CS 70 Spring 2004 — Discrete Mathematics for CS — Papadimitriou/Vazirani — Review 1

Don't worry if you haven't seen all of the terms used in the following problems—many of these are taken from past semesters, and the coverage of topics among CS70 classes isn't always the same. Also note that I haven't read all of these in detail (much less worked through them), so there could be some errors.

1. Suppose you have n infinite, straight lines in the plane, no two of which are parallel and no three of which meet in a common point. Prove that the lines divide the plane into $(n^2+n+2)/2$ regions.

2. This problem is concerned with quantifiers (such as $\forall$). We didn't cover these in much detail, so don't worry if the concepts here seem foreign. That said, you may want to try this anyway.

   (a) A binary operator $f(x,y)$ is commutative if the result does not depend on the order of the arguments. State using quantifier notation the proposition that f is *not* commutative. (Matrix multiplication is an example of a noncommutative operator.)

   (b) An *identity* for a binary operator is an element that, when combined with any element x, yields x itself. For example, the identity for + is 0; the identity for $\times$ is 1. For a *noncommutative* operator, a *left-identity* is an identity when it is the first argument; a *right-identity* is an identity when it is the second argument. State using quantifier notation the propositions that i_L is a left-identity for f and that i_R is a right-identity for f.

   (c) Prove that for any i_L and i_R satisfying your definitions, it must be the case that $i_L = i_R$.

   (d) Does your proof imply that f has exactly one identity element? If not, what have you proved?

3. Double induction

   (a) Prove that $k^n > n!$ for all positive integers k,n with $n \leq 2k$.

   (b) Prove that $k^n < n!$ for all positive integers k,n with $n \geq 3k$.

4. Prove that $1 \cdot 1! + 2 \cdot 2! + \cdots + n \cdot n! = (n+1)! - 1$ for all integers $n \in \mathbf{N}$.

5. Strong induction

   (a) Which amounts of postage can be formed from combinations of 3¢ and 11¢ stamps?

   (b) Prove your claim using simple induction.

   (c) Prove your claim using strong induction.

   (d) Conjecture (and prove if possible) a general theorem concerning the amounts of postage that can be formed using stamps of values a and b.

6. Assign a grade of A (correct) or F (failure) to each of the following proofs. If you give a F, please explain exactly what is wrong with the structure or the reasoning in the *proof*. You should justify all your answers (remember, saying that the claim is false is *not* a justification).

(a) **Theorem 0.1**: *For all positive integers n, $\sum_{i=1}^{n} i = (n-1)(n+2)/2$.*
Proof: The proof will be by induction.
Base case: The claim is valid for $n = 1$.
Induction step: Assume that $\sum_{i=1}^{k} i = (k-1)(k+2)/2$. Then $\sum_{i=1}^{k+1} i = \left(\sum_{i=1}^{k} i\right) + (k+1)$. By the inductive hypothesis, $\sum_{i=1}^{k+1} i = (k-1)(k+2)/2 + (k+1)$. Collecting terms and simplifying, the right-hand side becomes $k(k+3)/2$. Thus $\sum_{i=1}^{k+1} i = ((k+1)-1)((k+1)+2)/2$, which completes the induction step. □

(b) **Theorem 0.2**: *For every $n \in \boldsymbol{N}$, $n^2 + n$ is odd.*
Proof: The proof will be by induction.
Base case: The natural number 1 is odd.
Inductive step: Suppose $k \in \mathbf{N}$ and $k^2 + k$ is odd. Then,

$$(k+1)^2 + (k+1) = k^2 + 2k + 1 + k + 1 = (k^2 + k) + (2k + 2)$$

is the sum of an odd and an even integer. Therefore, $(k+1)^2 + (k+1)$ is odd. By the Principle of Mathematical Induction, the property that $n^2 + n$ is odd is true for all natural numbers n. □

(c) **Theorem 0.3**: *For all $x, n \in \boldsymbol{N}$, if $nx = 0$ and $n > 0$, then $x = 0$.*
Proof: The proof will be by induction.
Base case: If $n = 1$, then the equation $nx = 0$ implies $x = 0$, since $nx = 1 \cdot x = x$ in this case.
Induction step: Fix $k > 0$, and assume that $kx = 0$ implies $x = 0$. Suppose that $(k+1)x = 0$. Note that $(k+1)x = kx + x$, hence we can conclude that $kx + x = 0$, or in other words, $kx = -x$. Now there are two cases:

Case 1: $x = 0$. In this case, $kx = -x = -0 = 0$, so $kx = 0$. Consequently, the inductive hypothesis tells us that $x = 0$.

