Ανάκληση Πληροφορίας CE634 Project

**Αρχικοποίηση προγράμματος

Χρησιμοποίησα Matlab2012b με ενσωματωμένο το 'Statistics and Machine Learning Toolbox'.
Το κύριο πρόγραμμα που τρέχουμε είναι το proj.m
Στο DataHasher.java γραμμες 45 και 58 βάζουμε το path για το Desktop.
Το μεταγλωτίζουμε με την εντολή 'javac -source 1.6 -target 1.6 DataHasher.java'
καθώς το Matlab2012b χρησιμοποιεί την εκδωση Java 1.6.0_17. Για να μπορούμε να
τρέξουμε μέσω της Matlab κώδικα Java, γράφουμε στο workspace της Matlab,
'which classpath.txt' πηγαινουμε στο αρχείο και στο τέλος προσθέτουμε
το path μέχρι το directory που είναι το DataHasher.class

**Περιγραφή κώδικα:

Έχουμε 2 πιθανές εισόδους, είτε από το restaurant.csv είτε
από αρχείο *.warc.wet (http://commoncrawl.org/the-data/get-started/).
Στην πρώτη περίπτωση διαβάζω το .csv και τοποθετω τα δεδομένα σε ένα
πίνακα. Αφαιρώ ότι δεν είναι κείμενο (πχ: . / -) και δεν κάνω stemming
από τη στιγμή που δεν χρειάζεται. Έπειτα,για 64bit hash, καλώ τη συνάρτηση
hash_word στην οποία γίνονται τα εξής: καλόυμε το πρόγραμμα java DataHaser, το
οποίο παίρνει τη λέξη και γράφει το hash της στο αρχείο 'arxeio.txt'.Μετά η
hash_word διαβάζει απο τα αρχέιο και το μετατρέπει σε 64bit vector και γυρνάει.
(Δεν υπάρχει τρόπος στη Matlab2012b να επιστρέψεις το αποτέλεσμα ενος προγράμματος
Java, για αυτό το έκανα έτσι).
Για 32bit hash, καλώ την string2hash και μετατρέπω κατάλληλα το αποτέλεσμα σε
32bit vector. Για κάθε εγγραφή προσθέτω τα 6 hash(1 για κάθε κελί με πληροφορία)
και αφού μετατρέψω σε 1 τα θετικα στοιχεία και σε 0 τα αρνητικά του hash,
καταληγω σε ένα πίνακα όπου σε κάθε γραμμή έχει το 64bit ή 32bit fingerprint

της αντίστοιχης εγγραφής. Έτσι τελείωσε το simhash.
Μετά διαλέγω ποιό fingerprint θα συγκρίνω και δημιουργώ πίνακες τους οποιους τους κάνω shift κατά διαφορετικό αριθμό τον καθένα(για να έχουμε τους permutated tables). Τους κάνω sorting και μετά έχω υλοποιήσει binary και interpolation search με σκοπό να συγκρίνουμε τα πρώτα bits του permutated fingerprint με κάθε πίνακα. Αν είναι ίσα συγκρίνουμε όλα τα bits και αν η hamming distance είναι μικροτερη απο αυτή που έχουμε ορίσει, έχουμε διπλότυπο.

**Αποτελέσματα δοκιμών

- 32/64 bit hash: Έχει σημασία γιατί όταν έχουμε λίγες σελίδες να συγκρίνουμε (πχ 2^10) τότε όσο περισσότερα bits hash έχουμε, τόσο πιο αραιούς άσσους έχει μέσα το διάνυσμα hash. Με αποτέλεσμα στους πίνακες να υπάρχουν πολλά μηδενικά και να βγαίνει αρκετά hashes ότι είναι ίσα.

- αριθμός πινάκων: Όπως είναι φυσικό υπάρχει tradeoff μεταξύ χώρου(πολλοί πίνακες->γρήγορο) και ταχύτητας(λίγοι πίνακες->αργό) καθώς όσους λιγότεροι ειναι οι πίνακες τόσο περισσότερα permutated fingerpritnts πρέπει να ελέγξουμε.

- KAPPA(κατώφλι ομοιότητας): Για μικρές τιμές του Κ περιορίζουμε τη σύγκριση σε fingerprints που είναι 'πολύ ίδια' αλλά αυτό σημαίνει πως σελίδες που είναι παρόμοιες αλλά όχι 'πολύ ιδιες' δεν θα τις λάβει υπόψιν. Άρα χάνουμε σε recall. Αντίστοιχα αν έχουμε μεγάλο Κ τότε γυρνάμε πάρα πολλά αποτελέσματα που είναι 'λίγο ίδια' άρα γυριζούμε αρκετές σελίδες που δεν είναι διπλότυπες.