Homework 3 Solutions

Tuesday, July 8

CS 70: Discrete Mathematics and Probability Theory, Summer 2014

1. [12 points] Practice with modular arithmetic.

1a. [3 points] Calculate $4^{273} \bmod 11$ using repeated squaring. Show your work. You should not need a calculator.

Answer: First note that $4^{273} = 4^{256} \cdot 4^{16} \cdot 4^1$.
To find each of these powers of 4, we compute:

$$\begin{aligned} 4^1 &\equiv 4 \pmod{11} \\ 4^2 &\equiv 4^2 = 16 \equiv 5 \pmod{11} \\ 4^4 &\equiv (4^2)^2 \equiv 5^2 = 25 \equiv 3 \pmod{11} \\ 4^8 &\equiv 3^2 \equiv 9 \pmod{11} \\ 4^{16} &\equiv 9^2 \equiv (-2)^2 \equiv 4 \pmod{11} \end{aligned}$$

Using the pattern, we know $4^{256} \equiv 4 \pmod{11}$. Plugging into the above,

$$\begin{aligned} 4^{273} &= 4^{256} \cdot 4^{16} \cdot 4^1 \\ &\equiv 4 \cdot 4 \cdot 4 \equiv 5 \cdot 4 = 20 \equiv 9 \pmod{11} \end{aligned}$$

1b. [3 points] Find a trick to calculate $4^{273} \bmod 11$ more easily using Fermat's Little Theorem. Justify that your trick works.

Answer: Since 11 is prime, we know that $\forall a \in \{1, 2, \dots, 10\}$, $a^{10} \equiv 1 \pmod{11}$. Thus, $4^{273} = (4^{10})^{27} \cdot 4^3 \equiv 1^{27} \cdot 4^3 = 64 \equiv 9 \pmod{11}$.

Note: In fact, from this, we can claim something even stronger: $\forall a \in \{1, 2, \dots, 10\}$, $a^n \equiv a^{(n \pmod{10})} \pmod{11}$.

1c. [3 points] Similarly to part 1b, calculate $4^{5^{273}} \bmod 11$ using Fermat's Little Theorem. (Note that $4^{5^{273}}$ means $4^{(5^{273})}$.)

Answer: From what we found earlier, we first compute $5^{273} \equiv 5 \pmod{10}$. Thus, $4^{5^{273}} \equiv 4^5 \equiv 2^{10} \equiv 1 \pmod{11}$.

Note: It is tempting to first evaluate $5^{273} \equiv 4 \pmod{11}$, then compute $4^4 \equiv 3 \pmod{11}$. It is **not true** in general that $x^y \equiv x^{(y \pmod{p})} \pmod{p}$. For a simple counterexample, consider $2^5 \pmod{5}$. $32 \equiv 2 \pmod{5}$, but $2^0 \equiv 1 \pmod{5}$.

1d. [3 points] Compute $5^{82} \bmod 21$. You shouldn't need much arithmetic for this!

Answer: Recall that $x^{(p-1)(q-1)} = 1 \bmod pq$ if p and q are primes. Since $pq = 21$, we deduce that $p = 3$ and $q = 7$. Then $(p-1)(q-1) = 12$. So $5^{12k} = 1 \bmod 21$. We reduce the exponent $82 = 10 \bmod 12$. Then $5^{10} = (5^2)^5 = 25^5 = 4^5 = (4^3)(4^2) = 64 * 16 = 1 * 16 = 16 \bmod 21$.

2. [15 points] Bob decides to use RSA with $p = 11$, $q = 23$, and $e = 7$. Bob publishes $N = pq = 253$ and e as his public key. (You should convince yourself that e is indeed a valid encryption exponent.)

   2a. [5 points] Show how to find Bob's private key d using the extended Euclidean algorithm. (Hint: If you do it right, you should get $d = 63$.)

