CS 70 Discrete Mathematics and Probability Theory
Fall 2017 Satish Rao and Kannan Ramchandran

HW 6

Sundry

Before you start your homework, write down your team. Who else did you work with on this homework? List names and email addresses. (In case of homework party, you can also just describe the group.) How did you work on this homework? Working in groups of 3-5 will earn credit for your "Sundry" grade.

Please copy the following statement and sign next to it:

I certify that all solutions are entirely in my words and that I have not looked at another student's solutions. I have credited all external sources in this write up.

1 How Many Polynomials?

Let $P(x)$ be a polynomial of degree at most 2 over GF(5). As we saw in lecture, we need $d+1$ distinct points to determine a unique d-degree polynomial.

(a) Assume that we know $P(0) = 1$, and $P(1) = 2$. Now we consider $P(2)$. How many values can $P(2)$ have? How many distinct polynomials are there?

(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 different polynomials are there?

(c) How many different polynomials of degree at most d over GF(p) are there if we only know k values, where $k \leq d$?

2 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.

3 Error-Correcting Codes

(a) Recall from class the error-correcting code for erasure errors, which protects against up to k lost packets by sending a total of $n+k$ packets (where n is the number of packets in the original message). Often the number of packets lost is not some fixed number k, but rather a *fraction* of the number of packets sent. Suppose we wish to protect against a fraction α of lost packets (where $0 < \alpha < 1$). At least how many packets do we need to send (as a function of n and α)?

(b) Repeat part (a) for the case of general errors.

4 Error-Correcting Polynomials

(a) Alice has a length 8 message to Bob. There are 2 communication channels available. When n packets are fed through channel A, the channel will only deliver 5 packets (picked at random). Similarity, channel B will only deliver 5 packets (picked at random), but it will also corrupt (change the value) of one of the delivered packets. All channels can only work if at least 10 packets are sent through it. Using the 2 channels, how can Alice send the message to Bob?

(b) Alice wishes to send a message to Bob as the coefficients of a degree 2 polynomial P. For a message $[m_1, m_2, m_3]$, she creates polynomial $P = m_1x^2 + m_2x + m_3$ and sends 5 packets: $(0,P(0)),(1,P(1)),(2,P(2)),(3,P(3)),(4,P(4))$. However, Eve interferes and changes one of the values of a packet before it reaches Bob. If Bob receives

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

and knows Alice's encoding scheme and that Eve changed one of the packets, can he still figure out what the original message was? If so find it as well as the x-value of the packet that Eve changed, if not, explain why he can not. (Work in mod 11.)

(c) Alice decides that putting the message as the coefficients of a polynomial is too inefficient for long messages because the degree of the polynomial grows quite large. Instead, she

decides to encode the message as values in a degree 2 polynomial. For a 5 length message $[m_1, m_2, m_3, m_4, m_5]$, she creates a degree 2 polynomial P such that $P(0) = m_1, P(1) = m_2, P(2) = m_3, P(3) = m_4, P(4) = m_5$. She then sends the length 5 message directly to Bob as 5 packets: $(0, m_1), (1, m_2), (2, m_3, (3, m_4), (4, m_5)$. Eve again interfere and changes the value of a packet before it reaches Bob. If Bob receives $(0,0), (1,3), (2,0), (3,3), (4,0)$ and knows Alice's encoding scheme and that Eve changed one of the packets, can he still figure out what the original message was? If so find it as well as the x-value of the packet that Eve changed, if not, explain why he can not. (Work in mod 11.)

(d) After getting tired of decoding degree 2 polynomials, Bob convinces Alice to send messages using a degree 1 polynomial instead. To be on the safer side, Alice decides to continue to send 5 points on the polynomial even though it is only degree 1. She encodes and sends a length 5 message in the same way as part (c) (except using a degree 1 polynomial). Eve however, decides to change 2 of the packets. After Eve interferes, Bob receives $(0, -3), (1, -1), (2, x), (3, -3), (4, 5)$. If Alice sent $(0, -3), (1, -1), (2, 1), (3, 3)$, $(4, 5)$, for what values of x will Bob not be able to uniquely determine the Alice's message? (Assume Bob knows that Eve changed 2 of the packets and work in mod 13.)

5 Distance Properties

Imagine that you want to send a message x of length n over the binary alphabet $\{0, 1\}$. However, you want to prepare for the possibility that one of your bits will be corrupted. You therefore send a codeword y of length $m > n$ given by an encoding function $E : y = E(x)$. The encoding function is one-to-one (or else there would be ambiguity in which message you sent), and furthermore the encoding function has the property that for any two distinct codewords y_1, y_2 in the range of E, y_1 and y_2 have a Hamming distance of at least 3 (this ensures that you can correct for a corruption of at most one bit).

Prove that $m \geq n + \log_2(m+1)$, that is, the number of extra bits that you have to send is at least logarithmic in your original message length.

Hint: Try a counting approach. For each codeword y, let B_y be the set of length-m strings with a Hamming distance of at most 1 away from y. Observe that for distinct codewords y_1 and y_2, B_{y_1} and B_{y_2} do not overlap.