

Πανεπιστήμιο Κρήτης
Τμήμα Επιστήμης Υπολογιστών
ΗΥ463 Συστήματα Ανάκτησης Πληροφοριών
Εξάμηνο: Άνοιξη 2017

Γραπτή Αναφορά Έργου

Στοιχεία Φοιτητών

Μέλος	1 ^ο
Ονοματωπώνυμο	Αναστασάς Αναστάσιος
ΑΜ	3166
Email	csd3166@csd.uoc.gr

Μέλος	2 ^ο
Ονοματωπώνυμο	Γιακουμής Γιώργος
ΑΜ	3157
Email	csd3157@csd.uoc.gr

Πίνακας Περιεχομένων

<u>1</u>	<u>ΕΙΣΑΓΩΓΗ</u>	<u>3</u>
<u>2</u>	<u>< ΥΛΟΠΟΙΗΣΗ ></u>	<u>3</u>
2.1	< ΑΝΑΛΥΤΙΚΗ ΠΕΡΙΓΡΑΦΗ >	3
2.2	< ΤΡΟΠΟΣ ΥΛΟΠΟΙΗΣΗΣ >	4
2.3	< ΑΠΑΙΤΗΣΕΙΣ ΠΡΙΝ ΤΗΝ ΕΚΤΕΛΕΣΗ >	4
2.3.1	< INDEXER >	4
2.3.2	< SEARCHER >	4
2.4	< ΤΡΟΠΟΣ ΕΚΤΕΛΕΣΗΣ >	5
<u>3</u>	<u>ΜΕΤΡΗΣΕΙΣ</u>	<u>6</u>
<u>4</u>	<u>ΕΠΙΛΟΓΟΣ</u>	<u>7</u>
<u>5</u>	<u>ΑΝΑΦΟΡΕΣ</u>	<u>7</u>

1 Εισαγωγή

Το project χωρίζετε σε δύο προγράμματα. Το 1^ο πρόγραμμα είναι το “Indexer.jar” με το οποίο ευρετηριάζουμε μια συλλογή και παράγουμε 3 αρχεία (Vocabulary, Posting, Documents). Δημιουργήσαμε ένα αρχείο για να κρατάμε τις πληροφορίες για κάθε λέξη (2^η μέθοδος). Αυτά τα 3 αρχεία συνδέονται μεταξύ τους μέσω δεικτών που αποθηκεύονται στο εκάστοτε αρχείο από αριστερά προς τα δεξιά όπως αναφέρεται στην εκφώνηση. Στην συνέχεια με το 2^ο πρόγραμμα το “Searcher.jar” μπορούμε να κάνουμε κάποιες ερωτήσεις και να βρούμε εκείνα τα έγγραφα που είναι σχετικότερα με την ερώτηση μας.

2 < Υλοποίηση >

2.1 < Αναλυτική Περιγραφή >

Έχουμε υλοποιήσει 2 UI, ένα για τον indexer και ένα για τον searcher.

Τρέχοντας τον indexer πρέπει να έχουν τοποθετηθεί κάποιοι φάκελοι στο ίδιο path με το .jar θα αναφερθούμε παρακάτω αναλυτικότερα. Στη συνέχεια επιλέγουμε τον φάκελο ή το αρχείο που θέλουμε να κάνουμε indexing. Αφού δώσουμε την εντολή να αρχίσει το indexing τότε φορτώνουμε τα stopwords τα οποία τα κρατάμε σε ένα String variable και αρχίζουμε να διαβάζουμε σειριακά τα 8 tags του αρχείου. Το κείμενο ή την δομή με κείμενα που θα πάρουμε τις φιλτράρουμε ώστε να πάρουμε κάθε λέξη του κειμένου.

Τα φιλτραρίσματα είναι:

- 1) Το κείμενο θα χωριστεί με βάσει τα σημεία στίξης που έχουν οριστεί.
- 2) Να γίνουν όλοι οι χαρακτήρες της λέξης μικρά γράμματα και να μην υπάρχουν κεφαλαία γράμματα.
- 3) Εάν η λέξη αυτή είναι σύνολο των stopwords τότε εξαιρείται και προχωράμε στην επόμενη.
- 4) Κάνουμε stemming στην λέξη αυτή

