Erdős Magic

Solving counting problems with probability

By: Grant Koh

The Erdős Magic course, taught by Corrine, was about using probability to solve **combinatorics** problems. To start out, we reviewed some key probability concepts such as **conditional probability** and **expected value**. Once we made sure that our foundations were solid, it was time for some magic.

The first problem we looked at involved stars and galaxies. Essentially, each galaxy contained some number of stars, and each star was either red or purple. We wanted to say that for some set of n stars and m galaxies of size k, we can give some **bound** to m such that for all m equal to or less than that bound, there would be some coloring of the stars such that each galaxy has both red and purple stars. We solved this problem by injecting some randomness into it. We said that we could flip a fair coin, and that if it was heads, the star would be red, and if it was tails, the star would be purple. If we can expect less than one galaxy to be monochromatic, it would follow that there would be some arrangement with zero monochromatic galaxies. We were able to use expected value to figure out for which m we had an expected number of monochromatic galaxies less than one, and hence were able to use probability to find solutions to this counting problem. That's Erdős magic!

After doing some more problems using this method, we were ready to explore an environment which combines both combinatorics and randomness - random dance bubbles/maps (more formally referred to as **Erdős-Rényi Random Graphs**). We construct a random dance bubble by considering some selection of multipuses/cities, and then for each group of two multipuses, we flip a coin. If heads, we draw a handlink/tunnel between the two; if tails, we don't. We were able to use probability to answer many questions about these systems, such as how many loops we could expect to see in it, or how many multipuses wouldn't have any handlinks.

Incmletenes f Arithtc

By: Maia

The premise of this class was that Sam had gotten mad and "deleted" math. Our job as students was to "recreate" math. On Day 1, we began by defining one entity: a **container**, denoted {}. Containers could be empty or contain other containers/sets of containers as long as there were no repeats. We let A be the empty container, {}; B be a container that contains A, {{}}; C be a container that contains A and B, {{}{{}}}; and so on.

On Day 2, we defined our first operation: **Union**, denoted "U". The union of two containers Alpha and Beta forms another container that contains all of the objects from Alpha and all of the objects from Beta. For example, since A contains nothing and B contains A, A U B = {A}= B. We renamed each container

A,B,C etc. to be the number of containers that it contains such that A now was named "0", B was named "1", C was named "2" and so on.
We made and proved our first function: S(x) = x+ 1.

Next, we created a set which we called " 🎁". It essentially contains all non-negative integers. On Day 3 we defined and proved more functions through a technique we called "**recurfusion**". We defined "**Fusions**" as functions that take n>0 objects from 🎁 and output one object. They are either proved via "recurfusion" or are constant. The fusions we defined and proved are as follows:

$$L(x) = x$$
$$A(x,y) = x+y$$
$$M(x,y) = x*y$$
$$E(x,y) = x^y$$

On Day 4 we defined functions that show relations. Relations are comparisons of 2 objects such as "=", ">", "<" etc. We defined and proved the following functions using "recurfusion":

$$P(x) = x - 1 \text{ if } x \neq 0, \text{ and } P(x) = 0 \text{ if } x=0$$
$$T(x,y) = 0 \text{ if } x < y, \text{ and } T(x,y) = x - y \text{ if } x > y$$

On Day 5 we defined and proved more relations:

$$W_{and}(x,y,), W_{<}(x,y), W_{=}(x,y), \text{ and } W_{\sim}$$

We even were able to define division with our previous functions!

$$W_{x/y}(x,y) = E(j)[W_{AND}(W_{AND}(W_{=}(M(y,j),x), W_{<=}(j,x)), W_{<}(0,j))]$$

We ended by discussing how easily we could make W_{prime} that tests if a number is prime.

A Tour of Turing

Punishable by Death

By: Maanas

On the first day of "A Tour of Turing," Jonah introduced us to **machines**, where we plug the letters of a word (or number) and track its paths through some nodes and arrows until it ends up at some **node**. If the node has a check mark in it, the word is called "good" and otherwise we reject it! No bad words for us!

From here, (after establishing some conventions of course) we began exploring these machines, making the machines check for divisibility by 2,3,4,5,6, & 7, alternating AB words, and many more desired words. While some were easy, others were very large machines! So, while some of us proved that a machine M3 such that the good words of M3, G(M3), could always be made such that G(M3) = G(M1)∩G(M2), G(M3) = G(M1) ∪G(M2), and G(M3) = G(M1)+G(M2) where + represents concatenation, others showed that these machines couldn't solve some problems! Oh no!

So, Jonah gave us a gift: **psychic nodes**! We could draw multiple paths for some letter from some node and we could magically choose the path that could lead to a check. While this certainly seemed nice, psychic nodes actually couldn't solve anything more than regular machines. So, after much pleading, Jonah gave us a (real) gift called: **memory**! I don't want to spoil what it actually was, but memory was a great tool for us, as it allowed us to solve so many more problems!

At the end, we connected our work to the work done by Alan Turing, mathematician, computer scientist, and cryptanalyst extraordinaire. Turing, the namesake of many things in math (and even a UK law that retroactively pardoned LGBTQ+ people previously criminalized), was the namesake of our double memory machine: **the Turing Machine**!

Top Secret Messages and Very Big Primes

Sorry, I Fell Asleep On My Keyboard and Now I Have a Very Big Prime...100563804078685150820128135754 3

By Emily

There comes a point when whispering is too loud. There [redacted] point [redacted] messages [redacted] redacted [redacted] receiver [redacted] no idea [redacted] say. There comes a point when keyboard smashing into Sage to find huge primes is simultaneously satisfying and an excellent way to send messages.

With some very big primes, very big exponents, and even bigger-primes, we can send messages!

1) Person A finds two very big primes, p and q, and multiplies them to get n.
2) Person A then computes L(n)=(p-1)(q-1), the number of natural numbers less than n and co-prime to n.
3) Person A chooses a number e that is co-prime to L(n) and computes d such that $ed \equiv 1 \pmod{L(n)}$.
4) Person A sends e and n via a public messaging system to Person B.
5) Person B encrypts a message M, making sure that gcd(M, n)=1, and sends Person A the value $E=M^e \pmod{n}$.
6) Using d and E, Person A computes $E^d=M^{ed}$ and mods the result mod n, getting $M^{ed} \equiv M \pmod{n}$, thus decrypting the message.

Wait...nothing was said super quietly, nothing was redacted, and everything was sent publicly. Still, only Person A is able to decrypt the message since to decrypt the message, one must know d. To know d, one must know L(n)=(p-1)(q-1), which means knowing the prime factors of n. Factoring huge semi-primes is HARD.

Wanna give our messaging system a try?

e=673096023231379724170793534639467508734821457
n=566950058968051540997142079714186971418905409791 4409