Ordonnancement avec préemption

Épreuve pratique d'algorithmique et de programmation
Concours commun des écoles normales supérieures

Durée de l'épreuve: 3 heures 30 minutes

Juillet 2009

ATTENTION !

N'oubliez en aucun cas de recopier votre u_0
à l'emplacement prévu sur votre fiche réponse

Important.

Sur votre table est indiqué un numéro u_0 qui servira d'entrée à vos programmes. Les réponses attendues sont généralement courtes et doivent être données sur la fiche réponse fournie à la fin du sujet. À la fin du sujet, vous trouverez en fait deux fiches réponses. La première est un exemple des réponses attendues pour un $\widetilde{u_0}$ particulier (précisé sur cette même fiche et que nous notons avec un tilde pour éviter toute confusion!). Cette fiche est destinée à vous aider à vérifier le résultat de vos programmes en les testant avec $\widetilde{u_0}$ au lieu de u_0. Vous indiquerez vos réponses (correspondant à votre u_0) sur la seconde et vous la remettrez à l'examinateur à la fin de l'épreuve.

En ce qui concerne la partie orale de l'examen, lorsque la description d'un algorithme est demandée, vous devez présenter son fonctionnement de façon schématique, courte et précise. Vous ne devez en aucun cas recopier le code de vos procédures!
Quand on demande la complexité en temps ou en mémoire d'un algorithme en fonction d'un paramètre n, on demande l'ordre de grandeur en fonction du paramètre, par exemple: $O(n^2)$, $O(n \log n)$,...
Il est recommandé de commencer par lancer vos programmes sur de petites valeurs des paramètres et de ***tester vos programmes sur des petits exemples que vous aurez résolus préalablement à la main ou bien à l'aide de la fiche réponse type fournie en annexe.*** Enfin, il est recommandé de lire l'intégralité du sujet avant de commencer afin d'effectuer les bons choix de structures de données dès le début.

1 Notations

Dans cette épreuve, on utilisera les notations suivantes :
- $\mathbb{N}$ représente l'ensemble des entiers ;
- $\mathbb{R}$ représente l'ensemble des réels ;
- $\mathbb{I}$ représente l'ensemble des unions finies d'intervalles de $\mathbb{R}$ ouverts à droite et de longueur strictement positive.
- $\lceil x \rceil$ dénote la partie entière supérieure de x.
- On définit l'ordre strict $\prec$ sur l'ensemble des permutations de {0,...,n-1} par :

$$\pi \prec \mu \Leftrightarrow \exists i \in \{0, \ldots, n-1\}, \forall j < i : \pi(j) \leqslant \mu(j) \text{ et } \pi(i) < \mu(i).$$

Notons que cet ordre est complet.

2 Introduction

On dispose d'un serveur (un ordinateur) dont la fonction est de traiter des requêtes (ou tâches, les deux termes étant utilisés de façon équivalente par la suite). Ces requêtes arrivent au fur et à mesure et le temps nécessaire à leur traitement par le serveur est propre à chacune d'elle et connue du serveur.

Définition 1 (Tâche) *Une tâche T_i est donc définie par une date de soumission $r_i \in \mathbb{R}$ et un temps de traitement $p_i \in \mathbb{R}_+^*$. On appellera instance un ensemble fini de tâches.*

Étant donné une instance $\mathcal{I}$ de n requêtes, le serveur doit donc décider à chaque instant sur quelle requête il travaille. Il se peut également que le serveur reste inactif, par exemple parce qu'aucune requête n'est en mesure d'être exécutée. On note donc *Inactif* un tel état.

Définition 2 (Ordonnancement) *Un ordonnancement préemptif d'une instance $\mathcal{I}$ de n requêtes, σ, est donc une fonction de $\mathbb{R}$ dans $\mathcal{I} \cup \{Inactif\}$ telle que pour tout $T_i = (r_i, p_i) \in \mathcal{I}$:*

1. *$\sigma^{-1}(T_i) \in \mathbb{I}$ (la tâche T_i n'est préemptée qu'un nombre fini de fois) ;*
2. *$\sigma^{-1}(T_i) \subset [r_i, +\infty[$ (le traitement de la tâche T_i ne commence qu'après sa date de soumission) ;*
3. *$\displaystyle\int_{t \in \sigma^{-1}(T_i)} 1.dt = p_i$ (la tâche T_i est complètement traitée).*

Définition 3 (Date de complétion) *On définit le temps de traitement restant $\varrho_i^\sigma(t)$ de la tâche T_i pour un ordonnancement σ, à la date t par :*

$$\varrho_i^\sigma(t) = p_i - \int_{\substack{x \in \sigma^{-1}(T_i) \\ x < t}} 1.dx$$

