Étude comparative des méthodes de calcul de PageRank

Saint-Jean A. O. Djungu[1] Pierre Manneback Fabien Mathieu

Université de Kinshasa
R. D. Congo
djungu@yahoo.com

Faculté Polytechnique de Mons
Belgique
Pierre.Manneback@fpms.ac.be

France Télécom R&D
France
fabien.mathieu@normalesup.org

RÉSUMÉ : Dans cet article nous avons analysé, en faisant varier le facteur *zap*, la convergence en temps et en itérations de deux classes de méthodes de résolution du problème de *PageRank*. La première classe est constituée de méthodes d'extrapolation et la deuxième de méthodes itératives. Nos expériences ont montré que généralement les méthodes de Krylov sont les plus rapides.

ABSTRACT: In this paper we have analyzed, while varying the *damping* factor, convergence in time and iterations of two classes of methods for solving the PageRank problem. The first class consists in extrapolation methods and the second in iterative methods. Our experiments have shown that Krylov methods are generally the fastest.

MOTS-CLÉS : PageRank, graphe du Web, chaîne de Markov, espace de Krylov

KEYWORDS : PageRank, Web graph, Markov chain, Krylov space

[1] Boursier Doctorant financé par la C.T.B. (Coopération Technique Belge). Tous nos remerciements à la C.T.B.

1. Introduction

L'analyse de la structure du graphe formé par les pages Web et les liens hypertextes qui les relient, communément appelé *graphe du Web*, a permis d'améliorer considérablement la performance des moteurs de recherche. Ainsi, lancé en 1998, le moteur de recherche Google classe les pages grâce à la combinaison de plusieurs facteurs, dont le plus important porte le nom de *PageRank* [9]. Ce dernier utilise le nombre de liens pointant sur chaque page Web pour lui affecter un indice de popularité.

Le contenu et la structure du Web sont fondamentalement dynamiques: Cho *et al*. ont par exemple constaté dans [7] qu'en une semaine, 25% de liens peuvent être modifiés et plus de 5% de nouveau contenu peut été créé. Ce résultat indique à suffisance que les moteurs de recherche basés sur l'indice de popularité des pages, tel Google, doivent régulièrement mettre à jour leur base d'indices. Cette dynamicité, ainsi que la taille du graphe du Web (plusieurs milliards de pages) motive le besoin d'algorithmes rapides de calcul de PageRank. D'autres besoins tels que la nécessité d'assigner plusieurs valeurs de PageRank à chaque page Web pour définir des préférences d'utilisateurs [6,15], la détection de spams [16], les méthodes d'exploration de crawleur [8] nécessitent également de développer des méthodes efficaces pour le calcul de PageRank.

Le PageRank est défini comme le vecteur propre associé à la valeur propre dominante d'une matrice, dite *matrice du Web*. Une façon simple de le calculer est l'utilisation de la méthode de la puissance. Cette dernière prend cependant un temps considérable, qui peut s'exprimer en jours de calcul, étant donné la taille de la matrice et la lenteur de la convergence. Pour accélérer sa convergence, plusieurs méthodes ont été développées dont notamment des techniques d'extrapolation dérivées de la méthode de Δ^2 Aiken [13 14], des méthodes d'agrégation de nœuds du graphe [5 12] et des méthodes adaptatives [11]. Le problème de PageRank a aussi été formulé sous forme de système linéaire et différents solveurs d'équations ont été utilisés [1 3 4 10]. Toutes ces méthodes sont réputées produire une convergence plus rapide que la méthode de la puissance. Dans cette contribution, nous nous intéressons à la comparaison de ces différentes méthodes de calcul de PageRank sur ordinateur séquentiel. Nous envisageons de poursuivre cette étude sur architecture parallèle.

La structure de cet article se présente comme suit: dans la section 2, nous rappelons la formulation mathématique du PageRank. Nous présentons dans la section 3 les différents algorithmes étudiés. La section 4 est consacrée à l'examen des expérimentations numériques. Enfin, des conclusions et perspectives sont présentées.

