2η Άσκηση - Χωρικά Δεδομένα

Νεφέλη-Ελένη Κατσιλέρου

ΑΜ : 4385

Μέρος 1

Ξεκινώντας την υλοποίηση της άσκησης διαβάζω και φορτώνω τα δεδομένα χρησιμοποιώντας τη συνάρτηση read_data.

Αναλυτικότερα : Ανοίγω το αρχείο για ανάγνωση με την open() , διαβάζω και μετατρέπω την πρώτη γραμμή του αρχείου σε integer (δείχνει τον συνολικό αριθμό των εστιατορίων που περιέχει το αρχείο) και διαβάζω επαναληπτικά κάθε επόμενη γραμμή του αρχείου. Τα δεδομένα κάθε γραμμής (αντιστοιχούν στις x και y συντεταγμένες των εστιατορίων) μετατρέπονται σε float και τα αποθηκεύω σε πλειάδα μαζί με το αντίστοιχο αναγνωριστικό γραμμής.

Αφού ολοκληρωθεί η φόρτωση των δεδομένων ταξινομώ τα σημεία με βάση την χ συντεταγμένη.

Στη συνέχεια με τη συνάρτηση slice_points() διαιρώ τη λίστα με τα ταξινομημένα σημεία σε μικρότερες ομάδες με βάση το layer_size .Κάθε slice ταξινομείται τώρα με βάση την y συντεταγμένη.

Για την δημιουργία του R-tree, ξεκινάω φτιάχνοντας την συνάρτηση create_leaf() όπου κάθε slice διαιρείται ξανά σε ομάδες των n σημείων (όπου n είναι ο μέγιστος αριθμός σημείων που μπορεί να χωρέσει ενας κόμβος με βάση το capacity_size). Για κάθε τέτοια ομάδα κατασκευάζω ένα κόμβο leaf και προσθέτω εγγραφή που περιέχει το αναγνωριστικό και τις συντεταγμένες κάθε σημείου. Στη λίστα tree_nodes αποθηκεύω τον κόμβο και του δίνω ένα μοναδικό id που αυξάνεται.

Έπειτα αφού έχω κατασκευάσει τους κόμβους leaf τους χρησιμοποιώ στην συνάρτηση create_Rtree(). Αποθηκεύω τα φύλλα και εφόσον υπάρχουν περισσότεροι από έναν κόμβο στο τρέχον επίπεδο γίνονται τα εξής :

Διαιρούνται σε ομάδες των node_points και για κάθε ομάδα υπολογίζω το MBR της. Με αυτά τα MBR δημιουργούνται νέοι κόμβοι που είναι γονείς των προηγούμενων και οι νέοι κόμβοι αποθηκεύονται σε ένα νεό επίπεδο της λίστας levels. Αυτή η διαδικασία γίνεται επαναληπτικά μέχρι να φτιάσουμε στη ρίζα.

Ανέφερα τον υπολογισμό MBR που γίνεται μέσω της συνάρτησης calculate_mbr().Αρχικά ανάλογα με τη διάσταση του πρώτου στοιχείου της λίστας coordinates καθορίζω αν τα δεδομένα είναι 2D συντεταγμένες όπου σε αυτή τη περίπτωση υπολογίζω τις ελάχιστες και μέγιστες τιμές των συντεταγμένων ή ήδη MBRS (4D συντεταγμένες) πράγμα που σημαίνει ότι τα δεδομένα προέρχονται από ενδιάμεσους κόμβους και εκτελείται παρόμοιος υπολογισμός για να δημιουργηθεί ένα νέο MBR .

Τέλος στη συνάρτηση write_output() υπολογίζω τον συνολικό αριθμό των κόμβων και του ύψους του δέντρου και εκτυπώνω τα ζητούμενα στατιστικά. Κάθε κόμβος έχει το id του , τον αριθμό εγγραφών που περιέχει , το αν είναι φύλλο ή μη και τα δεδομένα των εγγραφών του. Για κάθε επίπεδο υπολογίζω επίσης τη μέση έκταση των MBR.

