CS 70 Discrete Mathematics and Probability Theory
Spring 2021 HW 5

1 Lagrange? More like Lamegrange.

In this problem, we walk you through an alternative to Lagrange interpolation.

(a) Let's say we wanted to interpolate a polynomial through a single point, (x_0, y_0). What would be the polynomial that we would get? (This is not a trick question.)

(b) Call the polynomial from the previous part $f_0(x)$. Now say we wanted to define the polynomial $f_1(x)$ that passes through the points (x_0, y_0) and (x_1, y_1). If we write $f_1(x) = f_0(x) + a_1(x - x_0)$, what value of a_1 causes $f_1(x)$ to pass through the desired points?

(c) Now say we want a polynomial $f_2(x)$ that passes through (x_0, y_0), (x_1, y_1), and (x_2, y_2). If we write $f_2(x) = f_1(x) + a_2(x - x_0)(x - x_1)$, what value of a_2 gives us the desired polynomial?

(d) Suppose we have a polynomial $f_i(x)$ that passes through the points (x_0, y_0), ..., (x_i, y_i) and we want to find a polynomial $f_{i+1}(x)$ that passes through all those points and also (x_{i+1}, y_{i+1}). If we define $f_{i+1}(x) = f_i(x) + a_{i+1} \prod_{j=0}^{i} (x - x_j)$, what value must a_{i+1} take on?

Solution:

(a) We want a degree zero polynomial, which is just a constant function. The only constant function that passes through (x_0, y_0) is $f_0(x) = y_0$.

(b) By defining $f_1(x) = f_0(x) + a_1(x - x_0)$, we get that

$$f_1(x_0) = f_0(x_0) + a_1(x_0 - x_0) = y_0 + 0 = y_0.$$

So now we just need to make sure that $f_1(x_1) = y_1$. This means that we need to choose a_1 such that

$$f_1(x_1) = f_0(x_1) + a_1(x_1 - x_0) = y_1.$$

Solving this for a_1, we get that

$$a_1 = \frac{y_1 - f_0(x_1)}{x_1 - x_0}.$$

(c) We apply similar logic to the previous part. From our definition, we know that

$$f_2(x_0) = f_1(x_0) + a_2(x_0 - x_0)(x_0 - x_1) = y_0 + 0 = y_0.$$

and that

$$f_2(x_1) = f_1(x_1) + a_2(x_1 - x_0)(x_1 - x_1) = y_1 + 0 = y_1.$$

Thus, we just need to choose a_2 such that $f_2(x_2) = y_2$. Putting in our formula for $f_2(x)$, we get that we need a_2 such that

$$f_1(x_2) + a_2(x_2 - x_0)(x_2 - x_1) = y_2.$$

Solving for a_2, we get that

$$a_2 = \frac{y_2 - f_1(x_2)}{(x_2 - x_0)(x_2 - x_1)}.$$

(d) If we try to calculate $f_{i+1}(x_k)$ for $0 \leq k \leq i$, we know one of the $(x - x_j)$ terms (specifically the kth one) will be zero. Thus, we get that

$$f_{i+1}(x_k) = f_i(x_k) + a_{i+1}(0) = y_k + 0 = y_k.$$

So now we just need to pick a_i such that $f_{i+1}(x_{i+1}) = y_{i+1}$. This means that we need to choose a_{i+1} such that

$$f_i(x_{i+1}) + a_{i+1} \prod_{j=0}^{i} (x_{i+1} - x_j) = y_{i+1}.$$

Solving for a_{i+1}, we get that

$$a_{i+1} = \frac{y_{i+1} - f_i(x_{i+1})}{\prod_{j=0}^{i}(x_{i+1} - x_j)}.$$

The method you derived in this question is known as Newtonian interpolation. (The formal definition of Newtonian interpolation uses divided differences, which we don't cover in this class, but it's in effect doing the same thing.) This method has an advantage over Lagrange interpolation in that it is very easy to add in extra points that your polynomial has to go through (as we showed in part (c), whereas Lagrange interpolation would require you to throw out all your previous work and restart. However, if you want to keep the same x values but change the y values, Newtonian interpolation requires you to throw out all your previous work and restart. In contrast, this is fairly easy to do with Lagrange interpolation–since changing the y values doesn't affect the δ_is, you don't have to recalculate those, so you can skip most of the work.

2 Polynomials in Fields

Define the sequence of polynomials by $P_0(x) = x + 12$, $P_1(x) = x^2 - 5x + 5$ and $P_n(x) = xP_{n-2}(x) - P_{n-1}(x)$.

