Κ23γ: Ανάπτυξη Λογισμικού για Αλγοριθμικά Προβλήματα Χειμερινό εξάμηνο 2016-17

1η Προγραμματιστική Εργασία Υλοποίηση του Locality Sensitive Hashing (LSH) στη γλώσσα C++

Ομάδα :

- Αγγελική Φελιμέγκα, ΑΜ: 1115201300192, mail: sdi1300192@di.uoa.gr

- Παγώνα Βούρου, ΑΜ: 1115201300254, mail: sdi1300254@di.uoa.gr

Τίτλος και Περιγραφή :

Το πρόγραμμα ονομάζεται “proj-phase1” και έχει υλοποιηθεί σε γλώσσα c++.

Το πρόγραμμα δημιουργεί L(σε περίπτωση που ο χρήστης δεν το δώσει, δίνεται μέσα στο πρόγραμμα) πίνακες κατακερματισμου μέσα στους οποίους αποθηκεύονται τα σημεία που δίνονται από συγκεκριμένα αρχεία. Ανάλογα με το αρχείο που θα δώσει ο χρήστης επιλέγεται και μια συγκεκριμένη LSH μέθοδος(cosine, hamming, euclidean). Η κάθε μέθοδος δημιουργεί μια κατάλληλη συνάρτηση g, για κάθε σημείο(συνολικά k), η οποία άναλογα μ το αποτέλεσμα της αποθηκεύει το κάθε σημείο στο κατάλληλο bucket του πίνακα κατακερματισμού. Έπειτα, δίνοντας άλλα σημεία μέσα απο αρχεία, άναλογα με τη μέθοδο που επιλέγεται μέσα στο αρχείο, βρίσκονται οι κοντινότεροι γείτονες με βάση την ακτίνα, και έπειτα ο κοντινότερος γείτονας που βρίσκεται από τον LSH. Τα αποτελέσματα εκτυπώνονται σε αρχείο εξόδου.

Κατάλογος Αρχείων/ Επικεφαλίδων :

- main.cpp : Δημιουργεί L πίνακες κατακερματισμού και διαβάζει το αρχείο που δίνεται στο πρόγραμμα(CreateHash). Ανάλογα με τη μέθοδο που διαβάζει στην πρώτη γραμμή του αρχείου πηγαίνει στην αντίστοιχη κλάση. Η κάθε κλάση δημιουργεί και επιστρέφει μια συνάρτηση g η οποία δείχνει σε ποιό bucket αποθηκεύεται το κάθε item. Στη συνέχεια, η συνάρτηση InsertIntoHashtable() της κλάσης Hashtable εισάγει το κάθε στοιχείο στον πίνακα κατακερματισμού(ανάλογα με τη συνάρτηση g του στοιχείου). Αφού αποθηκευτούν όλα τα στοιχεία, το πρόγραμμα δέχεται ένα δεύτερο αρχείο με άλλα στοιχεία. Με τη συνάρτηση QuerySearch(), η οποία δηλώνεται και υλοποιείται στη main, ανοίγει το αρχείο με ένα νέο σύνολο αντικειμένων. Για κάθε αντικείμενο βρίσκει τους γείτονες του ακτίνας R, αν έχει ακτίνα, και τον μοναδικό γείτονα απο το LSH. Εάν η ακτίνα είναι 0, βρίσκει μόνο τον κοντινότερο γείτονα με την ελάχιστη απόσταση.

- Hashtable.h : Αυτό το αρχείο επικεφαλίδας περιέχει την αρχικοποίηση και τον ορισμό του πίνακα κατακερματισμού. Ο πίνακας κατακερματισμού αποτελείται απο αντικείμενα την Συνδεδεμένης Λίστας (Linked List).

- Hashtable.cpp : Σε αυτό το αρχείο εκτός από τους constructors, τον destructor του πίνακα, και την συνάρτηση εκτύπωσης του, printTable(), υλοποιούνται και δύο άλλες συναρτήσεις, η InsertIntoHashtable() , και η SearchBucket(). Η InsertIntoHashtable() ανάλογα με τη μέθοδο και τη g , εισάγει τα στοιχεία στο κατάλληλο bucket. Έχει γίνει διαχωρισμός στον τρόπο εισαγωγής των στοιχείων από Hamming,Cosine και DistanceMatrix από την Euclidean. Η υλοποιήση αυτή έγινε διότι οι τρεις πρώτες επιστρέφουν g, ενώ η Euclidean επιστρέφει την συνάρτηση fi που αποτελείται απο τις g. Η SearchBucket() ανάλογα με τη μέθοδο, Hamming,Cosine και DistanceMatrix, ή Euclidean καλεί τις κατάλληλες συναρτήσεις για τον εντοπισμό των κοντινότερων γειτόνων ακτίνας και τον κοντινότερο γείτονα με την μικρότερη απόσταση.

-LinkedList.h : Αυτό το αρχείο επικεφαλίδας περιέχει την αρχικοποίηση και τον ορισμό της συνδεδεμένης λίστας που χρησιμοποιείται για την υλοποίηση του πίνακα κατακερματισμού.

