Algorithmische Geometrie
Vorlesung vom 15.06.05

In der letzten Vorlesung bereits Bereichsbäume für den zweidimensionalen Raume betrachtet. Diese Überlegungen können zu *d*-dimensionalen Bereichsbäumen verallgemeinert werden.

_d_-dimensionale Bereichsbäume

Sei $S \subset R^d$, $|S| = n$
Man kann nun zwei Fälle bezüglich der Dimension für die Definition eines *d*-dimensionalen Bereichsbaums unterscheiden:

$d > 1$:

Für eine Dimension $d > 1$ besteht die Primärstruktur aus einem balancierten binären Suchbaum bezüglich der x_1-Koordinaten von S.
Jeder Knoten v in diesem Suchbaum enthält einen Zeiger auf eine Sekundärstruktur: dies ist ein (d-1)-dimensionaler Bereichsbaum für die Koordinaten ($x_2, x_3, \ldots, x_d$) der Punkte aus S, die im Unterbaum mit der Wurzel v der Primärstruktur liegen.

$d = 1$:

Bei nur einer Dimension besteht der zugehörige Bereichsbaum aus dem üblichen balancierten binären Suchbaum.

Platzbedarf eines _d_-dimensionale Bereichsbaums

Jeder Knoten v der Primärstruktur (d.h. der zugehörige Punkt $\in S$) ist in der Sekundärstruktur seiner $\leq \log n$ Vorgänger enthalten. Entsprechend sind die Punkte zu Knoten der Sekundärstruktur in der Tertiärstruktur aller ihrer Vorgänger enthalten usw.
Es ergibt sich daraus, dass es in der m-ären Struktur für jeden Punkt $\leq \log^{m-1} n$ Knoten gibt.

Insgesamt ist der Platz für alle Strukturen demnach wie folgt abschätzbar, wobei sich die geometrische Reihe $n \cdot \left(\log^0 n + \log^1 n + \ldots + \log^{d-1} n\right)$ in $\frac{\log^d - 1}{\log n - 1}$ umformbar ist:

$$= \left(n \cdot \underbrace{\left(\log^0 n + \log^1 n + \ldots + \log^{d-1} n\right)}_{\substack{= \frac{\log^d - 1}{\log n - 1} \\ = O(\log^{d-1} n)}} \right)$$

$$= O(n \cdot \log^{d-1} n)$$

Vorverarbeitungszeit für einen *d*-dimensionale Bereichsbaum

Sei *T(n,d)*, wobei *n* = Anzahl der Knoten, *d* = Dimension

Aus den Betrachtungen des zweidimensionalen Bereichsbaums wissen wir bereits, dass $T(n,2) = O(n \cdot \log n)$ und $T(n,d) = 2T\left(\frac{n}{2}, d\right) + T(n, d-1)$ ist. Damit ist nun eine Verallgemeinerung wie folgt möglich:

$$T(n,2) = O(n \cdot \log n)$$

$$T(n,d) = \underbrace{2T\left(\frac{n}{2}, d\right)}_{4T\left(\frac{n}{4},d\right)+2T\left(\frac{n}{2},d-1\right)} + T(n, d-1)$$

$\ldots$

$$= 2^j T\left(\frac{n}{2^j}, d\right) + \sum_{i=0}^{j-1} 2^i T\left(\frac{n}{2^i}, d-1\right),$$

$$mit \quad j = \log n \quad ergibt \quad sich$$

$\ldots$

$$= \underbrace{n \cdot T(1,d)}_{O(n)} + \sum_{i=0}^{\log n - 1} 2^i T\left(\frac{n}{2^i}, d-1\right)$$

