CS 70 Discrete Mathematics and Probability Theory
Summer 2020 Course Notes

HW 3

1 Modular Exponentiation

Compute the following. You only need repeated squaring in one of these questions!

(a) $8^{11111} \pmod 9$

(b) $3^{160} \pmod{23}$

(c) $218^3 \pmod 9$

(d) $998^{156} \pmod{13}$

Solution:

(a) 8 is its own inverse mod 9, therefore, if 8 is raised to an odd power, the number will be 8 mod 9. So the answer is 8.

Also notice that $8 \equiv -1 \pmod 9$ so $8^{11111} \equiv (-1)^{11111} \equiv -1 \equiv 8 \pmod 9$. In general, $m-1 \equiv -1 \pmod m$, so $m-1$ is always its own inverse. This is a useful trick so you can avoid computing the inverse of $m-1$ by hand. You can also check that $(m-1)^2 \equiv m^2 - 2m + 1 \equiv 1 \pmod m$, which is another proof that $m-1$ is its own inverse modulo m.

(b) We can notice that $160 = 128 + 32$, the sum of two powers of two. Then, we can use repeated squaring to compute this problem.

$3^2 \equiv 9 \pmod{23}$

$3^4 \equiv (3^2)^2 \equiv 9^2 \equiv 12 \pmod{23}$

$3^8 \equiv (3^4)^2 \equiv 12^2 \equiv 6 \pmod{23}$

$3^{16} \equiv (3^8)^2 \equiv 6^2 \equiv 13 \pmod{23}$

$3^{32} \equiv (3^{16})^2 \equiv 13^2 \equiv 8 \pmod{23}$

$3^{64} \equiv (3^{32})^2 \equiv 8^2 \equiv 18 \pmod{23}$

$3^{128} \equiv (3^{64})^2 \equiv 18^2 \equiv 2 \pmod{23}$

$3^{160} \equiv (3^{128})(3^{32}) \equiv (2)(8) \equiv 16 \pmod{23}$

Note that in this problem we can also simplify the exponent first by using FLT.

(c) $218^3 \equiv (200 + 10 + 8)^3 \equiv (2 * 1 * 1 + 1 + -1)^3 \equiv 8 \pmod 9$

(d) $998^{156} \equiv (998^{12})^{13} \pmod{13}$. Thus from FLT, the answer is 1.

2 Sparsity of Primes

A prime power is a number that can be written as p^i for some prime p and some positive integer i. So, $9 = 3^2$ is a prime power, and so is $8 = 2^3$. $42 = 2 \cdot 3 \cdot 7$ is not a prime power.

Prove that for any positive integer k, there exists k consecutive positive integers such that none of them are prime powers.

Hint: this is a Chinese Remainder Theorem problem

Solution:

We want to find x such that $x+1, x+2, x+3, \ldots x+k$ are all not powers of primes. We can enforce this by saying that $x+1$ through $x+k$ each must have two distinct prime divisors. So, select $2k$ primes, $p_1, p_2, \ldots, p_{2k}$, and enforce the constraints

$$\begin{aligned} x+1 &\equiv 0 \pmod{p_1 p_2} \\ x+2 &\equiv 0 \pmod{p_3 p_4} \\ &\vdots \\ x+i &\equiv 0 \pmod{p_{2i-1} p_{2i}} \\ &\vdots \\ x+k &\equiv 0 \pmod{p_{2k-1} p_{2k}} \end{aligned}$$

By Chinese Remainder Theorem, we can calculate the value of x so this x must exist, and thus, $x+1$ through $x+k$ are not prime powers.

What's even more interesting here is that we could select any $2k$ primes we want!

3 LOTUS but for CRT

Suppose that p and q are distinct odd primes and a is an integer such that $\gcd(a, pq) = 1$. Prove that $a^{(p-1)(q-1)+1} \equiv a \pmod{pq}$. *Hint: Use both Fermat's Little Theorem and Chinese Remainder Theorem!*

As an aside, perhaps after doing the problem, see the etymology section of the wiki page on Law of the unconscious statistician to see the derivation of this question's title.

