Assignment 5

Raz Reed

October 2, 2017

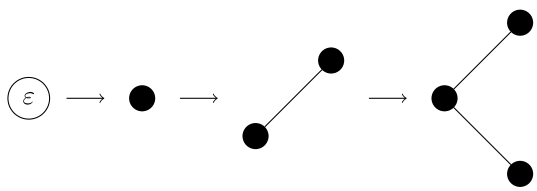
Problem 8.11

(a) An RBT is recursively defined:

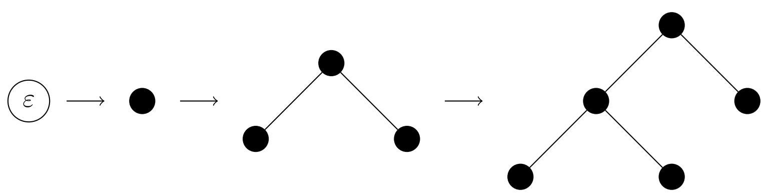
1. The empty tree ε is an RBT.
2. If T_1, T_2 are disjoint RBTs with roots r_1 and r_2, then linking r_1 and r_2 to a new root r gives a new RBT with root r.

Using this definition:

- The first diagram is an RBT.
  The empty tree ε is an RBT $\rightarrow$ a root node with no children is an RBT $\rightarrow$ a root node with one left child and no right child is an RBT $\rightarrow$ a root node with one left child with two children is an RBT

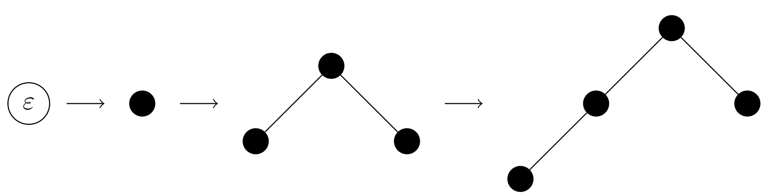


- The second diagram is an RBT.
  The empty tree ε is an RBT $\rightarrow$ a root node with no children is an RBT $\rightarrow$ a root node with two children is an RBT $\rightarrow$ a root node with two children whose left child has two children is an RBT



- The third diagram is not an RBT.
  Every RBT with $n \geq 1$ nodes has $n - 1$ links. The tree in the third diagram has 5 nodes and 5 links, therefore it is not an RBT.

- The fourth diagram is an RBT.
  The empty tree ε is an RBT $\rightarrow$ a root node with no children is an RBT $\rightarrow$ a root node with two children is an RBT $\rightarrow$ a root node with two children and one left child is an RBT



Only the first, second, and fourth diagrams are RBTs.

(b) An RBT is an RFBT if and only if each node in the tree either has no children or has two.

- The first diagram is an RBT, but has a node with one child. Therefore it is not an RFBT.
- In the second diagram, every node in the tree has either no children or has two, therefore it is an RFBT.
- The third diagram is not an RBT, therefore it is not an RFBT.
- The fourth diagram is an RBT, but has a node with one child. Therefore it is not an RFBT.

Only the second diagram is an RFBT.

Problem 9.2

(g)

$$
\begin{aligned}
\sum_{i=0}^{n}\left(\sum_{j=0}^{i} 2^i\right) &= \sum_{i=0}^{n}\left((i+1) * 2^i\right) \\
&= \sum_{i=0}^{n}\left(i2^i\right) + \sum_{i=0}^{n}\left(2^i\right) \\
&= \sum_{i=0}^{n}\left(i2^i\right) + 2^{n+1} - 1
\end{aligned}
$$

$$
\begin{aligned}
S(n) &= \sum_{i=0}^{n}\left(i2^i\right) \\
S(n) &= 1 \cdot 2^1 + 2 \cdot 2^2 + 3 \cdot 2^4 + ... + n2^n \\
2S(n) &= 2 \cdot 1 \cdot 2^1 + 2 \cdot 2 \cdot 2^2 + 2 \cdot 3 \cdot 2^3 + ... + 2(n-1)2^{n-1} + 2n2^n \\
&= 1 \cdot 2^2 + 2 \cdot 2^3 + 3 \cdot 2^4 + ... + (n-1)2^n + n2^{n+1} \\
S(n) &= 2S(n) - S(n) \\
&= \qquad\qquad 1 \cdot 2^2 + 2 \cdot 2^3 + 3 \cdot 2^4 + ... + (n-1)2^n + n2^{n+1} \; - \\
&\quad 1 \cdot 2^1 + 2 \cdot 2^2 + 3 \cdot 2^3 + 4 \cdot 2^4 + ... + n2^n \\
&= -2^1 - 2^2 - 2^3 - 2^4 - ... - 2^n + n2^{n+1} \\
&= n2^{n+1} - \sum_{i=1}^{n} 2^i \\
&= n2^{n+1} - \sum_{i=0}^{n} 2^i + 1 \\
&= n2^{n+1} - \left(2^{n+1} - 1\right) + 1 \\
&= n2^{n+1} - 2^{n+1} + 1 + 1 \\
&= 2^{n+1}(n-1) + 2
\end{aligned}
$$

$$
\begin{aligned}
\sum_{i=0}^{n}\left(\sum_{j=0}^{i} 2^i\right) &= \left(2^{n+1}(n-1) + 2\right) + \left(2^{n+1} - 1\right) \\
&= n2^{n+1} + 1
\end{aligned}
$$

Problem 6.19

(a) Unstacking 4 boxes:

Method 1:

$$\begin{array}{ll} 4 \rightarrow & \\ 2,2 \rightarrow & \$4 \\ 1,1 \quad 2 \rightarrow & \$1 \\ 1,1 \quad 1,1 & \$1 \\ \hline & \$6 \end{array}$$

Method 2:

$$\begin{array}{ll} 4 \rightarrow & \\ 1,3 \rightarrow & \$3 \\ 1 \quad 1,2 \rightarrow & \$2 \\ 1,1 \quad 1,1 & \$1 \\ \hline & \$6 \end{array}$$

Unstacking 5 boxes:

Method 1:

$$\begin{array}{ll} 5 \rightarrow & \\ 2,3 \rightarrow & \$6 \\ 1,1 \quad 3 \rightarrow & \$1 \\ 1,1 \quad 1,2 \rightarrow & \$2 \\ 1,1,1 \quad 1,1 & \$1 \\ \hline & \$10 \end{array}$$

Method 2:

$$\begin{array}{ll} 5 \rightarrow & \\ 1,4 \rightarrow & \$4 \\ 1 \quad 2,2 \rightarrow & \$4 \\ 1 \quad 1,1 \quad 2 \rightarrow & \$1 \\ 1,1,1 \quad 1,1 \rightarrow & \$1 \\ \hline & \$10 \end{array}$$

Method 3:

$$\begin{array}{ll} 5 \rightarrow & \\ 1,4 \rightarrow & \$4 \\ 1 \quad 1,3 \rightarrow & \$3 \\ 1,1 \quad 1,2 \rightarrow & \$2 \\ 1,1,1 \quad 1,1 \rightarrow & \$1 \\ \hline & \$10 \end{array}$$

(b) $P(n)$: the number of turns needed to split a stack of n boxes into n stacks of one box is $n-1$.

Conjecture: $P(n)$ is T for all $n \geq 1$. Base case: $P(1)$ is T – splitting 1 box into one stack of one box takes 0 turns.

Strong induction: assume $P(1), P(2), P(3), ..., P(n)$, and prove $P(n+1)$: the number of turns needed to split a stack of $n+1$ boxes into $n+1$ stacks of one box is n.

> Consider a stack of $n+1$ boxes. Splitting this produces stacks of k boxes and $n+1-k$ boxes and uses up one turn. Based on the induction hypothesis, the stack of k boxes will take $k-1$ turns to split into k stacks of one box, and the stack of $n+1-k$ boxes will take $n-k$ turns to split into $n+1-k$ stacks of one box. So, the number of turns it would take to split $n+1$ boxes into $n+1$ stacks of one box is:
>
> $$1 + (k-1) + (n-k) = n$$
>
> $P(n+1)$ is T. Therefore, $P(n)$ is T for all $n \geq 1$.

(c) $P(n)$: The maximum \$ you can earn by splitting n boxes is $\$\frac{1}{2}n(n-1)$.

Conjecture: $P(n)$ is T for all $n \geq 2$.

Base case: $P(2)$ is T – there is only one way to split a stack of 2 boxes into 2 stacks of one box, and that earns you $\$1 = \$\frac{1}{2} \cdot 2 \cdot 1$.

Strong induction: assume $P(1), P(2), P(3), ..., P(n)$, and prove $P(n+1)$: the maximum \$ you can earn by splitting n boxes is $\frac{1}{2}n(n+1)$.

> Consider a stack of $n+1$ boxes. Splitting this produces stacks of k boxes and $n+1-k$ boxes. Based on the induction hypothesis, the stack of k boxes makes $\$\frac{1}{2}k(k-1)$ and the stack of $n+1-k$ makes $\$\frac{1}{2}(n+1-k)(n-k)$. The initial split of the $n+1$ boxes makes $\$k(n+1-k)$. So, the total amount of money that can be earned by splitting $n+1$ boxes into $n+1$ stacks of one box is:
>
> $$\frac{1}{2}(n+1-k)(n-k) + \frac{1}{2}k(k-1) + k(n+1-k) = \frac{1}{2}n(n+1)$$
>
> $P(n+1)$ is T. Therefore, $P(n)$ is T for all $n \geq 2$.

Problem 8.9

(c) Claim: In any RFBT, $n = 2F + 1$ where n is the number of nodes in the tree and F is the number of full nodes.

An RFBT is recursively defined:

1. $\bullet$ (node) $\in$ RFBT
2. $l, r \in$ RFBT $\rightarrow$ a tree with root $\bullet$ with left child l and right child $r \in$ RFBT

Base case: $|\bullet| = 1 = 2 \cdot 0 + 1$ ($|\bullet| = \#$ of nodes in RFBT with one node)

Induction: assume $n = 2F + 1$, prove that the production rules preserve this property.

$|l| = 2F + 1$ ($\#$ of nodes in l)
$|r| = 2F + 1$ ($\#$ of nodes in r)
$|\text{new RFBT}| = |l| + |r| + |\bullet| = 2F + 1 + 2F + 1 + 1 = 2(2F + 1) + 1$

The number of nodes in the new tree is $2(2F + 1) + 1$. However, the $2F + 1$ nodes are now all full nodes, so we represent them with F and rewrite the equation:

$|\text{new RBFT}| = 2F + 1$

Thus the property $n = 2F + 1$ is preserved. Therefore the claim is true.

Problem 9.15

Solution:

$$\sqrt{n},\ F_{H_n}^2,\ H_{F_n},\ ln^3(n),\ n \cdot ln(n),\ n^2,\ n^3,\ n^{100},\ n^{ln(n)},\ (1.5)^n,\ 2^n,\ n^2 \cdot 2^n,\ ln(n)^n,\ n!,\ n^{n^2},\ n^{2^n}$$