CS 70 Spring 2005 — Discrete Mathematics for CS — Clancy/Wagner

MT 2 Solns

Problem 1. (8 points)

Consider the function $a(\cdot)$ defined as follows:

$$\begin{aligned} a(0) &= 1 \\ a(1) &= 2 \\ a(n) &= (a(n-1) + 2a(n-2)) \bmod 11 \qquad \text{for } n \geq 2 \end{aligned}$$

(Corrections to the original version of the exam were made in class.)

(a) Fill in the blanks:

$a(2)$ = 4

$a(3)$ = 8

$a(4)$ = 5

(b) Determine $a(2005)$—we want a number, not an unevaluated expression—and show how you got your answer.

Solution #1: One can easily verify by induction that $a(n) = 2^n \bmod 11$. The answer to part (a) follows.

By Fermat's Little Theorem, $2^{11-1} \equiv 1 \pmod{11}$. Thus

$$a(2005) \equiv 2^{2005} \equiv 2^5 \cdot 2^{2000} \equiv 32 \cdot (2^{10})^{200} \equiv 32 \cdot 1 \equiv 10 \pmod{11}$$

In short, $a(2005) = 10$.

Solution #2: One can verify by direct computation that the sequence begins $1, 2, 4, 8, 5, 10, 9, 7, 3, 6, 1, 2, 4,$ and so on. We can see that $a(10) = a(0)$ and $a(11) = a(1)$. Since each element in the sequence depends on only the previous two elements, this means that the sequence is periodic with period 10. In particular, one can easily verify by strong induction on n that $a(n) = a(n \bmod 10)$.

It follows that $a(2005) = a(2005 \bmod 10) = a(5) = 10$.

Common mistakes: The most common error was failing to justify periodicity properly. For instance, some people said that since $a(10) = a(0)$, the sequence repeats (this isn't enough, since each element depends on the previous *two* elements). Some only calculated $a(0), \ldots, a(9)$, which isn't enough to conclude periodicity. Some tried to use the fact that 2 has an inverse modulo 11 to conclude periodicity; however, this condition is neither necessary nor sufficient.

Grading: 2 points for concluding that $a(n) = 2^n \bmod 11$ and filling in the values for $n = 2, 3, 4$. 6 points for applying Fermat's Little Theorem (or periodicity) correctly.

Problem 2. (6 points)

David Wagner has set up n identical dominoes in a line, in preparation for a demonstration of a proof by induction. Mike Clancy wants to sneak in and grab two dominoes from the line to take home as souvenirs; however, if he takes two adjacent dominoes, the induction demonstration won't work.

In how many ways can Mike select a subset of two *nonadjacent* dominoes from the line of n? Briefly explain your answer.

Let $F(n) =$ the number of ways to pick two nonadjacent dominoes from n.

Solution #1: Let $G(n) =$ the number of ways to do so that include the leftmost domino in the line, and $H(n) =$ the number of ways that don't include the first domino. Then $F(n) = G(n) + H(n)$. There are $G(n) = n-2$ ways to select another domino if the first is chosen. Also $H(n) = F(n-1)$. This yields the recurrence relation $F(n) = n-2+F(n-1)$. The base cases for this recurrence are $F(2) = F(1) = 0, F(3) = 1$, so $F(n) = (n-2)+(n-3)+\cdots+1 = (n-2)(n-1)/2 = \binom{n-1}{2}$.

Solution #2: $F(n) =$ the number of ways to pick two dominoes (without worrying about adjacency) minus the number of ways to pick two adjacent dominoes. This is $\binom{n}{2} - (n-1) = \binom{n-1}{2}$.

Solution #3: Start with a line of $n-2$ dominoes. We'll count the number of ways to add two new dominoes to the line, subject to the condition that the two new dominoes cannot be adjacent to each other. Since this is just a time-reversal of the original domino-stealing process, this will give us $F(n)$.

Examine the positions *between* the $n-2$ dominoes, including the position before the first domino and after the last. These positions represent places where the new dominoes could have placed; at most one new domino can be placed into any one of these positions, since the new dominoes mustn't be adjacent. There are $n-1$ such positions. The number of ways of recreating the line of n dominoes is the number of ways of selecting two out of these $n-1$ positions, or $\binom{n-1}{2}$.

Solution #4: Examine the $n-1$ positions *between* the dominoes. Mike steals two dominoes, a leftmost one and a rightmost one. Let's ask Mike to make a red mark at the in-between position just to the right of his leftmost stolen domino, and a blue mark at the position just to the left of his rightmost stolen domino. Since Mike's stolen dominoes are not adjacent, he won't mark the same position both red and blue; also, the red mark is always to the left of the blue mark. Thus, even a totally colorblind person can identify the color of each mark. Moreover, if you tell me where the two marks are, I can tell which two dominoes Mike stole—and vice versa. (For any pair of non-adjacent stolen dominoes, we get a pair of marks, possibly adjacent. For any pair of marks, possibly adjacent, we can identify a pair of non-adjacent dominoes that Mike could have stolen.)

