Aufgabenblatt 6 (Teamaufgabe)

Einführung in die Programmierung 2
LVA-Nr. 185.A92
2018 S
TU Wien

Thema:
Suche in mehrdimensionalen Daten

Aufwand:
ca. 16 Stunden pro Person

Ausgabe:
9. 4. 2018

Feedback (Deadline):
25. 5. 2018

Abgabe (Deadline):
4. 6. 2018, 6:00 Uhr
Lösungen hochladen

Einführung

In vielen Gebieten der Informatik (Geoinformationssysteme, Künstliche Intelligenz, Computergraphik etc.) ist es wichtig, mehrdimensionale Daten effizient verwalten zu können.

Für dieses Aufgabenblatt wird Ihnen eine Datei (`junctions.csv`) mit den Koordinaten von Bahnhöfen und Flughäfen zur Verfügung gestellt.

Ihre Aufgabe besteht darin, die Daten aus `junctions.csv` einzulesen und die unter "Abfragen" spezifizierten Abfragen zu implementieren.

Datenformat

Verwenden Sie die Klasse `Scanner` um die Daten aus der mitgelieferten Datei `junctions.csv` einzulesen. Initialisieren Sie das `Scanner`-Objekt auf folgende Weise um die Datei auf die gleiche Weise einlesen zu können wie Daten aus der Standardeingabe:

```
try(Scanner s = new Scanner(
        new File(System.getProperty("user.dir") +
            "/data/junctions.csv"), "UTF-8")) {
    // Benutzen Sie das Scanner-Objekt s hier
} catch(FileNotFoundException e) {
    // junctions.csv wurde nicht gefunden
    System.exit(1);
}
```

Tipp: Machen Sie sich mit der Methode `useDelimiter` in `Scanner` vertraut um das Einlesen zu vereinfachen.

In der Datei steht jeder Datensatz in einer eigenen Zeile. Innerhalb einer Zeile werden die verschiedenen Dateneinträge mit einem Semikolon (`;`) getrennt. Die Einträge sind in folgender Reihenfolge:

```
Name;x-Koordinate;y-Koordinate;Typ
```

Zum Beispiel:

```
Vienna International Airport;1844.53060;5801.04868;AIRPORT
Wien Westbahnhof;1818.54657;5813.29982;TRAINSTATION
```

Die Daten kommen aus der echten Welt und wurden in die euklidische Ebene projiziert. Lösen Sie die Aufgabe in dieser Ebene.

Abfragen

Implementieren Sie eine Methode, die für einen beliebigen Punkt in der euklidischen Ebene und einen Radius die Anzahl der Flughäfen und Bahnhöfe berechnet, die sich innerhalb dieses Radius befinden.

Geben sich die Anzahlen *separat* zurück.

Benutzen Sie diese Methode, um eine weitere Methode zu implementieren, die die Anzahl aller Flughäfen berechnet, in deren `r`-Längeneinheiten-Umkreis sich mindestens `n` Bahnhöfe befinden (`r` und `n` sollen Parameter der Methode sein).

Implementieren Sie die beiden Abfragen zunächst mit einer naiven Datenstruktur (z.B. einem Array oder einer Liste) und danach mit einer für große Datenmengen effizienteren Datenstruktur (z.B. einer der unter "Datenstrukturen" beschriebenen). Vergleichen Sie die beiden Ansätze bezüglich ihres Laufzeitverhaltens miteinander.

Implementieren Sie den naiven Ansatz und *mindestens* einen effizienteren.

Datenstrukturen

Es gibt in der Literatur verschiedene Datenstrukturen mit denen die Anfragen implementiert werden können. Drei unserer Empfehlungen finden Sie im Folgenden kurz beschrieben. Arbeiten Sie außerdem die allgemeinen und spezifischen Fragen aus, sodass Sie sie in der Übungseinheit 6 gegebenenfalls beantworten können.

