Algorithmique et Programmation
TD n° 9 : Algèbre linéaire

École normale supérieure - Département d'informatique
algoL3@di.ens.fr

2014-2015

L'algorithme de Strassen ne fonctionne que sur des anneaux et non sur le quasi-anneau des booléens $(\{0,1\}, \vee, \wedge, 0, 1)$ puisque 1 n'a pas d'inverse pour $\vee$. Les deux exercices suivants proposent des algorithmes de calcul du produit de matrices booléennes.

Exercice 1 MÉTHODE DES « QUATRE RUSSES »

Soient A et B deux matrices booléennes $n \times n$. Nous supposons que $\log_2(n)$ divise n.

1. Montrer que nous pouvons écrire $AB = \sum_{i=1}^{n/\log_2(n)} A_i B_i$ où chaque A_i est une matrice booléenne de taille $n \times \log_2(n)$ et chaque B_i est une matrice booléenne de taille $\log_2(n) \times n$.
2. Montrer que l'algorithme naïf pour calculer chaque matrice $A_i B_i$ a complexité $O(n^2 \log_2(n))$.
3. Proposer un algorithme, qui, après un certain calcul préliminaire (indépendant de A et B), peut calculer chaque matrice $A_i B_i$ en $O(n^2)$ opérations.

Cette méthode de multiplication (dite des « 4 russes »et due à Arlazarov, Dinic, Kronrod, et Faradzev) permet donc de calculer le produit de deux matrices booléennes en $O(n^3/\log_2(n))$.

Exercice 2 MÉTHODE PROBABILISTE DE SHAMIR

Dans cet exercice, l'idée (due à Shamir) est de travailler sur un anneau où on peut appliquer l'algorithme de Strassen, à savoir l'anneau $R = (\{0,1\}, \oplus, \wedge, 0, 1)$ où $\oplus$ représente le *ou exclusif* (c'est-à-dire la somme modulo 2), puis de relier les resultats pour qu'ils s'appliquent sur le quasi-anneau des booléens.

Soient $A = (a_{i,j})$ et $B = (b_{i,j})$ deux matrices booléenes $n \times n$ et soit $C = (c_{i,j})$ leur produit dans le quasi-anneau des booléens. À partir de A, nous fabriquons une matrice aléatoire $A' = (a'_{i,j})$ en utilisant la procédure probabiliste suivante : si $a_{i,j} = 0$, alors on pose $a'_{i,j} = 0$; si $a_{i,j} = 1$, alors $a'_{i,j}$ est tiré uniformément aléatoirement dans $\{0,1\}$, les choix aléatoires pour chaque entrée étant indépendants. Soit $C' = (c'_{i,j})$ le produit $A'B$ dans l'anneau R.

1. Montrer que si $c_{i,j} = 0$, alors $c'_{i,j} = 0$.
2. Montrer que si $c_{i,j} = 1$, alors $c'_{i,j} = 1$ avec probabilité $1/2$.
3. Soit $M(n)$ le temps nécessaire pour multiplier deux matrices dans R par l'algorithme de Strassen. Donner un algorithme probabiliste en $O(M(n)\log(n))$ qui calcule le produit de deux matrices $n \times n$ dans le quasi-anneau booléen avec une probabilité d'au moins $1 - n^{-10}$. Les seules opérations autorisées sur les composantes des matrices sont $\vee$, $\wedge$ et $\oplus$.

Exercice 3 PAVAGE

On considère l'algorithme suivant de pavage d'un polyomino sans trou donné par son mot de contour, un élément de $\{N, S, E, O\}^*$.

1. Donner hauteur 0 à un point du contour arbitraire.
2. Parcourir le bord dans le sens direct en donnant une hauteur à chaque point en fonction de la hauteur h du point précédent et de la couleur (noire ou blanche) de la case longée à gauche : $h+1$ si noire, $h-1$ si blanche.
3. Répéter :
   (a) Trouver un point du bord de hauteur maximum
   (b) Placer un domino le long du bord de façon que ce point ne soit plus sur le nouveau bord. Mettre à jour bord et hauteurs

jusqu'à ce que le polyomino soit pavé ou qu'il y ait une impossibilité.

1. Donner un exemple de polyomino tel que la hauteur ne soit pas bien définie dans l'étape 2.
2. Montrer que la hauteur est bien définie ssi P contient autant de cases blanches que de cases noires.
3. Montrer que si P est pavable alors, étant donné un pavage de P, on peut étendre la définition de la hauteur'a tous les point de l'intérieur de P.
4. Montrer que si P est pavable, alors il existe un pavage tel que le point de hauteur maximum soit au bord.
5. Montrer que si P est pavable, alors l'algorithme donne un pavage correct.
6. Expliquer comment implémenter cet algorithme en temps O(surface).

Exercice 4 COMPOSITION DE POLYNÔMES

Soit un anneau commutatif unitaire et soient $g, h \in A[x]$ deux polynômes tels que $\deg g, \deg h < \deg f = n$. Le but de l'exercice est de proposer un exercice qui calcule $g(h) \bmod x^n$ dans $A[x]$.

1. Proposer un algorithme naïf de complexité $O(nM(n))$ où $M(n)$ est la complexité arithmétique dans A du calcul du produit de deux polynômes de degré au plus n.
2. En utilisant une approche de type « pas-de-bébé, pas-de-géant », proposer un algorithme de complexité $O(n^{1,686}) = O(n^{(\omega+1)/2})$ (en supposant que $M(n) \in O(n \log n \log\log n)$ et que le produit de deux matrices carrées de taille n d'éléments de A a une complexité $O(n^\omega)$ avec $\omega < 2.3727$).