ΑΝΑΠΤΥΞΗ ΛΟΓΙΣΜΙΚΟΥ ΓΙΑ ΑΛΓΟΡΙΘΜΙΚΑ ΠΡΟΒΛΗΜΑΤΑ

Ομάδα 31

Ανάργυρος Γιαβρής 1115201600029

Ευστράτιος Ζωγραφάκης 1115201600049

Αναζήτηση και συσταδοποίηση διανυσμάτων στη C/C++

Περιγραφή Προγράμματος

LSH: Το πρόγραμμα αποθηκεύει τα δεδομένα του dataset σε L HashTables χρησιμοποιώντας ως hash-function τη συνάρτηση g, η οποία προκύπτει απο concatenation k h συναρτήσεων, στη συνέχεια διαβάζει το query file και ψάχνειστα HashTables για να βρει τον προσεγγιστικά κοντινότερο γείτονα της κάθε εικόνας, τους N προσεγγιστικά κοντινότερους γείτονες, τους γείτονες εντος ακτίνας R και τους πραγματικούς κοντινότερους γείτονες και εκτυπώνει τα αποτελέσματα σε αρχείο κειμένου.

Hypercube:Το πρόγραμμα αποθηκεύει τα δεδομένα του dataset σε HashTable χρησιμοποιώντας ως hash-function τη συνάρτηση p, η οποία μετατρέπει τις συναρτήσεις 0 h σε ακολουθίες απο 0 και 1 και αντιπροσωπεύουν τις κορυφές του υπερκύβου. Στη συνέχεια διαβάζει το query file και ψάχνει στο HashTable μέχρι M εικόνες και μέχρι probes διαφορετικές κορυφές του υπερκύβου για να βρει τον προσεγγιστικά κοντινότερο γείτονα της κάθε εικόνας,τους N προσεγγιστικά κοντινότερους γείτονες, τους γείτονες εντος ακτίνας R και τους πραγματικούς κοντινότερους γείτονες και εκτυπώνει τα αποτελέσματα σε αρχείο κειμένου.

Clustering:Το πρόγραμμα αρχικοποιεί τα κεντροειδή με την τεχνική k-Means++ και τα ενημερώνει με τον υπολογισμο του μέσου διανύσματος. Στη συνέχεια ανατίθονται στα clusters ανάλογα με τη μέθοδο που εχει επιλεξει ο χρήστης(Classic/LSH/Hypercube). Στην πρωτη περίπτωση χρησιμοποιείται ο αλγόριθμος Lloyd's, στην δεύτερη η αντίστροφη ανάθεση μέσω Range Search με LSH και στην τρίτη η αντίστροφη ανάθεση μέσω Range Search με τυχαία προβολή στο HyperCube.

Κατάλογος Αρχείων

Για το LSH έχουμε τα αρχεία: lsh.c ,interface.c, interface.h

- **Συναρτήσεις του lsh.c:**

lshTrain: Δέχεται τον πίνακα στον οποίο έχει γίνει η αποθήκευση των διανυσμάτων και τα διαμοιράζει στα L Hashtables βάση των συναρτήσεων h και g που βρισκονται στις διαφάνειες.

lshSearch: Δέχεται ένα query διάνυσμα και με βάση τα Hashtables που εχει αποθηκευτει το data set απαντά στα ερωτήματα του χρήστη, δηλαδη N κοντινότεροι γείτονες και γείτονες που υπάρχουν μέσα σε μια συγκεκριμένη ακτίνα R.

lsh: Δέχεται το αρχείο με τα queries που δίνει ο χρήστης και καλεί τις συναρτήσεις lshTrain και lshSearch και δημιουργεί τις κατάλληλες δομές δεδομένων. Επίσης ρωτάει τον χρήστη αν επιθυμεί να συνεχίσει και να εισάγει νέο query file ή να τερματίσει το πρόγραμμα.

main: Διαβάζει αν εχουν δοθει τα ορίσματα απο την γραμμή εντολών και αρχικοποιεί μεταβλητες που χρειάζονται για τους υπολογισμούς του lshTrain. Καλεί την συνάρτηση lsh.

- **Συναρτήσεις του interface.c:**

countFile: Μετράει πόσα διανύσματα περιέχει το αρχείο εισόδου. Επίσης μετράει πόσες διαστάσεις εχουν τα διανύσματά μας.

saveFile: Αποθηκεύει τα διανύσματα στις κατάλληλες δομές.

writeResults: Γραφει στο αρχείο outFile τα αποτελέσματα του προγράμματός μας σύμφωνα με τις προδιαγραφές που αναφέρονται στην εκφώνηση.

query_knn: Βρίσκει τις πραγματικές αποστάσεις με brute force ανάμεσα στα διανύσματα και τα queries.

eucmod: Πραγματοποιεί πράξη euclidean mod ώστε να αποφύγουμε αρνητικές τιμές στο g.

inner: Εσωτερικό γινόμενο δύο διανυσμάτων.

normalDistr: Παράγει τυχαία διανύσματα βάση της κανονικής κατάνομης.

euclideanDistance: Βρίσκει την ευκλείδια απόσταση μεταξυ δυο διανυσματων.

hash: Εισάγει τα διανυσματα στην κατάλληλη θέση του hashtable και αποθηκεύει τα id.

