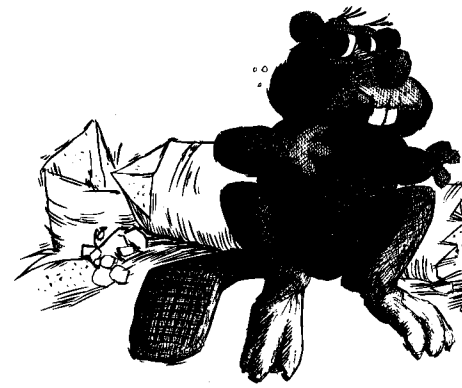
UNIVERSITY OF WATERLOO
COMPUTER SCIENCE DEPARTMENT

UNIVERSITY OF WATERLOO
COMPUTER SCIENCE DEPARTMENT



The Internal Path Length of Red-Black Trees

Helen Cameron
and
Derick Wood

Data Structuring Group
Research Report CS-89-50

October, 1989

The Internal Path Length of Red-Black Trees *

Helen Cameron† Derick Wood†

Abstract

In this paper, we show that the internal path length of a red-black tree of size N is bounded above by $2N(\log N - \log\log N) + O(N)$ and that this is, asymptotically, tight. To establish the asymptotic tightness of the bound, we introduce a class of red-black trees, the $C(k,h)$ trees, which achieve the bound when $k = h - 2\log h + \epsilon$, where $|\epsilon| \leq 1/2$.

1 Introduction

The internal path length (IPL) of a class of search trees is a measure of the running time of various algorithms that operate on trees in the class. Thus, it is a well-studied measure of performance for classes of search trees.

Knuth, in [Knu73], examined the multiway trees that have the smallest and largest internal path length among all multiway trees with the same number of internal nodes. The binary trees with minimal IPL for their sizes are the perfect binary trees, which have external nodes on at most two adjacent levels. These binary trees are members of the class of AVL trees and the class of red-black trees; thus, the minimal IPL trees for each size have already been characterized for AVL and red-black trees. In [MPRS79], the 2,3 trees with minimal and maximal IPL are characterized, and, in [KW89], an upper bound for the IPL of AVL trees is given and a family of AVL trees that achieve this bound is presented.

In this paper, we derive a tight upper bound for the IPL of red-black trees similar to the bound for AVL trees in [KW89]. In Section 2, we define the terminology we use and derive an upper bound on the IPL of red-black trees of size N. In Section 3, we introduce the $C(k,h)$ trees and prove that, by carefully choosing k as a function of h, the upper bound is achieved by these trees.

*This work was supported under a Natural Sciences and Engineering Research Council of Canada Grant No. A-5692 and under a grant from the Information Technology Research Centre.

†Data Structuring Group, Department of Computer Science, University of Waterloo, WATERLOO, Ontario N2L 3G1, CANADA

2 An Upper Bound on the Internal Path Length

If a node has children, then it is *internal*; otherwise, it is *external*. A *binary search tree* T is a search tree in which each internal node has two children. The *maximum height* of a tree T is denoted by $maxht(T)$ and is the length of a longest root-to-external-node path. The *minimum height* of a tree T, denoted by $minht(T)$, is the length of a shortest root-to-external-node path. Similarly, the maximum height of a node v in a tree T is the length of a longest path from node v to an external node that is a descendent of v, and the minimum height of node v is the length of a shortest such path. The *weight* of a tree T, denoted by $wt(T)$, is the number of external nodes and the *size* of T is the number of internal nodes. Note that the size of binary tree T is one less than the weight of T. The *level* of a node in a tree T is the distance of the node from the root of tree T; the root is at level 0, its children are at level 1, and so on.

The class of trees in which we are interested are the red-black trees; see [GS78]. The following definition is due to Olivié in [Oli82], who called them half-balanced binary search trees. The trees were introduced by Bayer in [Bay72], while Tarjan, in [Tar83], showed the two classes to be equivalent.

Definition 2.1 *A red-black tree is a binary search tree such that, for each node v in the tree, $maxht(v) \leq 2 \cdot minht(v)$; that is, a shortest path from v to an external node is at least one half as large as a longest such path.*

The *internal path length* of tree T is defined to be

$$IPL(T) = \sum_{v \text{ binary}} length(path(v)),$$

where $path(v)$ is the access path from the root of T to node v, and the length of the access path to a node on level i is $i + 1$. The *external path length* of tree T is defined to be

$$EPL(T) = \sum_{v \text{ external}} length(path(v)).$$

The external path length and the internal path length of a (binary) tree are related by the formula

$$EPL(T) = IPL(T) + 2 \cdot wt(T) - 1.$$

Lemma 2.1 *Let T be a binary tree of size N and maximum height h, and let l_i be the level on which the ith external node of T appears. Then,*

$$\mathrm{IPL}(T) = (h-1)(N+1) + 1 - \sum_{i=1}^{N+1} (h - l_i).$$

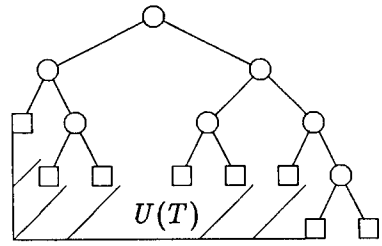


Figure 1: The "area" represented by $U(T)$.

Proof: From the relation between $IPL(T)$ and $EPL(T)$, we get

$$\begin{aligned} IPL(T) &= EPL(T) - 2(N+1) + 1 \\ &= \sum_{i=1}^{N+1} (l_i + 1) - 2(N+1) + 1 \\ &= \left[(h+1)(N+1) - \sum_{i=1}^{N+1} (h - l_i) \right] - 2(N+1) + 1 \\ &= (h-1)(N+1) + 1 - \sum_{i=1}^{N+1} (h - l_i). \end{aligned}$$

□

The approach used to obtain an upper bound for the internal path length of AVL trees in [KW89] was based on the following notion, which we also use.

