A New Proof for the Sequential Access Theorem for Splay Trees

Amr Elmasry
Computer Science Department
Rutgers University
New Brunswick, NJ 08903
USA

Abstract: We give a new, simple proof for the sequential access theorem for splay trees. For an n-node splay tree, our bound on the number of rotations is $4.5n$, with a smaller constant than the bound of $10.8n$ concluded by Tarjan. Our proof provides additional insights into the workings of splay trees.

Key-words: Algorithms - Data structures - Splay trees - Amortized analysis - Combinatorial problems - Self-adjusting structures.

1 Introduction

A binary search tree is a binary tree whose nodes contain items in symmetric order. In other words, for any node, x, all the items in the left sub-tree of x are less or equal to the item in x, and all the items in the right sub-tree of x are greater. A splay tree [1] is a self-adjusting binary search tree, which supports a restructuring operation of the tree called *splay*. The splay operation consists of a sequence of rotations of edges. A rotation of an edge maintains the symmetric property of the tree. When any node of the tree is accessed, a splay is performed at this node. Let $p(x)$ be the parent of x. A splay at x repeats the following step until x becomes the root of the tree.

Zig case. If $p(x)$ is the root: Make x the new root by rotating the edge joing x and $p(x)$.

Zig-zig case. If x and $p(x)$ are both left children or right children: Rotate the edge joining $p(x)$ to its parent, and then rotate the edge joining x to $p(x)$.

Zig-zag case. If x is a left child and $p(x)$ is a right child or vice versa: Rotate the edge joining x to $p(x)$, and then rotate the edge joining x to its new parent.

We give a new proof for the sequential access theorem, bounding the number of rotations with at most $4.5n$. The sequential access theorem states that the number of rotations required to access each of the n nodes of an arbitrary initial splay tree once, in symmetric order, is linear. Tarjan [3] proved that the bound for the number of rotations is $10.8n$. Sundar [2] gave an easier proof that uses a potential function technique. His bound for the number of rotations is $15n$.

Tarjan conjectured that a sequence of k deque operations on an arbitrary n-node splay tree takes $O(n + k)$ rotations. He proved a special case of this conjecture [3] for output restricted deques. Sundar [2] gave an inverse Ackerman upper bound. Applying our proof reduces the constant involved in Tarjan's proof. More interesting is to try to extend our proof to prove the deque conjecture.

2 The proof

In a binary tree, the left spine of a sub-tree is defined to be the path from the root of this sub-tree to its leftmost leaf. In other words, every node on the path is the left child of its predecessor. The right spine is defined analogously.

We may think about the sequential access theorem for splay trees as repeated splaying on the leftmost leaf of the right sub-tree of the root. As a result of a splaying operation on such node, this node becomes the root of the tree and the old root becomes the root of the left sub-tree. Hence, we may ignore the left sub-tree (which is always a path) entirely and only keep track of the right sub-tree. We call the left spine of the right sub-tree of the root, the splaying spine.

Before a splay operation, name the nodes on the splaying spine, x_i, such that x_i is the left child of x_{i+1}, for all i starting from 0. As a result of a splaying operation performed on x_0, the following restructuring takes place. The node x_0 becomes the root of the tree, leaving, on the splaying spine, its right child and the nodes on the left spine of this right child's sub-tree. For every even value of i, x_i is linked to x_{i-1} as its right child, and the right sub-tree of x_{i-1} becomes the left sub-tree of x_i.

For the purpose of the proof, we use a coloring scheme to distinguish some nodes from others. The following coloring rules are applied:

- Initially, all the nodes are uncolored.
- When an uncolored node becomes a node on the splaying spine, it is colored yellow.
- When a yellow node is linked to another node, this yellow node is colored red.

- When a red node becomes a node on the splaying spine, this node and all the red descendents of its right child are colored green.
- When a green node is linked to a yellow node, the yellow node is colored green.
- All the nodes on the right spine of the tree are colored black, overriding the above coloring rules.

The splay operations are numbered, starting from 1. We assume that the splay operation number t takes place at time t. Consider any node x in the tree. Let $g_x(t)$ be the number of the colored nodes on the right spine of x, after the splay operation t. Let $h_x(t)$ be the number of the colored nodes on the right spine of the left child of x, after the same splay operation. If x does not have a left child, then $h_x(t)$ is equal to 0. Before any splay operation, $g_x(0)$ and $h_x(0)$ are defined. Define $v_x(t)$ to be equal to $g_x(t) - h_x(t)$.

Consider any node w and its left child z, such that w is linked to z during the splay operation $t+1$, for any $t \geq 0$. The following relations hold:

$$h_w(t) = g_z(t) \tag{1}$$

$$g_w(t+1) = g_w(t) \tag{2}$$

$$h_w(t+1) = h_w(t) - 1 \tag{3}$$

$$g_z(t+1) = g_w(t) + 1 \tag{4}$$

$$h_z(t+1) \leq h_z(t) + 1 \tag{5}$$

The following lemma follows

Lemma 1
1. For any non-black node x, $v_x(t) \geq 0$, and $v_x(t+1) \geq v_x(t)$.
2. If, at time t, x is red or green, then $v_x(t) > 0$.
3. If, at time t, w is not black, and is linked to z at time $t+1$, then $v_w(t+1) > v_w(t)$. If, at time t, w is green, then $v_z(t+1) > v_z(t)$.

