ICCS200: Assignment 5

Vikrom Narula
vikrom.nar@gmail.com
07/06/2019
People in 1408 & 1409

Exercise 1:

Proposition A :

We know that node that will fit the rule of having either two or zero child the amount of nodes have to be odd. I will prove the theorem by using strong induction.

Predicate:

$$P(i) := \frac{i+1}{2};\ for\ all\ odd\ i$$

Base Case:

$$P(1) := 1 == \frac{1+1}{2}$$

$$P(1) := 1 == 1\ \checkmark$$

Inductive Step:

Let assume that $1 \le k \le i$ is true. We also know that the tree is created by the combination of two other odd node tree. We want to prove it works for all odd number.

$$Tree\ Leaves = \frac{p+1}{2} + \frac{q+1}{2}$$

$$Tree\ Leaves = \frac{p+q+1+1}{2}$$

$$Tree\ Leaves = \frac{p+q+2}{2}$$

We know p and q are just some number which means

$$Tree\ Leaves = \frac{k+2}{2}$$

Which this is equal to P(k+1)

$$P(k+1) := \frac{(k+1)+1}{2} = \frac{k+2}{2}$$

Hence we know it works for the next number and it also work in it's tree recursion.

Proposition B:

We know from the Invariant of Binary Heap tree are

- Parent has a greater or equal to value to their children.
- Tree is full at all levels and it's left justified.

With the second invariant we know the tree will be full and reach the last level. We also know that a parent will have at most two children. By this logic we can assume that the lower level will have at most twice the number of node to the current level. Hence we can say that we assume n is the number of nodes and the depth increased by twice every level we also know the tree will reach the last level while the other level before are full so we can say $2^k = n$ while k is the depth of the level and when we move the equation to see how deep it can get we have $k = log_2 n$ hence we know at most the depth of a binary heap tree is $O(log\ n)$.