Bachelorarbeit

Andreas Windorfer

21. Juni 2020

Inhaltsverzeichnis

1 Tango Baum **3**
1.1 Interleave Lower Bound 3
1.2 Aufbau des Tango Baum 6
1.3 Die *access* Operation beim Tango Baum 6
1.4 Laufzeitanalyse 6

1 Tango Baum

Der Tango Baum ist ein aus BSTs, den **auxiliary trees**, bestehender BST. Auf die Anforderungen an die auxiliary trees wird in Abschnitt 1.2 eingegangen und mit dem Rot-Schwarz-Baum wird eine mögliche Variante noch detailliert vorgestellt. Der Tango Baum wurde in [1],von Demaine, Harmon, Iacono und Patrascu beschrieben, inklusive eines Beweises über seine *lg lg n*-competitiveness. Ebenfalls in [1] enthalten ist eine als **Interleave Lower Bound** bezeichnete Variation der ersten unteren Schranke von Wilber. Da diese für das Verständnis des Tango Baumes wesentlich ist, wird mit ihr gestartet, bevor es zur Beschreibung der Struktur selbst kommt.

1.1 Interleave Lower Bound

Sei X eine Zugriffsfolge und sei $K = \{k \in \mathbb{N} | k \textit{ ist in } X \textit{ enthalten}\}$. Auch hier wird ein lower bound tree verwendet, dieser ist jedoch etwas anders definiert als in Abschnitt **??**. Hier ist der lower bound tree Y zu einer Zugriffsfolge X, der vollständig balancierte BST mit Schlüsselmenge K, bei dem die unterste Ebene von links startend nach rechts besetzt wird. Auch in der untersten Ebene gibt es also keine Möglichkeit einen Knoten links von einem bereits vorhandenen Knoten einzufügen. Anders als in Abschnitt **??** gibt es hier somit zu jeder Zugriffssequenz nur genau einen lower bound tree. Abbildung 1 zeigt den lower bound tree zur Zugriffsfolge $1, 2, .., 15$. Zu jedem Knoten v in Y werden zwei Mengen definiert. Die **linke Region** von v enthält den Schlüssel von v, sowie die im linken Teilbaum von v enthaltenen Schlüssel. Die **rechte Region** von v enthält die im rechten Teilbaum von v enthaltenen Schlüssel. Sei l der kleinste Schlüssel im Teilbaum mit Wurzel v und r der größte. Sei $X = \{x_1, x_2, ., x_m\}$ die Zugriffssequenz und $X_l^r = x_{1'}, x_{'2}, .., x_{jm'}$ wie in Abschnitt **??** definiert. $i \in \{2, 3, .., m'\}$ ist ein „Interleave durch v“ wenn $x_{(i-1)}$ in der linken Region von v liegt und x_i in der rechten Region von v, oder umgekehrt. Sei *inScore(v)* die Funktion die zu dem Knoten v die Anzahl der Interleaves durch v zurückgibt. Sei U die Menge der Knoten von Y. Die Funktion $IB(X)$ ist definiert durch:

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

Sei T_0 der BST mit Schlüsselmenge K auf X ausgeführt wird. Für $i \in \{1, 2, .., m\}$ sei T_i der BST, der entsteht nachdem $access(x_i)$ auf T_{i-1} ausgeführt wurde. Zu $u \in U$ und $j \in \{0, 1, .., m\}$ gibt es einen **transition point** v in T_j. v ist ein Knoten mit folgenden Eigenschaften:

1. Der Pfad von der Wurzel von T_j zu v enthält einen Knoten dessen Schlüssel in der linken Region von u enthalten ist.

2. Der Pfad von der Wurzel von T_j zu v enthält einen Knoten dessen Schlüssel in der rechten Region von u enthalten ist.

3. In T_i ist kein Knoten mit Eigenschaft 1 und 2 enthalten, der eine kleinere Tiefe als v hat.

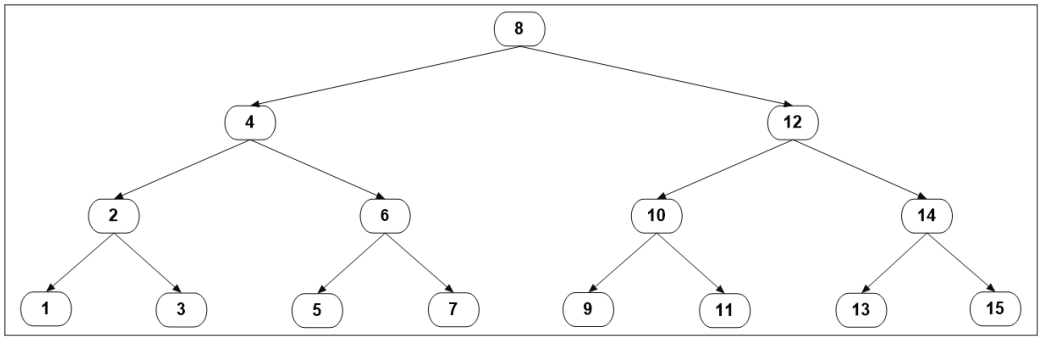

Abbildung 1: Der lower bound tree zur Zugriffsfolge $1, 2, .., 15$

Im Beweis dieses Abschnittes wird gezeigt das $OPT\,(X) \geq IB\,(X)$ gilt. Dafür werden jedoch noch drei Lemmas zu den Eigenschaften von Y benötigt.