Solution:

Because $\gcd(a, pq) = 1$, we have that a does not divide p and a does not divide q. By Fermat's Little Theorem,

$$a^{(p-1)(q-1)+1} = (a^{(p-1)})^{(q-1)} \cdot a \equiv (1)^{q-1} \cdot a \equiv a \pmod{p}.$$

Similarly, by Fermat's Little Theorem, we have

$$a^{(p-1)(q-1)+1} = (a^{(q-1)})^{(p-1)} \cdot a \equiv (1)^{p-1} \cdot a \equiv a \pmod{q}.$$

Now, we want to use this information to conclude that $a^{(p-1)(q-1)+1} \equiv a \pmod{pq}$. We cannot just conclude this without proof! We must prove it. Let us take a detour to show the more general result (you could write this out separately as a lemma if you want).

Consider the system of congruences

$$x \equiv a \pmod{p}$$
$$x \equiv a \pmod{q}.$$

Let's run the Chinese Remainder Theorem (CRT). By the CRT, the solution to our system of congruences will be

$$x \equiv a \cdot q^{-1} \pmod{p} \cdot q + a \cdot p^{-1} \pmod{q} \cdot p \pmod{pq}.$$

We also know that from Bezout's lemma, since p and q are relatively prime, we know there exist integers g, h such that

$$g \cdot p + h \cdot q = 1.$$

This g and h are respectively $p^{-1} \pmod{q}$ and $q^{-1} \pmod{p}$. (Verify this for yourself - take bezout's lemma mod q and see what happens! (then start over and take it mod p))

Finally we can plug in to the solution to get

$$x \equiv a \cdot h \cdot q + a \cdot g \cdot p \equiv a(h \cdot q + g \cdot p) \equiv a(1) \equiv a \pmod{pq}.$$

Therefore by the CRT we know that the set of solutions that satisfy both $x \equiv a \pmod{p}$ and $x \equiv a \pmod{q}$ is exactly the set of solutions that satisfy $x \equiv a \pmod{pq}$.

So now going back to our first result from Fermat's Little Theorem, since $a^{(p-1)(q-1)+1} \equiv a \pmod{p}$ and $a^{(p-1)(q-1)+1} \equiv a \pmod{q}$, then by the CRT we know that $a^{(p-1)(q-1)+1}$ satisfies $a^{(p-1)(q-1)+1} \equiv a \pmod{pq}$.

Now that you have seen and understand this, you may use this on exams without proof.

4 Squared RSA

(a) Prove the identity $a^{p(p-1)} \equiv 1 \pmod{p^2}$, where a is coprime to p, and p is prime. (Hint: Try to mimic the proof of Fermat's Little Theorem from the notes.)

(b) Now consider the RSA scheme: the public key is $(N = p^2q^2, e)$ for primes p and q, with e relatively prime to $p(p-1)q(q-1)$. The private key is $d = e^{-1} \pmod{p(p-1)q(q-1)}$. Prove that the scheme is correct for x relatively prime to both p and q, i.e. $x^{ed} \equiv x \pmod{N}$. (Hint: Try to mimic the proof of RSA correctness from the notes.)

Solution:

(a) We mimic the proof of Fermat's Little Theorem from the notes.
Let S be the set of all numbers between 1 and p^2-1 (inclusive) which are relatively prime to p. We can write

$$S = \{1, 2, \ldots, p-1, p+1, \ldots, p^2-1\}$$

Define the set

$$T = \{a, 2a, \ldots, (p-1)a, (p+1)a, \ldots, (p^2-1)a\}$$

We'll show that $S \subseteq T$ and $T \subseteq S$, allowing us to conclude $S = T$:

- $S \subseteq T$: Let $x \in S$. Since $\gcd(a,p) = 1$, the inverse of a exists $\pmod{p^2}$. For ease of notation, we use a^{-1} to denote the quantity $a^{-1} \pmod{p^2}$. We know $\gcd(a^{-1}, p) = 1$, because a^{-1} has an inverse $\pmod{p^2}$ too. Combining this with the fact that $\gcd(x,p) = 1$, we have $\gcd(a^{-1}x, p) = 1$. This tells us $a^{-1}x \in S$, so $a(a^{-1}x) = x \in T$.
- $T \subseteq S$: Let $ax \in T$, where $x \in S$. We know $\gcd(x,p) = 1$ because $x \in S$. Since $\gcd(a,p) = 1$ as well, we know the product xs cannot share any prime factors with p as well, i.e. $\gcd(xs, p) = 1$. This means $xs \in S$ as well, which proves the containment.

