CS 70 Discrete Mathematics and Probability Theory
Fall 2021 HW 6

1 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.

Solution:

(a) Suppose we send a total of m packets (where m is to be determined). Since at most a fraction α of these are lost, the number of packets received is at least $(1-\alpha)m$. But in order to reconstruct the polynomial used in transmission, we need at least n packets. Hence it is sufficient to have $(1-\alpha)m \geq n$, which can be rearranged to give $m \geq n/(1-\alpha)$.

(b) Suppose we send a total of $m = n+2k$ packets, where k is the number of errors we can guard against. The number of corrupted packets is at most αm, so we need $k \geq \alpha m$. Hence $m \geq n+2\alpha m$. Rearranging gives $m \geq n/(1-2\alpha)$.

Note: Recovery in this case is impossible if $\alpha \geq 1/2$.

2 Berlekamp-Welch Algorithm

In this question we will send the message $(m_0, m_1, m_2) = (1,1,4)$ of length $n = 3$. We will use an error-correcting code for $k = 1$ general error, doing arithmetic over GF(5).

(a) Construct a polynomial $P(x) \pmod 5$ of degree at most 2, so that

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

What is the message $(c_0, c_1, c_2, c_3, c_4)$ that is sent?

(b) Suppose the message is corrupted by changing c_0 to 0. Set up the system of linear equations in the Berlekamp-Welch algorithm to find $Q(x)$ and $E(x)$.

(c) Assume that after solving the equations in part (b) we get $Q(x) = 4x^3 + x^2 + x$ and $E(x) = x$. Show how to recover the original message from Q and E.

Solution:

(a) We use 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 GF(5), so that i.e. $2^{-1} = 3 \pmod 5$,

$$\begin{aligned}
\Delta_0(x) &= \frac{(x-1)(x-2)}{(0-1)(0-2)} = \frac{x^2-3x+2}{2} \equiv 3(x^2-3x+2) \pmod 5 \\
\Delta_1(x) &= \frac{(x-0)(x-2)}{(1-0)(1-2)} = \frac{x^2-2x}{-1} \equiv 4(x^2-2x) \pmod 5 \\
\Delta_2(x) &= \frac{(x-0)(x-1)}{(2-0)(2-1)} = \frac{x^2-x}{2} \equiv 3(x^2-x) \pmod 5 \\
P(x) &= m_0\Delta_0(x) + m_1\Delta_1(x) + m_2\Delta_2(x) \\
&= 1\Delta_0(x) + 1\Delta_1(x) + 4\Delta_2(x) \\
&\equiv 4x^2 + x + 1 \pmod 5
\end{aligned}$$

For the final message we need to add 2 redundant points of P. Since 3 and 4 are the only points in GF(5) that we have not used yet, we compute $P(3) = 0, P(4) = 4$, and so our message is $(1,1,4,0,4)$.

(b) The message received is $(c'_0, c'_1, c'_2, c'_3, c'_4) = (0,1,4,0,4)$. Let $R(x)$ be the function such $R(i) = c'_i$ for $0 \le i < 5$. 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 $Q(i) = P(i)E(i) = R(i)E(i)$ for $1 \le i < 5$, we have the following equalities $\pmod 5$

$$\begin{aligned}
Q(0) &= 0E(0) \\
Q(1) &= 1E(1) \\
Q(2) &= 4E(2) \\
Q(3) &= 0E(3) \\
Q(4) &= 4E(4)
\end{aligned}$$

which can be rewritten as a system of linear equations

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

(c) From the solution, we know

$$\begin{aligned}
Q(x) &= 4x^3 + x^2 + x, \\
E(x) &= x + b_0 = x.
\end{aligned}$$

Since $Q(x) = P(x)E(x)$, the recipient can compute $P(x) = Q(x)/E(x) = 4x^2 + x + 1$, the polynomial $P(x)$ from part (a) used by the sender. The error locating polynomial $E(x)$ is degree one, so there is only one error, and as $E(x) = x = x - 0$, the corrupted bit was the first one. To correct this error we evaluate $P(0) = 1$ and combine this with the two uncorrupted bits m_1, m_2, to get the original message

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

3 Error-Detecting Codes

Suppose Alice wants to transmit a message of n symbols, so that Bob is able to *detect* rather than *correct* any errors that have occurred on the way. That is, Alice wants to find an encoding so that Bob, upon receiving the code, is able to either

(I) tell that there are no errors and decode the message, or

(II) realize that the transmitted code contains at least one error, and throw away the message.