Αφού περάσει το φιλτράρισμα αυτό η λέξη τότε την κρατάμε σε μια δομή, εάν δεν υπήρχε τότε είναι ένα νέο entry, εάν υπήρχε τότε ενημερώνονται κατάλληλα κάποια πεδία. Επίσης κρατάμε οποιαδήποτε έξτρα πληροφορία μπορεί να μας φανεί χρήσιμη αργότερα. Αφού τοποθετήσουμε και την τελευταία λέξη και από το τελευταίο φάκελο υπολογίζουμε το tf της εκάστοτε λέξης. Το επόμενο στάδιο είναι να γράψουμε το vocabulary και το posting file. Όσο γράφεται το posting υπολογίζουμε και τις νόρμες (για κάθε λέξη και για κάθε έγγραφο που βρέθηκε υπολογίζουμε την νόρμα και την κρατάμε σε μια δομή) Αφού ολοκληρωθεί αυτό γράφουμε και το Documents File (που χρειάζεται τις νόρμες).



Τρέχοντας τον searcher αρχικά πρέπει να υπάρχουν 2 φάκελοι στο ίδιο path με το searcher (λεπτομέρειες στο κεφ. 2.3). Μόλις φορτωθεί το vocabulary τότε μπορούμε να αναζητήσουμε διάφορες λέξεις. Έχουμε γενικά 4 τύπους για αναζήτηση None, Diagnosis, Test, Treatment. Με βάσει αυτόν τον τύπο εκτός του None το σύστημα μας μπορεί να βρει έγγραφα που σχετίζονται με διάγνωση, εξέταση και θεραπεία. Όταν ο τύπος είναι none τότε βρίσκουμε όσα έγγραφα είναι σχετικά με το query. Τέλος εμφανίζονται οι απαντήσεις από την πιο σχετική στην λιγότερο πιο σχετική με την ερώτηση. Συνολικά θα εμφανιστούν το πολύ 1000 αποτελέσματα εάν υπάρχουν (για λόγους ταχύτητας). Σε κάθε αποτέλεσμα εμφανίζεται το σχετικό αρχείο, το score και ένα snippet που περιέχει την σχετική λέξη του query. Πάνω από τα αποτελέσματα εμφανίζεται ο χρόνος που πήρε στο σύστημα μας να βρει όλες τις απαντήσεις και πόσα συνολικά έγγραφα έχουμε βρει.

2.2 < Τρόπος υλοποίησης >




Ουσιαστικά το project υλοποιήθηκε σε συνεργασία και των δύο μελών του. Στην αρχή το project «έσπασε» σε δυο τμήματα. Το 1^ο τμήμα ήταν η γραφική διεπαφή και το 2^ο τμήμα ήταν η υλοποίηση των βημάτων B1,B2,B3 και των βασικών δομών που χρειαζόντουσαν εκείνη την στιγμή ώστε να υλοποιηθούν αυτά τα 3 ερωτήματα. Επομένως το 1^ο μέλος Τάσος Αναστασάς ανέλαβε την γραφική διεπαφή κυρίως τα αρχεία IndexingGui.java, SearchingGui.java. Το 2^ο μέλος Γιώργος Γιακουμής έκανε την βασική υλοποίηση του TextAnalyzer.java καθώς και τα αρχεία Delimiter.java, Stopwords.java. Επίσης οι δομές που είχαν γραφτεί στην αρχή από το 2^ο μέλος ώστε να κρατηθούν τα δεδομένα είναι ενσωματωμένες στα αρχεία Vocabulary.java και στο Occurrences.java. Στη συνέχεια αφού ενώθηκαν τα 2 κομμάτια η υλοποίηση του project προχώρησε ομαδικά, επειδή παρατηρήσαμε ότι το project ήταν δύσκολο να χωριστεί σε 2 κομμάτια επομένως αποφασίσαμε να δουλέψουμε είτε ταυτόχρονα για να καταφέρουμε να υλοποιήσουμε και να κατανοήσουμε κάποιες από τις λειτουργίες του project είτε εργαζόταν τότε ο ένας (όταν ο άλλος δεν μπορούσε) και τότε ο άλλος (όταν ο ένας δεν μπορούσε). Έγινε χρήση ενός git repository όπου ο καθένας ανέβαζε τις αλλαγές που έκανε με κατάλληλα commit και έτσι ήταν πολύ ευκολότερο για το άλλο μέλος να πάρει, να μελετήσει και να συνεχίσει την υλοποίηση μετά τις τελευταίες αλλαγές.