We now follow the proof of Fermat's Little Theorem. Since $S = T$, we have:

$$\prod_{s_i \in S} s_i \equiv \prod_{t_i \in T} t_i \pmod{p^2}$$

However, since we defined $T = \{a, 2a, \ldots, (p-1)a, (p+1)a, \ldots, (p^2-1)a\}$:

$$\prod_{t_i \in T} t_i \equiv \prod_{s_i \in S} a s_i \equiv a^{|S|} \prod_{s_i \in S} s_i \pmod{p^2}$$

We can now conclude $(\prod_{s_i \in S} s_i) \equiv a^{|S|} (\prod_{s_i \in S} s_i) \pmod{p^2}$.
Each $s_i \in S$ is coprime to p, so their product $\prod_{s_i \in S} s_i$ is as well. Then, we can multiply both sides of our equivalence with the inverse of $\prod_{s_i \in S} s_i$ to obtain $a^{|S|} \equiv 1 \pmod{p^2}$. Since $|S| = p(p-1)$, we have gotten the desired result.

Alternate Solution: We can use Fermat's Little Theorem, combined with the Binomial Theorem, to get the result. Since $\gcd(a,p) = 1$ and p is prime, $a^{p-1} \equiv 1 \pmod{p}$, so we can write $a^{p-1} = \ell p + 1$ for some integer ℓ. Then,

$$(a^{p-1})^p = (\ell p + 1)^p = \sum_{i=0}^{p} \binom{n}{i} (\ell p)^i = 1 + p \cdot (\ell p) + \binom{p}{2} (\ell p)^2 + \cdots + (\ell p)^p,$$

and since all of the terms other than the first term are divisible by p^2, $a^{p(p-1)} \equiv 1 \pmod{p^2}$.

(b) By the definition of d above, $ed = 1 + kp(p-1)q(q-1)$ for some k. Look at the equation $x^{ed} \equiv x \pmod{N}$ modulo p^2 first:

$$x^{ed} \equiv x^{1+kp(p-1)q(q-1)} \equiv x \cdot (x^{p(p-1)})^{kq(q-1)} \equiv x \pmod{p^2}$$

where we used the identity above. If we look at the equation modulo q^2, we obtain the same result. Hence, $x^{ed} \equiv x \pmod{p^2q^2}$.

Remark: The first part of the question mirrors the proof of Fermat's Little Theorem. The second and third parts of the question mirror the proof of correctness of RSA.

5 Polynomials over Galois Fields

Real numbers, complex numbers, and rational numbers are all examples of *fields*. A field is a set of numbers that has some nice properties over some operations. Galois fields are fields with only a finite number of elements, unlike fields such as the real numbers. Galois fields are denoted by GF(q), where q is the number of elements in the field.

(a) In the field GF(p), where p is a prime, how many roots does $q(x) = x^p - x$ have? Use this fact to express $q(x)$ in terms of degree one polynomials. Justify your answers.

(b) Prove that in GF(p), where p is a prime, whenever $f(x)$ has degree $\geq p$, it is equivalent to some polynomial $\tilde{f}(x)$ with degree $< p$.

(c) Show that if P and Q are polynomials over the reals (or complex numbers, or rationals) and $P(x)Q(x) = 0$ for all x, then either $P(x) = 0$ for all x, $Q(x) = 0$ for all x, or both. (*Hint*: You may want to prove first this lemma, which is true for all fields: The roots of $R(x) = P(x)Q(x)$ are the union of the roots of P and Q.)

(d) Show that the claim in part (c) is false for finite fields GF(p), where p is a prime.

Solution:

(a) We can factor $q(x) = x^p - x$ into $x(x^{p-1} - 1)$. By Fermat's Little Theorem, for any $a \in \{1, 2, \ldots p-1\}$, $q(a) = a(a^{p-1} - 1) = a(1-1) = 0$. And $q(0) = 0(0^{p-1} - 1) = 0$. So every element of $GF(p)$ is a root. The polynomial $q(x)$ has p roots.

