Université Lille1 — Sciences et Technologies
UFR IEEA

Master d'Informatique-S1

11 juin 2014
Documents autorisés

AAC

Les algorithmes seront écrits en pseudo-langage ou dans un langage de votre choix (C, Java, Caml). La clarté de vos réponses et de votre code sera prise en compte. Le barème indiqué est le barème "minimal".

Exercice 1 : Quelques questions de base - *4.5 pts* -

Q 1. *A classer - 1.5 pts -*

On veut trier des fonctions par "ordre" asymptotique de grandeur : si $f = o(g)$, f sera "avant" g pour cet ordre. Trier les fonctions suivantes :

$$n^3 \quad 4^n \quad \lg\lg n \quad \lg n \quad (\sqrt{2})^n \quad n^2 \quad n\sqrt{n} \quad n^n \quad 2^n + n^5$$

Q 2. *Le plus rapide ? - 1.5 pts -*

On suppose que deux programmes A et B ont une complexité respective exacte de $T_A(n) = c_A n \log_{10} n$, $T_B(n) = c_B n$ pour traiter une donnée de taille n (quelque soit la donnée de taille n). Dans un test, A a mis 100 millisecondes pour traiter une donnée de taille 10^4, B a mis 200 millisecondes.

Pour une donnée de taille 10^8 quel sera le programme le plus rapide ? Même question pour une donnée de taille 10^{10}. Justifiez.

Q 3. *La classe NP - 1.5 pts -*

Soient deux problèmes de décision, X et Y. On sait que X se réduit polynomialement en Y.

Si Y est NP-complet, que peut-on en déduire pour X ? Pourquoi ?

Si X est NP-complet, que peut-on en déduire pour Y ? Pourquoi ?

Exercice 2 : Dépendance au chocolat - *9 pts* -

Vos chargés de TD ont toujours une addiction pour le chocolat. Ils ont maintenant une tablette de chocolat, c'est-à-dire une grille de m colonnes et n lignes, dans laquelle un certain nombre -k- de carrés sont marqués d'un signe spécial. Pour fixer les idées, le carré en haut à gauche est le carré de coordonnées $(0,0)$, tandis que le carré en bas à droite est le carré de coordonnées $(m-1, n-1)$. Une table S de booléens de m colonnes et n lignes indique quels carrés sont marqués du symble spécial : S[i][j]= true si et seulement si le carré (i,j) est marqué du signe spécial.

La donnée du problème sera donc, m le nombre de colonnes, n, le nombre de lignes et une table S de booléens indiquant les carrés spéciaux.

On veut découper tous les carrés spéciaux - c.à.d. à la fin, tous les carrés spéciaux doivent être séparés individuellement. Une découpe élémentaire est : prendre une tablette et la casser en deux (horizontalement ou verticalement). L'objectif est de minimiser le nombre de découpes élémentaires. Au cours de la découpe, on manipulera donc un certain nombre de tablettes. Une tablette sera identifiée par les coordonnées du carré en haut à gauche (i,j) et de celui en haut à droite (k,l) dans la tablette initiale.

Q 1. *Exemples - 1 pt -*

Pour l'exemple A ci-dessous, la découpe nécessite 8 opérations élémentaires, par exemple comme proposé sur la figure. Combien de découpes élementaires sont nécessaires pour le cas de la figure B ? Donnez une séquence optimale de découpes et justifiez brièvement son optimalité.

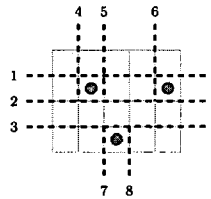


Figure A

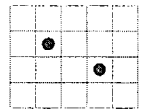


Figure B

Q 2. *Meilleur des cas ? - 1 pt -*

Montrer que si il y a k carrés spéciaux, au minimum $k-1$ découpes élémentaires sont nécessaires. Y a-t-il un cas où cela est suffisant ?

Q 3. *Pire des cas ? - 1 pt -*

Justifier qu'au maximum $4k$ découpes élémentaires sont nécessaires si il y a k carrés spéciaux. Y a-t-il un cas où $4k$ découpes sont vraiment nécessaires ?