This means that we have found a bijective correspondence between the set of ways to choose a pair of marks, and the set of ways to choose a pair of non-adjacent dominoes. There are exactly $\binom{n-1}{2}$ ways to choose 2 positions to mark from the set of $n-1$ positions, so this also counts the number of ways to choose a pair of non-adjacent dominoes.

Solution #5: Whatever Mike steals, he breaks the line up into three segments of dominoes (interrupted by his two stolen dominoes). Call the lengths of these three segments (a,b,c). Note that we must have $a \geq 0$, $b \geq 1$, $c \geq 0$, and $a+b+c = n-2$. Moreover, any triplet (a,b,c) satisfying these constraints uniquely identifies a way to choose a pair of non-adjacent dominoes. Therefore, there is a bijective correspondence between the set of non-adjacent domino-pairs and the set $S = \{(a,b,c) : a \geq 0, b \geq 1, c \geq 0, a+b+c = n-2\}$. Also, S can be put into bijective correspondence with the set $T = \{(a,b',c) : a \geq 0, b' \geq 0, c \geq 0, a+b'+c = n-3\}$, as can be seen by taking $b' = b-1$. Finally, by a standard stars-and-bars argument, $\#T = \binom{n-3+3-1}{3-1} = \binom{n-1}{2}$, so this must be the number of ways to choose a pair of non-adjacent dominoes.

Common mistakes: (1) Forgetting to divide by two, either because of overcounting, or thinking that order matters. (2) Some tried to break it up into two cases: (a) one of Mike's two dominoes is an end domino; (b) both dominoes are from the middle. However, case (b) is quite tricky to count correctly. There are $n-3$ ways to choose a first middle domino. One would like to say there are $n-5$ choices for the second middle domino, but this is not true if the first middle domino was right next to the end (it should be $n-4$).

Grading: -1 point for a minor arithmetic mistake; -2 points for a significant counting error (e.g., forgetting to divide by 2); -3 to -6 points for more severe errors.

Problem 3. (6 points)

Let $f(n) = n^2 + 1000n$.

(a) $\boxed{\text{True}}$ or false: $f(n) \in \mathbf{O}(n^3)$.

(b) Prove that your answer in part (a) is correct.

Solution: The answer to part (a) is true. For a proof, we need a constant c_0 and a starting point n_0 such that for all $n > n_0$, we have $n^2 + 1000n \le c_0 \cdot n^3$. Taking $n_0 = 1000$ and $c_0 = 2$, we find that $n^2 + 1000n \le n^2 + n^2 \le 2n^2 \le 2n^3 \le c_0 \cdot n^3$ for all $n \ge n_0$.

Common mistakes: This question exposed many misconceptions and bad proof habits.

- Backwards reasoning. Some started by writing down the condition $(\exists c_0, n_0.\ \forall n \ge n_0.\ f(n) \le c_0 \cdot n^3)$, then manipulating it until you get something true. This is a converse error.
- Informal handwaving. You were supposed to provide a proof. Informal statements like "n^3 grows more quickly than $1000n$", while accurate, need to be justified.
- Use of calculus, limits, or other results about big-Oh, without citing the theorem or result. We asked you to cite all theorems or results from the book or lectures that you used. While calculus or limits are useful in informally establishing big-Oh results, they are not ideal for use in a formal proof (unless you are willing to look up and cite an appropriate theorem), because there are technical restrictions on when they can be applied. Most who used calculus did not state the general conditions under which calculus can be applied or check that the technical conditions are satisfied.
- Proofs with no words. A proof is an essay. Any proof has some logical structure, and you usually need to explain that structure in English. If you don't include any words, the reader is left to guess at how the formulas on the page are related to each other. Does one follow from another? A good rule of thumb: look at the page. If you see more mathematical symbols than words, your proof is probably lacking.

Grading: (1) Just saying "true": 1/6 points. (2) Picking an appropriate c_0, n_0: 2/6 points. (3) Picking an appropriate c_0, n_0; checking that $f(n_0) \le c_0 \cdot n_0^3$; and handwaving about the case where $n > n_0$ (or a good start on proving what happens when $n > n_0$, but ultimately wrong reasoning): 4/6 points. (4) A correct and rigorous proof: 6/6 points.

