

## Polynomials Intro

**Note 8** **Polynomial:**  $f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_nx^n$ ; in terms of roots,  $f(x) = a(x - r_1)(x - r_2) \cdots (x - r_k)$

**Degree of a polynomial:** the highest exponent in the polynomial

**Galois Field:** denoted as  $\text{GF}(p)$ , it's basically just a fancy way of saying that we're working modulo  $p$ , for a prime  $p$

**Properties** (true over  $\mathbb{R}$  and also over  $\text{GF}(p)$ ):

- Polynomial of degree  $d$  has at most  $d$  roots.
- Exactly one polynomial of degree at most  $d$  passes through  $d + 1$  points.

**Lagrange Interpolation:** Given  $d + 1$  points  $(x_1, y_1), (x_2, y_2), \dots, (x_{d+1}, y_{d+1})$ , we define

$$\Delta_i(x) = \frac{\prod_{j \neq i} (x - x_j)}{\prod_{j \neq i} (x_i - x_j)}, \quad i = 1, 2, \dots, d + 1.$$

The unique polynomial through all points is  $f(x) = \sum_{i=1}^{d+1} y_i \cdot \Delta_i(x)$ .

## 1 Polynomial Fill in the Blank

**Note 8** Fill in each blank, and provide a brief justification.

- A degree  $d$  nonzero polynomial over  $\mathbb{R}$  has at most \_\_\_\_ roots.
- A degree  $d$  nonzero polynomial over  $\text{GF}(p)$  has at most  $\min(\text{____}, p)$  roots.
- $d$  distinct points determine a unique polynomial of degree at most \_\_\_\_.
- For every prime  $p$ , every polynomial over  $\text{GF}(p)$  with degree  $\geq p$  is equivalent to a polynomial of degree at most \_\_\_\_\_. (Two polynomials are equivalent if they evaluate to the same value for every  $x \in \text{GF}(p)$ .)

### Solution:

- $d$ . Each root  $r$  gives a factor  $(x - r)$ , and a degree  $d$  polynomial has at most  $d$  linear factors.
- $d$ . The same factoring argument works over  $\text{GF}(p)$ ; the number of roots is also at most  $p$ , since  $\text{GF}(p)$  has only  $p$  elements.

- (c)  $d - 1$ . A polynomial of degree at most  $d - 1$  has  $d$  coefficients, and  $d$  points fix them (for example, by Lagrange interpolation).
- (d)  $p - 1$ . By Fermat's Little Theorem,  $x^p \equiv x \pmod{p}$  for every  $x$ , so any term  $x^k$  with  $k \geq p$  can be replaced by  $x^{k-(p-1)}$ . Repeating this brings every exponent down to at most  $p - 1$ .

## 2 How Many Polynomials?

**Note 8** Let  $P(x)$  be a polynomial of degree at most 2 over  $\text{GF}(5)$ . As we saw in lecture, we need  $d + 1$  distinct points to determine a unique polynomial of degree at most  $d$ , so knowing the values for say,  $P(0)$ ,  $P(1)$ , and  $P(2)$  would be enough to recover  $P$ . (For this problem, we consider two polynomials to be distinct if they return different values for some input.)

- (a) Assume that we know  $P(0) = 1$ , and  $P(1) = 2$ . Now consider  $P(2)$ . How many values can  $P(2)$  have? How many distinct possibilities for  $P$  do we have?
- (b) Now assume that we only know  $P(0) = 1$ . We consider  $P(1)$  and  $P(2)$ . How many different  $(P(1), P(2))$  pairs are there? How many distinct possibilities for  $P$  do we have?
- (c) Now, let  $P$  be a polynomial of degree at most  $d$  on  $\text{GF}(p)$  for some prime  $p$  with  $p > d$ . Assume we only know  $P$  evaluated at  $k \leq d + 1$  different values. How many different possibilities do we have for  $P$ ?

### Solution:

- (a) 5 polynomials, each for different values of  $P(2)$ .
- (b) Now there are  $5^2$  different polynomials.
- (c)  $p^{d+1-k}$  different polynomials. For  $k = d + 1$ , there should only be 1 polynomial.