Proof. We prove the lemma by induction on time. For any node, x, the base case follows from the fact that $v_x(0) = 0$. Using the induction hypothesis for node w at time t, assuming that w is not black, then $v_w(t) \geq 0$. Using (2) and (3), then $v_w(t+1) > v_w(t)$, which implies $v_w(t+1) > 0$. Since w becomes red after this link, the hypothesis is true for the node w at time $t+1$. The relation $v_w(t) \geq 0$ together with (1) and (4) implies $g_z(t+1) \geq g_z(t) + 1$. Using the latter relation and (5), then $v_z(t+1) \geq v_z(t)$, and the hypothesis is true for node z at time $t+1$. If w was green at time t, then the induction hypothesis becomes $v_w(t) > 0$. The same calculations imply $v_z(t+1) > v_z(t)$. The hypothesis is then true for all the non-black nodes throughout the algorithm. □

We use the accounting method for bounding the number of rotations. When a node is colored yellow, it is given one credit for a total of n credits. Other than the links that involve black nodes, there are four possible types of links: yellow-to-yellow, yellow-to-green, green-to-yellow and green-to-green. For the first three types, the color of a yellow node changes. We use the credit on this node to pay for the link. Using Lemma 1, for any green node, z, $v_z(t) > 0$. If a green node is linked to z while $v_z(t) = 1$, we call this link a green-to-green A-link. Using Lemma 1, any node may gain at most one child by a green-to-green A-link. Hence, the number of rotations accompanying these links is at most n. If $v_z(t) \geq 2$, the link is called a green-to-green B-link. We keep the invariant that, after the splay operation t, there are $\frac{h_x^2(t)}{2}$ credits on any green node, x. Next, we show that these credits are enough to pay for all the

green-to-green B-links, while maintaining the invariant. Assume that w is linked to z by a green-to-green B-link. Let d be the difference between the sum of the number of credits on w and z before the splay operation $t+1$ and those needed after the splay operation $t+1$. Then

$$d = \frac{h_z^2(t)}{2} + \frac{h_w^2(t)}{2} - \frac{h_z^2(t+1)}{2} - \frac{h_w^2(t+1)}{2}.$$

Using (3) and (5), then

$$d \geq h_w(t) - h_z(t) - 1.$$

Using (1) together with the fact that for all the green-to-green B-links $v_z(t) \geq 2$, then $d \geq 1$. This extra credit is used to pay for the link of w to z.

To keep the invariant hold, when a red node becomes a node on the splaying spine at time t, this red node and all the red descendents of its right child are colored green and are given credits. Precisely, a credit of $\frac{h_x^2(t)}{2}$ is given to every red node, x, of this sub-tree. These extra credits are enough to keep the invariant hold for all the green nodes. The following lemmas bound the total number of such credits.

Lemma 2 *For any m-node sub-tree τ at time t, there are at most $\frac{m}{2^{i+1}}$ red nodes whose $h(t)$ equals i.*

Proof. An equivalent statement to the lemma would be: Except for the right spine of τ, there are at most $\frac{m}{2^{i+1}}$ red nodes whose $g(t)$ equals i. The prove is by backward induction on i. For the base case, we show that, except for the right spine of τ, there is no red node whose $g(t)$ is greater or equal to $\log m$. Assume that such a node exists, and call it x. This means that the, at least, $\log m$ nodes on the right spine of x must have been nodes on the splaying spine, for at least $\log m$ splaying steps. For each of these steps, the number of the nodes on the splaying spine should have been halved. This means that x must be on the right spine of τ, which is a contradiction. It also follows that, except for the right spine of τ, there is at most one red node whose $g(t)$ equals $\log m - 1$. Using the induction hypothesis, there are at most $\frac{m}{2^{i+1}}$ red nodes, not on the right spine of τ, whose $g(t)$ equals i. Consider any of these nodes, and call it y. The value of $g(t)$ for the right child of y is $i-1$. Using Lemma 1, the value of $g(t)$ for the left child of y is at most $i-1$. Therefore, every node, y, may have at most two red descendents whose $g(t)$ equals $i-1$. Using Lemma 1, the red nodes whose $g(t)$ equals $i-1$ must be descendents of the red nodes whose $g(t)$ equals i. Therefore, the total number of the red nodes, not on the right spine of τ, whose $g(t)$ equals $i-1$ is at most $\frac{n}{2^i}$. The hypothesis is true, and the lemma follows. □

Lemma 3 *The total number of credits given to the red nodes to cover the green-to-green B-links is less than* $1.5n$.

Proof. Let C be the total number of such credits. Using Lemma 2, and taking the summation over all the credited nodes, then

$$\begin{aligned} C &= \sum_x \frac{h_x^2}{2}, \\ &\leq \sum_{i=1}^{\log n - 1} \frac{n}{2^{i+1}} \frac{i^2}{2}, \\ &< 1.5n. \end{aligned}$$

□

When the number of nodes on the splaying spine is even, the root of the splaying spine is not linked to another node. The relations defined earlier may not hold for this node. Fortunately, this node is colored black. The black nodes are involved in at most one rotation per splay operation. This costs at most n extra rotations.

The sequential access theorem follows by adding the following bounds: The n credits on

the yellow nodes, the n rotations accompanying the green-to-green A-links, the $1.5n$ credits accompanying the green-to-green B-links, and the n rotations that involve black nodes.

References:
[1] D. Sleator and R. Tarjan, Self-adjusting binary search trees, *J. ACM,* 32(3), 1985, pp. 652-686.
[2] R. Sundar, On the deque conjecture for the splay algorithm, *Combinatorica,* 12, 1992, pp. 95-124.
[3] R. Tarjan, Sequential access in splay trees takes linear time, *Combinatorica,* 5, 1985, pp. 367-378.