TP3 – La recherche

M2 informatique – Université Paris Cité

Année 2022-2023

Traitement de la requête

Exercice 1 *Requête*

La requête de l'utilisateur consiste en un ensemble de mots.

1. Enlever de cet ensemble les mots redondants et les mots « vides » (voir TP 1).
   (Optionnel) Pour chaque mot, si nécessaire en corriger l'orthographe si vous avez programmé ces fonctions au TP 1. Sinon, on considérera seulement les requêtes à base de mots corrects.
   (Optionnel) Pour chaque mot, trouver sa racine si vous avez programmé ces fonctions au TP 1.
2. Selon le temps dont vous disposez : programmer l'une des variantes ci-dessous pour calculer le résultat de la requête.

Rappel sur les scores

Le score d'une page d par rapport à la requête r est $s(d,r) = \alpha f(d,r) + \beta p(d)$, où $f(d,r)$ est le score obtenu par la fréquence des mots, et $p(d)$ le pagerank de d, avec α et β des coefficients à choisir de manière pertinente ($\alpha + \beta = 1$).

NB : il peut être judicieux de réduire la variance du pagerank en prenant $p(d)^\gamma$, pour un certain $\gamma < 1$, plutôt que $p(d)$ dans la fonction f. Faire des tests pour trouver la valeur de γ qui augmente la pertinence des résultats.

Le score de fréquence $f(d,r)$ est calculé ainsi (en reprenant les notations du TP 1) :

— soit

$$N_r = \sqrt{\sum_{m \in r} IDF(m)^2}$$

la norme du vecteur associé à la requête (la somme est prise sur l'ensemble des mots m contenus dans la requête) ;

— on rappelle que N_d est la norme du vecteur TF associé à d calculée au TP 1 ;

— alors

$$f(d,r) = \Big(\sum_{m \in r} IDF(m) \times TF(m,d)\Big) / \Big(N_d \times N_r\Big).$$

Calcul des résultats

Exercice 2 *Version simple*

1. Donner un algorithme *efficace* qui, à partir de la relation mots-pages, énumère toutes les pages contenant *tous* les mots de la requête. On ne fera qu'un seul parcours des listes concernant les mots de la requête.
2. Calculer le score $s(d, r)$ de chacune de ces pages et les trier par score décroissant.

Exercice 3 *Cas d'une requête d'un seul mot*

(Optionnel) Lorsque la requête ne contient qu'un seul mot m, la liste des pages contenant m peut être grande et le calcul ci-dessus prendrait trop de temps.

Au prix d'un espace mémoire accru, on peut précalculer les scores $s(d, m)$ par rapport à la requête m de chaque page contenant m, et stocker une fois pour toute la liste des pages par score décroissant, permettant une réponse immédiate sur ce genre de requête.

Exercice 4 *Version optimisée WAND*

1. Dans la relation mots-pages, trier par ordre de pagerank $p(d)$ décroissant la liste des pages d associées à chaque mot.
2. Pour chaque mot m, parcourir de droite à gauche la liste des pages d associées et calculer pour chacune d'elles le maximum des valeurs $IDF(m) \times TF(m, d)/N_d$ vues jusqu'à présent : on notera $v(m, d)$ ce maximum.
3. Pour accélérer les calculs ci-dessous, on trouvera la position à laquelle avancer chaque pointeur grâce à une recherche dichotomique.
4. Programmer l'algorithme WAND vu en cours pour obtenir les k meilleures pages :
   — un tas t contient les meilleures pages vues jusqu'à présent ;
   — le score de la k-ème meilleure page est γ ;
   — on calcule N_r d'après la requête r ;
   — chaque liste de pages pour un mot m de la requête possède un pointeur vers la page d^m en cours ;
   — on trie ces pointeurs par $p(d^m)$ décroissant : on notera $d_1, \ldots, d_n$ les pages pointées dans l'ordre, et $m_1, \ldots, m_n$ les mots correspondants ;
   — soit j minimal tel que $\beta p(d_j) + (\alpha/N_r) \sum_{i=1}^{j} v(m_i, d_i) \geq \gamma$ (on appelle d_j le « pivot ») ;
   — pour i de 1 à $j-1$, avancer le pointeur d_i jusqu'à la première page dont le pagerank est $\leq p(d_j)$;
   — si d_j contient suffisamment de mots de la requête, calculer son score $s(d_j)$;
   — si $s(d_j) > \gamma$, mettre à jour γ et le tas t ;
   — avancer d'un cran tous les pointeurs qui pointent vers d_j ;
   — recommencer.

Déploiement

Exercice 5 *Site*

Programmer un serveur et réaliser un site web fonctionnel où l'utilisateur entre sa requête et où s'affiche la liste des résultats de la recherche.

Penser à écrire une fonction de conversion des liens Wikipédia donnés dans le fichier `frwiki.xml` en liens `html` pour qu'on puisse suivre les liens donnés par votre moteur de recherche.

Exercice 6 *Améliorations possibles*

Quels sont les défauts de votre moteur de recherche ? Comment les améliorer ? (Inutile de programmer ces améliorations.)