Assuming that we are guaranteed a maximum of k errors, how should Alice extend her message (i.e. by how many symbols should she extend the message, and how should she choose these symbols)? You may assume that we work in GF(p) for very large prime p. Show that your scheme works, and that adding any lesser number of symbols is not good enough.

Solution:

Since k bits can break, it seems reasonable to extend our message by k symbols for a total of $n+k$. And indeed, we show that this works: Let Alice generate her message $y_0, \dots y_{n-1}$ of length n by constructing the unique polynomial f of degree $\leq n-1$ that passes through (i, y_i) for $i \in \{0, \dots, n-1\}$, and add the k extra symbols $y_j = f(j)$, where $j \in \{n, \dots, n+k-1\}$. Now Bob receives the message $r_i, i \in \{0, \dots, n+k-1\}$, upon which he interpolates the unique degree $\leq n-1$ polynomial g that passes through the points $(0, r_0), \dots, (n-1, r_{n-1})$. We claim that the message is corrupted if and only if $g(i) \neq r_i$ for some $i \in \{n, \dots, n+k-1\}$.

The backward direction becomes clear when stated as its contrapositive: If the message contains no error, then $g(i)$ and $f(i)$ coincide on all of n points $\{0, \dots, n-1\}$. Since they are both of degree $n-1$, they must be the same polynomial and hence $g(i) = f(i) = r_i$ for all i.

Let us prove the forward direction: Since we know that at most k errors occured, there must exist a subset $A \subset \{0, \dots, n+k-1\}$ of size n on which $r_i = y_i$. Now either

1. $A = \{0, \dots, n-1\}$, in which case $g = f$ and at least one error must have occured for some $j_0 \in \{n, \dots, n+k-1\}$. But then $r_{j_0} \neq y_{j_0} = f(j_0) = g(j_0)$, which is what we wanted to show.
2. Or at least one error occured for an index $i \in \{0, \dots, n-1\}$ in which case $g \neq f$. But since g and f are of degree $n-1$ and $| A | = n$, f and g cannot take the same values on A, so there must be some element $j_0 \in A, j_0 \in \{n, \dots, n+k-1\}$ for which $g(j_0) \neq f(j_0) = y_{j_0} = r_{j_0}$.

Lastly, we need to show that our algorithm doesn't work if Alice extends her message by less than k symbols, which we can do by crafting a counterexample: Assume Alice sends $m < n+k-1$ symbols in the same fashion as above, then we may corrupt $y_{n-1}, \dots y_{m-1}$ by setting $r_{n-1} \neq y_{n-1}$ and $r_j = h(j)$ for $j \in \{n, \dots, m-1\}$, where h is the unique polynomial of degree $\leq n-1$ passing through $(0, y_0), \dots (n-2, y_{n-2}), (n-1, r_{n-1})$. Since Bob is going to reconstruct $g = h$, $g(j) = r_j$ for all $j \in \{n, \dots, m-1\}$ and he will not notice the corruption.

4 Counting, Counting, and More Counting

The only way to learn counting is to practice, practice, practice, so here is your chance to do so. Although there are many subparts, each subpart is fairly short, so this problem should not take any longer than a normal CS70 homework problem. You do not need to show work, and **Leave your answers as an expression** (rather than trying to evaluate it to get a specific number).

(a) How many ways are there to arrange n 1s and k 0s into a sequence?

(b) How many 7-digit ternary (0,1,2) bitstrings are there such that no two adjacent digits are equal?

(c) A bridge hand is obtained by selecting 13 cards from a standard 52-card deck. The order of the cards in a bridge hand is irrelevant.
i. How many different 13-card bridge hands are there? ii. How many different 13-card bridge hands are there that contain no aces? iii. How many different 13-card bridge hands are there that contain all four aces? iv. How many different 13-card bridge hands are there that contain exactly 6 spades?

(d) Two identical decks of 52 cards are mixed together, yielding a stack of 104 cards. How many different ways are there to order this stack of 104 cards?

(e) How many 99-bit strings are there that contain more ones than zeros?

(f) An anagram of ALABAMA is any re-ordering of the letters of ALABAMA, i.e., any string made up of the letters A, L, A, B, A, M, and A, in any order. The anagram does not have to be an English word.
i. How many different anagrams of ALABAMA are there? ii. How many different anagrams of MONTANA are there?

