TP PageRank

1 Introduction

Une page web est importante si des pages web importantes pointent vers elle. C'est sur cette idée que Brin et Page ont construit l'algorithme PageRank à la fin des années 90 — un brevet a été déposé en 1998 et validé en 2001.

Soit $\mathcal{P}$ un ensemble de pages. Il s'agit d'associer à chacune un score d'importance. On considère le graphe orienté $G = (\mathcal{P}, A)$ contenant un sommet pour chaque page $P_i \in \mathcal{P}$, et un arc $(P_i, P_j) \in A$ pour chaque hyperlien de la page P_i vers la page P_j. L'ensemble $pred(P_i)$ des prédécesseurs de P_i dans G correspond à l'ensemble des pages pointant (*backlink*) sur P_i (et le demi-degré intérieur de P_i, $d^{\circ -}(P_i)$, est la cardinalité de cet ensemble). L'ensemble $succ(P_i)$ des successeurs de P_i dans G correspond à l'ensemble des pages sur lesquelles P_i pointe (et le demi-degré extérieur de P_i, $d^{\circ +}(P_i)$, est la cardinalité de cet ensemble).

Comment exprimer le score $r(P_i)$ traduisant l'importance de la page $P_i \in \mathcal{P}$? Une première possibilité consiste à définir ce score par la somme des importances des pages pointant sur P_i normalisées par le nombre de liens dans chacune de ces pages :

$$r(P_i) = \sum_{P_j \in pred(P_i)} \frac{r(P_j)}{d^{\circ +}(P_j)}$$

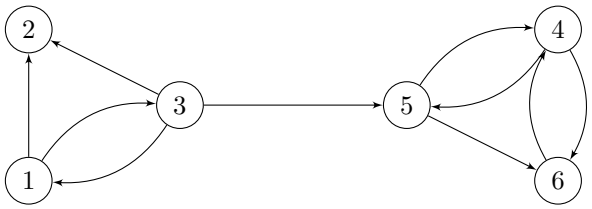
Pour passer de cette définition à un algorithme, on propose de la transformer en une relation de récurrence : initialement, toutes les pages ont le même score $\frac{1}{|\mathcal{P}|}$; à chaque étape, on applique la définition précédente pour modifier le score de chaque page en fonction des scores des pages pointant sur elle.

$$r_0(P_i) = \frac{1}{|\mathcal{P}|}$$
$$r_{k+1}(P_i) = \sum_{P_j \in pred(P_i)} \frac{r_k(P_j)}{d^{\circ +}(P_j)}$$

Sous cette forme, on voit que l'idée de normaliser par $d^{\circ +}(P_j)$ est guidée par l'intention de conserver l'importance au fil des itérations. On remarque cependant que la convergence de ce processus n'est pas assurée.

2 Exemple

Voici un exemple de graphe avec sa matrice d'adjacence (M). On introduit également la matrice de marche aléatoire correspondante (H) : cette matrice est obtenue à partir de M en divisant chaque coefficient $M[i][j]$ par $d^{\circ +}(i)$.



$$M = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$H = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

3 Expression matricielle de l'algorithme itératif

Exercice 1 : Exprimer sous la forme du produit d'un vecteur par une matrice l'algorithme itératif décrit par la relation de récurrence introduite plus haut. Remarquer que l'itération n de l'algorithme correspond à la valeur de H^n.

4 Représentation des matrices creuses

Le graphe du web comprend plusieurs milliards de pages. [1] Une page ne possède en général que quelques dizaines de liens. Ainsi, la matrice d'adjacence associée au graphe du web est extrêmement creuse. Autrement dit, le demi-degré intérieur de chaque sommet du graphe est très petit par rapport au nombre de sommets. C'est pourquoi il est plus efficace de considérer une représentation du graphe sous forme de listes d'adjacence.

