Une nouvelle caractérisation des intervalles d'intérêt pour le raisonnement énergétique

Alban Derrien

TASC (Mines Nantes, LINA, CNRS, INRIA),
4, Rue Alfred Kastler, FR-44307 Nantes Cedex 3, France.
alban.derrien@mines-nantes.fr

Résumé

Le raisonnement énergétique (ER) pour la contrainte Cumulative permet un filtrage des domaines supérieur à la plupart des algorithmes de l'état de l'art. Hélas, il est généralement trop coûteux pour être utilisé en pratique. Une des raisons de ce coût important est que trop d'intervalles de temps sont considérés. Dans la littérature, des approches heuristiques ont été développées dans le but de réduire le nombre d'intervalles à considérer, entrainant une perte de filtrage. Dans cet article, nous démontrons qu'il est possible de réduire le nombre d'intervalles d'intérêt jusqu'à un facteur sept pour la condition d'échec et le propagateur, sans diminuer la puissance de filtrage. En outre, nous montrons que, pour certaines classes de problèmes, associer ER à un algorithme de Time-Table constitue le meilleur compromis opérationnel.

1 Introduction

Les problèmes d'ordonnancement Cumulatifs (CuSP) sont présents dans de nombreux contextes industriels. Ils ont été très largement été étudiés en Programmation Par Contraintes (PPC). Le problème CuSP est défini par un ensemble d'activités $\mathcal{A}$ et une ressource de capacité C. C'est un problème NP-difficile. Chaque activité $a \in \mathcal{A}$ est définie par quatre variables : sa date de début de consommation s_a, sa durée p_a, sa date de fin e_a et sa consommation en ressource h_a. Nous utiliserons la notation $a = \{s_a, p_a, e_a, h_a\}$. Usuellement, les variables de durée et de consommation sont fixées. Une solution du CuSP est un ordonnancement de l'ensemble des taches vérifiant :

$$\forall a \in \mathcal{A} : s_a + p_a = e_a \quad \wedge \quad \forall i \in \mathbb{N} : \sum_{\substack{a \in \mathcal{A} \\ i \in [s_a, e_a[}} h_a \leq C$$

En PPC ce problème est généralement modélisé par la contrainte Cumulative [1].

Le raisonnement énergétique de Baptiste et al. (ER) est un des algorithmes de filtrage les plus puissants pour la contrainte Cumulative [2]. Cet algorithme utilise une caractérisation des intervalles dit *d'intérêt*, c'est-à-dire des intervalles suffisants pour vérifier que toutes les règles de filtrage sont satisfaites. Malheureusement, ER est souvent trop coûteux pour être utilisé en pratique. Tout d'abord, sa complexité en temps est $O(n^3)$. De plus, la constante cachée dans cette complexité en temps est élevée. En effet, de nombreux intervalles sont caractérisés comme étant d'intérêt alors que la plupart d'entre eux pourraient être ignorés. Dans la littérature, seules des approches heuristiques ont été proposées pour réduire ce nombre d'intervalles [3].

Cet article propose une caractérisation précise des intervalles d'intérêt pour la condition d'échec, que nous appelons *checker énergétique*, ainsi que pour le propagateur. Nous démontrons qu'il est possible de réduire ce nombre d'intervalles par un facteur sept pour le checker et pour le propagateur, sans perte de déduction. Nos expériences montrent une réduction non négligeable du temps d'exécution par rapport à l'existant. L'association d'un checker énergique et d'un algorithme de filtrage Time-Table (basé sur la base du profil cumulatif des parties obligatoires [7]) donne des résultats prometteurs. En outre, notre approche permet de répondre par l'affirmative à une question théorique ouverte [2] concernant les algorithmes de filtrage effectuant un raisonnement énergétique : les intervalles d'intérêt pour le checker énergétique sont suffisants pour réaliser un filtrage énergétique complet.

2 État de l'art

Pour une variable x, $\underline{x}$ représente la valeur minimum dans son domaine, et $\overline{x}$ la valeur maximale. Le raisonnement énergétique consiste à comparer l'énergie disponible

dans un intervalle (la durée de l'intervalle $\times$ la capacité C du problème) avec la quantité de ressource nécessairement consommée par les activités qui coupent cet intervalle. Pour une activité a, la longueur minimale occupée dans un intervalle $[t_1, t_2]$ est notée $MI(a, t_1, t_2)$:
$MI(a, t_1, t_2) = \max\left(0, \min\left(p_a, t_2 - t_1, \underline{e_a} - t_1, t_2 - \overline{s_a}\right)\right)$.

Proposition 1 (Checker énergétique [4]) *Si la condition*

$$\forall t_1, t_2 \in \mathbb{N}^2, t_1 < t_2$$

$$C \times (t_2 - t_1) \geq \sum_{a \in \mathcal{A}} h_a \times MI(a, t_1, t_2) \quad (1)$$

n'est pas respectée, alors la contrainte Cumulative n'admet pas de solution.

La question est alors de trouver le plus petit ensemble d'intervalles $[t_1, t_2],\ t_2 > t_1$ devant être vérifiés pour détecter l'infaisabilité. À partir de cette condition une règle d'ajustement des bornes des domaines peut être définie : pour une activité a la borne $\underline{s_a}$ peut être mise à jour si l'activité l'ordonnancement de a à $\underline{s_a}$ entraîne nécessairement une surcharge (et de même concernant $\overline{s_a}$). L'étude du propagateur sera effectuée dans la section 4.

Caractérisation de Baptiste et al. Pour s'assurer que la condition (1) est respectée, Baptiste et al. [2] ont montré qu'il est suffisant de vérifier les intervalles de la forme suivante :

- $[t_1, t_2], t_1 \in O_1 < t_2 \in O_2$
- $[t_1, t_2], t_1 \in O_1 < t_2 \in O(t_1)$
- $[t_1, t_2], t_2 \in O_2 > t_1 \in O(t_2)$

avec $O_1 = \{\underline{s_a}, \forall a \in \mathcal{A}\} \cup \{\overline{s_a}, \forall a \in \mathcal{A}\} \cup \{\underline{e_a}, \forall a \in \mathcal{A}\}$, $O_2 = \{\overline{e_a}, \forall a \in \mathcal{A}\} \cup \{\overline{s_a}, \forall a \in \mathcal{A}\} \cup \{\underline{e_a}, \forall a \in \mathcal{A}\}$, $O(t) = \{\underline{s_a} + \overline{e_a} - t, \forall a \in \mathcal{A}\}$. Notons O_B l'ensemble de ces intervalles.

