Εργασία 1

Ανάπτυξη Λογισμικού για Αλγοριθμικά Προβλήματα

Θεοφανόπουλος Μιχαήλ ΑΜ : 1115201800053

Δημησίανος Σπυρίδων ΑΜ : 1115201800042

Οργάνωση αρχείων και φακέλων:

Φάκελος Cluster: Περιέχει τα αρχεία mainCLUSTER.cpp και mainCLUSTER.h , τα οποία περιέχουν την main συναρτηση για την υλοποίηση του Clustering.

Φάκελος CUBE: Περιέχει τα αρχεία mainCUBE.cpp και mainCUBE.h , τα οποία περιέχουν την main συναρτηση για την υλοποίηση του HyperCube.

Φάκελος LSH : Περιέχει τα αρχεία mainLSH.cpp και mainLSH.h , τα οποία περιέχουν την main συναρτηση για την υλοποίηση του αλγορίθμου LSH.

Φάκελος brute : Περιέχει τα αρχεία bruteFunctionsCUBE.cpp και bruteFunctionsLSH.cpp , τα οποία περιέχουν τις υλοποιησεις των σηναρτησεων του HyperCube και αντιστοιχα του LSH με brute τροπο (brute έυρεση αποστασεων μέσα απο τα αρχεία των δεδομένων)

Φάκελος data : Περιέχει το αρχείο items.cpp , το οποίο περιέχει τις υλοποιήσεις των μεθόδων της κλάσης item.

Φάκελος src : Περιέχει τα αρχεία utilCLUSTER.cpp utilCUBE.cpp utilLSH.cpp , τα οποία περιεχουν το καθενα τις αντιστοιχες βοηθητικες συναρτησεις για την υλοποιηση του Clustering , HyperCube και του LSH.

Φάκελος structures : Περιέχει τα αρχεία Cluster.cpp, G.cpp, H.cpp, Hash.cpp, HashTable.cpp και HyperCube.cpp τα οποια περιεχουν τις υλοποιήσεις των μεθόδων των αντίστοιχων κλάσεων.

Φάκελος tests : Περιέχει τα αρχεία με τα δεδομένα της εργασίας (queries , dataset) οπως επίσης και το cluster.conf που χρησιμοποιείται για την εισαγωγή των δεδομένων που απαιτούνται για το Clustering.

Φάκελος headers : Περιεχει ολα τα Header files , εκτος των headers των main συναρτησεων που βρισκονται στους αντιστοιχους Cluster, Cube και LSH φακέλους.

Τελος εχουμε το **makefile** και το αρχειο **output.txt** για την συλλογή των εξόδων του προγράμματος.

Εκτέλεση του προγράμματος:

Η μεταγλώττιση του προγράμματος γίνεται με τη χρήση της εντολής:

> make

Η εκτέλεση του καθενός από τα τρία προγράμματα γίνεται με τη χρήση των εντολών:

> make out1, για την εκτέλεση του LSH.

> make out2, για την εκτέλεση του Hypercube.

> make out3, για την εκτέλεση του Clustering.

Αν δεν επιθυμείται η χρήση του makefile για την εκτέλεση των προγραμμάτων, το πρότυπο των ορισμάτων είναι ως εξής:

> ./outLSH -i dataset.txt -q query.txt -k [int] -L [int] -o output.txt -N [int] -r [int] για το LSH.

> ./outCUBE -i dataset.txt -q query.txt -k [int] -M [int] -probes [int] -o output.txt -N [int] -r [int] για το Hypercube.

> ./outCLUSTER -i dataset.txt -c cluster.conf -o output.txt complete(optional) -m [string] για το Clustering.

Περιγραφή των κλάσεων:

- Κλάση item, αντιπροσωπεύει ένα διάνυσμα και περιέχει ένα id(string) και έναν πίνακα από συντεταγμένες(double). Τα χαρακτηριστικά trick, flag χρησιμοποιούνται για τον αλγόριθμο LSH και Clustering αντίστοιχα.
- Κλάση Hi, αντιπροσωπεύει τις συναρτήσεις h που αναφέρονται στην εκφώνηση της εργασίας. Περιέχουν μια τιμή t(double) που ορίζεται μοναδική για κάθε ξεχωριστό Hi, μια τιμή w που είναι μοναδική για όλο το πρόγραμμα και έναν πίνακα v που είναι μοναδικός για κάθε ξεχωριστό Hi.
- Κλάση Gi, αντιπροσωπεύει τις συναρτήσεις g που αναφέρονται στην εκφώνηση της εργασίας. Περιέχουν μια τιμή k(int) που είναι μοναδικό για όλο το πρόγραμμα και αφορά τον αριθμό των h συναρτήσεων που θα χρησιμοποιήσει, μια τιμή w που επίσης είναι μοναδική για όλο το πρόγραμμα, έναν πίνακα από Hi συναρτήσεις και ένα vector από r(int32_t) που είναι μοναδικός για κάθε Gi.
- Κλάση HashTable και HashNode, είναι μια υλοποίησή μας ενός hash table.

- Κλάση Hash, αντιπροσωπεύει τη δομή που ζητείται για την υλοποίηση του αλγορίθμου του LSH. Περιέχει τις τιμές k,L,w που χρειαζόμαστε, το μέγεθος size του hash table, το dimension που είναι η διάσταση στην οποία βρίσκονται τα διανύσματα, έναν πίνακα από items, έναν πίνακα από hash tables και έναν πίνακα από Gi συναρτήσεις.