Definition 2.2 *Let T be a binary tree of maximum height h and size N, and let the ith external node of tree T appear on level l_i. Define the* area under T, *denoted by $U(T)$, as*

$$U(T) = \sum_{i=1}^{N+1} (h - l_i).$$

The sum $U(T)$ is the empty "area" under the external nodes of tree T; see Figure 1. We can rewrite the equation for $IPL(T)$ in Lemma 2.1, using the definition of $U(T)$, to arrive at

$$IPL(T) = maxht(T) \cdot wt(T) + 1 - (U(T) + wt(T)).$$

Hence, if we derive a lower bound for $U(T) + wt(T)$, then we obtain an upper bound for $IPL(T)$. Let us now examine a family of red-black trees that, as we shall see, minimize $U(T) + wt(T)$ among the red-black trees of the same maximum height.

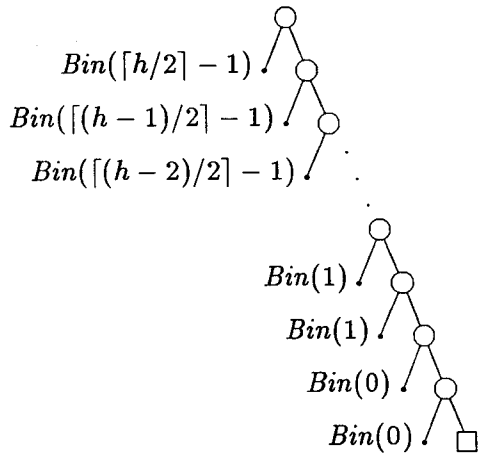


Figure 2: The definition of $Skinny(h)$.

We begin by defining the skinniest trees of a given height h, the $Skinny(h)$ trees; the definition is given in Figure 2. (Note: $Bin(h)$ is the complete binary tree of height h.) In fact, there is more than one skinny tree of maximum height h; $Skinny(h)$ is a family of trees of maximum height h each of which look like the tree displayed in Figure 2 with some sibling subtrees interchanged. Each $Skinny(h)$ tree has a "spine" (a longest root-to-external-node path) with pairs of complete binary trees hanging from the spine, starting with a pair of $Bin(0)$ trees at the bottom and increasing up to a $Bin(\lceil h/2 \rceil - 1)$ tree as a child of the root. For convenience, we use $Skinny(h)$ to denote both the family of trees of maximum height h and individual trees in the family.

We will show that the $Skinny(h)$ trees minimize the value of $U(T)+wt(T)$ among all red-black trees T with maximum height h. As a first step, we obtain a lower bound for $U(Skinny(h)) + wt(Skinny(h))$.

Lemma 2.2 *The weight of $Skinny(h)$ is given by*

$$wt(Skinny(h)) = \begin{cases} 2 \cdot 2^{h/2} - 1, & \text{if } h \text{ is even} \\ 3 \cdot 2^{(h-1)/2} - 1, & \text{if } h \text{ is odd} \end{cases}$$

and this is the minimum weight among all red-black trees of maximum height h.

Proof: See [Oli82]. □

Lemma 2.3 *The area under Skinny(h) is given by*

$$U(Skinny(h)) = \begin{cases} (h-3)2^{h/2}+3 & \text{if } h \text{ is even} \\ 3/2 \cdot (h-3)2^{(h-1)/2}+3 & \text{if } h \text{ is odd.} \end{cases}$$

Proof: Consider a pair of $Bin(i)$ subtrees hanging from the spine of $Skinny(h)$. Each external node of the $Bin(i)$ subtree nearest the root contributes $i+1$ to $U(Skinny(h))$, and each external node of the $Bin(i)$ subtree further from the root contributes i to $U(Skinny(h))$. If there is only one $Bin(i)$ subtree (that is, if $i = (h-1)/2$ and h is odd), then each external node of the $Bin(i)$ subtree contributes i to $U(Skinny(h))$.

Thus, if h is even, then, from Figure 2, we see that

$$\begin{aligned} U(Skinny(h)) &= \sum_{i=0}^{(h-2)/2} 2^i(2i+1) \\ &= 2\sum_{i=0}^{(h-2)/2} i2^i + \sum_{i=0}^{(h-2)/2} 2^i \\ &= (h-3)2^{h/2}+3. \end{aligned}$$

If h is odd, then

$$\begin{aligned} U(Skinny(h)) &= 2^{(h-1)/2} \cdot \frac{h-1}{2} + \sum_{i=0}^{(h-3)/2} 2^i(2i+1) \\ &= 3/2 \cdot (h-3)2^{(h-1)/2}+3. \end{aligned}$$

□

Corollary 2.4 *If $h \geq 4$, then*

$$U(Skinny(h)) + wt(Skinny(h)) \geq h2^{(h-1)/2}.$$

Proof: If h is even, then, by the above two lemmas,

$$\begin{aligned} U(Skinny(h)) + wt(Skinny(h)) &= (h-3)2^{h/2}+3+2\cdot 2^{h/2}-1 \\ &= (h-1)2^{h/2}+2 \\ &> h2^{(h-1)/2}, \quad \text{if } h \geq 4. \end{aligned}$$

If h is odd, then

$$\begin{aligned} &U(Skinny(h)) + wt(Skinny(h)) \\ &= 3/2 \cdot (h-3)2^{(h-1)/2}+3+3\cdot 2^{(h-1)/2}-1 \\ &= 3/2 \cdot (h-1)2^{(h-1)/2}+2 \\ &> h2^{(h-1)/2}. \end{aligned}$$

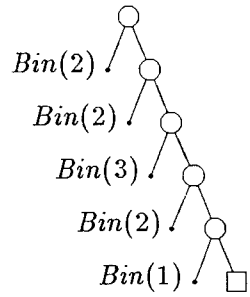


Figure 3: A red-black tree with the same weight but larger IPL than *Skinny*(7). (Note: $IPL(Skinny(7)) = 88$)

□

Even though *Skinny*(h) minimizes $U(T) + wt(T)$ among all red-black trees with maximum height h, *Skinny*(h) does not necessarily have the maximum internal path length among all red-black trees of the same weight; see Figure 3.