Cette caractérisation permet de réduire le nombre d'intervalles d'intérêt à $9+3+3=15$ pour chaque paire d'activité. Baptiste et al. ont démontré ([2], proposition 19) que cette caractérisation est suffisante pour détecter une surcharge en analysant la fonction représentant l'évolution de l'énergie disponible : $Slack(\mathcal{A}, t_1, t_2) = C \times (t_2 - t_1) - \sum_{a \in \mathcal{A}} h_a \times MI(a, t_1, t_2)$.

Sachant que $MI(a, t_1, t_2)$ peut être calculé en temps constant et compte tenu des définitions de O_1, O_2 et $O(t)$, nous obtenons un checker naïf en $O(n^3)$, en calculant $MI(a, t_1, t_2)$ pour tout t_1, t_2 et a. Cependant, la fonction de $Slack$ est linéaire et continue par morceaux. Un extremum local ne peut être trouvé que dans les points critiques de la fonction $Slack(\mathcal{A}, t_1, t_2)$. Ces points critiques ne peuvent exister que pour des valeurs de t_1 et t_2 correspondant aux intervalles caractérisés. Ces propriétés conduisent à un checker en $O(n^2)$ [2]. Deux questions subsistent, présentées comme ouvertes dans [2].

1. L'ensemble d'intervalles d'intérêt a été démontré comme étant suffisant, mais est il minimal en taille ?
2. Cet ensemble reste t-il suffisant pour appliquer un filtrage aux bornes ?

Caractérisation de Schwindt. Schwindt a proposé une caractérisation plus fine se basant sur une étude plus précise des extrémums de la fonction $f_1 : (t_1, t_2) \rightarrow Slack(\mathcal{A}, t_1, t_2)$ (théorèmes 3.7 et 3.8 dans [8], écrite en allemand). Cette fonction à deux variables est localement minimale seulement si sa dérivée à gauche est plus petite que sa dérivée à droite, à la fois pour t_1 et pour t_2. Comme $Slack(\mathcal{A}, t_1, t_2)$ est une somme d'intersections minimales, il doit exister une activité i (resp. j) telle que son intersection minimale a une dérivée à gauche plus grande que sa dérivée à droite sur t_1 (resp. t_2). Ces observations conduisent au théorème 1.

Théorème 1 *La fonction $Slack(\mathcal{A}, t_1, t_2)$ est localement minimum seulement si il existe deux activités i, j telles que les deux propositions suivantes sont satisfaites.*

$$\frac{\partial^- MI(i, t_1, t_2)}{\partial t_1} > \frac{\partial^+ MI(i, t_1, t_2)}{\partial t_1} \quad (2)$$

$$\frac{\partial^- MI(j, t_1, t_2)}{\partial t_2} > \frac{\partial^+ MI(j, t_1, t_2)}{\partial t_2} \quad (3)$$

Les valeurs de t_1 pour lesquelles la condition (2) est respectée sont appelées dates d'intérêt pour le début d'intervalle. De même, les valeurs de t_2 pour lesquelles la condition (3) est respectée sont appelées dates d'intérêt pour la fin d'intervalle. Leur conjonction donne alors les conditions nécessaires pour qu'un intervalle $[t_1, t_2[$ puisse être considéré comme d'intérêt pour le checker énergétique.

Schiwndt en a proposé une caractérisation en analysant les variations de la fonction d'intersection minimale, donnant 8 intervalles possibles, tous inclus dans la caractérisation de Baptiste et al. Cette caractérisation répond à la première question laissé ouverte : Le nombre d'intervalles dans la caractérisation de Baptiste et al. peut être réduit.

Nous proposons dans la section suivante une nouvelle analyse de la fonction d'intersection minimale, menant a une caractérisation plus précise des intervalles d'intérêt.

3 Caractérisation pour le checker

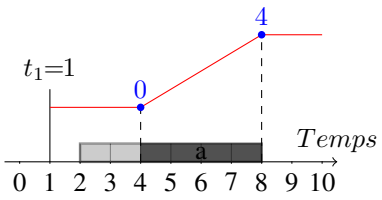
Dans cette section nous allons étudier l'inégalité (3) pour déterminer sous quelles conditions une date de fin peut être considérée comme étant d'intérêt. Le problème cumulatif étant symétrique, nous en déduirons de (2) les conditions pour l'intérêt d'une date de début.

Nous souhaitons caractériser, pour une date de début t_1 et une activité a les points d'intérêt issus de la fonction $f_2 : t_2 \rightarrow MI(a, t_1, t_2)$: Les valeurs de t_2 pour lesquelles la dérivée à gauche est plus grande que la dérivée à droite.

Propriété 1 *Pour une date de début d'intervalle t_1 et une activité a, la fonction $f_2 : t_2 \to MI(a, t_1, t_2)$ a au plus 1 point d'intérêt et ce point appartient à $O_2 \cup O(t_1)$.*

Preuve Nous démontrons la propriété 1 par une étude de cas : nous démontrons que pour les 4 positions de t_1 par rapport à a il existe au plus un point d'intérêt. Pour cela nous prouvons que la fonction f_2 est linéaire et continue par morceaux, composée d'au plus 3 morceaux. Les deux points critiques correspondent au début et à la fin de consommation. Nous démontrons ensuite que la fin de consommation est l'unique point avec une dérivée plus grande à gauche qu'à droite. Nous prenons un exemple graphique avec une même activité pour les quatre cas : $a = \{s_a \in [2, 4],\ p_a = 4,\ e_a \in [6, 8], h_a\}$.

1. $t_1 \leq \underline{s_a}$:



Nous rappelons la définition de $MI(a, t_1, t_2) = \max\left(0, \min\left(p_a,\ t_2 - t_1,\ \underline{e_a} - t_1,\ t_2 - \overline{s_a}\right)\right)$. Ce qui nous amène à étudier trois cas :

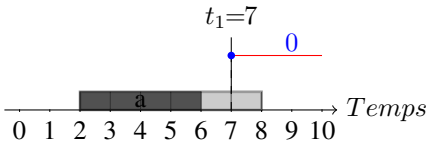
(a) si $t_2 \leq \overline{s_a}$ alors $MI(a, t_1, t_2) = 0$.

