Πανεπιστήμιο Κρήτης, Τμήμα Επιστήμης Υπολογιστών
ΗΥ463 – Συστήματα Ανάκτησης Πληροφοριών
2009-2010 Φθινοπωρινό Εξάμηνο

Εργασία Μαθήματος
Αξία: 40% του τελικού σας βαθμού
Ανάθεση:
Παράδοση:

Σκοπός αυτής της άσκησης είναι να κατανοήσετε κάποιες βασικές έννοιες και τεχνικές, φτιάχνοντας εξ' αρχής ένα δικό σας Σύστημα Ανάκτησης Πληροφοριών. Για να δοκιμάσετε το σύστημα σας, μπορείτε να χρησιμοποιήσετε μια συλλογή εγγράφων (κειμένου και ιστοσελίδων) που θα σας δοθεί.

Μπορείτε να ακολουθήσετε τα εξής βήματα αφού διαβάσετε ολόκληρη την εκφώνηση της άσκησης:

ΦΑΣΗ Α

Ευρετηρίαση(Π1-Π5)

Π1) (5) Γράψτε ένα πρόγραμμα σε Java το οποίο να μπορεί να διαβάζει αρχεία κειμένου και να τυπώνει το πλήθος των διαφορετικών λέξεων και την κάθε διαφορετική λέξη συνοδευόμενη από το πλήθος εμφανίσεών της[1]. Το πρόγραμμα σας θα πρέπει να αγνοεί τις λέξεις αποκλεισμού (περιλαμβάνονται στο αρχείο stopwords.txt), τους αριθμούς και τα σημεία στίξης που πιθανόν να εμφανίζονται σε κάποια από τα έγγραφα της συλλογής.

Π2) (2) Επεκτείνετε το σύστημα ώστε να μπορεί να διαβάσει όχι μόνο ένα, αλλά πολλά αρχεία (π.χ. όσα βρίσκονται σε ένα συγκεκριμένο φάκελο του λειτουργικού συστήματος). Να μπορεί να κάνει ότι περιγράφεται στο (Π1) (τόσο για κάθε αρχείο ξεχωριστά, όσο και συγκεντρωτικά) για όλα τα αρχεία του φακέλου (ήτοι να μπορεί να μεταχειριστεί το σύνολο των αρχείων σαν να ήταν ένα μεγάλο αρχείο).

Π3) (4) Επεκτείνετε το σύστημα ώστε να δημιουργεί ένα φάκελο CollectionIndex στον οποίο να δημιουργεί ένα αρχείο με όνομα VocabularyFile.txt που θα καταγράφει όλες τις διαφορετικές λέξεις σε αύξουσα (λεξικογραφική) σειρά. Δίπλα σε κάθε λέξη να καταγράφεται το πλήθος των εγγράφων στα οποία εμφανίζεται (document frequency-df).

Π4) (5) Επεκτείνετε το σύστημα ώστε να δημιουργεί ένα ακόμη αρχείο με όνομα DocumentsFile.txt στον φάκελο CollectionIndex που θα καταγράφει για κάθε αρχείο της συλλογής μία τριάδα αποτελούμενη από ένα μοναδικό αριθμητικό αναγνωριστικό, το πλήρες μονοπάτι του αρχείου, και τον τύπο του αρχείου (txt, htm, jpg, png). Οι εγγραφές αυτές να είναι καταγεγραμμένες σε αύξουσα σειρά ως προς το αναγνωριστικό του εγγράφου.

Π5) (10) Επεκτείνετε το σύστημα ώστε να μπορείτε να κρατάτε πληροφορία σχετική με τα έγγραφα που εμφανίζεται η κάθε λέξη. Μπορείτε να ακολουθήσετε μία από τις δύο επιλογές που περιγράφονται παρακάτω.

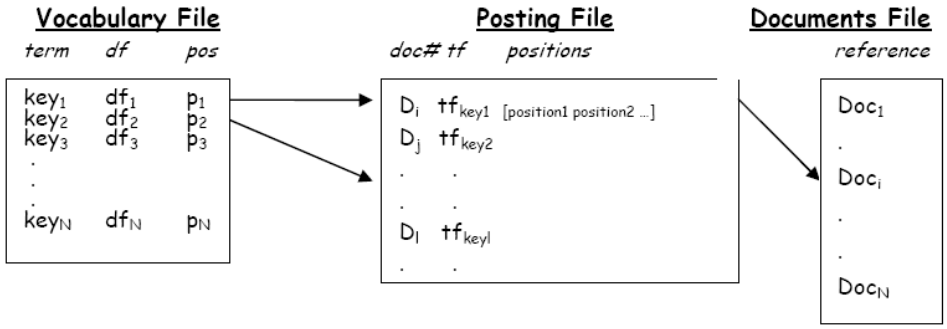
i) Μπορείτε να δημιουργείτε ένα αρχείο για κάθε λέξη (που εμφανίζεται στο VocabularyFile.txt) όπου για κάθε έγγραφο που εμφανίζεται η λέξη θα έχετε μία εγγραφή που θα περιέχει:
   a. το αναγνωριστικό του εγγράφου στο οποίο εμφανίζεται η λέξη αυτή.
   b. το tf της λέξης στο αντίστοιχο έγγραφο
   c. τις θέσεις εμφάνισης της λέξης στο έγγραφο αυτό