On définit la date de complétion C_i^σ de la tâche T_i pour un ordonnancement σ par :

$$C_i^\sigma = \sup\{t \in \sigma^{-1}(T_i)\} = \inf\{t | \varrho_i^\sigma(t) = 0\}$$

Lorsque le contexte est suffisamment clair, on allégera les notations en écrivant C_i au lieu de C_i^σ.
À partir de la notion de date de complétion, on peut également définir les deux mesures suivantes :
– *le temps de réponse de la tâche T_i est définie par $F_i = C_i - r_i$;*
– *le ralentissement de la tâche T_i est définie par $S_i = \frac{C_i - r_i}{p_i}$.*

> **Note :** Évidemment, on donne des exemples et on fait des petits dessins pour expliquer tout ça.

Intuitivement, ces mesures permettent de définir un certain degré de *satisfaction* de chacune des requêtes. Pour déterminer si un ordonnancement est satisfaisant on agrégera donc ces mesures de la façon suivante :

Pire temps de réponse : le pire temps de réponse d'un ordonnancement σ est

$$MF^\sigma = \max_i F_i.$$

Pire ralentissement : le pire ralentissement d'un ordonnancement σ est

$$MS^\sigma = \max_i S_i.$$

Temps de réponse moyen : le temps de réponse moyen d'un ordonnancement σ est

$$SF^\sigma = \frac{1}{n} \sum_i F_i.$$

Ralentissement moyen : le ralentissement moyen d'un ordonnancement σ est

$$SS^\sigma = \frac{1}{n} \sum_i S_i.$$

L'objectif de cette épreuve est l'étude des performances de divers algorithmes vis-à-vis des métriques précédentes.

3 Génération aléatoire d'instances

On note $M = 64\,007$. Considérons la suite d'entiers (u_k) définie pour $k \geqslant 0$ par :

$$u_k = \begin{cases} \textit{votre } \mathtt{u_0} \textit{ (à reporter sur votre fiche)} & \text{si } k = 0 \\ 15\,091 \times u_{k-1} \mod M & \text{si } k \geqslant 1 \end{cases}$$

Question 1 *Que valent :*
a) u_{10} ***b)*** u_{100} ***c)*** u_{1000}.

Pour $x > 0$, on définit w par :

$$w_i(x) = \left\lceil \frac{\ln M - \ln u_i}{x} \right\rceil$$

Définition 4 (Instance Poissonnienne) *Soient λ et μ deux réels strictement positifs. On définit par récurrence les suites r_i et p_i de la façon suivante :*

$$\begin{pmatrix} r_0 \\ p_0 \end{pmatrix} = \begin{pmatrix} 0 \\ w_1(\mu) \end{pmatrix} \text{ et } \begin{pmatrix} r_i \\ p_i \end{pmatrix} = \begin{pmatrix} r_{i-1} + w_{2i}(\lambda) \\ w_{2i+1}(\mu) \end{pmatrix} \text{ pour } i \geqslant 1.$$

On note $\mathcal{I}_n(\lambda, \mu)$ l'instance constituée des tâches $\{T_0, \ldots, T_{n-1}\}$.
Dans ce type de construction λ représente la vitesse moyenne d'arrivée des tâches et μ représente le temps de traitement moyen des tâches.

Question 2 *Que vaut la dernière date de soumission des instances suivantes :*
a) $\mathcal{I}_{10}(0.2, 0.05)$, ***b)*** $\mathcal{I}_{100}(0.2, 0.05)$, ***c)*** $\mathcal{I}_{1000}(0.2, 0.05)$.

4 Calcul de l'ordonnancement optimal

Soit σ un ordonnancement d'une instance $\mathcal{I}$. Considérons l'ordre π de terminaison des tâches. Soit σ_π l'ordonnancement qui ordonnance toujours au plus tôt les tâches disponibles et non terminées en respectant la priorité π. Cet ordonnancement est appelé ordonnancement de liste associé à la priorité π.
Plus formellement, σ_π est définit par :

$$\sigma_\pi(t) = T \text{ tel que } \pi(T) = \min\{\pi(T_j) | r_j \geqslant t \text{ et } \varrho_j^{\sigma_\pi}(t) > 0\}$$

Question à développer pendant l'oral : Montrer que pour tout i, $C_i^{\sigma_\pi} \leqslant C_i^{\sigma}$.
Question à développer pendant l'oral : Donner une instance et une liste de priorité telle que l'ordre de terminaison selon l'ordonnancement de liste associé est différent de la liste de priorité.

