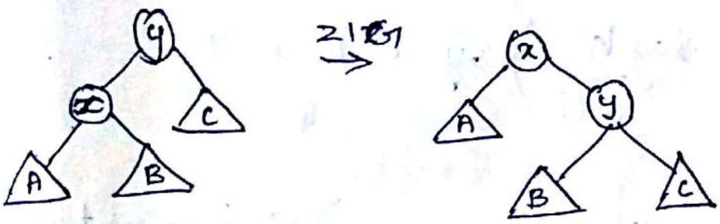
Homework Assignment-4

Dhayalini Nagaraj
A20359686

1. We could get the same effect as splaying (moving the accessed item to the root) using only zig and zag steps, not bothering with the more complicated zig-zig, zig-zag, zag-zig and zag-zag steps. If we do that, we only need case 1 on pages 4-5 in the notes. What is the amortized cost of an access with such simple splaying?

Amortizedcost = Actual cost + change in potential

$$\hat{c}_i = c_i + \Phi(T_i) - \Phi(T_i - 1)$$



The change in potential is

$$\Delta\Phi = r'(x) + r'(y) - r(x) - r(y)$$

where $r'(x)$ and $r'(y)$ denote the ranks of x and y after the ZIG operation

By case 1 of Access Lemma, the amortized cost of ZIG or ZAG at node x is at most

$$1 + r'(x) - r(x)$$

Amortized cost is $1 + r'(x) - r(x)$ where 1 is the actual cost and $r'(x) - r(x)$ is the change in potential.

Let i be a node at depth d. Then for a node at depth d, the sum of the amortized ~~amortized~~ cost is

$$d + \log \frac{W}{w(i)}$$

Where W - the total weight of the items in the tree and w(i) - weight of the node i

With such simple splaying, the amortized cost of an access is

$$O\left(d + \log \frac{W}{w(i)}\right) \text{ if } i \text{ is in } T$$

$$O\left(d + \log \frac{W}{\min\{w(i^-), w(i^+)\}}\right) \text{ if } i \text{ is not in } T.$$

The point to be noted is, the cost is proportional to the depth of the splay and it is not O(1).

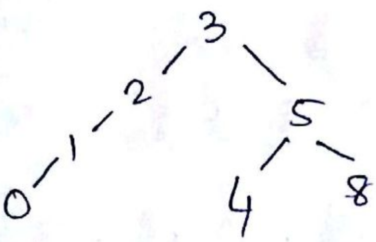
2) cut (i_1, i_2, T) that deletes all items i, $i_1 \le i \le i_2$ from the splay tree T

~~Given the tree contains many~~ many ~~nodes~~ The tree T contains many nodes.

cut (i_1, i_2, T) deletes all items i from splay tree.

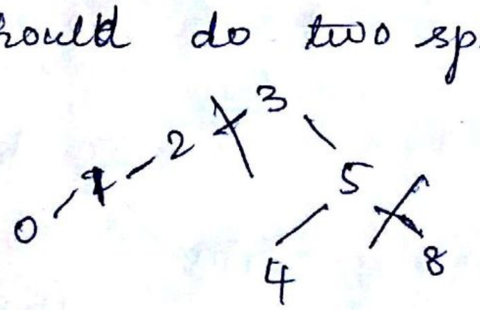
Given $i_1 \le i \le i_2$

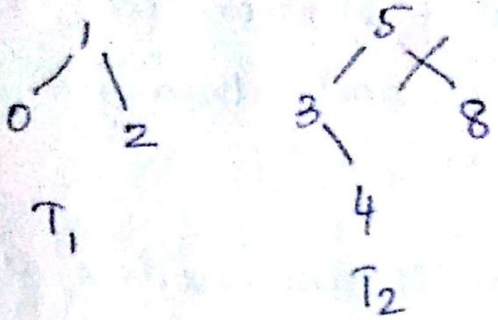
For example



cut (3, 5, T) should delete all the i that is ~~equ~~ greater than or equal to 3 and less than or equal 5 and the nodes inbetween

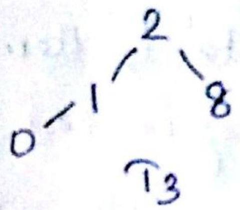
So inorder to perform cut (i_1, i_2, T) we should do two splits





after split only 8 will remain.