Q 4. On va maintenant concevoir un algorithme pour calculer le nombre minimum de découpes élémentaires nécessaires.

Q 4.1. *Préliminaire - 2 pts -*

On veut d'abord pouvoir tester facilement si une tablette de coin en haut à gauche (i,j) et en haut à droite (k,l) contient au moins un carré spécial. Pour cela, on crée une table *Pres*, avec pour objectif Pres[i][j][k][l]=true Ssi la tablette $(i,j),(k,l)$ contient au moins un carré spécial.

Proposer un algorithme en $O(n^2 * m^2)$ pour remplir cette table.

Q 4.2. - *4 pts* - Proposer un algorithme en $O(n^2 * m^2 * (m+n))$ qui détermine le nombre minimum de découpes élémentaires nécessaires pour découper tous les carrés spéciaux.

Bonus : En déduire un algorithme qui sort - sous une forme laissée à votre choix- une suite minimale de découpes élémentaires.

Exercice 3 : Variations autour de l'indépendance - *5 pts* -

Rappel : dans un graphe, un *ensemble indépendant* est un ensemble de sommets qui ne sont **pas** reliés entre eux - aucune paire n'est reliée. Le problème Independent Set consiste à déterminer s'il existe dans un graphe donné un ensemble indépendant de cardinal supérieur ou égal à un entier k donné. Le problème de décision Independent Set est donc défini par :

Entrée :
$G = (S, A)$ - -un graphe
k - -un entier
Sortie : Oui, si il existe un sous-ensemble indépendant de cardinal $\geq k$, i.e. une partie I de S de cardinal $\geq k$ telle qu'aucun arc ne relie deux sommets de I ($(I \times I) \cap A = \emptyset$).

Independent Set est connu NP-complet.

Soient les trois problèmes suivants :

100-Ind
Entrée : $G = (S, A)$ –un graphe
Sortie : Oui, si il existe un sous-ensemble indépendant de cardinal égal à 100.

Double-Ind
Entrée : $G = (S, A)$ –un graphe
k - -un entier
Sortie : Oui, si il existe deux sous-ensembles distincts indépendants de cardinal supérieur ou égal à k.

Presque-Tous-Ind
Entrée : $G = (S, A)$ –un graphe, $|S| \geq 1$ ($|S|$ dénote le cardinal de S).

Sortie : Oui, si il existe un sous-ensemble indépendant de cardinal égal à $|S| - 1$.

Q 1. - *4 pts* -
Parmi ces trois problèmes, 100-Ind, Double-Ind, Presque-Tous-Ind, quel est le problème NP-dur, quels sont les problèmes P ? Justifier.

Q 2. - *1 pt* -
Les trois problèmes sont-ils NP ? Justifier.

Exercice 4 : La ligne des toits - *6.5 pts* -

On cherche à dessiner en 2D la ligne des toits d'un ensemble d'immeubles rectangulaires vus par une personne qui les regarde de face. Plus précisément, on a n rectangles alignés sur l'axe des abscisses dans le premier quadrant, le rectangle $(G_i, 0), (G_i, H_i), (D_i, H_i), (D_i, 0)$ étant donné par le triplet (G_i, H_i, D_i) $(0 \leq G_i < D_i, 0 < H_i)$. L'entrée du problème est donc :
n le nombre d'immeubles
Im la liste des triplets (G_i, H_i, D_i).

