CS 330: Introduction to Algorithms Spring 2019

CS 131 (Combinatoric Structures) Revision

Question 1. Proof by induction
Consider a network of n computers. We want to connect them with wires which allow communication between two computers in both directions.

(a) Suppose that we want to construct a completely connected network. That is, we would like to connect each of every pairs. Prove that the number of wires needed is $\frac{n(n-1)}{2}$ using mathematical induction.

> **Answer**: Our basis case is when $n = 2$, and the given statement is true for the case: $P(2) = 2(2-1)/2 = 1$. Now suppose that the statement holds for a natural number k. If one introduces a new $(k+1)^{\text{th}}$ computing node, it must be connected to k other nodes, creating k new connections. Then
>
> $$P(k+1) = P(k) + k = \frac{k(k-1)}{2} + k = \frac{k^2+k}{2} = \frac{(k+1)k}{2};$$
>
> and hence $P(k+1)$ holds if $P(k)$ holds. Thus, the statement is true for all $k \geq 2$.

(b) Now we want to construct a network in such a way that any two computers are connected by a **unique** route passing each computer at most once. Prove that the number of wires needed is $n-1$ using mathematical induction.

> **Answer**: For $k = 2$, which is our basis case, $P(k = 2) = 2 - 1 = 1$, and so the statement is true. Now suppose that $P(k) = k - 1$ for an arbitrary natural number k. Now add a new computing node to the system.
>
> Observe that the newly introduced node has exactly one edge: If the new node u is connected more than one pre-existing node, paths are no longer unique.
>
> > (*Proof by Contradiction*) Let u, the newly introduced node is connected to v_1 and v_2. Before its introduction, there exists a unique path connecting v_1 and v_2, which does not includes u. Introduction of u creates a new path connecting v_1 and v_2; that is, (v_1, u, v_2). Thus, u cannot be connected to more than a single node.
>
> Hence, $P(k+1) = P(k)+1 = (k-1)+1 = (k+1)-1$; $P(k+1)$ holds, if $P(k)$ holds. Thus, the statement holds for all $k \geq 2$.

Question 2. Big-Oh notation

a) Use the definiton of big-theta to prove that: $\frac{1}{2}n^2 - 3n = \Theta(n^2)$

b) Given that: $1^3 + 2^3 + \ldots n^3 = (1 + 2 + \ldots + n)^2$, prove that $\sum_{k=1}^{n} k^3 = \Theta(n^4)$

b) Suppose that $f(x)$ and $g(x)$ are two functions such that for some other function $h(x)$, we have $f(x) = O(h(x))$ and $g(x) = O(h(x))$. Then $f(x) + g(x) = O(h(x))$.

c) Suppose that f(x) and g(x) are two functions (taking nonnegative values) such that $g(x) = O(f(x))$. Then $f(x) + g(x) = \Theta(f(x))$. In other words, $f(x)$ is an asymptotically tight bound for the combined function f(x)+g(x).

Solution

a) We must determine positive constants c_1, c_2, n_0, such that:

$$c_1 n^2 \leq \frac{1}{2} n^2 - 3n \leq c_2 n^2$$

for all $n \geq n_0$. Dividing by n^2, yields:

$$c_1 \leq \frac{1}{2} - \frac{3}{n} \leq c_2$$

Right-hand inequality holds for any value of $n \geq 1$, by choosing $c_2 = \frac{1}{2}$. Likewise, left-hand inequality is true for any $n \geq 7$, by choosing $c_1 = \frac{1}{14}$.
Therefore, choosing $c_1 = \frac{1}{14}, c_2 = \frac{1}{2}, n_0 = 7$ we prove that $\frac{1}{2} n^2 - 3n = \Theta(n^2)$

b) From the given equation, we can observe that:

$$\sum_{k=1}^{n} k^3 = \left[\sum_{k=1}^{n} k\right]^2 = \left(\frac{n(n+1)}{2}\right)^2 = \frac{n^4}{4} + \frac{n^3}{2} + \frac{n^2}{4}$$