Question 3 *Donner la date de complétion de la dernière tâche soumise (c'est-à-dire T_{n-1}) pour l'ordonnancement de liste associé aux priorités et aux instances suivantes :*
a) $\mathcal{I}_{10}(0.2, 0.05)$, $\pi = (0, 2, 4, 6, 8, 1, 3, 5, 7, 9)$
b) $\mathcal{I}_{100}(0.2, 0.05)$, $\pi = (0, 2, \ldots, 98, 1, 3, \ldots, 99)$
c) $\mathcal{I}_{1000}(0.2, 0.05)$, $\pi = (0, 2, \ldots, 998, 1, 3, \ldots, 999)$

Toutes les métriques précédentes (pire temps de réponse, pire ralentissement, ...) étant croissantes en les C_i, pour trouver un ordonnancement optimal, il "suffit" donc de tester tous les ordres possibles.
Pour cela, on notera que l'ensemble des permutations $\mathbb{P}(X)$ de l'ensemble $X = \{x_1, \ldots, x_n\}$ peut se définir récursivement par

$$\mathbb{P}(X) = \bigcup_{x \in X} \bigcup_{\pi \in \mathbb{P}(X \setminus \{x\})} [x].\pi,$$

où $[x].\pi$ est la permutation qui à 0 associe x et à $i > 0$ associe $\pi(i-1)$.

Question 4 *Écrire une fonction qui énumère les permutations suivant l'ordre $\prec$ et l'utiliser pour déterminer la i-ème permutation de l'ensemble $\{0, n-1\}$ (la première permutation est bien évidemment $(0, 1, \ldots, n-1)$) pour les valeurs de i et de n suivantes :*
a) $i = u_0 + 30$, $n = 5$ ***b)*** $i = u_0 + 300$, $n = 10$ ***c)*** $i = u_0 + 30000$, $n = 10$
Note : *Il est recommandé de ne pas stocker l'ensemble des permutations mais uniquement de les énumérer.*

Question 5 *Adapter la fonction précédente afin de déterminer la performance optimale pour chacune des quatre métriques sur l'instance $\mathcal{I}_8(0.2, 0.005)$. Indiquer également la liste de priorité optimale pour chaque métrique (la plus petite pour l'ordre $\prec$ parmi les éventuelles multiples candidates).*

Question 6 *Adapter la fonction précédente afin de déterminer la meilleure performance, la performance la pire ainsi que la performance moyenne pour chacune des quatre métriques sur l'instance $\mathcal{I}_8(0.2, 0.005)$.*

Question à développer pendant l'oral : Que vous inspire ces "statistiques" sur la pertinence de ces différentes métriques.

5 Premier arrivé, premier servi !

Comme vous l'aurez probablement remarqué dans la section précédente la liste de priorité $\{0, \ldots, n-1\}$ semble optimale pour le critère du pire temps de réponse. Cette stratégie consiste à toujours servir les tâches sans interruption dans leur ordre d'arrivée et est appelée FIFO[1].
Question à développer pendant l'oral : Démontrer que cette stratégie est effectivement optimale pour le pire temps de réponse.

Question 7 *Évaluer cette stratégie vis-à-vis des quatre métriques sur les instances suivantes :*
a) $\mathcal{I}_{10}(0.2, 0.05)$, ***b)*** $\mathcal{I}_{100}(0.2, 0.05)$, ***c)*** $\mathcal{I}_{1000}(0.2, 0.05)$.

Intuitivement, cette stratégie qui consiste à toujours servir les requêtes dans leur ordre d'arrivée n'est pas très réactive puisqu'une petite tâche arrivée juste après une très grosse tâche devra attendre très longtemps alors qu'elle ne prend que très peu de temps à être traitée.

6 Les petites d'abord !

Dans cette section, nous considérons trois stratégies. La première, SPT[2], sert à chaque instant t la tâche active (c'est-à-dire disponible et non encore terminée) ayant le plus petit temps de traitement (c'est-à-dire le plus petit p_i). La seconde, SRPT[3], sert à chaque

[1] *First In First Out*
[2] *Shortest Processing Time first*
[3] *Shortest Remaining Processing Time first*

instant t la tâche active (c'est-à-dire disponible et non encore terminée) ayant le plus petit temps de traitement restant (c'est-à-dire le plus petit $\varrho_i(t)$). La troisième enfin, SWRPT[4], sert à chaque instant t la tâche active (c'est-à-dire disponible et non encore terminée) ayant le plus petit $p_i\varrho_i(t)$.
Note : En cas d'égalité de temps de traitement, de temps de traitement restant ou de temps de traitement restant pondéré, on prendra toujours la tâche soumise en premier.

