Méthode de Ranking - Compte Rendu

Nassime Kabache - Sofiane Atmani - Jean Destribois

Mai 2020

1 Introduction

Le projet effectué dans le cadre du module Méthode de Raking porte sur le calcul du pagerank par la méthode de Gauss Siedel descendant (sujet 5). Le but de ce projet étant de comparer le temps de calcul et le nombre d'itération nécessaire à la convergence entre le calcul du pagerank par la méthode des puissance et le calcul du pagerank par la méthode de Gauss Siedel descendant.

2 Implémentation

Le projet contient 4 versions différentes du calcul du pagerank. Voici le détail de ces 4 versions. Pour chacune des 4 versions, un surfeur aléatoire est compris dans le calcul du pagerank et nous arrêtons le calcul lorsque le résultat converge à 0.000001. Nous avons fait le choix d'utiliser le langage C pour programmer ce projet.

2.1 Version 1

Cette version est notre version initiale du pagerank. Le stockage de la matrice se fait par ligne. Notre structure de données stock une matrice composée d'un tableau de lignes. Chaque ligne est composé d'un tableau d'éléments. Ces éléments représentent une page web et contiennent le numéro des pages auxquels ils font référence avec leur probabilités.
Cette manière de stocker la matrice implique de faire le calcul du pagerank de manière "inverse". En effet, on ne peut pas calculer :

$$x^{k+1}[i] = \sum_{j=1}^{n} x^k[j]G[j,i]$$

car pour un $x[i]$ donné, on est pas capable de connaitre les $x[j]$ lui faisant référence à moins de parcourir toute la matrice. On fait donc le calcul de cette manière :

$$x^{k+1}[G[i,j]\ .destination] = x^k[i]G[i,j].probabilite$$

En itérant sur i puis pour chaque i, on itère sur j.

2.2 Version 2

Nous avons eu une nouvelle idée pour nous permettre de limiter le coût en mémoire du programme précédent. C'est ce que nous avons implémenté dans cette version. La version 2 fait le calcul du pagerank de la même manière que la version 1. Ce qui diffère est le chargement du fichier dans notre structure de matrice : au lieu de charger toute la matrice, on la charge ligne par ligne pendant le calcul. Cela permet de limiter grandement le coût en mémoire mais ralenti le calcul car pour chaque ligne, on doit faire une lecture de fichier.

2.3 Version 3

Nous avons réalisé que la structure de donnée qu'on utilisait pour faire le calcul du pagerank ne pouvait pas fonctionner pour la méthode de Gauss Siedel descendant car à aucun moment dans le calcul d'une itération, on ne peut être certain que $x^{k+1}[i]$ a fini d'être calculé. Nous avons donc changé notre structure de données faisant maintenant un stockage en colonne. Ainsi on ne stock plus pour un élément, les éléments auxquels il fait référence mais on stock pour un élément, les éléments qui lui font référence. On calcul donc les $x^{k+1}[i]$ de cette manière :

$$x^{k+1}[i] = \sum_{j=1}^{n} x^k[j]G[j,i]$$

2.4 Version 4

Une fois la version 3 fini, nous nous sommes intéressés au calcul du pagerank par la méthode de Gauss Siedel descendant. C'est ce que nous avons implémenté dans la version 4. On calcul donc les $x^{k+1}[i]$ de cette manière :

$$x^{k+1}[i](1 - G[i,i]) = \sum_{j=1}^{i-1} x^k[j]G[j,i] + \sum_{j=i+1}^{n} x^{k+1}[j]G[j,i]$$

3 Utilisation du programme

Le projet contient un Makefile permettant d'automatiser la compilation et les différentes exécution du programme. Il contient un fichier README.md dans lequel l'utilisation du programme est détaillé.

4 Interprétation des résultats

Tous les résultats que nous mettons ici on été obtenu sur la même machine. Voici les tableaux des résultats :

Pour une matrice de taille petite (14 éléments non nuls)

versions	itérations	temps lecture fichier	temps calcul pagerank	mémoire allouée dynamiquement
1	43	0.161 ms	0.154 ms	656 octets
2	43	0 ms	0.757 ms	432 octets
3	43	0.137 ms	0.471 ms	624 octets
4	27	0.14 ms	0.09 ms	624 octets

Pour une matrice de taille moyenne (59995 éléments non nuls)

versions	itérations	temps lecture fichier	temps calcul pagerank	mémoire allouée dynamiquement
1	15	34.621 ms	11.241 ms	1439968 octets
2	15	0 ms	506.892 ms	480048 octets
3	15	33.248 ms	28.571 ms	1759912 octets
4	9	33.531 ms	20.494 ms	1759912 octets

Pour une matrice de taille grande (2312497 éléments non nuls)

versions	itérations	temps lecture fichier	temps calcul pagerank	mémoire allouée dynamiquement
1	65	1071.541 ms	5482.14 ms	50531296 octets
2	65	0 ms	63602.972 ms	13531344 octets
3	65	1354.597 ms	14068.791 ms	64520824 octets
4	35	1382.416 ms	7725.151 ms	64520824 octets

Grâce à ces résultats on voit bien que la version 1 et 3 on sensiblement les mêmes performance (la version 1 étant un peu meilleure en temps et en mémoire pour le grandes matrice). Ces deux versions sont sensiblement les mêmes (seule la structure de données utilisée diffère).
On voit également que la version 2 est bien meilleure que les autres en mémoire. Cependant elle devient très lente pour une grande matrice ce qui est problématique.
Et pour finir la version 4 implémentant la méthode de Gauss Siedel descendant est bien meilleure en terme de nombre d'itérations que les autres et aussi en temps de calcul.

4.1 Conclusion

La méthode de Gauss Siedel descendant apporte réellement un gain de rapidité et de calcul pour le calcul du pagerank par rapport à la méthode des puissances. Ce projet a été très intéressant pour nous et réellement satisfaisant.