   Answer: We need to compute $e^{-1} \pmod{(p-1)(q-1)}$, or $7^{-1} \pmod{220}$.

egcd(220, 7)	$1 =$	-2	$\cdot$	220	$+$	63	$\cdot$	7
egcd(7, 3)	$1 =$	1	$\cdot$	7	$+$	-2	$\cdot$	3
egcd(3, 1)	$1 =$	0	$\cdot$	3	$+$	1	$\cdot$	1
egcd(1, 0)	$1 =$	1	$\cdot$	1	$+$	0	$\cdot$	0

   Thus, $d = 63$.

   2b. [5 points] Alice wants to send the message 44 (an integer between 0 and 252) to Bob. What is the encrypted message that Alice sends? You may use a calculator, but you must show all intermediate steps of the repeated squaring algorithm.

   Answer: We now must compute $44^e \pmod N$, or $44^7 \pmod{253}$. Using our repeated-squaring algorithm,

$$\begin{aligned}
44^1 &\equiv 44 \pmod{253} \\
44^2 &\equiv 44^2 = 1936 \equiv 165 \pmod{253} \\
44^4 &\equiv 165^2 = 27225 \equiv 154 \pmod{253}
\end{aligned}$$

   Therefore, the message Alice should send is $44^7 = 44^4 \cdot 44^2 \cdot 44^1 \equiv 154 \cdot 165 \cdot 44 \equiv 33 \pmod{253}$.

   2c. [5 points] Suppose Bob receives from Alice the ciphertext 103. What was the original message that Alice sent? You may use a calculator, but you must show all intermediate steps of the repeated squaring algorithm.

   Answer: We must now compute $103^d \pmod N$, or $103^{63} \pmod{253}$.

$$\begin{aligned}
103^1 &\equiv 103 \pmod{253} \\
103^2 &\equiv 103^2 = 10609 \equiv 236 \pmod{253} \\
103^4 &\equiv 236^2 = 55696 \equiv 36 \pmod{253} \\
103^8 &\equiv 36^2 = 1296 \equiv 31 \pmod{253} \\
103^{16} &\equiv 31^2 = 961 \equiv 202 \pmod{253} \\
103^{32} &\equiv 202^2 = 40804 \equiv 71 \pmod{253}
\end{aligned}$$

   Therefore, Alice's original message was $103^{63} = 103^{32+16+8+4+2+1} \equiv 71 \cdot 202 \cdot 31 \cdot 36 \cdot 236 \cdot 103 \equiv 130 \pmod{253}$.

3. [13 points] Suppose we tried to simplify the RSA cryptosystem by using just a prime p instead of the composite modulus $N = pq$. Similarly to RSA, we would have an encryption exponent e that is relatively prime to $p-1$, and the encryption of message x would be $(x^e \bmod p)$. Show that this scheme is not secure by giving an efficient algorithm that, given p, e, and $(x^e \bmod p)$, computes $x \bmod p$.

   3a. [4 points] Describe your algorithm.

   Answer: We propose the following algorithm:

   Step 1: Compute d, the multiplicative inverse of $e \pmod{p-1}$, using egcd.

   Step 2: Compute $(x^e)^d \pmod{p}$ with repeated squaring.

   We claim this algorithm will correctly recover the original message x.

   3b. [6 points] Prove that your algorithm is correct.

   Answer: To show that our algorithm is correct, we must show:

$$(x^e)^d \equiv x \pmod{p}, \forall x \in \{0, 1, \ldots, p-1\}$$

   Note: if $x = 0$, our claim is trivially true. For the following, we will consider the case when $x \in \{1, 2, \ldots, p-1\}$.

   By definition, $ed \equiv 1 \pmod{p-1}$, or $ed = k(p-1)+1$ for some integer k (note that this is not a modular equation!). We now consider

$$x^{ed} = x^{k(p-1)+1} = (x^{p-1})^k \cdot x.$$

   From Fermat's Little Theorem, we can rewrite the above as

$$(x^{p-1})^k \cdot x \equiv 1^k \cdot x \equiv x \pmod{p}.$$

   Therefore, we have shown that our algorithm will always correctly recover the original message x. □

   3c. [3 points] Analyze the running time of your algorithm.

   Answer: This algorithm runs in $O(\log^3 p)$ time, since we only used egcd and repeated squaring.