(g) How many different anagrams of ABCDEF are there if: (1) C is the left neighbor of E; (2) C is on the left of E (and not necessarily E's neighbor)

(h) We have 9 balls, numbered 1 through 9, and 27 bins. How many different ways are there to distribute these 9 balls among the 27 bins? Assume the bins are distinguishable (e.g., numbered 1 through 27).

(i) How many different ways are there to throw 9 identical balls into 27 bins? Assume the bins are distinguishable (e.g., numbered 1 through 27).

(j) We throw 9 identical balls into 7 bins. How many different ways are there to distribute these 9 balls among the 7 bins such that no bin is empty? Assume the bins are distinguishable (e.g., numbered 1 through 7).

(k) There are exactly 20 students currently enrolled in a class. How many different ways are there to pair up the 20 students, so that each student is paired with one other student? Solve this in at least 2 different ways. **Your final answer must consist of two different expressions.**

(l) How many solutions does $x_0+x_1+\cdots+x_k=n$ have, if each x must be a non-negative integer?

(m) How many solutions does $x_0+x_1=n$ have, if each x must be a *strictly positive* integer?

(n) How many solutions does $x_0+x_1+\cdots+x_k=n$ have, if each x must be a *strictly positive* integer?

Solution:

(a) $\binom{n+k}{k}$

(b) There are 3 options for the first digit. For each of the next digits, they only have 2 options because they cannot be equal to the previous digit. Thus, $3*2^6$

(c) We have to choose 13 cards out of 52 cards, so this is just $\binom{52}{13}$.

We now have to choose 13 cards out of 48 non-ace cards. So this is $\binom{48}{13}$.

We now require the four aces to be present. So we have to choose the remaining 9 cards in our hand from the 48 non-ace cards, and this is $\binom{48}{9}$.

We need our hand to contain 6 out of the 13 spade cards, and 7 out of the 39 non-spade cards, and these choices can be made separately. Hence, there are $\binom{13}{6}\binom{39}{7}$ ways to make up the hand.

(d) If we consider the 104! rearrangements of 2 identical decks, since each card appears twice, we would have overcounted each distinct rearrangement. Consider any distinct rearrangement of the 2 identical decks of 52 cards and see how many times this appears among the rearrangement of 104 cards where each card is treated as different. For each identical pair (such as the two Ace of spades), there are two ways they could be permuted among each other (since $2!=2$). This holds for each of the 52 pairs of identical cards. So the number 104! overcounts the actual number of rearrangements of 2 identical decks by a factor of 2^{52}. Hence, the actual number of rearrangements of 2 identical decks is $104!/2^{52}$.

(e) **Answer 1:** There are $\binom{99}{k}$ 99-bit strings with k ones and $99-k$ zeros. We need $k>99-k$, i.e. $k\geq 50$. So the total number of such strings is $\sum_{k=50}^{99}\binom{99}{k}$.
This expression can however be simplified. Since $\binom{99}{k}=\binom{99}{99-k}$, we have

$$\sum_{k=50}^{99}\binom{99}{k}=\sum_{k=50}^{99}\binom{99}{99-k}=\sum_{l=0}^{49}\binom{99}{l}$$

by substituting $l = 99 - k$. Now $\sum_{k=50}^{99} \binom{99}{k} + \sum_{l=0}^{49} \binom{99}{l} = \sum_{m=0}^{99} \binom{99}{m} = 2^{99}$. Hence, $\sum_{k=50}^{99} \binom{99}{k} = (1/2) \cdot 2^{99} = 2^{98}$.

Answer 2: Symmetry Since the answer from above looked so simple, there must have been a more elegant way to arrive at it. Since 99 is odd, no 99-bit string can have the same number of zeros and ones. Let A be the set of 99-bit strings with more ones than zeros, and B be the set of 99-bit strings with more zeros than ones. Now take any 99-bit string x with more ones than zeros i.e. $x \in A$. If all the bits of x are flipped, then you get a string y with more zeros than ones, and so $y \in B$. This operation of bit flips creates a one-to-one and onto function (called a bijection) between A and B. Hence, it must be that $|A| = |B|$. Every 99-bit string is either in A or in B, and since there are 2^{99} 99-bit strings, we get $|A| = |B| = (1/2) \cdot 2^{99}$. The answer we sought was $|A| = 2^{98}$.