2.3 < Απαιτήσεις πριν την εκτέλεση >

2.3.1 < Indexer >

Πριν τρέξουμε το  Indexer.jar πρέπει να τοποθετήσουμε στον ίδιο φάκελο που βρίσκεται και το .jar τον φάκελο  5_Resources_Stoplists όπου εκεί μέσα θα έχουμε τοποθετήσει τα stopwords αρχεία μας. Στην συνέχεια μπορούμε να ξεκινήσουμε το πρόγραμμα μας. Όταν ανοίξει η γραφική διεπαφή τότε αφού επιλέξουμε την συλλογή που θέλουμε να ευρετηριάσουμε πατάμε start indexing και ξεκινάμε ...

2.3.2 < Searcher >

Όμοια και εδώ πριν τρέξουμε το  Searcher.jar πρέπει σαφώς να έχουμε τρέξει τον indexer ο οποίος θα μας δημιουργήσει στον φάκελο που βρίσκεται το indexer.jar τον φάκελο  CollectionIndex ο οποίος θα περιέχει εσωτερικά τα αρχεία στα δεξιά μας. Όπως και στον indexer και εδώ θα χρειαστούμε τον φάκελο  5_Resources_Stoplists ο οποίος θα περιέχει μέσα τα stopwords. Στην συνέχεια μπορούμε να τρέξουμε το searcher.jar και να αναζητήσουμε ότι θέλουμε.



Documents
File.txt



PostingFile.
txt



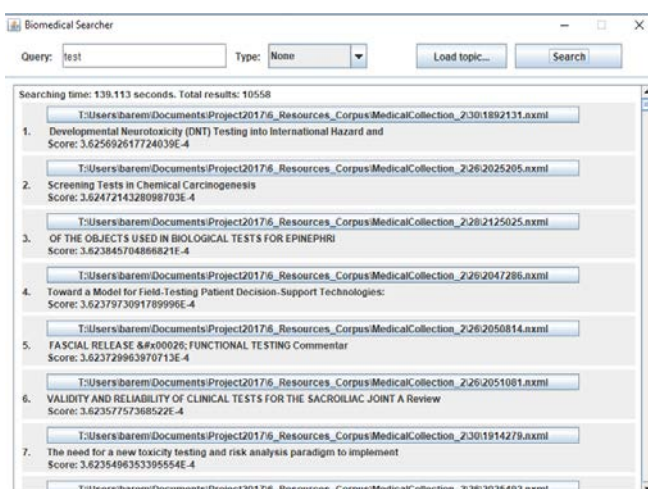
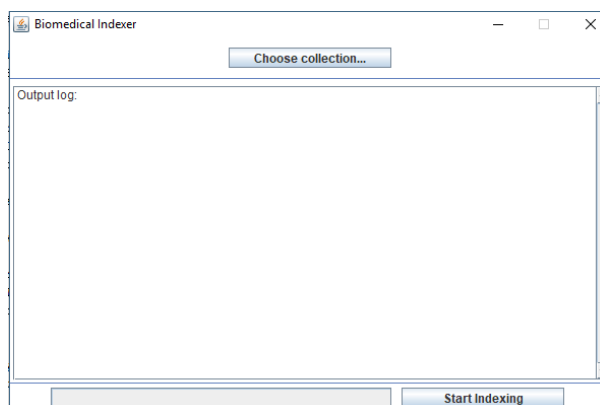
Vocabulary
File.txt

2.4 < Τρόπος εκτέλεσης >

- Εάν θέλουμε να κάνουμε indexing κάποια συλλογή μεγάλη (περισσότερα από 8 φακέλους από την MedicalCollection που βρίσκεται στο moodle) θα ήταν καλό να δώσουμε έξτρα RAM memory στο σύστημα μας. Επομένως τρέχαμε το indexer.jar αφού είχαμε κάνει τα βήματα που αναφέρονται στο 2.3.1

Java -jar ./Indexer.jar -Xms6g -Xmx6g

Με το Xms, Xmx δηλώνουμε ελάχιστη και μέγιστη μνήμη αντίστοιχα.

