Geometrische Datenstrukturen: Range Tree

Elmar Langetepe
University of Bonn

k-dimensionaler Range Tree

Theorem 3.11: Ein k-dimensionaler Bereichsbaum für n Punkte im $\mathbb{R}^k$ kann in Zeit $O(n(\log n)^{k-1})$ mit Platz $O(n(\log n)^{k-1})$ aufgebaut werden. Eine Bereichsanfrage mit Hyperrechteck $q \subset \mathbb{R}^k$ kann in Zeit $O(a + (\log n)^k)$ beantwortet werden. Dabei ist a die Größe der Antwort.

Beweis: Speicherplatz Punkt $p = (x_1, x_2, \ldots, x_k)$

- Baum T^1:
  1) Im Baum T^1 einmal
  2) In max. $2 \times \log n$ vielen Intervallen $I(v_i)$ von T^1
- Bäume $T^{k-1}_{v_i}$, $i = 1, \ldots, l$
  1) In Baum $T^{k-1}_{v_i}$ einmal
  2) In max. $2 \times \log n$ vielen Intervallen $I(w_j)$ von $T^{k-1}_{v_i}$

- Insgesamt: $\sum_{i=0}^{k-1}(\log n)^i \in O((\log n)^{k-1})$
- Alle Punkte $O(n(\log n)^{k-1})$

k-dimensionaler Range Tree

Theorem 3.11: Ein k-dimensionaler Bereichsbaum für n Punkte im $\mathbb{R}^k$ kann in Zeit $O(n(\log n)^{k-1})$ mit Platz $O(n(\log n)^{k-1})$ aufgebaut werden. Eine Bereichsanfrage mit Hyperrechteck $q \subset \mathbb{R}^k$ kann in Zeit $O(a + (\log n)^k)$ beantwortet werden. Dabei ist a die Größe der Antwort.

Beweis: Query! $q = (I_1, I_2, \ldots, I_k)$

- T^1: Intervalle $I(v_i)$ die I_1 ausschöpfen max. $2 \log n$ viele
- Induktiv: $O((\log n_i)^{k-1} + a_i)$ für $T^{k-1}_{v_i}$, Ind. Anfang: $k = 2$ klar, Ind. Schluss: T^k

  Bereichsanfrage für alle $T^{k-1}_{v_i}$:

  $C \cdot \log n + \sum_{i=1}^{2 \log n} (\log n_i)^{k-1} + a_i \in O((\log n)^k + a)$

  Punkte in den Intervallen $I(v_i)$ disjunkt: $\sum a_i = a$

k-dimensionaler Range Tree

Theorem 3.11: Ein k-dimensionaler Bereichsbaum für n Punkte im $\mathbb{R}^k$ kann in Zeit $O(n(\log n)^{k-1})$ mit Platz $O(n(\log n)^{k-1})$ aufgebaut werden. Eine Bereichsanfrage mit Hyperrechteck $q \subset \mathbb{R}^k$ kann in Zeit $O(a + (\log n)^k)$ beantwortet werden. Dabei ist a die Größe der Antwort.

Beweis: Aufbau!

Prioritätssuchbaum

Prioritätssuchbaum

- Einfache Struktur für Punkte in der Ebene

Prioritätssuchbaum

- Einfache Struktur für Punkte in der Ebene
- Halbstreifenanfrage $H = [x_1, x_2] \times (-\infty, y]$

Prioritätssuchbaum

- Einfache Struktur für Punkte in der Ebene
- Halbstreifenanfrage $H = [x_1, x_2] \times (-\infty, y]$
- Finde Punkte aus D in H

Prioritätssuchbaum

- Einfache Struktur für Punkte in der Ebene
- Halbstreifenanfrage $H = [x_1, x_2] \times (-\infty, y]$
- Finde Punkte aus D in H
- Platzoptimal und effizient!

