Πανεπιστήμιο Πειραιώς
Σχολή Τεχνολογιών Πληροφορικής και Επικοινωνιών
Τμήμα Ψηφιακών Συστημάτων
Δομές Δεδομένων 2017-2018 – 2η Εργασία

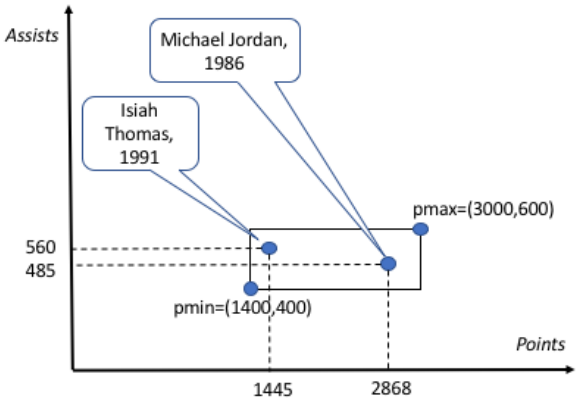
Χρήστος Δουλκερίδης Ορέστης Τελέλης

1 Περιγραφή

Σε μία Σχεσιακή Βάση Δεδομένων, m εγγραφές με n πεδία οργανώνονται σε έναν *Πίνακα* m γραμμών και n στηλών[1], κάθε γραμμή του οποίου αντιστοιχεί σε μία εγγραφή και κάθε στήλη αντιστοιχεί σε ένα πεδίο των εγγραφών. Η αναζήτηση εγγραφών των οποίων τα πεδία ικανοποιούν συγκεκριμένα κριτήρια μπορεί να γίνει ***(α)*** με *σειριακή σάρωση (scan)* των εγγραφών στον πίνακα και έλεγχο ικανοποίησης των κριτηρίων για καθεμία, ***(β)*** με αναζήτηση σε δενδρικές ευρετηριακές δομές που ορίζονται πάνω στις στήλες του πίνακα (πεδία των εγγραφών), προς εντοπισμό των εγγραφών που ικανοποιούν τα κριτήρια.

Στην εργασία θα προσομοιωθεί η αναζήτηση σε πίνακα m γραμμών (εγγραφών), με n αριθμητικές στήλες (πεδία), του υποσυνόλου των εγγραφών που βρίσκονται σε μία *περιοχή* του n-διάστατου χώρου, οριζόμενη από το χρήστη με τη μορφή επερώτησης (query). Ο πίνακας θα υλοποιηθεί ως δομή δεδομένων τύπου διδιάστατου πίνακα *(array)* $m \times n$, που προσφέρει λειτουργίες αναζήτησης μέσω σειριακής σάρωσης. Η δομή αυτή αναπαρίσταται από την κλάση με το όνομα `Records`.

Επιπλέον, σε μία κλάση με όνομα `Forest`, θα υλοποιηθεί μία πολυ-ευρετηριακή δομή, αποτελούμενη από n Δυαδικά Δέντρα Αναζήτησης (ΔΔΑ), ένα για κάθε στήλη του πίνακα. Το j-οστό ΔΔΑ, για $j = 0, \ldots, n-1$, θα επιτρέπει προσδιορισμό των γραμμών του πίνακα που ικανοποιούν δεδομένα κριτήρια στην j στήλη τους. Η κλάση `Forest` θα εξυπηρετεί αναζήτηση με χρήση ευρετηρίων. Στα πλαίσια της εργασίας θα συγκριθεί η απόδοση της αναζήτησης με χρήση ευρετηρίων έναντι της σειριακής σάρωσης.



Σχήμα 1: Παράδειγμα επερώτησης.

Παράδειγμα. Στο Σχήμα 1 απεικονίζονται δύο καλαθοσφαιριστές που αναπαρίστανται ως 2-διάστατα σημεία στο χώρο Πόντοι-Ασίστ, οι οποίοι θα ήταν αποτελέσματα μιας επερώτησης οριζόμενης από το 2-διάστατο ορθογώνιο κουτί που ορίζεται από τα σημεία $\langle p_{\min}, p_{\max} \rangle = \langle (1400, 400), (3000, 600) \rangle$. Αυτό αποτελεί και ένα παράδειγμα επερώτησης contains.

