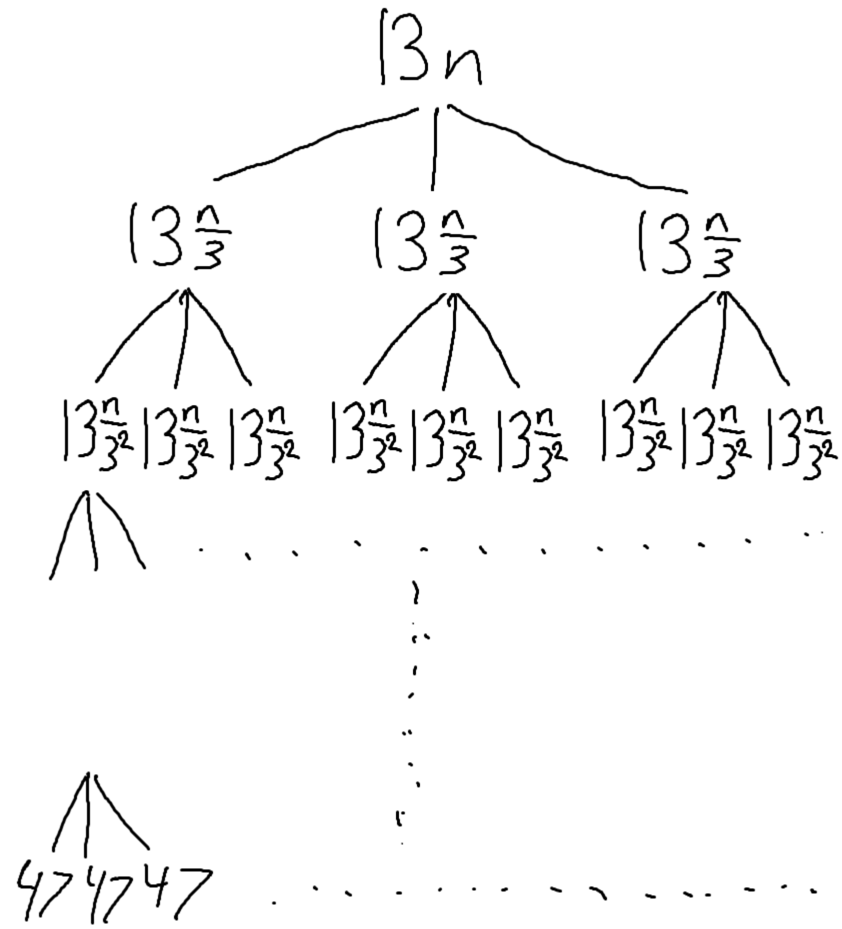
Trees Tutorial Solutions

13.1a Recursion trees

Assume n is a power of 3 so that the input will always be an integer. Then we get the following tree:



The tree is described by the following table:

level	"problem size"	# nodes	work per node	total for level
0	n	1	$13n$	$13n$
1	$\frac{n}{3}$	3	$13\frac{n}{3}$	$13n$
2	$\frac{n}{3}$	3^2	$13\frac{n}{3^2}$	$13n$
3	$\frac{n}{3^2}$	3^3	$13\frac{n}{3^3}$	$13n$
...				
k	$\frac{n}{3^k}$	3^k	$13\frac{n}{3^k}$	$13n$
...				
h	$\frac{n}{3^h} = 1$	3^h	$T(1) = 47$	$47 * 3^h$

(Notice that the final row (the leaf level) follows the same pattern for problem size and number of nodes as the rows above it, but that we also know the problem size must be 1 *since that's the function's base case - this is why I've written both* $\frac{n}{3^h}$ *and* 1 *in that cell, and this is how we are able to solve for* h*. Note that the work per node and hence total for level does* not

follow the pattern of the levels above it; this is why our later summation only sums through $h-1$ *and then we have to add in the work in the leaves separately.)*

We have $\frac{n}{3^h} = 1$, i.e. $h = \log_3 n$, so there are $3^{\log_3 n} = n$ leaves. Thus the total work at the leaves is $n \cdot T(1) = 47n$.

From the table, the total work for all non-leaf levels is
$\sum_{k=0}^{(\log_3 n)-1} 13n = 13n \log_3 n$.

Putting it all together, our final closed form is $47n + 13n \log_3 n$.

13.3b Non-grammar tree induction

Let T be a parity tree; we will prove T has the parity property by induction on its height h.

Base: For height 0, T is just a solitary root. That root is also a leaf so it is orange by rule 1 of parity trees. Thus there is an odd number of leaves (1) and the root is orange, so T has the parity property.