(b) si $\overline{s_a} \leq t_2 \leq \overline{e_a}$ alors $MI(a, t_1, t_2) = t_2 - \overline{s_a}$.

(c) si $\overline{e_a} \leq t_2$ alors $MI(a, t_1, t_2) = p_a$.

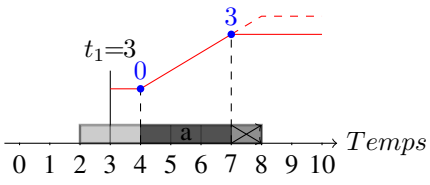
$p_a - (\overline{e_a} - t_2)$ est égale à 0 lorsque $t_2 = \overline{s_a}$ et p_a lorsque $t_2 = \overline{e_a}$. Donc lorsque $t_1 \leq \underline{s_a}$ la fonction de consommation est continue et linaire par morceaux, composée de trois morceaux, avec un unique point d'intérêt : $\overline{e_a}$.

2. $t_1 \geq \underline{e_a}$:



Dans ce cas $MI(a, t_1, t_2) = 0$ pour tout intervalle. La fonction est trivialement continue et linéaire par partie sans point d'intérêt.

3. $t_1 > \underline{s_a}$ et $t_1 < \underline{e_a}$ et $t_1 < \overline{s_a}$:



soit $\Delta = t_1 - \underline{s_a}$. Etudions trois cas pour les valeurs de t_2 :

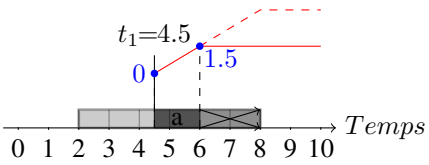
(a) si $t_2 \leq \overline{s_a}$ alors $MI(a, t_1, t_2) = 0$.

(b) si $\overline{s_a} \leq t_2 \leq \overline{e_a} - \Delta$ alors $MI(a, t_1, t_2) = t_2 - \overline{s_a}$.

(c) si $t_2 \geq \overline{e_a} - \Delta$ alors $MI(a, t_1, t_2) = p_a - \Delta$.

Dans chacun des trois cas, la fonction est linéaire. Puisque $t_2 - \overline{s_a}$ est égale à 0 lorsque $t_2 = \overline{s_a}$ et que $t_2 = \overline{s_a}$ et $p_a - \Delta$ lorsque $t_2 = \overline{e_a} - \Delta$, la fonction de consommation est continue et linéaire par morceaux, composée de 3 morceaux, avec un unique point d'intérêt : $\underline{s_a} + \overline{e_a} - t_1$.

4. $t_1 > \underline{s_a}$ et $t_1 < \underline{e_a}$ et $t_1 \geq \overline{s_a}$:



Nous avons ici deux cas distincts :

(a) si $t_2 \leq \underline{e_a}$ then $MI(a, t_1, t_2) = t_2 - t_1$

(b) si $MI(a, t_1, t_2) = p_a - \Delta$

Sachant que $t_2 - t_1 = p_a - \Delta$ lorsque $t_2 = \underline{e_a}$, la fonction de consommation est continue et linaire par morceaux, composée de deux parties, avec un unique point d'intérêt : $\underline{e_a}$.

Nous avons montré que quelque soit la date de début d'intervalle t_1 et quelque soit l'activité a, la fonction $f_2 : t_2 \to MI(a, t_1, t_2)$ est continue et linéaire par morceaux, avec au plus un point d'intérêt. Selon les cas les différents points d'intérêt sont : $\underline{e_a}$, $\underline{s_a} + \overline{e_a} - t_1$ et $\overline{e_a}$ appartenant tous les trois à $O_2 \cup O(t_1)$. Ce qui démontre la propriété. □

Nous avons caractérisé, pour une activité a, les valeurs possibles de fin d'intervalle pour lesquelles la condition (3) est vérifiée (Table 1). Par symétrie, nous pouvons déduire les valeurs de début d'intervalle pour lesquelles la condition (2) est vérifiée (Table 2).

conditions	t_2	
$t_1 \leq \underline{s_i}$	$\overline{e_i}$	cas 1
$t_1 > \underline{s_i} \wedge t_1 < \underline{e_i} \wedge t_1 < \overline{s_i}$	$\underline{s_i} + \overline{e_i} - t_1$	cas 3
$t_1 > \underline{s_i} \wedge t_1 < \underline{e_i} \wedge t_1 \geq \overline{s_i}$	$\underline{e_i}$	cas 4

TABLE 1 – Conditions d'intérêt de fin d'intervalle.

Le cas 3 et son symétrique, bien que satisfaisant les conditions nécessaires individuellement, ne peuvent amener ensemble à un intervalle d'intérêt. En effet, localement, la fonction $\delta \to MI(i, t_1 + \delta, t_2 - \delta)$ est égale à