4. [18 points] Let f be a polynomial of degree at most d. The *coefficient representation* of f is the sequence $a_d, a_{d-1}, \ldots, a_1, a_0$ of coefficients of f. A *point-value representation* of f is a collection $\{(x_1, f(x_1)),\ (x_2, f(x_2)),\ \ldots,\ (x_t, f(x_t))\}$ of the values of f at any t points $x_1, x_2, \ldots, x_t$, where $t \geq d+1$. (Recall from Lecture Note 7 that a polynomial of degree d is completely determined by its values at any $d+1$ points. Note that t may be greater than $d+1$, so more points than necessary may be given.)

   In the following questions, let f and g be any two real polynomials of degree at most d.

   4a. [3 points] What is the maximum degree of the product polynomial fg?

   Answer: Suppose the polynomial $f(x) = a_0 + a_1 x + \cdots + a_k x^k$ and $g(x) = b_0 + b_1 x + \cdots + b_l x^l$ where k and l are at most d. The coefficient of x^s in fg is

$$\sum_{i+j=s} a_i b_j = a_0 b_s + a_1 b_{s-1} + \cdots + a_s b_0$$

Here, we see that this is 0 if $s > k + l$, meaning that the maximum degree if fg is $k + l \leq 2d$.

4b. [5 points] Given coefficient representations of f and g, explain how to compute the coefficient representation of fg using $O(d^2)$ arithmetic operations (additions / subtractions / multiplications / divisions) over real numbers.

Answer: Our expression from (a) tells us that we can compute each coefficient of fg with at most $s + 1$ multiplications and s additions, which is $O(s)$. We also know that $s \leq 2d$, which means each coefficient can be computed in $O(d)$.
Thus, since there are at most $2d + 1$ coefficients, the coefficients of fg can be computed with $O(d^2)$ operations.

4c. [5 points] Now suppose that f and g are specified by point-value representations at t points for some $t \geq d + 1$, i.e., f is specified as $(x_1, f(x_1)),\ (x_2, f(x_2)),\ \ldots,\ (x_t, f(x_t))$, and g as $(x_1, g(x_1)),\ (x_2, g(x_2)),\ \ldots,\ (x_t, g(x_t))$. With a suitable value of t (which you should specify), show how to compute a point-value representation of fg using only $O(d)$ arithmetic operations.

Answer: First note that $fg(x_i) = f(x_i) \cdot g(x_i)$. From (a), we found that the maximum number of non-zero coefficients fg has is $2d + 1$. Thus, if we are given $t \geq 2d + 1$ points, we can compute the point-value representation of fg with at most $2d + 1 = O(d)$ operations.

4d. [5 points] Suppose that polynomial g divides polynomial f, and that f, g are given in point-value representation as in part (4c) with $t = d + 1$. Show how to compute a point-value representation for the quotient f/g using $O(d)$ arithmetic operations. **Be very careful proving that your algorithm is correct.**

Answer: Similar to (c), we can compute $[f/g](x_i) = f(x_i)/g(x_i)$ with one division. However, we must be careful, because $g(x_i)$ might be 0. Recall that the most roots g can have is $\mathit{degree}(g)$, meaning that $f(x)/g(x)$ is defined for at least $d + 1 - \mathit{degree}(g)$ points. And since we are given that g divides f, we know

$$\mathit{degree}(f/g) = \mathit{degree}(f) - \mathit{degree}(g) \leq d - \mathit{degree}(g).$$

Thus, we can find a point-value representation of f/g using $\mathit{degree}(f/g) + 1 = O(d)$ divisions.

5. [10 points] In this problem, you will give two different proofs of the following theorem: For every prime p, every polynomial over $GF(p)$ is equivalent to a polynomial of degree at most $p - 1$. (Two polynomials f, g over $GF(p)$ are said to be equivalent iff $f(x) = g(x)$ for all $x \in GF(p)$.)

5a. [5 points] Show how the theorem follows from Fermat's Little Theorem. (Hint: Be careful! It is not true that $x^{p-1} \equiv 1 \pmod{p}$ for all $x \in \{0, 1, \ldots, p - 1\}$. Why not?)

