WEM : Web Mining
Laboratoire n°2
Analyse des hyperliens

24.03.2017

Objectifs

Dans ce laboratoire, vous allez implémenter l'analyse des hyperliens et l'intégrer dans votre moteur de recherche. Plus précisément, les points étudiés dans ce laboratoire seront :

- Le Hub et l'Autorité
- Le PageRank

Durée

- 3 périodes. A rendre le **28.04.2017** à 14h00 au plus tard.

Références

- Cours «Web Mining» de Nastaran Fatemi et Laura Raileanu

Donnée

Ce laboratoire se déroule en deux étapes. Dans la première, vous implémenterez l'analyse des liens et dans la seconde vous intégrerez cette analyse à votre moteur de recherche.

1. **Implémentation de l'analyse des hyperliens**

   Soit :

   - $G = (V, E)$ un graphe représentant un ensemble de sites où $V = (p_1, p_2, \dots, p_n)$
   - M la matrice d'adjacence de G où $m_{i,k} = 1$ si $(p_i, p_k) \in E$, 0 sinon

   On peut calculer le Hub (h), l'Autorité (a), et le PageRank (Pr) comme:

$$h^c = M \cdot a^{c-1}$$
$$a^c = M^T \cdot h^{c-1}$$
$$Pr^c = normalize(0.85 \cdot (Mt^T \cdot Pr^{c-1}) + E)$$

   Avec (pour le PageRank) :

   Mt : Matrice d'adjacence (ATTENTION utiliser la matrice de TRANSITION !!!)
   E : Vecteur de probabilités de saut aléatoire : others → $0.15 / |V|$
   Pr^0 : others → $1/|V|$

Dans cette première partie il vous est demandé d'implémenter ces 3 algorithmes. Pour vous aider, plusieurs fichiers Java sont mis à votre disposition à l'emplacement habituel. Le travail consiste à compléter la classe *LinkAnalysis* (3 méthodes à implémenter). Pour ce faire, vous avez à disposition les méthodes définies dans l'interface *AdjacencyMatrix*.

Pour tester votre travail, vous devez vous aider de la classe *GraphFileReader* qui vous permet d'obtenir une instance de matrice d'adjacence (un objet de type *AdjacencyMatrix*) à partir d'un graphe défini dans un fichier texte.

Le fichier doit impérativement respecter le format de l'exemple suivant :

```
# nodes
a
b
c
d
# edges
a;c
c;a
d;b
```

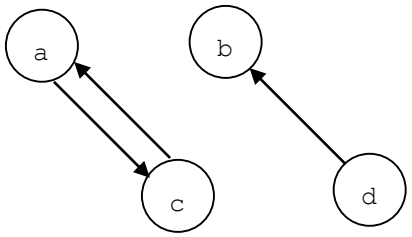

Figure 1 – Fichier de graphe au format texte ainsi que sa représentation graphique.

Votre programme devra **effectuer une analyse du graphe représenté sur la Figure 2**, et afficher sur la console ou dans un fichier les informations suivantes pour chaque nœud (effectuez 5 itérations en normalisant à chaque étape):

- Le Hub
- L'Autorité
- Le PageRank

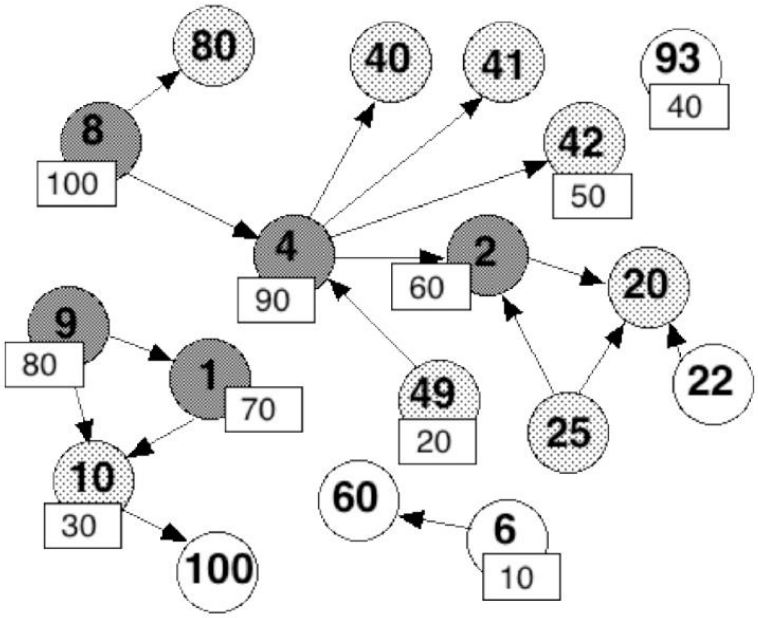

Figure 2 - Graphe à analyser

2. Intégration dans le moteur de recherche

La seconde partie du travail consiste à **intégrer l'analyse des liens dans votre moteur de recherche**. Pour ce faire, il vous faut générer une matrice d'adjacence représentant votre corpus en utilisant la classe *ArrayListMatrix*. Pour vous faciliter la tâche, inspirez-vous du code de *GraphFileReader*. Le site choisi ne doit pas contenir plus de 500 pages pour ne pas provoquer un débordement de mémoire. Vous pouvez augmenter la taille de la mémoire avec le flag *-Xmx 2048m* lors du lancement du programme (paramètre de la commande *java*).

Ensuite, vous pourrez réutiliser le code développé en première partie pour connaître notamment le PageRank de chaque page. Finalement, **combinez le score obtenu par *TFIDF* des pages avec leur PageRank pour ordonner les résultats retournés par votre moteur de recherche**. La technique la plus simple consiste à multiplier le PageRank par le score *TFIDF*. Vous pouvez cependant choisir une autre façon de les combiner.

Vous devez prendre garde aux URLs multiples qui pointent vers la même page, sinon vous risquez d'obtenir un graphe où deux nœuds différents représentent la même page.

Pour éviter cela vous pouvez :

a. Stocker en mémoire le hash des pages avec le nœud qu'il représente pour faire pointer tous les liens vers une seule et unique page. L'utilisation d'une fonction de hash cryptographique est loin d'être optimale, il existe d'autres méthodes (voir point b.).

b. Utiliser l'algorithme discuté par des ingénieurs de Google pour la détection de pages web dupliquées décrit dans l'article « Detecting near-duplicates for web crawling »[1].

Rendu/Evaluation

Vous remettrez *sur Moodle* un projet Eclipse zippé contenant les sources, libraires utilisées, fichiers d'entrée, etc. Attention aux indexes qui peuvent être volumineux et qu'il n'est pas nécessaire de rendre.

Nous ne demandons pas de rapport, mais veuillez bien commenter votre code, au minimum les étapes importantes, les choix que vous avez dû faire ainsi que tout ce dont vous jugerez utile.

Vous pouvez discuter entre les groupes mais il est strictement interdit d'échanger du code.

Adresse E-Mail de l'assistant : fabien.dutoit@heig-vd.ch

Bonne chance !

[1] Vous pouvez le trouver à l'adresse suivante : http://dl.acm.org/citation.cfm?id=1242572.1242592