2. Formulation mathématique du PageRank

Soit G = (V, E) un graphe du Web, où V un ensemble de *n* pages Web et E l'ensemble des couples de pages de V dont la première composante possède un lien hypertexte vers la deuxième composante. Considérons la matrice d'adjacence M du

graphe G dont les éléments m_{ij} valent 1 s'il y a un lien de la page i à la page j et 0 dans le cas contraire. Pour des pages i pointant vers d'autres pages, le degré sortant d(i) est strictement positif et les lignes i de M correspondantes peuvent être normalisées sous la forme $a_{ij}=m_{ij}/d(i)$. Supposons que le graphe G soit fortement connexe et apériodique. Le vecteur PageRank P est défini comme solution limite du processus itératif suivant:

$$\forall j \in V, \quad P_{n+1}(j) = \sum_{i} a_{ij} P_n(i) = \sum_{i \to j} P_n(i) / d(i) \qquad (1)$$

À chaque étape de ce processus itératif, une page donnée transmet son indice courant de popularité, celui-ci étant réparti vers les différentes pages pointées. L'expression (1) correspond à des transitions d'une chaîne de Markov liée à un modèle de surfeur aléatoire et dont la matrice $A=(a_{ij})$ est la matrice de transition [5 9].

Dans la pratique, le graphe G n'est cependant pas fortement connexe. Il existe un bon nombre de pages sans lien sortant ou comportant uniquement des liens sortants non indexés. Les lignes de la matrice A correspondant à ces pages sont nulles, et, dès lors, A n'est pas stochastique [5]. Pour résoudre ce problème, plusieurs approches ont été proposées. L'une d'entre elles que nous exploitons ici est la possibilité de *zapper* une fois qu'on se retrouve sur une page sans lien sortant. Ceci se traduit par la définition du défaut stochastique de la matrice A comme étant le vecteur

$$s = e - A.e \qquad (2)$$

où e est un vecteur colonne unitaire, i.e. ayant tous ses éléments égaux à 1. Une complétion de la matrice A par son défaut stochastique permet d'obtenir la matrice stochastique *A'*.

$$A' = A + s.Z^T, \qquad (3)$$

où Z définit une distribution de probabilité par défaut sur V, appelée *distribution de zap*. Initialement, Z a été choisi comme ayant une distribution uniforme, soit $Z_i=1/n$, mais la personnalisation du zapping en utilisant des distributions non uniformes a intéressé de nombreux auteurs [6,15].

Un autre problème que présente le graphe du Web est l'existence d'états récurrents. Ces états ne font qu'accumuler du rang mais ne les redistribuent pas en dehors de leur classe récurrente (états absorbants). La matrice de transition, dans ce cas, est non irréductible. Dès lors, le théorème de *Perron-Frobenius* ne garantit plus l'unicité du vecteur PageRank. Il est également possible qu'un grand nombre de pages se voient attribuer un rang nul. Pour résoudre ce problème, Page *et al* [9] ont proposé une autre incorporation de la distribution de zap, en remplaçant la matrice A' de la relation (3) par la matrice irréductible suivante :

$$A'' = \alpha.A' + (1-\alpha).e.Z^T \qquad (4)$$

où α est un coefficient de zap. Dans la pratique, on prendra le coefficient α entre 0.85 et 1. A chaque étape du processus stochastique décrit par A'', le surfeur aléatoire va

cliquer au hasard sur un des liens sortants, avec une probabilité α , ou zapper avec une probabilité (1-α) quelque part sur le Web selon la distribution de probabilité Z [5]. Le graphe sous-jacent devient une clique et la matrice A'' est à la fois stochastique-ligne et irréductible.

Le vecteur PageRank peut alors être obtenu en résolvant le problème de vecteur propre associé à la plus grande valeur, λ_1=1, de la matrice A'' :

$$P = A''^{T} P \tag{5}$$

ou par la résolution du système linéaire homogène

$$(I - A''^{T})P = 0 \tag{6}$$

où *I* est la matrice unité. Dans l'une ou l'autre formulation, on ajoute l'équation de normalisation $e^T.P=1$.

3. Méthodes de calcul de PageRank

3.1. Méthode de la puissance

La méthode de la puissance est une méthode itérative simple qui permet de trouver la plus grande valeur propre, en valeur absolue, d'une matrice et son vecteur propre correspondant. En combinant les équations (2) à (5), le vecteur PageRank P peut être calculé par le code sous-dessous.

$$\begin{aligned} P_{n+1} &= \alpha . A^{T} P_n , \\ \beta &= 1 - \left\| P_{n+1} \right\|_1 , \\ P_{n+1} &= P_{n+1} + \beta . Z \end{aligned} \tag{7}$$

3.2. Méthodes d'extrapolation

Les méthodes d'extrapolation sont basées sur l'expression des itérations P_n comme une combinaison linéaire

$$P_n = u_1 + \lambda_2{}^n u_2 + ... + \lambda_m{}^n u_m + ... , \tag{8}$$

où les termes u_m sont des vecteurs propres de A (ici λ_1=1). Ces méthodes essayent d'améliorer l'approximation de la solution u_1 en P_n par la suppression intelligente des termes les plus élevés.