-LinkedList.cpp : Σε αυτό το αρχείο εκτός από τους constructors, τον destructor της λίστας και την συνάρτηση εκτύπωσης της ,printList(), υλοποιούνται και δύο άλλες συναρτήσεις, η Search() και η NN_Search(). Η Search ανάλογα με τη μέθοδο που το γράφει το αρχείο, και που της έχει στείλει η SearchBucket του πίνακα κατακερματισμού, ψάχνει και εκτυπώνει τους κοντινότερους γείτονες ακτίνας R. Η κάθε μέθοδος, ανάλογα με τον ορισμό της, έχει και διαφορετικο υπολογισμό των κοντινότερων γειτόνων. Η NN_Search(), αντίστοιχα, ανάλογα με τη μέθοδο που το γράφει το αρχείο, και που της έχει στείλει η SearchBucket του πίνακα κατακερματισμού, ψάχνει και επιστρέφει τον (έναν) κοντινότερο γείτονα, αυτόν με τη μικρότερη απόσταση.Η κάθε μέθοδος, ανάλογα με τον ορισμό της, έχει και διαφορετικο υπολογισμό του κοντινότερου γείτονα.

- CosineSim.h : Δήλωση/αρχικοποίηση της κλάσης CosineSim και των μεθόδων της.

- CosineSim.cpp : Υλοποίηση της κλάσης CosineSim και των μεθόδων της. Η CosineSim λειτουργεί με βάση την ομοιότητα συνιμητόνου. Περιέχει τους constructors, και τον destructor της κλάσης καθώς και την μέθοδο ConstructGFunctionC(int k), για την δημιουργία της συνάρτησης g. Μέσα σε αυτή τη συνάρτηση υπολογίζεται το εσωτερικό γινόμενο ενός τυχαίου αριθμού με τις συντεταγμένες του αντικειμένου και ανάλογα ,αν το γινόμενο είναι θετικό ή αρνητικό δημιουργούμε μία συμβολοσειρά από παράθεση τιμών 0 και 1, η οποία και επιστρέφεται.

-DistanceMatrix.cpp : Δήλωση/αρχικοποίηση της κλάσης DistanceMatrix και των μεθόδων της.

-DistanceMatrix.h : Υλοποίηση της κλάση DistanceMatrix και των μεθόδων της. Περιέχει τους constructors, και τον destructor της κλάσης καθώς και την μέθοδο ConstructGFunctionC(int item,int k). Στον constructor της κλάσης αυτής διαβάζεται το αρχείο και εισάγωνται τα στοιχεία σε έναν πίνακα. Έπειτα στην συνάρτηση κατασκευής της g, με τις κατάλληλες μαθηματικές πράξεις δημιουργείται το αποτέλεσμα της g (παράθεση τιμών 0 και1) και επιστρέφεται.

-Euclidean.h : Δήλωση/αρχικοποίηση της κλάσης Euclidean και των μεθόδων της.

-Euclidean.cpp : Υλοποίηση της κλάσης Euclidean και των μεθόδων της. Περιέχει τους constructors, και τον destructor της κλάσης καθώς και την μέθοδο ConstructFiFunctionC(int item,int k), όπου κατασκευάζεται η fi. Μέσα στη συνάρτηση αυτή, αφου αποθηκευτούν οι διαστάσεις του σημείου σε έναν πίνακα, και μετά απο συγκεκριμένες μαθηματικές πράξεις επιστρέφεται το αποτέλεσμα της fi, το οποίο είναι ένας ακέραιος αριθμός.

-Hamming.h : Δήλωση/αρχικοποίηση της κλάσης Hamming και των μεθόδων της.

-Hamming.cpp : Υλοποίηση της κλάσης Hamming και των μεθόδων της. Περιέχει τους constructors, και τον destructor της κλάσης καθώς και την μέθοδο ConstructGFunctionC(int item,int k), όπου κατασκευάζεται η g (συμβολοσειρά από παράθεση τιμών 0 και 1) και επιστρέφεται.

-NeighbourSearch.h : Δήλωση/ αρχικοποίηση των συναρτήσεων αναζήτησης του κοντινότερου γείτονα, RangeNeighbourSearch και Nearest_Neighbor_Search.

-NeighbourSearch.cpp : Υλοποίηση των συναρτήσεων RangeNeighbourSearch και Nearest_Neighbor_Search.

RangeNeighbourSearch : Ανάλογα με τη μέθοδο, καλείται η κατασκευάστρια της g, της οποίας το αποτέλεσμα, μαζί με την ακτίνα που διαβάζεται απο το αρχείο, στέλνεται σαν όρισμα στη Searchbucket για την έυρεση των κοντινότερων γειτόνων ακτίνας R.

Nearest_Neighbor_Search : Ανάλογα με τη μέθοδο, καλείται η κατασκευάστρια της g, της οποίας το αποτέλεσμα, στέλνεται σαν όρισμα στη Searchbucket για την έυρεση του κοντινότερου γειτόνα , με τη μικρότερη απόσταση.

-randomfunc.h : Δήλωση/ αρχικοποίηση των συναρτήσεων marsagliarandom(int j) και mod (int a, int b).

- randomfunc.cpp : Υλοποίηση των συναρτήσεων marsagliarandom(int j) και mod (int a, int b).

marsagliarandom(int j) : Γεννήτρια τυχαίων μεταβλητών με βάση την μέθοδο marsaglian.

mod (int a, int b): Συνάρτηση υπολογισμού υπολοίπου.

Οδηγίες μεταγλώττισης του προγράμματος :

μέσα στο φάκελο του πρότζεκτ κάνουμε make all, και από τη στιγμή που υπάρχει makefile το πρόγραμμα μετταγλωτίζεται.

Οδηγίες χρήσης του προγράμματος :

μετά τη μεταγλώττιση γράφουμε :

```
$./lsh –d <input file> –q <query file> –k <int> -L <int> -o <output file>
```

```
πχ: ./lsh 1 DataEuclidean.csv 3 QueryEuclidean.csv 5 4 7 4 9 QueryResults.txt
```