Case 2: $x > 0$. In this case, $-x < 0$ (since $x > 0$). At the same time, $kx \geq 0$ (since $k, x \geq 0$). But this is impossible, since we know $kx = -x$. We have a contradiction, and therefore Case 2 cannot happen.

In either case, we can conclude that $x = 0$. This completes the proof of the induction step. □

(d) **Theorem 0.4**: *For all $x, y, n \in \boldsymbol{N}$, if $\max(x, y) = n$, then $x = y$.*
Proof: The proof will be by induction.
Base case: Suppose that $n = 0$. If $\max(x, y) = 0$ and $x, y \in N$, then $x = 0$ and $y = 0$, hence $x = y$.
Induction step: Assume that, whenever we have $\max(x, y) = k$, then $x = y$ must follow. Next suppose x, y are such that $\max(x, y) = k + 1$. Then it follows that $\max(x-1, y-1) = k$, so by the inductive hypothesis, $x - 1 = y - 1$. In this case, we have $x = y$, completing the induction step. □

(e) **Theorem 0.5**: *$\forall n \in \boldsymbol{N}$. $n^2 \leq n$.*
Proof: The proof will be by induction.
Base case: When $n = 0$, the statement is $0^2 \leq 0$ which is true.
Induction step: Now suppose that $k \in \mathbf{N}$, and $k^2 \leq k$. We need to show that

$$(k+1)^2 \leq k+1$$

Working backwards we see that:

$$\begin{aligned}
(k+1)^2 &\leq k+1 \\
k^2 + 2k + 1 &\leq k+1 \\
k^2 + 2k &\leq k \\
k^2 &\leq k
\end{aligned}$$

So we get back to our original hypothesis which is assumed to be true. Hence, for every $n \in \mathbf{N}$ we know that $n^2 \leq n$. $\square$

7. Induction over strings

   (a) Write a recursive definition of the length function $l(s)$ on strings.

   (b) Using induction over strings, prove that

   $$\forall x, y \in \Sigma^* \; l(x \cdot y) = l(x) + l(y)$$

8. Let B_n be the number of bit strings of length n that do not contain 00 as a substring. (For example, $A_3 = 5$, since the allowed bit-strings are 010, 011, 101, 110, and 111.) Let $C_n = B_n/2^n$, so C_n represents the fraction of bit strings of length n that do not contain 00. Prove that C_n gets arbitrarily close to zero as n increases. In other words: prove that, for every real number $\varepsilon > 0$, there exists n so that $C_n \leq \varepsilon$.

9. Consider the following variation on the stable marriage problem: In Hawkinsville this year, we have n eligible boys and girls. Everyone knows the preferences of all the boys, but the girls have managed to keep their preferences totally secret. Tomorrow night, the girls will throw a secret meeting (no boys allowed!), privately share their (true) preferences, and collude to come up with a set of fake preferences that they will announce to the village elders. As a twist, each girl may declare some boys to be unacceptable (she'd rather be single than marry any of them). Next week, the village elders will run the traditional marriage algorithm on the boys' (true) preferences along with the girls' announced (and totally fake) preferences. The output of the traditional marriage algorithm will be used to marry off all the eligible youngsters in Hawkinsville.

   Show that the girls can conspire to choose fake preferences so that when the wedding bells ring next week, we'll end up with a girl-optimal pairing. (By "optimal," I mean optimal with respect to everyone's true preferences.)

10. Binary GCD

    (a) Prove that the following statements are true for all $m, n \in \mathbf{N}$.

    $$\begin{aligned} \text{If } m \text{ is even and } n \text{ is even,} \quad & \gcd(m,n) = 2\gcd(m/2, n/2). \\ \text{If } m \text{ is even and } n \text{ is odd,} \quad & \gcd(m,n) = \gcd(m/2, n). \\ \text{If } m, n \text{ are both odd and } m \geq n, \quad & \gcd(m,n) = \gcd((m-n)/2, n). \end{aligned}$$

    (b) Give an algorithm that computes $\gcd(m,n)$ using at most $\mathbf{O}(\lg m + \lg n)$ subtractions, halvings, doublings, and odd/even tests.