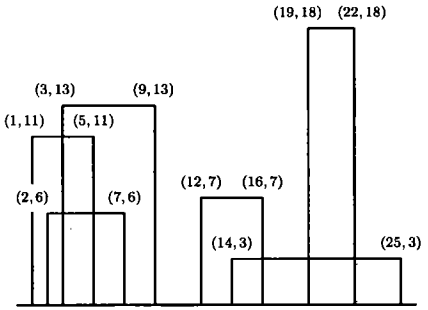


Figure A

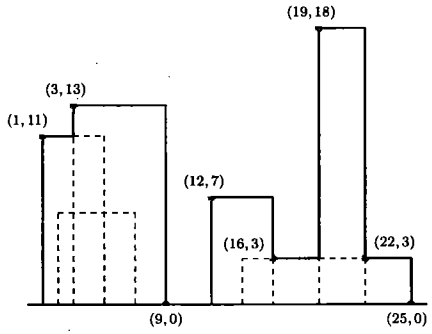


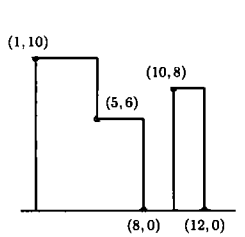
Figure B

La sortie sera une suite ordonnée par abscisses croissantes des points qui permettent de tracer la ligne des toits. Par exemple, pour 6 immeubles donnés par la liste (1,11,5), (2,6,7), (3,13,9), (12,7,16), (14,3,25), (19,18,22) (voir figure A), la sortie sera la liste (1,11), (3,13), (9,0), (12,7), (16,3), (19,18), (22,3), (25,0) (voir figure B).

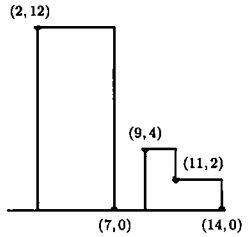
Remarque : une liste de points $(x_1, y_1), (x_2, y_2), \ldots, (x_p, y_p)$ triée par abscisses croissantes correspond à la ligne tracée en reliant dans l'ordre les points suivants :
$(x_1, 0), (x_1, y_1), (x_2, y_1), (x_2, y_2), (x_3, y_2) \ldots (x_i, y_i), (x_{i+1}, y_i), (x_{i+1}, y_{i+1}) \ldots (x_p, y_{p-1}), (x_p, y_p)$
On pourra supposer que pour une telle liste, $y_1 > 0$, $y_p = 0$ et que deux points consécutifs ont des ordonnées différentes.

On va appliquer une méthode de type diviser pour régner pour construire la ligne des toits des immeubles. Pour cela, on va d'abord réaliser la fusion de deux lignes de toits.

Q 1. Fusion - *4.5 pts* Le problème est donc défini par :
Entrée : deux lignes de toits données par la suite ordonnée par abscisses croissantes des points qui permettent de les tracer
Sortie : la ligne de toits obtenue par fusion
Par exemple, soit la première ligne donnée par (1,10), (5,6), (8,0), (10,8), (12,0) et la seconde par (2,12), (7,0), (9,4), (11,2), (14,0), comme dans les figures ci-dessous :

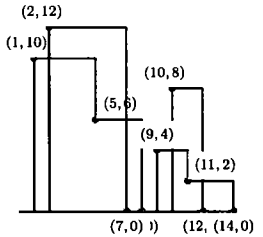


Ligne 1

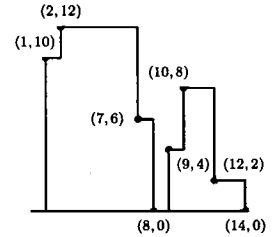


Ligne 2

La fusion des lignes des toits est (1,10), (2,12), (7,6), (8,0), (9,4), (10,8), (12,2), (14,0) :



Lignes 1 et 2



Fusion

Q 1.1. Si la première ligne de toits est donnée par (1,4), (6,0), (9,7), (12,0) et la seconde par (3,1), (5,5), (10,2), (15,0), quelle est la ligne de toits obtenue par fusion ?
Q 1.2. Proposez un algorithme en $O(n)$ pour fusionner deux listes de toits, n étant le max des deux longueurs des lignes de toits. Vous pouvez supposer que les listes et les points sont représentés comme vous le souhaitez.

Q 2. Diviser pour régner - *2 pts*
Proposez un algorithme en $O(n \log n)$ de type "Diviser pour régner" pour résoudre le problème initial, i.e. sortir à partir de la liste des immeubles représentés par leurs triplets, la ligne de toits correspondante. Vous pouvez supposer que les listes, points et triplets sont implémentés comme vous le souhaitez. Vous pouvez supposer avoir répondu à la question précédente et donc disposer d'un algorithme en $O(n)$ pour la fusion de deux lignes de toits.