TD 1 - Indexation

Exercice 1 – Pondération td-idf

Soient :

- un document qui contient le texte "maison belle maison"
- une collection de 100 documents
- le terme "maison" apparaît dans 20 documents pour un nombre d'occurrences de 35 au total
- le terme "belle" apparaît dans 35 documents pour un nombre d'occurrences de 40 au total.

x	2	3	5	7	10
$\log x$	0.7	1.1	1.8	2.0	2.3

Q 1.1 Quelle est la pondération "tf-idf" des termes "maison" et "belle" pour le document ? Commentez les valeurs obtenues.

Exercice 2 – Structure des fichiers d'indexation et pondération

On considère la collection de documents suivantes :

- Doc 1 : the new home has been sold on top forecasts
- Doc 2 : the home sales rise in july
- Doc 3 : there is an increase in home sales in july
- Doc 4 : july encounter a new home sales rise

Ainsi qu'une liste de mots vides : the, a, an, have, be, on, behind, under, there, in, ...

Q 2.1 Identifier l'ensemble des termes devant être indexés pour chaque document.

Q 2.2 Calculer les poids des termes pour chaque document selon la pondération tf.

Q 2.3 Calculer les poids des termes pour chaque document selon la pondération idf.

Q 2.4 Modéliser l'index et l'index inversé pour cette collection de documents en considérant la pondération $tf - idf$.

Q 2.5 Normaliser la pondération Calculer les normes de chaque vecteur document.

Q 2.6 Quels sont les documents retournés pour les requêtes suivantes :

- Q1 : sales home
- Q2 : july new

Dérouler le processus d'interrogation des index (index et index inversés) vous permettant de trouver les documents pertinents. Cela vous aidera pour l'implémentation.

Exercice 3 – Recherche efficace

On considère l'index suivant avec des postings sous forme de couples (document ID d, valeur w).

terme	postings				
a	(1,1)	(2,2)	(3,2)	(4,3)	(5,2)
b	(2,7)	(10,5)			

On considère que la question est "a b" et que les termes ont la même pondération pour la question, i.e.

$$s(q, d) = w_{ad} + w_{bd}$$

On suppose également qu'on cherche le top-1 (le document avec le plus haut score)

Q 3.1 Effectuer une recherche TAAT (Term At A Time) sans optimisation

Q 3.2 Effectuer une recherche TAAT (Term At A Time) sans optimisation

Q 3.3 Effectuer une recherche DAAT (Document At A Time) sans optimisation

Q 3.4 Effectuer une recherche DAAT (Document At A Time) avec l'algorithme WAND dont l'algorithme est donné ci-dessous

```
1. Function next(θ)
2.   repeat
3.     /* Sort the terms in non decreasing order of
        DID */
4.     sort(terms, posting)
5.     /* Find pivot term - the first one with accumulated
        UB ≥ θ */
6.     pTerm ← findPivotTerm(terms, θ)
7.     if (pTerm = null) return (NoMoreDocs)
8.     pivot ← posting[pTerm].DID
9.     if (pivot = lastID) return (NoMoreDocs)
10.    if (pivot ≤ curDoc)
11.      /* pivot has already been considered, advance
          one of the preceding terms */
12.      aterm ← pickTerm(terms[0..pTerm])
13.      posting[aterm] ← aterm.iterator.next(curDoc+1)
14.    else /* pivot > curDoc */
15.      if (posting[0].DID = pivot)
16.        /* Success, all terms preceding pTerm belong
            to the pivot */
17.        curDoc ← pivot
18.        return (curDoc, posting)
19.      else
20.        /* not enough mass yet on pivot, advance
            one of the preceding terms */
21.        aterm ← pickTerm(terms[0..pTerm])
22.        posting[aterm] ← aterm.iterator.next(pivot)
23.    end repeat
```

Exercice 4 – Exercice théorique

Q 4.1 On suppose que la présence d'un mot dans un document est le résultat d'un tirage aléatoire avec remise dans l'ensemble des M différents mots de la collection de documents (M = vocabulaire). Des observations ont montré que dans ce corpus le mot le plus courant revenait 5 000 000 fois et le dixième mot 500 000 fois. Soit X_k la variable aléatoire binaire correspondant au résultat de l'évènement E_k : le k-ième mot le plus fréquent a été tiré lors d'un tirage. Quelle est la loi associée à l'évènement E_k ?
Montrer que $P(X_k = 1) = \frac{1}{k \log M}$ – pour simplifier, on suppose que la somme d'une série harmonique $\sum_{i=1}^{r} \frac{1}{i}$ est approchée par $ln(r)$.

Q 4.2 Soit une collection de 1 millions de documents avec un nombre moyen de mots par document de 416, un vocabulaire de 757 476 mots pour un nombre total d'occurrences de 696 668 157. Quel serait le nombre moyen d'apparition du mot le plus fréquent dans un document de taille 416 ? Quelle est la loi de probabilité associée à ce calcul ?

Q 4.3 On considère que la taille de tous les documents dans la collection est 416. Combien de mots apparaissent au moins une fois (en espérance) dans tous ces documents ?