We can write $q(x)$ as a product of its roots:

$$q(x) = \prod_{k=0}^{p-1} (x - k)$$

(b) One proof uses Fermat's Little Theorem. As a warm-up, let $d \geq p$; we'll find a polynomial equivalent to x^p. For any integer, we know

$$\begin{aligned} a^d &= a^{d-p} a^p \\ &\equiv a^{d-p} a \mod p \\ &\equiv a^{d-p+1} \mod p. \end{aligned}$$

In other words x^d is equivalent to the polynomial $x^{d-(p-1)}$. If $d - (p-1) \geq p$, we can show in the same way that x^d is equivalent to $x^{d-2(p-1)}$. Since we subtract $p-1$ every time, the sequence $d, d-(p-1), d-2(p-1), \ldots$ must eventually be smaller than p. Now if $f(x)$ is any polynomial with degree $\geq p$, we can apply this same trick to every x^k that appears for which $k \geq p$.

Another proof uses Lagrange interpolation. Let $f(x)$ have degree $\geq p$. By Lagrange interpolation, there is a unique polynomial $\tilde{f}(x)$ of degree at most $p-1$ passing through the points $(0, f(0)), (1, f(1)), (2, f(2)), \ldots, (p-1, f(p-1))$, and we designed it exactly so that it would be equivalent to $f(x)$.

(c) First, notice that if r is a root of P such that $P(r) = 0$, then r must also be a root of R, since $P(r)Q(r) = 0 \cdot Q(r) = 0$. The same is true for any roots of Q. Also notice that if some value s is neither a root of P nor Q, such that $P(s) \neq 0$ and $Q(s) \neq 0$, then s cannot be a root of R since $P(s)Q(s) \neq 0$. We therefore conclude that the roots of R are the union of the roots of P and Q.

Now we will show the contrapositive. Suppose that P and Q are both non-zero polynomials of degree d_P and d_Q respectively. Then $P(x) = 0$ for at most d_P values of x and $Q(x) = 0$ for at most d_Q values of x. This implies that R has at most $d_P + d_Q$ roots. Since there are an infinite number of values for x (because we are using complex, real, or rational numbers) we can always find an x, call it $x_{\text{not zero}}$, for which $P(x_{\text{not zero}}) \neq 0$ and $Q(x_{\text{not zero}}) \neq 0$. This gives us $P(x_{\text{not zero}})Q(x_{\text{not zero}}) \neq 0$ so R is non-zero.

(d) In GF(p), $x^{p-1} - 1$ and x are both non zero polynomials, but their product, $x^p - x$ is zero for all x by Fermat's Little Theorem.

Examples for a specific p are also acceptable. For example, for GF(2), $P(x) = x$ and $Q(x) = x + 1$.

6 Packet Requirements

In class, we learned that $n + 2k$ packets are required to protect against general errors when using polynomial interpolation methods. The Berlekamp-Welch algorithm provides an efficient method to recover the original message using only $n + 2k$ packets.

Alice is sending Bob a message of length n on a channel with k general errors by interpolating a polynomial through n points. Unfortunately, Bob hasn't watched the lecture on Error-Correcting Codes yet! He only knows about polynomial interpolation. Bob realizes that he needs to determine n points that are uncorrupted, which he can then use to interpolate the polynomial and recover Alice's original message.

Bob decides that he will interpolate polynomials through different combinations of n packets. Then he will compare the polynomials on their respective remaining additional packets to determine which polynomial was interpolated on uncorrupted points.

(a) Prove that with Bob's scheme there is not enough information to recover the original message when fewer than $n + 2k$ packets are sent. You may assume that Bob knows the length of the message, n, and the number of general errors, k. (*Hints*: How can Bob compare the polynomials on the leftover points? What would it mean for there not be enough information to recover the original message?)

(b) Prove that with Bob's scheme there is enough information to recover the original message when $n + 2k$ packets or more are sent. You may again assume that Bob knows the length of the message, n, and the number of general errors, k.

Solution:

(a) Using Bob's scheme, we must show that he won't be able to tell the difference between a polynomial interpolated on corrupted packets and a polynomial interpolated on uncorrupted points. We will show that when $n+2k-1$ packets are sent, Bob will not be able to differentiate between a corrupted polynomial and an uncorrupted polynomial.

Suppose Alice sends $n+2k-1$ packets. Then there are $n+k-1$ uncorrupted packets and k corrupted packets.

Suppose Bob correctly selects n uncorrupted packets to interpolate his polynomial. Then, of the $2k-1$ leftover packets, $k-1$ of them are uncorrupted and k of them are corrupted. So Bob's polynomial will go through $k-1$ of the additional packets.

