intervals

Reasoning over multiple intervals

Intuition

- Graph with time points as nodes
- Each edge represents an interval
- We find the longest path

Overcost

$Z \geq \texttt{cost}(0, 2) + \texttt{cost}(2, 3)$

Experiments

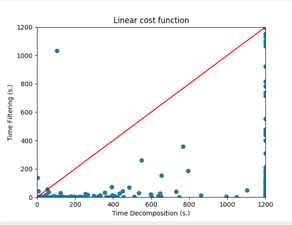
Benchmark

- Based on classical *RCPSP* instances
- Reduced resource capacities to force overload
- Comparison against a decomposition of the SoftCumulative

Decomposition

$$\texttt{SoftCumulative}(S, p, h, C, f(x), Z) \overset{\text{def}}{\iff} Z \geq \sum_{t \in T} f(\max(0, \sum_{\substack{i \in \mathcal{I}: \\ S_i \leq t < S_i + p_i}} h_i - C))$$

Experiments with linear cost function



Experiments with quadratic cost function