PROG2: Inversion de matrices

Rémi Hutin Rémy Sun

26 février 2016

Abstract

1 Implémentation de matrices

1.1 Extraction de sous-matrice

Il est nécessaire d'extraire une sous-matrice en retirant les lignes d'indice a et les colonnes d'indice b.

Ceci est fait en créant une matrice carrée de taille $n-1$, qu'on remplit par parcours de cette matrice en tirant parti du fait que l'expression (i ¿= a) renvoie 1 si $i \leq a$ ce qui permet d'engendrer un décalage de ligne/colonne quand nécessaire.

1.2 Temps de calculs

2 Optimisation: implémentation des sous-matrices

Jusqu'à maintenant, chaque matrice possédait un champ contents correspondant à un double vecteur de valeurs.

Cela veut notament dire que lorsqu'on veut extraire une sous matrice, il fallait pour créer la nouvelle sous-matrice recréer presque l'integralité de ce champ contents.

Ainsi, le coût en mémoire devenait très vite très important comme montré à la section précédente.

Nous allons maintenant prédsenter une façon de ne pas recréer un champ de valeurs.

2.1 Définition de matrices par vecteurs de bits d'activité

Nous passons contents en attribut static de la classe Matrix: toutes les matrics partagent le même champ contents.

L'idée est de faire de contents un triple vecteur telle que la premiére coordonnée repére l'index de la matrice considérée.

Ainsi, à chaque fois qu'une nouvelle matrice (à part entiére) est crée, on ajoute son double vecteur de valeurs au vecteur contents et on passe en champ son index dans contents.

De plus, pour chaque matrice, on crée deux vecteurs lines et rows qui repérent quels lignes et colonnes de la matrice repérée par index dans contents sont "actives".

A l'origine, toutes les lignes et colonnes sont actives.

2.2 Redéfinition des méthodes de base

La plus grosse difficulté posé par cette construction est que l'accés à la valeur (i,j) de la matrice n'est plus direct: certaines lignes et colonnes sont "désactivées". Pour remédier à cela, on crée deux méthodes get_real_i(int i) et get_real_j(j) qui font un parcours du vecteur de bits correspondant de maniére à trouver le "vrai" i/j.

Cela se fait en déclarant un compteur count, puis en faisant une boucle sur une variable k telle qu'à chaque itération count augmente ssi le bit d'index k est positif. Quand count ¿ i/j, on aréte la boucle et on récupére le k courant qui correspond à real_i/j + 1.

A partir de là, il devient très simple de redéfinir toutes le méthodes de base de Matrix en prenant soin de se rappeller que contents est maintenant un triple vecteur.

2.3 Création de sous-matrices

La création de sous-matrice devient très simple: on crée une copie de la matrice en question, puis on modifie les vecteurs rows et lines pour refléter le changement dans la matrice conssidéré.

2.4 Comparaisons avec l'ancienne méthode

2.5 Insuffisances

Cette méthode présente cependant un gros inconvénient: les sous-matrices ainsi déclarées ont un contenu directement liée à celui des matrices mères. Si on modifie leur contenu, on modifie aussi celui de la matrice mère. et vice-versa.

Cela ne pose pas de probléme dans le projet proposé puisque l'application qui en est faite agit en place sur le probléme et ne modifie pas les valeurs des matrices.