Prioritätssuchbaum

- Einfache Struktur für Punkte in der Ebene
- Halbstreifenanfrage $H = [x_1, x_2] \times (-\infty, y]$
- Finde Punkte aus D in H
- Platzoptimal und effizient!
- Eindimensionaler Bereichsbaum für X Koordinaten

Prioritätssuchbaum

- Einfache Struktur für Punkte in der Ebene
- Halbstreifenanfrage $H = [x_1, x_2] \times (-\infty, y]$
- Finde Punkte aus D in H
- Platzoptimal und effizient!
- Eindimensionaler Bereichsbaum für X Koordinaten
- Heap für Y-Koordinaten

Prioritätssuchbaum

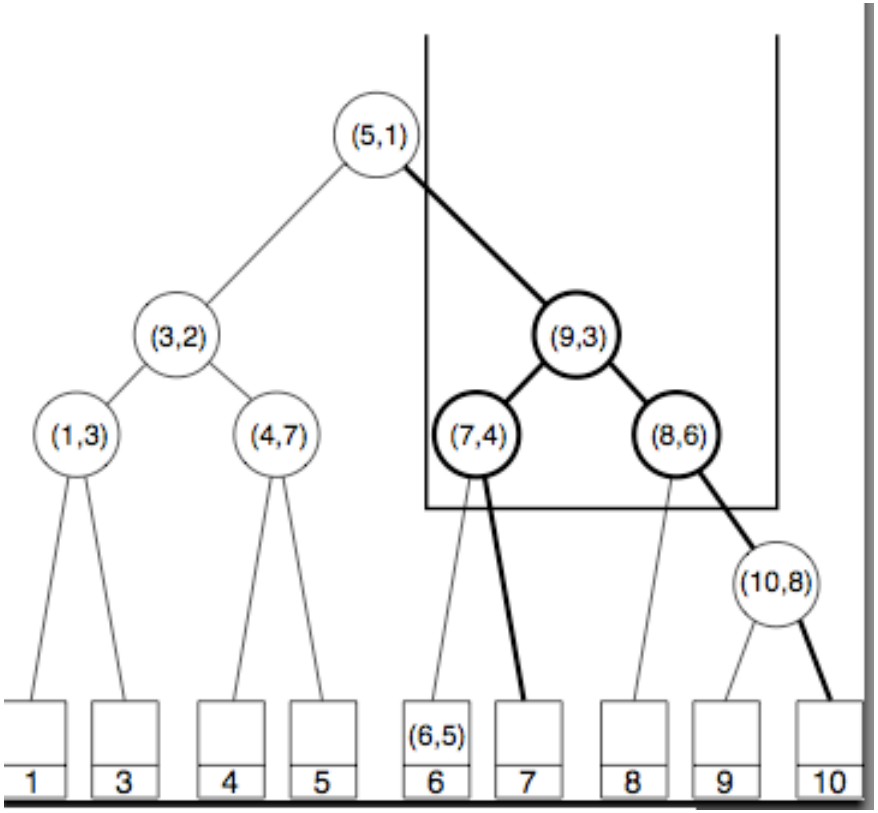
- Einfache Struktur für Punkte in der Ebene
- Halbstreifenanfrage $H = [x_1, x_2] \times (-\infty, y]$
- Finde Punkte aus D in H
- Platzoptimal und effizient!
- Eindimensionaler Bereichsbaum für X Koordinaten
- Heap für Y-Koordinaten
- 3 Bedingungen

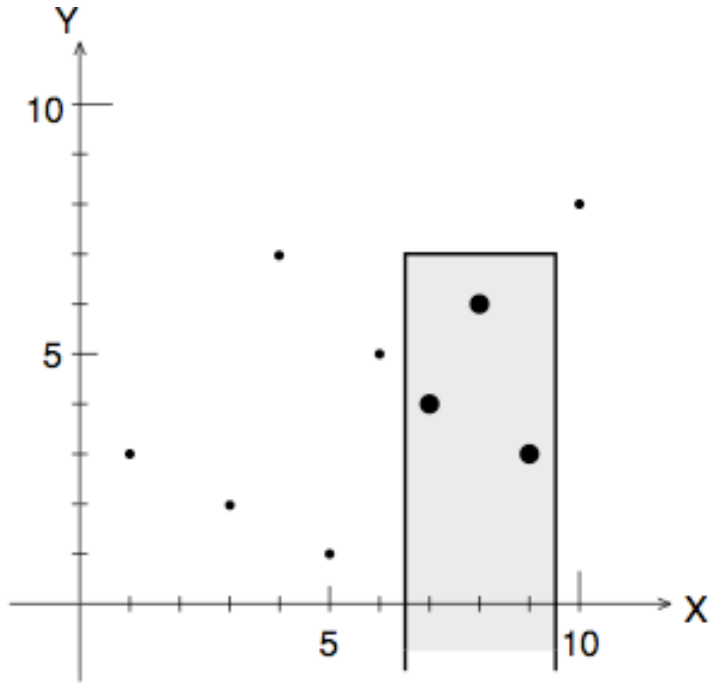
Bedingungen: Prioritätssuchbaum

Bedingungen: Prioritätssuchbaum

- 1. Jeder Punkt auf dem Weg zu seiner X-Koordinate
- 2. Punkte entlang des Pfades nach Y-Koordinaten (Heap)
- 3. So nah wie möglich an der Wurzel