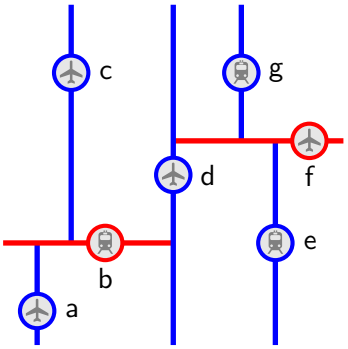
Wenn Sie eine andere Datenstruktur implementieren, werden Ihnen natürlich ebenfalls Fragen dazu gestellt.

Allgemeine Fragen

- Wieso haben Sie sich für die verwendete Datenstruktur entschieden?
- Wie verhält sich diese Datenstruktur bezüglich verschiedener Eingabedaten? (gleichmäßig verteilt, um wenige Punkte konzentriert, etc.)
- Wie verhält sich der Implementierungsaufwand für die Konstruktion der Datenstruktur zum Implementierungsaufwand für die Abfrage?
- Wie verhält sich der Laufzeitaufwand von Konstruktion und Abfrage?
- Wie verhalten sich Implementierungsaufwand und Laufzeit von naiver und effizienter Lösung zueinander?

2D-Baum

Die Konstruktion eines 2D-Baumes funktioniert ähnlich wie bei einem gewöhnlichen binären Suchbaum, mit der Ausnahme, dass horizontal und vertikal orientierte Knoten unterschieden werden. Analog zum gewöhnlichen binären Suchbaum teilt ein Knoten den unter ihm liegenden Suchraum in jene Elemente die einen kleineren, respektive einen größeren, Wert als der Knoten haben. Speziell am 2D-Baum ist, dass der Wert, der zur Teilung des Suchraums benutzt wird, zwischen der X- und Y-Koordinate alterniert.



Ein Interface kann den Umgang mit mehreren Arten von Knoten erleichtern (Vorlesungen am 17. und 23. April).

Fragen zum 2D-Baum

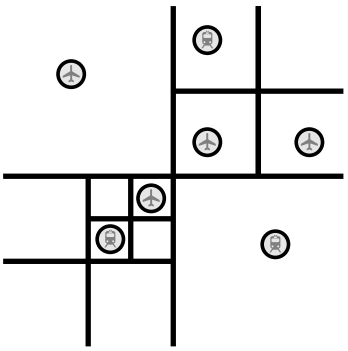
- Haben Sie darauf geachtet, dass ihr 2D-Baum balanciert ist? Falls ja: wie haben Sie die Balancierung implementiert?
- Welche Auswirkungen kann es haben, wenn ein Suchbaum unbalanciert ist?

Literatur zum 2D-Baum

- Folien 14-25:
  `https://algs4.cs.princeton.edu/lectures/99GeometricSearch-2x2.pdf`
- Video:
  `https://algs4.cs.princeton.edu/lectures/99DemoKdTree.mov`
- Wikipedia: `https://en.wikipedia.org/wiki/K-d_tree`

Quadtree

Ein Quadtree unterscheidet sich von einem binären Suchbaum dadurch, dass ein Nicht-Blatt-Knoten vier Kindknoten hat, die gleich große Quadranten repräsentieren; es gibt also im Gegensatz zu einem Suchbaum keinen Schlüsselwert, der den Suchbereich in links/rechts bzw. oben/unten teilt. In Quadtree-Knoten befindet sich entweder maximal ein Datensatz (Blattknoten), oder ein weiterer Nicht-Blattknoten, der wiederum in vier feiner aufgelöste Unterquadranten unterteilt ist.



Fragen zum Quadtree

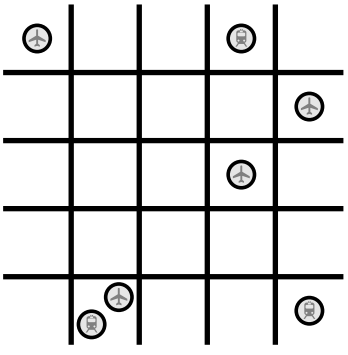
- Welche Art von Datensätzen können beim Quadtree Probleme verursachen? Auf was müssen Sie besonders achten?
- Wie verhält sich der von Ihnen implementierte Quadtree in Bezug auf seine Balancierung?