So 0, 1, 2 in tree T_1 and 8 in Tree T_2 has to be joined to form :

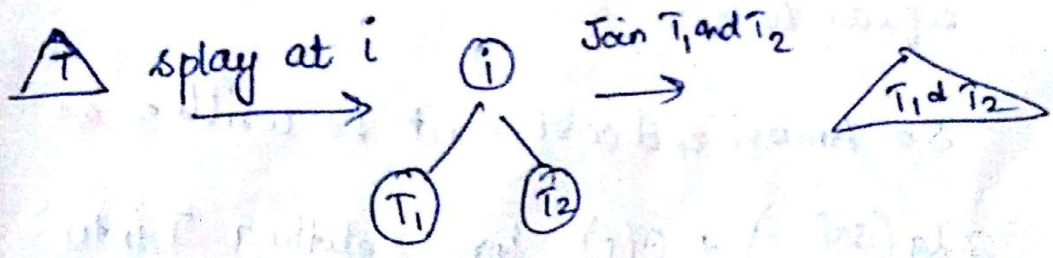


T_3 is the final tree after the cut $(3, 5, T)$.

cut (i_1, i_2, T) deletes all items i, $i_1 \le i \le i_2$

Delete : Takes a tree T and ~~an~~ deletes i from T. To perform delete we again do a split(T, i), then remove i and join the resulting subtrees.

(ie)



Inorder to perform cut operation we need to perform two splits and then join the remaining subtrees.

So split (T, i_1) takes T and node i_1 as input and creates two trees T_1 and T_2 such that T_1 contains all items that are smaller than i_1 and T_2 contains all items greater than or equal to i_1

Now split (T_2, i_2+1) takes T and nod i_2 as input and creates two trees T_3 and T_4 such that lesser than and equal to i_2 items are in T_3 and greater elements of i_2 are in T_4

Now join T_1 and T_3. then the cut (i_1, i_2, T) operation is completed.

i_2+1 is the node greater than i_2. so the split should happen in such a way T_4 has greater element of i_2 and T_3 lesser and equal to i_2.

So Amortized cost can be written as.

$3 \lg\left(\frac{W}{w(i_1)}\right) + O(1)$ for splitting T into T_1 and T_2

$3\lg\left(\frac{W}{w(i_2+1)}\right) + O(1)$ for splitting T_2 into T_3 and T_4

and Join (T_1, T_4) amortized cost is

$3\lg\left(\frac{W}{w(i)}\right) + O(1)$ when i is the last item in T_1.

So the sum of the amortized cost of these operations and change in potential that is the T_3 which has i. items that are $i_1 \le i \le i_2$ that is removed from the tree T gives the amortized cost of cut (i_1, i_2, T).

$$= 3\lg\left(\frac{W}{w(i_1)}\right) + 3\lg\left(\frac{W}{w(i_2+1)}\right) + 3\lg\left(\frac{W}{w(i)}\right) + O(1)$$

$$- 3\lg\left(\frac{W}{s(T_3)}\right)$$

$$W = s(T_1) + s(T_4)$$

This can also ~~written~~ as done. by deleting all the elements i, $i_1 \le i \le i_2$ by repeatedly doing the delete operation.

Then the amortized cost is

$$\cancel{3\lg\left(\frac{W - \sum_{i=i_1}^{i_2} w(i)}{\sum_{i=i_1}^{i_2}}\right)}$$

Then the amortized cost is

$$3 \lg \left(\frac{W - \sum_{i=i_1}^{i_2} w(i)}{\sum_{i=i_1}^{i_2} w(i^-)} \right) + 3 \lg \left(\frac{W}{\sum_{i=i_1}^{i_2} w(i)} \right) + O(1)$$

or

$$3 \lg \left(\frac{W - \sum_{i=i_1 \text{ to } i_2} w(i)}{\sum_{i=i_1 \text{ to } i_2} w(i^-)} \right) + 3 \lg \left(\frac{W}{\sum_{i=i_1 \text{ to } i_2} w(i)} \right) + O(1)$$

3) a) Given $d(x)$ be the depth of an internal node x.

Let n be the height of the tree T.

So. $S(n) = \sum_{\text{internal nodes } x \in T} 3^{-d(x)}$ for a complete binary tree with height 'n'

$$\therefore S(n) = \sum_{i=1}^{n} 2^{(i-1)} 3^{-i} \quad — ①$$

$$= \sum_{i=1}^{n} 2^{i} \cdot 2^{-1} 3^{-i}$$

$$= \sum_{i=1}^{n} 2^{i} 3^{-i} \cdot \frac{1}{2}$$

$$2S(n) = \sum_{i=1}^{n} 2^{i} 3^{-i}$$

$$= 2\left(1 - \frac{2}{3}\right)^{n} \leq 2 \quad - ②$$

Based on equality ② we can say,

$$S(n) \leq 1 \text{ for all } n.$$

Therefore for any ~~binary~~ finite binary tree we can get

$$\sum_{\text{internal nodes } x \in T} 3^{-d(x)} \leq 1$$

Hence Pseudo-Kraft inequality is proved.

b) We know that when $\lim_{n \to \infty} S(n) = 1$

Hence for complete infinite binary tree it is exactly 1.