Prioritätssuchbaum Beispiel





Suchbaum für X-Koordinate, Heap für Y-Koordinate

Ergebnis

Ergebnis

Theorem 3.14 Ein Prioritätssuchbaum für n Punkte in der Ebene kann in Zeit $O(n \log n)$ aufgebaut werden. Er benötigt $O(n)$ viel Platz. Eine Halbstreifenanfrage kann in Zeit $O(a + \log n)$ beantwortet werden. Dabei ist a die Größe der Antwort.

Ergebnis

Theorem 3.14 Ein Prioritätssuchbaum für n Punkte in der Ebene kann in Zeit $O(n \log n)$ aufgebaut werden. Er benötigt $O(n)$ viel Platz. Eine Halbstreifenanfrage kann in Zeit $O(a + \log n)$ beantwortet werden. Dabei ist a die Größe der Antwort.

Beweis: Geht das immer?, Aufbau, Query!

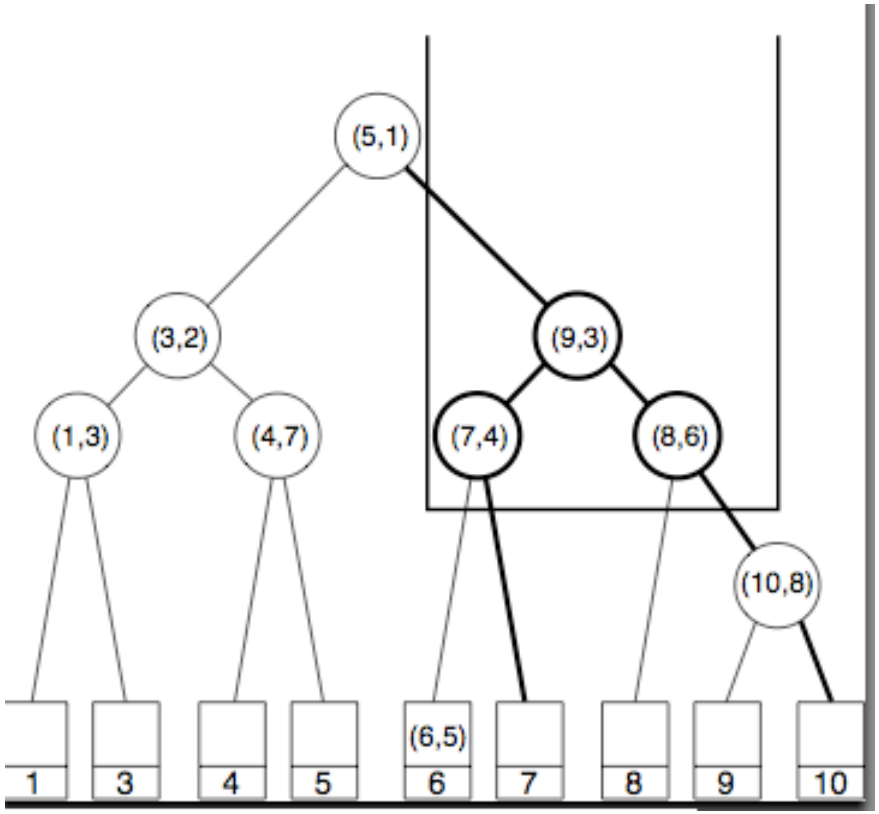
Ergebnis

Theorem 3.14 Ein Prioritätssuchbaum für n Punkte in der Ebene kann in Zeit $O(n \log n)$ aufgebaut werden. Er benötigt $O(n)$ viel Platz. Eine Halbstreifenanfrage kann in Zeit $O(a + \log n)$ beantwortet werden. Dabei ist a die Größe der Antwort.

Beweis: Geht das immer?, Aufbau, Query!

