AD Praktikum: Aufgabe 08, Binärer Suchbaum Teil 2

Sönke Peters
Karl-Fabian Witte

28. Mai 2017

Abstract

Der binäre Suchbaum von Ganzzahlen bietet folgende Funktionen an: Einfügen, Ausgabe in inOrder, preOrder und postOrder und Berechnung der Summe der Werte Baumknoten zwischen zwei Werten. Diese Berechnung ist möglichst schnell (Komplexität relativ klein) implementiert. Eine Messung der Komplexität wird zudem erhoben.

SumBetween

Die Funktion des Suchbaumes, der Summe zuwischen zwei Werten aus den Werten der Knoten a_i dazwischen in numerischer Reihenfolge (inOrder) berechnet, soll im Folgendenden $sumBetween(m, M)$ genannt werden, wobei gilt:

$$sumBetween(m, M) = \sum_i a_i \qquad m \leq a_i.wert \leq M$$

Der Algorithmus sollte nur die Knoten `a_m` und `a_M` suchen und aus diesen die Summe berechnen, wobei gilt (a_{i+1} ist der Folgeknoten in numerischer Reihenflge von a_i):

$$a_m.wert \geq m > a_{m-1}.wert \qquad a_M.wert \leq m < a_{M+1}.wert$$

Es wird dafür in den Knoten eine weitere Information, nämlich die Summe aller vorherigen Werte (inOrder), hinzugefügt. Diese hier genannte *excludSum* wir bei jedem einfügen eines Neuen Knotens neu berechnet, in dem in inOrder alle Werte der vorherigen Knoten aufsummmiert in den aktuellen Knoten gespeichert werden.

In der $sumBetween(n, M)$ wird dann nur noch nach den enstprechenden Knoten gesucht. Der Algorithmus ist in pseudocode wie folgt:

```
int sumBetween(int m, int M){
        Knoten a_m = find_min(m);
        Knoten a_M = find_max(M);
        return a_M.wert + a_M.excludeSum - a_m.excludeSum;
}
```

Um die Bedingung $a_m.wert \geq m > a_{m-1}.wert$ bzw. $a_M.wert \leq m < a_{M+1}.wert$ zu gewärleisten, muss ein extra Knoten gespeichert werden, der als letztes größer war oder kleiner. Bei der Suche nach den entsprechenden Knoten wird hier als pseudo code von $find_min(m)$ dargestellt:

```
Knoten find_min(int m){
        Knoten a_m = 0;
        Knoten tmp = root;
        Knoten last_gt = root;
        while (a_m == 0){
          if(m == tmp.wert){
            a_m = tmp;
```

```
            } else {
              if ( m < tmp.wert ){
                last_gt = tmp;
                if (tmp.linkesKind  == 0) am = last_gt;
                else                      tmp = tmp.linkesKind;
              }
              else if (m > tmp.wert ){
                if (tmp.rechtesKind  == 0) am = last_gt;
                else                       tmp = tmp.rechtesKind;
              }
            }
          return a_m;
}
```

Der $find_max(M)$ funktioniert ähnlich, nur das der $last_gt$ (last greater then) Pointer $last_lt$ (last less then) heißt und in der rechten hälfte gesetzt wird ($if(m > tmp.wert)$).

Messung

Die Komplexotät der Suche hängt um einen von der Baumtiefe und zum anderen von der der Wahl von m und M.

Gemessen wurden die Bewegungen und die Vergleiche in $sumBetween(n, M)$.

Für die Messungen `best tree, worst search` , `random` und `worst tree` sind m und M Werte außerhalb vom Wertebereich der Baumes. Bei `best tree, best search` wurde nur nach den Wurzelknotenwert gesucht.

Die Baumstruktur bei `best three` ist die ideale Baumstrucktur, mit der Tiefe $\log_2 N$. Die Baumstruktur von `worst tree` gleicht der einer Liste. (Tiefe N). `ramdom` ist unberechenbar, weswegen der mittelwert aus 20 Messungen gewählt wurde.

In Abbildung **??** hat der `worst tree` eine lineare Komplexität $O(N)$, was auch auf die listenenartige Baumstruktur zurückzuführen ist. `best tree, best search` weißt eine Knostante Komplexität auf $O(1)$, nicht mehrmals die Schleife durchläuft, da die Wurzel das Ziel ist. `best tree, worst search` zeigt deutlich die Baumtiefe des Baumes wieder, da die Suche bis nach ganz unten bis zur letzten Ebene geht $O(logN)$. `random` weist auch eine Komplexität von $O(logN)$, sie liegt jedoch unter der von `best tree, worst search`, da die Tiefe der einzelnen Zweige variiert und die Randzweige bei unserer Messung leider zu kurz kamen...