Kamvar et al. [13,14] présentent une variété de méthodes d'extrapolation, la plus simple étant celle d' *Aiken*. La méthode d'Aiken vise à éliminer le deuxième terme dans (8). En prenant m=2, on obtient les 3 équations suivantes

$$\begin{aligned} P_{n-2} &= u_1 + u_2 \quad + ... \\ P_{n-1} &= u_1 + \lambda_2 u_2 \; + ... \\ P_n \; &= u_1 + \lambda_2{}^2 u_2 + ... \end{aligned} \tag{9}$$

De ces équations, on déduit une approximation de u_1 en éliminant u_2. On peut généraliser le procédé en tenant compte de plus de termes. Ainsi, l'*extrapolation quadratique* est obtenue en prenant m=3. Les méthodes d'Aiken et d'extrapolation quadratique sont appliquées périodiquement pour accélérer la convergence de la méthode de la puissance.

3.3. Méthodes itératives linéaires classiques

Une autre façon de résoudre le problème du PageRank est de considérer le système d'équations linéaires (6). En remplaçant la matrice A'' par sa valeur, nous obtenons le système suivant :

$$\left(I - \alpha A^T\right)P = \mu Z \qquad avec \quad e^T.P = 1 \tag{10}$$

où

$$\mu = \alpha.\mathrm{s}^{\mathrm{T}}.\mathrm{P} + (1-\alpha).\mathrm{e}^{\mathrm{T}}.\mathrm{P} = \left\|\mathrm{P}\right\|_1 - \alpha.\left\|\mathrm{A}^{\mathrm{T}}P\right\|_1$$

La solution au système (10) est une distribution de probabilité grâce au choix de μ. Il est aussi possible de choisir une valeur arbitraire, quitte à appliquer à la fin l'équation de re-normalisation $\mathrm{e}^{\mathrm{T}}.\mathrm{P}=1$. Nous obtenons ainsi (par exemple) :

$$\left(I - \alpha A^T\right)P = (1-\alpha)Z \tag{11}$$

La matrice $(I - \alpha.A^T)$ est une matrice carrée à la fois non singulière et non symétrique et dont la diagonale principale est strictement dominante. La méthode de Jacobi

$$P_{n+1} = \alpha.A^T P_n + (1-\alpha).Z \tag{12}$$

peut être utilisée pour trouver une solution du système linéaire (11). Notons qu'il existe un large éventail de méthodes itératives pour résoudre ce système, dont les plus simples sont celles de Jacobi et Gauss-Seidel [1,3]. La méthode de Gauss-Seidel est similaire à celle de Jacobi, sauf qu'elle réutilise les composantes déjà calculées :

$$P_{n+1} = \alpha \sum_{j<i} a_{ji} P_{n+1}[j] + \alpha \sum_{j>i} a_{ji} P_n[j] + (1-\alpha)Z[i] \tag{13}$$

Notons que les méthodes de la puissance et de Jacobi sont très proches et leur convergence est asymptotiquement géométrique, avec raison α. La méthode de Gauss-Seidel converge aussi géométriquement, mais avec une raison plus petite [1 5].

3.4. Méthodes basées sur les sous-espaces de Krylov

Notons par $Q = (I - \alpha.A^T)$ et $b = (1 - \alpha).Z$ respectivement la matrice et le second membre du système (11). Soient P_0 une approximation initiale du vecteur PageRank et $r_0=b-Q.P_0$ le résidu. Une méthode itérative est dite solveur de Krylov si les itérations successives de P sont obtenues comme suit :

$$P_{n+1} = P_n + \mu_{n+1} \tag{14}$$

où la direction de descente μ_{n+1} est un vecteur du sous-espace de Krylov de dimension n+1 engendré par les vecteurs $\{r_0, Qr_0, \ldots, Q^n r_0\}$, choisit de façon à minimiser $b–Q.P_{n+1}$.