We now wish to show that *Skinny*(h) minimizes the value of $U(T)+wt(T)$ among all red-black trees T of maximum height h.

Definition 2.3 *The replacement of an external node in a binary tree by a new internal node with two external node children is called a* simple insertion. *That is, a simple insertion is the insertion of new node without any adjustments made to the structure of the tree to retain some desired property (for example, the red-black property).*

Similar to results for brother trees in [OW82], we show that any red-black tree of maximum height h can be produced from some tree in the family of *Skinny*(h) trees by a series of insertions that do not require any promotions to be performed to retain the correct ratio between the maximum height and minimum height of each node in the tree; that is, by a series of simple insertions. Then, we show that the insertions do not decrease the value of $U(T) + wt(T)$.

Suppose a red-black tree T' can be produced by a sequence of simple insertions from some *Skinny*(h) tree T. How would such a sequence appear? We will see that the area beneath each *Bin*(i) subtree hanging from the spine of T is filled in until each of these subtrees matches the corresponding subtree in T'. Part of this proof will be to show that the area beneath a

$Bin(i)$ tree can be filled in, one level at a time, until it matches any red-black tree T'' such that $maxht(T'') \geq i$ and $minht(T'') \geq i$, and each intermediate tree in the sequence is a red-black tree. The first step towards this proof is to show that the reverse of the sequence from $Bin(i)$ to T'' can be performed; that is, we can remove the nodes from T'' one at a time, starting at the level furthest from the root, removing nodes, level by level, until we are left with $Bin(i)$, and each intermediate tree in the sequence is a red-black tree.

Lemma 2.5 *Let T be an arbitrary red-black tree of maximum height h and minimum height l. If we remove an internal node v on level $h-1$ of tree T (that is, replace v and its two external node children by an external node), then the resulting tree T' is a red-black tree.*

Proof: The only nodes whose $minht/maxht$ ratios can be affected by the removal of node v from tree T are the ancestors of v on the path P from v to the root of tree T. Let w be a node on level j of tree T such that w is on path P. There are four cases to consider: the removal of v changes both $maxht(w)$ and $minht(w)$, the removal changes $maxht(w)$ but not $minht(w)$, the removal changes $minht(w)$ but not $maxht(w)$, and neither $maxht(w)$ nor $minht(w)$ are changed by the removal.

- Neither $maxht(w)$ nor $minht(w)$ are changed by the removal of node v.

  Clearly, $maxht(w) \leq 2 \cdot minht(w)$ in the resulting tree since the inequality must hold in the red-black tree T.

- Only $maxht(w)$ is changed; $minht(w)$ remains unchanged.

  Since T is a red-black tree, we know that $maxht(w) \leq 2 \cdot minht(w)$ held before the removal of node v. The removal of v can only change $maxht(w)$ by decreasing it by one. Thus, $maxht(w) - 1 \leq 2 \cdot minht(w)$, so the red-black property still holds at node w in the resulting tree T'.

- Only $minht(w)$ changes; $maxht(w)$ remains the same.

  Since v was on level $h-1$, the last level of internal nodes of tree T, and the value of $minht(w)$ changed with the removal of v, all other external nodes descendents of w must be on level h. In other words, before the removal, node w was the root of a $Bin(h-j)$ subtree in T, and $maxht(w) = minht(w) = h - j$.

  Thus, after the removal, $minht(w) = h-j-1$ and $maxht(w) = h-j$. But, $h - j \leq 2(h - j - 1) = 2(h - j) - 2$, if $j < h - 1$. Since node v was on level $h - 1$, node w was an ancestor of node v, and node w is on level j, we have $j < h - 1$.

- Both $maxht(w)$ and $minht(w)$ are changed.

  This implies that the external node children of v are the only external node descendants of w; otherwise, both values could not be changed by this single removal. Thus, w must be v, a contradiction.

Thus, in the resulting tree T', the red-black property holds for all nodes that could have been affected by the removal of node v. Thus, tree T' is also a red-black tree. □

Now we can show how to fill in a $Bin(j)$ tree by a sequence of simple insertions until we have a desired red-black tree.

Lemma 2.6 *Let T be a red-black tree with $maxht(T) = h$ and $minht(T) = k \geq j$. There exists a sequence of simple insertions that can be applied to $Bin(j)$ to produce tree T such that each intermediate tree produced by the sequence is a red-black tree.*

Proof: We examine the reverse sequence. Using Lemma 2.5 repeatedly, we can remove all the nodes on level h of tree T one by one, then the nodes on level $h-1, \ldots$, then the nodes on level $j+1$, and, after each removal, the resulting tree is red-black. The end result of the removals is $Bin(j)$, since $j \leq minht(T)$.

