MPRO - BOR
Bases de l'ordonnancement

Examen Janvier 2020

L'examen dure 3h. Les notes de cours sont autorisées. Les livres ne sont pas autorisés. La clarté de rédaction sera prise en compte dans la notation. Le barème est indicatif.

Exercice 1 (5 points)

On considère l'énoncé E suivant du problème $1|r_i, d_i, pmtn, prec|T_{max}$ où il faut ordonnancer sur une machine, n tâches préemptives $J_1, \cdots, J_n$ de durées $p_1, \cdots, p_n$, soumises à des contraintes de précédence, des dates de disponibilité r_i et des dates d'échéance d_i, de manière à minimiser le retard maximal.
Le graphe des relations de précédence de E est représenté sur la figure 1.

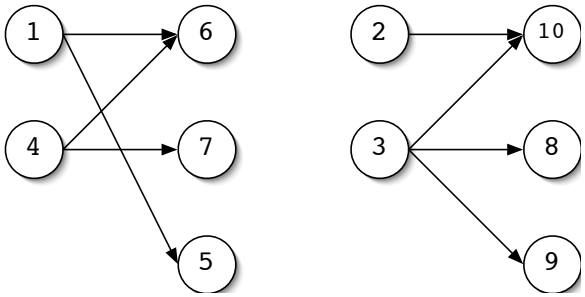

FIGURE 1 – Le graphe des précédences

Le tableau suivant donne les valeurs des durées, des dates de disponibilités et des dates d'échéance des 10 tâches de E.

i	1	2	3	4	5	6	7	8	9	10
p_i	2	1	4	1	4	3	3	2	1	1
r_i	0	1	3	4	7	10	13	16	18	19
d_i	5	4	10	7	22	15	19	20	21	21

Question 1 (1/5) — Les dates de disponibilités de E sont-elles compatibles avec le graphe des précédences de E ?
Question 2 (2/5) — Déterminer l'ordonnancement non préemptif associé à la séquence $(1, 2, \cdots, n)$.
Question 3 (2/5) — Calculer un ordonnancement optimal de E. Vous préciserez l'algorithme utilisé et les principales étapes de calcul.

Exercice 2 (6 points)

Partie 1

On considère le problème noté $P|p_i = 1|\sum f_i(C_i)$, où n tâches $J_1, \cdots, J_n$ indépendantes et de durée unitaire doivent être exécutées sur m machines identiques de telle sorte que $\sum_{i=1}^{n} f_i(C_i)$ soit minimale. C_i est la date de fin d'exécution de la tâche J_i et pour tout $i \in \{1, \cdots, n\}$, f_i est une fonction croissante au sens large. On pose $n = mq + r$ (division euclidienne de n par m) et l'on appelle slot t (t entier positif) l'intervalle de temps $[t-1, t]$.

Question 1 (1.5/6) — Montrer que les ordonnancements pour lesquels :

1. les m machines sont occupées pendant les slots $1, \cdots, q$;
2. r machines sont occupées pendant le slot $q+1$.

sont dominants.
Question 2 (1.5/6) — Déduire de la question précédente que le problème $P|p_i = 1|\sum f_i(C_i)$ peut être résolu par un algorithme de flot maximum de coût minimum dans un réseau que l'on précisera.

Partie 2

On considère le problème noté $P|p_i = 1|f_{max}$ où le coût maximun $f_{max} = \max_{i\in\{1,\cdots,n\}} f_i(C_i)$ de terminaison d'une tâche doit être minimisé.

Question 3 (1/6) — Montrer que la propriété de dominance de la question 1 de la partie 1 est toujours satisfaite.
Question 4 (2/6) — On pose

$$D = \begin{cases} q+1 & \text{si } r > 0 \\ q & \text{si } r = 0 \end{cases}$$

et l'on suppose que les tâches sont numérotées de telle sorte que :

$$f_1(D) \leq f_2(D) \leq \cdots \leq f_n(D).$$

Montrer qu'il existe un ordonnancement optimal tels que les tâches $J_1, \cdots, J_r$ sont exécutées pendant le slot D. En déduire un algorithme pour le calcul d'un ordonnancement optimal.

Exercice 3 (7 points)

On considère le problème d'ordonnancement à une machine et n tâches pour lequel chaque tâche J_i a une date échue $d_i = d$. On note C_i la date de fin d'exécution de la tâche J_i, p_i sa durée opératoire, α_i sa pénalité d'avance et β_i sa pénalité de retard. On s'intéresse dans le cadre de cet exercice à la **détermination de la valeur de la date d'échéance commune** d. Pour cela, on introduit une pénalité d'échéance γ dans la fonction à optimiser. Le critère de minimisation est alors $\sum_{i=1}^{n} \alpha_i E_i + \beta_i T_i + \gamma d$ où $E_i = \max(d - C_i, 0)$ et $T_i = \max(C_i - d, 0)$ représentent respectivement l'avance et le retard de la tâche J_i.

Question 1 (0.5/7) — Que peut-on dire de la complexité du problème $1||\sum_{i=1}^{n} \alpha_i E_i + \beta_i T_i + \gamma d$?

On se place dans la suite de l'exercice dans le cas où $p_i = 1$.
Question 2 (1/7) — Montrer que l'ordonnancement optimal ne contient pas de temps mort entre deux tâches consécutives.
Question 3 (2/7) — Montrer que la propriété en V où les tâches en avance ou à l'heure sont séquencées dans l'ordre décroissant des valeurs $1/\alpha_i$ et les tâches en retard dans l'ordre croissant des $1/\beta_i$ pour un ordonnancement optimal est vérifiée.
Question 3 (0.5/7) — Que peut-on dire de la valeur optimale de d pour une valeur de γ très élevée ?
Question 4 (2/7) — Montrer que dans un ordonnancement optimal, $d = 0$ ou d coïncide avec la date de fin d'exécution d'une des tâches.
Question 5 (1/7) — En tenant compte de l'hypothèse $p_i = 1$, que peut-on dire de l'ensemble des valeurs possibles de d ?
En déduire un algorithme de résolution du problème dans le cas où la séquence de tâches est fixée.

Exercice 4 (5 points)

Ordonnancement avec communications

On considère un problème d'ordonnancement avec communications *sans duplication*, sous l'hypothèse *petits délais de communication*, i.e. $\max\{c_{ij}\} \leq \min\{p_i\}$. Le *nombre de processeurs est illimité*.

On note $G = (T, A)$ le graphe de précédence.

Pour toute tâche $i \in T$ on notera $t_i \geq 0$ sa date de début d'exécution, π_i le processeur l'exécutant.

Le critère à minimiser est la *durée de l'ordonnancement*, i.e. $\min_{\mathcal{O}} \max_{i \in T}\{C_i\}$ où $\mathcal{O}$ est l'ensemble des ordonnancements réalisables.

Deux tâches u et v sont dites *indépendantes* s'il n'existe pas de chemin de u à v ou de v à u dans G.

Question 1 —Montrer qu'il existe un ordonnancement optimal tel que pour toute paire de tâches indépendantes $\{i, j\}$, on a $\pi_i \neq \pi_j$.

Deux tâches u et v constituent un *pont* (u, v) si v est l'*unique successeur* de u, et u est l'*unique prédécesseur* de v.

Question 2 —Montrer qu'il existe un ordonnancement optimal tel que pour tout pont (i, j), on a $\pi_i = \pi_j$ et $t_j = t_i + p_i$.