Suppose Bob selects $c \in \{1,\dots n\}$ corrupted packets and $n-c$ uncorrupted packets to interpolate his polynomial. We will assume the corrupted packets are *a*dversarially corrupted; that is, we assume that they are corrupted so as to make it as hard for Bob as possible. Will Bob be able to tell he has a corrupt polynomial? Of the $2k-1$ additional packets, $k-1+c$ are uncorrupted and $k-c$ are corrupted. Note that the corrupted polynomial can pass through at most $n-1$ packets of the original polynomial; if it passed through all n packets, it would in fact be the correct polynomial. Since the corrupted polynomial already passes through $n-c$ uncorrupted packets, it can pass through at most $(n-1)-(n-c) = c-1$ remaining uncorrupted packets. It can also pass through the leftover $k-c$ corrupted packets. So the corrupted polynomial will pass through $(k-c)+(c-1) = k-1$ of the leftover packets.

Whether Bob picks a corrupted polynomial or a correct one, it's possible for the corrupted polynomial to pass through the same number of leftover packets! So if Bob finds a polynomial that passes through $k-1$ of the leftover packets, he can't be sure he has the correct polynomial.

(b) We use a similar argument to the one we gave in part (a).

Suppose Bob correctly selects n uncorrupted points to interpolate his polynomial. Then there will be $2k$ packets leftover, k of which will be uncorrupted and k of which will be corrupted. So Bob's polynomial will go through k of the leftover packets.

Suppose Bob selects $c \in \{1,\dots n\}$ corrupted packets and $n-c$ uncorrupted packets to interpolate his polynomial. We again assume that the packets are adversarially corrupted to make it as hard as possible for Bob. Of the $2k$ leftover packets, $k+c$ are uncorrupted and $k-c$ are corrupted. By the same argument as in (a), this corrupted polynomial can pass through $c-1$ of the leftover uncorrupted packets and all $k-c$ of the corrupted packets, so a total of $k-1$ of the leftover packets.

So if Bob picks a corrupted polynomial, it will go through $k-1$ of the leftover points in the worst case. However, if Bob picks the correct polynomial, it will go through k of the leftover points, and Bob will know it's correct.

Long story short, Bob should have just watched the lecture and done it the easy way; while this method works, its runtime is absolutely terrible compared to Berlekamp-Welch.

7 Alice and Bob

(a) Alice decides that instead of encoding her message as the values of a polynomial, she will encode her message as the coefficients of a degree 2 polynomial $P(x)$. For her message $[m_1, m_2, m_3]$, she creates the polynomial $P(x) = m_1x^2 + m_2x + m_3$ and sends the five packets $(0,P(0))$, $(1,P(1))$, $(2,P(2))$, $(3,P(3))$, and $(4,P(4))$ to Bob. However, one of the packet y-values is changed by Eve before it reaches Bob. If Bob receives

$$(0,1),(1,3),(2,0),(3,1),(4,0)$$

and knows Alice's encoding scheme and that Eve changed one of the packets, can he recover the original message? If so, find it as well as the x-value of the packet that Eve changed. If he can't, explain why. Work in mod 7.

(b) Bob gets tired of decoding degree 2 polynomials. He convinces Alice to encode her messages on a degree 1 polynomial. Alice, just to be safe, continues to send 5 points on her polynomial even though it is only degree 1. She makes sure to choose her message so that it can be encoded on a degree 1 polynomial. However, Eve changes two of the packets. Bob receives $(0,5)$, $(1,7)$, $(2,x)$, $(3,5)$, $(4,0)$. If Alice sent $(0,5)$, $(1,7)$, $(2,9)$, $(3,-2)$, $(4,0)$, for what values of x will Bob not uniquely be able to determine Alice's message? Assume that Bob knows Eve changed two packets. Work in mod 13.

(c) Alice wants to send a length 9 message to Bob. There are two communication channels available to her: Channel A and Channel B. When n packets are fed through Channel A, only 6 packets, picked at random, are delivered. Similarly, Channel B will only deliver 6 packets, picked at random, but it will also corrupt (change the value) of one of the delivered packets. Each channel will only work if at least 10 packets are sent through it. Using each of the two channels once, provide a way for Alice to send her message to Bob.