Un code source est fourni [2] qui comprend des structures de données pour représenter matrices creuses et vecteurs, des fonctions pour allouer/libérer la mémoire nécessaire pour ces structures et pour lire/écrire la matrice d'un graphe depuis/vers un format textuel. Le graphe exemple introduit plus haut est stocké dans le fichier `g.dat`. Un plus gros graphe formé de pages traitant du mot clé "genetic" est également fourni avec le fichier `genetic.dat`.

Exercice 2 : Étudier les structures de données utilisées pour représenter une matrice creuse.

1. `https://www.worldwidewebsize.com/`
2. `http://peportier.me/teaching_2019_2020/`

Exercice 3 : Implémenter une première version de l'algorithme à partir de la formulation découverte à l'exercice 1. Il s'agit de modifier itérativement le vecteur d'importance, c'est-à-dire de calculer r_{k+1} à partir de r_k, et non de calculer explicitement H^n.

5 Nœuds absorbants

Pour l'instant, l'algorithme ne converge pas de manière satisfaisante. En particulier, le nœud 2, dit *absorbant*, ne redistribue pas son importance. Autrement dit, H n'est pas stochastique car certaines de ses lignes ne sont pas des vecteurs de probabilité (i.e. la somme des valeurs d'une ligne peut être différente de 1).

Exercice 4 : Comment transformer la matrice H en une matrice S stochastique en changeant le moins possible la signification de H (i.e., une marche aléatoire le long des hyperliens).

Exercice 5 : Modifier l'implémentation de l'algorithme pour travailler sur S au lieu de H tout en continuant à bénéficier du caractère creux de H.

6 Ergodicité

La convergence n'est toujours pas satisfaisante. Voici des questions qui peuvent guider l'analyse de ce phénomène.

Exercice 6 : À quoi correspondent les puissances de la matrice d'adjacence M du graphe G ? Par exemple, que représentent les valeurs sur la diagonale principale de M^3 ? Comment interpréter les puissances de la matrice stochastique S ?

Une matrice stochastique est dite *ergodique* si, pour deux nœuds quelconques du graphe sous-jacent, la probabilité de rejoindre le second en partant du premier en un nombre fini d'étapes n'est pas nulle.

Pour rendre la matrice S ergodique, Brin et Page ont introduit la métaphore du *surfeur aléatoire*. Le surfeur suit la structure des hyperliens du web mais il peut parfois saisir une URL dans la barre de navigation pour *se téléporter* vers une page quelconque. Pour la téléportation, chaque page web cible est équiprobable.

La combinaison convexe de S avec une matrice de rang 1 qui représente la téléportation la rend ergodique :

$$E = \alpha S + (1 - \alpha)\frac{1}{n}ee^T$$

Avec $0 \leq \alpha \leq 1$ et e un vecteur de dimension n ne comportant que des 1.

La matrice E est ergodique mais dense. La matrice H est creuse. Pour que le calcul du vecteur d'importance puisse en bénéficier, il faut exprimer E en fonction de H. Dans le calcul ci-dessous, a est un vecteur indicateur des nœuds absorbants (il possède des zéros partout sauf aux indices des nœuds absorbants).

$$
\begin{aligned}
E &= \alpha S + (1-\alpha)\frac{1}{n}ee^T \\
&= \alpha\left(H + \frac{1}{n}ae^T)\right) + (1-\alpha)\frac{1}{n}ee^T \\
&= \alpha H + (\alpha a + (1-\alpha)e)\frac{1}{n}e^T
\end{aligned}
$$

La mise à jour du vecteur d'importance s'exprime alors ainsi :

$$r_{k+1}^T = \alpha r_k^T H + (\alpha r_k^T a + 1 - \alpha)\frac{1}{n}e^T$$

Exercice 7 : Étudier la dérivation ci-dessus et vérifier qu'elle permet à la mise à jour du vecteur d'importance de continuer à bénéficier du caractère creux de H.

Exercice 8 : Modifier l'implémentation de l'algorithme pour travailler sur E au lieu de S tout en continuant à bénéficier du caractère creux de H. Tester votre algorithme sur `genetic.dat`.