(f) ALABAMA: The number of ways of rearranging 7 distinct letters and is $7!$. In this 7 letter word, the letter A is repeated 4 times while the other letters appear once. Hence, the number $7!$ overcounts the number of different anagrams by a factor of $4!$ (which is the number of ways of permuting the 4 A's among themselves). Aka, we only want $1/4!$ out of the total rearrangements. Hence, there are $7!/4!$ anagrams.

MONTANA: In this 7 letter word, the letter A and N are each repeated 2 times while the other letters appear once. Hence, the number $7!$ overcounts the number of different anagrams by a factor of $2! \times 2!$ (one factor of $2!$ for the number of ways of permuting the 2 A's among themselves and another factor of $2!$ for the number of ways of permuting the 2 N's among themselves). Hence, there are $7!/(2!)^2$ different anagrams.

(g) (1) We consider CE is a new letter X, then the question becomes counting the rearranging of 5 distinct letters, and is $5!$. (2) Symmetry: Let A be the set of all the rearranging of ABCDEF with C on the left side of E, and B be the set of all the rearranging of ABCDEF with C on the right side of E. $|A \bigcup B| = 6!, |A \cap B| = 0$. There is a bijection between A and B by construct a operation of exchange the position of C and E. Thus $|A| = |B| = 6!/2$.

(h) Each ball has a choice of which bin it should go to. So each ball has 27 choices and the 9 balls can make their choices separately. Hence, there are 27^9 ways.

(i) Since there is no restriction on how many balls a bin needs to have, this is just the problem of throwing k identical balls into n distinguishable bins, which can be done in $\binom{n+k-1}{k}$ ways. Here $k = 9$ and $n = 27$, so there are $\binom{35}{9}$ ways.

(j) **Answer 1:** Since each bin is required to be non-empty, let's throw one ball into each bin at the outset. Now we have 2 identical balls left which we want to throw into 7 distinguishable bins. There are 2 cases to consider:
Case 1: The 2 balls land in the same bin. This gives 7 ways.
Case 2: The 2 balls land in different bins. This gives $\binom{7}{2}$ ways of choosing 2 out of the 7 bins for the balls to land in. Note that it is *not* 7×6 since the balls are identical and so there is no order on them.
Summing up the number of ways from both cases, we get $7 + \binom{7}{2}$ ways.

Answer 2: Since each bin is required to be non-empty, let's throw one ball into each bin at the outset. Now we have 2 identical balls left which we want to throw into 7 distinguishable bins. From class (see note 11), we already saw that the number of ways to put k identical balls into n distinguishable bins is $\binom{n+k-1}{k}$. Taking $k = 2$ and $n = 7$, we get $\binom{8}{2}$ ways to do this.
EASY EXERCISE: Can you give an expression for the number of ways to put k identical balls into n distinguishable bins such that no bin is empty?

(k) **Answer 1:** Let's number the students from 1 to 20. Student 1 has 19 choices for her partner. Let i be the smallest index among students who have not yet been assigned partners. Then no matter what the value of i is (in particular, i could be 2 or 3), student i has 17 choices for her partner. The next smallest indexed student who doesn't have a partner now has 15 choices for her partner. Continuing in this way, the number of pairings is $19 \times 17 \times 15 \times \cdots \times 1 = \prod_{i=1}^{10}(2i-1)$.

Answer 2: Arrange the students numbered 1 to 20 in a line. There are 20! such arrangements. We pair up the students at positions $2i-1$ and $2i$ for i ranging from 1 to 10. You should be able to see that the 20! permutations of the students doesn't miss any possible pairing. However, it counts every different pairing multiple times. Fix any particular pairing of students. In this pairing, the first pair had freedom of 10 positions in any permutation that generated it, the second pair had a freedom of 9 positions in any permutation that generated it, and so on. There is also the freedom for the elements within each pair i.e. in any student pair (x,y), student x could have appeared in position $2i-1$ and student y could have appeared in position $2i$ and also vice versa. This gives 2 ways for each of the 10 pairs. Thus, in total, these freedoms cause $10! \times 2^{10}$ of the 20! permutations to give rise to this particular pairing. This holds for each of the different pairings. Hence, 20! overcounts the number of different pairings by a factor of $10! \times 2^{10}$. Hence, there are $20!/(10! \cdot 2^{10})$ pairings.