[1] Μπορείτε να δείτε (ή να θυμηθείτε) τα παραδείγματα που υπάρχουν στο http://www.csd.uoc.gr/~hy252/Lectures07/pdf/CS252CollectionClassesInterfaces07.pdf

d. ένα δείκτη προς τις αντίστοιχες πληροφορίες του συγκεκριμένου εγγράφου στο DocumentsFile.txt.

Τα αρχεία αυτά θα δημιουργούνται στο φάκελο CollectionIndex/InvertedLists.

ii) Στην περίπτωση όπου επιθυμείτε να μην κρατάτε ένα αρχείο για κάθε λέξη, έχετε την δυνατότητα να χρησιμοποιήσετε ένα μόνο αρχείο (ας το πούμε PostingFile.txt) που θα περιέχει την παραπάνω πληροφορία για όλες τις λέξεις. Στην περίπτωση αυτή, θα χρειαστεί να καταγράφετε για κάθε λέξη t_i στο αρχείο VocabularyFile.txt άλλο έναν αριθμό ο οποίος θα περιγράφει τη θέση στο αρχείο PostingFile.txt από την οποία αρχίζουν τα στοιχεία που αφορούν τη λέξη t_i (πεδίο pos στο Vocabulary File στην Εικόνα 1). Για ευκολία, θεωρούμε ότι τα postings όλων των όρων μπορούν να κρατηθούν στην μνήμη πριν αυτά γραφούν στο Posting File στο τέλος της ευρετηρίασης (άρα δεν χρειάζεται δηλαδή να εφαρμόσετε partial indexing and merging όπως έχετε διδαχθεί στο μάθημα).



Εικόνα 1. *Το ανεστραμμένο ευρετήριο*

Αποτίμηση επερωτήσεων(Π6-Π10)

Π6) (5) Επεκτείνετε το σύστημα σας ώστε να μπορείτε να πραγματοποιήσετε αποτίμηση επερωτήσεων βάσει του ανεστραμμένου ευρετηρίου που έχετε κατασκευάσει. Με αυτόν τον τρόπο θα επιβεβαιώσετε ότι έχετε κατασκευάσει το ευρετήριο σας σωστά. Προς το παρόν δεν ενδιαφερόμαστε για την κατάταξη των εγγράφων (με βάσει κάποιο συγκεκριμένο μοντέλο ανάκτησης), απλώς θέλουμε να ανακτούμε τα έγγραφα που εμφανίζεται μία λέξη που δίνει ο χρήστης ως επερώτηση.

Πριν αρχίσετε την αποτίμηση επερωτήσεων, βεβαιωθείτε ότι έχετε δημιουργήσει τα απαραίτητα αρχεία στον φάκελο CollectionIndex. Στην συνέχεια, φορτώστε το λεξιλόγιο (Vocabulary File) στη μνήμη. Τα postings όμως κάθε όρου (τα αρχεία στον φάκελο Collection/InvertedIndex ή εάν έχετε μόνο ένα αρχείο το Posting File) καθώς και πληροφορίες σχετικά με τα έγγραφα της συλλογής (Documents File) θεωρούμε ότι έχουν μεγάλο μέγεθος και δεν πρέπει να κρατηθούν στην μνήμη, γι' αυτό και παραμένουν στα αντίστοιχα αρχεία στον δίσκο.

Επεκτείνετε το σύστημα που φτιάξατε έτσι ώστε:

Π7) (5) Να διαβάζει μία ή περισσότερες λέξεις (λογική έκφραση) από την κονσόλα και να τυπώνει την απάντηση ως προς το λογικό μοντέλο (boolean retrieval model).

Π8) (10) Να διαβάζει μία ή περισσότερες λέξεις από την κονσόλα (δηλαδή μια επερώτηση σε φυσική γλώσσα) και να τυπώνει την απάντηση ως προς το διανυσματικό μοντέλο (vector space model).

