A Simplified BST Complexity Analysis

In this document we would like to give a simple derivation of complexity analysis for the cost of BST search under the average case. Please be aware that the purpose of this type of analysis is to estimate the magnitude of the complexity rather than an accurate calculation. So we will drop those lower order terms in our analysis.

First, we define the internal path length to be the sum of the depth of all nodes in a tree. For any search operation in a BST, the target key is compared with the keys in those nodes. So the average internal path length gives a good estimation of the cost for BST search. If we analyze the details of insertion and deletion, we will see they are all of the same magnitude.

Let $T(N)$ be the average internal path length for a BST of N nodes. Then $T(0) = T(1) = 0$ (the depth of root is 0). We denote the keys of the N input data as $x_1 \leq x_2 \leq \cdots \leq x_i \leq x_{i+1} \leq \cdots \leq x_N$. If x_i is present in the root node, then the left subtree has $i - 1$ nodes and the right subtree has $N - i$ nodes. For each node in the left subtree ($i - 1$ nodes in total), the depth in the original tree is the depth in the subtree plus one. We can formulate it as:

$$\text{Depth}(root) = \text{Depth}(root.left) + 1. \tag{1}$$

Similarly for each node in the right subtree ($N - i$ nodes in total),

$$\text{Depth}(root) = \text{Depth}(root.right) + 1. \tag{2}$$

So if we sum all the nodes up and then take the average for all possible i's, we have

$$T(N) = \frac{1}{N} \sum_{i=1}^{N} [T(i-1) + T(N-i)] + N - 1. \tag{3}$$

Note that

$$\sum_{i=1}^{N} T(i-1) = \sum_{i=1}^{N} T(N-i) = \sum_{i=1}^{N-1} T(i).$$

We then have

$$T(N) = \frac{2}{N} \sum_{i=1}^{N-1} T(i) + N - 1. \tag{4}$$

Now we will try to ESTIMATE $T(N)$. The key is to treat it as an iterative formula by eliminating the sum. We first rewrite (4) as

$$N\,T(N) = 2 \sum_{i=1}^{N-1} T(i) + N(N-1). \tag{5}$$

We then replace N in (5) by $N + 1$ and have

$$(N + 1)T(N + 1) = 2 \sum_{I=1}^{N} T(i) + (N + 1)N. \tag{6}$$

Now subtract (6) by (5), we have

$$(N + 1)T(N + 1) - N\,T(N) = 2T(N) + 2N. \tag{7}$$

and thus

$$(N + 1)T(N + 1) = (N + 2)T(N) + 2N. \tag{8}$$

Divide both sides of the equation by $(N + 1)(N + 2)$, we have

$$\frac{T(N + 1)}{N + 2} = \frac{T(N)}{N + 1} + \frac{2N}{(N + 1)(N + 2)}\ . \tag{9}$$

Now if we define $S(N) = \frac{T(N)}{N+1}$, we have

$$S(N + 1) = S(N) + \frac{2N}{(N + 1)(N + 2)}\ . \tag{10}$$

As an estimation we simplify the right hand side by drop those constants but keeping only N terms, we have

$$S(N + 1) \approx S(N) + \frac{2}{N}\ . \tag{11}$$

Now with $S(1) = 0$, we have

$$S(N) \approx \sum_{K=1}^{N} \frac{2}{k} \approx 2 \ln N = 2 \log e \log N, \tag{12}$$

and thus

$$T(N) \approx 2 \log e \log N(N + 1) \approx O(N \log N). \tag{13}$$

Now since $T(N)$ is the average (under all possible mutations of the data) of the sum of the depth of all nodes, the average depth should be divided by N. Thus the search cost is estimated by $O(\log N)$.