Distanciel
Le problème MIN MAKESPAN

Novembre 2017 – Janvier 2018

Ce "projet distanciel" se compose de deux parties : un exercice (de type TD), et un mini-projet de programmation. Ce projet est à réaliser en **binômes** (un seul monôme autorisé par groupe de TD, si celui-ci possède un nombre impair d'étudiants). Ne pas hésiter à utiliser le forum Madoc "Échanges et questions autour du projet Min Makespan" pour échanger entre vous, et aussi poser vos questions à l'enseignant.

Un compte-rendu est à rédiger, qui contiendra donc deux parties : d'une part, vos réponses aux questions de l'exercice ; d'autre part un rapport de type "rapport de projet de programmation".

L'ensemble des éléments de ce "projet distanciel" est à rendre sous la forme d'une archive NOM1-NOM2.zip. La décompression de l'archive doit produire un répertoire NOM1-NOM2 contenant tous les éléments de votre travail (réponses aux questions de l'exercice, rapport de projet, sources du projet, exécutable du projet, jeux d'essai) et *un fichier texte contenant les instructions détaillées de compilation et d'exécution du projet*.

Cette archive est à **déposer sous Madoc**, au plus tard le **Vendredi 10 Janvier 2018 à 20h18**. Un malus sera appliqué lorsqu'une ou plusieurs des consignes données ci-dessus n'auront pas été respectées.

1 MIN MAKESPAN – Exercice

On propose un nouvel algorithme pour résoudre MIN MAKESPAN, que nous appellerons LPT (pour LARGEST PROCESSING TIME) :

(a) ordonner les tâches suivant l'ordre décroissant de leur durée : $d'_1 \geq d'_2 \geq \ldots \geq d'_n$
(b) affecter les tâches aux machines comme dans l'algorithme LSA (voir Exercice 3.2 de la feuille TD3)

1. Indiquer ce que donne l'algorithme LPT sur l'exemple de l'Exercice 3.2 de la feuille TD3, sous la forme d'un dessin similaire à la Figure 1 de l'Exercice 3.2.
2. Quel est le ratio d'approximation obtenu par LPT sur cet exemple ?

Pour toute instance I de MIN-MAKESPAN, on note $T_{LPT}(I)$ le temps obtenu par l'algorithme LPT sur I et $T_{opt}(I)$ le temps optimal recherché pour I.

3. Montrer que pour toute instance I, $T_{LPT}(I) \leq \frac{\sum_{k \neq j} d'_k}{m} + d'_j$, où j est le numéro de la tâche qui se termine en dernier.
4. Supposons pour commencer que $n \leq m$. En déduire dans ce cas que LPT est toujours optimal.

Supposons maintenant que $n \geq m + 1$.

5. Montrer que $T_{opt}(I) \geq 2d'_{m+1}$.
6. Appelons j le numéro de la tâche qui se termine en dernier quand LPT est appliqué. Montrer que l'on est nécessairement dans un des deux cas suivants : (a) $T_{LPT}(I) = T_{opt}(I)$ ou (b) $j \geq m + 1$.
7. En vous appuyant sur les questions précédentes, montrer que pour toute instance I, $T_{LPT}(I) \leq r \cdot T_{opt}(I)$, où r est une constante dont vous donnerez la valeur.
8. Conclure quant à l'approximabilité de l'algorithme LPT.

2 MIN MAKESPAN – Projet de programmation

Dans ce mini-projet, la programmation se fera dans le langage de votre choix. *Assurez-vous que vos programmes compilent et fonctionnent sous Linux sur les machines du CIE.*

2.1 Présentation Générale

Le but de ce projet est d'étudier le problème MIN MAKESPAN (voir feuille de TD3, et voir aussi les transparents 34 à 42 disponibles dans la partie "Distanciel" de Madoc). Plus précisément, on vous demande d'implémenter plusieurs algorithmes répondant au problème, de les tester et de fournir les résultats de ces tests à l'utlisateur. Les trois algorithmes à implémenter sont :

1. l'algorithme List Scheduling Algorithm (LSA) ;
2. l'algorithme Largest Processing Time (LPT) ;
3. un algorithme de votre choix (MYALGO), que vous devez donc inventer, et que vous expliquerez en détail (en argumentant vos choix) dans le rapport.