(For instance, $P_2(x) = 17x - 5$ and $P_3(x) = x^3 - 5x^2 - 12x + 5$.)

(a) Show that $P_n(7) \equiv 0 \pmod{19}$ for every $n \in \mathbb{N}$.

(b) Show that, for every prime q, if $P_{2017}(x) \not\equiv 0 \pmod{q}$, then $P_{2017}(x)$ has at most 2017 roots modulo q.

Solution:

(a) Prove by strong induction. Base cases:

$$P_0(7) \equiv 7+12 \equiv 19 \equiv 0 \pmod{19}$$
$$P_1(7) \equiv 7^2 - 5\cdot 7 + 5 \equiv 49 - 35 + 5 \equiv 19 \equiv 0 \pmod{19}$$

Inductive step: Assume $P_n(7) \equiv 0 \pmod{19}$ for every $n \leq k$. Then

$$\begin{aligned} P_{k+1}(7) &\equiv xP_{k-1}(7) - P_k(7) \pmod{19} \\ &\equiv x\cdot 0 - 0 \pmod{19} \\ &\equiv 0 \pmod{19}. \end{aligned}$$

Hence, we have $P_n(7) \equiv 0 \pmod{19}$ for all naturnal numbers n.

(b) This question asks to prove that, for all prime numbers q, if $P_{2017}(x)$ is a non-zero polynomial $\pmod{q}$, then $P_{2017}(x)$ has at most 2017 roots $\pmod{q}$.

The proof of Property 1 of polynomials (a polynomial of degree d can have at most d roots) still works in the finite field GF(q). Therefore we need only show that P_{2017} has degree at most 2017. We prove that $\deg(P_n) \leq n$ for $n > 1$ by strong induction. Base cases:

$$\begin{aligned} \deg(P_0) &= \deg(x+12) = 1 \\ \deg(P_1) &= \deg(x^2 - 5x + 5) = 2 \\ \deg(P_2) &= \deg(xP_0(x) - P_1(x)) \leq 2 \\ \deg(P_3) &= \deg(xP_1(x) - P_2(x)) \leq 3 \end{aligned}$$

Assuming degree of $P_n \leq n$ for all $2 \leq n \leq k$, then

$$\begin{aligned} \deg(P_{k+1}(x)) &\leq \max\{\deg(xP_{k-1}(x)), \deg(P_k(x))\} \\ &= \max\{1 + \deg(P_{k-1}(x)), \deg(P_k(x))\} \\ &\leq \max\{1 + k - 1, k\} \\ &\leq k \\ &\leq k+1. \end{aligned}$$

Thus the proof holds for all $n \geq 2, n \in \mathbb{N}$.

3 Equivalent Polynomials

This problem is about polynomials with coefficients in GF(q) for some prime $q \in \mathbb{N}$. We say that two such polynomials f and g are *equivalent* if $f(x) = g(x)$ for every $x \in$ GF(q).

(a) Use Fermat's Little Theorem to find a polynomial equivalent to $f(x) = x^5$ over GF(5); then find one equivalent to $g(x) = 1 + 3x^{11} + 7x^{13}$ over GF(11).

(b) Prove that whenever $f(x)$ has degree $\geq q$, it is equivalent to some polynomial $\tilde{f}(x)$ with degree $< q$.

Solution:

(a) Fermat's Little Theorem says that for any nonzero integer a and any prime number q, $a^{q-1} \equiv 1 \mod q$. We're allowed to multiply through by a, so the theorem is equivalent to saying that $a^q \equiv a \mod q$; note that this is true even when $a = 0$, since in that case we just have $0^q \equiv 0 \mod q$. The problem asks for a polynomial $\tilde{f}(x)$, different from $f(x)$, with the property that $\tilde{f}(a) \equiv a^5 \mod 5$ for any integer a. Directly using the theorem, $\tilde{f}(x) = x$ will work. We can do something similar with $g(x) = 1 + 3x^{11} + 7x^{13}$ modulo 11: set $\tilde{g}(x) = 1 + 3x + 7x^3$.

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

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

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

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

4 Secret Sharing with Spies

An officer stored an important letter in her safe. In case she is killed in battle, she decides to share the password (which is a number) with her troops. However, everyone knows that there are 3 spies among the troops, but no one knows who they are except for the three spies themselves. The 3 spies can coordinate with each other and they will either lie and make people not able to open the safe, or will open the safe themselves if they can. Therefore, the officer would like a scheme to share the password that satisfies the following conditions:

- When M of them get together, they are guaranteed to be able to open the safe even if they have spies among them.
- The 3 spies must not be able to open the safe all by themselves.