Literatur zum Quadtree

- Beschreibung: `https://www.geeksforgeeks.org/quad-tree/`
- Visualisierung: `http://jimkang.com/quadtreevis/`

2D-Grid

Für ein 2D-Grid wird der betrachtete Bereich in gleichmäßige Flächen unterteilt. Jeder Fläche werden die in ihr enthaltenen Datensätze in einer nicht-hierarchischen Datenstruktur (etwa einem Array oder einer Liste) gespeichert. Um ein Element im 2D-Grid zu finden, wird zuerst aus den Abfrage-Koordinaten die Fläche berechnet, in der sich der gesuchte Datensatz befinden könnte. Danach wird die der Fläche zugeordnete nicht-hierarchische Datenstruktur durchsucht.



Fragen zum 2D-Grid

- Wie wird bei Ihrer Implementierung die Auflösung des Grids bestimmt?

- Welche Auswirkungen hat eine zu hohe/niedrige Auflösung des Grids?
- Welche nicht-hierarchische Datenstruktur verwenden Sie? Warum?

Literatur zum 2D-Grid

- Folien 14-19:
  `https://algs4.cs.princeton.edu/lectures/99GeometricSearch-2x2.pdf`
- Wikipedia: `https://de.wikipedia.org/wiki/Gridfile`

Vorgehensweise

Die wichtigsten Kenntnisse, auf denen die Teamaufgabe aufbaut, sollten Sie bereits erworben haben oder in den nächsten Vorlesungseinheiten erwerben (baumförmige Datenstrukturen). Es wird erwartet, dass Sie sich darauf aufbauend selbständig weitergehendes Wissen über den effizienten Umgang mit zweidimensionalen Daten erarbeiten. Dieses Thema wird in künftigen Vorlesungen nicht behandelt. **Fangen Sie so bald wie möglich an, um eventuell auftretende Fragen rechtzeitig klären zu können.**

nicht zuwarten

Manche in künftigen Vorlesungen eingeführte Techniken könnten Details der Implementierung verbessern. Diese dürfen natürlich zum Einsatz kommen, es gibt aber keine Verpflichtung dazu. Essenzielle Schwierigkeiten in der Aufgabenstellung werden davon wahrscheinlich nicht betroffen sein. Die Details zum Einlesen von Dateien werden erst nach der Abgabe behandelt.

Dies ist eine Teamaufgabe: Arbeiten sie an jedem Aspekt der Aufgabe gemeinsam; optimalerweise am gleichen Ort und zur gleichen Zeit. Arbeiten Sie in den selben Teams wie bei den Ad-Hoc-Aufgaben (sollte das nicht möglich sein, suchen Sie sich eine Partnerin oder einen Partner aus der selben Übungsgruppe).

Teams wie bei Ad-hoc-Aufgaben; im Team lösen

Um die volle Punktzahl (33) zu erreichen, müssen Sie spätestens bis 25. 5. 2018 Feedback bei der Tutorin oder beim Tutor Ihrer Übungsgruppe einholen und bis zur Abgabe einarbeiten – je früher, desto besser. Wenn Sie kein Feedback einholen, können Sie höchstens die halbe Punktzahl erreichen.

Feedback einholen, Mail an Tutorin/Tutor

Falls es im Team Probleme gibt (wenn zum Beispiel ein Teil des Teams die Übung abbricht oder nicht mitarbeitet), wenden Sie sich bitte ebenfalls möglichst frühzeitig an die Tutorin oder den Tutor Ihrer Übungsgruppe.

Mail an Tutorin/Tutor bei Problemen jeder Art

Für inhaltliche Fragen steht Ihnen das Diskussionsforum in TUWEL und das Programmiercafé zur Verfügung.

Beispielanfragen

```
Junctions less than 575.0 units from x=0.0 y=0.0
  > Airports: 0   Trainstations: 0
Junctions less than 100.0 units from x=1818.54657 y=5813.29982
  > Airports: 7   Trainstations: 536
Airports with at least  5 Trainstations less than  1.0 units away
  > 1
Airports with at least 20 Trainstations less than 15.0 units away
  > 293
```