README
1η Εργασία Ανάπτυξη Λογισμικού για Αλγοριθμικά Προβλήματα

Μυρτώ Πλευράκη ΑΜ: 1115201500132
Κωνσταντίνα Στόικου ΑΜ: 1115201500151

Αρχεία:

- **metrics** : συνάρτηση η οποία βρίσκει την manhattan distance δέχοντας σαν ορίσματα ενα x και ενα y τα οποία είναι η μία διάσταση για τα δύο σημεία αντίστοιχα.
- **point** : η κλάση που αντιπροσωπεύει το κάθε σημείο και αποτελείται από ένα vector μεγέθους dimension (που είναι το διάνυσμα του σημείου) και το id του.
- **read_functions** : Συναρτήσεις για να διαβάζουν τα command line arguments, τα αρχεία με τα διανύσματα (και τα εισάγει σε μία δομή vector), την είσοδο του χρήστη (τα αρχεία που δίνει αν δεν τα έχει ήδη δώσει σαν command line arguments).
- **print_functions** : κάποιες συναρτήσεις για debugging (εκτύπωση vectors με σημεία και διανύσματα, εκτύπωση κοντινότερων γειτόνων) και η συνάρτηση που εκτυπώνει με το κατάλληλο format στο αρχείο output τα αποτελέσματα του LSH.
- **h** : η κλάση που αντιπροσωπεύει τις συναρτήσεις h. Κάθε κλάση έχει έναν vector με τις τιμές s_i που αρχικοποιείται στον constructor με τιμές από ομοιόμορφη κατανομή. Η συνάρτηση-μέλος h_func υπολογίζει την τιμή h για το δοθέν σημείο χρησιμοποιώντας την συνάρτηση-μέλος a. Για την αποφυγή overflow στις μεταβλητές της h_func χρησιμοποιούνται οι ιδιότητες του modulo:
  - (a+b) mod c = ((a mod c) + (b mod c)) mod c
  - (a*b) mod c = ((a mod c) * (b mod c)) mod c
  - (a^b) mod c = ((a mod c)^b) mod c
  - (a mod c) mod c = a mod c

  γι' αυτό υπάρχουν επαναλήψεις μέσα στην h_func που κάνουν σταδιακά mod στις μεταβλητές.
- **dtw** : συνάρτηση που υπολογίζει την dtw απόσταση, δέχεται δύο vectors (x και y) και επιστρέφει την απόσταση σε vector.
- **euclidean** : συνάρτηση που υπολογίζει την ευκλείδεια απόσταση, δέχεται δύο vectors (x και y) και επιστρέφει την απόσταση σε double.
- **hamm** :
- **f_function** : επιστρέφει τυχαία 0 ή 1 χρησιμοποιώντας την κανονική κατανομή
- **defines** : αρχείο με defined τους κωδικούς των χρωμάτων για τον terminal.
- **hash** : η κλάση για την hash function g. Αποτελείται από έναν vector με Η κλάσεις που είναι και η οικογένεια συναρτήσεων για την συγκεκριμένη g (και συνεπώς για τον συγκεκριμένο hashtable). Η concat_hash_values υπολογίζει την τιμή της g για ένα συγκεκριμένο σημείο κάνοντας concatenate όλες τις τιμές του vector με τις Η (αφού καλεί για κάθε Η του vector την h_func για να υπολογίσει την τιμή της h για το συγκεκριμένο σημείο).
- **structures** : περιέχει το struct που χρησιμοποιείται για το hashtable του υπερκύβου και περιλαμβάνει έναν int και μία λίστα από pointers σε σημεία.
- **hypercube_ht** : περιέχει την κλάση για το hashtable το οποίο είναι μεγέθους n (όσα τα σημεία) και αποθηκεύει τα σημεία σε πλειάδες, από τις οποίες η κάθε μία

αποτελείται από έναν vector που αποτελείται από 0 και 1 και είναι μεγέθους d'=logn και το σημειο. Και το hashing γινεται με βάση τον vector.