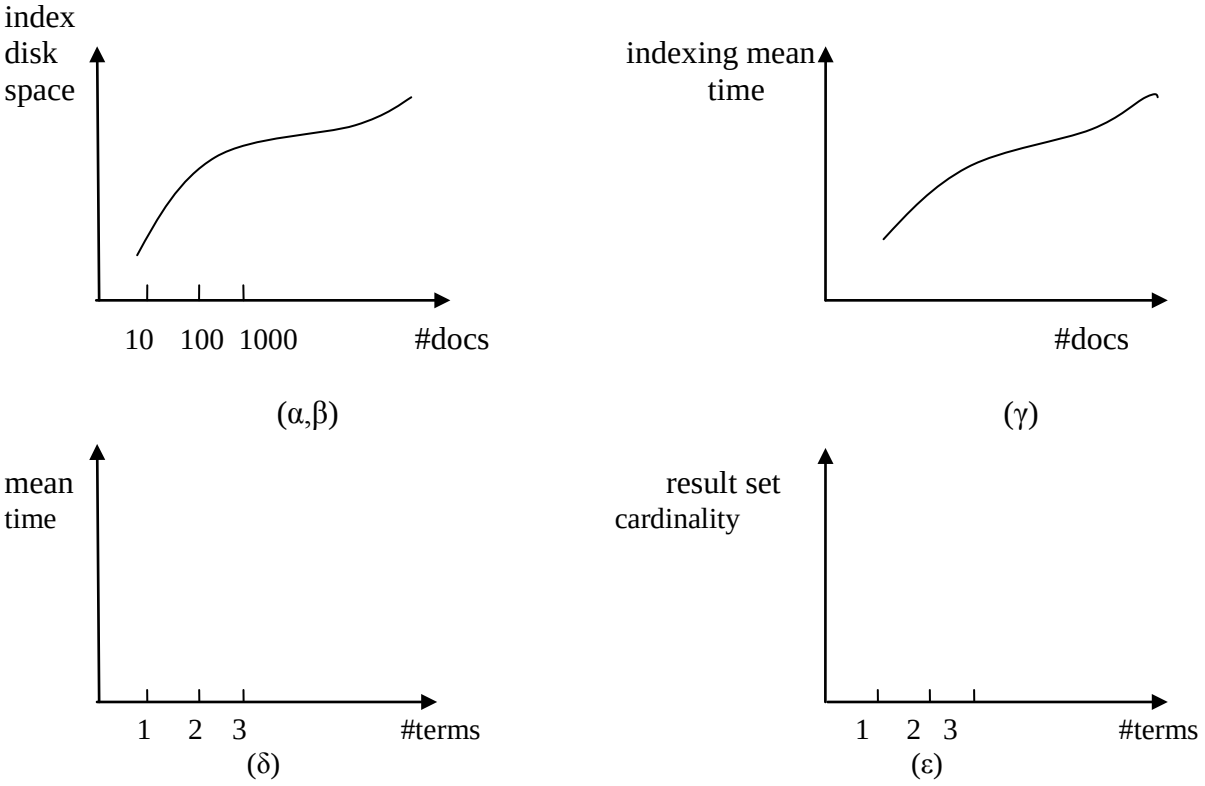
Π9) (5) Να κάνει ό,τι το (Π9) αλλά η απάντηση να υπολογίζεται βάσει του Okapi BM25[2].

Π10) (9) Συντάξτε γραπτή αναφορά με μετρήσεις από πειράματα που αφορούν την επίδοση του συστήματος (χωρικές απαιτήσεις – χρονικές αποκρίσεις). Η αναφορά σας πρέπει να περιέχει τις εξής γραφικές παραστάσεις: α) μία που να απεικονίζει το χώρο που καταλαμβάνει το ευρετήριο στο δίσκο

[2] http://en.wikipedia.org/wiki/Probabilistic_relevance_model_(BM25)

σε συνάρτηση με το πλήθος των εγγράφων καθώς και μία β) για τον ίδιο άξονα y σε συνάρτηση με το μέγεθος σε bytes των εγγράφων. Στη συνέχεια δώστε τα εξής γραφήματα: γ) Ένα που να απεικονίζει το συνολικό χρόνο ευρετηρίασης σε συνάρτηση με το πλήθος των εγγράφων, δ) ένα που να απεικονίζει το μέσο χρόνο απόκρισης σε συνάρτηση με το πλήθος των λέξεων της επερώτησης και ε) ένα που να απεικονίζει το μέσο πλήθος των αποτελεσμάτων σε συνάρτηση με το πλήθος των λέξεων της επερώτησης.

