The Height Bound of a Red-Black Tree

date: 20260904
status: #finished
tags: #computer_science #data_structure #tree #binary_tree #binary_search_tree #red_black_tree #balanced_tree

---

0. Introduction

(i) The motivating question

For a binary search tree T, we often care about how the height h grows with the size n, since the height determines the time complexity of various operations. In the worst case, T is extremely imbalanced, implying $h = O(n)$.

For a perfect or complete binary tree, the height-size relation can be easily derived.

- For a perfect binary tree, $n = 2^{h+1} - 1$. Therefore, $h = \log_2(n+1) - 1$.
- For a complete binary tree, $n \geq 2^h$. Therefore, $h \leq \log_2(n)$.

A natural question is: Can we develop a similar theorem for a red-black tree, which is neither perfect nor complete but only guaranteed to be balanced?

(ii) The theorem bounding a red-black tree's height

The following theorem states the height-size relation of a red-black tree. Through this theorem, we can ensure that the time complexity of search, insertion, and deletion is $O(\log n)$ within a red-black tree, a balanced binary search tree.

Moreover, this theorem explicitly shows the superiority of using the red-black tree as a data storage container, beating the benchmark complexity of $O(n)$.

> **Theorem (Height of a Red-Black Tree)**
>
> *For a red-black tree with n internal nodes, the height h is less than or equal to* $2\log_2(n+1)$.

1. Preliminary

(i) Height

Definition (Height)

The height of a tree data structure T, written $\text{height}(\text{root})$, is the number of edges contained in the longest path from the root to a leaf.

Note that defining the height of a subtree by the number of edges or by the number of nodes is equivalent, as the root is excluded in the node count.

(ii) Black height

Definition (Black Height)

The black height of T, written $\text{bh}(\text{root})$, is the number of black nodes, excluding the root, on **any** path from the root to a leaf.

(iii) Color rules

The color of the root and of the leaf nodes of a red-black tree is always black, and no two red nodes are consecutive.

2. Lemma 1: Black Height Versus Height

Lemma (Black Height Versus Height)

$$\text{bh}(\text{root}) \geq \frac{h}{2}$$

Proof

Consider the longest path from the root to a leaf. The number of nodes on this path, excluding the root, is h. Since no two red nodes are consecutive and the leaf node is always black, the maximum number of red nodes on this path is $\left\lfloor \frac{h}{2} \right\rfloor$. Therefore, the black height of the tree is bounded as follows:

$$
\begin{aligned}
\mathrm{bh}(\text{root}) &= h - \text{number of red nodes on the path with length } h \\
&\geq h - \underbrace{\left\lfloor \frac{h}{2} \right\rfloor}_{\text{maximum number of red nodes}} \\
&\geq h - \frac{h}{2} \\
&= \frac{h}{2}
\end{aligned}
$$

The intuition of this lemma is that as there are no consecutive red nodes, given an arbitrary path from the root to a leaf, the number of black nodes will definitely be greater than the number of red nodes.

---

3. Lemma 2: Subtree Size Versus Black Height

Consider a subtree of T rooted at an arbitrary node x. Denote the total number of internal nodes in this subtree by n_x, and the black height of this subtree by $\mathrm{bh}(x)$. Then we have the following lemma.

Lemma (Subtree Size Versus Black Height)

$$n_x \geq 2^{\mathrm{bh}(x)} - 1.$$

Proof

We prove this by induction, taking the subtree rooted at a leaf as the base case.

When the black height is equal to 0, which is the subtree rooted at a leaf node, the subtree holds no internal node. Hence, $n_{\text{leaf}} = 0$ and the inequality holds in this case: $n_{\text{leaf}} = 0 \geq 2^{\mathrm{bh}(\text{leaf})} - 1 = 2^0 - 1 = 0$.

Suppose that for any node x and its children $y_i, i \in \{1, 2\}$, representing the left and right child respectively, the inequality holds for y_i: $n_{y_i} \geq 2^{\mathrm{bh}(y_i)} - 1$.

Notice that if y_i is red, $\mathrm{bh}(y_i) = \mathrm{bh}(x)$, and if y_i is black, $\mathrm{bh}(y_i) = \mathrm{bh}(x) - 1$. Hence $\mathrm{bh}(y_i) \geq \mathrm{bh}(x) - 1$.

Finally,

$$
\begin{aligned}
n_x &= n_{y_1} + n_{y_2} + 1 \\
&\geq 2^{\mathrm{bh}(y_1)} - 1 + 2^{\mathrm{bh}(y_2)} - 1 + 1 \\
&\geq 2 \cdot 2^{\mathrm{bh}(x)-1} - 1 \\
&= 2^{\mathrm{bh}(x)} - 1
\end{aligned}
$$

By induction, for every node x, the total number of internal nodes in the subtree rooted at x is greater than or equal to $2^{\mathrm{bh}(x)} - 1$.

Note that the +1 term in $n_{y_1} + n_{y_2} + 1$ accounts for the node x itself.

---

4. Proof of the Theorem

Proof

Combining the first lemma with the second lemma,

$$
\begin{aligned}
n = n_{\text{root}} &\geq 2^{\mathrm{bh}(\text{root})} - 1 \geq 2^{\frac{h}{2}} - 1. \\
\implies \quad h &\leq 2\log_2(n+1)
\end{aligned}
$$