If, somewhere in this sequence, we remove one node on the final level i of some intermediate tree T' to get the next intermediate tree T'', then, by Lemma 2.5, since the original tree T is a red-black tree, T' and T'' are red-black trees. The only difference between T' and T'' is the node that was removed. Thus, if we insert the node into tree T'' using the normal insertion routine for search trees, the result will be T' and no promotions are required to maintain the properties of a red-black tree.

Hence, if we start with $Bin(j)$, reverse the order of the sequence of removals, and use it as a sequence of insertions, we will have a sequence of simple insertions that produces the red-black tree T from $Bin(j)$. □

Finally, we can show that there exists a sequence of simple insertions that leads from some member of the family of $Skinny(h)$ to any red-black tree of maximum height h.

Theorem 2.7 *Let T' be a red-black tree of maximum height h. Then, there exists a $Skinny(h)$ tree T and a sequence I of simple insertions such that when sequence I is applied to tree T the result is tree T' and each intermediate tree is a red-black tree.*

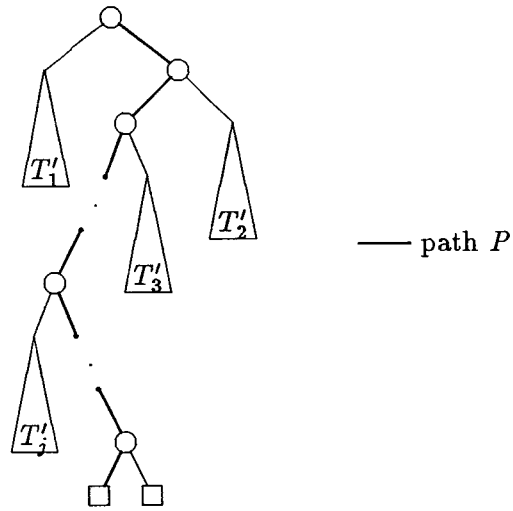


Figure 4: The path P in tree T' and the subtrees hanging from it.

Proof: Since T' has maximum height h, there exists some root-to-external-node path P of length h. Choose T to be the member of $Skinny(h)$ whose spine matches path P.

Consider the subtree T'_j in T' that is rooted on level j and does not contain any nodes of path P, but is a child of the node v_j in path P on level $j-1$; see Figure 4. The other child of node v_j, which contains a portion of path P, has maximum height $h-j$. The subtree T'_j has maximum height at most $h-j$, since the maximum height of tree T' is h. Since T' is a red-black tree, we must have $maxht(v_j) \leq 2 \cdot minht(v_j)$. But $maxht(v_j) = h-j+1$ and $minht(v_j) \leq minht(T'_j)+1$. Thus, we have $h-j+1 \leq 2(minht(T'_j)+1)$ or $minht(T'_j) \geq (h-j+1)/2-1$. Since $minht(T'_j)$ is an integer, we have $minht(T'_j) \geq \lceil (h-j+1)/2 \rceil - 1$.

Let T_j be the subtree in $Skinny(h)$ tree T that corresponds to T'_j; that is, let T_j be the subtree rooted on level j that is the child of the spine node on level $j-1$ and does not contain any of the spine. Subtree T_j is $Bin(\lceil (h-j+1)/2 \rceil - 1)$.

Now, by Lemma 2.6, there exists a sequence I_j of simple insertions that, when applied to $Bin(\lceil (h-j+1)/2 \rceil - 1)$, results in T'_j and each intermediate tree is a red-black tree.

We now have $h-1$ sequences of simple insertions, $I_1, I_2, \ldots, I_{h-1}$, where sequence I_j, when applied to $Bin(\lceil (h-j+1)/2 \rceil - 1)$, results in subtree T'_j of tree T'. We now show that these sequences, when applied one after another to the proper subtrees of $Skinny(h)$ tree T, result in tree T' and each intermediate tree is a red-black tree. Since I_j transforms $Bin(\lceil (h-j+1)/2 \rceil -$

1) into T'_j without any promotions required within $Bin(\lceil (h-j+1)/2 \rceil - 1)$, the only possible problem is if one of the insertions in I_j causes some node v outside the $Bin(\lceil (h-j+1)/2 \rceil - 1)$ subtree to no longer satisfy $maxht(v) \leq 2 \cdot minht(v)$. The node v must be on the spine of $Skinny(h)$ tree T, since the only nodes that can be affected by an insertion are ancestors of the newly added node. But the maximum height of node v can never be changed by any of these insertions, since both $Skinny(h)$ tree T and the red-black tree T' have maximum height h. Thus, an insertion may increase the value of the minimum height of a spine node, but not its maximum height. But any spine node v satisfies $maxht(v) \leq 2 \cdot minht(v)$ before any insertions are performed because T is a red-black tree, and thus node v must also satisfy the relation after some insertions are performed because the only changes that may occur are that $minht(v)$ may be increased. □

Now we prove that a simple insertion does not decrease the value of $U(T) + wt(T)$.

Lemma 2.8 *Let T and T' be binary trees such that T can be transformed into T' by one simple insertion. Then,*

$$U(T) + wt(T) \leq U(T') + wt(T').$$

Proof: Observe that $wt(T') = wt(T) + 1$. There are two cases to consider: either the insertion increased the maximum height of the tree, or it did not.

Case 1: $maxht(T) = maxht(T')$. Since the insertion into T replaced an external node u on some level $k < h$ by an internal node with two external node children, $U(T)$ and $U(T')$ differ only with respect to node u and its replacement. The contribution of u to $U(T)$ is $h-k$, whereas the contribution of u's children in T' to $U(T')$ is $2(h-k-1)$. Hence,

$$\begin{aligned} U(T') - U(T) &= 2(h-k-1) - (h-k) \\ &= h-k-2 \end{aligned}$$

and

$$\begin{aligned} U(T') + wt(T') - U(T) - wt(T) &= h-k-2+1 \\ &\geq 0, \end{aligned}$$

since $k < h$.

