Ανάπτυξη Λογισμικού
για
Αλγοριθμικά Προβλήματα

Εργασία 1

Μέλη ομάδας:
Μητσοπούλου Άννα, 1115201500097
Ρόκα Κωνσταντίνα, 1115201500139

Περιγραφή Προγράμματος

Στην εργασία μας καλούμαστε να επιλύσουμε το προσεγγιστικό πρόβλημα εύρεσης κοντινότερου γείτονα για σημεία και καμπύλες. Οι αλγόριθμοι που υλοποιούνται είναι οι εξής:

1. Για σημεία:
   - LSH: Δημιουργία ενός μεταβλητού (ορισμένο από την έναρξη του προγράμματος) αριθμού τυχαία επιλεγμένων πλεγμάτων, πάνω στα οποία τοποθετούνται τα σημεία μας. Όταν ο αλγόριθμός μας δεχτεί ένα σημείο, ψάχνει για κάθε πλέγμα, τα σημεία που βρίσκονται στο ίδιο κουτάκι με το σημείο μας και βρίσκει ποιο είναι αυτό που είναι κοντινότερό του.
   - Hypercube: Δημιουργία ενός υπερκύβου διάστασης ορισμένης από την έναρξη του προγράμματος, στις κορυφές του οποίου αντιστοιχίζονται τα σημεία μας. Όταν ο αλγόριθμός μας δεχτεί ένα σημείο, ψάχνει την αντίστοιχη κορυφή (και τις γειτονικές του αν αυτό έχει οριστεί από το πρόγραμμα) και ελέγχει ποιο από τα σημεία που βρίσκονται σε αυτή την κορυφή είναι το κοντινότερό του.
2. Για καμπύλες:
   - Grid: Δημιουργία πολλών πλεγμάτων τα οποία χρησιμοποιούνται προκειμένου να γίνει snapping των καμπυλών, ώστε να μπορούν να δωθούν σαν δεδομένα στους αλγορίθμους LSH και Hypercube. Αυτό γίνεται προβάλλοντας κάθε σημείο της καμπύλης σε ένα σημείο του πλέγματος και δημιουργώντας σημείο διαστάσεων ανάλογων με το πλήθος των σημείων του πλέγματος που αντιστοιχίζονται με σημεία της καμπύλης . Έπειτα, κάθε πλέγμα αντιστοιχίζεται με μία δομή LSH ή Hypercube, από τις οποίες επιστρέφεται κοντινότερος γείτονας.
   - Projection: Δημιουργία δομής αναπαράστασης όλων των πιθανών traversals (που ακολουθούν τα στοιχεία της διαγωνίου) ανάμεσα σε καμπύλες διαφόρων μεγεθών και αντιστοίχισης κάθε traversal με μια δομή LSH ή Hypercube. Οι καμπύλες μετατρέπονται σε σημεία, κρατώντας τα σημεία της καμπύλης τα οποία αντιστοιχούν στο κάθε traversal. Ο κοντινότερος γείτονας βρίσκεται εξετάζοντας τους πιθανούς γείτονες που επιστρέφουν οι δομές μας για κάθε πιθανό traversal ανάμεσα στην καμπύλη-ερώτημα και σε καμπύλες με μήκος περίπου ίσο με αυτό της δικής μας.

Δομή Αρχείων Εργασίας

Το Project μας αποτελείται από τους εξής καταλόγους:

- **bin:** Κατάλογος με τα τελικά εκτελέσιμα όπως περιγράφονται από την εκφώνηση.
- **build:** Κατάλογος που περιέχει τα ενδιάμεσα αρχεία .o.
- **include:** Κατάλογος που περιέχει όλες τα αρχεία επικεφαλίδας μας και τον κώδικά τους (.hpp, .cpp).
  - **DataHandling (.hpp, .cpp):** Περιέχει όλες τις συναρτήσεις που μας βοηθούν στο διάβασμα των αρχείων, την αποθήκευσή τους στο πρόγραμμα και την δημιουργία των αρχείων εξόδου.
  - **Utilities (.hpp, .cpp):** Περιέχει όλες τις κλάσεις που χρησιμοποιούνται για την αναπαράσταση των δεδομένων (σημείου, καμπύλης, κοντινότερου γείτονα), καθώς και τις συναρτήσεις οι οποίες δεν σχετίζονται άμεσα με την φύση των αλγορίθμων που μελετάμε (πχ. Brute force, συναρτήσεις υπολογισμού απόστασης, κλπα)
  - **LSH_Structure (.hpp, .cpp):** Περιέχει όλες τις κλάσεις και συναρτήσεις που είναι χρήσιμες για την υλοποίηση του αλγορίθμου LSH
  - **Hypercube_Structure (.hpp, .cpp):** Περιέχει όλες τις κλάσεις και συναρτήσεις που είναι χρήσιμες για την υλοποίηση του αλγορίθμου Hypercube
  - **Grid (.hpp, .cpp):** Περιέχει όλες τις κλάσεις και τις συναρτήσεις που είναι χρήσιμες για την υλοποίηση του αλγορίθμου εύρεσης κοντινότερου γείτονα μέσω Grid και LSH/Hypercube.
  - **Projection (.hpp, .cpp):** Περιέχει όλες τις κλάσεις και τις συναρτήσεις που είναι χρήσιμες για την υλοποίηση του αλγορίθμου εύρεσης κοντινότερου γείτονα μέσω προβολής καμπύλης σε σημείο και LSH/Hypercube.
- **src:** Κατάλογος που περιέχει τον κώδικα για τις main μας.
  - **./Points/lsh:** Περιέχει τη main για το lsh σημείων.
  - **./Points/cube:** Περιέχει τη main για το hypercube σημείων.
  - **./Curves/Grid/grid_lsh:** Περιέχει την main για το grid_lsh καμπυλών.
  - **./Curves/Grid/grid_hypercube:** Περιέχει τη main για το grid_hypercube καμπυλών.
  - **./Curves/Projection/projection_lsh:** Περιέχει τη main για το projection_lsh καμπυλών.
  - **./Curves/Projection/projection_hypercube:** Περιέχει τη main για το projection_hypercube καμπυλών.

- **Makefile:** Αρχείο makefile για να είναι πιο εύκολη η μεταγλώττιση.

Οδηγίες μεταγλώττισης / εκτέλεσης

Για τη μεταγλώττιση του προγράμματος αρκεί η εντολή make. Τα αρχεία build/bin πρέπει να προϋπάρχουν.

Για την εκτέλεση του προγράμματος ακολουθήσαμε τις οδηγίες της εκφώνησης, απλά χρειάζεται να προστεθεί ένα ./bin στην αρχή του μονοπατιού εκτέλεσης.

Συμπεράσματα

Συγκρίνοντας τους χρόνους εκτέλεσης των προγραμμάτων για τα σημεία lsh και hypercube, συμπεράναμε πως ο lsh υπερτερεί του hypercube σε θέμα αποτελεσμάτων αλλά είναι λίγο πιο αργός. Ο hypercube παρουσιάζει αρκετά καλούς χρόνους και με κατάλληλες παραμέτρους πλησιάζει το average που εμφανίζει και ο lsh, αλλά βλέπουμε πως για συγκεκριμένα queries δεν δουλεύει πολύ καλά (παρατήρηση του μέγιστου κλάσματος προσέγγισης).

Παρατηρώντας τους αλγόριθμους καμπυλών, συμπεραίνουμε πως οι χρόνοι και ο χώρος που απαιτείται για τον αλγόριθμο προβολών κάνει την πιθανότητα χρήσης του απαγορευτική, ιδιαίτερα αν αναφερόμαστε σε καμπύλες μεγάλου μήκους. Ο αλγόριθμος πιθανά να δουλεύει καλύτερα άμα το dataset αποτελείται από πολλές καμπύλες, αλλά στο τρέχον dataset, η εύρεση του κοντινότερου γείτονα με brute force είναι κατά πολύ αποδοτικότερη. Ο αλγόριθμος (για το τρέχον dataset) βρίσκει αρκετά καλούς γείτονες, αλλά σε πολλές περιπτώσεις δεν βρίσκει κανένα γείτονα. Ο αλγόριθμος που αφορά τα grids, με κατάλληλες παραμέτρους, βρίσκει πάντα κοντινότερο γείτονα, με καλή μετρική average, αλλά όχι καλό max, γεγονός που υποψιαζόμαστε ό,τι οφείλεται στη μεγάλη απόκλιση των μηκών των καμπυλών που απαρτίζουν το dataset.

Σχόλια υλοποίησης

Στα προγράμματά μας επιλέξαμε να γίνεται ο υπολογισμός του w (παράμετρος lsh) κατά τη διάρκεια της εκτέλεσης, ώστε να μπορεί να τρέξει σωστά για διαφορετικά dataset. Η διαδικασία όμως αυτή απαιτεί πολύ χρόνο, ειδικά για μεγάλα dataset ή για τους αλγορίθμους του 2β που υλοποιούν πάρα πολλές δομές lsh.

Επίσης, επειδή το πρόγραμμα ζητάει συγκρίσεις πραγματικών με προσεγγιστικές τιμές, θεωρήσαμε πως το πρόγραμμά μας πρέπει να περιέχει το brute force, γεγονός που αυξάνει αρκετά τον χρόνο εκτέλεσής του.