(Commentary: You might think you need two base cases here: height 0 for an orange-root case and height 1 for blue-root. However, while including an extra base case doesn't invalidate the proof, it's not actually necessary here - to see that, try following through the logic of the induction step below using the concrete height 1 tree plugged in for T *everywhere.)*

Induction: Suppose that all parity trees with height less than h have the parity property. Then for parity tree T with height h, consider its left and right subtrees T_ℓ and T_r, and let n_l and n_r be the number of leaves in the respective subtrees. Notice that T_ℓ and T_r are also parity trees, so since they have height smaller than h, by the IH we know they both have the parity property. *(You can* not *say that they have height* $h-1$ *- one of them definitely does, but the other could be arbitrarily shorter. This is why it is important that we are using a* strong *IH.)* Now we get four cases:

Case 1: n_ℓ and n_r are both even. Then by the parity property, T_ℓ and T_r both have blue roots. Then by rule 2 of parity trees, T also has a blue root. And we know the total number of leaves is $n_\ell + n_r$ which is even (because its the sum of two evens), so T has the parity property.

Case 2: n_ℓ and n_r are both odd. Then by the parity property, T_ℓ and T_r both have orange roots. Then by rule 2 of parity trees, T has a blue root. And we know the total number of leaves is $n_\ell + n_r$ which is even (because its the sum of two odds), so T has the parity property.

Case 3: n_ℓ is even and n_r is odd. Then by the parity property, T_ℓ has a blue root and T_r has an orange root. Then by rule 2 of parity trees, T has an orange root. And we know the total number of leaves is $n_\ell + n_r$ which is odd (because its the sum of an even and an odd), so T has the parity property.

Case 4: n_ℓ is odd and n_r is even. See case 3 with the roles of T_ℓ and T_r reversed.

Thus T has the parity property in every case, QED.

13.2a Grammar Trees

Proof by induction on the tree height.

Base: Notice that trees from this grammar always have height at least 1. The only ways to produce a tree of height 1 are the third and fourth rules; in each case the tree ends up with one node labeled a and at most one labeled b.

Induction: Assume that any tree of height less than some $k > 1$ has at least as many a nodes as bs. Now consider a generated tree with height k. The root must be labelled S and the grammar rules that can produce trees of height greater than 1 give us two cases for what the children are:

Case 1: The root's children are labeled a, S, b, and S. Let T_1 and T_2 be the subtrees rooted at the nodes labeled S, and let a_1, a_2, b_1, b_2 be how many a nodes and b nodes are in each subtree. Since T_1 and T_2 have height less than k, the IH applies to them, so $a_1 \geq b_1$ and $a_2 \geq b_2$. Putting these two inequalities together and adding one, we establish that $a_1 + a_2 + 1 \geq b_1 + b_2 + 1$. And $a_1 + a_2 + 1$ is just the total number of a nodes in the tree while $b_1 + b_2 + 1$ is the total number of b nodes, so we have shown that there are at least as many as overall as bs.

Case 2: The root's children are labeled S, a, S. The logic here is exactly like case 1 except with one fewer b node, so there are definitely at least as many as as bs.

Thus in every case there are at least as many as as bs, induction complete.

13.4 Challenge Example

a) Proof by induction on the order k of the tree.

Base: A binomial tree of order 0 is defined to have just $1 = 2^0$ node.

Induction: Let k be positive and suppose that for every $i < k$, a binomial tree of order i has 2^i nodes. A binomial tree of order k is built from 2 binomial trees of order $k-1$, which by the IH have 2^{k-1} nodes each. Thus the whole tree has $2^{k-1} + 2^{k-1} = 2^k$ nodes, QED.

b) Proof by induction on the order k of the tree.

Base: A binomial tree of order 0 is defined to have just 1 node at level 0. $\binom{0}{0} = 1$.

Induction: Fix $k \geq 0$ and suppose that for every binomial tree with order $r \leq k$, at each level i the tree has $\binom{r}{i}$ nodes. Now consider a binomial tree of order $k+1$. By the definition of a binomial tree, it consists of two binomial trees T_1 and T_2 each of order k, where each node in T_2 has been 'shifted down' one level since T_2's root was connected as the rightmost child of the root of T_1. Note that the IH applies to both T_1 and T_2. Now fix a level i. There are three cases:

Case 1: $i = 0$. In this case T_1 contributes 1 node to the level and T_2 contributes 0, so in total there is $1 = \binom{k+1}{0}$. *(Commentary: note that for this case we don't actually need the IH. This is fine. But if you ever find that your* entire *inductive step doesn't use the IH, that would be a major red flag that you're almost certainly doing something wrong.)*

Case 2: $i = k+1$. In this case T_1 has no nodes at level i. By the IH, T_2 has $\binom{k}{k} = 1$, so in total there is $1 = \binom{k+1}{k+1}$.

Case 3: $0 < i < k+1$. In this case, by the IH we get $\binom{k}{i}$ nodes from T_1 and $\binom{k}{i-1}$ nodes from T_2. Thus the total number of nodes at level i is $\binom{k}{i} + \binom{k}{i-1}$. We simplify that as follows: $\binom{k}{i} + \binom{k}{i-1} = \frac{k!}{i!(k-i)!} + \frac{k!}{(i-1)!(k-i+1)!} = \frac{k!(k-i+1)+k!(i)}{i!(k-i+1)!} = \frac{k!(k+1)}{i!(k-i+1)!} = \frac{(k+1)!}{i!(k-i+1)!} = \binom{k+1}{i}$. So there are $\binom{k+1}{i}$ nodes at level i, QED.