Case 2: $maxht(T) < maxht(T')$. Then, $maxht(T) = maxht(T') - 1$, because the simple insertion at node u in tree T can increase the height by at most one.

Now, the contribution of u to $U(T)$ is zero, since node u must be on level $maxht(T)$. The contribution of the children of u in tree T' to $U(T')$ must also be zero, since they are on level $maxht(T) + 1 = maxht(T')$. But, the contribution of all other external nodes is increased by one; that is, $U(T') = U(T) + wt(T) - 1$. But, this implies that

$$\begin{aligned} U(T') + wt(T') - U(T) - wt(T) &= wt(T) - 1 + 1 \\ &= wt(T) \\ &\geq 0. \end{aligned}$$

□

Using Lemma 2.8, we can show that a sequence of simple insertions cannot decrease the value of $U(T) + wt(T)$.

Theorem 2.9 *Let T and T' be binary trees such that there is a sequence I of simple insertions that transforms T into T'. Then,*

$$U(T) + wt(T) \leq U(T') + wt(T').$$

Proof: By induction on the length of the sequence I.

If the length is zero, then $T = T'$ and the result clearly holds.

Otherwise, there is at least one simple insertion in I, so $wt(T') > wt(T)$ and $maxht(T') \geq maxht(T)$. Consider the first insertion in I. It transforms T into some tree T''. By Lemma 2.8, $U(T) + wt(T) \leq U(T'') + wt(T'')$. Let I'' be the sequence I without the first insertion; I'' is a sequence of simple insertions which transforms T'' into T'. The induction hypothesis applies to T', T'', and I'', since the length of I'' is less than the length of I, so we also have $U(T'') + wt(T'') \leq U(T') + wt(T')$. Combining these inequalities gives us the required result. □

We can now prove that any member of $Skinny(h)$ has the minimum value for $U(T) + wt(T)$ among all red-black trees T of maximum height h.

Corollary 2.10 *For any red-black tree T of maximum height h,*

$$U(T) + wt(T) \geq U(Skinny(h)) + wt(Skinny(h)).$$

Proof: Let T be an arbitrary red-black tree of maximum height h. By Theorem 2.7, there exists a sequence I of simple insertions that transforms a member of $Skinny(h)$ into T and each intermediate tree is a red-black tree. Therefore, by Theorem 2.9, we have $U(Skinny(h)) + wt(Skinny(h)) \leq U(T) + wt(T)$. □

By Corollary 2.4, we have $U(Skinny(h)) + wt(Skinny(h)) \leq h2^{(h-1)/2}$. By Corollary 2.10, we have $U(T) + wt(T) \geq U(Skinny(h)) + wt(Skinny(h))$, for any red-black tree of maximum height h. Therefore,

$$IPL(T) \leq h \cdot wt(T) - h2^{(h-1)/2} + 1,$$

where T is a red-black tree of maximum height h. But there may be red-black trees of different maximum heights with the same weight, so we need to discover which maximum height maximizes our bound.

Theorem 2.11 *Let T be a red-black tree of size N. Then,*

$$\mathrm{IPL}(T) \leq 2N(\log N - \log\log N) + O(N).$$

Proof: By the above discussion, we have $IPL(T) \leq h(N+1) - h2^{(h-1)/2} + 1$. Let $f(x) = x(N+1) - x2^{(x-1)/2} + 1$. This function takes its extremal value at the zero of its first derivative:

$$\frac{df}{dx} = (N+1) - (2^{(x-1)/2} + \frac{1}{2} \cdot \ln 2 \cdot x2^{(x-1)/2}).$$

Consider the second derivative of $f(x)$:

$$f''(x) = -1/2 \cdot \ln 2 \cdot 2^{(x-1)/2}(2 + 1/2 \cdot \ln 2 \cdot x).$$

Clearly, $f''(x) < 0$, for $x > 0$. Thus, $f'(x)$ is a decreasing function for $x > 0$. But,

$$\begin{aligned} f'(0) &= (N+1) - (2^{-1/2} + 1/2 \cdot \ln 2 \cdot 0 \cdot 2^{-1/2}) \\ &\approx (N+1) - 0.7071 \\ &> 0, \quad \text{since } N \geq 0. \end{aligned}$$

Therefore, $f'(x)$ has exactly one zero at some point $x_0 > 0$. Furthermore, we have $f'(x) > 0$, for $0 < x < x_0$, and we have $f'(x) < 0$, for $x_0 < x$. Thus, the zero of $f'(x)$ is a maximum of $f(x)$.

Now, we wish to find an x satisfying

$$2^{(x-1)/2}\left(1 + \frac{\ln 2}{2}x\right) = (N+1).$$

Taking logarithms on both sides, we get

$$(x-1)/2 + \log x + O(1) = \log N + O(1).$$

Adding $\log\log N - \log x$ to both sides, we get

$$\begin{aligned}(x-1)/2 + \log\log N &= \log N + \log\left(\frac{\log N}{x}\right) + O(1)\\ &= \log N + \log\left(\frac{(x-1)/2 + \log x + O(1)}{x}\right) + O(1)\\ &= \log N + \log\left(\frac{1}{2} + \frac{\log x + O(1)}{x}\right) + O(1)\\ &= \log N + O(1).\end{aligned}$$

Therefore, $f(x)$ takes its maximum value at $x = 2(\log N - \log\log N) + O(1)$, and the value of $f(x)$ at that point is

$$\begin{aligned}&[2(\log N - \log\log N) + O(1)](N+1)\\ &- [2(\log N - \log\log N) + O(1)]2^{\log N - \log\log N + O(1)} + 1\\ &= 2N(\log N - \log\log N) + 2(\log N - \log\log N) + O(1)(N+1)\\ &\quad - [2(\log N - \log\log N) + O(1)]\frac{N}{\log N}2^{O(1)} + 1\\ &= 2N(\log N - \log\log N) + O(N).\end{aligned}$$

□

3 A Class of Red-Black Trees with Asymptotically Pessimal IPL

Having established an upper bound on the internal path length, we now prove that it is asymptotically tight. We begin by demonstrating that the *Skinny*(h) trees do not achieve the upper bound.