conditions	intervalle	
$\underline{s_i} \leq \underline{s_j} \ \wedge \ \overline{e_j} \geq \overline{e_i}$	$[\underline{s_i},\ \overline{e_j}]$	A
$\underline{s_i} > \underline{s_j} \ \wedge \ \underline{s_i} < \underline{e_j} \ \wedge \ \underline{s_i} < \overline{s_j} \ \wedge \ \underline{s_j} + \overline{e_j} - \underline{s_i} \geq \overline{e_i}$	$[\underline{s_i},\ \underline{s_j} + \overline{e_j} - \underline{s_i}]$	B
$\underline{s_i} > \underline{s_j} \ \wedge \ \underline{s_i} < \underline{e_j} \ \wedge \ \underline{s_i} \geq \overline{s_j} \ \wedge \ \underline{e_j} \geq \overline{e_i}$	$[\underline{s_i},\ \underline{e_j}]$	C
$\overline{s_i} \leq \underline{s_j} \ \wedge \ \overline{e_j} < \overline{e_i} \ \wedge \ \overline{e_j} > \overline{s_i} \ \wedge \ \overline{e_j} \leq \underline{e_j}$	$[\overline{s_i}, \overline{e_j}]$	D
$\overline{s_i} > \underline{s_j} \ \wedge \ \overline{s_i} < \underline{e_j} \ \wedge \ \overline{s_i} < \overline{s_j} \ \wedge \ \overline{s_i} < \underline{s_j} + \overline{e_j} - \overline{s_i} \leq \underline{e_i} \ \wedge \ \underline{s_j} + \overline{e_j} - \overline{s_i} < \overline{e_i}$	$[\overline{s_i},\ \underline{s_j} + \overline{e_j} - \overline{s_i}]$	E
$\overline{s_i} > \underline{s_j} \ \wedge \ \overline{s_i} < \underline{e_j} \ \wedge \ \overline{s_i} \geq \overline{s_j} \ \wedge \ \underline{e_j} < \overline{e_i} \ \wedge \ \underline{e_j} > \overline{s_i} \ \wedge \ \underline{e_j} \leq \underline{e_i}$	$[\overline{s_i},\ \underline{e_j}]$	F
$\overline{e_j} < \overline{e_i} \ \wedge \ \overline{e_j} > \overline{s_i} \ \wedge \ \overline{e_j} > \underline{e_i} \ \wedge \ \underline{s_i} + \overline{e_i} - \overline{e_j} \leq \overline{s_j}$	$[\underline{s_i} + \overline{e_i} - \overline{e_j},\ \overline{e_j}]$	G
$\underline{e_j} < \overline{e_i} \ \wedge \ \underline{e_j} > \overline{s_i} \ \wedge \ \underline{e_j} > \underline{e_i} \ \wedge \ \overline{s_j} \leq \underline{s_i} + \overline{e_i} - \underline{e_j} < \underline{e_j} \ \wedge \ \underline{s_j} < \underline{s_i} + \overline{e_i} - \underline{e_j}$	$[\underline{s_i} + \overline{e_i} - \underline{e_j},\ \underline{e_j}]$	H

TABLE 3 – Intervalles d'intérêt pour une paire d'activités (i,j)

conditions	t_1	
$t_2 \geq \overline{e_i}$	$\underline{s_i}$	Sym. 1
$t_2 < \overline{e_i} \ \wedge \ t_2 > \overline{s_i} \ \wedge \ t_2 > \underline{e_i}$	$\underline{s_i} + \overline{e_i} - t_1$	Sym. 3
$t_2 < \overline{e_i} \ \wedge \ t_2 > \overline{s_i} \ \wedge \ t_2 \leq \underline{e_i}$	$\overline{s_i}$	Sym. 4

TABLE 2 – Conditions d'intérêt de début d'intervalle.

$MI(i, t_1, t_2) + \delta$. Nous ne sommes donc pas en présence d'un extremum pour l'intersection minimale. L'intervalle n'est donc pas d'intérêt pour trouver un minimum de la fonction de $Slack$.

Nous pouvons donc déduire 8 cas (de A à H) de la conjonction des conditions pour t_1 et t_2. La Table 3 résume les intervalles d'intérêt pour toute paire d'activités (i, j).

Notre caractérisation est plus précise que celle proposée par Schwindt : par exemple le cas *B* dans la Table 3 est plus précis que le cas équivalent *iii* dans la table 3.5 page 84 de [8]. Notons que pour tout couple d'activités seulement deux des cas sont possibles, les conditions étant mutuellement incompatibles. Nous avons donc réduit la caractérisation de 15 à 2 intervalles d'intérêt par couple d'activités.

De plus, nous pouvons remarquer que tout intervalle d'intérêt commence seulement aux dates de début au plus tôt et au plus tard $\underline{s_a}, \overline{s_a}$, ou alors finit à des valeurs de fins (plus tard ou plus tôt). Cette remarque amène à un allégement de l'algorithme en $O(n^2)$ pour le checker proposé par Baptiste et al. [2], en réduisant l'ensemble des dates de début d'intervalle O_1 à un nouvel ensemble : $O_S = \{\underline{s_a},\ \forall a \in \mathcal{A}\} \cup \{\overline{s_a}, \forall a \in \mathcal{A}\}$.

3.1 Nouvel algorithme pour le checker

Dans cette section nous décrivons un nouvel algorithme pour le checker énergétique, basé sur les principes de celui de Baptiste et al. [2], en limitant les dates de début d'intervalle à l'ensemble O_S tel que défini dans la section précédente ($O_S = \{\underline{s_a},\ \forall a \in \mathcal{A}\} \cup \{\overline{s_a}, \forall a \in \mathcal{A}\}$).

En triant les intervalles par date de fin croissante on peut calculer le changement de consommation entre deux intervalles, incrémentalement. Chaque début et fin de consommation sont représentés par un événement. Ces événements, triés par ordre de date croissante, appartiennent d'après la caractérisation précédente à quatre ensembles :

- $\mathcal{S}_M = (\overline{s_a}, a), \forall a \in \mathcal{A}$;
- $\mathcal{E}_m = (\underline{e_a}, a), \forall a \in \mathcal{A}$;
- $\mathcal{E}_M = (\overline{e_a}, a), \forall a \in \mathcal{A}$;
- $\mathcal{L}' = (\underline{s_a} + \overline{e_a} - t_1, a), \forall a \in \mathcal{A}$.

Nous avons vu dans la section précédente que le début de consommation d'énergie d'une activité (SoC_a) est soit au début de l'intervalle si le début d'intervalle intersecte la partie obligatoire de l'activité (cas 4), soit en $\overline{s_a}$ (cas 1&3). De plus, la fin de consommation d'énergie (EoC_a) est soit $\overline{e_a}$ si $t_1 \leq \underline{s_a}$ (cas 1), soit $\underline{e_a}$ si t_1 intersecte la partie obligatoire (cas 4), ou bien sinon $\underline{s_a} + \overline{e_a} - t_1$ si $t_1 > \underline{s_a}$ *et* $t_1 < \underline{e_a}$ *et* $t_1 < \overline{s_a}$ (cas 3).

```
1  foreach t_1 ∈ O_S do
2      pente = C − Σ_{a∈A} MI(a, t_1, t_1 + 1);
3      charge = 0 ;  t_2^old = t_1;
4      L' = {(t' − t_1, a) | (t', a) ∈ L};
5      foreach event(t_2, a) > t_1 in S_M, E_m, E_M, L' do
6          charge += pente × (t_2 − t_2^old) ;
7          if charge < 0 then Fail;
8          if event est un SoC_a then
9              pente −= h_a;
10         else if event est un EoC_a then
11             pente += h_a;
12         end
13         t_2^old = t_2;
14     end
15 end
```

Algorithm 1: ERC, un nouveau checker énergétique.