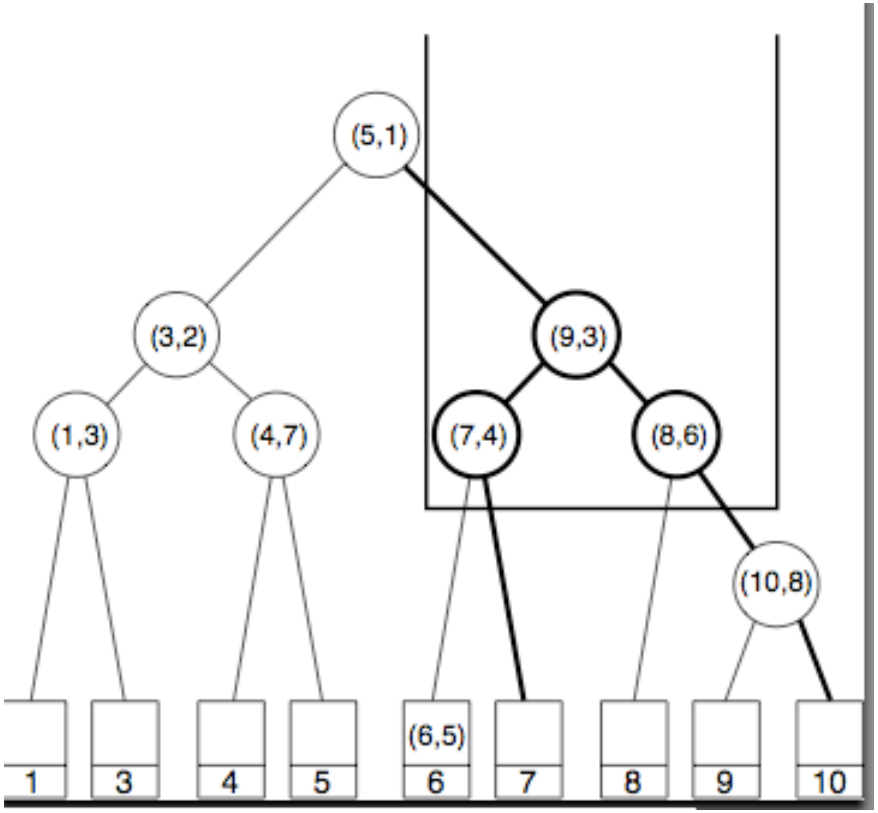
- X-sortiertes Skelett, nach aufsteigenden Y-Koordinaten einfügen
- 1. Jeder Punkt auf dem Weg zu seiner X-Koordinate
- 2. Punkte entlang des Pfades nach Y-Koordinaten (Heap)
- 3. So nah wie möglich an der Wurzel
- Induktiv: Wurzel, Teilbäume v_1, v_2

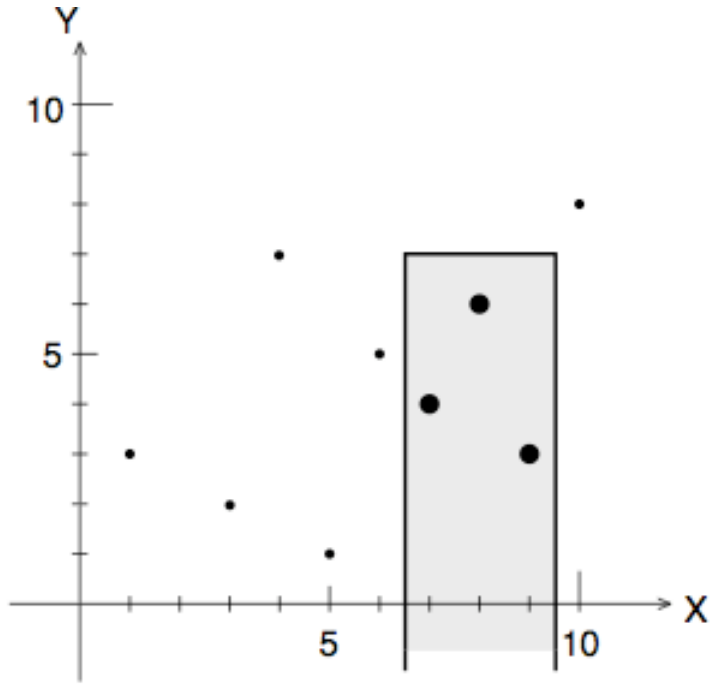
Prioritätssuchbaum Aufbau, Platz



X-sortiertes Skelett und dann nach aufsteigenden Y-Koordinaten einfügen.