Theorem 3.1 *Let Skinny(h) have size N. Then,*

$$IPL(Skinny(h)) \leq 4/3 \cdot N \lg N + O(N).$$

Proof: From Lemma 2.1, we know that

$$IPL(Skinny(h)) = h \cdot wt(Skinny(h)) + 1 - (U(Skinny(h)) + wt(Skinny(h))).$$

If h is even, we have

$$\begin{aligned}IPL(Skinny(h)) &= h(2 \cdot 2^{h/2} - 1) + 1 - ((h-1)2^{h/2} + 2)\\ &= (h+1)2^{h/2} - h - 1.\end{aligned}$$

But, $wt(Skinny(h)) = N + 1 = 2 \cdot 2^{h/2} - 1$, when h is even. Therefore, $h = 2\log((N+2)/2)$. Thus, we have

$$\begin{aligned} IPL(Skinny(h)) &= (2\log((N+2)/2)+1)2^{\log((N+2)/2)} \\ &\quad - 2\log((N+2)/2) - 1 \\ &= N\log N + O(N), \quad \text{since } \log(N+2) = \log N + O(1). \end{aligned}$$

If h is odd, we have

$$\begin{aligned} IPL(Skinny(h)) &\leq h(3 \cdot 2^{(h-1)/2} - 1) + 1 - h2^{(h-1)/2} \\ &= 2 \cdot h \cdot 2^{(h-1)/2} - h + 1, \end{aligned}$$

since $U(Skinny(h)) + wt(Skinny(h)) \geq h2^{(h-1)/2}$ when h is odd. Also, $wt(Skinny(h)) = N + 1 = 3 \cdot 2^{(h-1)/2} - 1$, when h is odd. Therefore, $h = 2\log((N+2)/3) + 1$. Thus, we have

$$\begin{aligned} IPL(Skinny(h)) &\leq 2(2\log((N+2)/3)+1)2^{\log((N+2)/3)} \\ &\quad - 2\log((N+2)/3) \\ &= 4/3 \cdot N\log N + O(N). \end{aligned}$$

□

We now present a class of red-black trees, some of which achieve the upper bound for the internal path length presented in the previous section. These are the $C(k,h)$ trees. Their definition is given in Figure 5.

Lemma 3.2 *The weight of a $C(k,h)$ tree is*

$$2^h + (2 + (h+k) \bmod 2)2^{\lfloor (h+k)/2 \rfloor} - (2 + h \bmod 2)2^{\lfloor h/2 \rfloor}.$$

Proof: If h and k are both even, then, from Figure 5, we see that

$$\begin{aligned} wt(C(k,h)) &= 2^h + \sum_{j=h/2}^{(h+k-2)/2} 2 \cdot 2^j \\ &= 2^h + 2 \cdot 2^{(h+k)/2} - 2 \cdot 2^{h/2}. \end{aligned}$$

Since h and k are both even, we have $(h+k) \bmod 2 = 0$ and $h \bmod 2 = 0$. Furthermore, $(h+k)/2 = \lfloor (h+k)/2 \rfloor$ and $h/2 = \lfloor h/2 \rfloor$. Thus, we obtain

$$wt(C(k,h)) = 2^h + (2 + (h+k) \bmod 2)2^{\lfloor (h+k)/2 \rfloor} - (2 + h \bmod 2)2^{\lfloor h/2 \rfloor}$$

when both h and k are even.

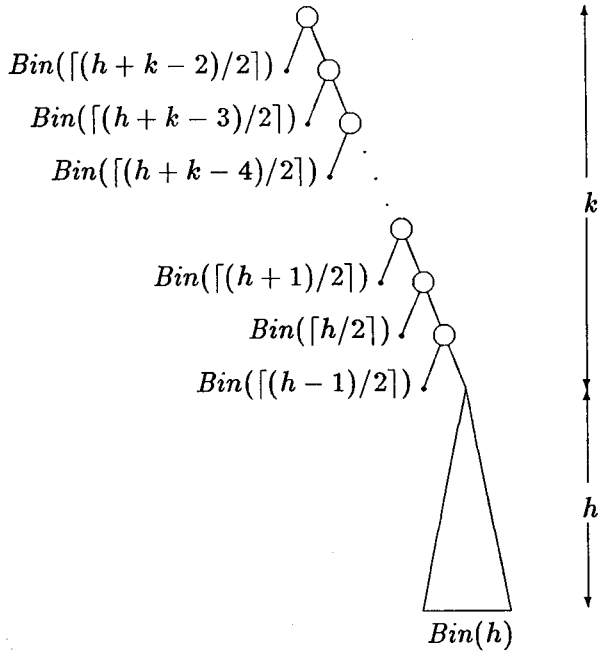


Figure 5: A $C(k, h)$ tree.