Answer 3: In the first step, pick a pair of students from the 20 students. There are $\binom{20}{2}$ ways to do this. In the second step, pick a pair of students from the remaining 18 students. There are $\binom{18}{2}$ ways to do this. Keep picking pairs like this, until in the tenth step, you pick a pair of students from the remaining 2 students. There are $\binom{2}{2}$ ways to do this. Multiplying all these, we get $\binom{20}{2}\binom{18}{2}\ldots\binom{2}{2}$. However, in any particular pairing of 20 students, this pairing could have been generated in 10! ways using the above procedure depending on which pairs in the pairing got picked in the first step, second step, ..., tenth step. Hence, we have to divide the above number by 10! to get the number of different pairings. Thus there are $\binom{20}{2}\binom{18}{2}\ldots\binom{2}{2}/10!$ different pairings of 20 students.

You may want to check for yourself that all three methods are producing the same integer, even though they are expressed very differently.

(l) $\binom{n+k}{k}$. This is just n indistinguishable balls into k+1 distinguishable bins (stars and bars). There is a bijection between a sequence of n ones and k plusses and a solution to the equation: x_0 is the number of ones before the first plus, x_1 is the number of ones between the first and second plus, etc. A key idea is that if a bijection exists between two sets they must be the same size, so counting the elements of one tells us how many the other has. Note that this is the exact same answer as part a - make sure you understand why!

(m) $n-1$. It's easy just to enumerate the solutions here. x_0 can take values $1,2,\ldots,n-1$ and this uniquely fixes the value of x_1. So, we have $n-1$ ways to do this. But, this is just an example of the more general question below.

(n) $\binom{(n-(k+1))+k}{k} = \binom{n-1}{k}$. This is just n-(k+1) indistinguishable balls into distinguishable k+1 bins. By subtracting 1 from all $k+1$ variables, and $k+1$ from the total required, we reduce it to problem with the same form as the previous problem. Once we have a solution to that we reverse the process, and adding 1 to all the non-negative variables gives us positive variables.

5 Shipping Crates

A widget factory has four loading docks for storing crates of ready-to-ship widgets. Suppose the factory produces 8 indistinguishable crates of widgets and sends each crate to one of the four loading docks.

(a) How many ways are there to distribute the crates among the loading docks?

(b) Now, assume that any time a loading dock contains at least 5 crates, a truck picks up 5 crates from that dock and ships them away. (e.g., if 6 crates are sent to a loading dock, the truck removes 5, leaving 1 leftover crate still in the dock). We will now consider two configurations to be identical if, for every loading dock, the two configurations have the same number of leftover crates in that dock. How would your answer in the previous part compare to the number of outcomes given the new setup? Do not compute the actual value in this part, we will doing that in parts (c) - (e). We are looking for a qualitative answer (greater than part (a), equal to part (a), less than or equal to part (a), etc.) Justify your answer.

(c) We will now attempt to count the number of configurations of crates. First, we look at the case where crates are removed from the dock. How many ways are there to distribute the crates such that some crate gets removed from the dock?

(d) How many ways are there to distribute the crates such that no crates are removed from the dock; i.e. no dock receives at least 5 crates?

(e) Putting it together now, what are the total number of possible configurations for crates in the modified scenario? *Hint:* Observe that, regardless of which dock receives the 5 crates, we end up in the same situation. After all the shipping has been done, how many possible configurations of leftover crates in loading docks are there?

Solution:

(a) This can be solved using stars and bars. We are simply distributing 8 indistinguishable balls into 4 distinguishable bins; the total amount of ways to count this is $\binom{11}{3}$.

(b) There are less possible outcomes in the new setup. The effect of the truck is that of a function, mapping the configurations that we counted in the previous part to a new set of outcomes; although the function may map two distinct configurations to the same outcome, it will certainly not map the same configuration to two different new outcomes. Thus, $\binom{11}{3}$ is a valid upper bound, which indicates that the number of outcomes is still finite (which means we can count it)!

