MPRO - Examen d'ordonnancement 2018

David Savourey, Christophe Picouleau

8 janvier 2018

L'examen dure 2h30. Les notes de cours sont autorisées. Les livres ne sont pas autorisés. La clarté de rédaction sera prise en compte dans la notation.

1 Bornes inférieures pour $1|r_i|\sum C_i$ et $1|r_i|\sum T_i$

On s'intéresse au problème d'ordonnancement à une machine suivant : on dispose de n jobs, chaque job étant défini une durée p_i et une date de disponibilité r_i. On cherche à minimiser la somme des dates de fin. Le problème étudié se note $1|r_i|\sum C_i$.

Question 1.1. *Dans le cas où les dates de disponibilité sont nulles (problème $1||\sum C_i$), rappeler comment résoudre optimalement le problème en temps polynomial.*

Étant donnés un problème P et un entier a, on note P_a le sous-problème où seuls les jobs de P qui ont une date de disponibilité égale à a sont conservés.

On note $opt(P)$ la valeur d'une solution optimale du problème P.

Question 1.2. *Soit un problème P où toutes les dates de disponibilité ne peuvent prendre que deux valeurs distinctes : $\exists a, b > 0, \forall i \in [1, n], r_i \in \{a, b\}$. À quelle condition a-t-on $opt(P) = opt(P_a)+opt(P_b)$? Lorsque cette condition n'est pas vérifiée, que peut-on dire de la valeur $opt(P_a) + opt(P_b)$ par rapport au problème P ?*

Question 1.3. *En déduire une borne inférieure dans le cas où les dates de disponibilité sont quelconques. En donner la complexité. Justifier brièvement.*

Question 1.4. *Généraliser ce calcul en passant de 2 à k valeurs distinctes pour les dates de disponibilité. Donner la complexité. Justifier brièvement.*

Nous venons de calculer des bornes inférieures pour le problème $1|r_i|\sum C_i$. Nous souhaitons désormais en déduire des bornes inférieures pour le problème $1|r_i|\sum T_i$.

Soit I une instance pour le problème $1|r_i|\sum T_i$ (chaque job est défini par une durée p_i, une date de disponibilité r_i et une date d'échéance d_i). On note I_C l'instance pour le problème $1|r_i|\sum C_i$, obtenue à partir de I en ne retenant que les durées et les dates de disponibilité dans I.

Étant donnée une instance J associée à un problème P, on note $LB(J,P)$ une borne inférieure valide pour cette instance.

Question 1.5. *En se basant sur la définition du retard T_i, proposer une méthode pour calculer $LB(I, 1|r_i|\sum T_i)$ à partir de $LB(I_C, 1|r_i|\sum C_i)$. Justifier brièvement.*

2 Borne inférieure de l'algorithme approché pour $UET - UCT$

Nous avons vu en cours que le programme linéaire suivant modélise le problème $UET-UCT$ avec un nombre non limité de processeurs. Le graphe de précédence est noté $G = (I, <)$.

$$
\begin{array}{rcll}
\min D & & & \\
t_i + 1 + x_{ij} & \leq & t_j & \forall (i,j) \in < \\
\Sigma_{j \in \Gamma^+(i)} x_{ij} & \geq & |\Gamma^+(i)| - 1 & \forall i, \Gamma^+(i) \neq \emptyset \\
\Sigma_{j \in \Gamma^-(i)} x_{ji} & \geq & |\Gamma^-(i)| - 1 & \forall i, \Gamma^-(i) \neq \emptyset \\
t_i + 1 & \leq & D & \forall i \in I \\
t_i & \geq & 0 & \forall i \in I \\
x_{ij} \in \{0,1\} & & & \forall (i,j) \in <
\end{array}
$$

Nous avons montrer qu'un algorithme basé sur la relaxation continue du programme linéaire garantit une performance relative de ratio 4/3, c'est-à-dire que pour toute instance $G = (I, <)$, le rapport entre $A(G)$, la valeur de la solution calculée par l'algoritme, et $OPT(G)$, la valeur d'une solution optimale, vérifie $A(G)/OPT(G) \leq 4/3$.

Rappelons que l'algorithme basé sur la relaxation continue du programme linéaire utilise la technique d'arrondi suivante :

— si $x^*_{ij} < 1/2$ alors $x_{ij} = 0$;

— si $x^*_{ij} \geq 1/2$ alors $x_{ij} = 1$;

où x^*_{ij} est la variable continue correspondant à la relaxation de la contrainte d'intégrité de x_{ij}.

Nous allons montrer que la borne 4/3 est atteinte asymptotiquement.

Pour tout entier $n \geq 1$, soit $G_n = (I_n, <)$ le graphe de précédence suivant :

— $I_n = \{a_0, a_1, \ldots, a_n\} \cup \{b_1, \ldots, b_n\} \cup \{c_1, \ldots, c_n\}$;

— $\Gamma^+(a_0) = \{b_1, c_1\}, \Gamma^+(a_1) = \{b_2, c_2\}, \ldots, \Gamma^+(a_{n-1}) = \{b_n, c_n\}$;

— $\Gamma^-(a_1) = \{b_1, c_1\}, \Gamma^-(a_2) = \{b_2, c_2\}, \ldots, \Gamma^-(a_n) = \{b_n, c_n\}$.

Question 2.1. *Montrer qu'il existe une solution optimale de la relaxation continue du programme linéaire dans laquelle*

$$x^*_{ij} = 1/2, \ \forall (i,j) \in <$$

Question 2.2. *Pour cette solution optimale de la relaxation continue du programme linéaire donner en fonction de n, en le justifiant, la durée $A(G_n)$ de l'ordonnancement fourni par l'algorithme.*

Question 2.3. *Donner en fonction de n, en le justifiant, $OPT(G_n)$ la durée optimale d'un ordonnancement.*

Question 2.4. *Déduire des questions précédentes que*

$$\lim_{n \to \infty} \frac{A(G_n)}{OPT(G_n)} = 4/3$$

Question 2.5. *Montrer que la solution optimale de la relaxation continue du programme linéaire dans laquelle $x^*_{ij} = 1/2, \ \forall (i,j) \in <$, n'est pas unique. Donner parmi les solutions optimales équivalentes une solution amenant à une solution optimale pour G_n une fois la technique d'arrondi mise en œuvre.*