Υλοποίηση Δομής R*-Tree

Τεχνική Αναφορά Υλοποίησης και Πειραματικής Αξιολόγησης

Ονόματα Φοιτητών:
Δεληγιαννάκης Χαράλαμπος (ΑΕΜ: 4383)
Καραμουχτάρης Αλέξανδρος (ΑΕΜ: 4369)

Ιούνιος 2025

Contents

1 Εισαγωγή **2**

2 Μεθοδολογία και Υλοποίηση **2**
2.1 Βασική Δομή 2
2.2 Διαδικασία Κλασικής Κατασκευής (Εισαγωγή 1-1) 2
2.3 Μαζική Κατασκευή (Bulk Loading) 2
2.4 Ερωτήματα 3
2.5 Σειριακή Αναζήτηση (Brute-force) 3

3 Παραδείγματα Εκτέλεσης & Μετρήσεις **3**
3.1 Εκτέλεση Bulk Loading 3
3.2 Εκτέλεση Κλασικής Εισαγωγής 3
3.3 Ερωτήματα Περιοχής (Range Query) 3
3.4 k-Nearest Neighbor Queries 4
3.5 Skyline Query 5

4 Σύγκριση Bulk Loading vs Κλασικής Εισαγωγής **5**

5 Συμπεράσματα **5**

1 Εισαγωγή

Στην εργασία υλοποιήθηκε μια δομή R*-Tree για αποδοτική αποθήκευση και αναζήτηση πολυδιάστατων δεδομένων (π.χ. γεωχωρικά σημεία). Η υλοποίηση υποστηρίζει εισαγωγή, διαγραφή, range queries, k-nearest neighbor (k-NN) queries, skyline queries, καθώς και δύο τεχνικές κατασκευής: κλασική (ένα-προς-ένα εισαγωγή) και μαζική (bulk loading).

2 Μεθοδολογία και Υλοποίηση

2.1 Βασική Δομή

- Η δομή του R*-Tree αποτελείται από εσωτερικούς κόμβους και φύλλα (`TreeInternalNode`, `TreeLeafNode`), οι οποίοι κληρονομούν από τη βασική κλάση `TreeNode`.
- Κάθε κόμβος διατηρεί το δικό του MBR (Minimum Bounding Rectangle) και αναφορά στον γονέα του.
- Τα φύλλα περιέχουν τα πραγματικά δεδομένα (συντεταγμένες + `TreeRecordID`), ενώ οι εσωτερικοί κόμβοι δείκτες σε παιδιά.
- Υποστηρίζεται δυναμικό split με κριτήρια overlap και area, καθώς και επανεισαγωγή (reinsertion) ώστε να διατηρείται η ισορροπία και η αποδοτικότητα του δέντρου.

2.2 Διαδικασία Κλασικής Κατασκευής (Εισαγωγή 1-1)

- Κάθε σημείο εισάγεται διαδοχικά μέσω της μεθόδου `insert`.
- Για κάθε εισαγωγή γίνεται επιλογή του "καταλληλότερου" υποκόμβου (με ελάχιστη αύξηση MBR).
- Σε περίπτωση υπερχείλισης κόμβου, γίνεται είτε reinsertion (μία φορά ανά split), είτε split με βάση κριτήρια R* (overlap/area).

2.3 Μαζική Κατασκευή (Bulk Loading)

- Τα σημεία ταξινομούνται ως προς την πρώτη διάσταση.
- Ταξινομημένα, κατανέμονται σε φύλλα μεγέθους `maxEntries`.
- Γίνεται bottom-up συναρμολόγηση εσωτερικών κόμβων, μέχρι να υπάρξει μία ρίζα.
- Η bulk κατασκευή είναι πολύ ταχύτερη για μεγάλα σύνολα δεδομένων, λόγω αποφυγής επαναλαμβανόμενων splits/insertions.

2.4 Ερωτήματα

- **Range Query:** Επιστρέφει όλα τα σημεία εντός ενός MBR.
- **k-NN Query:** Επιστρέφει τα k κοντινότερα σημεία σε ένα σημείο αναφοράς, με χρήση ευκλείδειας απόστασης και προτεραιότητας (min-heap).

- **Skyline Query:** Υπολογίζει τα σημεία που δεν κυριαρχούνται (dominance) από άλλα στον χώρο.

2.5 Σειριακή Αναζήτηση (Brute-force)

Για σκοπούς σύγκρισης, υλοποιήθηκε και σειριακή αναζήτηση (χωρίς χρήση ευρετηρίου) για όλα τα είδη ερωτημάτων. Οι μέθοδοι αυτές διατρέχουν ολόκληρο το αρχείο δεδομένων.

