MPRO - Examen d'ordonnancement 2015

Christophe Picouleau, David Savourey

12 janvier 2015

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 Le problème $1|prec, pmtn, r_j|F_{max}$

On souhaite résoudre le problème d'ordonnancement suivant : on dispose de n jobs qui doivent être ordonnancés sur une machine. La préemption est autorisée sur cette machine. Chaque job i est soumis à une date de disponibilité r_i. Les jobs sont également soumis à des contraintes de précédence $i \to j$, signifiant que l'exécution du job j ne peut pas débuter avant la fin de l'exécution du job i.

La fonctionnelle F_{max} que l'on souhaite minimiser est de la forme $F_{max} = \max_{j=1}^{n} f_j(C_j)$, où C_j représente la date de fin d'exécution du job j. Les fonctions f_j ne sont pas fixées, on suppose simplement qu'elles sont croissantes.

On suppose que les jobs sont indicés selon un ordre topologique : si $i \to j$, alors $i < j$.

Question 1.1. *Rappeler brièvement une méthode pour résoudre optimalement le problème $1|prec|F_{max}$.*

On se propose tout d'abord d'augmenter les dates de disponibilités r_j par l'algorithme 1 :

Question 1.2. *Pourquoi cette modification n'augmente-t-elle pas la valeur optimale du problème ?*

pour $i = 1$ *à* $n - 1$ **faire**
  pour $j = i + 1$ *à* n **faire**
    si $i \rightarrow j$ **alors**
      $r_j = max(r_j, r_i + p_i)$

Algorithme 1 : Modification des dates de disponibilité

Comme les jobs sont soumis à des dates de disponibilité, les ordonnancements réalisables ont potentiellement des temps morts. On dit que les ordonnancements sont composés de blocs. Plus précisément, un bloc est un ensemble maximal de jobs ordonnancés sans temps morts.

L'algorithme 2 décrit le calcul des blocs.

$i = 1$, $j = 1$;
tant que $i \leq n$ **faire**
  $t = r_i$, $B_j = \emptyset$;
  tant que $r_i \leq t$ *et* $i \leq n$ **faire**
    $B_j = B_j \cup \{i\}$;
    $t = t + p_i$;
    $C_i = t$;
    $i = i + 1$;
  $j = j + 1$;

Algorithme 2 : Construction des blocs

Voici un exemple d'ordonnancement produit par cet algorithme :

Le bloc B_1 est composé des jobs 1,2 et 3. Le bloc B_2 est composé des jobs 4 et 5. Le bloc B_3 est composé du job 6.

Pour le bloc B_u, on note $s(B_u)$ sa date de début, $t(B_u)$ sa date de fin, et $p(B_u)$ sa durée.

Question 1.3. *L'ordonnancement produit par l'algorithme 2 est-il toujours réalisable ? Justifier brièvement.*

Question 1.4. *Montrer qu'il existe un ordonnancement optimal pour le problème $1|prec, pmtn, r_j|F_{max}$ pour lequel les temps morts coincïdent exactement avec ceux de l'ordonnancement produit par l'algorithme 2.*

On peut facilement constater que la résolution globale du problème se décompose de la manière suivante : appliquer l'algorithme des blocs, résoudre optimalement chaque sous-problème formé par un bloc, déduire la valeur optimale globale comme le maximum des valeurs optimales pour chacun des blocs.

On note $f^*_{max}(S)$ la valeur optimale du problème composé de l'ensemble de jobs S.

Par exemple, sur l'exemple de la figure, il faut résoudre le problème pour l'ensemble de jobs $B_1 = \{1, 2, 3\}$ et ainsi obtenir une valeur optimale $f^*_{max}(B_1)$. Faire de même pour B_2 puis B_3. Finalement, la valeur optimale du problème est :

$$f^*_{max}(N = \{1, 2, 3, 4, 5, 6\}) = max(f^*_{max}(B_1), f^*_{max}(B_2), f^*_{max}(B_3)).$$

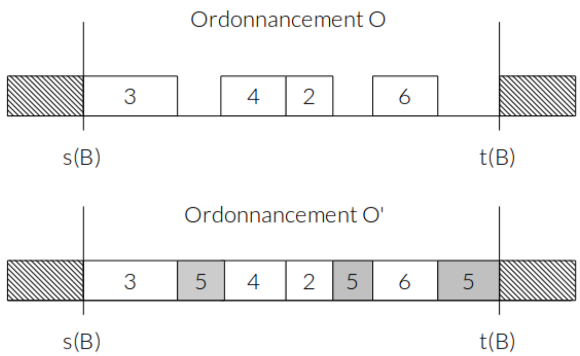
Question 1.5. *Soit B un bloc. Expliquer succintement pourquoi $f^*_{max}(B) \geq \max_{j \in B} f^*_{max}(B \setminus \{j\})$.*

On note $f_l(t(B)) = \min(f_j(t(B), j \in B$ et j sans successeurs dans $B))$. Le job l est donc celui qui minimise le coût s'il est ordonnancé à la date $t(B)$, parmi les jobs sans successeurs dans le bloc B.

Question 1.6. *Soit B un bloc. Expliquer pourquoi $f^*_{max}(B) \geq f_l(t(B))$.*

Soit un bloc B. Retirons de B le job l. Maintenant, on note O un ordonnancement optimal pour l'ensemble $B \setminus \{l\}$. Par définition, on a donc $f^*_{max}(B \setminus \{l\}) = f^*(O)$. On note alors O' l'ordonnancement obtenu en ajoutant à O le job l, dans les temps morts laissés par O entre les dates $s(B)$ et $t(B)$.

En voici une illustration. Supposons que $B = \{2, 3, 4, 5, 6\}$. Supposons que $l = 5$.



L'ordonnancement O' est clairement réalisable pour le sous-problème sur le bloc B.

Question 1.7. *Montrer que* $f_{max}(O') = \max(f^*_{max}(B \setminus \{l\}), f_l(t(B)))$*. En déduire que* O' *est optimal pour le sous-problème du bloc* B*.*

Question 1.8. *Proposer désormais une méthode générale de résolution pour le problème* $1|prec, pmtn, r_j|F_{max}$*.*