2 Ζητούμενα Εργασίας

2.1 Υλοποίηση

Θα υλοποιήσετε τις κλάσεις `Records` και `Forest`, που προδιαγράφονται στα αρχεία `Records.java` και `Forest.java`, που δίνονται. ***Μπορείτε να προσθέσετε μόνο `private` πεδία ή μεθόδους.***

[1] Οι πίνακες στις Σχεσιακές Βάσεις Δεδομένων αναπαριστούν n-μελείς *σχέσεις*, με τη μαθηματική έννοια.

2.1.1 Κλάση `Records`

Ένα αντικείμενο της κλάσης `Records` αποθηκεύει ένα σύνολο από m στο πλήθος n-διάστατα σημεία, σε έναν διδιάστατο $m \times n$ πίνακα `data`. Επιπλέον, περιλαμβάνει μεθόδους που θα υλοποιήσετε.

```
class Records {
    public long       nComp;
    public double[][] data;
    private double    euclidean(double[] p1, double[]   p2)
    public            Records(int m, int n)
    public double[]   getRecord(int i)
    public void       setRecord(int i, double[] rec)
    public Integer[]  containsScan(double[] pmin, double[] pmax)
    public Integer[]  rangeScan(double[] p, double r)
}
```

Ο κατασκευαστής `Records` και οι μέθοδοι `getRecord`, `setRecord`, `euclidean` δίνονται υλοποιημένες. Το πεδίο `nComp` θα χρησιμεύσει στον πειραματισμό σας, όπως εξηγείται παρακάτω. Θα υλοποιήσετε τις ακόλουθες μεθόδους.

`public Integer[] containsScan(double[] pmin, double[] pmax)` Επιστρέφει σε `Integer[]` τα αναγνωριστικά (αριθμούς γραμμής στον πίνακα `data`), των εγγραφών που βρίσκονται εντός n-διάστατου κουτιού οριζόμενου από τα σημεία `pmin` και `pmax`. Για τον σκοπό αυτόν, σαρώνει τις εγγραφές του πίνακα `data`. Επιστρέφει `null`, αν δεν υπάρχει καμιά τέτοια εγγραφή. Για την υλοποίηση της μεθόδου μπορεί να χρησιμοποιηθεί δομή δυναμικού πίνακα, π.χ.: `ArrayList<Integer>`.

`public Integer[] rangeScan(double[] p, double r)` Επιστρέφει σε `Integer[]` τα αναγνωριστικά (αριθμούς γραμμής στον πίνακα `data`), των εγγραφών που περιέχονται σε έναν κύκλο (μια υπερσφαίρα σε n διαστάσεις) με κέντρο το `p` και ακτίνα `r`. Για τον σκοπό αυτόν, σαρώνει τις εγγραφές του πίνακα `data`. Επιστρέφει `null` αν δεν υπάρχει καμιά εγγραφή εντός του κύκλου (υπερσφαίρας). Για την υλοποίηση της μεθόδου μπορεί να χρησιμοποιηθεί δομή δυναμικού πίνακα, π.χ.: `ArrayList<Integer>`. Για τον υπολογισμό της απόστασης δύο n-διάστατων σημείων θα χρησιμοποιηθεί η Ευκλείδεια απόσταση (μέθοδος `euclidean`, υλοποιημένη).

2.1.2 Κλάση `Forest`

Ένα αντικείμενο της κλάσης `Forest` ενθυλακώνει ***(α)*** ένα πίνακα *(array)* με όνομα `roots`, που αποθηκεύει n κόμβους-ρίζες Δυαδικών Δέντρων Αναζήτησης (ΔΔΑ), ***(β)*** ένα αντικείμενο `records` της κλάσης `Records`. Το j-οστό ΔΔΑ ορίζεται βάσει των τιμών της στήλης (πεδίου) j των εγγραφών που αποθηκεύονται στο `records`. Κάθε κόμβος του j-οστού ΔΔΑ, για $j = 0, 1, \ldots, n-1$, υλοποιείται από την κλάση `Node`, που δίνεται, και αποθηκεύει:

- στο πεδίο `dData`, την τιμή της στήλης j *ως κλειδί*, για μία εγγραφή του πίνακα που αναπαριστά η κλάση `Records`,
- στο πεδίο `id`, τον αριθμό (γραμμής) της εγγραφής στον πίνακα *ως αναγνωριστικό*, δηλαδή, τη θέση της εγγραφής στον πίνακα `data` εντός του αντικειμένου `records`.

```
class Forest {
    public long     nComp;
    private Node[]  roots;
    private Records records;
    public          Forest(int m, int n)
}
```

```
class Node {
    public int    id;
    public double dData;
    public Node   leftChild;
    public Node   rightChild;
}
```

Η κλάση `Forest` υλοποιεί δομή ευρετηριοποίησης n διαστάσεων που παρέχει τις ακόλουθες μεθόδους.

`public Forest(int m, int n)` Κατασκευάζει τον πίνακα `roots` που περιέχει n ρίζες ΔΔΑ, και το αντικείμενο `records`. Το πλήθος των εγγραφών m δίνεται μόνο για να είναι δυνατή η αρχικοποίηση του αντικειμένου `records`.

`public void load(String filename, int lines, int subDim)` Εισάγει δεδομένα σε μία δομή κλάσης `Forest` διαβάζοντας τις πρώτες `lines` εγγραφές ενός συνόλου δεδομένων από αρχείο με όνομα `filename`. Ουσιαστικά, εισάγει τα δεδομένα του αρχείου στις δομές `records` και `roots`. Αυτή η μέθοδος δίνεται υλοποιημένη για τη διευκόλυνσή σας. Βασίζεται στην `public` μέθοδο `insert` που επίσης δίνεται υλοποιημένη. Η `load` χρησιμοποιείται ως εξής:

```
Forest forest = new Forest(1000, 4); forest.load("NBA-5d-17265n.txt", 1000, 4);
```

για εισαγωγή των πρώτων 1000 εγγραφών και 4 διαστάσεων του αρχείου: `NBA-5d-17265n.txt`. Το αρχείο αυτό περιέχει συνολικά 17265 εγγραφές (συν μια εγγραφή επικεφαλίδα) και κάθε εγγραφή περιέχει 5 διαστάσεις (συν κάποια άλλα αλφαριθμητικά στοιχεία).

`public insert(int j, int id, double dd)` Η μέθοδος αυτή εισάγει στο j-οστό ΔΔΑ το αναγνωριστικό `id` μίας εγγραφής (αντιστοιχεί στη θέση της εγγραφής στον πίνακα `data` που ενθυλακώνει το αντικείμενο `records`), και την τιμή του j-οστού πεδίου, `dd`, για την εγγραφή αυτή, που λειτουργεί ως κλειδί για το j-οστό ΔΔΑ. Η μέθοδος δίνεται υλοποιημένη.

`public Integer[] findInterval(int j, double k1, double k2)` Αναζητά **στο j-οστό ΔΔΑ** και επιστρέφει τα αναγνωριστικά των εγγραφών που ανήκουν στο εύρος τιμών `[k1,k2]`.

`public Integer[] containsIndex(double[] pmin, double[] pmax)` Επιστρέφει σε `Integer[]` τα αναγνωριστικά των εγγραφών που βρίσκονται εντός ενός n-διάστατου κουτιού που ορίζεται από τα σημεία $pmin$ και $pmax$. Χρησιμοποιεί τα υπάρχοντα ΔΔΑ για αυτό το σκοπό. Ανακτά από καθένα ΔΔΑ τα αναγνωριστικά των εγγραφών που ικανοποιούν τον περιορισμό στη συγκεκριμένη διάσταση. Επιστρέφει `null`, αν δεν υπάρχει καμία τέτοια εγγραφή.