## 3 Lagrange Interpolation in Finite Fields

**Note 8** In this problem, we will break down the terms of Lagrange interpolation by working through an example, where we want to find a unique polynomial  $p(x)$  of degree at most 2 that passes through points  $(-1, 3)$ ,  $(0, 1)$ , and  $(1, 2)$  in modulo 5 arithmetic.

- (a) First, assume we have polynomials  $\Delta_{-1}(x)$ ,  $\Delta_0(x)$ , and  $\Delta_1(x)$  satisfying the following congruences modulo 5:

$$\begin{array}{lll} \Delta_{-1}(-1) \equiv 1, & \Delta_{-1}(0) \equiv 0, & \Delta_{-1}(1) \equiv 0, \\ \Delta_0(-1) \equiv 0, & \Delta_0(0) \equiv 1, & \Delta_0(1) \equiv 0, \\ \Delta_1(-1) \equiv 0, & \Delta_1(0) \equiv 0, & \Delta_1(1) \equiv 1. \end{array}$$

Construct  $p(x)$  using a linear combination of  $\Delta_{-1}(x)$ ,  $\Delta_0(x)$ , and  $\Delta_1(x)$ .

(Note: Notice that for the  $\Delta_i$ 's in this question, the subscript  $i$  refers to an actual value of  $x$ .)

- (b) Find  $\Delta_{-1}(x)$ . In other words, find a degree 2 polynomial that has roots at  $x = 0$  and  $x = 1$  and evaluates to 1 at  $x = -1$  (all in modulo 5).
- (c) Find  $\Delta_0(x)$ .
- (d) Find  $\Delta_1(x)$ .
- (e) Now, let's put it all together! Find  $p(x)$  by using your linear combination in part (a) and your polynomials constructed above.
- (f) How does performing Lagrange interpolation over  $\text{GF}(p)$  differ from interpolation over  $\mathbb{R}$ ?

**Solution:**

- (a) We know that each respective  $\Delta_n(x)$  will be 1 when  $x = n$ , and 0 at the two other relevant points. Thus,  $p(x)$  can be created by a linear combination of  $\Delta_n(x)$ 's multiplied by the desired  $y$  value at  $x = n$ . This gives us  $p(x) = 3\Delta_{-1}(x) + 1\Delta_0(x) + 2\Delta_1(x)$ .

- (b) We see

$$\begin{aligned}\Delta_{-1}(x) &\equiv (x-0)(x-1)((-1-0)(-1-1))^{-1} \\ &\equiv (2)^{-1}x(x-1) \pmod{5} \\ &\equiv 3x(x-1) \pmod{5}.\end{aligned}$$

- (c) We see

$$\begin{aligned}\Delta_0(x) &\equiv (x+1)(x-1)((0+1)(0-1))^{-1} \\ &\equiv (-1)^{-1}(x-1)(x+1) \pmod{5} \\ &\equiv 4(x-1)(x+1) \pmod{5}.\end{aligned}$$

- (d) We see

$$\begin{aligned}\Delta_1(x) &\equiv (x+1)(x-0)((1+1)(1-0))^{-1} \\ &\equiv (2)^{-1}x(x+1) \pmod{5} \\ &\equiv 3x(x+1) \pmod{5}.\end{aligned}$$

- (e) Putting everything together,

$$\begin{aligned}p(x) &= 3\Delta_{-1}(x) + 1\Delta_0(x) + 2\Delta_1(x) \\ &= 9x(x-1) + 4(x-1)(x+1) + 6x(x+1) \\ &\equiv 4x^2 - 3x - 4 \pmod{5} \\ &\equiv 4x^2 + 2x + 1 \pmod{5}.\end{aligned}$$

- (f) The method is the same: the  $\Delta_i(x)$  and the linear combination have the same form, and the resulting polynomial is still unique, since  $\text{GF}(p)$  is a field. The difference is that all arithmetic is done modulo  $p$ . Instead of dividing by  $\prod_{j \neq i} (x_i - x_j)$ , we multiply by its inverse modulo  $p$ , which exists because  $p$  is prime and the  $x_i$  are distinct modulo  $p$ . As a result, all coefficients and values lie in  $\{0, 1, \dots, p-1\}$ , and we can interpolate through at most  $p$  points.