- Κλάση HyperCube, αντιπροσωπεύει τη δομή που ζητείται για την υλοποίηση του αλγορίθμου του Hypercube. Περιέχει τις τιμές k,w,M,probes που χρειαζόμαστε, το dimension που είναι η διάσταση στην οποία βρίσκονται τα διανύσματα, έναν πίνακα από items, έναν πίνακα fVector από unordered maps που χρησιμοποιούμε για τις τιμές f, έναν πίνακα nearVertices που περιέχει τις κορυφές, ένα hash table cube για την αποθήκευσει των items και τέλος έναν πίνακα hVector που περιέχει τις Hi συναρτήσεις.

- Κλάσεις Clustering και Cluster, αντιπροσωπεύουν τη δομή που ζητείται για την υλοποίηση του αλγορίθμου του Clustering. Περιέχουν όλες τις τιμές (K,L,kLSH,M,kCUBE,probes,w) που χρειαζόμαστε σύμφωνα με την εκφώνηση και ουσιαστικά το Clustering είναι η γενική δομή που περέχει έναν πίνακα από Clusters, καθώς και όλα τα items. Κάθε Cluster διατηρεί έναν δείκτη στον κεντροειδή του καθώς και το dimension των διανυσμάτων του και έναν πίνακα από τα διανύσματα που περιέχονται στο εκάστοτε cluster.

<u>Γενική προσέγγιση:</u>

Ο κώδικας περιέχει σχόλια με σκοπό να γίνεται εύκολη η διαδικασία ανάγνωσής του.

- *LSH,* αρχικά δημιουργούμε ένα Hash, διαβάζουμε ένα ένα τα διανύσματα από το αρχείο που δίνεται από τον χρήστη και δημιουργούμε **μοναδική φορά** ένα item για κάθε γραμμή του αρχείου και ύστερα γεμίζουμε τη δομή με τα items. Για το hashing των διανυσμάτων και την εισαγωγή τους στο κατάλληλα buckets των hash tables χρησιμοποιούνται τόσο οι συναρτήσεις h όσο και οι g με τον τρόπο που αναφέρεται στις διαφάνειες του μαθήματος. Στο τέλος καλείται η συνάρτηση answerQueries() η οποία διαβάζει ένα ένα τα διανύσματα από το αρχείο με τα queries και παράγει κατάλληλες απαντήσεις, σύμφωνα με την εκφώνηση, και τις γράφει στο αρχείο output.txt. Ύστερα αποδεσμεύει κατάλληλα όση μνήμη δεσμεύτηκε.

- *Hypercube,* η διαδικασία είναι παρόμοια με αυτή του LSH, με τη μόνη διαφόρα ότι αυτή τη φορά δημιουργούμε ένα HyperCube και για το hashing των items και την εισαγωγή τους χρησιμοποιούμε διαφορετική προσέγγιση από ότι στον LSH, όπως ακριβώς αναφέρεται στις διαφάνειες του μαθήματος. Στο τέλος πάλι αποδεσμεύουμε πάλι κατάλληλα όση μνήμη δεσμεύτηκε.

- *Clustering,* η προσέγγιση μας για αυτό το κομμάτι είναι επίσης παρόμοια με των προηγούμενων δύο όσων αφορά την αρχικοποίηση. Αυτή τη φορά καλείται η συνάρτηση functionality(), η οποία ανάλογα το method που δίνεται ως παράμετρος καλέι τη συναρτηση kMeansPP, την συνάρτηση Assign με το αντίστοιχο όρισμα και τέλος τη συνάρτηση Silhouette() για την “αξιολόγηση” της μεθόδους μας.

- Και για τα τρία προγράμματα, επιλέξαμε να ορίσουμε τη τιμή του w ως μια τυχαία τιμή στο διάστημα [2,6], κάθως έτσι αναφέρεται στις διαφάνειες του μαθήματος και παρατηρήσαμε ότι οι τιμές αυτές παράγουν σωστά αποτελέσματα για την υλοποίηση μας.
- Όλη η λειτουργία της εργασίας μας ακολουθεί πιστά τις οδηγίες που αναφέρονται στην εκφώνηση.

Ενδεικτικά αποτελέσματα Silhouette:

Παρακάτω σας παρουσιάζουμε ορισμένα ενδεικτικά αποτελέσματα της διαδικασίας του Silhouette για το αρχείο με τα 10 χιλιάδες entries που δόθηκε στο e-class. Τα προγράμματα έτρεξαν στα μηχανήματα Linux της σχολής.

- *Classic:*

```
Algorithm: Lloyds
CLUSTER-0 {size: 2543,centroid: 15.418797 8.607157
CLUSTER-1 {size: 2357,centroid: 29.992787 15.61349
CLUSTER-2 {size: 5100,centroid: 35.420588 21.40509
clustering_time: 6.000000
Silhouette: [0.129152,0.270853,0.420248,0.273418]
```

- *LSH:*

```
Algorithm: LSH
CLUSTER-0 {size: 2500,centroid: 15.194800 8.550000 6
CLUSTER-1 {size: 5100,centroid: 35.420588 21.405098
CLUSTER-2 {size: 2400,centroid: 29.965000 15.547500
clustering_time: 136.000000
Silhouette: [0.130641,0.359879,0.894580,0.461700]
```

- *Hypercube:*

```
Algorithm: Hypercube
CLUSTER-0 {size: 2497,centroid: 15.200240 8.535042 6
CLUSTER-1 {size: 5100,centroid: 35.420588 21.405098
CLUSTER-2 {size: 2403,centroid: 29.940907 15.554307
clustering_time: 63.000000
Silhouette: [0.130764,0.359862,0.893456,0.461361]
```

Η εργασία υλοποιήθηκε με την χρήση git. Το link του repository μας είναι:

https://github.com/miketheofan/Software-Development-for-Algorithmic-Problems

Σας ευχόμαστε καλή διόρθωση!