TD 11 – PTAS et FPTAS

Exercice 1. *C'est toi la pétasse !*

Le but de cet exerice est d'obtenir un PTAS pour le problème d'ordonnancement suivant, où M est une constante fixée :

$PM||C_{max}$
Instance : n jobs $J_1, \ldots, J_n$ ayant chacun un temps d'execution p_j $(1 \leq j \leq n)$, à assigner sur M machines identiques fonctionnant en parallèle. Les jobs ne sont pas divisibles.
Mesure à minimiser : La date totale de fin d'execution C_{max} (*makespan*), c'est-à-dire le temps pendant lequel au moins une machine est encore en train de calculer.

Si cela n'a pas encore été vu en cours/TD, on admettra que ce problème est $\mathcal{NP}$-complet. On fixe $\varepsilon > 0$ et K une constante plus grande que M (on verra plus tard comment la choisir intelligemment en fonction de ε). Considérons l'algorithme suivant :

– Trier les jobs par temps d'execution décroissant de façon à avoir $p_1 \geq \ldots \geq p_n$.
– Obtenir par bruteforce une solution optimale S_K sur les K premiers jobs.
– Compléter la solution de manière gloutonne en choisissant la machine la moins chargée, pour j de $K+1$ à n.

1. Quel est le temps d'execution de l'agorithme ? Cela correspond-il à nos exigences ?
   Soit I une instance du problème, OPT la mesure de la solution optimale et $A(I)$ la mesure de la solution renvoyée par notre algorithme. Soit T la mesure de la solution optimale renvoyée par la deuxième étape de l'algorithme, sur les K premiers jobs. Le but est de montrer que $A(I) \leq (1+\varepsilon)OPT$.
2. Traiter le cas où $A(I) = T$.
   Supposons maintenant que $A(I) > T$. On note $p_{sum} = \Sigma_{j=1}^{n} p_j$.
3. Montrer que $p_{sum} \geq M \cdot A(I) - (M-1)p_{K+1}$.
4. En utilisant le principe des tiroirs, montrer que $OPT \geq \frac{K}{M} \cdot p_K$.
5. En déduire que
$$A(I) \leq \left(1 + \frac{M-1}{K}\right) \cdot OPT \, .$$
6. Choisissez K de manière à obtenir un PTAS.

Exercice 2. *Le retard coûte cher...*

Le but de cet exerice est d'obtenir un FPTAS pour le problème d'ordonnancement suivant :

$1||\Sigma T_j$ (Total Tardiness on a Single Machine)
Instance : n jobs $J_1, \ldots, J_n$ ayant chacun un temps d'execution p_j $(1 \leq j \leq n)$, ayant une deadline entière d_j, à assigner 1 machine. Les jobs ne sont pas divisibles.
Mesure à minimiser : Le retard accumulé $\Sigma_{j=1}^{n} T_j$ où $T_j = C_j - d_j$ et C_j est la date de fin de J_j (*completion time*).

Si cela n'a pas encore été vu en cours/TD, on admettra que ce problème est $\mathcal{NP}$-complet. On admettra également l'existence de l'algorithme de Lawler qui, basé sur le principe de programmation dynamique permet de résoudre ce problème en temps $\mathcal{O}(n^5 T_{EDD})$ où T_{EDD} est le retard maximal $\max T_j$. Cette valeur T_{EDD} est obtenu en temps polynomial grâce à la règle "du plus urgent", *Earliest-Due-Date rule*, c'est-à-dire que l'on classe les jobs par deadline croissante.

1. A quoi correspond le cas $T_{EDD} = 0$?
Soit I une instance du problème, OPT la mesure de la solution optimale. On suppose $T_{EDD} > 0$ et on pose :

$$Z = \frac{2\varepsilon}{n(n+3)} \cdot T_{EDD} \; .$$

Partie 1 : Construisons une instance simplifiée $I^{\#}$ à partir de I.
Plus exactement, commençons par construire une instance intermédiaire I' en arrondissant :
– chaque temps d'execution p_j au multiple de Z le plus proche **et inférieur** à p_j,
– et chaque deadline d_j au multiple de Z le plus proche **et supérieur** à d_j
Ensuite on construit $I^{\#}$ à partir de I' en divisant les temps d'execution et les deadlines par Z, autrement dit $p_j^{\#} = \lfloor p_j/Z \rfloor$ et $d_j^{\#} = \lceil d_j/Z \rceil$.

2. Montrer que le retard maximal $T_{EDD}^{\#}$ de $I^{\#}$ est inférieur ou égal à T_{EDD}/Z.

3. Comment obtenir alors en temps polynomial en n et en $1/\varepsilon$ la solution optimale à $I^{\#}$?
Partie 2 : Utilisons la solution de $I^{\#}$ pour obtenir une solution à I : ici, on garde le même ordre d'assignation des jobs.
Partie 3 : Prouvons que la solution est une $(1+\varepsilon)$-approximation.
Quitte à re-numéroter les jobs, on suppose pour simplifier les notations que l'ordre d'assignation est $J_1, \dots, J_n$. On note C_j et T_j la date de fin et le retard pour le job J_j dans la solution de I, et $C_j^{\#}$ et $T_j^{\#}$ les valeurs correspondantes dans la solution de $I^{\#}$.

4. Montrer que $C_j = \Sigma_{i=1}^{j} p_j \leq Z \cdot C_j^{\#} + jZ$.

5. En déduire que
$$\Sigma_{j=1}^{n} T_j \leq (1+\varepsilon) \cdot OPT \; .$$

6. Conclure.