4 Caractérisation pour le propagateur

4.1 Suffisance des intervalles d'intérêt

Le raisonnement énergétique permet de filtrer les bornes des variables. Si l'ordonnancement d'une activité à son placement au plus tôt (placement le plus à gauche possible) induit une surcharge alors ce placement n'est pas valide et

la valeur $\underline{s_a}$ peut être filtrée. La consommation par décalage gauche de l'activité a qui coupe l'intervalle $[t_1, t_2[$ est définie par :

$$LS(a, t_1, t_2) = \max(0, \min(\underline{e_a}, t_2) - \max(\underline{s_a}, t_1))$$

Réciproquement la consommation par décalage à droite est :

$$RS(a, t_1, t_2) = \max(0, \min(\overline{e_a}, t_2) - \max(\overline{s_a}, t_1))$$

Par souci de lisibilité notons $Dispo(a, t_1, t_2)$ l'énergie disponible pour une activité a dans l'intervalle $[t_1, t_2[$:

$$Dispo(a, t_1, t_2) = Slack(\mathcal{A} \backslash a, t_1, t_2)$$

La règle de filtrage définie par Baptiste et al. s'écrit alors :

Proposition 2 *Pour une activité a si il existe un intervalle $[t_1, t_2[$ tel que :*

$$Dispo(a, t_1, t_2) \geq h_a \times LS(a, t_1, t_2) \quad (4)$$

n'est pas vérifiée alors le placement à gauche de l'activité a n'est pas valide ; et l'activité ne peut commencer avant :

$$t_2 - \frac{1}{h_a} \times Dispo(a, t_1, t_2) \quad (5)$$

Cette règle vérifie si l'activité a placée à son placement au plus tôt créée une surcharge. Si c'est le cas alors un filtrage de $\underline{s_a}$ peut être effectué en calculant la quantité d'énergie disponible dans l'intervalle pour l'activité a et en fixant sa date de début au plus tôt à ce moment là. Une règle symétrique utilisant le placement à droite $RS(a, t_1, t_2)$ permet de calculer une date après laquelle l'activité ne peut finir et d'effectuer un filtrage sur $\overline{e_a}$.

Proposition 3 *Pour une activité a si il existe un intervalle $[t_1, t_2[$ tel que :*

$$Dispo(a, t_1, t_2) \geq h_a \times RS(a, t_1, t_2) \quad (6)$$

n'est pas vérifiée alors, le placement à droite de l'activité a n'est pas valide ; et l'activité ne peut finir après :

$$t_1 + \frac{1}{h_a} \times Dispo(a, t_1, t_2) \quad (7)$$

Les règles de filtrage (4) et (6) s'appliquent à tout intervalle $[t_1, t_2[$. Il convient alors de trouver le plus petit ensemble d'intervalles devant être vérifiés pour détecter si un filtrage peut être effectué. Baptiste et al. proposent d'utiliser la caractérisation du checker pour le propagateur. Ils laissent malgré tout ouverte la question de complétude de cette caractérisation. Nous allons démontrer cette complétude puis nous en affinerons la caractérisation.

Théorème 2 $\forall [t_1, t_2[$, *(4) et (6) sont vérifiés.*

$$\Longleftrightarrow$$

$$\begin{cases} \forall [t_1, t_2[, t_1 \in O_1, t_2 \in O_2, & \textit{(4) et (6) sont vérifiés.} \\ \forall [t_1, t_2[, t_1 \in O_1, t_2 \in O(t_1), & \textit{(4) et (6) sont vérifiés.} \\ \forall [t_1, t_2[, t_2 \in O_2, t_1 \in O(t_2), & \textit{(4) et (6) sont vérifiés.} \end{cases}$$

Pour démontrer ce théorème commençons par démontrer l'équivalence pour la condition (4). De manière analogue à la démonstration précédente étudions les variations de la fonction associée :

$$f_3 : t_2 \rightarrow Dispo(a, t_1, t_2) - h_a \times LS(a, t_1, t_2) \quad (8)$$

Si toutes les valeurs de la fonction sont positives alors aucun filtrage n'est nécessaire d'après la règle de filtrage d'ER. Il suffit alors de chercher la présence de valeur négative. Démontrer le théorème revient alors à démontrer que tous les minimums de f_3 correspondent à des intervalles de O_B. Cette fonction étant la somme de fonctions linéaires et continues par morceaux les minimums locaux ne peuvent exister que si une partie de la somme est minimale.

L'étude des minimums induit par $Dispo(a, t_1, t_2)$ a déjà été effectuée (propriété 1) ; pour rappel $Dispo(a, t_1, t_2) = Slack(\mathcal{A} \setminus a, t_1, t_2)$. Il nous reste donc à caractériser les minimums de $-h_a \times LS(a, t_1, t_2)$. Cela revient à chercher les points d'intérêt de la fonction $f_4 : t_2 \rightarrow LS(a, t_1, t_2)$; les points pour lesquels la dérivée à gauche est plus petite que la dérivée à droite (théorème 1).

Propriété 2 *Pour une date de début d'intervalle t_1 et une activité a la fonction $f_4 : t_2 \rightarrow LS(a, t_1, t_2)$ a au plus 1 point d'intérêt : $\underline{e_a}$.*

Preuve En suivant le modèle de la démonstration de la propriété 1 nous étudions différents cas.

1. $t_1 \leq \underline{s_a}$:
   Alors $LS(a, t_1, t_2) = \max(0, \min(\underline{e_a}, t_2) - \underline{s_a})$ Etudions alors 3 sous cas :
   (a) si $t_2 \leq \underline{s_a}$ alors $LS(a, t_1, t_2) = 0$.
   (b) si $\underline{s_a} \leq t_2 \leq \underline{e_a}$ alors $LS(a, t_1, t_2) = t_2 - \underline{s_a}$.
   (c) si $\underline{e_a} \leq t_2$ alors $LS(a, t_1, t_2) = p_a$.
   Ce qui montre que lorsque $t_1 \leq \underline{s_a}$ alors la fonction de consommation est continue et linéaire par morceaux, composée de trois morceaux, avec seulement un point d'intérêt : $\underline{e_a}$.
2. $\underline{s_a} \leq t_1$:
   Alors $LS(a, t_1, t_2) = \max(0, \min(\underline{e_a}, t_2) - t_1)$ Etudions alors 2 sous cas :
   (a) si $t_2 \leq \underline{e_a}$ alors $LS(a, t_1, t_2) = t_2 - t_1$.
   (b) si $\underline{e_a} \leq t_2$ alors $LS(a, t_1, t_2) = \underline{e_a} - t_1$.
   Ce qui montre que lorsque $t_1 \geq \underline{s_a}$ alors la fonction de consommation est continue et linéaire par morceaux, composée de trois morceaux, avec seulement un point d'intérêt : $\underline{e_a}$.