Problem 4. (8 points)

Prove that $3^{3n+1} + 2^{n+1}$ is always divisible by 5, for all $n \in \mathbf{N}$.

Solution #1: Proof by induction on n. Let $F(n) = 3^{3n+1} + 2^{n+1}$. Base cases: $F(0) = 3+2$ and $F(1) = 81+4$, both of which are divisible by 5.

Induction step: assume $F(n)$ is divisible by 5, and examine $F(n+1)$:

$$\begin{aligned}
F(n+1) &= 3^{3(n+1)+1} + 2^{(n+1)+1} \\
&= 3^{3n+4} + 2^{n+2} \\
&= 3^3 \cdot 3^{3n+1} + 2 \cdot 2^{n+1} \\
&= 27 \cdot 3^{3n+1} + 2 \cdot 2^{n+1} \\
&= 27 \cdot 3^{3n+1} + 27 \cdot 2^{n+1} - 25 \cdot 2^{n+1} \\
&= 27F(n) - 25 \cdot 2^{n+1}
\end{aligned}$$

$F(n)$ is divisible by 5 by the induction hypothesis. Also $25x$ is divisible by 5 for any integer x. Therefore the difference, which is $F(n+1)$, is also divisible by 5.

Solution #2: A number is divisible by 5 if and only if it is zero modulo 5. Let n be arbitrary, and calculate:

$$F(n) \equiv 3^{3n+1} + 2^{n+1} \equiv 3 \cdot (3^3)^n + 2 \cdot 2^n \equiv 3 \cdot 27^n + 2 \cdot 2^n \equiv 3 \cdot 2^n + 2 \cdot 2^n \equiv 5 \cdot 2^n \equiv 0 \pmod 5,$$

using the fact that $27 \equiv 2 \pmod 5$.

Solution #3: Proof by strong induction on n. Base cases: $F(0) = 3+2 = 5$. $F(1) = 81+4 = 85$. $F(2) = 2187+8 = 2195$. $F(3) = 59049+16 = 59065$. All of these are divisible by 5.

Induction step: Assume that $n \geq 4$, and that $F(n-4)$ is divisible by 5. Fermat's little theorem says that $3^4 \equiv 1 \pmod 5$, so $3^{3n+1} \equiv 3^{3(n-4)+1} \cdot 3^{12} \equiv 3^{3(n-4)+1} \pmod 5$. Likewise, $2^{n+1} \equiv 2^{(n-4)+1} \pmod 5$. This means that $F(n) = 3^{3n+1} + 2^{n+1}$ differs from $F(n-4) = 3^{3(n-4)+1} + 2^{(n-4)+1}$ by a multiple of 5. Since $F(n-4)$ is divisible by 5, $F(n)$ must be, too.

Common mistakes: (1) Some tried a few numbers, noticed a pattern emerge, and stopped there. This question asked for a proof, so noticing an apparent pattern is not enough. (2) Some followed Solution #1 or #2, got as far as the line with 27, and couldn't make any useful progress from there. (3) Backwards reasoning.

Grading: (1) Notice a pattern, without explanation (or with bogus reasoning): 2/8 points. (2) Notice a pattern, and handwave or give an informal explanation: 3/8 points. (3) Get to the line with 27, and get stuck there: 3/8 points. (4) A correct and rigorous proof: 8/8 points. (5) Also: -1 point for adding "$\equiv 0$ mod 5" to every line, where it doesn't belong.

Problem 5. (6 points)

Below is an iterative implementation of a fast exponentiation algorithm.

```
int exp (int x, int n) {
    int m = n;
```

```
    int r = 1;
    int z = x;
    while (true) {
      // ***
      boolean isOdd = (m%2 != 0);
      m = m/2;
      if (isOdd) {
        r = z*r;
      }
      if (m == 0) {
        return r;
      }
      z = z*z;
    }
}
```

There is an invariant that is always true when control reaches the position marked "***". In particular, there is a relation $x^n = f(r,z,m)$ that holds between x, n, r, z, and m at the start of the loop.

Give an expression for f. (You do not need to justify your answer.)

Solution: The answer is $x^n = r \cdot z^m$.

You did not need to justify your answer, but for the curious, this invariant can be verified by induction on the number of iterations of the loop. Base case: It is true before the first iteration, since then $m = n$, $r = 1$, and $z = x$, so $r \cdot z^m = 1 \cdot x^n = x^n$.

Induction step: Let m, r, z denote the values at the start of one iteration of the loop, and m', r', z' denote the updated values at the end of the loop body. Assume $x^n = r \cdot z^m$. We will show that $x^n = r' \cdot (z')^{m'}$.