Prioritätssuchbaum Query





Anfrage: $[6.5, 9.5] \times [-\infty, 7)$: X-Grenzen und nie zu tief!

Ergebnis

Ergebnis

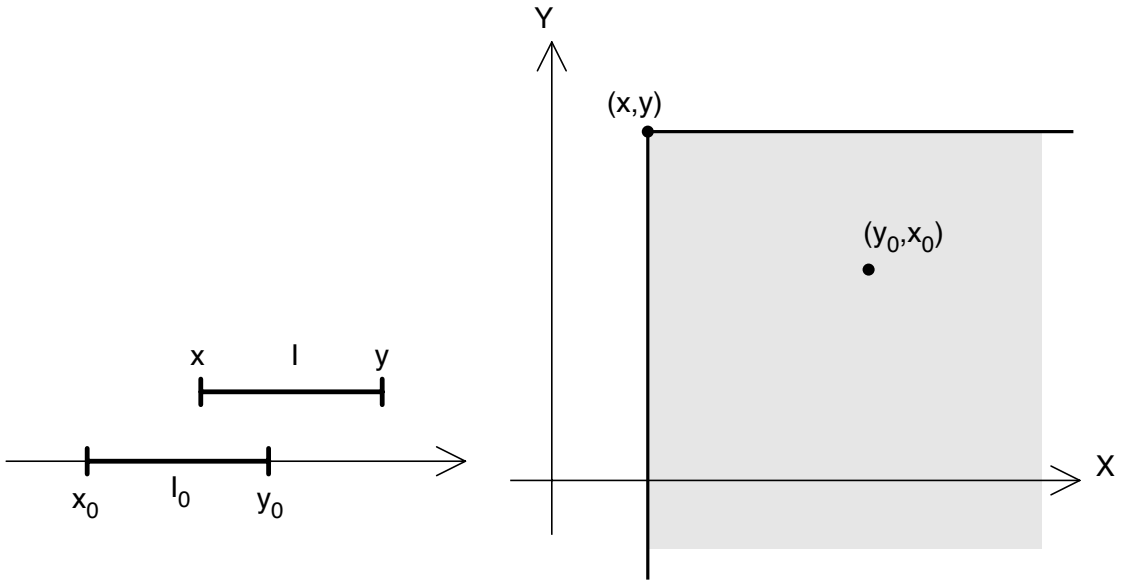
Theorem 3.14 Ein Prioritätssuchbaum für n Punkte in der Ebene kann in Zeit $O(n \log n)$ aufgebaut werden. Er benötigt $O(n)$ viel Platz. Eine Halbstreifenanfrage kann in Zeit $O(a + \log n)$ beantwortet werden. Dabei ist a die Größe der Antwort.

Ergebnis

Theorem 3.14 Ein Prioritätssuchbaum für n Punkte in der Ebene kann in Zeit $O(n \log n)$ aufgebaut werden. Er benötigt $O(n)$ viel Platz. Eine Halbstreifenanfrage kann in Zeit $O(a + \log n)$ beantwortet werden. Dabei ist a die Größe der Antwort.

Beweis: Query!

Anwendung Schnittanfrage mit Intervallen



Lemma 3.16 Seien $I_0 = [x_0, y_0]$ und $I = [x, y]$ zwei Intervalle, dann gilt: I_0 überlappt mit $I \iff x \leq y_0$ und $x_0 \leq y$

Ergebnis Schnittanfrage

Ergebnis Schnittanfrage

Theorem 3.17 Man kann n Intervalle mit Platz $O(n)$ so abspeichern, dass sich eine Überlappungsanfrage eines Intervalls I_0 in Zeit $O(a + \log n)$ beantworten läßt. Dabei ist a die Größe der Antwort.

Ergebnis Schnittanfrage

Theorem 3.17 Man kann n Intervalle mit Platz $O(n)$ so abspeichern, dass sich eine Überlappungsanfrage eines Intervalls I_0 in Zeit $O(a + \log n)$ beantworten läßt. Dabei ist a die Größe der Antwort.

Intervalle in Punkte übertragen.
Anfrage mit Viertelebene $[x_0, \infty) \times (-\infty, y_0]$.