If h is even and k is odd, then, from Figure 5, we see that

$$\begin{aligned} wt(C(k,h)) &= 2^h + 2^{(h+k-1)/2} + \sum_{j=h/2}^{(h+k-3)/2} 2 \cdot 2^j \\ &= 2^h + 2^{(h+k-1)/2} + 2 \cdot 2^{(h+k-1)/2} - 2 \cdot 2^{h/2} \\ &= 2^h + 3 \cdot 2^{(h+k-1)/2} - 2 \cdot 2^{h/2}. \end{aligned}$$

Since h is even, we have $h \bmod 2 = 0$ and $h/2 = \lfloor h/2 \rfloor$. Since k is odd, we have $(h+k) \bmod 2 = 1$ and $(h+k-1)/2 = \lfloor (h+k)/2 \rfloor$. Thus, we obtain

$$wt(C(k,h)) = 2^h + (2 + (h+k) \bmod 2)2^{\lfloor (h+k)/2 \rfloor} - (2 + h \bmod 2)2^{\lfloor h/2 \rfloor}$$

when h is even and k is odd.

If h and k are both odd, then, from Figure 5, we see that

$$\begin{aligned} wt(C(k,h)) &= 2^h + 2^{(h-1)/2} + \sum_{j=(h+1)/2}^{(h+k-2)/2} 2 \cdot 2^j \\ &= 2^h + 2^{(h-1)/2} + 2 \cdot 2^{(h+k)/2} - 2 \cdot 2^{(h+1)/2} \\ &= 2^h + 2 \cdot 2^{(h+k)/2} - 3 \cdot 2^{(h-1)/2}. \end{aligned}$$

Since h is odd, we have $h \bmod 2 = 1$ and $(h-1)/2 = \lfloor h/2 \rfloor$. Since k is odd, we have $(h+k) \bmod 2 = 0$ and $\frac{h+k}{2} = \lfloor \frac{h+k}{2} \rfloor$. Again, we obtain

$$wt(C(k,h)) = 2^h + (2 + (h+k) \bmod 2)2^{\lfloor (h+k)/2 \rfloor} - (2 + h \bmod 2)2^{\lfloor h/2 \rfloor}$$

when h and k are both odd.

Finally, if h is odd and k is even, then, from Figure 5, we see that

$$\begin{aligned} wt(C(k,h)) &= 2^h + 2^{(h-1)/2} + 2^{(h+k-1)/2} + \sum_{j=(h+1)/2}^{(h+k-3)/2} 2 \cdot 2^j \\ &= 2^h + 2^{(h-1)/2} + 2^{(h+k-1)/2} + 2 \cdot 2^{(h+k-1)/2} - 2 \cdot 2^{(h+1)/2} \\ &= 2^h + 3 \cdot 2^{(h+k-1)/2} - 3 \cdot 2^{(h-1)/2}. \end{aligned}$$

Since h is odd, we have $h \bmod 2 = 1$ and $(h-1)/2 = \lfloor h/2 \rfloor$. Since k is even, we have $(h+k) \bmod 2 = 1$ and $(h+k-1)/2 = \lfloor (h+k)/2 \rfloor$. Therefore, as in the other cases, we have

$$wt(C(k,h)) = 2^h + (2 + (h+k) \bmod 2)2^{\lfloor (h+k)/2 \rfloor} - (2 + h \bmod 2)2^{\lfloor h/2 \rfloor}$$

when h is odd and k even. □

Now we show that the $C(k,h)$ trees achieve the upper bound for the internal path length when k is chosen appropriately.

Theorem 3.3 *Let $k = h - 2\log h + \epsilon$, where $|\epsilon| \leq \frac{1}{2}$, and let $C(k,h)$ have size N. Then, $IPL(C(k,h)) = 2N(\log N - \log\log N) + O(N)$.*

Proof: First, we show that $maxht(C(k,h)) = h+k = 2(\log N - \log\log N) + O(1)$.

Since $k = h - 2\log h + \epsilon$, we have

$$\begin{aligned} h + k &= 2(h - \log h) + \epsilon \\ &= 2\log\frac{2^h}{h} + \epsilon. \end{aligned} \tag{1}$$

Now,

$$\begin{aligned} wt(C(k,h)) &= 2^h + (2 + (h+k) \bmod 2)2^{\lfloor (h+k)/2 \rfloor} - (2 + h \bmod 2)2^{\lfloor h/2 \rfloor} \\ &= 2^h + 2^{(h+k)/2+1+\epsilon_1} - 2^{h/2+1+\epsilon_2}, \end{aligned}$$

where

$$\epsilon_1 = \begin{cases} 0 & \text{if } (h+k) \bmod 2 = 0 \\ \log 3 - \frac{3}{2} \approx .08496 & \text{otherwise} \end{cases}$$

and

$$\epsilon_2 = \begin{cases} 0 & \text{if } h \bmod 2 = 0 \\ \log 3 - \frac{3}{2} \approx .08496 & \text{otherwise.} \end{cases}$$

Thus, we have

$$wt(C(k,h)) = 2^h + (\sqrt{2})^{h+k}(2^{1+\epsilon_1} - \frac{2^{1+\epsilon_2}}{(\sqrt{2})^k}).$$

From Equation (1), we get

$$(\sqrt{2})^{h+k} = (\sqrt{2})^{2\log\frac{2^h}{h}+\epsilon} = \frac{2^h}{h}2^{\epsilon/2}.$$

Substituting this into the previous equation, we have

$$wt(C(k,h)) = 2^h\left(1 + \frac{2^{\epsilon/2}}{h}\left(2^{1+\epsilon_1} - \frac{2^{1+\epsilon_2}}{(\sqrt{2})^k}\right)\right) \tag{2}$$

$$= (\sqrt{2})^{h+k}\left(\frac{h}{2^{\epsilon/2}} + \left(2^{1+\epsilon_1} - \frac{2^{1+\epsilon_2}}{(\sqrt{2})^k}\right)\right). \tag{3}$$

