Gogol

Projet commun *Bases de données / Programmation C*

(IUT, département informatique, 1re année)

Laurent AUDIBERT

L'objectif du projet consiste à developper un petit moteur de recherche en appliquant les connaissances acquises dans les cours de bases de données et de programmation C.

1 Principe de fonctionnement d'un moteur de recherche

Un moteur de recherche est une application permettant de retrouver des ressources (pages web, articles de forums Usenet, images, vidéo, fichiers, etc.) associées à des mots quelconques. Le fonctionnement d'un moteur de recherche sur Internet se décompose en trois processus principaux décrits ci-dessous.

L'exploration ou crawl – Le web est systématiquement exploré par un robot d'indexation suivant récursivement tous les hyperliens qu'il trouve et récupérant les ressources jugées intéressantes. L'exploration est lancée depuis une ressource pivot, comme une page d'annuaire web. *Googlebot* est le user agent (signature) du crawler de *Google*.

L'indexation – Elle consiste à extraire les mots considérés comme significatifs du corpus à explorer. Les mots extraits sont enregistrés dans une base de données organisée comme un gigantesque dictionnaire inverse ou, plus exactement, comme l'index terminologique d'un ouvrage permettant de retrouver rapidement dans quel chapitre de l'ouvrage se situe un terme significatif donné. Les termes non significatifs s'appellent des mots vides (*stopwords* en anglais). Les termes significatifs sont associés à un poids. Celui-ci reflète à la fois la probabilité d'apparition du mot dans un document et le pouvoir discriminant de ce mot dans une langue, conformément au principe de la formule TF-IDF (de l'anglais *Term Frequency-Inverse Document Frequency*).

La recherche – Elle correspond à la partie requêtes du moteur, qui restitue les résultats. Un algorithme est appliqué pour identifier dans le corpus documentaire (en utilisant l'index), les documents qui correspondent le mieux aux mots contenus dans la requête, afin de présenter les résultats des recherches par ordre de pertinence supposée. Les algorithmes de recherche font l'objet de très nombreuses investigations scientifiques. Les moteurs de recherche les plus simples se contentent de requêtes booléennes pour comparer les mots d'une requête avec ceux des documents. Mais cette méthode atteint vite ses limites sur des corpus volumineux. Les moteurs plus évolués sont basés sur le paradigme du modèle vectoriel : ils utilisent la formule TF-IDF pour mettre en relation le poids des mots dans une requête avec ceux contenus dans les documents. Cette formule est utilisée pour construire des vecteurs de mots, comparés dans un espaces vectoriel, par une similarité cosinus. Pour améliorer encore les performances d'un moteur, il existe de nombreuses techniques, la plus connue étant celle du *PageRank* de *Google* qui permet de pondérer une mesure de cosinus en utilisant un indice de notoriété de pages.

Des modules complémentaires sont souvent utilisés en association avec les trois briques de base du moteur de recherche. Les plus connus sont décrits ci-dessous.

Le correcteur orthographique permet de corriger les erreurs introduites dans les mots de la requête.

Le lemmatiseur permet de réduire les mots recherchés à leur lemme (ex : le lemme de *introduisant* est *introduire*) et ainsi d'étendre la portée de la recherche en s'affranchissant des différentes formes fléchies des mots (ex : introduisant, introduis, introduit, introduisons, introduisez, introduisent, introduirai. . .).

L'antidictionnaire est utilisé pour supprimer à la fois dans l'index et dans les requêtes tous les mots *vides* (tels que : *de*, *le*, *la*. . .) qui sont non discriminants et perturbent le score de recherche en introduisant du bruit.

Bien entendu, ce projet d'initiation ne consiste pas à developper un véritable moteur de recherche.

***Remarque** : cette section a été rédigée en utilisant l'article Wikipedia Moteur de recherche (`http://fr.wikipedia.org/wiki/Moteur_de_recherche`).*

2 Objectifs du projet

L'objectif du projet est de réaliser un petit moteur de recherche capable d'identifier parmi un ensemble de fichiers texte ceux qui correspondent le mieux à la recherche d'un utilisateur. Pour réaliser ce projet, vous avez à disposition un répertoire `corpus` contenant environ 3000 petits fichiers texte ainsi qu'un répertoire `antidictionnaire` contenant un fichier texte énumérant les mots vides (un par ligne). Le corpus (*i.e.* l'ensemble des quelques 3000 fichiers) peut être considéré comme le résultat du premier processus du moteur de recherche : l'exploration. Il reste donc à implémenter les deux processus suivants :

- l'indexation
- puis la recherche.

L'implémentation de l'indexation consiste à développer un programme (*GogolIndex*) en `C` qui va lire chacun des fichiers du corpus pour en extraire les mots significatifs (*i.e.* autres que ceux qui figurent dans l'antidictionnaire) et les enregistrer dans une base de données organisée comme un gigantesque dictionnaire inverse. La base de donnée doit ainsi permettre de connaître l'ensemble des fichiers contenant chacun des mots significatifs extraits.