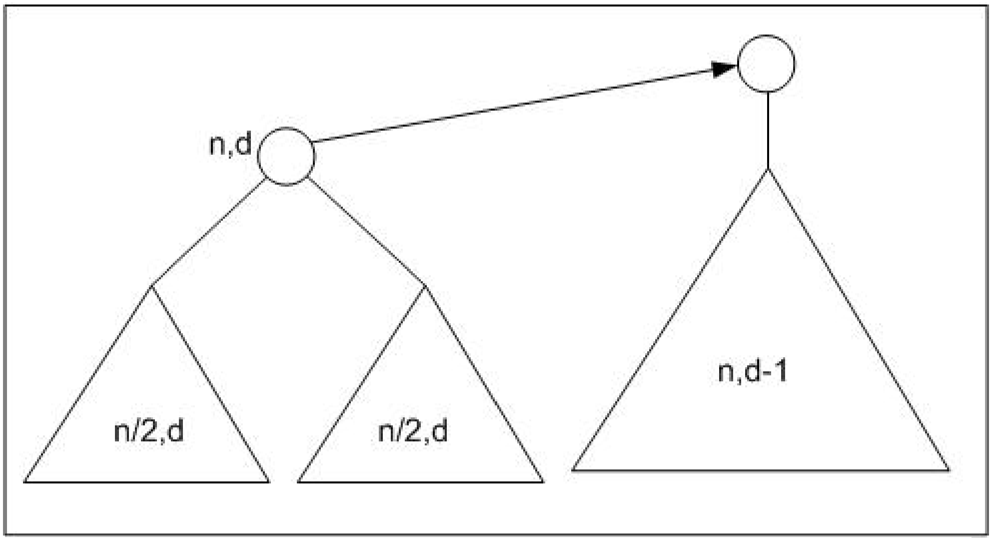

Abbildung 1: Verdeutlichung der Rekursion und Zeiger auf Sekundärstruktur

Aus den obigen Überlegungen ergibt sich folgende Behauptung: $T(n,d) \leq c \cdot n \cdot \log^{d-1} n$.
Der Beweis dieser Behauptung erfolgt mittels Induktion:

Induktionsanfang:

Für *d* = 2 stimmt die Behauptung, wie wir bereits gesehen haben.

Induktionsschritt: $(d-1) \to d$

$$T(n,d) = \sum_{i=0}^{\log n} 2^i T\left(\frac{n}{2^i}, d-1\right) + O(n)$$

$$\leq \sum_{i=0}^{\log n} 2^i \cdot c \cdot \frac{n}{2^i} \cdot \log^{d-2}\left(\frac{n}{2^i}\right) + O(n)$$

$$\leq c \cdot \sum_{i=0}^{\log n-2} n \cdot \log^{d-2} n + c \cdot n \cdot O(n)$$

$$\leq cn \cdot \log^{d-2} n \cdot (\log n - 1) + cn + \underbrace{en}_{aus\ O(n)}$$

$$= cn \cdot \log^{d-1} n + c(n - n \cdot \log^{d-2} n) + en$$

Dabei muss $(c(n - n \cdot \log^{d-2} n) + en) \leq 0$ für ein hinreichend großes c sein.

Zusammenfassung zu *d*-dimensionalen Bereichsbäumen

Sei $S \subset R^d$, $|S| = n$

Für die Vorverarbeitungszeit sowie den Platzbedarf wird jeweils $O(n \log^{d-1} n)$ Zeit benötigt.
Eine orthogonale Bereichsabfrage in einem *d*-dimensionalen Bereichsbaum ist in Zeit $O(\log^d n + k)$ möglich, wobei k die Größe der Antwort ist, d.h. die Anzahl der Punkte, die ausgegeben werden.

Dynamisierung von *d*-dimensionalen Bereichsbäumen (Einfügen und Streichen in *S*)

Die Dynamisierung von *d*-dimensionalen Bereichsbäumen durch Einfügen und Streichen in *S* ist möglich, wenn entsprechende balancierte Suchbäume benutzt werden (also z.B. AVL-Bäume, gewichtsbalancierte Bäume, rot-schwarz-Bäume). Dies ist in Zeit $O(\log^d n)$ möglich.

Segmentbäume

Sei $U \subset R$, $U = \{x_1, ..., x_N\}$ und $x_1 < x_2 < ... < x_N$.
Sei S eine Menge von n Intervallen („*Segmenten*“) mit Endpunkten $\in U$.

Ein *Segmentbaum* für S ist ein binärer Suchbaum mit $2N+1$ Blättern und Zusatzinformationen. Die Blätter des Baumes entsprechen „elementaren Intervallen“.