Nous avons montré que quelque soit la date de début d'intervalle t_1 et quelque soit l'activité a la fonction f_4 ($t_2 \rightarrow LS(a, t_1, t_2)$) est continue et linéaire par morceaux, avec au plus un unique point d'intérêt : $\underline{e_a}$. Ce qui démontre la propriété. □

preuve du théorème 2 L'implication de la droite vers la gauche est évidente.

D'après les propriétés 1 et 2 les points d'intérêt de la fonction f_3 appartiennent à $O_2 \cup O(t_1)$. Par symétrie les points d'intérêt pour le début d'intervalle appartiennent à $O_1 \cup O(t_2)$. Les minimums de f_3 sont alors dans O_B. Ce qui démontre l'implication de la droite vers la gauche pour la condition (4).

La démonstration de l'implication pour la condition (6) est equivalente. □

Le théorème 2 répond directement à la seconde question ouverte :

Propriété 3 *La caractérisation proposée par Baptiste et al. est suffisante pour effectuer un filtrage énergétique complet.*

4.2 Caractérisation des intervalles d'intérêt

La table 3 donne la caractérisation des intervalles d'intérêt induits par $Slack(\mathcal{A}, t_1, t_2)$. Ces intervalles sont d'intérêt pour toutes les activités puisque la fonction $Slack$ est commune aux conditions de filtrage de chaque activité. Chaque activité a en outre des intervalles d'intérêt qui lui sont propres : ceux induit par $LS(a, t_1, t_2)$ et $RS(a, t_1, t_2)$. Dans le but d'obtenir une caractérisation fine des intervalles d'intérêt pour le propagateur nous nous intéressons maintenant à caractériser les intervalles définis par $LS(a, t_1, t_2)$ et $RS(a, t_1, t_2)$ qui n'ont pas déjà été caractérisés par $Slack(\mathcal{A}, t_1, t_2)$.

Par exemple la fonction de consommation à gauche $LS(a, t_1, t_2)$ nous donne comme date de fin d'intervalle d'intérêt $\underline{e_a}$. Or lorsque $t_1 \geq \overline{s_a}$ la fin d'intervalle $\underline{e_a}$ est déjà induite (cas 4). Les conditions d'intérêt de $\underline{e_a}$ pour le placement à gauche peuvent alors être renforcées : $t_1 < \underline{e_a} \wedge t_1 < \overline{s_a}$. Symétriquement les conditions d'intérêt pour $\underline{s_a}$ comme date de début sont : $t_2 > \underline{s_a} \wedge t_2 < \overline{e_a}$.

La fonction de consommation à droite nous donne comme date de fin d'intervalle d'intérêt $\overline{e_a}$. Or lorsque $t_1 \leq \underline{s_a}$ la fin d'intervalle $\overline{e_a}$ est déjà induite (cas 1). Les conditions d'intérêt pour $\overline{e_a}$ peuvent alors être renforcées : $t_1 < \overline{e_a} \wedge t_1 > \underline{s_a}$. Symétriquement les conditions d'intérêt pour $\overline{s_a}$ comme date de début sont : $t_2 > \overline{s_a} \wedge t_2 > \underline{e_a}$.

conditions	t_2	
$t_1 < \underline{e_i} \ \wedge \ t_1 < \overline{s_i}$	$\underline{e_i}$	LS
$t_1 < \overline{e_i} \ \wedge \ t_1 > \underline{s_i}$	$\overline{e_i}$	RS

TABLE 4 – conditions d'intérêt de fin d'intervalles

conditions	t_1	
$t_2 > \underline{s_i} \ \wedge \ t_2 < \overline{e_i}$	$\underline{s_i}$	Sym.LS
$t_2 > \overline{s_i} \ \wedge \ t_2 > \underline{e_i}$	$\overline{s_i}$	Sym.RS

TABLE 5 – conditions d'intérêt de début d'intervalles

De manière analogue à la construction des intervalles d'intérêt pour le checker nous pouvons établir que les intervalles d'intérêt pour le placement à gauche se construisent par la conjonction des conditions d'intérêt pour la date de début et la date de fin.
Pour une activité a les intervalles d'intérêt sont alors construit de trois manières :

1. Le début est d'intérêt pour $LS(a, t_1, t_2)$ et la fin est d'intérêt pour $Slack(i, t_1, t_2)$ pour une activité i.
2. Le début est d'intérêt pour $Slack(i, t_1, t_2)$ pour une activité i et la fin est d'intérêt pour $LS(a, t_1, t_2)$.
3. Le début et la fin sont d'intérêt pour $LS(a, t_1, t_2)$.

Puisqu'il existe seulement une date de fin (resp. début) d'intérêt par activité le cas 1 (resp. 2) induit $n-1$ intervalles d'intérêt pour l'activité a. Le cas 3 nous indique que l'intervalle $[\underline{s_a}, \underline{e_a}[$ est toujours d'intérêt pour le placement à gauche de l'activité a. Ces intervalles sont précisément caractérisés dans la table 6. Nous avons donc défini $2{\times}n{-}1$ intervalles d'intérêt pour le placement à gauche de l'activité a. De même pour le placement à droite (Table 7). Avec les $2{\times}n^2$ intervalles d'intérêt commun à chaque activité notre caractérisation permet de réduire le nombre d'intervalle d'intérêt pour chaque activité de $15n^2$ à $2n^2+4n-2$. Soit un facteur jusqu'à 7 lorsque le nombre d'activité est supérieur à 30 .