Please help the officer to design a scheme to share her password. What is the scheme? What is the smallest M? Show your work and argue why your scheme works and any smaller M couldn't work. (The troops only have one chance to open the safe; if they fail the safe will self-destruct.)

Solution:

The key insight is to realize that both polynomial-based secret-sharing and polynomial-based error correction work on the basis of evaluating an underlying polynomial at many points and then trying to recover that polynomial. Hence they can be easily combined.

Suppose the password is s. The officer can construct a polynomial $P(x)$ such that $s = P(0)$ and share $(i, P(i))$ to the i-th person in her troops. Then the problem is: what should the degree of $P(x)$ be and what is the smallest M?

First, the degree of polynomial d should not be less than 3. It is because when $d < 3$, the 3 spies can decide the polynomial $P(x)$ uniquely. Thus, n will be at least 4 symbols.

Let's choose a polynomial $P(x)$ of degree 3 such that $s = P(0)$. We now view the 3 spies as 3 general errors. Then the smallest $M = 10$ since n is at least 4 symbols and we have $k = 3$ general errors, leading us to a "codeword" of $4 + 2 \cdot 3 = 10$ symbols (or people in our case). Even though the 3 spies are among the 10 people and try to lie on their numbers, the 10 people can still be able to correct the $k = 3$ general errors by the Berlekamp-Welch algorithm and find the correct $P(x)$.

Alternative solution:

Another valid approach is making $P(x)$ of degree $M-1$ and adding 6 public points to deal with 3 general errors from the spies. In other words, in addition to their own point $(i, P(i))$, everyone also knows the values of 6 more points, $(t+1, P(t+1)), (t+2, P(t+2)), \ldots, (t+6, P(t+6))$, where t is the number of the troops. The spies have access to total of $3+6 = 9$ points so the degree $M-1$ must be at least 9 to prevent the spies from opening the safe by themselves. Therefore, the minimum M is 10.

5 Trust No One

Gandalf has assembled a fellowship of eight peoples to transport the One Ring to the fires of Mount Doom: four hobbits, two humans, one elf, and one dwarf. The ring has great power that may be of use to the fellowship during their long and dangerous journey. Unfortunately, the use of its immense power will eventually corrupt the user, so it must not be used except in the most dire of circumstances. To safeguard against this possibility, Gandalf wishes to keep the instructions a secret from members of the fellowship. The secret must only be revealed if enough members of the fellowship are present and agree to use it.

Requiring all eight members to agree is certainly a sufficient condition to know the instructions, but it seems excessive. However, we also know that the separate peoples (hobbits, humans, elf, and dwarf) do not completely trust each other so instead we decide to require members from at least two different peoples in order to use the ring. In particular, we will require a unanimous decision by all members of one race in addition to at least one member of a different people. That is, if only the four hobbits want to use the ring, then they alone should not have sufficient information to figure out the instructions. Same goes for the two humans, the elf, and the dwarf.

More explicitly, some examples: only four hobbits agreeing to use the ring is not enough to know the instructions. Only two humans agreeing is not enough. Only the elf agreeing is not enough. Only the dwarf agreeing is not enough. All four hobbits and a man agreeing is enough. Both humans and a dwarf agreeing is enough. Both the elf and the dwarf agreeing is enough.

Gandalf has hired your services to help him come up with a secret sharing scheme that accomplishes this task, summarized by the following points:

- There is a party of four hobbits, two humans, an elf, and a dwarf.
- There is a secret message that needs to be known if enough members of the party agree.
- The message must remain unknown to everyone (except Gandalf) if not enough members of the party agree.
- If only the members of one people agree, the message remains a secret.
- If all the members of one people agree plus at least one additional person, the message can be determined.

Solution:

Solution 1

There will be two parts to this secret: a unanimity secret U and a multi-people secret M. U ensures that at least all members of one peoples are in agreement while M ensures that members of at least two peoples are in agreement. We will discuss these two in order below. Once both U and M are recovered, they can then be combined to reveal the original secret: each will be a point of the degree-1 polynomial $R(x)$ whose y-intercept contains the secret of the ring.

The *unanimity secret* involves creating a separate secret for each people. We will require all members of that people to join forces in order to reveal the secret. For example, the hobbits will each have distinct points of a degree-3 polynomial and the humans will each have distinct points of a degree-1 polynomial. When all members of a people come together, they will reveal U (encoded, for example, as the y-intercept of each of these polynomials). Note that the elf and the dwarf each know U already since they are the only members of their people.