Μέρος 2

Σε αυτό το μέρος εφαρμόζω τον αλγόριθμο εύρεσης πλησιέστερων γειτόνων βασισμένο στη καλύτερη πρώτη αναζήτηση στο R-tree που κατασκεύασα στο πρώτο μέρος. Αρχικά όπως είπα και παραπάνω κάθε κόμβος έχει ενα χαρακτηριστικό id (node_ID) , μια λίστα απο εγγραφές (records) και το leaf που δηλώνει αν ο κόμβος είναι φύλλο ή όχι. Οι εγγραφές μπορεί να είναι συντεταγμένες σημείων ή MBR αντίστοιχα.

Ξεκινώντας με τη συνάρτηση load_Rtree() φορτώνω το R-tree απο το αρχείο. Ανοίγω το αρχείο , η πρώτη γραμμή περιέχει το αναγνωριστικό της ρίζας (root_node_id) και το μετατρέπω σε ακέραιο. Για κάθε γραμμή του αρχείου διαχωρίζονται τα στοιχεία του κόμβου. Η συνάρτηση parse_node_records() αναλύει τις εγγραφές απο μια συμβολοσειρά και εξάγει τις συντεταγμένες και τα ids των εγγραφών και τα αποθηκεύει αντίστοιχα.

Στη συνέχεια υλοποιείται η συνάρτηση mindist όπως ζητείται και στην εκφώνηση , η οποία υπολογίζει την ελάχιστη Ευκλείδεια απόσταση μεταξύ του σημείου q και ενός άλλου σημείου ή ενός MBR. Ελέγχω αν η λίστα coordinates περιέχει 4 τιμες τότε ειναι MBR αλλιώς είναι 2D σημείο. Οι συντεταγμένες του σημείου q συγκρίνονται με τις συντεταγμένες του MBR. Για κάθε x και y υπολογίζω την ελάχιστη απόσταση απο τα όρια του MBR και αν το σημείο βρίσκεται εντός των ορίων για τη συγκεκριμένη διάσταση τότε η απόσταση είναι μηδέν. Η συνολική απόσταση υπολογίζεται όπως ζητείται στην εκφώνηση. Στη περίπτωση που έχουμε 2D σημείο η Ευκλείδεια απόσταση υπολογίζεται κατευθείαν χρησιμοποιώντας τον γνωστό τύπο.

Στη συνάρτηση incremental_nearest_neighbors() χρησιμοποιείται μια ουρά προτεραιότητας προκειμένου να διαχειρίζονται τις ελάχιστες αποστάσεις από το σημείο q στα διάφορα στοιχεία του δέντρου. Εισάγω στην ουρά τις εγγραφές της ρίζας υπολογίζοντας την απόσταση τους από το q με τη χρήση της mindist. Έπειτα οι εγγραφές ανακτώνται από την ουρά με βάση την ελάχιστη απόσταση. Αν ο κόμβος είναι φύλλο μπαίνει στη λίστα των πλησιέστερων γειτόνων , αλλιώς αν είναι ενδιάμεσος βρίσκω νέες πιθανές εγγραφές που θα μπουν στην ουρά. Η διαδικασία συνεχίζεται έως ότου βρεθούν οι k πλησιέστεροι γείτονες.

Η ουρά προτεραιότητας χρησιμοποιείται για την διαχείριση των κόμβων και των εγγραφών του Rtree. Κάθε στοιχείο είναι μια πλειάδα που περιέχει την ελάχιστη απόσταση από το q , το id εγγραφής ,τις συντεταγμένες και αν είναι φύλλο ή όχι.

Τέλος εκτυπώνω τις ζητούμενες πληροφορίες για τους πρώτους k πλησιέστερους γείτονες , επεκτείνω τη διαδικασία για να τυπώσω τους k+1 , k+2 καθώς επίσης και τα περιεχόμενα της ουράς προτεραιότητας μόλις έχει υπολογιστεί ένας γείτονας.