TD 3 – J'ai faim

Exercice 1. *Suivez le maître*

Appliquer le *Master Theorem* sur les cas suivants :

1. $T(n) = 9T(n/3) + n$;
2. $T(n) = T(2n/3) + 1$;
3. $T(n) = 3T(n/4) + n\log n$;
4. $T(n) = 2T(n/2) + n\log n$;
5. $T(n) = 2T(n/2) + n^3$;
6. $T(n) = T(9n/10) + n$;
7. $T(n) = 7T(n/3) + n^2$;
8. $T(n) = T(n-1) + n$;
9. $T(n) = T(\sqrt{n}) + 1$.

Exercice 2. *Couverture par intervalles*

Étant donné un ensemble $\{x_1, \ldots, x_n\}$ de n points sur une droite, décrire un algorithme qui détermine le plus petit ensemble d'intervalles fermés de longueur 1 qui contient tous les points donnés. Prouver la correction de votre algorithme et donner sa complexité.

Exercice 3. *Limitons la température*

Le but de cet exercice est d'illustrer la théorie des matroïdes. Nous cherchons ici à contruire un matroïde pondéré, à écrire l'algorithme glouton correspondant et à prouver qu'il renvoie la réponse optimale.

Soit $G = (V, E)$ un graphe dirigé où chaque arête $e \in E$ est munie d'une pondération entière $w(e)$, et une fonction de contrainte $f : V \to \mathbb{N}$. Le but est de trouver un sous-ensemble d'arêtes de poids maximal tel que le degré sortant de chaque noeud u est au plus $f(u)$.

1. Définir des ensembles indépendants et prouver qu'ils forment un matroïde.
2. Quel est le cardinal d'un ensemble indépendant maximal ?
3. Quel est l'algorithme glouton associé ? Pourquoi renvoie-t-il la solution optimale ? Quelle est sa complexité ?

Exercice 4. *Utilisation de la mémoire*

On souhaite enregistrer sur une mémoire de taille L un groupe de fichiers $P = (P_1, \ldots, P_n)$. Chaque fichier P_i nécessite une place a_i. Supposons que $\sum a_i > L$: on ne peut pas enregistrer tous les fichiers. Il s'agit donc de choisir le sous ensemble Q des fichiers à enregistrer.

On pourrait souhaiter le sous-ensemble qui contient le plus grand nombre de fichiers. Un algorithme glouton pour ce problème pourrait par exemple ranger les fichiers par ordre croissant des a_i.

Supposons que les P_i soient ordonnés par taille ($a_1 \leq \ldots \leq a_n$).

1. Écrivez un algorithme (en pseudo-code) pour la stratégie présentée ci-dessus. Cet algorithme doit renvoyer un tableau booléen S tel que $S[i] = 1$ si P_i est dans Q et $S[i] = 0$ sinon. Quelle est sa complexité en nombre de comparaisons et en nombre d'opérations arithmétiques ?
2. Montrer que cette stratégie donne toujours un sous-ensemble Q maximal tel que $\sum_{P_i \in Q} a_i \leq L$.
3. Soit Q le sous-ensemble obtenu. À quel point le quotient d'utilisation $(\sum_{P_i \in Q} a_i)/L$ peut-il être petit ?

Supposons maintenant que l'on souhaite enregistrer le sous-ensemble Q de P qui maximise ce quotient d'utilisation, c'est-à-dire celui qui remplit le plus de disque. Une approche *gloutonne* consisterait à considérer les fichiers dans l'ordre décroissant des a_i et, s'il reste assez d'espace pour P_i, on l'ajoute à Q.

4. On suppose toujours les P_i ordonnés par taille croissante. Écrivez un algorithme pour cette nouvelle stratégie.
5. Montrer que cette nouvelle stratégie ne donne pas nécessairement un sous-ensemble qui maximise le quotient d'utilisation. À quel point ce quotient peut-il être petit ? Prouvez-le.