`private Integer[] intersect(Integer[][] results)` Επιστρέφει την τομή πολλών πινάκων ακεραίων αριθμών ή `null`, αν η τομή είναι κενή. Χρησιμοποιείται ως βοηθητική συνάρτηση για την εξέταση των επιμέρους αποτελεσμάτων που επιστρέφουν τα ΔΔΑ, ώστε να εντοπιστούν οι εγγραφές που ικανοποιούν τους περιορισμούς **σε όλες** τις διαστάσεις.

`public Integer[] rangeIndex(double[] p, double r)` Επιστρέφει σε `Integer[]` τα αναγνωριστικά των εγγραφών που περιέχονται εντός κύκλου (υπερσφαίρας, σε πολλές διαστάσεις) με κέντρο το p και ακτίνα r. Χρησιμοποιεί τα υπάρχοντα ΔΔΑ για τον σκοπό αυτόν. Επιστρέφει `null` αν δεν υπάρχει καμιά εγγραφή εντός του κύκλου (υπερσφαίρας αντίστοιχα). Για την υλοποίηση της μεθόδου `rangeIndex` μπορεί να χρησιμοποιηθεί δομή δυναμικού πίνακα, π.χ.: `ArrayList<Integer>`. Για τον υπολογισμό της απόστασης δύο σημείων σε χώρο n διαστάσεων, θα χρησιμοποιηθεί η Ευκλείδια απόσταση (δίνεται υλοποιημένη συνάρτηση `double euclidean(double[] p1, double[] p2)`).

2.2 Πειραματισμός και Τεχνική Αναφορά

Δίνεται το αρχείο εγγραφών `NBA-5d-17265n.txt`. Το αρχείο περιέχει πραγματικά δεδομένα που αφορούν καλαθοσφαιριστές του NBA και καταγράφουν την απόδοσή τους σε βάθος ετών (από το 1946) σε ετήσια βάση και χρησιμοποιώντας $n = 5$ διαστάσεις. Οι 5 διαστάσεις είναι:

`pts`:	συνολικοί πόντοι
`asts`:	ασίστς
`pf`:	προσωπικά φάουλ
`fgm`:	σύνολο καλαθιών εντός πεδιάς
`ftm`:	σύνολο ελευθέρων βολών

Σε αυτό το αρχείο εγγραφών θα εκτελέσετε αναζητήσεις για παίκτες με βάση κριτήρια αναζήτησης (για παράδειγμα ανάκτηση των καλαθοσφαιριστών με συνολικούς πόντους από 50 έως 100, και ασίστς από 100 έως 200), με χρήση των μεθόδων `containsIndex` και `rangeIndex`. Θα συγκρίνετε τα αποτελέσματα ως προς το πλήθος απαιτούμενων συγκρίσεων για την αναζήτηση, σε σχέση με τις `containsScan` και `rangeScan` αντίστοιχα. Κάθε κλήση καθεμίας εκ των παραπάνω μεθόδων, μπορεί να ενημερώνει το πεδίο `private long nComp` με το πλήθος των συγκρίσεων που χρειάστηκαν. Δε θα μετρηθεί το πλήθος των συγκρίσεων που απαιτούνται για τον υπολογισμό της τομής λιστών στη μέθοδο `intersect()`.

Θα συντάξετε μία τεχνική αναφορά, στην οποία θα δείξετε και θα σχολιάσετε τα αποτελέσματα του πειραματισμού σας. Στην αναφορά, θα περιλαμβάνονται πίνακες, όπου θα μετρηθεί η απόδοση των αλγορίθμων ως προς το πλήθος συγκρίσεων που απαιτούνται για την αναζήτηση των εγγραφών. Επιπλέον θα παράξετε ένα ραβδόγραμμα που απεικονίζει τα αποτελέσματα των πινάκων, χρησιμοποιώντας στον ψ-άξονα το πλήθος των συγκρίσεων που μετρήθηκαν. Αναμένονται τουλάχιστον 3 ραβδογράμματα όπου στον x-άξονα θα μεταβάλλεται: ***(α)*** το πλήθος των εγγραφών (m) (π.χ. m={1000,5000,10000,15000}), ***(β)*** το πλήθος των διαστάσεων (n) (π.χ. n={1,2,3,4,5}), και (γ) η ακτίνα r (π.χ. r={50,75,100,125}).