~~Hence for incomplete ...~~

3b) $\omega(x) = 3^{-d(x)}$. Let W be total weight of items of tree.

Let $\omega(x)$ be the weight of the node x.

The amortized cost of access (x, T) is

$3\lg\left(\frac{W}{\omega(x)}\right)+1$ if x is in T

access (x,T) is $3\lg\left(\frac{W}{\min\{\omega(x^-), \omega(x^+)\}}\right)+1$ if x is not in T.

$$\text{Amortized cost} = 3\lg\left(\frac{W}{\omega(x)}\right)+1$$

$$= 3\lg\left(\frac{W}{3^{-d(x)}}\right)+1$$

$$\leq 3\lg\left(\frac{1}{3^{-d(x)}}\right)+1$$

$$= 3\lg\left(3^{d(x)}\right)+1$$

So, Amortized cost is $O\left(\lg 3^{d(x)}+1\right)$

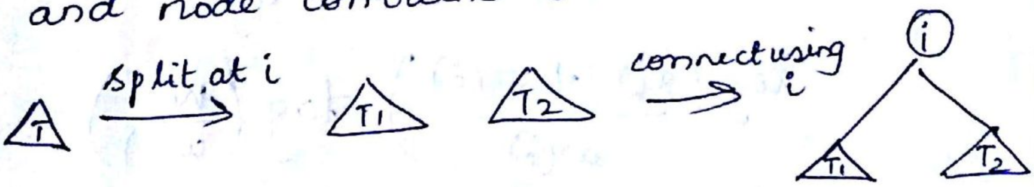
$$O\left(\lg 3^{d(x)}+1\right) = O\left(1+d(x)\right).$$

$\therefore$ Amortized cost of accessing ~~item~~ item x in a splay tree is $O(1+d(x))$.

Hence Proved.

4) Derive the bound for insert given in the table at the top of page 11 of the splay tree notes.

Insert takes a tree T and an item as an input and inserts a node containing i into T. To perform an insert, simply split(T,i) and then make a new tree whose left and right branches are the trees T_1 and T_2. This T_1 and T_2 is returned from split and node contains the item i.



Split (i, T) or Access (i, T) amortized cost is

$$3 \lg \left(\frac{W}{w(i)} \right) + O(1) \text{ if } i \text{ is in } T.$$

$$3 \log \left(\frac{W}{\min\{w(i^-), w(i^+)\}} \right) + O(1) \text{ if } i \text{ is}$$

not in T.

Then the new node weight $w(i)$ is added to the tree as root node and T_1 and T_2 are joined.

So the weight of the tree is added or increased by the weight of the node or item $w(i)$

(ie) If W is the weight of the resulting tree and $S(T)$ is the size function of the tree before insertion then W is

$$W = S(T) + w(i) \quad \text{— (1)}$$

Then the increase in potential is by adding a new root node i.

$$\therefore \log\left(\frac{S(T) + w(i)}{w(i)}\right) = \log\left(\frac{W}{w(i)}\right)$$

From (1) we can get.

$$S(T) = W - w(i) \quad \text{—— (2)}.$$

This can also be written as before insertion if the total weight is W then after insertion the total weight is $W + w(i)$.

T_1 and T_2 returned from split (i, T) should be connected using i as root node.

So the amortized cost of split and cost of inserting i should be ~~~~ added.

Amortized cost of split (i,T) can be written as

$$3\lg\left(\frac{S(T)}{w(i)}\right) + 1 \quad \text{if } i \text{ is in } T$$

(or)

$$3 \lg \left(\frac{S(T)}{\min\{w(i^-), w(i^+)\}} \right) + 1 \quad \text{when } i \text{ is not in } T$$

Substitute

$$S(T) = W - w(i).$$

then the

Amortized cost of insert

$$= 3 \lg \left(\frac{W - w(i)}{\min\{w(i^-), w(i^+)\}} \right) + O(1) + \log\left(\frac{W}{w(i)}\right)$$

= Actual cost + change in potential.