4.3 Nouvel algorithme pour le propagateur

Baptiste et al. proposent un algorithme calculant pour tous les intervalles d'intérêt l'énergie totale consommée puis vérifiant pour chaque activité si le placement à gauche (ou à droite) ne rajoute pas trop d'énergie, filtrant le cas échéant. Nous proposons dans notre nouvel algorithme de reprendre cette phase (ligne 1 à 16), en se limitant aux intervalles caractérisés dans la Table 3 (ce qui réduit le nombre d'intervalles calculés de $15n^2$ à $2n^2$).

```
1  foreach Intervalle d'intérêt [t1, t2[ (cf Table 3) do
2      W := Σ_{a∈A} h_a×MI(a, t1, t2);
3      if W > C × (t2 − t1) then
4          fail;
5      else
6          foreach activité a ∈ A do
7              Disp := C×(t2−t1) − W + h_a×MI(a, t1, t2);
8              if Disp < h_a.LS(a, t1, t2) then
9                  s_a := max(s_a, t2 − (1/h_a) × Disp);
10             end
11             if Disp < h_a.RS(a, t1, t2) then
12                 e_a := min(e_a, t1 + (1/h_a) × Disp);
13             end
14         end
15     end
16 end
```

conditions	intervalle	
$\underline{s_j} \leq \underline{s_i} \wedge \overline{e_i} < \overline{e_j}$	$[\underline{s_j},\ \overline{e_i}]$	LS_j1
$\underline{s_j} > \underline{s_i} \wedge \underline{s_j} < \underline{e_i} \wedge \underline{s_j} < \overline{s_i} \wedge \underline{s_i} + \overline{e_i} - \underline{e_j} > \underline{s_j} \wedge \underline{s_i} + \overline{e_i} - \underline{e_j} < \overline{e_j}$	$[\underline{s_j},\ \underline{s_i} + \overline{e_i} - \underline{e_j}]$	LS_j2
$\underline{s_j} > \underline{s_i} \wedge \underline{s_j} < \underline{e_i} \wedge \underline{s_j} \geq \overline{s_i} \wedge \overline{e_i} < \overline{e_j}$	$[\underline{s_j},\ \underline{e_i}]$	LS_j3
$\underline{s_j} < \overline{s_i} \wedge \overline{e_j} < \underline{e_i}$	$[\underline{s_j},\ \underline{e_i}]$	LS_i4
$\underline{s_j} + \overline{e_j} - \underline{e_i} < \underline{e_i} \wedge \underline{s_j} + \overline{e_j} - \underline{e_i} < \overline{s_i} \wedge \underline{e_i} < \overline{e_j} \wedge \underline{e_i} > \overline{s_j} \wedge \underline{e_i} > \underline{e_j}$	$[\underline{s_j} + \overline{e_j} - \underline{e_i},\ \underline{e_i}]$	LS_i5
$\overline{s_j} < \overline{s_i} \wedge \underline{e_i} < \overline{e_j} \wedge \underline{e_i} < \overline{s_j} \wedge \underline{e_i} \leq \underline{e_j}$	$[\overline{s_j},\ \underline{e_i}]$	LS_i6
	$[\underline{s_i},\ \underline{e_i}]$	LS_j7

TABLE 6 – Intervalles d'intérêt pour le placement à gauche.

conditions	intervalle	
$\underline{s_j} \leq \underline{s_i} \wedge \overline{e_i} > \underline{e_j}$	$[\overline{s_j},\ \overline{e_i}]$	RS_j1
$\overline{s_j} > \underline{s_i} \wedge \overline{s_j} < \underline{e_i} \wedge \overline{s_j} < \overline{s_i} \wedge \underline{s_i} + \overline{e_i} - \overline{s_j} < \overline{s_j} \wedge \underline{s_i} + \overline{e_i} - \overline{s_j} < \underline{e_j}$	$[\overline{s_j},\ \underline{s_i} + \overline{e_i} - \overline{s_j}]$	RS_j2
$\overline{s_j} > \underline{s_i} \wedge \overline{s_j} < \underline{e_i} \wedge \overline{s_j} \geq \overline{s_i} \wedge \underline{e_i} > \underline{e_j}$	$[\overline{s_j},\ \underline{e_i}]$	RS_j3
$\underline{s_j} > \underline{s_i} \wedge \overline{e_i} > \overline{e_j}$	$[\underline{s_j},\ \overline{e_i}]$	RS_i4
$\underline{s_j} + \overline{e_j} - \overline{e_i} < \overline{e_i} \wedge \underline{s_j} + \overline{e_j} - \overline{e_i} > \underline{s_i} \wedge \overline{e_i} < \underline{e_j} \wedge \overline{e_i} > \underline{e_j}$	$[\underline{s_j} + \overline{e_j} - \overline{e_i},\ \overline{e_i}]$	RS_i5
$\overline{s_j} > \underline{s_i} \wedge \overline{e_i} < \overline{e_j} \wedge \overline{e_i} > \overline{s_j} \wedge \overline{e_i} \leq \underline{e_j}$	$[\overline{s_j},\ \overline{e_i}]$	RS_i6
	$[\overline{s_i},\ \overline{e_i}]$	RS_i7

TABLE 7 – Intervalles d'intérêt pour le placement à droite.

L'algorithme doit ensuite calculer les intervalles d'intérêt pour chaque activité (il en existe $4n - 2$).

foreach *activité* $a \in \mathcal{A}$ **do**
  foreach $[t_1, t_2[$ *d'intérêt pour* LS_a *(cf Table 6)* **do**
    $Disp := C \times (t_2 - t_1) - \sum_{i \in \mathcal{A} \setminus a} MI(i, t_1, t_2)$;
    if $Disp < h_a.LS(a, t_1, t_2)$ **then**
      $\underline{s_a} := \max(\underline{s_a},\ t_2 - \frac{1}{h_a} \times Disp)$;
    end
  end
  foreach $[t_1, t_2[$ *d'intérêt pour* RS_a *(cf Table 7)* **do**
    $Disp := C \times (t_2 - t_1) - \sum_{i \in \mathcal{A} \setminus a} MI(i, t_1, t_2)$;
    if $Disp < h_a.RS(a, t_1, t_2)$ **then**
      $\overline{e_a} := \min(\overline{e_a},\ t_1 + \frac{1}{h_a} \times Disp)$;
    end
  end
end

Algorithm 2: Propagateur énergétique.

