EECS 70 Discrete Mathematics and Probability Theory
Fall 2015 Satish Rao

Discussion 7B

1. List decoding

1. Consider an n-character message encoded into $m = n + k$ characters over the field $GF(p)$ using polynomials. Suppose that one receives $n-1$ of the m packets. Give a method to find a list of size at most p of all possible n-character messages.

   Solution: Since we are trying to encode an n character message using polynomials, we are going to fit our message into a degree $n-1$ polynomial and then encode our message into the length m message $[P(0)P(1)\dots P(m-1)]$. Now, we receive $n-1$ of these characters; suppose without loss of generality that the character at position k, $P(k)$ was not received. Now, we know $n-1$ points of the polynomial $P(x)$, but knowing these $n-1$ points gives us no information about $P(k)$ since it is possible for us to construct a degree $n-1$ polynomial that goes through the $n-1$ known points no matter what the value of $P(k)$ is. However, suppose we fix the value of $P(k)$, then it turns out that there is exactly one polynomial that goes through the $n-1$ known points and $P(k)$, since a degree $n-1$ polynomial is uniquely determined by n of its points. Since there are p possible values of $P(k)$, there are at most p different polynomials $P(x)$ that could possibly be our encoding polynomial (one for each possible value of $P(k)$). We would generate the p possible messages as follows:

   ```
   foreach value of l in the range 0 to p-1
     P(x) = interpolate(n-1 known points, P(k) = l)
     generate the possible message [P(0) P(1) ... P(n-1)]
   end
   ```

2. Consider an n character message encoded into $m = n + 2k$ characters over the field $GF(p)$ using polynomials. Suppose that $k+1$ of the m received packets are corrupted. Give a method to find a list of all possible messages which contain the original message. What is the size of the list for your scheme?

   Solution: We can use a similar approach to above; we know that we have $k+1$; if we knew in advance that we were going to have $k+1$ errors we would have sent $n+2k+2$ packets in order to make sure that we could perform error correction using the Berlekamp-Welsh method to decode to the correct message. However, we ended up only sending $n+2k$ packets. If we knew the (correct) values of two more packets, then we could use Berlekamp-Welsh to decode the message. Since we do not know the values of two more packets, we can do what we did in part (a) and just guess what they are to generate possible messages. Since for the value of each packets there are p possible values, at most we will generate p^2 possible messages, and the real message will be included.

   ```
   foreach value of a in the range 0 to p-1
     foreach value of b in the range 0 to p-1
       P(x) = Berlekamp-Welsh(n+2k known values,
                               r[n+2k] = a, r[n+2k+1] = b)
       generate the possible message [P(0) P(1) ... P(n-1)]
     end
   end
   ```

3. Consider the protocol in (b) where we are working in GP(7). Let the original message have $n = 1$ and $k = 2$, so there are 5 symbols. Now suppose that there are 3 errors, but these three errors all landed on different values. Assume that we received: $0,0,1,2,3$. How does your list-decoding strategy perform?
   Solution: Given $n = 1$, we have encoded the message into a degree 0 polynomial, so it is just a repetition code (all symbols in the message should be the same). In this case, the list-decoding strategy will pick out the one right answer: 0. Everything else results in 4 errors.
   TA: provide another example: $0,0,1,1,3$. In this case, we know only two messages are plausible: 0 and 1. This is still informative in the way that we know 3 is not correct.

2. Pokemon Counting!

1. I have caught 30 different Pokemon so far. In how many ways can I choose a team of 6, such that the order of my team matters?
   Solution: $30 \cdot 29 \cdot 28 \cdot 27 \cdot 26 \cdot 25$. (30 choices for the first slot, 29 for the second slot, and so on.)
2. For this part and the next two, you can assume we no longer care about the order of the Pokemon. In how many ways can I choose a team of 6 under this assumption?
   Solution: $\binom{30}{6}$
3. Among my 30 caught Pokemon, only 4 can learn the move "Fly". In how many ways can I choose a team of 6, such that there is **exactly** one Pokemon who knows the move "Fly"?
   Solution: $\binom{4}{1} \cdot \binom{26}{5}$
4. In how many ways can I choose a team of 6, such that there is **at least** one Pokemon who knows the move "Fly"?
   Solution: This is the same as the number of ways to choose six Pokemon subtract the number of ways to choose six Pokemon, where none of which knows the move "Fly". $\binom{30}{6} - \binom{26}{6}$. We can also enumerate the possibilities $(1,2,3,4$ Pokemon that know how to fly), but it will take longer.
5. You were victorious against the Elite Four, and Professor Oak generously invited all six members of your team to the Hall of Fame. Suppose he wants to sit your team in a circular table for dinner (which consists of only Oran Berries), in how many ways can he do so?
   Solution: $\frac{6!}{6} = 5!$. Why are we dividing by 6? It's because if each Pokemon move one seat to the right (or to the left), it's the same seating! And there are 6 different ways the Pokemon can move and still preserve the same ordering. Alternatively, we can think of this as ordering 5 Pokemon in a straight line, then bend the line to form an almost-circle, and the last Pokemon only has one position to fit in to form a circle.
6. Suppose Charizard and Pikachu, two members of your team, want to sit next to each other. How would your answer to the above question change?
   Solution: $2 \cdot \frac{5!}{5} = 2 \cdot 4! = 48$. Treat Charizard and Pikachu as one Pokemon (Charichu), and then carry out the same step above under the assumption that there's only 5 Pokemon to be seated. Note that the two Pokemon can swap seats in $2! = 2$ ways, so we multiply by 2.
7. Suppose Meowth and Pikachu, two members of your team, don't want to sit next to each other. In how many ways can Professor Oak arrange the seatings?
   Solution: $5! - 2 \cdot \frac{5!}{5} = 5! - 2 \cdot 4! = 72$.

3. Pokemon Anagrams

An anagram of a word is any re-ordering of the letters of the word, in any order. It does not have to be an English word or an actual Pokemon name.

1. How many different anagrams of PIKACHU are there?

   Solution: Since the order of the letters matter and all of the letters are distinct, there are 7!

2. How many different anagrams of KADABRA are there?

   Solution: If we first pretend that the 3 A's are all distinct, then there are 7! anagrams. But since the 3 A's are identical, we counted each anagram an extra 3! ways. Hence, there are $7!/3!$ anagrams total. Another way to think about this: we first choose 3 of out of the 7 possible positions to place the A, there are $\binom{7}{3}$ choices. There are then 4 positions left to place the K, 3 positions to place the D, 2 positions to place the B, and one position to place the R, so there are $\binom{7}{3}(4!) = 7!/3!$ anagrams total.

3. How many different anagrams of RATTATA are there?

   Solution: Since the T's and A's are identical and appear three times each, the answer is $\frac{7!}{3!3!}$.