The *multi-people secret* involves creating a degree-1 polynomial $P_m(x)$ and giving one point to all members of each people. For example, the hobbits may each get $P_m(1)$ while the elf gets $P_m(2)$ and the humans each get $P_m(3)$. In this way if members of any two peoples are in agreement, they can reveal M (encoded, for example, as the y-intercept of $P_m(x)$).

Once U and M are each known, they can be *combined* to determine the final secret. U and M allow us to uniquely determine $R(x)$ and thus $R(0)$, the secret of the ring.

This scheme is an example of hierarchical secret sharing. Let's work out a specific example.

Example: Suppose the secret is $s = 4$, $M = 3$, and $U = 2$. From now on, we can work in GF(5) since $s < 5$ and $n < 5$ (n is the number of people who have pieces of the secret).

First we need to create a degree-1 polynomial $R(x)$ such that $R(0) = s = 4$, $R(1) = M = 3$, and $R(2) = U = 2$. By inspection, $R(x) = 4x + 4$ has these properties (e.g. $R(1) = 4 \cdot 1 + 4 = 8 \equiv 3$).

Now we can create the multi-people secret M. We choose degree-1 polynomial $P_m(x) = x+3$ and tell each hobbit $P_m(1) = 4$, the elf $P_m(2) = 5 \equiv 0$, each of the humans $P_m(3) = 6 \equiv 1$, and the dwarf $P_m(4) = 7 \equiv 2$. Now any two members of distinct peoples can determine $P_m(x)$ and thus $P_m(0)$ by interpolating their two values.

When creating the unanimity secret U, we first note that each of the dwarf and the elf will be told U directly since they are the only members of their respective people. On the other hand, the humans will each have a point on the degree-1 polynomial $P_{humans}(x)$. Suppose $P_{humans}(x) = 2x+2$. Then the first human receives $P_{humans}(1) = 4$ and the second receives $P_{humans}(2) = 4+2 = 6 \equiv 1$. When they interpolate using these values, they will discover the original polynomial and therefore $P_{humans}(0) = U = 2$. The hobbits will have a similar secret but with a degree-3 polynomial (e.g. $P_{hobbit}(x) = 4x^3 + x^2 + 2$).

Now suppose that two humans and one hobbit come together. The two humans work together to determine U as described above. Together the three of them also know $P_m(3) = 6$ and $P_m(1) = 4$, from which they can find $P_m(x)$ and thus $P_m(0) = M = 3$. Now that they have U and M, they can interpolate to find $R(x)$ and thus $R(0) = s = 4$.

Solution 2

Alternatively, we can construct a single degree 6 polynomial and distribute 1 point to each hobbit, 3 points to each human, 6 points to the elf, and 6 points to the dwarf. We can see that if all the hobbits agree, they will need 3 more points in order to interpolate successfully and each member of all the other races are given at least 3 points. Moreover, each of the other races have 6 points in total, meaning that if all the humans, the elf, or the dwarf agree, they'll only need one more point which can be provided by any additional member of the party outside the race. On the other hand, the most amount of points that could be obtained from an agreeing group that does not satisfy the requirements would be 6, from the group consisting of one human and all the hobbits. This would be insufficient to interpolate the polynomial so therefore, the scheme fulfills the requirements.

6 Green Eggs and Hamming

The *Hamming distance* between two length-n bit strings b_1 and b_2 is defined as the minimum number of bits in b_1 you need to flip in order to get b_2. For example, the Hamming distance between 101 and 001 is 1 (since you can just flip the first bit), while the Hamming distance between 111 and 000 is 3 (since you need to flip all three bits).

(a) Sam-I-Am has given you a list of n situations, and wants to know in which of them you would like green eggs and ham. You are planning on sending him your responses encoded in a length n bit string (where a 1 in position i says you would like green eggs and ham in situation i, while a 0 says you would not), but the channel you're sending your answers over is noisy and sometimes corrupts a bit. Sam-I-Am proposes the following solution: you send a length $n+1$ bit string, where the $(n+1)$st bit is the XOR of all the previous n bits (this extra bit is called the parity bit). If you use this strategy, what is the minimum Hamming distance between any two valid bit strings you might send? Why does this allow Sam-I-Am to detect an error? Can

he correct the error as well?

(b) If the channel you are sending over becomes more noisy and corrupts two of your bits, can Sam-I-Am still detect the error? Why or why not?