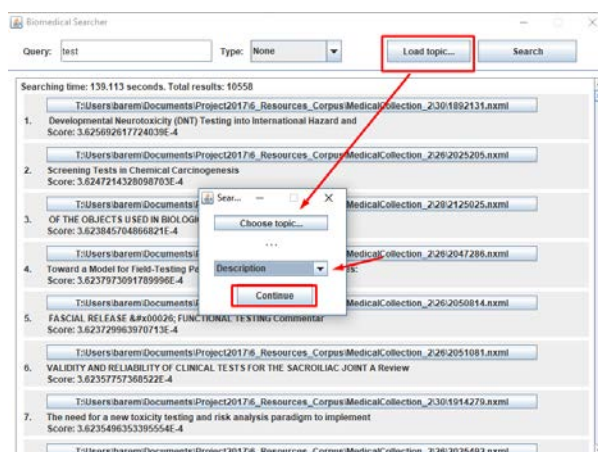


- Το searcher.jar απλά το τρέχουμε ως εξής:

Java -jar ./Searcher.jar

Επίσης πρέπει να είμαστε σίγουροι ότι στον ίδιο φάκελο υπάρχουν και οι φάκελοι που αναφέρονται στο 2.3.2. Παρακάτω βλέπουμε μια εκτέλεση τις ερώτησης «test». Στο score παρατηρούμε ότι σε κάποια αποτελέσματα αναγράφεται στο τέλος E-4. Αυτό σημαίνει ότι ο αριθμός που βλέπουμε είχε στην αρχή 4 μηδενικά.

Εάν θέλουμε να φορτώσουμε το αρχείο topic.xml πατάμε το κουμπί Load topic. Στην συνέχεια επιλέγουμε τον φάκελο ή το αρχείο topic.xml (ΠΡΟΣΟΧΗ! Εάν επιλέξουμε φάκελο τότε πρέπει μέσα στον φάκελο να υπάρχει το αρχείο με το όνομα topic.xml.) Επιλέγουμε από πια κατηγορία (description, summary) θέλουμε να κάνουμε τα query μας και πατάμε continue. Τα αποτελέσματα αποθηκεύονται στον φάκελο που βρίσκεται το topic.xml και με όνομα results.txt



3 Μετρήσεις

Indexing 1:

Collection: οι φάκελοι από την συλλογή που κατεβάσαμε από το moodle (μικρή)
00,02,04,26,28,30

Collection analyzed σε:
361,995 δευτερόλεπτα = 6 λεπτά

Distinct words:
582,797 λέξεις

Calculate Norms, store Vocabulary & Posting files σε:
3.8301,533 δευτερόλεπτα = 638,35 λεπτά = 10,6 ώρες

Store Documents file σε:
0,06 δευτερόλεπτα

Indexing 2:

Collection: οι φάκελοι από την συλλογή που κατεβάσαμε από το moodle (μικρή)
07,08,09,10,11,12,13,14,15,16

Collection analyzed σε:
712,698 δευτερόλεπτα = 11,9 λεπτά

Distinct words:
618,962 λέξεις

Calculate Norms, store Vocabulary & Posting files σε:
49.119,267 δευτερόλεπτα = 818,7 λεπτά = 13,6 ώρες

Store Documents file σε:
0,46 δευτερόλεπτα

Searching: Στο collection του indexing 1.

Query: 64-year-old obese female with diagnosis of diabetes mellitus and persistently elevated HbA1c. She is reluctant to see a nutritionist and is not compliant with her diabetes medication or exercise. She complains of a painful skin lesion on the left lower leg. She has tried using topical lotions and creams but the lesion has increased in size and is now oozing.

Type: Diagnosis

Total results: 70.431

Time: 867,84 δευτερόλεπτα = 14,4 λεπτά

Searching: Στο collection του indexing 1. (Results 1)

Query: topics.xml

Subject: Summary

Time: 6 ώρες και 30 λεπτά

Searching: Στο collection του indexing 2. (Results 2)

Query: topics.xml

Subject: Summary

Time: 7 ώρες και 30 λεπτά

4 Επίλογος

Γενικά υλοποιήθηκαν όλα τα ερωτήματα της εκφώνησης. Λόγο περιορισμένου χρόνου (και περιορισμένης πρόσβασης στο internet) δεν έγιναν πολλές βελτιστοποιήσεις. Κυριώς ο indexer θα μπορούσε να βελτιστοποιηθεί ώστε να εξοικονομεί περισσότερη μνήμη και να τρέχει πιο γρήγορα.

5 Αναφορές

- [1] <http://stackoverflow.com/>, 2017
- [2] <https://www.tutorialspoint.com/>, 2017
- [3] <https://www.google.gr/>, 2017
- [4] <http://www.csd.uoc.gr/~hy252/>, Lectures, 2007, 2012
- [5] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, Modern Information Retrieval: The Concepts and Technology behind Search (2nd Edition)
- [6] Γιάννης Τζιτζικας, Μια γεύση από ... Java, 2016