Ordonnancement SJF multicœurs

Motivation : Ordonnancement optimal pour des tâches connues à l'avance : utile pour des systèmes de traitement par lots. On généralise ici au cas multicœurs.

Problème :

Entrées : · $\{J_i\}_i$ ensemble de n tâches indépendantes, et $\{d_j\}_{j\in J}$ ensemble des durées associées.

· M cœurs indépendants notés $K_1, \dots, K_M$

Sortie : Liste des tâches ordonnées associées à chaque cœur minimisant la somme des dates de fin.

Définitions : On note c_j la date de fin de la tâche $j \in J$, et T_i les tâches (ordonnées) du cœur K_i.

Algorithme Shortest Job First multicœurs :

- Trier D par ordre croissant
- $T_i \leftarrow J_i, J_{M+i}, J_{2M+i}, \dots$

On se ramène à $n \equiv 0 \ [M]$ en ajoutant des 0 à D.

Exemple : $D = \{1, 1, 3, 4, 7\}$, $M = 3$

↳ $D' = \{0, 1, 1, 3, 4, 7\}$

K_1	K_2	K_3
0	1	1
3	4	7

$\rightarrow \sum_{j\in J} c_j = 2\times 0 + 3 + 2\times 1 + 4 + 2\times 1 + 7 = 18$

Propriété 1 : Ajouter des tâches de durée 0 ne change pas le coût total de l'ordonnancement renvoyé par l'algorithme.

Propriété 2 : Les ordonnancements de coût minimal sont équilibrés : chaque processeur a le même nombre de tâches.

Preuve : Par l'absurde, si $T_1, \dots, T_M$ est un ordonnancement de coût minimal avec $|T_i| > \frac{n}{M}$ et $|T_j| < \frac{n}{M}$, alors :

On pose : $T_i' \leftarrow T_i \setminus \{\alpha\}$ (avec $\alpha = \min(T_i)$) et

$T_j' \leftarrow \{\alpha\} \sqcup T_j$.

Alors : $$\sum_{t \in T_i'} c_t + \sum_{t \in T_j'} c_t = \sum_{t \in T_i} c_t - \alpha |T_i| + \sum_{t \in T_j} c_t + \alpha |T_j|$$

$$= \sum_{t \in T_i} c_t + \sum_{t \in T_j} c_t + \alpha \underbrace{(|T_j| - |T_i|)}_{<0}$$

$$< \sum_{t \in T_i} c_t + \sum_{t \in T_j} c_t$$

Donc : on a construit un nouvel ordonnancement de coût inférieur : absurde.

Propriété 3 : Soient $(a_i)_i$ et $(b_i)_i$ deux familles de $\mathbb{N}$.
Si (b_i) est décroissante, alors $\sum_{i=1}^{n} a_{\sigma(i)} \times b_i = \langle a | b \rangle$ est minimale pour σ telle que $(a_{\sigma(i)})$ est croissante et σ permutation.

Exemple : $\langle (3,1,2) | (7,4,2) \rangle = 29$

$\langle (1,2,3) | (7,4,2) \rangle = 21$

Preuve : On montre qu'à partir de σ, on peut se ramener à σ^* telle que $(a_{\sigma^*(i)})$ ↗ et $\langle (a_{\sigma(i)}) | (b_i) \rangle \geq \langle (a_{\sigma'(i)}) | (b_i) \rangle$

Soit σ une permutation telle que $\exists i<j$, $a_{\sigma(j)} < a_{\sigma(j)}$.

On pose : $\sigma' = \sigma \circ (i\ j)$:

$$\underbrace{\left(a_{\sigma(j)} - a_{\sigma(i)}\right)}_{<0} * \underbrace{\left(b_j - b_i\right)}_{\leq 0} \geq 0$$

$$\Leftrightarrow \quad a_{\sigma(j)} b_j - a_{\sigma(j)} b_i - a_{\sigma(i)} b_j + a_{\sigma(i)} b_i \geq 0.$$

$$\Leftrightarrow \quad a_{\sigma(j)} b_j + a_{\sigma(i)} b_i \geq a_{\sigma(j)} b_i + a_{\sigma(i)} b_j$$

$$= a_{\sigma'(i)} b_i + a_{\sigma'(j)} b_j$$

Donc $\langle (a_{\sigma'(i)}) | (b_i) \rangle \leq \langle (a_{\sigma(i)}) | (b_i) \rangle$.

On construit σ^* en itérant pour tous les couples $(i\ j)$.

Théorème : L'ordonnancement donné par l'algorithme a un coût minimal.

Preuve : On se ramène au cas équilibré par la propriété 2.

$$\sum_{t \in J} C_t = \sum_{i=1}^{n} d_i \times \left\lceil \frac{n-i}{M} \right\rceil$$: la tâche i est comptée $\left\lceil \frac{n-i}{M} \right\rceil$ fois

$$= \langle \underbrace{(d_i)}_{\nearrow} | \underbrace{\left(\left\lceil \frac{n-i}{M} \right\rceil\right)}_{\searrow} \rangle$$ est donc optimal par la propriété 3