Es wurde die API des Quellcodes mit angehängt.

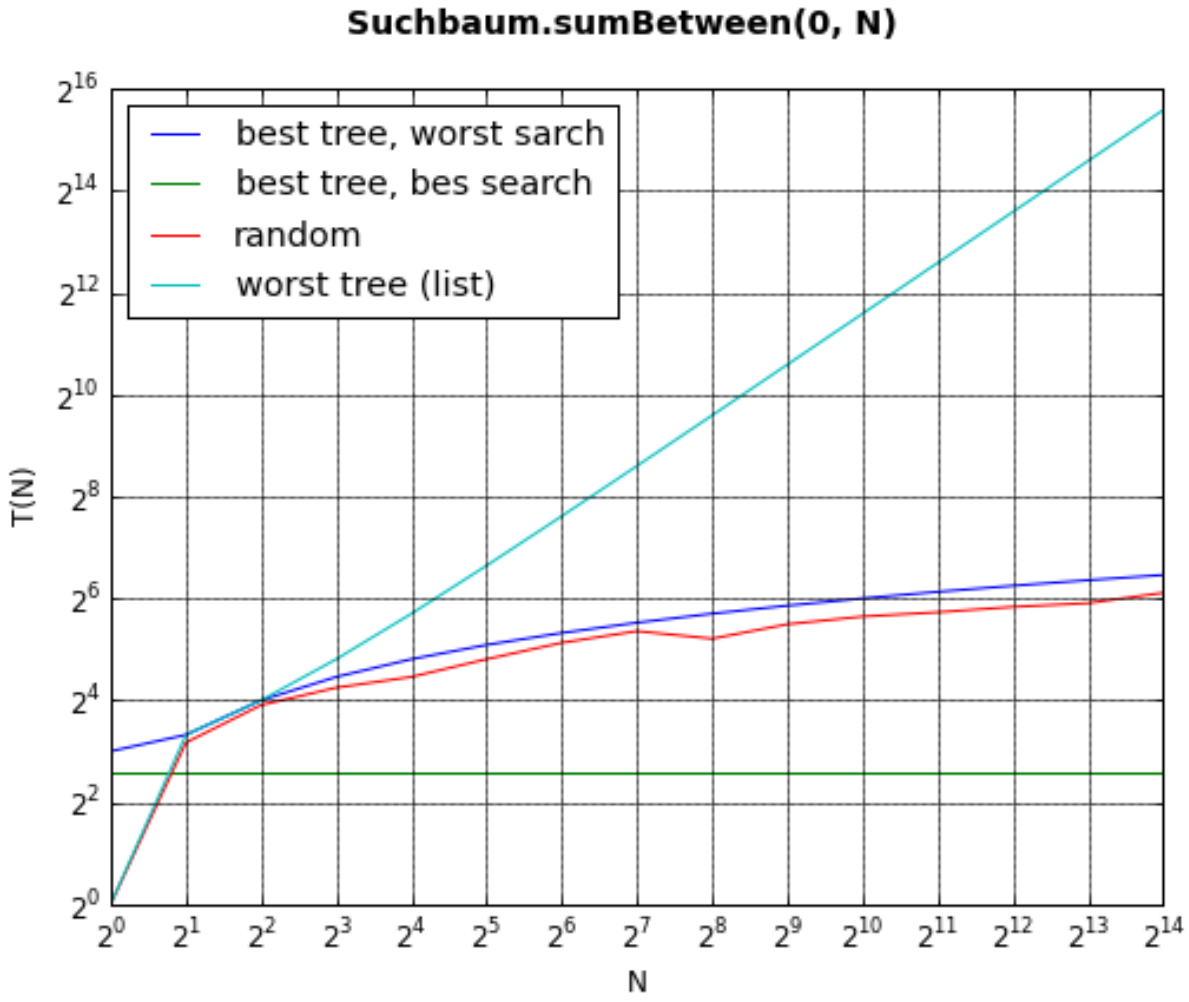

Abbildung 1: Aufwende der Suche in Abhängigkeit von der Baumstruktur und der geuschten Min und Max Werte, sowie der Anzahl der Knoten N