Algorithmique et Programmation
TD n°10 : Algèbre linéaire

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

2013-2014

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 et nous partitionnons A en sous-matrices A_i (avec $1 \leq i \leq n/\log_2(n)$) de taille $n \times \log_2(n)$ et B en sous-matrices B_j (avec $1 \leq j \leq n/\log_2(n)$) de taille $\log_2(n) \times n$.

1. Montrer que nous pouvons écrire $AB = \sum_{i=1}^{n/\log_2(n)} A_i B_i$ où chaque produit $A_i B_i$ est une matrice $n \times n$.
2. Montrer que l'algorithme naturel pour calculer chaque matrice $A_i B_i$ demande $O(n^2 \log_2(n))$ opérations.
3. Proposer un algorithme calculant chaque matrice $A_i B_i$ en $O(n^2)$ opérations.

Cette méthode de multiplication (dite des « 4 russes ») 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 et relier les resultats pour qu'ils s'appliquent sur l'anneau des booléens. Pour ce faire, il propose de travailler avec l'anneau $R = (\{0,1\}, \oplus, \wedge, 0, 1)$ où $\oplus$ représente le *ou exclusif*.

Soient $A = (a_{i,j})$ et B 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 $A' = (a'_{i,j})$ en utilisant la procédure probabiliste suivante :

- si $a_{i,j} = 0$, alors $a'_{i,j} = 0$;
- si $a_{i,j} = 1$, alors $a'_{i,j}$ est tiré uniformément aléatoirement dans $\{0,1\}$. Les choix aléatoires pour chaque entrée étant indépendants.

1. Soit $C' = (c'_{i,j})$ le produit $A'B$ dans l'anneau R. Montrer que si $c_{i,j} = 0$, alors $c'_{i,j} = 0$ et que si $c_{i,j} = 1$, alors $c'_{i,j} = 1$ avec probabilité $1/2$.
2. Soit $M(n)$ le temps nécessaire pour multiplier deux matrices dans R. 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 $1 - n^{-k}$ pour une constante $k > 0$ quelconque. Les seules opérations autorisées sur les composantes des matrices sont $\vee$, $\wedge$ et $\oplus$.

Exercice 3 PSEUDO-INVERSE

Soient $m, n \geq 1$ deux entiers. Nous appelons *pseudo-inverse* de $A \in \mathcal{M}_{m,n}$ toute matrice $G \in \mathcal{M}_{m,n}$ vérifiant $AGA = A$, $GAG = G$, ${}^t(AG) = AG$, ${}^t(GA) = GA$. Le but de cet exercice est de montrer que toute matrice admet un pseudo-inverse que nous notons A^g et de donner un un algorithme pour le calculer.

1. Montrer qu'une matrice A possède au plus un pseudo-inverse.
2. Soit $A \in \mathcal{M}_{m,n}$ une matrice de rang n. Montrer que $({}^tAA)^{-1}{}^tA$ est le pseudo-inverse de A.
3. Soit $A \in \mathcal{M}_{m,n}$ une matrice de rang r et supposons que A se mette sous la forme (U, UK) où $U \in \mathcal{M}_{m,r}$ est de rang r et $K \in \mathcal{M}_{r,n-r}$.
   Montrer que $\begin{pmatrix} WU^g \\ {}^tKWU^g \end{pmatrix}$ où $W = (I_r + K{}^tK)^{-1}$ est le pseudo-inverse de A.

4. En déduire que toute matrice possède un pseudo-inverse et donner un algorithme pour la calculer.

Exercice 4 COMPOSITION DE POLYNÔMES

Soit un anneau commutatif unitaire et soient $f, g, h \in A[x]$ trois polynômes tels que $\deg f, \deg h < \deg f = n$ avec $f \neq 0$ unitaire. Le but de l'exercice est de proposer un exercice qui calcule $g(h) \bmod f$ 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 $O(n^{1,686})$ (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 $\omega < 2.3727$).