5 Expérimentations

Nous avons effectué des expérimentations en Choco [9] version 3 (release 13.03), sur un ordinateur doté d'un processeur 2.9 GHz Intel Core i7. Nous avons effectué des tests sur des instances générées aléatoirement ainsi que sur des instances de la PSPLIB[6]. Les instances aléatoires ont 10 ou 20 activités, chacune ayant une durée choisie dans l'intervalle $[1, 10]$, et une hauteur dans $[1, 5]$. Nous avons utilisé une heuristique de branchement *first fail* [5].

5.1 Expérimentations du checker énergétique

Nous avons comparé trois algorithmes :
- L'algorithme 1, ERC ;
- L'algorithme de Baptiste et al. [2] ;
- L'algorithme naïf en $O(n^3)$ raisonnant sur l'ensemble des intervalles d'intérêt définis par la Table 3.

Le nombre de noeuds est identique pour toutes les instances, pour les trois algorithmes, comme nous l'attendions. La Table 10 montre un gain de temps d'exécution de l'ordre de 20 à 30% en utilisant l'algorithme 1 en comparaison à celui de Baptiste et al.

Instances	Algorithm 1 ($\mu s/node$)	Baptiste et al. ($\mu s/node$)	$O(n^3)$ ($\mu s/node$)
Random10	16.47	24.97	29.31
Random20	43.95	56.24	78.74
PspLib 30	450.67	618.77	1268.92
PspLib 120	1 339.24	1 683.26	11 288.54

TABLE 8 – Temps moyen par noeud

Nous avons aussi utilisé notre checker énergétique en combinaison avec l'algorithme de Time-Table de Letort et al. [7], en comparaison avec l'algorithme de propagation Time-Table Edge-Finding de Vilim [10]. Nous avons fixé une limite de temps à 5 minutes. De façon surprenante, sur les problèmes générés aléatoirement (mono ressource), le checker énergétique a pu dans le temps imparti prouver l'optimalité de 72 des 100 instances, alors que l'algorithme TTEF n'a pu en prouver que 7. Ces résultats montrent l'importance que peut avoir un checker énergétique dans un moteur de résolution de contraintes, dans certains contextes d'utilisation.

	TT	TT + TTEF	TT + Algorithm 1
Random20	6	7	72

TABLE 9 – #prouvé sur 100 instances.

5.2 Expérimentations du propagateur énergétique

Nous avons ensuite effectué des expérimentations du nouvel algorithme de propagation pour le raisonnement énergétique (Algorithme 2), en comparant deux algorithmes :

– L'algorithme 2 ;
– L'algorithme de Baptiste et al. [2] ;

Le nombre de noeuds est identique pour toutes les instances, conformément à ce que nous attendions. La Table 9 montre un temps d'exécution réduit d'un facteur 2 à 4 par l'utilisation de notre algorithme 2 en comparaison à celui de Baptiste et al.

Instances	Algorithm 2 ($\mu s/node$)	Baptiste et al. ($\mu s/node$)
Random10	91	244
Random20	327	641
PspLib 30	4 372	8 809
PspLib 120	41 418	151 390

TABLE 10 – Temps moyen par noeud

Nous avons aussi combiné notre propagateur énergétique avec l'algorithme de Time-Table, sur les mêmes 100 instances que celles présentées dans la Table 9. Cette combinaison a démontré l'optimalité pour 63 instances.

6 Conclusion et perspectives

Dans cet article nous avons proposé une nouvelle caractérisation des intervalles à considérer pour le checker et pour le propagateur énergétique. Notre caractérisation réduit le nombre d'intervalles d'intérêt par un facteur 7. De plus nous avons démontré que la caractérisation de Baptiste et al. est suffisante pour assurer un filtrage énergétique complet.

Nous avons de plus montré que l'utilisation de notre checker énergétique est le meilleur compromis pour prouver l'optimalité d'une certaine classe de problème.

Le filtrage énergétique est l'un des plus performant pour le problème cumulatif, ce papier montre qu'il est possible d'en améliorer le temps d'exécution sans perte de filtrage. Ceci ouvre des perspectives intéressantes pour obtenir un propagateur proposant un bon compromis filtrage-temps d'exécution.

Références

[1] Abder Aggoun and Nicolas Beldiceanu. Extending chip in order to solve complex scheduling and placement problems. *Math. Comput. Model.*, 17(7) :57–73, April 1993.

[2] Philippe Baptiste, Claude Le Pape, and Wim Nuijten. *Constraint-Based Scheduling : Applying Constraint Programming to Scheduling Problems*. International Series in Operations Research and Management Science. Kluwer, 2001.

[3] Timo Berthold, Stefan Heinz, and Jens Schulz. An approximative criterion for the potential of energetic reasoning. In *Proceedings of the First international ICST conference on Theory and practice of algorithms in (computer) systems*, TAPAS'11, pages 229–239, Berlin, Heidelberg, 2011. Springer-Verlag.

[4] Jacques Erschler, Pierre Lopez, and Catherine Thuriot. Scheduling under time and resource constraints. In *Proc. of Workshop on Manufacturing Scheduling, 11th IJCAI*, Detroit, USA, 1989.

[5] Robert M. Haralick and Gordon L. Elliot. Increasing tree search efficiency for constraint satisfaction problems. *Artificial Intelligence*, 14(3) :263–313, 1980.

[6] Rainer Kolisch and Arno Sprecher. Psplib – a project scheduling problem library. *EUROPEAN JOURNAL OF OPERATIONAL RESEARCH*, 96 :205–216, 1996.

[7] Arnaud Letort, Nicolas Beldiceanu, and Mats Carlsson. A scalable sweep algorithm for the cumulative constraint. In Michela Milano, editor, *Principles and Practice of Constraint Programming*, Lecture Notes in Computer Science, pages 439–454. Springer Berlin Heidelberg, 2012.

[8] Christoph Schwindt. *Verfahren zur Lösung des ressourcenbeschränkten Projektdauerminimierungsproblems mit planungsabhängigen Zeitfenstern*. PhD thesis, Fakultät für wirtschaftswissenschaften der Universität Fridericiana zu Karlsruhe, 1998. in German.

[9] CHOCO Team. Choco : an open source Java CP library. Research report 10-02-INFO, Ecole des Mines de Nantes, 2010.

[10] Petr Vilím. Timetable edge finding filtering algorithm for discrete cumulative resources. In Tobias Achterberg and J. Beck, editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, volume 6697 of *Lecture Notes in Computer Science*, pages 230–245. Springer Berlin / Heidelberg, 2011.