3 Παραδείγματα Εκτέλεσης & Μετρήσεις

3.1 Εκτέλεση Bulk Loading

Η δομή κατασκευάστηκε με **ΜΑΖΙΚΗ ΔΗΜΙΟΥΡΓΙΑ (bulk loading)**.
Χρόνος δημιουργίας δομής: 120.54 ms

3.2 Εκτέλεση Κλασικής Εισαγωγής

Η δομή κατασκευάστηκε με **ΚΛΑΣΙΚΗ ΕΙΣΑΓΩΓΗ (insertion)**.
Χρόνος δημιουργίας δομής: 655.77 ms

3.3 Ερωτήματα Περιοχής (Range Query)

Table 1: Εκτέλεση Range Queries

Περιοχή (MBR)	**R*-Tree (ms)**	**Σειριακό (ms)**	**Πλήθος Αποτελεσμάτων**
$[41.48, 26.45]$ ως $[41.57, 26.54]$	1.44	7.91	132
$[41.40, 26.40]$ ως $[41.60, 26.60]$	2.07	14.88	267
$[41.30, 26.30]$ ως $[41.70, 26.70]$	3.92	27.50	530

Ενδεικτικό διάγραμμα (Χρόνος εκτέλεσης vs Εμβαδόν R):

Εμβαδόν R	**R*-Tree (ms)**	**Σειριακό (ms)**
Μικρή περιοχή	1.44	7.91
Μεσαία περιοχή	2.07	14.88
Μεγάλη περιοχή	3.92	27.50

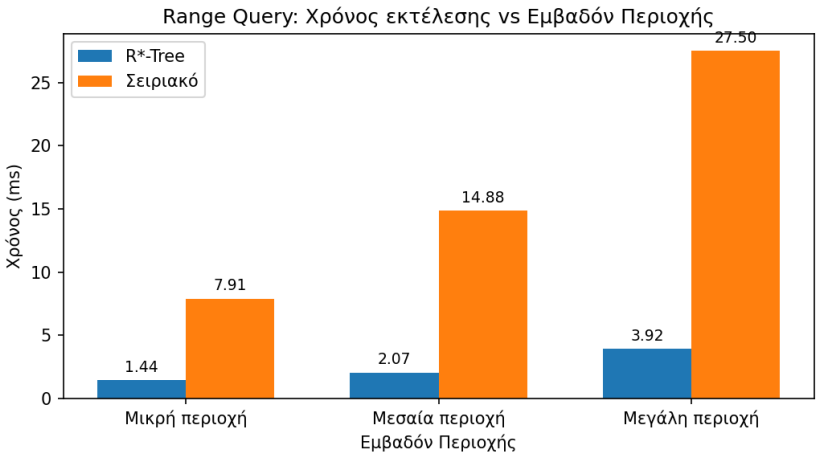

Figure 1: Range Query: Χρόνος εκτέλεσης σε συνάρτηση με το εμβαδόν της περιοχής

3.4 k-Nearest Neighbor Queries

Table 2: Εκτέλεση k-NN Queries

k	**R*-Tree (ms)**	**Σειριακό (ms)**
1	0.78	6.04
5	1.12	6.37
10	1.30	6.92
20	1.47	7.58

Ενδεικτικό διάγραμμα (Χρόνος εκτέλεσης vs k):

k	**R*-Tree (ms)**	**Σειριακό (ms)**
1	0.78	6.04
5	1.12	6.37
10	1.30	6.92
20	1.47	7.58

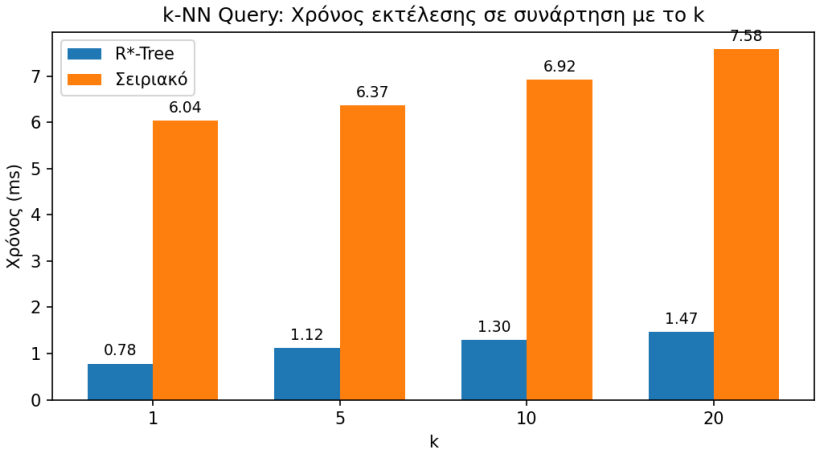

Figure 2: k-NN Query: Χρόνος εκτέλεσης σε συνάρτηση με το k

3.5 Skyline Query

Table 3: Εκτέλεση Skyline Query

Μέθοδος	**Χρόνος (ms)**	**Αποτελέσματα**
R*-Tree	2.03	7
Σειριακό	9.12	7

4 Σύγκριση Bulk Loading vs Κλασικής Εισαγωγής

Table 4: Σύγκριση Bulk Loading και Κλασικής Εισαγωγής

Μέθοδος Κατασκευής	**Χρόνος (ms)**	**Παρατηρήσεις**
Κλασική Εισαγωγή	655.77	Πολλά splits/reinserts
Bulk Loading	120.54	Ταχύτερη για μεγάλα N

5 Συμπεράσματα

- Η χρήση του R*-Tree προσφέρει δραματική επιτάχυνση σε range & kNN queries σε σχέση με σειριακή αναζήτηση.
- Το bulk loading είναι πολύ αποδοτικό για μεγάλη ποσότητα δεδομένων.
- Ο χρόνος για range/kNN queries αυξάνει υπογραμμικά με το μέγεθος της περιοχής ή το k αντίστοιχα, ενώ στη σειριακή αναζήτηση η αύξηση είναι γραμμική.
- Το skyline query ωφελείται επίσης σημαντικά από τη δομή, ειδικά σε μεγάλα σύνολα.