Η τεχνική αναφορά θα περιλαμβάνει συνοπτική περιγραφή υλοποίησης, για καθεμία από τις μεθόδους `findInterval`, `containsIndex`, `intersect`, `rangeIndex`, της κλάσης `Forest` – με ψευδοκώδικα ή/και επεξηγήσεις όπου χρειάζεται. Με βάση την περιγραφή σας, θα αναλύσετε την πολυπλοκότητα της υλοποίησής σας για καθεμία από αυτές τις μεθόδους, ως προς τις παραμέτρους: *πλήθος εγγραφών* m, *διάσταση σημείων* n. Συμπερασματικά, θα εξάγετε και θα αιτιολογήσετε την πολυπλοκότητα της υλοποίησης του αλγορίθμου `rangeIndex` και θα τη συγκρίνετε με αυτή του `rangeScan`.

3 Διαδικαστικά Θέματα

ΠΑΡΑΔΟΤΕΑ Θα πρέπει να παραδώσετε ένα αρχείο AM_Επώνυμο_Όνομα.zip (όπου ΑΜ ο αριθμός μητρώου) που θα περιλαμβάνει **τρία (3) αρχεία μόνο:**

1. Το αρχείο πηγαίου κώδικα **Forest.java**, επαρκώς σχολιασμένο.
   Προσοχή: σχόλια στα ελληνικά, μόνο σε κωδικοποίηση UTF-8. Άλλες κωδικοποιήσεις (ISO-8859-7, Windows-1253) *απορρίπτονται από τον Java Compiler*. Eclipse/Netbeans: δεξί click στο Project, επιλογή Properties και στην ιδιότητα «Encoding» επιλέγετε UTF-8.
2. Το αρχείο πηγαίου κώδικα **Records.java**, επαρκώς σχολιασμένο.
3. Την τεχνική αναφορά σας **σε μορφή pdf**. **ΘΑ ΑΓΝΟΗΘΕΙ** *κάθε άλλη μορφή εγγράφου*.

ΠΑΡΑΔΟΣΗ αποκλειστικά μέσω της πλατφόρμας «ΕΥΔΟΞΟΣ» του τμήματος, έως και **14/1/2018, 23:59**. Ανεβάστε το AM_Επώνυμο_Όνομα.zip στην περιοχή «Εργασίες».

ΕΡΩΤΗΣΕΙΣ/ΑΠΟΡΙΕΣ/ΔΙΕΥΚΡΙΝΙΣΕΙΣ αποκλειστικά μέσα από την Περιοχή Συζήτησης «ΕΡΓΑΣΙΑ 2» της πλατφόρμας «ΕΥΔΟΞΟΣ». Δε θα απαντηθούν Email με απορίες.

ΔΕ ΘΑ ΑΞΙΟΛΟΓΗΘΟΥΝ:

- Εργασίες που θα παραδοθούν εκπρόθεσμα ή με άλλο τρόπο (email, CD κ.λ.π.)
- Εργασίες που παραδίδονται σε μορφή συμπίεσης διαφορετική από .zip.
- Εργασίες που περιλαμβάνουν μέσα άσχετα αρχεία.

ΒΑΘΜΟΛΟΓΗΣΗ Η εργασία μετρά 20% στον τελικό βαθμό (εφόσον το γραπτό σας βαθμολογηθεί με τουλάχιστον 4).

- Φτωχή / Δυσανάγνωστη τεκμηρίωση ή έλλειψη αυτής στην αναφορά για κάποια λειτουργία, επιφέρει άμεση απώλεια 50% του μέγιστου δυνατού βαθμού για τη λειτουργία αυτή.
- Η τεχνική αναφορά ή τμήματα αυτής δε θα ληφθούν υπόψην στη βαθμολόγηση αν δε συνοδεύονται από κώδικα που μεταγλωττίζεται και εκτελείται.
- **Η εργασία είναι αυστηρά ατομική.** Αντιγραφή στον κώδικα και/ή στην τεχνική αναφορά επιφέρει άμεσο μηδενισμό της εργασίας. Οι εργασίες θα ελεγχθούν (όλες ανά ζεύγη).