If m is odd, then $m' = (m-1)/2$, $r' = zr$, and $z' = z^2$, so

$$r' \cdot (z')^{m'} = (zr) \cdot (z^2)^{(m-1)/2} = (zr) \cdot z^{m-1} = r \cdot z^m = x^n.$$

If m is even and positive, then $m' = m/2$, $r' = r$, and $z' = z^2$, so

$$r' \cdot (z')^{m'} = r \cdot (z^2)^{m/2} = r \cdot z^m = x^n.$$

Including this invariant as a comment in the code would be a good way to document the algorithm and make it easier for others to understand how the algorithm works. Also, it is a good way to check that your code is correct, since we can see that if the invariant holds, then the iteration where $m = 0$ trivially returns the correct value $x^n = r$.

Grading: 3 points for something that was close, e.g., $x^n = (r \cdot z)^m$ or $x^n = r^m \cdot z$.

Problem 6. (6 points)

Alice wants to send a RSA-encrypted message M to her friends Bob and Carol. Bob and Carol use public keys (e_1, n) and (e_2, n). Notice they use the same modulus, but different exponents. Alice sends $C_1 = M^{e_1} \bmod n$ to Bob, $C_2 = M^{e_2} \bmod n$ to Carol. You eavesdrop on Alice's transmissions and learn C_1 and C_2.

Show how you can recover M. You may assume that you know e_1, e_2, and n, and that e_1 and e_2 are two different primes.

Solution:
Insight 1: We would like somehow to isolate M with an exponent of 1.

Insight 2: Since e_1 and e_2 are relatively prime—i.e., their greatest common divisor is 1—there exist a and b such that $a \cdot e_1 + b \cdot e_2 = 1$.

To recover M, we first find a and b using the extended Euclidean algorithm. Then we calculate $C_1^a \times C_2^b \bmod n$. Note that

$$C_1^a \times C_2^b \equiv (M^{e_1})^a \times (M^{e_2})^b \equiv M^{a \cdot e_1 + b \cdot e_2} \equiv M^1 \equiv M \pmod{n},$$

so this reveals the message.

Note that every step above can be performed efficiently. We use the extended Euclidean algorithm, fast exponentiation algorithm, and algorithms for modular arithmetic, which all run in polynomial time.

Intuition: How could you have discovered this answer? Here is one possible thought process.

Suppose you are given $C_1 = M^3 \bmod n$, $C_2 = M^2 \bmod n$. Since $3 = 2 + 1$, you could calculate M as $M \equiv C_1 \cdot (C_2)^{-1} \equiv M^3 \cdot M^{-2} \pmod{n}$.

Suppose you are given $C_1 = M^5 \bmod n$, $C_2 = M^3 \bmod n$. Since $5 = 3 + 2$, you could calculate $M^2 \bmod n$ as $M^2 \equiv C_1 \cdot (C_2)^{-1} \equiv M^5 \cdot M^{-2} \pmod{n}$. Now you know $M^2 \bmod n$ and $M^3 \bmod n$. and then you could apply the method of the previous paragraph to deduce M.

Suppose you are given $C_1 = M^{103} \bmod n$, $C_2 = M^5 \bmod n$. Since $103 = 20 \cdot 5 + 3$, you could calculate $M^3 \bmod n$ as $M^3 \equiv C_1 \cdot (C_2^{20})^{-1} \equiv M^{103} \cdot M^{-100} \pmod{n}$. At this point, you have reduced the problem to that of the previous paragraph.

Finally, suppose you are given $C_1 = M^{e_1} \bmod n$ and $C_2 = M^{e_2} \bmod n$ with $e_1 > e_2$. Compute $a = \lfloor e_1/e_2 \rfloor$ and $e_3 = e_1 \bmod e_2$. Since $e_1 = a \cdot e_2 + e_3$, you can calculate $M^{e_3} \bmod n$ as $M^{e_3} \equiv C_1 \cdot (C_2^a)^{-1} \equiv M^{a \cdot e_2 + e_3} \cdot M^{-a \cdot e_2} \pmod{n}$. Then, you can recursively solve the problem with exponents e_2 and e_3. Since $0 < e_3 < e_2$ and $\gcd(e_1, e_2) = 1$, this process must eventually terminate with some $e_k = 1$, and then you know M. The sequence of exponents $e_1, e_2, e_3, \ldots, e_k = 1$ visited during this process mimics exactly the sequence of intermedate results computed by the extended Euclidean algorithm.

Grading: Hardly anyone got this problem. We gave 1 point for anyone who found a correct, exponential-time algorithm, or who found an equation of the form $M =$ (something) where the (something) contains values you don't have (e.g., p, q, d_1, d_2, $\varphi(n)$).