2.2 Travail demandé

Proposer un programme convivial (=*user friendly* en anglais) qui, par l'intermédiaire d'un menu, permet à l'utilisateur (1) de rentrer la/les instance/s de son choix et (2) de voir les résultats obtenus par les trois algorithmes évoqués ci-dessus sur cette/ces instance/s.

Instances d'entrée. L'utilisateur doit pouvoir choisir entre trois modes de saisie des instances d'entrée :

1. (**Depuis un fichier**) L'instance est chargée à partir d'un fichier F donné par l'utilisateur. Ce fichier ne contient qu'une ligne, qui est une chaîne $m : n : d_1 : d_2 : d_3 \dots d_n$. Cela signifie que l'instance de MIN MAKESPAN est une instance à m machines et n jobs, dont les durées sont $d_1, d_2, d_3 \dots d_n$.
2. (**Au clavier**) L'instance est fournie au clavier par l'utilisateur, sous la forme d'une chaîne $m : n : d_1 : d_2 : d_3 \dots d_n$, dont la signification est la même que dans le cas précédent.
3. (**Génération aléatoire de plusieurs instances**) L'utilisateur fournit 5 entiers $m, n, k, \min$ et $\max$. Il faut alors générer k instances distinctes : pour chacune de ces k instances, on a m machines, n jobs, et les durées des jobs sont générées de manière aléatoire, à la condition que chaque durée soit comprise entre $\min$ et $\max$.

On appellera ces trois modes de création d'instance I_f (fichier), I_c (clavier) et I_g (génération).

Production des résultats. Pour chaque mode de création d'instances (I_f, I_c ou I_g), les trois algorithmes seront exécutés. Selon le mode de création des instances, les résultats seront fournis de façons différentes.

1. (**Instances de type I_f et I_c**) Les résultats seront affichés à l'écran de la façon suivante.

   ```
   Borne inférieure ''maximum'' =
   Borne inférieure ''moyenne'' =
   Résultat LSA =
   Résultat LPT =
   Résultat MyAlgo =
   ```

   (pour davantage d'informations sur les bornes inférieures "maximum" et "moyenne," voir le transparent 40 du PDF fourni sur Madoc, et aussi les Questions 2. et 3. de l'Exercice 3.2 de la feuille TD3)

   La valeur attendue en fin de chaque ligne `Résultat` est le temps de réalisation des tâches (*makespan*) obtenu par l'algorithme étudié sur l'instance considérée.

2. (**Instances de type I_g**) Les résultats seront écrits dans un fichier dont le nom sera fourni par l'utilisateur. Pour chaque instance, on donnera les résultats comme dans le 1. ci-dessus, avec une ligne claire de séparation (par exemple un ligne remplie de "===") entre chaque résultat.

   En fin de fichier, on rajoutera trois valeurs (une par algorithme). Ces trois valeurs seront calculées de la même façon, indiquée ci-dessous :

— pour chaque instance I, on prend le maximum entre la borne inférieure “maximum” et la borne inférieure “moyenne”. Appelons ce maximum M_I.
— pour cette même instance, on appelle $R_I = \frac{sol_A(I)}{M_I}$, où $sol_A(I)$ est la solution fournie par l'algorithme A sur l'instance I (A est donc soit LSA, soit LPT, soit MYALGO).
— on calcule la moyenne des R_I sur les k instances considérées.

C'est cette moyenne (qui n'est autre que le *ratio d'approximation moyen* de l'algorithme étudié sur l'ensemble des instances considérées), que l'on ajoutera en fin de fichier. En fin de fichier, on s'attend donc à lire ceci :

```
ratio d'approximation moyen LSA =
ratio d'approximation moyen LPT =
ratio d'approximation moyen MyAlgo =
```