



Προγραμματιστική Άσκηση (MyIR)

Αξία: 15% του τελικού σας βαθμού

Ανάθεση: 13 Μαρτίου

Παράδοση: 3 Απριλίου (3 εβδομάδες)

Σκοπός αυτής της άσκησης είναι να κατανοήσετε κάποιες βασικές έννοιες και τεχνικές, φτιάχνοντας εξ' αρχής ένα δικό σας IR system. Για να δοκιμάσετε το σύστημα σας, μπορείτε να χρησιμοποιήσετε μια συλλογή εγγράφων (κειμένου) που θα σας δοθεί. Μπορείτε να ακολουθήσετε τα εξής βήματα αφού διαβάσετε ολόκληρη την εκφώνηση της άσκησης:

Π1) (1 μονάδα) 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 της¹. 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) (0.25 μονάδες) 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 αρχείο).

Π3) (0.75 μονάδες) 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).

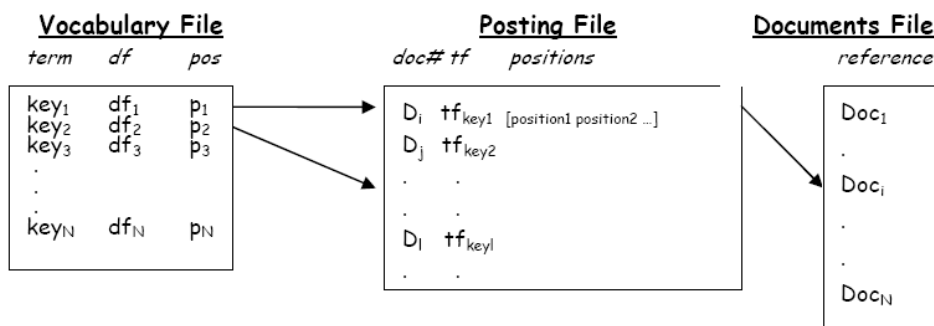
Π4) (1 μονάδα) Επεκτείνετε το σύστημα ώστε να δημιουργεί ένα ακόμη αρχείο με όνομα DocumentsFile.txt στον φάκελο CollectionIndex που θα καταγράφει για κάθε αρχείο της συλλογής ένα ζευγάρι αποτελούμενο από ένα μοναδικό αριθμητικό αναγνωριστικό και το πλήρες μονοπάτι του αρχείου. Οι εγγραφές αυτές να είναι καταγεγραμμένες σε αύξουσα σειρά ως προς το αναγνωριστικό του εγγράφου.

Π5) (2 μονάδες) Επεκτείνετε το σύστημα ώστε 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
 - d. ένα δείκτη προς τις αντίστοιχες πληροφορίες του συγκεκριμένου εγγράφου στο DocumentsFile.txt.Τα αρχεία αυτά θα δημιουργούνται στο φάκελο CollectionIndex/InvertedLists.

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

- 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

Π7) (1 μονάδα) 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) στην 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:

Π8) (1 μονάδα) Να διαβάζει μία ή περισσότερες λέξεις (λογική έκφραση) από την κονσόλα και να τυπώνει την απάντηση ως προς το λογικό μοντέλο (boolean retrieval model).

Π9) (2 μονάδες) 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).

Π10) (1 μονάδα) Να κάνει ό,τι το (Π9) αλλά η απάντηση να υπολογίζεται βάσει του Okapi BM25².

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

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

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.

Bonus (1 μονάδα)

Π11) (0.3 μονάδες) Δημιουργήστε μια γραφική διεπαφή χρήστη (GUI) για το σύστημα σας.

Π12) (0.7 μονάδες) Κάντε χρήση ελέγχων JUnit για τον έλεγχο ορθότητας του συστήματός σας.

Προαιρετικά

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

Παράρτημα

- Μπορείτε να χρησιμοποιήσετε την κλάση `java.util.StringTokenizer`, για να κατακερματίσετε κάθε γραμμή του αρχείου σε λέξεις.
- Για ανάγνωση από αρχείο, μπορείτε να χρησιμοποιήσετε 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 χρησιμοποιήσετε 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 αρχείο 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);
        }
    }
}
```