Homework15
Due: Tuesday October 30, 2006

1

Given the following definition of a binary tree:

$$\text{data NTree a} = \text{NLeaf a} \mid \text{NNode a (Ntree a) (Ntree a)}$$

We can write definitions for size and height that are very similar to our trees which do not contain information in the internal nodes.

$$\begin{aligned} &size :: NTree\ a \ \rightarrow \ Int \\ &\quad \text{size NLeaf x} = 1 \\ &\quad \text{size NNode } (x, t_1, t_2) \ = \ 1 + size(t_1) + size(t_2) \end{aligned}$$

We note that now we are including the internal nodes in the size of the tree.

And the height function remains the same as we saw before:

$$\begin{aligned} &height :: NTree\ a \ \rightarrow \ Int \\ &\quad \text{height NLeaf x} = 0 \\ &\quad \text{height NNode } (x, t_1, t_2) \ = \ 1 + max(height(t_1), height(t_2)) \end{aligned}$$

We saw that there was a relationship between the size and height of our Btrees as defined in Bird on page 179. There is a similar relationship between the size and height of our NTrees as defined here.

$$\forall t : NTree.size(t) \leq 2^{height(t)+1} - 1$$

Prove by structural induction on t that this relationship holds.

2 Complete trees

A **CompleteBinary** tree is defined to be one where

1. All the leaves of the tree have the same depth.
2. every node has exactly two subtress.

One way to think about the first item (the depth requirement) is that the difference between the max depth and min depth should be 0.

Some examples of complete binary trees over the integers (NTree Int)
NLeaf(5)
NNode 4 (NNode 2 (NLeaf 3) (NLeaf 6)) (NNode 8 (NLeaf 17) (NLeaf 23))

NNode 5 (NLeaf 7) (NLeaf 4)

Convince yourself that these are complete binary trees (by drawing pictures!).

In haskell write a function that returns True if a tree is complete and false otherwise.

$$complete :: NTree\ a \rightarrow Boolean$$

You will need to write a depth function for NTrees, as in the first section of Bird but his is defined on Btrees, to determine the depth of all leaves. And a predicate that determines if a tree has two subtrees.