Übungen zu Algorithmen
WSI für Informatik
Kaufmann/Bekos/Schneck

Sommersemester 2018
Universität Tübingen
08.06.2018

Übungsblatt 7

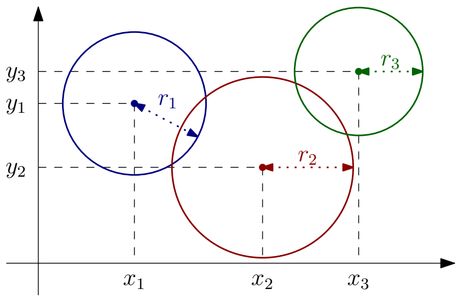
Abgabe bis 14.06.2018
Besprechung: 18.06.2018 – 21.06.2018

Aufgabe 1: Anwendung von AVL-Bäumen (8 Punkte)

Betrachten Sie folgendes Problem: Gegeben eine Menge C von Kreisen:

$$C = \{c_i = (x_i, y_i, r_i) \mid i = 1, \dots, n\}.$$

Dabei ist (x_i, y_i) der Mittelpunkt und $r_i > 0$ der Radius des Kreises c_i. Sie können annehmen, dass sich keine drei Kreise in einem Punkt schneiden und dass es keinen Kreis gibt, der vollkommen in einem anderen Kreis enthalten ist.



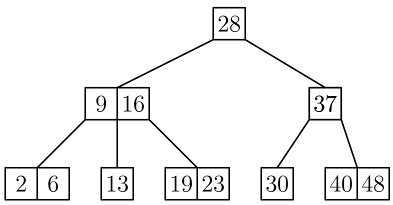
Gefragt ist nun nach dem *Schnittproblem*, das heißt nach allen Schnittpunkten der Kreise in C. Für Ihre Analyse dürfen Sie annehmen, dass die Schnittpunkte von zwei Kreisen mit einer gegebenen Funktion `Schnitt(`c_i`,`c_j`)` in Zeit $\mathcal{O}(1)$ berechnet werden können.

Geben Sie einen *Plane-Sweep* Algorithmus für das Schnittproblem in Pseudocode an. Halten Sie sich dabei an folgende Hilfestellungen: Verwenden Sie eine X-Struktur und eine Y-Struktur (beides AVL-Bäume), die „wichtige" x- und y-Werte enthalten.

Geben Sie die Laufzeit Ihres Algorithmus an und begründen Sie die Korrektheit Ihres Algorithmus.

Aufgabe 2: B-Bäume (2 + 4 Punkte)

In dieser Aufgabe geht es um B-Bäume der Ordnung 2, d. h. ein innerer Knoten hat mindestens zwei und höchstens drei Kinder. Gegeben ist der folgende B-Baum der Ordnung 2:



(a) Fügen Sie die folgenden Elemente in den gegebenen B-Baum ein: Zuerst 14 und dann 1.

(b) Löschen Sie die folgenden Elemente aus dem B-Baum, den Sie in Teil (a) erhalten haben: 37, 40, 19, 23 (in dieser Reihenfolge).

Zeigen Sie insbesondere deutlich, was bei den Spalt- und eventuellen Verschmelzungsvorgängen passiert.

Aufgabe 3: (2,4)-Bäume (5 + 5 Punkte)

(a) Seien S und T zwei (2,4)-Bäume, wobei alle Elemente aus S kleiner sind als die Elemente aus T, d.h. $\max S < \min T$. Geben Sie einen effizienten Pseudocode-Algorithmus `merge(`S, T`)` an, der einen (2,4)-Baum aus der Vereinigung von S und T erstellt.

(b) Geben Sie einen effizienten Pseudocode-Algorithmus `split(`T, x`)` an, der einen (2,4)-Baum T am Wert x in zwei (2,4)-Bäume $T_{\leq x}$ und $T_{>x}$ aufspaltet. Dabei soll $T_{\leq x}$ alle Elemente aus T enthalten, die kleiner oder gleich x sind und $T_{>x}$ alle Elemente aus T, die größer als x sind.

Begründen Sie jeweils die Korrektheit und Laufzeit Ihres Algorithmus.

Aufgabe 4: Amortisierte Analyse (2 + 4 Punkte)

Eine *Queue* ist eine Datenstruktur, die die beiden Operationen `enqueue` und `dequeue` unterstützt:

- `enqueue` zum Hinzufügen eines Objekts und
- `dequeue` zum Zurückholen und Entfernen eines Objektes.

Dabei wird nach dem Prinzip *first-in-first-out* gearbeitet, das heißt, es wird von `dequeue` immer das Objekt aus der Queue zurückgegeben, welches von den in der Queue noch vorhandenen Objekten als erstes mit `enqueue` hineingelegt wurde.

Ein *Stack* ist eine Datenstruktur, die die drei Operationen `push`, `pop` und `empty` unterstützt:

- `push` zum Hinzufügen eines Objekts,
- `pop` zum Zurückholen und Entfernen eines Objektes, und
- `empty` zum Prüfen, ob der Stack leer ist.

Dabei wird nach dem Prinzip *last-in-first-out* gearbeitet, das heißt, es wird von `pop` immer das Objekt aus dem Stack zurückgegeben, welches von den in dem Stack noch vorhandenen Objekten als letztes mit `push` hineingelegt wurde.

(a) Beschreiben Sie, wie man eine Queue mit Hilfe von zwei Stacks simulieren kann.

(b) Zeigen Sie, dass die amortisierte Laufzeit für jede `enqueue`- und `dequeue`-Operation von dieser simulierten Queue $\mathcal{O}(1)$ ist.