Για τον υπολογισμό του συνολικού μέσου χρόνου απόκρισης και μέσου πλήθους αποτελεσμάτων (δ,ε) να τεθούν τουλάχιστον 100 τυχαίες επερωτήσεις για κάθε πληθικότητα των όρων (100 επερωτήσεις με ενα όρο, 100 επερωτήσεις με δύο όρους κοκ) . Η επιλογή και ο συνδυασμός των όρων των επερωτήσεων να γίνεται από το VocabularyFile.txt με τυχαίο τρόπο.



Παρατηρήσεις:

- Αν είχατε υλοποιήσει στελέχωση και αποκλεισμό λέξεων, τότε εφαρμόστε αυτές τις λειτουργίες και στις επερωτήσεις (λέξεις εισόδου) του χρήστη.

- Καθώς ξεκινά το πρόγραμμα και αφού φορτωθεί το λεξιλόγιο στην μνήμη, ο χρήστης θα πρέπει να επιλέγει το μοντέλο ανάκτησης που επιθυμεί (ένα από τα 3 παραπάνω) και στην συνέχεια θα θέτει την επερώτηση του στο σύστημα. Το σύστημα σας θα επεξεργάζεται την επερώτηση και θα παρουσιάζει τα συναφή έγγραφα στον χρήστη.

- Τα συναφή έγγραφα θα πρέπει να τυπώνονται με φθίνουσα σειρά ως προς το σκορ τους (δηλ. το πιο συναφές έγγραφο τυπώνεται πρώτο, κ.ο.κ). Για κάθε έγγραφο θα τυπώνεται: το αναγνωριστικό του (docId), το όνομα του, καθώς και το σκορ που έλαβε βάσει του επιλεγμένου μοντέλου ανάκτησης.

Προαιρετικά
Στελέχωση κειμένου (stemming), διευθυνσιοδότηση «τμημάτων» (block addressing).

Χρονοδιάγραμμα
➔ Φάση Α: Π1-Π10 (27 Οκτώβρη- 17 Νοέμβρη)

- Παραδοτέα: Παραδώστε άνα αρχείο <<ΑΜ>>.zip το οποίο να περιέχει
  - /doc/report.{doc|pdf}: Γραπτή αναφορά
  - /src: Με τον κώδικας σας
  - /dist: Με ένα αρχείο <<ΑΜ>>.jar το οποίο πρέπει να επαρκεί για την εκτέλεση του συστήματος σας

Κάθε φάση θα εξεταστεί και θα βαθμολογηθεί ξεχωριστά.

Σχετικά με την παράδοση, είναι αποδεκτό η κάθε φάση να παραδοθεί μέχρι και 5 μέρες μετά από την ημερομηνία παράδοσης αλλά με 10% απώλειας του βαθμού ανά ημέρα.

Καλή εργασία

Παράρτημα

- Μπορείτε να χρησιμοποιήστε την κλάση java.util. StringTokenizer, για να κατακερματίσετε κάθε γραμμή του αρχείου σε λέξεις.

- Για ανάγνωση από αρχείο, μπορείτε να χρησιμοποιήστε τις κλάσεις java.io.BufferedReader και java.io.FileReader. Χρησιμοποιήστε το μηχανισμό των Exceptions για να γνωρίζετε πότε πρέπει να τυπώνονται κατάλληλα μηνύματα λάθους

- Για να γράψετε σε ένα αρχείο, μπορείτε να χρησιμοποιήστε τις κλάσεις PrintWriter και FileWriter. Όταν ολοκληρώσετε την εγγραφή σας, μην ξεχνάτε να καλέσετε τις μεθόδους flush και close (δείτε το API της Java για περισσότερες λεπτομέρειες).

Το παρακάτω πρόγραμμα δέχεται ως όρισμα ένα όνομα αρχείου και εκτυπώνει όλες τις λέξεις του αρχείου αυτού.

```
import java.io.*;
import java.util.*;

public class MyTokenizer {

	public static void main(String[] args){
		BufferedReader reader = null;
		StringTokenizer tokenizer = null;
		String delimiter = "\t\n\r\f";
		String line = null, currentToken = null;

		if(args.length == 0){
			System.err.println("You must give a filename as an argument");
			System.exit(1);
		}
		try {
			reader = new BufferedReader(new FileReader(args[0]));
			while ((line = reader.readLine()) != null){
				tokenizer = new StringTokenizer(line, delimiter);

				while(tokenizer.hasMoreTokens() ) {
					currentToken = tokenizer.nextToken();
					System.out.println(currentToken);
				}
			}
		}catch (FileNotFoundException e) {
			System.err.println("File "+args[0]+" not found.");
			System.exit(1);
		}
		catch (IOException e) {
			System.err.println("Error in reading file: " +args[0]);
			System.exit(1);
		}
	}
}
```