- **hashtable_lsh** : η κλάση με τον hashtable για τον αλγόριθμο LSH. Αποτελείται από buckets με λίστες (για να αποθηκεύονται τα collisions) με pointers σε κλάσεις Point (για να μην υπάρχει data duplication). Κάθε φορά που εισάγεται ένα σημείο και καλείται η insert_item, καλείται η concat_hash_values της g ώστε να υπολογιστεί το hash value του συγκεκριμένου σημείου και μετά γίνεται mod με το μέγεθος του πίνακα.
- **hashtable_cube** : η κλάση με τον hashtable για τον αλγόριθμο του υπερκύβου, η οποία λειτουργεί με παρόμοιο τρόπο όπως η κλάση με τον hashtable για τον αλγόριθμο LSH με τη διαφορά ότι τα buckets χαρακτηρίζονται με 0 ή 1. Χρησιμοποιεί τη δομή struct που υλοποιείται στο αρχείο structures.
- **main_functions** : συναρτήσεις που χρησιμοποιούνται στην main για τον αλγόριθμο lsh. Συγκεκριμένα συνάρτηση για να υπολογίζει την μέση απόσταση των κοντινότερων γειτόνων για τα αρχικά input σημεία (χρειάζεται για τον υπολογισμό του w), συνάρτηση για exhaustive nearest neighbor που αποθηκεύει σε vector τους αληθινούς κοντινότερους γείτονες, συνάρτηση που δημιουργεί και αρχικοποιεί τους L hashtables (στην αρχικοποίηση περιλαμβάνεται και η αρχικοποίηση των s_i για τις H κλάσεις), συνάρτηση για εύρεση κοντινότερων γειτόνων με βάση τον αλγόριθμο LSH (λεπτομέρειες παρακάτω), συνάρτηση εύρεσης ακρίβειας (πόσοι προσεγγιστικοί γείτονες είναι και οι αληθινοί), συνάρτηση εύρεσης μέσου απόλυτου σφάλματος, συνάρτηση για τον υπολογισμό του κλάσματος προσέγγισης (και εύρεση μεγίστου), συνάρτηση για εύρεση μέσου χρόνου αναζήτησης των γειτόνων.
- **main_lsh** : διαβάζει τα command line arguments και αν δεν έχουν δοθεί ορισμένα τα ζητάει από τον χρήστη σαν είσοδο. Υπολογίζει το w με βάση την μέση απόσταση των γειτόνων των input file σημείων, και δημιουργεί τους L hashtables. Κάνει exhaustive αναζήτηση των γειτόνων και αποθηκεύει τα αποτελέσματα σε έναν vector με πλειάδες <id query, id neighbor, distance, search time>. Κάνει το ίδιο και για την lsh αναζήτηση και συγκεκριμένα για κάθε query σημείο ψάχνει σε κάθε έναν από τους L hashtables το κελί στο οποίο θα έμπαινε αν καλούσαμε την insert_item (δηλαδή βρίσκει την τιμή της g για το συγκεκριμένο query). Ψάχνει σε αυτά τα buckets τα σημεία που έχουν ίδια g (γιατί κάποια σημεία δεν θα έχουν ίδια g λόγω του g mod tablesize). Βρίσκει τις αποστάσεις από αυτά τα σημεία και αποθηκεύει την μικρότερη. Αν σε όλα τα hashtables το bucket που "πέφτει" το query είναι άδειο τότε αποθηκεύεται -1 σαν απόσταση και σαν id γείτονα. Μετά τα αποτελέσματα του exhaustive search και του lsh search (δηλαδή τα vectors με πλειάδες) εκτυπώνονται με το κατάλληλο format στο αρχείο εξόδου. Επίσης εκτυπώνονται στον terminal η ακρίβεια, το μέσο απόλυτο σφάλμα, το κλάσμα προσέγγισης, το μέγιστο κλάσμα προσέγγισης και ο μέσος χρόνος αναζήτησης για τους προσεγγιστικά κοντινότερους γείτονες.
- **main_cube** : παρόμοιες λειτουργίες με τη main_lsh

<u>Σημειώσεις:</u>

- Στην main_lsh αποδεσμεύεται όλη η δεσμευμένη μνήμη.

- Έχουμε ολοκληρώσει το Α ερώτημα. Ο αλγόριθμος LSH λειτουργεί σωστά αν και για μεγάλα αρχεία αργεί (ειδικά με valgrind). Ο αλγόριθμος για τον hypercube δεν έχει δοκιμαστεί λόγω μεγάλης καθυστέρησης στην εκτέλεσή του (ειδικά με valgrind).
- Όσον αφορά το Β ερώτημα υλοποιήσαμε κάποιες μετρικές που χρειάζονται στις καμπύλες.