Parmi les solveurs de Krylov, on retrouve notamment les méthodes GMRES, BiCG et BiCGSTAB [4].

4. **Résultats expérimentaux**

Pour l'expérimentation numérique nous avons utilisé deux matrices standards dites matrices Web de Stanford[2], ainsi qu'une matrice générée par le crawleur surfer.m [2]. Ce dernier crawling a été effectué à partir de la page Web http://www.fpms.ac.be, page principale du site Web de la Faculté Polytechnique de Mons et a été arrêté après 20000 nœuds. Nous l'avons baptisé FPMs.

	#Nœuds	#Liens	Espace
FPMs	20.000	387.273	1,92 Mo
Stanford	281.903	2.312.497	27,80 Mo
Stanford-Berkeley	683.446	7.583.376	89,30 Mo

Tableau 1 : Caractéristiques des matrices utilisées

Le tableau 1 reprend les principales caractéristiques de ces matrices. Tous les programmes ont été écrits en Matlab et testés sur Pentium(R) 4 CPU 3.0 GHZ, 512 Mo de RAM. Le critère d'arrêt utilisé pour comparer les différentes méthodes est l'erreur relative exprimée comme suit :

$$\frac{\left\|P_{n+1}-P_n\right\|}{\left\|P_n\right\|} \le 10^{-7} \qquad (15)$$

où $\|P_n\|=1$ à chaque itération pour la méthode de la puissance et ses dérivées.

La figure 1 illustre les résultats obtenus en utilisant la matrice FPMs (α=0.85). Le tableau 2 donne des résultats plus détaillés pour les trois graphes étudiés. Notons que la méthode GMRES a été utilisée dans sa version avec redémarrage (GMRES(m)) toutes les 20 itérations, et cela afin d'éviter la manipulation de trop grosses matrices denses.

Les résultats de nos analyses expérimentales permettent de constater:

- Pour un bon choix de la fréquence d'extrapolation, la méthode d'Aiken converge plus vite en temps que celles d'extrapolation quadratique et de la puissance.

- La méthode de la puissance et celle de Jacobi ont presque le même taux de convergence et exigent moins d'espace de stockage.

- La méthode de Gauss-Seidel converge plus vite en itérations que la méthode de Jacobi; ce qui n'est pas le cas pour la convergence en temps. Pour certains graphes, la méthode de Jacobi est le meilleur choix par rapport aux méthodes de la puissance et Gauss-Seidel. Gauss-Seidel peut s'avérer désastreux en temps pour des grosses matrices, ce qui est dû, selon nous, à la pagination (accès fréquent à la mémoire virtuelle). Il est en de même pour BiCG, qui utilise aussi la transposée de la matrice du système.

[2] Ces matrices ont été téléchargées à partir de la page http://www.stanford.edu/~sdkamvar/research.html (dernière consultation le 21 janvier 2006)

- La convergence des méthodes d'extrapolation et des solveurs de Krylov est non monotone. Les solveurs BiCGSTAB et GMRES sont plus rapides et exigent moins d'itérations que toutes les autres méthodes étudiées. En plus ces deux méthodes sont moins sensibles à la variation du facteur zap.

De nos expériences, il semble que GMRES et BiCGSTAB soient les méthodes les plus indiquées pour architecture séquentielle d'ordinateurs. Il faudrait cependant confirmer ces tests par d'autres expériences sur de très grosses matrices du Web.

Méthode	FPMs		Stanford		Stanford-Berkeley	
	Iter.	Temps	Iter.	Temps	Iter.	Temps
Puissance	69	1.71	77	27.14	78	51.42
Aiken	54	1.39	73	26.09	70	47.21
Quadratique	67	1.82	55	34.75	68	51.10
Jacobi	61	1.31	77	26.20	78	46.68
Gauss-Seidel	37	1.84	41	977.19	43	1598.57
GMRES	35	0.79	68	26.39	85	52.10
BiCG	39	2.45	100	130.53	95	142.67
BiCGSTAB	43	1.54	41	22.93	42	37.31

