CSCE 4430
Inductive Proof for Programming Assignment 4
11/13/2016
Group 1

1) **The number of combinations of trees for a given size 'n' are given by Catalan numbers**

Catalan number is given by $C_n = \frac{2(2n-1)}{(n+1)} C_{n-1}$

For n= 0,1,2,3… they form sequence 1,1,2,5,

C_n is the number of rooted binary tree with n internal nodes and n+1 leaves

Proof:

We will prove that for all n >=1

$C_n = \frac{2(2n-1)}{(n+1)} C_{n-1}$

Base case

When n=1 ,

$C_n = \frac{2(2n-1)}{(n+1)} = C_1 = \frac{2(1)!}{(1+1)!1!} C_{1-1}$

$C_1 = \frac{2}{2} C_0$

$C_0 = 1$ so $C_1=1$

Inductive step:

Assuming that the formula is true for k such that

$C_k = \frac{2(2k-1)}{k+1} C_{k-1}$ is true

For when k+1

We have,

$C_{k+1} = \frac{2(2(k+1)-1)}{(k+1)+1} C_{(k+1)-1}$

$C_{k+1} = \frac{2(2k+2-1)}{k+2} C_k$

$C_{k+1} = \frac{2(2k+1)}{k+2} C_k$

This is a formula for when n=k+1 therefore

$C_n = \frac{2(2n-1)}{(n+1)} C_{n-1}$ holds true

CSCE 4430
Inductive Proof for Programming Assignment 4
11/13/2016
Group 1

---

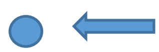
2) **For binary tree with size n, number of leaves is always one more than the number of internal nodes and number of internal node is always one less than the number of leaves. Internal node will always have 2 children and leaves are the nodes that do not have children.**

Number of Internal nodes in tree of size n

1) Statement:

Number of leaf node in a binary tree, where n is the total number of nodes is always one more than the number of internal node

2) Base Case:
If n = 0, then number of leaf node = 1



Number of leaf node = 1 (True)

There is no internal node when n = 0 (True)

3) Inductive Hypothesis:

P(n) = P(n+1) for $n \geq 1$

4) Inductive Steps:

Let B be the binary tree, with internal node (n+1)

Let intNode(B) be the number of internal node and Leaf(B) be the leaf node.

intNode(B) = n +1

In a binary tree if we select any internal node that has two children leaf node, there is atleast one internal node. If we replace this node with a leaf node, the resulting tree B' has one less one less leaf node than tree B.

intNode(B') = intNode(B) -1 = n+1-1 = n

Leaf(B') = Leaf(B) -1 = n-1

Leaf(B') = intNode(B') +1

Therefore, Leaf(B)

CSCE 4430
Inductive Proof for Programming Assignment 4
11/13/2016
Group 1

Select an internal node 'x' which is parent of two leaf nodes. Since n > 1, there is atleast one internal node. Replace the subtree rooted at 'x' by a leaf node. The resulting tree T 0 has one less internal node and one less leaf node than T , i.e., Int(T 0) = Int(T) − 1 = n + 1 − 1 = n Leaf(T 0) = Leaf(T) − 1

Since T 0 has n nodes, we can apply the inductive hypothesis and obtain Leaf(T 0) = Int(T 0) + 1 = n + 1. Substituting values from (1), we immediately get Leaf(T) = Int(T) + 1 = n + 1 + 1.

2nd Proof:

To Prove: The number of internal nodes in binarytree n is such that the number of leaves is always one more than the number of internal node

Let p(n) be the statement that states that the number of leaves is always one more than the number of internal node in a binary tree

Base Case

When n = 0, that is internal node = 0, number of leaves = 1 by inspection. Number of leaves is one more than number of internal node

P(0) is true

Inductive Case

If a binary tree has node n >=1, then we kow that it has two subtree. Let the number of internal nodes be given by i (left tree) and j (right tree) such that i+j +1 = n, where n is all of the internal node of a given tree distributed between the left subtree, right subtree and the root.

From the hypothesis, p(i) and p(j) tells us that the number of leaves in left and right subtree is i+1 and j+1 respectively. Therefore total number of tree = i+1 +j+1 = i+j+2

By the definition of number of internal node we know that, number of internal node of given binary tree = i+j+1. The total number leaves = (i+j+1) +1 which is exactly one more than the number of internal node. Thus, P(n) holds and by induction we can say that the original hypothesis is proven for any binary tree of a given structure.