Solution:

(a) We can use Berlekamp and Welch. We have: $Q(x) = P(x)E(x)$. $E(x)$ has degree 1 since we know we have at most 1 error. $Q(x)$ is degree 3 since $P(x)$ is degree 2. We can write a system of linear equations and solve:

$$\begin{aligned} d &= 1(0-e) \\ a+b+c+d &= 3(1-e) \\ 8a+4b+2c+d &= 0(2-e) \\ 27a+9b+3c+d &= 1(3-e) \\ 64a+16b+4c+d &= 0(4-e) \end{aligned}$$

Since we are working in mod 7, this is equivalent to:

$$\begin{aligned} d &= -e \\ a+b+c+d &= 3-3e \\ a+4b+2c+d &= 0 \\ 6a+2b+3c+d &= 3-e \\ a+2b+4c+d &= 0 \end{aligned}$$

Solving yields:

$$Q(x) = x^3+5x^2+5x+4, E(x) = x-3$$

To find $P(x)$ we divide $Q(x)$ by $E(x)$ and get $P(x) = x^2+x+1$. So Alice's message is $m_1 = 1, m_2 = 1, m_3 = 1$. The x-value of the packet Eve changed is 3.

Alternative solution: Since we have 5 points, we have to find a polynomial of degree 2 that goes through 4 of those points. The point that the polynomial does not go through will be the packet that Eve changed. Since 3 points uniquely determine a polynomial of degree 2, we can pick 3 points and check if a 4th point goes through it. (It may be the case that we need to try all sets of 3 points.) We pick the points $(1,3),(2,0),(4,0)$. Lagrange interpolation can be used to create the polynomial but we can see that for the polynomial that goes through these 3 points, it has 0s at $x = 2$ and $x = 4$. Thus the polynomial is $k(x-2)(x-4) = k(x^2-6x+8) \pmod 7 \equiv k(x^2+x+1) \pmod 7$. We find $k \equiv 1$ by plugging in the point $(1,3)$, so our polynomial is x^2+x+1. We then check to see if the this polynomial goes through one of the 2 points that we didn't use. Plugging in 0 for x, we get 1. The packet that Eve changed is the point that our polynomial does not go through which has x-value 3. Alice's original message was $m_1 = 1, m_2 = 1, m_3 = 1$.

(b) Since Bob knows that Eve changed 2 of the points, the 3 remaining points will still be on the degree 1 polynomial that Alice encoded her message on. Thus if Bob can find a degree 1 polynomial that passes through at least 3 of the points that he receives, he will be able to uniquely recover Eve's message. The only time that Bob cannot uniquely determine Alice's message is if there are 2 polynomials with degree 1 that pass through 3 of the 5 points that he receives. Since we are working with degree 1 polynomials, we can plot the points that Bob receives and then see which values of x will cause 2 sets of 3 points to fall on a line. $(0,5),(1,7),(4,0)$ already fall on a line. If $x = 6$, $(1,7),(2,6),(3,5)$ also falls on a line. If $x = 5$, $(0,5),(2,5),(3,5)$ also falls on a line. If $x = 9$, $(0,5),(2,9),(4,0)$ falls on the original line, so here Bob can decode the message. If $x = 10$, $(2,10),(3,5),(4,0)$ also falls on a line. So if $x = 6,5,10$, Bob will not be able to uniquely determine Alice's message.

(c) Channel A will deliver 6 packets so we can send a message of length 6 encoded on a polynomial of degree 5 though it. If we send 10 points though channel A, it doesn't matter which 6 points Bob gets, he will still be able to reconstruct our degree 5 polynomial. Since the channel B has 1 general error, we can only send a message of length 4 encoded on a degree 3 polynomial through it. If we send 10 points, Bob will get 6 points to calculate a degree 4 polynomial

with 1 general error, which he is able to do. Thus to send our length 8 message, we can send the character 1 - 6 through a channel A and the characters 7 - 9 through channel B.

Alternative Solution: Alice can interpolate a polynomial of degree 8 encoding the message of length 9. She sends 10 points from that polynomial through channel A and another 10 points from the same polynomial through channel B. Bob will receive 6 points from channel A and 6 points from channel B, with one of them corrupted. He can use Berlekamp-Welch with $n = 9$ and $k = 1$ to recover the original polynomial. He retrieves the message by evaluating the polynomial on relevant points.

8 Homework Process and Study Group

You must describe your homework process and study group in order to receive credit for this question.