The sum of heights in a perfect binary tree of \( n \) nodes is \( O(n) \)

**Observation:** A perfect binary tree of height \( h \) has:

- \( n = n(h) = 2^{h+1} - 1 \) nodes,  
- \( \sum_{h} = n - h - 1 \).

We explain the observation with a small inductive proof. The induction is on \( h \), the height of the tree. We first show that the observation holds for \( h = 0 \) (the inductive basis). We then assume that the observation holds for \( h = k \), and show that this implies that it holds for \( h = k + 1 \) (the inductive step).

**Basis:**
Let \( h = 0 \), this is the binary tree consisting of a single node; its sum-of-heights is zero (we count edges). The observation holds, we have:

- \( n = 2^{h+1} - 1 = 2^0 + 1 = 2 - 1 = 1 \), (one node)
- \( \Sigma_h = n - h - 1 = 1 - 0 - 1 = 0 \). (sum-of-heights is zero)

**Step:**
We now assume that the observation holds for \( h = k \), and show that this implies that it holds for \( h = k + 1 \). The assumption gives the number of nodes and sum-of-heights for a tree of height \( k \):

- \( n = 2^{k+1} - 1 \),
- \( \Sigma_h = n - h - 1 = 2^{k+1} - 1 - k - 1 \).

We then move one step up, to a tree of height \( k + 1 \), by adding a new row of \( 2^{k+1} \) leafs. A tree of height \( k + 1 \) has the same number of non-leafs as the number of nodes in a tree of height \( k \). Each such non-leaf is now one level higher up the tree than in the tree of height \( k \). We get:

- \( n' = (2^{k+1} - 1) + (2^{k+1}) = 2^{k+2} - 1 \), (adding \( 2^{k+1} \) leafs)
- \( \Sigma_h = (2^{k+1} - 1 - k - 1) + (2^{k+1} - 1) = 2^{k+2} - 1 - k - 1 + 2^{k+1} - 1 = (2^{k+2} - 1) - (k + 1) - 1 \), (adding 1 for all non-leafs)

and the observation holds for \( h = k + 1 \).

We know, of course, that the height of a perfect binary tree of \( n \) nodes is \( O(\log n) \)\(^\dagger\), so that the sum-of-heights is \( n - O(\log n) - 1 = O(n) \), which is what we really wanted to prove.

\(^\dagger\) To be exact, the height of complete or perfect binary trees of \( n \) nodes is \( \lceil \log_2 (n + 1) \rceil - 1 \), if we count edges.
A leftist heap of $n$ nodes has a right path of at most $\lfloor \log (n + 1) \rfloor$ nodes

**Observation:** If the right path of a leftist heap has $r$ nodes, then the heap has $n = n(r) \geq 2^r - 1$ nodes.

We again apply a small inductive proof, this time on $r$, the number of nodes in the right path. We first show that the observation holds for $r = 1$; we then assume that it holds for $r = k$, and show that this implies that it holds for $r = k + 1$.

**Basis:**
Let $r = 1$. The observation holds, the tree has at least $2^1 - 1 = 1$ node.

**Step:**
We now assume that the observation holds for $r = k$, and show that this implies that it holds for $r = k + 1$.

A leftist heap with a right path of $k + 1$ nodes consists of a root with one left and one right subtree, both subtrees must by definition be leftist. The right subtree must have a right path of $k$ nodes for our tree to have a right path of $k + 1$ nodes. The left subtree must have at least $k$ nodes on its right path; otherwise the root of the left subtree would have a null path length shorter than the null path length of the right subtree. Therefore, by the assumption, the number of nodes in both subtrees is at least $2^k - 1$. This, plus the root, gives us $n \geq (2^k - 1) + (2^k - 1) + 1 = 2^{k + 1} - 1$ nodes, as wanted.

If a heap with right path of $r$ nodes has $n \geq 2^r - 1$ nodes in total, it follows that a heap of $n$ nodes has a right path of at most $\lfloor \log (n + 1) \rfloor$ nodes, which is what we really wanted to prove.