(c) If you know your channel might corrupt up to k bits, what Hamming distance do you need between valid bit strings in order to be sure that Sam-I-Am can detect when there has been a corruption? Prove as well that that your answer is tight—that is, show that if you used a smaller Hamming distance, Sam-I-Am might not be able to detect when there was an error.

(d) Finally, if you want to *correct* up to k corrupted bits, what Hamming distance do you need between valid bit strings? Prove that your condition is sufficient.

Solution:

(a) The minimum Hamming distance is 2. In order to prove this, we need to show both that there exits two valid strings we could send that have a Hamming distance of 2 from eachother, and that there are no valid strings we could send that are at Hamming distance 1. The former is easy: if we do not like green eggs and ham in any situation, our answer to Sam-I-Am would just be $00\ldots0$. Since the XOR of n zeros is 0, our $(n+1)$st digit would also be a zero, so we would send a message containing $n+1$ zeros. In addition, if we only like green eggs and ham in the nth situation, but not any of the previous ones, our answer to Sam-I-Am would be $00\ldots01$, meaning that our message would be $n-1$ zeros followed by two ones (since the XOR of $0\cdots010\cdots0$ is one). These two bit strings are at Hamming distance 2 from one another, which is what we wanted to show.

To show that there is no pair of valid strings at Hamming distance 1, there are two cases to consider: either the bit that is different between them is in the first n, or it is the last (parity) bit. In the former case, flipping one of the n bits in our answer also flips the XOR of all our bits, meaning that the parity bit would have to change. However, both strings have to have the same parity bit, so one of the strings must have an incorrect parity bit and thus be invalid. In the second case, we know that the two strings share the same first n bits, but have different parity bits. However, the parity bit is uniquely determined by the first n bits, so one of the two strings must have an incorrect parity bit. Thus, in either case, one of the two strings must be invalid, so we can conclude that now two valid strings can have Hamming distance 2.

Finally, why is this relevant to error detecting? Let's call the list of possible strings you could send to Sam-I-Am your *codebook*. If no two strings in your codebook are within Hamming distance 2 of one another, then upon receiving a string with at most one bit flipped, Sam-I-Am can detect the error: if a bit was flipped, the new string is at Hamming distance 1 from the intended string, and thus is no longer in the codebook.

(b) Sam-I-Am cannot necessarily still detect the error. Because two valid strings can be a Hamming distance of two from eachother, it is possible that when the channel flips two bits, we end up at another valid string. As an example, say $n = 3$, Sam-I-Am received the string 1100. This message could have either resulted from us sending 1100 and not having any bits flipped or

from sending 0000 and having the first two bits flipped. Thus, Sam-I-Am has no idea whether or not an error occurred.

(c) You need a Hamming distance of $k+1$ between valid strings. If no valid strings are closer than $k+1$, then there is no way to get from one valid code word to another by corrupting at most k bits. Thus, if Sam-I-Am gets a valid string, he can conclude that your message got through without error; otherwise, he has to ask you to send it again.

We also have to prove that this is tight, meaning that if our minimum Hamming distance was k or smaller, Sam-I-Am might not be able to detect an error. If there existed two valid strings b_1 and b_2 at a Hamming distance of k or less from each other and Sam-I-Am received b_1, he wouldn't know if we had sent b_1 and it had gotten through uncorrupted, or if we had tried to send b_2 and the proper bits had gotten flipped to make it look like b_1. Thus, if the minimum Hamming distance was k or less, Sam-I-Am wouldn't necessarily be able to detect whether or not error(s) occurred.

(d) You need a minimum Hamming distance of $2k+1$. The idea is that, upon receiving your possibly corrupted message, Sam-I-Am finds the string in the codebook closest in Hamming distance to the one he received. To prove that this works, we need to know that if we start at a bit string b and flip $\ell \leq k$ different bits, the Hamming-nearest string in our codebook is still b. Call the corrupted string b' and assume (looking for contradiction) that some string c in our codebook is closer to b' than b is to b'. This means c can be obtained from b' by flipping $m \leq \ell \leq k$ bits, and b' can be obtained from b by flipping $\ell \leq k$ bits. Concatenating these sequences of bit flips, we can get c from b by flipping no more than $m+\ell \leq 2k$ bits. But this means the Hamming distance between b and c is less than $2k+1$, and we assumed that the minimum such distance was $2k+1$.

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 arbitrarily, are delivered. Similarly, Channel B will only deliver 6 packets, picked arbitrarily, 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 so that he can always reconstruct it.

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 $k(x-2)(x-4) = k(x^2-6x+8)$

$(\text{mod } 7) \equiv k(x^2+x+1) \ (\text{mod } 7)$. 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.