Answer: From Fermat's Little Theorem, we know $\forall x \not\equiv 0,\ x^{p-1} \equiv 1 \pmod{p}$.
Multiplying both sides by x, and noting that $0^p \equiv 0 \pmod{p}$, we can see that

$$\forall x,\ x^p \equiv x \pmod{p}.$$

Therefore, any x^k, where $k \geq p$ will be equivalent to x^n, where $n \in \{0, 1, \dots, p-1\}$, and will have a degree at most $p-1$.

5b. [5 points] Now prove the theorem using what you know about Lagrange interpolation.

Answer: Since a polynomial f of degree d is described completely by $d+1$ points, we cannot specify a polynomial of degree $\geq p$ because $f(p) = f(0)$ (and so forth for other values larger than p) over $GF(p)$. Thus, we can only specify at most p unique points. Therefore, every polynomial is equivalent to a polynomial of degree at most $p-1$.

6. [10 points] Consider the following variant of the secret sharing problem. We wish to share a secret among twenty-one people, divided into three groups of seven, so that the following condition is satisfied. A subset of the twenty-one people can recover the secret if and only if it contains majorities (at least four out of seven) of at least two of the groups. How would you modify the standard secret sharing scheme to achieve this condition? (Hint: Try a two-level scheme, one level for groups, the other for people within the group.)

Answer: First, we have the secret s as the constant term in a degree-1 polynomial $f(x) = ax + s$ over $GF(p)$, where $p > 7$. We hand out $f(1)$, $f(2)$, and $f(3)$ to each of the three groups. However, we will hand them out as a secret to be shared among each of the seven people within the groups (so each point of f given out becomes another secret, hence the hierarchical part of the scheme).
For each group i, we have a polynomial g_i of degree 3 such that $g_i(0) = f(i)$. Then, we hand out $g_i(1), g_i(2), \dots, g_i(7)$ to each of the seven group members.
Now, if any four members of a group get together, they can recover their group's secret, which they can share with another group (who also must have at least four people) to recover the original secret.

7. [12 points] Alice wants to send the message (a_0, a_1, a_2) to Bob, where each $a_i \in \{0, 1, 2, 3, 4\}$. She encodes it as a polynomial P of degree ≤ 2 over $GF(5)$ such that $P(0) = a_0$, $P(1) = a_1$, and $P(2) = a_2$, and she sends the packets $(0, P(0))$, $(1, P(1))$, $(2, P(2))$, $(3, P(3))$, $(4, P(4))$. Two packets are dropped, and Bob only learns that $P(0) = 4$, $P(3) = 1$, and $P(4) = 2$. Help Bob recover Alice's message. Use Lagrange interpolation and show all your work.

Answer: Let's do some Lagrange Interpolation! Inverse pairs mod 5: $(1,1), (2,3), (4,4)$. All operations performed mod 5.

$$\begin{aligned}
\Delta_0 &= \frac{(x-3)(x-4)}{(0-3)(0-4)} = \frac{x^2 - 7x + 12}{(-3)(-4)} = 3(x^2 + 3x + 2) = 3x^2 + 4x + 1 \\
\Delta_3 &= \frac{(x-0)(x-4)}{(3-0)(3-4)} = \frac{x^2 - 4x}{(3)(-1)} = 3(x^2 + x) = 3x^2 + 3x \\
\Delta_4 &= \frac{(x-0)(x-3)}{(4-0)(4-3)} = \frac{x^2 - 3x}{(4)(1)} = 4(x^2 + 2x) = 4x^2 + 3x
\end{aligned}$$

Thus, our original polynomial P is

$$\begin{aligned}
4\Delta_0 + 1\Delta_3 + 2\Delta_4 &= 4(3x^2 + 4x + 1) + (3x^2 + 3x) + 2(4x^2 + 3x) \\
&= (2x^2 + x + 4) + (3x^2 + 3x) + (3x^2 + x) \\
&= 3x^2 + 4
\end{aligned}$$

To recover (a_0, a_1, a_2), we compute

$$\begin{aligned}
P(0) &= 4 \\
P(1) &= 2 \\
P(2) &= 1
\end{aligned}$$