which we can prove is $\Theta(n^4)$, by using either the limit definition, or the quantification definition

c) We are given that for some constants c and n_0, we have $f(n) \leq c \cdot h(n)$ for all $n \geq n_0$. Also, for some (potentially different) constants c and n_0', we have $g(n) \leq c' h(n)$ for all $n \geq n_0'$. So consider any number n that is at least as large as both n_0 and n_0' . We have $f(n) + g(n) \leq ch(n) + c'h(n)$. Thus $f(n) + g(n) \leq (c + c')h(n)$ for all $n \geq max(n_0, n_0')$, which is exactly what is required for showing that $f(x) + g(x) = O(h(x))$.

d) We know that $g(x) = O(f(x))$. Thus, $\exists c, n_0$, such that $g(x) \leq c \cdot f(x), \forall n \geq n_0$ for some $c > 1$. Also, trivially, we can say that for the same n_0, $f(x) \leq cf(x), \forall n \geq n_0$. Adding these two we get $f(x) + g(x) \leq 2 \cdot cf(x)$. Also, $f(x) \leq f(x) + g(x), \forall n \geq n_0$. Setting $c_1 = \frac{1}{2c_1}$ and $c_2 = 1$ we conclude our proof.

Question 3. Let S_n denote the number of n-bit strings that do not contain the pattern 00.

(a) Provide a base case and initial conditions for S_1 and S_2.

(b) Write the recursive case for $S_i, i \geq 3$.

Solution

(a) For n=1, we have the following strings: $0, 1$. Thus $S_1 = 2$.
For n=2, we have the following strings $00, 01, 10, 11$. Thus $S_2 = 3$.

(b) We call safe strings, those that do not contain the pattern 00. We can obtain a n-bit string by i) either getting the safe (n-1)-bit strings and appending a 1 at the end, or ii) getting all safe (n-2)-bit strings and append 10 at the end. Notice, that these two sets are disjoint and contain all safe n-bit strings. Thus, $S_i = S_{i-1} + S_{i-2}$.

Question 4. There are 10 copies of one book and one copy each of 10 other books. In how many ways can we select distinct backpacks of 10 books?

Solution
We have 10 copies of one book, let's call it A, and one copy of each of the other 10 books.

We have the following cases:
Case 1: We do not pick any of the A books. So, we have to choose 10 books from the rest 10 books. This is $\binom{10}{10}$.
Case 2: We pick one of the A books. So, we we have to choose 9 books from the rest 10 different books. This is $\binom{10}{9}$.
Case 3: We pick two of the A books. So, we have to choose 8 from the rest 10 books. This is $\binom{10}{8}$.
...
Case 10: We pick all ten A books. Now, we can choose zero from the rest books. This is $\binom{10}{0}$ (actually just one way to do this).

To get the total number of ways to select distinct backpacks we have to sum the above cases. Thus, $\binom{10}{10} + \binom{10}{9} + \binom{10}{8} + ... + \binom{10}{0}$
From the identity$\sum_{i=0}^{n} \binom{n}{i} = 2^i$, the above sum is equal to 2^{10}

Question 5. A rook on a chessboard is said to put another chess piece under attack if they are in the same row or column. How many ways are there to arrange 8 rooks on a chessboard (the usual 8 x 8 one) so that none are under attack?

Solution
We can think of two different ways to count all possible different arrangements of 8 rooks on a chessboards.

1^{st} way: In the first row (or column equivalently) you have 8 available positions to place a rook, then for the second row there are 7 valid positions available, 6 for the 3^{rd} and so on. Thus, we have 8! positions.
2^{nd} way: The first rook can be placed in any of the 8*8 positions. Then, the second rook can be placed in any of the 7*7 available positions, etc. Thus, we get $(8!)^2$ possible arrangements of the rooks in the chessboard. As we do not care for the order of rooks, we have to devide by 8!, getting the final result, 8!.