# Error Correcting Codes and Secret Sharing Intro

Note 8  
Note 9

**Secret Sharing:** We make use of the fact that there is a unique polynomial of degree  $d$  passing through a given set of  $d + 1$  points. This means that if we require  $k$  people to come together in order to find a secret, we should use a polynomial of degree  $k - 1$ , and give each person one point. There are more complicated schemes if there are more conditions, but they all use the same concept.

**Erasur errors:** A packet/point  $(x, P(x))$  is *lost* in the communication channel.

$$\begin{array}{|c|c|c|c|c|} \hline 1 & 5 & 3 & 4 & 3 \\ \hline \end{array} \longrightarrow \begin{array}{|c|c|c|c|c|} \hline 1 & 5 & & 4 & 3 \\ \hline \end{array}$$

*Protection:* ( $n$  packets,  $k$  errors) interpolate polynomial through the message points, send  $n + k$  packets on the polynomial

**General errors:** A packet is *modified* in the communication channel.

$$\begin{array}{|c|c|c|c|} \hline 1 & 5 & 3 & 4 \\ \hline \end{array} \longrightarrow \begin{array}{|c|c|c|c|} \hline 1 & 5 & 6 & 4 \\ \hline \end{array}$$

*Protection:* ( $n$  packets,  $k$  errors) interpolate polynomial through the message points, send  $n + 2k$  packets on the polynomial

## Berlekamp–Welch Algorithm:

Variables: sent message packets  $m_i$ , received packets  $r_i$ , error locations  $e_i$

Polynomials:

- $P(x)$ : original polynomial through message (this is what we want)
- $E(x) = (x - e_1)(x - e_2) \cdots (x - e_k)$ : error locator polynomial
- $Q(x) = P(x)E(x)$ , or  $Q(i) = P(i)E(i) = r_i E(i)$  for all  $i$

$Q(x)$  and  $E(x)$  are unknown, but we can solve for them using a system of equations.

## 4 Berlekamp-Welch Warm Up

### Note 9

Suppose you want to send a length- $n$  message over an unreliable channel. You obtain  $P(x)$ , the polynomial that passes through your message packets.

- (a) What is the degree of  $P(x)$ ? Why?
- (b) If there are at most  $k$  erasure errors, how many packets should you send? If there are at most  $k$  general errors, how many packets should you send?

Suppose the unreliable channel can corrupt up to  $k$  general errors, and you send an appropriate number of packets based on part (b). Your recipient uses Berlekamp-Welch to correct the general errors.

- (c) What do the roots of the error polynomial  $E(x)$  represent? Does your recipient know these roots?
- (d) When does  $r_i = P(i)$ ? When does  $r_i$  not equal  $P(i)$ ? Using these two cases, prove that  $Q(i) = P(i)E(i) = r_iE(i)$  for each received packet index  $i$ .
- (e) What is the maximum degree of  $E(x)$ ? Using the information about the degree of  $E(x)$  and  $P(x)$  from part (a), what is the maximum degree of  $Q(x)$ ?
- (f) How many unknown coefficients are there in total across  $Q(x)$  and  $E(x)$ ?

Upon knowing the degrees of  $Q(x)$  and  $E(x)$ , your recipient writes the equation  $Q(i) = r_iE(i)$  for each received packet.

- (g) How many equations does your recipient have? Do they have enough equations to solve for all of the unknown coefficients?
- (h) Your recipient obtains  $Q(x)$  and  $E(x)$ . How do they recover  $P(x)$  and the original message?

### Solution:

- (a)  $P$  has degree at most  $n - 1$ , since  $n$  points determine a unique polynomial of degree at most  $n - 1$ .
- (b) We send  $n + k$  packets when we have  $k$  erasures and  $n + 2k$  packets for  $k$  general errors.
- (c) The roots of error polynomial  $E(x)$  represent the locations of corrupted packets. The recipient does not know the roots of  $E(x)$ .
- (d)  $r_i = P(i)$  when the received packet is correct.  $r_i$  does not equal  $P(i)$  when the received packet is corrupted.

If there is no error at point  $i$ ,  $P(i) = r_i$  and then multiplying each side by  $E(i)$  gives  $P(i)E(i) = r_iE(i)$ . If there is an error at point  $i$ , then  $E(i) = 0$ , which means  $P(i)E(i) = r_iE(i) = 0$ .

