compile : για να κάνουμε compile χρησιμοποιούμε την εντολή make μεσα στο φάκελο Project/makefile.

Execute: Από το compile προκύπτουν τα εξής 6 εκτελέσιμα:

: ./**lsh** -d <input file> -q <query file> -o <output_file> -L <table_size> -k <k for lsh> -w

: ./**cube** -d <input file> -q <query file> -o <output_file> -k <table_size(αν δεν δοθει log n)> - M <M πόσους θα ψάξει)> -p <probes>

:**curve_grid_lsh** -d <input file> -q <query file> -o <output_file> -o -k_vec <> -L_grid <> -w -st <search threshold> --delta < > --identical <check for same grid curve>

:**curve_grid_hypercube** -d <input file> -q <query file> -o <output_file> - k <table size (default logn)> M <ποσους να ψαξει> -p <probes> -L <hash tables> -D <delta>

:**curve_projection_lsh** -d <input file> -q <query file> -o <output_file-o -k_vec <> -L_grid <> -w -st <search threshold> --eps <epsilon> --delta < > --identical <check for same grid curve>

:**curve_projection_hypercube** -d <input file> -q <query file> -o <output_file> -e <ε> - k <table size (default logn)> M <ποσους να ψαξει> -p <probes>

Οι μόνες υποχρεωτικές παράμετροι είναι οι ‘-d’, ‘-q’, ‘-o’. Για όλες τις υπόλοιπες υπάρχουν default μεσα στα προγράμματα.

Αν στα lsh/cube δοθεί w = -1, υπολογίζεται στην αρχη του προγραμματος. Επίσης το δελτα υπολογίζεται στην αρχη του προγραμματος σε περιπτωση που δεν δωθει κατι από την γραμμη εντολών.

LSH

Class LSH: Η βασική δομή του LSH. Περιλαμβάνει, μεταξύ άλλων, έναν πινάκα απο **L** hash tables και έναν πινάκα από δυνάμεις του **m**.

Class Hash_Table: Το hash table στο όποιο αποθηκεύουμε και αναζητούμε διανύσματα. Περιλαμβάνει ένα unordered_multimap με κλειδί, τιμές της g και values, δείκτες σε διανύσματα (Class Item).

Class Item: Αναπαριστά ένα διάνυσμα. Περιλαμβάνει το όνομα του διανύσματος και έναν πίνακα από τις συντεταγμένες του διανύσματος**.**

Class Query_Result: Αναπαριστά το αποτέλεσμα ενος query. Περιλαμβάνει το όνομα του κοντινότερου γείτονα, την απόσταση από αυτόν και το χρόνο εύρεσης του.

Υπολογισμός της συνάρτησης **g (g_hash_function):** Οι δυνάμεις του **m** υπολογίζονται κατα την δημιουργία του **LSH** και αποθηκεύονται σε πίνακα ώστε να μειωθεί ο χρόνος υπολογισμού. Ως **M** χρησιμοποιούμε την τιμή 2^32 - 5.

Hypercube

Class hypercube :Η βασική δομη του cube περιέχει μέσα όλα τα hash tabes M κ table_size κλπ

Class hash_table : Είναι το hash_table μας που ουσιαστικά αποθηκεύω τα vectors από s εχω 1 vector από map που κάνω mapping τα αποτελέσματα που μου δίνει η g και άλλο ένα που multimap που κάνω mapping τα αποτελέσματα τις p με το item

Curve Grids

Curve_Grid_LSH/Curve_Grid_Hypercube: Η βασική δομή της προβολής των καμπυλών σε grids με LSH/Hypercube. Περιλαμβάνει, μεταξύ άλλων, έναν πίνακα απο **L** σημεία t (Class Point), ενα για κάθε grid.

Class Curve: Αναπαριστά μια καμπύλη. Περιλαμβάνει το ονομα της καμπύλης και έναν πίνακα απο σημεία (Class Point). Επίσης, περιλαμβάνει έναν δείκτη σε μια άλλη κάμπυλη. Ο δείκτης αυτός χρησιμοποιείται στην περίπτωση που η καμπύλη αναπαριστά ενα grid curve.

Class Point: Αναπαριστά ένα σημείο της καμπύλης. Περιλαμβάνει έναν πίνακα από συντεταγμένες του σημείου. Στην περίπτωση μας, στην οποία οι καμπύλες βρίσκονται στο επίπεδο, ο πίνακας περιέχει 2 μόνο στοιχεία (x, y).

Μετατροπή καμπύλης σε διάνυσμα (**convert_2d_curve_to_vector**): Για κάθε σημείο της καμπύλης βρίσκουμε το κοντινότερο σημείο πάνω στο grid, διαγράφουμε τα διπλότυπα, κάνουμε padding με τη μεγαλύτερη συντεταγμένη των καμπυλών και κάνουμε concatenate όλα τα σημεία ώστε να δημιουργηθεί ένα διάνυσμα μεγέθους 2*max_curve_length.

Υπολογισμος του **δ** (**calculate_delta**)**:** Το δ μπορεί να δοθεί ως όρισμα απο την γραμμή εντολών. Σε περίπτωση που δεν δοθεί, υπολογίζεται ως ο μέσος όρος απόστασης των διαδοχικών κορυφών όλων των καμπυλών απο το input.

Curve Projections

Class Curve_Projection_LSH/Curve_Projection_Hypercube: Η βασική δομή της προβολής των καμπυλών με την χρήση των relevant traversals με LSH/Hypercube. Περιλαμβάνει, μεταξύ άλλων, τον πίνακα **G_matrix** και τον πινακα **MxM** από **relevant traversals** (**Class Relevant_Traversals**).

Class Relevant_Traversals: Αναπαριστά τον κατάλογο απο relevant traversals που βρίσκονται σε κάθε κελί του πινακα **MxM**. Περιλαμβάνει, μεταξύ άλλων, μια λιστα απο πινακες με ζεύγη ‘δεικτών’ σε σημεία καμπυλής (**Class Tuple**).

Class Tuple: Αναπαριστά ένα ζεύγος ‘δεικτών’ σε σημεία δύο καμπυλών. Περιλαμβάνει 2 ακεραίους (x, y).

Μετατροπή καμπύλης σε διάνυσμα (**convert_2d_curve_to_vector_by_projection**): Πολλαπλασιάζουμε τον πίνακα **G** με τα σημεια της καμπυλης που αντιστοιχουν στους δεικτες απο τα ζεύγη των traversal. Στην περιπτωση της εισαγωγής μια καμπύλης στην δομή χρησιμοποιούμε τον πρώτο ακέραιο (x) του ζεύγους (**Class Tuple**), ενώ στην περίπτωση της αναζήτησης χρησιμοποιούμε τον δεύτερο ακέραιο (y) του ζεύγους.

Σημειωσεις:

1. Για εξοικονόμηση μνήμης βρίσκουμε τα traversals και δημιουργούμε τις δομες ANN μονο όταν χρειάζονται.

1. Για τον υπολογισμο των traversals λαμβανουμε υποψιν μονο την διαγωνιο και τα κελια που βρισκονται μια θεση παντω από αυτην.

Παρατηρήσεις

Μετά από πολλες μετρήσεις συμπεράναμε τα εξής:

1. Η επιλογη του w είναι πολύ σημαντική και πρέπει να είναι ανάλογο των αποστάσεων των διανυσμάτων.
2. Η επιλογή του δελτα για grids είναι πολύ σημαντική και πρεπει να είναι αναλογη των αποστασεων διαδοχικων κορυφών των καμπυλών.