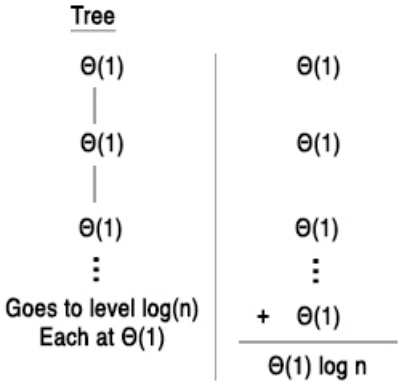
Intro to Algorithms, COMP-160, Homework #1

Benjamin Tanen, 02/02/2016

6. $H(n) = H(\frac{n}{2}) + \Theta(1)$. **Give matching upper and lower bounds for $H(n)$ with a recursion tree and then by substitution.**

Proof by recursion tree: We can see that each level of our tree simply branches down but doesn't split up at all. Each level of this tree has constant behavior $\Theta(1)$. Since we know (because each recursion is splitting the list in half) there are $\log n$ levels, making our total work $\Theta(1) \cdot \log n$.

Thus, $H(n) = \Theta(\log n)$



Proof by substitution: Assume $H(n) = c \cdot \log n$

For $k < n$, $H(k) = c \cdot \log n$

$c_1 \cdot \log n \leq c \cdot \log(\frac{n}{2}) + \Theta(1) \leq c_2 \cdot \log n$

$c_1 \cdot \log n \leq c \cdot \log(n) - c \cdot \log(2) + d_1 \leq c_2 \cdot \log n$

$c_1 \cdot \log n \leq c \cdot \log(n) + d_2 \leq c_2 \cdot \log n$

Since we know $d_2 = \Theta(1) - c \cdot \log(2) \geq 0$, we know $c_1 \leq c$, proving $H(n) = \Omega(\log n)$

We can also assume that $c_2 + \log n \leq c_2 \cdot \log n$

This leads us to: $\log n + d_2 \leq \log n + c_2 \leq c_2 \cdot \log n$

Thus, $c_2 \geq d_2 = d_1 - c \cdot \log(2)$, providing a valid upper bound, proving $H(n) = O(\log n)$

Since we've shown $H(n) = \Omega(\log n)$ and $H(n) = O(\log n)$, we have proven that:

$$H(n) = \Theta(\log n)$$