L'implémentation de la recherche consiste à développer un programme (*GogolRecherche*) en `C` qui va demander à l'utilisateur de saisir une requête puis retourner la liste des fichiers contenant au moins l'un des mots significatifs de la requête. Cette recherche est bien entendu réalisée en effectuant des requêtes sur la base de données. Les noms des fichiers retournés seront ordonnés en mettant en premier ceux qui contiennent le plus de mots significatifs et en dernier ceux qui en contiennent le moins.

Exemple de requête :

```
Requête : le tribunal criminel a condamné

===== Résultat contenant 3 mots clés
fichier_0866.txt : Lucerne, 30 août (ats) Le tribunal criminel de Lucerne a condamné un
fichier_0922.txt : Lausanne, 15 sept (ats) Le tribunal criminel de Lausanne a condamné jeudi un
fichier_0992.txt : Lucerne, 11 oct (ats) Le tribunal criminel de Lucerne a condamné un
fichier_1481.txt : Lucerne, 21 fév (ats) Le tribunal criminel de Lucerne a condamné mardi deux
fichier_1713.txt : Lucerne, 18 avril (ats) Le tribunal criminel de Lucerne a condamné un Suisse
fichier_1746.txt : En première instance, le tribunal criminel avait condamné l'accusée à 10 ans
fichier_2288.txt : Lucerne, 29 août (ats) Le tribunal criminel de Lucerne a condamné sept
===== Résultat contenant 2 mots clés
fichier_0911.txt : d'emprisonnement pour trafic de drogue. Le tribunal criminel de Lucerne l'a
fichier_0244.txt : criminels de guerre. Il s'agit d'un tribunal privé, dont les jugements n'ont
fichier_0483.txt : été condamné comme criminel de guerre par le tribunal international de
fichier_1229.txt : et criminelle d'Allemagne. Elle demande au tribunal de Karlsruhe d'annuler sa
fichier_0866.txt : Lucerne, 30 août (ats) Le tribunal criminel de Lucerne a condamné un
fichier_0911.txt : d'emprisonnement pour trafic de drogue. Le tribunal criminel de Lucerne l'a
...
```

La requête de l'exemple est *le tribunal criminel a condamné*. Une fois les mots vides retirés de la requête, il ne reste plus que trois mots significatifs : *tribunal*, *criminel* et *condamné*. L'exemple montre des extraits des fichiers pour chacun des résultats. Cette fonctionnalité n'est pas demandée mais constitue un plus pour le projet.

3 Partie bases de données du projet

Concernant la partie base de données du projet vous devrez :

- Modéliser la base de données à utiliser pour le projet en produisant un modèle entités-associations.
- Créer la base de données correspondant au modèle entités-associations produit.
- Être capable d'établir une connexion à votre base de données pour chacun des deux programmes *GogolIndex* et *GogolRecherche*.
- Alimenter votre base de données dans le programme *GogolIndex*.
- Écrire les bonnes requêtes dans le programme *GogolRecherche*.

4 Partie programmation C du projet

La partie programmation C du projet consiste à developper en `C` les deux programmes *GogolIndex* et *GogolRecherche*.

Le programme ***GogolIndex*** doit être capable de :
- Parcourir tous les fichiers du dossier `corpus`.
- Pour chacun des fichiers, découper le fichier en mots, supprimer les mots qui figurent dans l'anti-dictionnaire, alimenter la base de données avec les mots significatifs restants.

Le programme ***GogolRecherche*** doit être capable de :
- Demander à l'utilisateur d'écrire sa requête.
- Découper la requête en mots et en extraire les mots significatifs.
- Interroger la base de données.
- Afficher le résultat de la requête.

5 Modalités de remise du projet

Ce projet est à réaliser en binômes. Le rendu se fera impérativement sous forme de deux dossiers papiers à remettre le lundi 11 février 2013 au secrétariat, à l'attention de M. Audibert pour celui concernant la partie bases de données, et à l'attention de M. Lebbah pour celui concernant la partie programmation C. Tout retard sera sanctionné : 2 points de pénalité seront appliqués par jour de retard.

Une soutenance aura ensuite lieu jeudi 14 février. Elle permettra d'évaluer le modèle entité association, les choix de requêtes et d'implémentation ainsi que le travail de l'étudiant au sein du binôme. Deux étudiants d'un même binôme pourront avoir deux notes différentes en fonction de leur implication.

Contenu du dossier bases de données :
- Modèle entités-associations accompagné d'une explication ;
- Extraits des requêtes de l'application *GogolRecherche* accompagnés d'une explication.

Contenu du dossier programmation C :
- Un descriptif technique : organisation générale, choix des structures de données (les variables et leur type, les structures, etc .), découpage en fonctions, etc.
- Une analyse succincte expliquant les choix techniques.
- Un guide d'utilisation.
- Un listing du programme (commenté et indenté avec une entête pour le programme principal et une entête pour chaque fonction).

Déroulement de la soutenance :
- Présentation du modèle entités-associations ;
- Présentation des requêtes de l'application *GogolRecherche* ;
- Démonstration des applications *GogolIndex* et *GogolRecherche* ;
- Réponses à des questions individuelles.