- **Συναρτήσεις του interface.h**

Στο αρχείο αυτο περιέχονται οι δηλώσεις των συναρτήσεων και των δομών δεδομένων.

Για το Hypercube έχουμε τα αρχεία: hyperCube.c ,interface.c, interface.h

- **Συναρτήσεις του hyperCube.c**

hammingDistance: Υπολογίζει την hamming distance μεταξύ δύο ακεραίων αριθμών.

cubeTrain: Δέχεται την δομή με τα vectors και τα αποθηκεύει κατάλληλα στον κύβο.

cubeSearch: Δέχεται ένα query και κοιτάζει σε max probes για N κοντινότερους γείτονες καθώς και για γείτονες σε απόσταση μικρότερη του radius.

cube: Καλέι τις συναρτήσεις cubeTrain και cubeSearch και αφού τελειώσει με το πρώτο query file ρωτάει τον χρήστη αν επιθυμεί να συνεχίσει και να εισάγει νέο query file ή να τερματίσει το πρόγραμμα.

main: Διαβάζει αν εχουν δοθει τα ορίσματα απο την γραμμή εντολών και αρχικοποιεί μεταβλητες που χρειάζονται για τους υπολογισμούς του cubeTrain. Καλεί την συνάρτηση cube.

Τα αρχεία interface.c και interface.h είναι τα ίδια με το LSH

Για το Cluster έχουμε τα αρχεία: cluster.c ,implementationCluster.c, methodCluster.c, interfaceCluster.h.

- **Συναρτήσεις του cluster.c**

main: Διαβάζει απο τη γραμμή εντολών και απο το config file και αρχικοποιεί τις δομές με βάση αυτα τα στοιχεία. Επίσης καλεί την κατάλληλη μέθοδο μεσω της οποίας θα γίνει το clustering.

- **Συναρτήσεις του implementationCluster.c**

vectorIsCentroid: Ελέγχει αν ενα διάνυσμα είναι κεντροειδές.

kmeansInit: Δέχεται τον πίνακα με τα διανύσματα και αρχικοποιεί τυχαία Κ κεντροειδή.

lloydAssignment: Δέχεται τον πίνακα με τα διανυσματα μας και τον πίνακα με τα κεντροειδή που προέκυψαν απο την kmeansInit και αναθέτει όλα τα διανύσματα στα κατάλληλα κεντροειδή. Στην συνέχεια καλείται κάθε φορά που γίνονται update τα κεντροειδή.

lshTrain: Είναι η ίδια με αυτη που χρησιμοποιούμε στον αλγόριθμο του LSH.

lshSearch: Είναι η ίδια με αυτη που χρησιμοποιούμε στον αλγόριθμο του LSH.

cubeTrain: Είναι η ίδια με αυτη που χρησιμοποιούμε στον αλγόριθμο του Hypercube.

cubeSearch: Είναι η ίδια με αυτη που χρησιμοποιούμε στον αλγόριθμο του Hypercube.

update: Η συνάρτηση δέχεται τον πινακα με τα διανυσματα και τον πινακα με τα υπαρχοντα κεντροειδη και στη συνεχεια κανει update τα κεντροειδή με τον υπολογισμο του μέσου διανύσματος.

silhouette: Υπολογίζεται ο δείκτης εσωτερικής αξιολόγησης της συσταδοποίησης.

- **Συναρτήσεις του methodCluster.c**

Είναι οι ίδιες συναρτήσεις που χρησιμοποιούνται στο interface.c του LSH.

- **Συναρτήσεις του interfaceCluster.h**

Στο αρχείο αυτο περιέχονται οι δηλώσεις των συναρτήσεων και των δομών δεδομένων.

Οδηγίες Μεταγλώττισης

Τα προγράμματα μεταγλωτίζονται με την εντολή make.

Οδηγίες Χρήσης

LSH: ./lsh –i <input file> –q <query file> –k <int> -L <int> -o <output file> -N <number of nearest> -R <radius>

HyperCube: ./cube –i <input file> –q <query file> –k <int> -M <int> -probes <int> -o <output file> -N <number of nearest> -R <radius>

Cluster: ./cluster –i <input file> –c <configuration file> -o <output file> -complete <optional> -m <method: Classic OR LSH or Hypercube>

GitHub Link: https://github.com/stratisz/Project-Algorithms