Examen MIM1 – Algorithmique

Yves Robert

Mercredi 9 Janvier 2002, 9h–12h

Les notes de cours sont autorisées. Les trois parties sont indépendantes.

1 Algorithmique

1.1 Amusettes pour s'échauffer

Amusette 1 Soit n entiers $a_1, \ldots, a_n$ avec n pair. On cherche à les partitionner en $n/2$ paires telles que le maximum des sommes des deux éléments de chaque paire soit minimal. Donner la solution optimale avec une petite preuve de correction.

Amusette 2 Soit un tableau de n entiers $A[1..n]$ tel que deux éléments consécutifs différent au plus d'une unité : $|A[i+1] - A[i]| \leq 1$ pour $1 \leq i < n$. Donner un algorithme efficace, avec une petite preuve de correction, pour le problème suivant : étant donné un entier x, existe-t-il un indice i tel que $A[i] = x$.

1.2 Plus courte super-suite

On a vu en cours le calcul d'une plus longue sous-suite commune à deux chaînes A et B. On s'intéresse au calcul d'une plus courte super-suite de A et B, définie comme une suite S dont A et B sont des sous-suites et de longueur minimale.

1. Déterminer S pour $A = ababab aab$ et $B = aabbbbaab$.
2. Proposer un algorithme pour déterminer une plus courte super-suite de A et B et donner sa complexité. Attention, il faut justifier soigneusement la correction.

2 NP-complétude

Montrer que les trois problèmes suivants sont NP-complets :

Chevaliers de la table ronde Etant donnés n chevaliers, et connaissant toutes les paires de féroces ennemis parmi eux, est-il possible de les placer autour d'une table circulaire de telle sorte qu'aucune paire de féroces ennemis ne soit côte à côte ?

Deux cliques Etant donné un graphe $G = (V, E)$, et un entier $K > 0$, existe-t-il deux cliques disjointes de taille K dans G ? On rappelle qu'une clique est un sous-ensemble de sommets tous adjacents deux à deux.

Chemin avec paires interdites Soient $G = (V, E)$ un graphe orienté (on distingue l'arc (u,v) de l'arc (v,u)), deux sommets de ce graphe $s, t \in V$ et une liste $C = \{(a_1, b_1), \ldots, (a_n, b_n)\}$ de paires de sommets de G. Existe-t-il un chemin orienté de s vers t dans G qui contient au plus un sommet de chaque paire de la liste C ?

3 Approximation

On s'intéresse au problème *Minimum Bin Packing* : ranger des objets dans des boîtes de capacité donnée, en minimisant le nombre de boîtes. Formellement :

Instance : Un ensemble fini I de n nombres rationnels $\{a_1, a_2, \ldots, a_n\}$ avec $0 < a_i \leq 1$ pour $1 \leq i \leq n$.
Solution : Une partition $\{B_1, B_2, \ldots, B_k\}$ de I telle que $\sum_{a_i \in B_j} a_i \leq 1$ pour tout $1 \leq j \leq k$ (les boîtes ont une capacité unitaire).
Mesure : Le cardinal k de la partition.

Préliminaire : Formuler le problème de décision associé et démontrer brièvement qu'il est NP-complet.

3.1 Premier glouton : NextFit

Dans l'algorithme glouton NextFit, on examine les éléments a_i dans un ordre arbitraire. Le premier élément examiné est placé dans la boîte B_1. Pour placer un nouvel élément, soit il tient dans la dernière boîte utilisée, auquel cas on l'y met, soit il n'y tient pas, et on le range dans une nouvelle boîte.

1. Montrer que NextFit est toujours à un facteur deux de l'optimal, i.e.

$$k(NextFit) \leq 2 \times k(Opt).$$

2. Soit l'instance du problème avec $2n$ entiers $a_{2i-1} = \frac{1}{2}$et $a_{2i} = \frac{1}{2n}$ pour $1 \leq i \leq n$. Comparer NextFit et la solution optimale. Conclusion ?

3.2 Deuxième glouton : FirstFit

Dans l'algorithme glouton FirstFit, on place chaque élément dans la première boîte qui peut le contenir, sinon on crée une nouvelle boîte pour lui. En d'autres termes :
- créer une première boîte B_1 pour ranger a_1
- pour tout i de 2 à n dans l'ordre, soit $B_1, \ldots, B_r$ les boîtes déjà utilisées. Ranger a_i dans la première boîte B_j qui peut le contenir ; si ce n'est pas possible, créer une nouvelle boîte B_{r+1} et ranger a_i dedans.

1. Donner un exemple où FirstFit est meilleur que NextFit.
2. Donner un exemple où FirstFit n'est pas optimal.
3. On considère la variante DecreasingFirstFit, où on commence par trier les les éléments a_i dans l'ordre décroissant avant d'appliquer FirstFit.
   (a) Donner un exemple pour lequel DecreasingFirstFit est meilleur que FirstFit
   (b) Donner un exemple pour lequel FirstFit est meilleur que DecreasingFirstFit
   (c) (difficile) Montrer que DecreasingFirstFit est toujours à un facteur un et demi de l'optimal, à une unité près, i.e.
$$k(DecreasingFirstFit) \leq \frac{3}{2} \times k(Opt) + 1.$$