- (e) Since there are at most  $k$  errors, the maximum degree of  $E(x)$  is  $k$ . The maximum degree of  $Q$  is  $(n - 1) + (k) = n + k - 1$  since the degree of  $P$  is  $n - 1$  and the degree of  $E$  is at most  $k$ .

- (f) The maximum degree of  $Q(x)$  is  $n+k-1$ , so the number of unknowns is  $n+k$ . The maximum degree of  $E(x)$  is  $k$ , which would mean there would be  $k+1$  unknowns. However, we know that the coefficient of  $x^k$  is 1 in  $E(x)$ , so the number of unknowns is  $k$ .

The total number of unknowns is  $(n+k) + (k) = n+2k$

- (g) There are  $n+2k$  equations, which is enough to solve for  $n+2k$  unknowns.
- (h) We can compute  $P(x)$  using the equation:  $P(x) = Q(x)/E(x)$ . To recover the message, we evaluate  $P$  at the  $n$  indices of the original message packets.

## 5 Berlekamp-Welch Algorithm

**Note 9** Steve wants to send you a message  $(m_0, m_1, m_2) = (1, 1, 4)$  of length  $n = 3$ , using an error-correcting code for  $k = 1$  general error, doing arithmetic over  $\text{GF}(5)$ .

[Note: The packet indices for this problem begin with 0, not 1.]

- (a) Steve constructs a polynomial  $P(x) \pmod{5}$  of degree at most 2, so that

$$P(0) = 1, \quad P(1) = 1, \quad P(2) = 4.$$

Find  $P(x)$ , and determine the codeword  $(c_0, c_1, c_2, c_3, c_4)$  that Steve transmits.

You receive the message  $(0, 1, 4, 0, 4)$  from Steve, knowing that one packet was corrupted. (Recall that since you're the recipient, you do not know the original message and  $P(x)$ .)

- (b) Write, but do not solve, the system of linear equations that you will use in the Berlekamp-Welch algorithm to find  $Q(x)$  and  $E(x)$ .
- (c) You solve the system of equations you wrote in part (b) to obtain  $Q(x) = 4x^3 + x^2 + x$  and  $E(x) = x$ . How do you recover the original message from  $Q(x)$  and  $E(x)$ ?

### Solution:

- (a) Steve uses Lagrange interpolation to construct the unique quadratic polynomial  $P(x)$  such that  $P(0) = m_0 = 1, P(1) = m_1 = 1, P(2) = m_2 = 4$ . Doing all arithmetic over  $\text{GF}(5)$  (so, for example,  $2^{-1} = 3 \pmod{5}$ ),

$$\Delta_0(x) = \frac{(x-1)(x-2)}{(0-1)(0-2)} = \frac{x^2 - 3x + 2}{2} \equiv 3(x^2 - 3x + 2) \pmod{5}$$

$$\Delta_1(x) = \frac{(x-0)(x-2)}{(1-0)(1-2)} = \frac{x^2 - 2x}{-1} \equiv 4(x^2 - 2x) \pmod{5}$$

$$\Delta_2(x) = \frac{(x-0)(x-1)}{(2-0)(2-1)} = \frac{x^2 - x}{2} \equiv 3(x^2 - x) \pmod{5}$$

$$\begin{aligned} P(x) &= m_0\Delta_0(x) + m_1\Delta_1(x) + m_2\Delta_2(x) \\ &= 1\Delta_0(x) + 1\Delta_1(x) + 4\Delta_2(x) \\ &\equiv 4x^2 + x + 1 \pmod{5} \end{aligned}$$

For the transmitted codeword, Steve adds 2 redundant points from  $P(x)$ . Since 3 and 4 are the only  $x$ -values in  $\text{GF}(5)$  that have not been used yet, Steve computes  $P(3) = 0, P(4) = 4$ , and so the codeword is  $(1, 1, 4, 0, 4)$ .