Tableau 2 : Résultats en temps et en itérations

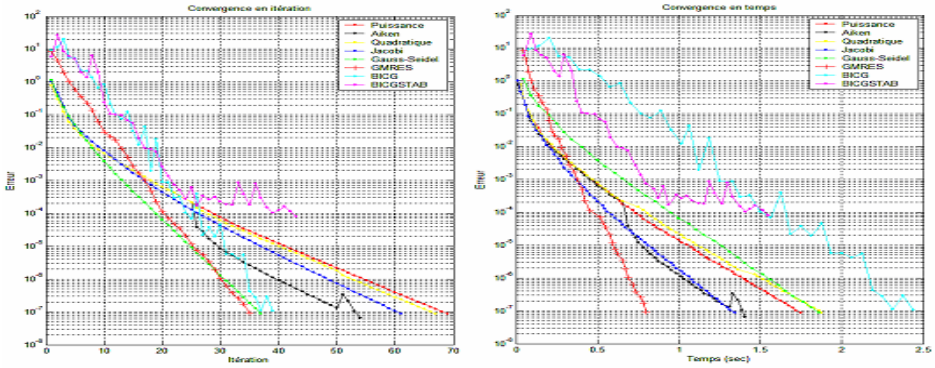

Figure 1 : Convergence des méthodes itératives sur le graphe FPMs (α=0.85)

5. Conclusion

Dans cet article nous avons implémenté, en Matlab, les méthodes de la puissance, Aiken, extrapolation quadratique, Jacobi, Gauss-Seidel, GMRES, BiCG et BiCGSTAB pour calculer le vecteur PageRank. Les 8 méthodes ont été testées sur des matrices Web de taille moyenne. Nous avons constaté que les solveurs BiCGSTAB et GMRES exigent en général moins d'itérations et convergent plus vite que les autres méthodes.

Pour des travaux futurs, nous proposons de paralléliser ces solveurs et d'en faire la comparaison avec la méthode de la puissance.

Bibliographie

[1] A. Arasu, J. Novak, A. Tomkins et J. Tomlin. *PageRank Computation and the Structure of the Web: Experiments and Algorithms*. The Eleventh International World Wide Web Conference, 2002.

[2] Cleve B. Moler. *Numerical Computing with MATLAB*. Philadelphia, PA: SIAM, 2004.

[3] D. Corso, A. Gulli et F. Romani. *Exploiting Web Matrix Permutations to Speedup PageRank Computation*. Technical Report IIT TR-04, Istituto di Informatica e Telematica, 2004.

[4] D. Gleich, L. Zhukov et P. Berkhin. *Fast Parallel PageRank: A linear System Approch*. Technical Report YRL-2004-038, Yahoo!, 2004.

[5] Fabien Mathieu, Graphes du Web: Mesures d'importance à la PageRank, Thèse, Université Montpellier II, 2004.

[6] G. Jeh et J. Windom. *Scaling personalized Web search*. In WWW12, pages 271-279.ACM Press, 2003.

[7] J. Cho et S. Roy. *Impact of Web search engines on page popularity*. In Proceedings of the Thirteenth International WWW Conference, 2004.

[8] J. Cho, H. Garc´ıa-Molina et L. Page. *Efficient crawling through URL ordering*. Computer Networks and ISDN Systems, 30(1–7):161–172, 1998.

[9] L. Page, S. Brin, R. Motwani et T. Winograd. *The PageRank Citation Ranking: Bringing Order to the Web*. Technical report, Computer Science Department, Stanford University, 1998.

[10] M. Bianchini, M. Gori et Franco Scarselli. *Inside PageRank*. ACM Press, 2005.

[11] S. Kamvar, T. Haveliwala et G. Golub. *Adaptive methods for the computation of pagerank.* Technical Report, Stanford University,2003.

[12] S. Kamvar, T. Haveliwala, C. Manning et G. Golub. *Exploiting the block structure of the Web for computing PageRank*. Technical Report , Stanford University, 2003.

[13] S. Kamvar, T. Haveliwala, C. Manning et G. Golub. *Extrapolation methods for accelerating pagerank computations*. In Proceedings of the Twelfth International World Wide Web Conference., 2003.

[14] T. Haveliwala, S. Kamvar, D. Klein, C. Manning et G. Golub. *Computing PageRank using Power Extrapolation*. Technical report, Stanford University, 2003.

[15] T. Haveliwala, S. Kamvar et G. Jeh. *An analytical comparison of approaches to personalizing PageRank*. Technical report, Stanford University, 2003.

[16] Z. Gyongyi, H. Garcia-Molina et J. Pedersen. *Combating web spam with trustrank*. In 30th International Conference on Very Large Data Bases, pages 576–587, 2004.