CS 624: Analysis of Algorithms

Assignment 1

Due: Monday, January 29, 2018

1. Let $\{f_n : n = 0, 1, \dots\}$ be the Fibonacci sequence (where by convention $f_0 = 0$ and $f_1 = 1$).

   (a) Prove that

   $$\sum_{n=1}^{\infty} \frac{n f_n}{2^{n-1}} = 20$$

   Do this by using a generating function as shown in the last section of the Lecture 2 notes, and differentiating.

   (b) Show why (in the same way as you proved the first part of this problem) you might think that

   $$\sum_{n=1}^{\infty} n f_n = 2$$

   Then show why this could not possibly be true[1].

2. Prove that for positive integers a, b, and n,

   $$\sum_{k=0}^{n} \binom{a}{k} \binom{b}{n-k} = \binom{a+b}{n}$$

   Use the following facts in your proof:

   (a)

   $$(1+x)^a (1+x)^b = (1+x)^{a+b}$$

   (b) You may find it convenient at some point to remember that if m and j are integers (and $m \geq 0$), then

   $$\binom{m}{j} = 0 \qquad \text{if } j < 0 \text{ or } j > m$$

[1] And I don't mean something like, "It seems implausible to me." or "I don't believe it.". It has to be something that you could tell anyone else, anywhere in the world, and they would be convinced without a doubt. It doesn't have to be a long explanation. In fact, it could be very short. But it has to be absolutely convincing. That's what we mean by a proof.

3. Decide whether each of the following statements is true or false, and prove that your conclusion is correct.

   - $n^2 = O(2^n)$
   - $2^{n+1} = O(2^n)$
   - $2^{2n} = O(2^n)$
   - $f(n) = O\big(g(n)\big)$ implies $2^{f(n)} = O\big(2^{g(n)}\big)$

4. Prove that $\log_a x = O(\log_b x)$ for any $a > 1$ and $b > 1$.

5. Suppose $p(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$ is a polynomial.

   (a) Suppose you know that $a_n > 0$. Prove carefully that $p(x) = O(x^n)$.

   (b) Suppose you know that $p(x) = O(x^d)$ for some number d such that $d < n$. Prove that $a_n = 0$.

6. Prove that if $f = O(g)$ and $g = O(h)$ then $f = O(h)$.

7. Suppose that r and s are two positive numbers, and that $r \neq s$. In fact, for the purpose of this problem we can assume without loss of generality that $0 < r < s$.

   Consider the statement $s^n = O(r^n)$.

   Is this statement true? If so, prove it. If not, prove that it is false.

8. Problem 4-1 (a, b, c, f, g) (page 107).

9. Exercise 5.3 in the Lecture 1 handout (page 16 in the handout). Actually, just do the first sentence of the problem. You don't have to do the second one.

10. Exercise 4.1 in the Lecture 2 handout (page 13 in the handout).

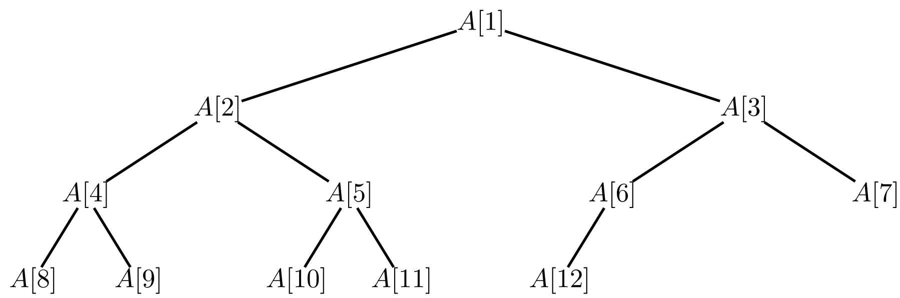
11. Suppose we have n points in the plane, each one specified by its two coordinates, and suppose we want to write a program that will find a pair of points that are closest together. Here is an algorithm that does this; it is what is called a *brute-force search*: Make a list of all the pairs of points. Walk down this list, and for each pair, figure out the distance between the two points. Keep track of the smallest distance you have found so far, and the two points that gave you that distance. When you have finished, you will have found a pair of points with the smallest distance between them.

    (a) Could there be more than one such pair? (If so, you have to give an example. If not, you have to prove that would be impossible.)

    (b) Show that the running time of this algorithm is $\Theta(n^2)$ (where n is the number of points)[2].

12. Suppose we have a 1-based array $A[1..n]$ which holds the elements of a binary tree, like this:

[2] It may seem that you couldn't do better; but actually there *are* more efficient algorithms.



Each row is filled in from the left, and the last row may not be complete, but it too is filled in from the left.

The point of this exercise is to prove that in this tree (well actually, in *any* tree of this shape), the children of $A[n]$ (if they exist) are $A[2n]$ and $A[2n+1]$. I want you to do this as follows:

Let us say that the root node of the tree is at "level 0". Its two children are at "level 1". And so on. You should be able to see that in any row (say at level k) which is completely filled in (in other words, any row except possibly the last one), the first element is $A[2^k]$ and the last one is $A[2^{k+1}-1]$. That's one of the things we will prove immediately below, but you should make sure that it makes sense to you first. **Don't go on until it does make sense, and if it doesn't, then send me email right away.**

I want you to prove the following sequence of statements:

- If level 0 has a node[3] then
  (a) It has exactly one node, and that node is $A[1]$.
  (b) the children of that node (assuming those children exist) are $A[2]$ and $A[3]$.

  Actually, you don't have to prove this. It's obvious. But I wanted to state it for completeness. You can just assume it is true, and you can use it if you need to.
- If level 1 is completely filled out, then
  (a) The first node at that level is $A[2]$ and the last node is $A[3]$.
  (b) the children of any node $A[n]$ at that level (assuming those children exist) are $A[2n]$ and $A[2n+1]$.
- If level 2 is completely filled out, then
  (a) The first node at that level is $A[4]$ and the last node is $A[7]$.
  (b) the children of any node $A[n]$ at that level (assuming those children exist) are $A[2n]$ and $A[2n+1]$.
- If level 3 is completely filled out, then
  (a) The first node at that level is $A[8]$ and the last node is $A[15]$.
  (b) the children of any node $A[n]$ at that level (assuming those children exist) are $A[2n]$ and $A[2n+1]$.

  ...

Note that there are infinitely many statements here. You have to prove all of them. You can't just prove a few of them and then say "and so on" or something like that.

[3] This would be the root node. If there is no root node, there aren't any nodes at all, right?