Ανάπτυξη Λογισμικού για Αλγοριθμικά Προβλήματα - Άσκηση 2

Κωνσταντίνα Έλληνα - 1115201600046
Απόστολος Λύρας - 1115201600097

Η δεύτερη εργασία βασίζεται σε μεγάλο βαθμό στην πρώτη. Υπάρχουν δύο αρχεία με main και συναρτήσεις που χρησιμοποιούνται μέσα σε αυτή. Μεταγλωτίζονται με make και οι εντολές με τις οποίες τρέξαμε την κάθε περίπτωση είναι οι εξής:

A1-lsh) ./search -i nasd_input.csv -q nasd_query.csv -k 3 -L 2 -M 10 -probes 2 -o out_lsh -algorithm LSH -metric discrete -delta 1

A1-cube) ./search -i nasd_input.csv -q nasd_query.csv -k 3 -L 2 -M 10 -probes 2 -o out_cube -algorithm Hypercube -metric discrete -delta 1

A2) ./search -i nasd_input.csv -q nasd_query.csv -k 3 -L 2 -M 10 -probes 2 -o out_frechet_discrete -algorithm Frechet -metric discrete -delta 1

A3) ./search -i nasd_input.csv -q nasd_query.csv -k 3 -L 2 -M 10 -probes 2 -o out_frechet_continuous -algorithm Frechet -metric continuous -delta 1

B1-vectors) ./cluster -i nasd_input.csv -c configuration.ini -o out_classic -update Mean_Vector -assignment Classic -complete -silhouette

B1-curves) ./cluster -i nasd_input.csv -c configuration.ini -o out_classic_frechet -update Mean_Frechet -assignment Classic -complete -silhouette

B2) ./cluster -i nasd_input.csv -c configuration.ini -o out_lsh -update Mean_Frechet -assignment LSH_Frechet -complete -silhouette

B3) ./cluster -i nasd_input.csv -c configuration.ini -o out_lsh -update Mean_Vector -assignment LSH -complete -silhouette

B4) ./cluster -i nasd_input.csv -c configuration.ini -o out_cube -update Mean_Vector -assignment Hypercube -complete -silhouette

Οι διαφορές από την 1η εργασία βρίσκονται κυρίως στο A2, A3, B1-curves, B2.

Στο A2 κάνουμε snapping σε 2D grid και βάζουμε το αποτέλεσμα σε ένα πίνακα από struct Node* όπου η κάθε δομή έχει τις συντεταγμένες x,y του dataset. Φτιάχνουμε λοιπόν έναν πίνακα με 2*dimension θέσεις και τοποθετούμε όλα τα στοιχεία που υπάρχουν στον πρώτο πίνακα. Αυτά τα στοιχεία θα έχουν μετακινηθεί λόγω του grid και θα είναι πολλαπλάσια του δ, που μας δίνεται. Κάνουμε και padding για να έχουμε ίδιες διαστάσεις σε όλες τις καμπύλες. Αυτόν τον 2 dimensional πίνακα τον στέλνουμε για hashing. Φτιάχνουμε τις h function και έχουμε το αποτέλεσμα που θέλουμε για να βάλουμε τις καμπύλες μας στο hash table. Η απόσταση που χρησιμοποιούμε είναι discrete frechet.

Περίπου το ίδιο γίνεται και στο A3 μόνο που τώρα κάνουμε filtering, μετά snapping σε 1D grid, minima-maxima και padding. Filtering γίνεται ανάμεσα σε 3 στοιχεία, όπου διαγράφουμε τα “αχρείαστα”, δηλαδή αυτά που δεν αλλάζουν κατά πολύ το αποτέλεσμά μας. Αυτό γίνεται με τη χρήση μιας σταθεράς ε και την απόσταση των σημείων αυτών με τη σταθερά. Minima-maxima γίνεται πάλι ανάμεσα σε 3 στοιχεία και αυτή τη φορά διαγράφουμε το μεσαίο στοιχείο. Τα υπόλοιπα είναι ίδιας λογικής με το A2. Κανονικά η απόσταση θα έπρεπε να είναι continuous frechet αλλά δεν μπορούσαμε να συνδέσουμε αυτό που μας δόθηκε με το πρόγραμμά μας που είναι σε C και χρησιμοποιήσαμε πάλι discrete frechet.

Στο B ερώτημα αυτό που αλλάζει είναι ότι πρέπει το update να γίνει με mean curve. Γι’ αυτό το λόγο φτιάξαμε ένα binary complete tree όπου βάζαμε τυχαία τα στοιχεία του dataset και βρίσκαμε τη mean discrete frechet curve ανάμεσα στα φύλλα του δέντρου. Χρησιμοποιήσαμε Post-order tree traversal και βρίσκαμε τα καινούργια κάθε φορά κεντροειδή. Η υπόλοιπη διαδικασία είναι η ίδια.
Στο B2 εκτός από την αλλαγή στο update, κάναμε πρώτα snapping και padding στις καμπύλες για να τις διαχειριστούμε ως διανύσματα. Μετά από αυτό το βήμα το hashing γίνεται κανονικά και το update με mean curve.

Τέλος, κάναμε unit testing στο search.c σε συναρτήσεις από το cube_funcs που είναι καθαρά μαθηματικές.