6.1 SPT

Question 8 *Écrire une fonction réalisant l'ordonnancement SPT et évaluer cette stratégie vis-à-vis des quatre métriques sur les instances suivantes :*
a) $\mathcal{I}_{10}(0.2, 0.05)$, ***b)*** $\mathcal{I}_{100}(0.2, 0.05)$, ***c)*** $\mathcal{I}_{1000}(0.2, 0.05)$.

Question à développer pendant l'oral : Donner une instance simple où la stratégie SPT n'est pas optimale pour le temps de réponse moyen. Vous pourrez éventuellement vous aider des fonctions précédentes.

6.2 SRPT

Question 9 *Écrire une fonction réalisant l'ordonnancement SRPT et évaluer cette stratégie vis-à-vis des quatre métriques sur les instances suivantes :*
a) $\mathcal{I}_{10}(0.2, 0.05)$, ***b)*** $\mathcal{I}_{100}(0.2, 0.05)$, ***c)*** $\mathcal{I}_{1000}(0.2, 0.05)$.

Question à développer pendant l'oral : Démontrer que la stratégie SRPT est optimale pour le temps de réponse moyen.
Question à développer pendant l'oral : Donner une instance simple où la stratégie SRPT n'est pas optimale pour le ralentissement moyen. Vous pourrez éventuellement vous aider des fonctions précédentes.
Question à développer pendant l'oral : Proposer une instance telle que le pire temps de réponse de la stratégie SRPT est arbitrairement grand en comparaison de la stratégie FIFO.

6.3 SWRPT

Question 10 *Écrire une fonction réalisant l'ordonnancement SWRPT et évaluer cette stratégie vis-à-vis des quatre métriques sur les instances suivantes :*
a) $\mathcal{I}_{10}(0.2, 0.05)$, ***b)*** $\mathcal{I}_{100}(0.2, 0.05)$, ***c)*** $\mathcal{I}_{1000}(0.2, 0.05)$.

Question à développer pendant l'oral : Donner une instance simple où la stratégie SWRPT n'est pas optimale pour le ralentissement moyen.

[4] *Shortest Weighted Remaining Processing Time first*

7 Ordonnancement à base d'échéances

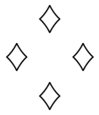
Aucune des stratégies précédentes ne semble adaptée à l'optimisation du pire ralentissement. Il existe cependant une stratégie simple basée sur la définitions d'échéances. Pour qu'un ordonnancement σ ait un pire ralentissement inférieur à S, il faut que pour toute tâche T_i, on ait :

$$S_i \leqslant S, \text{ soit } C_i \leqslant r_i + S.p_i = d_i(S).$$

Autrement dit, il faut que chaque tâche T_i termine avant l'échéance $d_i(S)$. D'autre part, pour savoir s'il est possible de respecter un ensemble d'échéances, il suffit d'ordonnancer toujours la tâche active ayant l'échéance la plus proche. Un tel ordonnancement sera appelé EDF[5].

Question 11 *Écrire une fonction permettant de déterminer si le pire ralentissement est inférieur à $S = 30$ ou pas pour les instances suivantes :*
a) $\mathcal{I}_{10}(0.2, 0.05)$, ***b)*** $\mathcal{I}_{100}(0.2, 0.05)$, ***c)*** $\mathcal{I}_{1000}(0.2, 0.05)$.

Question 12 *Écrire une fonction effectuant une dichotomie sur S pour déterminer le pire ralentissement optimal (avec une précision de 10^{-3}) sur les instances suivantes :*
a) $\mathcal{I}_{10}(0.2, 0.05)$, ***b)*** $\mathcal{I}_{100}(0.2, 0.05)$, ***c)*** $\mathcal{I}_{1000}(0.2, 0.05)$.



[5] *Earliest Deadline First*

Fiche réponse type: Ordonnancement avec préemption

$\widetilde{u_0}$: .....

Question 1

a)

b)

c)

Question 2

a)

b)

c)

Question 3

a)

b)

c)

Question 4

a)

b)

c)

Question 5

Question 6

Question 7

a)

b)

c)

Question 8

a)

b)

c)

Question 9

a)

b)

c)

Question 10

a)

b)

c)

Question 11

a)

b)

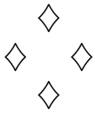
c)

Question 12

a)

b)

c)



Fiche réponse: Ordonnancement avec préemption

Nom, prénom, u_0: .....

Question 1

a)

b)

c)

Question 2

a)

b)

c)

Question 3

a)

b)

c)

Question 4

a)

b)

c)

Question 5

Question 6

Question 7

a)

b)

c)

Question 8

a)

b)

c)

Question 9

a)

b)

c)

Question 10

a)

b)

c)

Question 11

a)

b)

c)

Question 12

a)

b)

c)