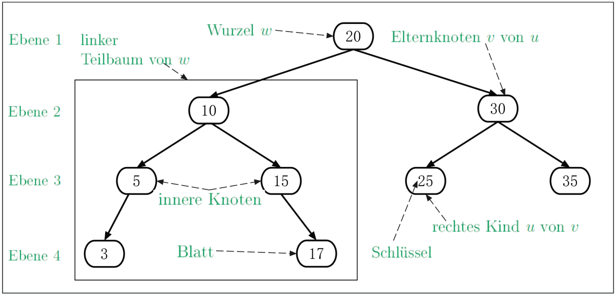
Tango Bäume

Andreas Windorfer

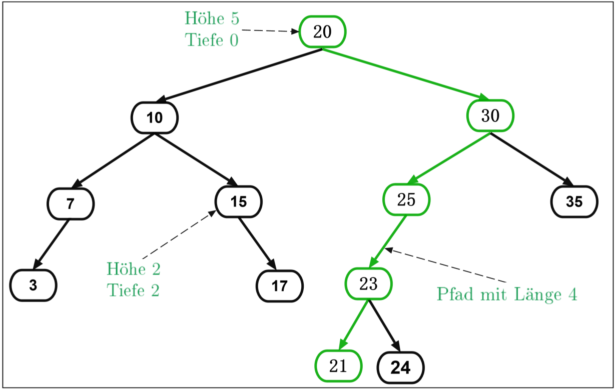
22. Oktober 2020

Übersicht

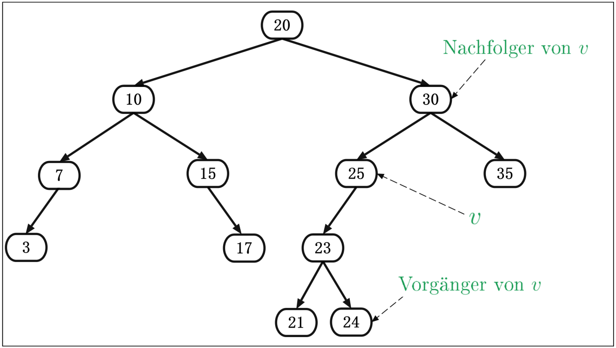
Binärer Suchbaum



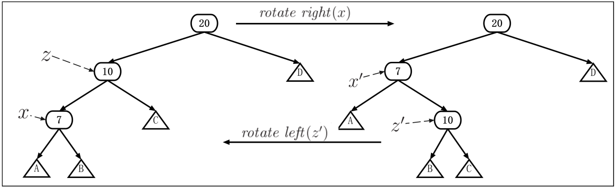
Binärer Suchbaum



Binärer Suchbaum



Binärer Suchbaum



Übersicht

Binärer Suchbaum

Dynamische Optimalität

Tango Baum

access(k) Operation

Parameter / Rückgabe

- Parameter k: Schlüssel im BST (Schlüsselmenge K)
- Rückgabe: Knoten mit Schlüssel k

$access(k)$ Operation

Einschränkungen

Ein Zeiger p in die Struktur:

- Setze p auf ein Kind von p
- Setze p auf den Elternknoten von p
- Rotationen

access(k) Operation

Einschränkungen

Ein Zeiger p in die Struktur:

- Setze p auf ein Kind von p
- Setze p auf den Elternknoten von p
- Rotationen

Berechnung der Kosten

Einheitskosten von „1“.

Zugriffsfolgen

Zugriffsfolgen

- $X = x_1, x_2, ..., x_m$, mit $\forall i \in \{1, 2, .., m\} : x_i \in K$
- $access(x_1)$, $access(x_2)$, ..., $access(x_m)$

Zugriffsfolgen

Zugriffsfolgen

- $X = x_1, x_2, ..., x_m$, mit $\forall i \in \{1, 2, .., m\} : x_i \in K$
- $access(x_1)$, $access(x_2)$, ..., $access(x_m)$

dynamische BST

- Anpassung der Struktur

Kostenrechnung

Anzahl der Einzelschritte $+\ m$

dynamisch Optimal

$OPT(X)$

Niedrigste Kosten zum Ausführen von X

dynamisch Optimal

$OPT(X)$

Niedrigste Kosten zum Ausführen von X

dynamisch Optimal

BST mit Kosten von $O(OPT(X))$, für beliebige X

c-competitive

BST mit Kosten von $O(c \cdot OPT(X))$, für beliebige X

Tango Baum

Eigenschaten

- Aus BSTs bestehender BST
- $\log(\log(n))$-competitive

Literatur

Erik D. Demaine, Dion. Harmon, John. Iacono, and Mihai. Patrascu. Dynamic optimality-almost. SIAM Journal on Computing, 37(1):240 251, 2007.

Interleave Lower Bound

Motivation

- Berechnung einer unteren Schranke zu $OPT(X)$
- Beweis der $\log(\log(n))$-competitiveness

Lower Bound Tree

Definition

Zu $X = x_1, x_2, ., x_m$ und $K = \{k \in \mathbb{N} | k \ ist \ in \ X \ enthalten\}$

Lower Bound Tree

Definition

Zu $X = x_1, x_2, ., x_m$ und $K = \{k \in \mathbb{N} | k \textit{ ist in } X \textit{ enthalten}\}$ ist der komplette BST P mit der Schlüsselmenge K der LBT.

Beispiel LBT

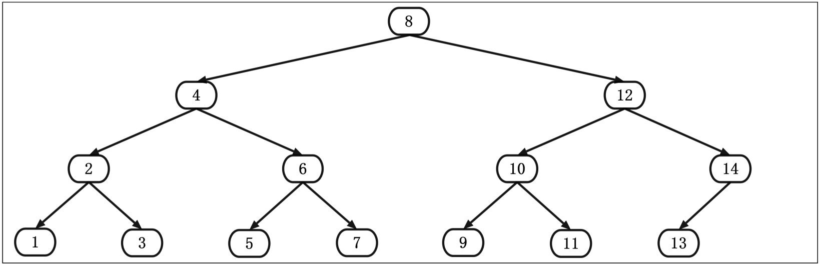


Abbildung: Der Lower Bound Tree zur Zugriffsfolge $1, 2, .., 14$.

Lower Bound Tree

Linke Region eines Kontens v

Schlüssel des linken Teilbaumes von v und $key(v)$

Linke Region eines Kontens v

Schlüssel des rechten Teilbaumes von v

Interleave durch v

x_{i-1} liegt in der linken Region und x_i in der rechten, oder umgekehrt.

Lower Bound Tree

inScore (X, v)

Anzahl der Interleaves durch v

IB (X)

$IB\,(X) = \sum_{u \in U} inScore\,(X, u)$

Transition Points

T_0 Startzustand, T_i nach ausführen von *access* x_i.

Transition Points

T_0 Startzustand, T_i nach ausführen von *access* x_i. $j \in 0, 1, ..m$

Zu jedem Knoten u aus P, mit nicht leerer rechter Region, existiert ein transition point in T_j

Transition Points

Sei v der Transition Point zu u und T_j

1. Im Pfad von der Wurzel zu v ist ein Knoten mit einem Schlüssel aus der linken Region von u enthalten.
2. Im Pfad von der Wurzel zu v ist ein Knoten mit einem Schlüssel aus der rechten Region von u enthalten.
3. Kein anderer Knoten mit kleinere Tiefe erfüllt die Eigenschaften eins und zwei.