11. The purpose of this problem is to teach you Big-**O** notation in a careful way. First, study the following. (If you'd like additional reading on this subject, you may refer to Rosen, Chapter 2.2.)

    Formally: If $f(n), g(n)$ are two non-negative functions of a single integer variable, the statement $f(n) \in \mathbf{O}(g(n))$ means that

    $$\exists N_0 \in \mathbf{N}. \quad \exists C \in \mathbf{N}. \quad \forall x \in \mathbf{N}. \quad x \geq N_0 \implies f(x) \leq C \cdot g(x).$$

    In other words, $\mathbf{O}(g(n))$ is the set of functions $\{f_i(n) : \exists N_0 \in \mathbf{N}. \exists C \in \mathbf{N}. \forall x \in \mathbf{N}. x \geq N_0 \implies f_i(x) \leq C \cdot g(x)\}$. This is the definition of Big-**O** notation.

    Informally: $f(n) \in \mathbf{O}(g(n))$ means, roughly, that $f(n)$ grows "no faster than" $g(n)$ (except possibly for a constant factor), as n gets large. For instance, $n^2 \in \mathbf{O}(n^2)$, $n(n+1)/2 \in \mathbf{O}(n^2)$, and $10000n^2 \in \mathbf{O}(n^2)$, because these functions all grow at asymptotically the same rate (ignoring constant factors). Also, $n^2 \in \mathbf{O}(n^3)$, because n^2 grows more slowly than n^3 does, as n gets large.

Some basic facts: If $f(n) \in \mathbf{O}(g(n))$ and $g(n) \in \mathbf{O}(h(n))$, then $f(n) \in \mathbf{O}(h(n))$. If $f(n) \in \mathbf{O}(g(n))$ and $f'(n) \in \mathbf{O}(g'(n))$, then $f(n) + f'(n) \in \mathbf{O}(g(n) + g'(n))$. If $f(n) \in \mathbf{O}(g(n))$ and $f'(n) \in \mathbf{O}(g'(n))$, then $f(n) \times f'(n) \in \mathbf{O}(g(n) \times g'(n))$.

Common notation: Instead of writing $f(n) \in \mathbf{O}(g(n))$, almost everyone instead writes $f(n) = \mathbf{O}(g(n))$. Strictly speaking, this is a sloppy abuse of notation, but this practice is widespread; you are guaranteed to see it throughout your studies of computer science, so be prepared. Also, we often write something like n^2 as a shorthand for the function $f(n) = n^2$, just to make our life easier.

Now, with that background established, do the following problems:

(a) Prove that $n^2 + 2003 \in \mathbf{O}(n^3)$.

Hint: One possible approach is to give an example of constants N_0, C that satisfy the definition.

(b) Prove that $100n^2 \lg n \in \mathbf{O}(n^3)$.

(c) True or false: There exists $e \in \mathbf{N}$ such that $2^n \in \mathbf{O}(n^e)$. (You do not need to justify your answer.)

(d) Critique the following argument. Is the reasoning valid? If not, why not?

> We have $n^2 = \mathbf{O}(n^4)$.
> Also, we have $n^2 = \mathbf{O}(n^3)$.
> By transitivity, it follows that $\mathbf{O}(n^4) = \mathbf{O}(n^3)$.
> This means that $n^4 = \mathbf{O}(n^3)$.

12. Z_N^* denotes the set $X = \{x : 0 < x < N \textit{ and } gcd(x,N) = 1\}$, together with the binary operation of multiplication modulo N. Show that Z_N^* is a group.

    The order of a group $G = (X, \cdot)$ is the cardinality (number of elements) of the set X. If $N = PQ$, where P and Q are distinct primes, what is the order of Z_N^*?

13. Use the polynomial interpolation formula to find a polynomial of degree 2, $P(x)$, over the finite field GF_{11}, such that $P(1) = 3$, $P(2) = 1$ and $P(3) = 7$.

14. Calculations

    (a) What is $7^{1,000,000,000} \bmod 101$? Justify your answer.

    (b) What is $2^{3^{1,000}} \bmod 47$? Show your work. (Note that a^{b^c} means $a^{(b^c)}$, not $(a^b)^c$.)