- (b) You receive the message  $(r_0, r_1, r_2, r_3, r_4) = (0, 1, 4, 0, 4)$ .  $E(x)$  is degree 1, since there is 1 corruption, and since  $P(x)$  is degree 2,  $Q(x)$  is degree  $2 + 1 = 3$ . Let  $E(x) = x + b_0$  be the error-locator polynomial, and  $Q(x) = P(x)E(x) = a_3x^3 + a_2x^2 + a_1x + a_0$ . Since  $Q(i) = P(i)E(i) = r_iE(i)$  for  $0 \leq i \leq 4$ , we have the following equalities (mod 5)

$$Q(0) = 0E(0)$$

$$Q(1) = 1E(1)$$

$$Q(2) = 4E(2)$$

$$Q(3) = 0E(3)$$

$$Q(4) = 4E(4)$$

which can be rewritten as a system of linear equations

$$\begin{array}{ccccccccc} & & & & & a_0 & & & = & 0 \\ a_3 & + & & a_2 & + & a_1 & + & a_0 & - & b_0 & = & 1 \\ 8a_3 & + & 4a_2 & + & 2a_1 & + & a_0 & - & 4b_0 & = & 3 & . \\ 27a_3 & + & 9a_2 & + & 3a_1 & + & a_0 & & & = & 0 \\ 64a_3 & + & 16a_2 & + & 4a_1 & + & a_0 & - & 4b_0 & = & 1 \end{array}$$

- (c) You are given that

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

$$E(x) = x + b_0 = x.$$

Since  $Q(x) = P(x)E(x)$ , you can compute  $P(x) = Q(x)/E(x) = 4x^2 + x + 1$ . The error locating polynomial  $E(x)$  is degree 1, so there is only one error. Since  $E(x) = x = x - 0$ , the corrupted packet was the first one (index 0). To correct this error, you evaluate  $P(0) = 1$  and combine this with the two uncorrupted packets  $r_1 = 1$  and  $r_2 = 4$  to recover the original message

$$(m_0, m_1, m_2) = (1, 1, 4).$$

## 6 Secrets in the United Nations

### Note 8

A vault in the United Nations can be opened with a secret combination  $s \in \mathbb{Z}$ . In only two situations should this vault be opened: (i) all 193 member countries must agree, or (ii) at least 55 countries, plus the U.N. Secretary-General, must agree.

- (a) Propose a scheme that gives private information to the Secretary-General and all 193 member countries so that the secret combination  $s$  can only be recovered under either one of the two specified conditions.

- (b) The General Assembly of the UN decides to add an extra level of security: each of the 193 member countries has a delegation of 12 representatives, all of whom must agree in order for that country to help open the vault. Propose a scheme that adds this new feature. The scheme should give private information to the Secretary-General and to each representative of each country.

**Solution:**

- (a) Create a polynomial of degree 192 and give each country one point. Give the Secretary General  $193 - 55 = 138$  distinct points, so that if the Secretary-General collaborates with 55 countries, they will have a total of 193 points and can reconstruct the polynomial. Without the Secretary-General, the polynomial can still be recovered if all 193 countries come together. (We do all our work in  $\text{GF}(p)$  for some large prime  $p$  with  $p > s$  and  $p \geq 1 + 193 + 138$ , since we need to distribute a point to each of the countries, 138 points to the Secretary General, and one point for the secret).

Alternatively, we could have one scheme for condition (i) and another for (ii). The first condition is the secret-sharing setup we discussed in the notes, so a single polynomial of degree 192 suffices, with each country receiving one point, and evaluation at zero returning the combination  $s$ . For the second condition, create a polynomial  $f$  of degree 1 with  $f(0) = s$ , and give  $f(1)$  to the Secretary-General. Now create a second polynomial  $g$  of degree 54, with  $g(0) = f(2)$ , and give one point of  $g$  to each country. This way any 55 countries can recover  $g(0) = f(2)$ , and then can consult with the Secretary-General to recover  $s = f(0)$  from  $f(1)$  and  $f(2)$ .

- (b) We'll layer an *additional* round of secret-sharing onto the scheme from part (a). If  $t_i$  is the key given to the  $i$ th country, produce a degree-11 polynomial  $f_i$  so that  $f_i(0) = t_i$ , and give one point of  $f_i$  to each of the 12 delegates. Do the same for each country (using different  $f_i$  each time, of course).