

Amortized Analysis

Theorem 1. *Using splay trees, a sequence of m operations involving n inserts takes time $O(m \log n)$.*

We will use the accounting method.

We assume that each node x of the splay tree has a savings account containing a certain number of **credits**. When x is inserted, its account is opened and some credits deposited. These will be used later for restructuring operations.

For a tree S , define $\mu(S) = \lfloor \log(|S|) \rfloor$, where $|S|$ is the size of the (number of nodes in) tree S .

For a node x , let $S(x)$ denote the subtree rooted as x . We define $\mu(x) = \lfloor \log(|S(x)|) \rfloor$ where $|S(x)|$ is the size of the subtree $S(x)$.

We maintain the following **credit invariant** for every node x : x always has at least $\mu(x)$ credits on deposit in its account.

Lemma 1. *Each operation $SPLAY(x, S)$ requires no more than $3(\mu(S) - \mu(x)) + 1$ credits to perform the operation and maintain the credit invariant.*

Proof: Let y be the parent of x and z be the parent of y , if it exists.

Let μ and μ' be the values of μ before and after the $SPLAY$ operation, respectively.

We consider 3 cases corresponding to the 3 cases of rotation.

Case 1: Node z does not exist.

This is the last rotation in the splay — we do a single $\text{ROTATE}(x)$.

We are willing to pay no more than $3(\mu'(x) - \mu(x)) + 1$ credits for this rotation.

Note that $\mu'(x) = \mu(y)$ and $\mu'(y) \leq \mu'(x)$.

To maintain the invariant, we need to spend

$$\mu'(x) + \mu'(y) - \mu(x) - \mu(y) = \mu'(y) - \mu(x)$$

$$\leq \mu'(x) - \mu(x)$$

$$\leq 3(\mu'(x) - \mu(x))$$

Thus we are left with at least one credit to pay for the constant number of low-level operations.

Case 2: Node x is the left child y and y is the left child of z (or both x and y are right children).

We show that it costs no more than $3(\mu'(x) - \mu(x))$ credits to perform $\text{ROTATE}(y)$ and then $\text{ROTATE}(x)$ and maintain the credit invariant.

Thus if a sequence of Case 2 rotations are performed till x becomes the root, the total amount spent will be no more than $3(\mu(S) - \mu(x)) + 1$.

To maintain the invariant, we need to spend

$$\begin{aligned} & \mu'(x) + \mu'(y) + \mu'(z) - \mu(x) - \mu(y) - \mu(z) \quad \text{--- (1)} \\ &= \mu'(y) + \mu'(z) - \mu(x) - \mu(y) \\ &= (\mu'(y) - \mu(x)) + (\mu'(z) - \mu(y)) \\ &\leq (\mu'(x) - \mu(x)) + (\mu'(x) - \mu(x)) \\ &= 2(\mu'(x) - \mu(x)). \end{aligned}$$

If $\mu'(x) > \mu(x)$, we have $\mu'(x) - \mu(x)$ credits left over to pay for constant number of low-level operations needed to perform the two rotations.

Otherwise, if $\mu'(x) = \mu(x)$, we show that quantity (1) is strictly negative. Thus invariant is maintained for free and we can use the saved credit to pay for the low level operations.

We show by contradiction. Suppose $\mu'(x) = \mu(x)$ and quantity (1) is ≥ 0 .

Since $\mu(z) = \mu'(x) = \mu(x)$, we have $\mu(x) = \mu(y) = \mu(z)$.

Therefore, $\mu'(x) + \mu'(y) + \mu'(z) \geq 3\mu(z) = 3\mu'(x)$.

So, $\mu'(y) + \mu'(z) \geq 2\mu'(x)$.

Since $\mu'(y) \leq \mu'(x)$ and $\mu'(z) \leq \mu'(x)$, it follows that

$$\mu'(x) = \mu'(y) = \mu'(z).$$

Thus, $\mu(x) = \mu(y) = \mu(z) = \mu'(x) = \mu'(y) = \mu'(z)$.

If a is the size of the subtree rooted at x *before* the operation and b is the size of the subtree rooted at z after the operation, then the above implies:

$$\lfloor \log a \rfloor = \lfloor \log(a + b + 1) \rfloor = \lfloor \log b \rfloor.$$

Assuming w.l.o.g, $a \leq b$, we get a contradiction:

$$\lfloor \log(a + b + 1) \rfloor \geq \lfloor \log 2a \rfloor = 1 + \lfloor \log a \rfloor > \lfloor \log a \rfloor.$$

Case 3: Node x is a left child of y and y is a right child of z , or vice versa.

Here we do $\text{ROTATE}(x)$ followed by $\text{ROTATE}(x)$ again. We are willing to pay no more than $3(\mu'(x) - \mu(x))$ credits for these two rotations and maintain the credit invariant. The analysis proceeds similar to Case 2.

Main Result

Theorem 2. *Using splay trees, a sequence of m operations involving n inserts takes time $O(m \log n)$.*

Proof. We note that maximum value of $\mu(x)$ is $\lfloor \log n \rfloor$.

Thus, from the lemma, at most $3\lfloor \log n \rfloor + 1$ credits are needed for each splay operation.

Since each of the operations SEARCH, INSERT, DELETE, SPLIT, and JOIN can be done using a constant number of splay operations and a constant number of low-level operations each of these operations costs $O(\log n)$ credits.

When we insert a new item, we charge at most $O(\log n)$ credits to be deposited into its account for future use.

Since each operation uses at most $O(\log n)$ credits, the total cost for m operations is at most $O(m \log n)$. \square