Die Intervalle werden wie folgt dargestellt:

$(-\infty, x_1), [x_1, x_1], (x_1, x_2), [x_2, x_2], ..., [x_N, x_N], (x_N, \infty)$, wobei $(-\infty, x_1)$ das Blatt ganz links und (x_N, ∞) das Blatt ganz rechts im Intervallbaum repräsentiert.

Die Markierung der inneren Knoten geschieht so, dass die Suche nach $x \in R$ in dem Intervall endet, das x enthält.

Beispiel:

$S = \{[1,5], [2,3], [1,2], [3,4], [3,5]\}$

$U = \{1,2,3,4,5\}$

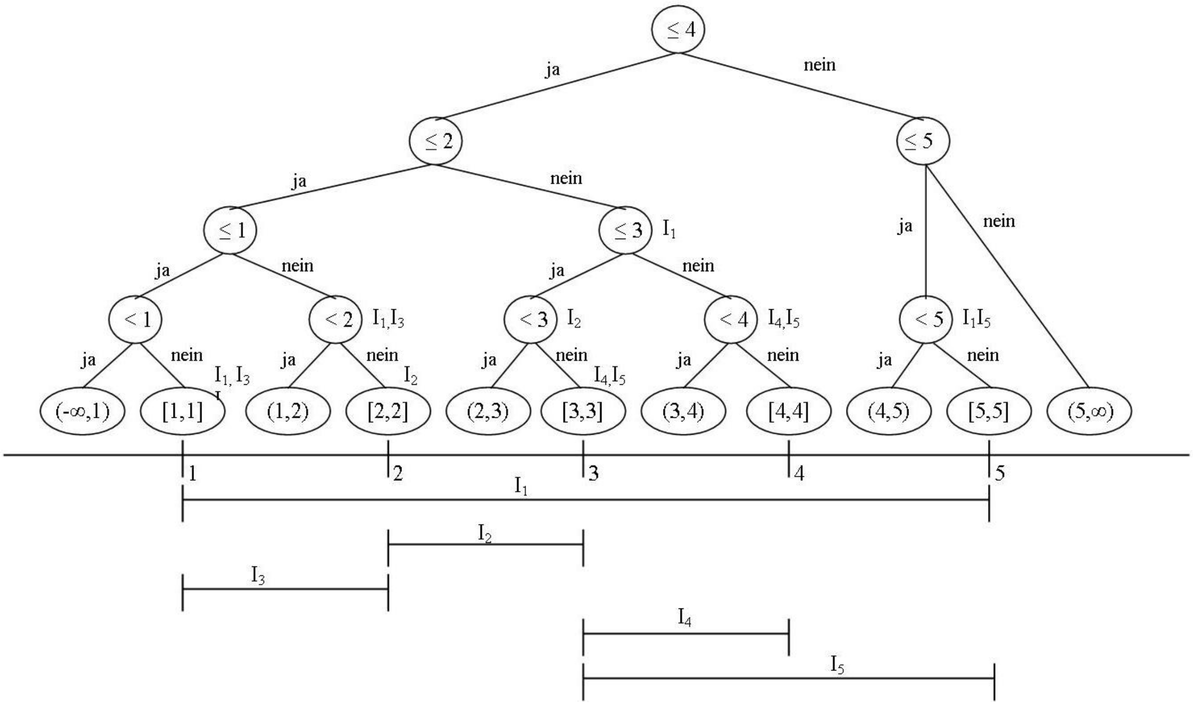

Abbildung 2: Beispiel für einen Segmentbaum mit Knotenlisten

Def.: Der x-Bereich eines Knotens v ist definiert als die Vereinigung der Intervalle in den Blättern des Teilbaums mit der Wurzel v.

Mit den Knoten v werden folgende zusätzliche Informationen abgespeichert:

In die Knotenliste $KL(v)$ (Zeiger nach) werden alle Intervalle von S aufgenommen, die den x-Bereich von v überdecken, aber nicht den seiner Vaters.

Lemma: Wenn T ein Segmentbaum der Höhe h ist für S und $I \in S$, dann gilt:

a.) I gehört zu $\leq 2 \cdot h$ Knotenlisten.

b.) I ist die disjunkte Vereinigung $\bigcup_{\substack{v \quad mit \\ I \in KL(v)}} x - Bereich(v)$

Beweis:

a.) I kann auf jeder Ebene von T zu höchstens zwei Knoten gehören.