(c) You may notice that it's only possible for one truck to receive five crates; we will leverage this fact in order to simplify our counting. We will count the number of ways to distribute the crate such that some dock receives 5 crates. Observe that, regardless of which dock receives the 5 crates, the scenario reduces to the same thing: we are simply distributing the leftover 3 crates among 4 docks. There are 4 ways to choose which dock has ≥ 5 crates, and $\binom{6}{3}$ ways to distribute the leftover 3 crates. Thus, there are $4\binom{6}{3}$ ways to distribute the crates such that some dock receives 5 crates.

(d) You can put all outcomes into one of two categories: outcomes in which some dock receives more than five crates, and outcomes in which no dock receives more than five crates. Note that these categories are mutually exclusive. As a result, we can take the complement of the previous part; the number of outcomes in which no crates are removed from the dock is simply the total number of outcomes, subtracted by the number of outcomes in which some dock receives more than five crates (found in the previous part). Putting it together, this gives us $\binom{11}{3} - 4\binom{6}{3}$.

(e) Regardless of which dock receives the 5 crates, we are left distributing 3 indistinguishable crates amount 4 docks. Using stars and bars, the total number of outcomes is $\binom{6}{3}$. Thus, to obtain the total amount of configurations, we count the amount of outcomes without removal of the crates, and add this to the amount of outcomes after the removal of the crates. Putting it together, we have that the total number of outcomes is

$$\binom{11}{3} - 4\binom{6}{3} + \binom{6}{3} = \binom{11}{3} - 3\binom{6}{3} = 105.$$

6 Grids and Trees!

Suppose we are given an $n \times n$ grid, for $n \geq 1$, where one starts at $(0,0)$ and goes to (n,n). On this grid, we are only allowed to move left, right, up, or down by increments of 1.

(a) How many shortest paths are there that go from $(0,0)$ to (n,n)?

(b) How many shortest paths are there that go from $(0,0)$ to $(n-1,n+1)$?

Now, consider shortest paths that meet the conditions where we can only visit points (x,y) where $y \leq x$. That is, the path cannot cross line $y = x$. We call these paths n-legal paths for a maze of side length n. Let F_n be the number of n-legal paths.

(c) Compute the number of shortest paths from $(0,0)$ to (n,n) that cross $y = x$. (Hint: Let (i,i) be the first time the shortest path crosses the line $y = x$. Then the remaining path starts from $(i,i+1)$ and continues to (n,n). If in the remainder of the path one exchanges y-direction moves with x-direction moves and vice versa, where does one end up?)

(d) Compute the number of shortest paths from $(0,0)$ to (n,n) that do not cross $y = x$. (You may find your answers from parts (a) and (c) useful.)

(e) A different idea is to derive a recursive formula for the number of paths. Fix some i with $0 \leq i \leq n-1$. We wish to count the number of n-legal paths where the last time the path touches the line $y = x$ is the point (i,i). Show that the number of such paths is $F_i \cdot F_{n-i-1}$. (Hint: If $i = 0$, what are your first and last moves, and where is the remainder of the path allowed to go?)

(f) Explain why $F_n = \sum_{i=0}^{n-1} F_i \cdot F_{n-i-1}$.

(g) Create and explain a recursive formula for the number of trees with n vertices ($n \geq 1$), where each non-root node has degree at most 3, and the root node has degree at most 2. Two trees are different if and only if either left-subtree is different or right-subtree is different.

(Notice something about your formula and the grid problem. Neat!)

Solution: Let $(x,y) \to (x+1,y)$ be a move 'right' command, and $(x,y) \to (x,y+1)$ be a move 'up' command.

(a) There are $\binom{2n}{n}$ paths, as there are total number of $2n$ moves, and n of them must be move 'right' command, the rest of them must be the move 'up' command.

(b) There are $\binom{2n}{n-1}$ paths as there are now $n-1$ move 'right' command.

(c) We will argue that the set of all paths that cross $y = x$, form a bijection with the set of all paths from $(0,0)$ to $(n-1,n+1)$. Suppose we have a path that crosses $y = x$. Once a path crosses $y = x$, we can flip the later portion of the path. Let the first time the invalid path crosses $y = x$ be at (i,i) and arrives at $(i,i+1)$. Then if we do not flip the path, it will arrive at (n,n) by taking $n-i$ "right" commands, and $n-1-i$ "up" commands. If we flip these commands, it will go to $(i+(n-1-i),(i+1)+(n-i)) = (n-1,n+1)$. So all invalid paths map to a path from $(0,0)$ to $(n-1,n+1)$. Next we argue that all path from $(0,0)$ to $(n-1,n+1)$ maps to a invalid path. Paths from $(0,0)$ to $(n-1,n+1)$ must cross the line $y = x$, let it first cross the line at (i,i) and arrives $(i,i+1)$. Then it must take $(n-1)-i$ "right" commands, and $(n+1)-(i+1)$ "up" commands. We flip these commands and so we now have, $n-1-i$ "up" commands, $n-i$ "right" commands. Then the path will arrive $(i+(n-i),(i+1)+(n-1-i)) = (n,n)$ and this new path is considered as invalid path since it crosses $y = x$ at point (i,i). So all paths from $(0,0)$ to $(n-1,n+1)$ can be mapped to an invalid paths.