Taking logarithms on both sides of Equation (3), we obtain

$$\log N + O(1) = \frac{h+k}{2} + \log\left(\frac{h}{2^{\epsilon/2}} + \left(2^{1+\epsilon_1} - \frac{2^{1+\epsilon_2}}{(\sqrt{2})^k}\right)\right) \tag{4}$$

and, from Equation (2), we get

$$\log N + O(1) = h + \log\left(1 + \frac{2^{\epsilon/2}}{h}\left(2^{1+\epsilon_1} - \frac{2^{1+\epsilon_2}}{(\sqrt{2})^k}\right)\right).$$

But

$$2^{1+\epsilon_1} - \frac{2^{1+\epsilon_2}}{(\sqrt{2})^k} < 2^2,$$

so

$$h = \log N + O(1).$$

Returning to Equation (4), we see that

$$\begin{aligned}
&\log\left(\frac{h}{2^{\epsilon/2}} + \left(2^{1+\epsilon_1} - \frac{2^{1+\epsilon_2}}{(\sqrt{2})^k}\right)\right) \\
&= \log\frac{h}{2^{\epsilon/2}} + O(1) \\
&= \log h - \frac{\epsilon}{2} + O(1) \\
&= \log h + O(1) \\
&= \log\log N + O(1), \quad \text{since } h = \log N + O(1).
\end{aligned}$$

Substituting this into Equation (4), we have

$$\log N + O(1) = \frac{h+k}{2} + \log\log N + O(1),$$

or

$$h + k = 2(\log N - \log\log N) + O(1),$$

as required.

Now, let us show that $U(C(k,h)) = O(N)$. Because $C(k,h)$ is *Skinny*$(h+k)$ with the area beneath some of the external nodes of *Skinny*$(h+k)$ completely filled in, we know that $U(C(k,h)) \leq U(\textit{Skinny}(h+k))$. There are two cases to consider: $h+k$ is even and $h+k$ is odd.

- $h+k$ is even.

  Then,

$$\begin{aligned}
U(C(k,h)) &\leq U(\textit{Skinny}(h+k)) \\
&= (h+k-3)2^{(h+k)/2} + 3 \\
&= [2(\log N - \log\log N) + O(1) - 3]\frac{N}{\log N}2^{O(1)} + 3 \\
&= 2^{O(1)}[2N - 2\frac{\log\log N}{\log N} + O(1)\frac{N}{\log N}] + 3 \\
&= O(N).
\end{aligned}$$

- $h + k$ is odd.

  Then,

$$\begin{aligned} U(C(k,h)) &\leq U(Skinny(h+k)) \\ &= \frac{3}{2}(h+k-3)2^{(h+k-1)/2}+3 \\ &= \frac{3}{2}[2(\log N - \log\log N)+O(1)-3]\frac{N}{\log N}\frac{2^{O(1)}}{2}+3 \\ &= 2^{O(1)}\frac{3}{4}[2N - 2\frac{\log\log N}{\log N}+O(1)\frac{N}{\log N}]+3 \\ &= O(N). \end{aligned}$$

In both cases, $U(C(k,h)) = O(N)$, and by Lemma 2.1,

$$\begin{aligned} IPL(C(k,h)) &= (maxht(C(k,h))-1)(N+1)+1-U(C(k,h)) \\ &= [2(\log N - \log\log N)+O(1)](N+1)+1-O(N) \\ &= 2N(\log N - \log\log N)+O(N). \end{aligned}$$

□

4 Conclusion

We have shown that for a red-black tree T of size N

$$IPL(T) \leq 2N(\log N - \log\log N)+O(N)$$

and have shown that the $C(k,h)$ trees with $k = h - 2\log h + \epsilon$, where $|\epsilon| \leq 1/2$, achieve this bound.

Although a subset of the $C(k,h)$ trees achieve the bound, we do not know whether these trees have maximal IPL for their size. The problem of characterizing maximal IPL red-black trees is still very much open. (The same problem remains open for AVL trees, too.)

References

[Bay72] R. Bayer. Symmetric binary B-trees: Data structure and maintenance algorithms. *Acta Informatica*, 1(4):290–306, 1972.

[GS78] L.J. Guibas and R. Sedgewick. A dichromatic framework for balanced trees. In *Proceedings of the 19th Annual IEEE Symposium on Foundations of Computer Science*, pages 8–21, 1978.

[Knu73] Donald E. Knuth. *The Art of Computer Programming, Vol. 1: Fundamental Algorithms.* Addison-Wesley, Reading, MA, 1973.

[KW89] Rolf Klein and Derick Wood. A tight upper bound for the path length of AVL trees. *Theoretical Computer Science*, 1989. To appear.

[MPRS79] Raymond E. Miller, Nicholas Pippenger, Arnold L. Rosenberg, and Lawrence Snyder. Optimal 2-3-trees. *SIAM Journal on Computing*, 8(1):42–59, 1979.

[Oli82] H.J. Olivié. A new class of balanced search trees: Half-balanced binary search trees. *RAIRO Informatique Théorique*, 16:51–71, 1982.

[OW82] Th. Ottmann and D. Wood. A comparison of iterative and defined classes of search trees. *International Journal of Computer and Information Sciences*, 11(3):155–178, June 1982.

[Tar83] Robert Endre Tarjan. Updating a balanced search tree in $O(1)$ rotations. *Information Processing Letters*, 16:253–257, 1983.