Lemma 1.1. *Sei $X = x_1, x_2, .., x_m$ eine Zugriffssequenz und Y ein zu X erstellter lower bound tree mit Schlüselmenge K. Sei T_0 der BST mit Schlüsselmenge K auf dem X ausgeführt wird. Für $i \in \{1, 2, .., m\}$ sei T_i der BST der durch Ausführen von access(x_i) auf $T_{(i-1)}$ entsteht. Sei U die Menge der Knoten von Y. Dann gibt es zu jedem Knoten $u \in U$ und $j \in \{0, 1, .., m\}$ genau einen transition point in T_j.*

Beweis. Sei l der kleinste Schlüssel in der linken Region von u und r der Größte. Im Teilbaum mit Wurzel u sind genau die Schlüssel $K_l^r = \{k \in K | k \in [l, r]\}$ enthalten. Sei v_l der gemeinsame Vorfahre aller Knoten mit einem Schlüssel aus der linken Region von u in T_j, mit der kleinsten Tiefe. Sei r der gemeinsame Vorfahre aller Knoten mit einem Schlüssel aus der rechten Region von u in T_j, mit der kleinsten Tiefe. $key\,(l)$ bzw. $key\,(r)$ muss selbst in der linken bzw. rechten Region von u enthalten sein, vergleiche **??**. Sei w der gemeinsame Vorfahre aller Schlüssel aus der linken und der rechten Region von u in T_l^r. Es muss $key\,(w) \in [l, r]$ gelten. Somit muss $key\,(w)$ entweder in der linken oder rechten Region von u enthalten sein. Da w der Knoten mit der kleinsten Tiefe sein muss, für den $key\,(w) \in [l, r]$ gilt, muss

entweder $w = v_l$ oder $w = v_r$ gelten, je nachdem wessen Tiefe kleiner ist. Für den Fall $w = v_l$ ist v_r der transition point in T_l^r zu u und für den Fall $w = v_r$ ist es v_l. Es wird der Fall $w = v_l$ betrachtet, der andere kann direkt daraus abgeleitet werden. Im Pfad $P_u = v_0, v_1, .., v_r$ von der Wurzel zu v_r ist v_l enthalten und da v_r ein gemeinsamer Vorfahre der Schlüssel aus der rechten Region von u ist muss v_r der einige Knoten mit einem Schlüssel aus der rechten Region von u in P_u sein. In jedem Pfad P in T_l^r von der Wurzel zu einem Knoten mit einem Schlüssel aus der rechten Region von u muss mit $v_0, v_1, .., v_r$ beginnen, somit kann es keinen weiteren transition point für u in T_j geben.

□

Der Knoten auf den der Zeiger p zum ausführen von *access* gerade zeigt wird als **berührter** Knoten bezeichnet. Im zweiten Lemma geht es darum das sich der transition point v eines Knoten nicht verändern kann, solange v nicht wenigstens einmal der berührte Knoten war. In den zwei verbleibenden Lemmas seien T_j, X, Y und u wie in Lemma 1.1 definiert.

Lemma 1.1. *Sei v der transition point zu u in T_j. Sei $k \in N$, mit $j \leq k \leq m$. Gilt für alle x_i, mit $i \in [j, k]$, während der Ausführung von access(x_i), v war nicht wenigstens einmal der berührte Knoten, dann ist v während der gesamten Ausführungszeit von $access(x_j), access(x_{j+1}), .., access(x_k)$ der transition point zu u in T_k.*

Beweis. Sei v_l der gemeinsame Vorfahre aller Knoten mit einem Schlüssel aus der linken Region von u in T_j, mit der kleinsten Tiefe. Sei r der gemeinsame Vorfahre aller Knoten mit einem Schlüssel aus der rechten Region von u in T_j, mit der kleinsten Tiefe. Hier wird wieder ohne Verlust der Allgemeinheit der Fall $v = v_r$ betrachtet. Da v_r nicht berührt wird, wird auch kein Knoten in der rechten Region von u berührt. v_r ist somit während der gesamten Ausführungszeit von $access(x_j), access(x_{j+1}), .., access(x_k)$ der gemeinsame Vorfahre der Schlüssel aus der rechten Region von u mit der kleinsten Tiefe. Knoten mit Schlüssel in der linken Region von u könnten berührt werden. Zu einem Ausführungszeitpunkt i kann deshalb ein Knoten $v_{li} \neq v_l$ mit einem Schlüssel aus der linken Region von u der gemeinsame Vorfahre der Knoten mit diesen Schlüsseln mit der kleinsten Tiefe sein. Da v_r nicht berührt wird kann zu keinem Zeitpunkt v_l im Teilbaum mit Wurzel v_r enthalten sein. Somit kann auch v_{li} nicht in diesem Teilbaum enthalten sein. Somit muss die Tiefe von v_{li} kleiner sein, als die von v_r und v_r bleibt der transition point von u.

□

1.2 Aufbau des Tango Baum

1.3 Die *access* Operation beim Tango Baum

1.4 Laufzeitanalyse

Literatur

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