So there is a bijective mapping between invalid paths and paths from $(0,0)$ to $(n-1,n+1)$. Hence, the number of invalid paths is $\binom{2n}{n-1}$.

(d) The number of paths that don't cross $y = x$ is given by the number of paths minus the number of paths that do cross $y = x$, or $\binom{2n}{n} - \binom{2n}{n-1}$.

(e) Let F_n be the total number of different ways from $(0,0)$ to (n,n) satisfies the condition above. We know $F_1 = 1$. Let (i,i) be the last point on line $y = x$ that a path touches except for (n,n). Then total number of such path is $F_i \cdot F_{n-1-i}$ where F_i is the total number of paths from $(0,0)$ to (i,i). Since (i,i) is the last boundry point it touches, so for all later steps, it must not cross the line $y = x-1$, it's equivalent to say the total number of paths from $(i+1,i)$ to $(n,n-1)$, it's F_{n-1-i}.

(f) From the previous part, the number of n-legal paths where the last time the path touches the line $y = x$ at the point (i,i) is $F_i \cdot F_{n-1-i}$. To get the total number of paths that cross the $y = x$, we simply need to sum across every possible point (i,i) that the path could have passed through. Hence, $F_n = \sum_{i=0}^{n-1} F_i * F_{n-1-i}$.

(g) Let T_n be the total number of different trees with n nodes. The number of different trees when the left subtree has size i and right subtree has size $n-i-1$ is $T_i \cdot T_{n-i-1}$. If we sum over all possible sizes of left subtrees, we can get the total number of different trees: $T_n = \sum_{i=0}^{n-1} T_i T_{n-i-1}$, with the base cases $T_0 = 1, T_1 = 1$. Note that a similar counting argument captures totally different objects (mazes and trees)!

If you are interested in why these problems are related, check out the Catalan numbers.

7 Fermat's Wristband

Let p be a prime number and let k be a positive integer. We have beads of k different colors, where any two beads of the same color are indistinguishable.

(a) We place p beads onto a string. How many different ways are there to construct such a sequence of p beads with up to k different colors?

(b) How many sequences of p beads on the string are there that use at least two colors?

(c) Now we tie the two ends of the string together, forming a wristband. Two wristbands are equivalent if the sequence of colors on one can be obtained by rotating the beads on the other. (For instance, if we have $k = 3$ colors, red (R), green (G), and blue (B), then the length $p = 5$ necklaces RGGBG, GGBGR, GBGRG, BGRGG, and GRGGB are all equivalent, because these are all rotated versions of each other.)

How many non-equivalent wristbands are there now? Again, the p beads must not all have the same color. (Your answer should be a simple function of k and p.)

[*Hint*: Think about the fact that rotating all the beads on the wristband to another position produces an identical wristband.]

(d) Use your answer to part (c) to prove Fermat's little theorem.

Solution:

(a) k^p. For each of the p beads, there are k possibilities for its colors. Therefore, by the first counting principle, there are k^p different sequences.

(b) $k^p - k$. You can have k sequences of a beads with only one color.

(c) Since p is prime, rotating any sequence by less than p spots will produce a new sequence. As in, there is no number x smaller than p such that rotating the beads by x would cause the pattern to look the same. So, every pattern which has more than one color of beads can be rotated to form $p-1$ other patterns. So the total number of patterns equivalent with some bead sequence is p. Thus, the total number of non-equivalent patterns are $(k^p - k)/p$.

(d) $(k^p - k)/p$ must be an integer, because from the previous part, it is the number of ways to count something. Hence, $k^p - k$ has to be divisible by p, i.e., $k^p \equiv k \pmod{p}$, which is Fermat's Little Theorem.