Xiang LI
Cédric BARRETEAU

Rapport NF16

Liste des structures et fonctions

Nous n'avons implémenté ni de nouvelles structures ni de nouvelles fonctions par rapport à celles proposées par l'énoncé. Cependant, nous avons parfois déclaré un élément “prédécesseur” dans certaines fonctions, lorsque cela était préférable. Nous avons préféré ce choix plutôt que de rendre la structure element plus lourde alors que cela n'est pas nécessaire dans la majorité des cas.

Analyse de complexité

Dans cette partie, nous allons analyser la complexité temporelle dans le pire cas des différentes fonctions de notre programme.

constructMat :

Cette fonction ne fait pas appel à d'autres fonctions, la détermination de sa complexité est donc simple. Dans le pire des cas, on rentre dans les deux boucles imbriquées, avec N et M itérations au maximum respectivement, ainsi que dans la boucle while, elle aussi imbriquée. Celle-ci peut avoir jusqu'à M itérations, car elle navigue à travers les colonnes de la matrice.

Il est à noter que cette fonction créé les matrices dans l'ordre, dans le sens que l'ordre d'occurrence des éléments est croissant en termes de numéros de colonnes.

On a donc une complexité en $O(M^2*N)$.

lireFichier :

Cette fonction comporte deux boucles similaires, à la différence près que la deuxième comporte en plus un appel à putValue. La complexité est la somme de ces deux boucles (notamment), mais la première étant d'ordre inférieur puisque la complexité de putValue est d'ordre supérieur à 1, on la négligera.

La première boucle est en fait une paire de boucles imbriquées, parcourant les lignes et les colonnes. À l'intérieur de celles-ci, on trouve un appel à putValue, qui est de complexité O(M).

On a donc une complexité en $O(M^2*N)$.

Xiang LI
Cédric BARRETEAU

afficherMat :

Puisque la construction de matrices et l'ajout de valeurs dans celles-ci se fait dans l'ordre, selon la définition précédente, on peut afficher les matrices avec simplement une boucle sur les colonnes imbriquée dans une autre sur les lignes.

On a donc une complexité en O(N*M).

getValue :

Puisque la ligne est donnée par l'utilisateur, on ne doit parcourir que les colonnes d'une seule ligne afin d'arriver à la valeur cherchée.

On a donc une complexité en O(M).

putValue :

Cette fonction comporte beaucoup de cas différents. La complexité dans le pire cas est le maximum des complexités des différents cas, en l'occurence si l'on doit parcourir la ligne pour trouver l'endroit à rajouter la nouvelle valeur. Dans ce cas, puisqu'on a déjà la ligne, il ne faut qu'une boucle parcourant les colonnes, qui a donc une complexité de l'ordre de M, puisqu'il n'y a au maximum que M colonnes.

On a donc une complexité en O(M).

Il est à noter que cette fonction conserve l'ordre établit par constructMat, et justifie donc la complexité d'afficheMat.

addMat :

Cette fonction est constituée de deux boucles imbriquées sur les lignes et les colonnes, au sein desquelles on a deux appels à getValue, et un à putValue. La complexité est donc de O(M*N*C(max(putValue,getValue)).

On a donc une complexité en $O(M^2*N)$.

nombreOctetsGagnes :

De même que pour addMat, on trouve ici un getValue dans deux boucles imbriquées sur les lignes et les colonnes.

On a donc une complexité en $O(M^2*N)$.

Nous avons justifié le choix de baisser la complexité de l'affichage au coût de celui de construction par le fait que l'affichage se fait le plus souvent que la construction de matrice.

Xiang LI
Cédric BARRETEAU

Utilisation du programme

Certains choix, non précisés dans le sujet, ont été fait et doivent être pris en compte par l'utilisateur. Ces choix seront spécifiés dans cette partie.

Les matrices stockées dans les fichiers textes doivent comporter des points virgules entre les colonnes, et des retours à la ligne uniquement (pas de point-virgule) entre les lignes.

Une seule matrice est gardée en mémoire à la fois : c'est celle qui sera affichable par la l'option 3 du menu.

Un tableau à deux dimensions est généré automatiquement, et peut être transformé en matrice creuse par l'utilisateur grâce à l'option 1 du menu.

La matrice creuse peut aussi être obtenue par l'option 2 du menu, qui utilise un fichier.

Enfin, il est possible de modifier la matrice en mémoire avec l'option 4 du menu. L'utilisateur entre les valeurs d'une matrice, qui sera sommée à celle en mémoire. La somme écrase la matrice gardée en mémoire.

Il est possible d'obtenir un retour négatif en utilisant l'option 5 du menu. En effet, dans certains cas, par exemple pour une matrice pleine de 1, la forme de tableau est plus performante en terme de mémoire. Dans ce cas, la valeur absolue de la valeur renvoyée représente le gain d'espace de la forme de tableau par rapport à celle proposée dans ce TP.