Discrete Math

Taught by Po-Shen Loh, Spring 2019

Scribe:
Nathan Moss

Last Updated:
May 16, 2019

Contents

Part

I

Combinatorics

CHAPTER 1

Basic Counting and Induction

One friend from Sesame Street, plus one friend from Sesame Street, equals two friends from Sesame Street! Ah, ha, ha!

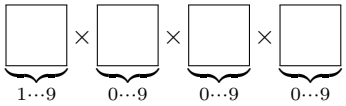
—Count von Count

Contents

1.1 Counting Principles

Example 1.1 (4-Digit Numbers).
How many 4-digit numbers are there?

When we say 4 digits, we exclude those that start with a zero. There are 4 slots for us to fill:

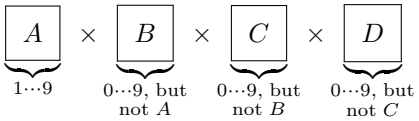


The leftmost digit can be any of $1 \cdots 9$ (since 0's aren't allowed). The remaining digits can be any of $0 \cdots 9$. Notice that no matter our choice of number for previous slots (we'll work left to right, say), the number for the next slot doesn't change.

Therefore, we can multiply the options we have for each slot and conclude there are $9 \times 10 \times 10 \times 10 = 9000$ 4-digit numbers. ☺

Example 1.2 (4-digit numbers with a twist).
How many 4-digit numbers are there where no side by side digits are equal?

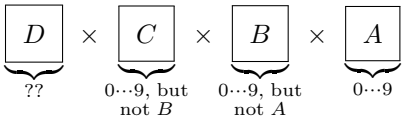
Again, there are 4 slots to fill:



We have 9 options for A, and then 9 options for B, C, and D each, since each has 10 possible digits, and one is excluded. Here, it is important that we work left to right! In total, we get that there are $9 \times 9 \times 9 \times 9 = 6561$ such numbers. ☺

Note 1.3 (Working Right-to-Left).
In the above example, we implicitly worked from left-to-right. What if we had tried to work right to left on the previous problem?

There would still be 4 slots to fill:



We have a problem at D. It would be nice if we could say that there were 8 options; the digits $1 \cdots 9$ and exclude the digit in box C. But what if C is a zero? Then there are 9 options! We can do two things from here:

(1) We can put bounds on the answer. There are always at least 8 options for the last digit, and at most 9, so we can say

$$10 \times 9 \times 9 \times 9 \times 8 \leq \text{ANS} \leq 10 \times 9 \times 9 \times 9 \times 9$$

Notice that this agrees with our answer from before. Indeed $\text{ANS} = 6561$ satisfies $6480 \leq \text{ANS} \leq 7290$.

(2) We can reduce the problem to an "easier" one. Notice that we get

$$\text{ANS} = \left(\#\text{ of } \underbrace{\boxed{0}\boxed{\text{any}}\boxed{\text{any}}}_{\substack{\text{no equal numbers}\\ \text{next to each other}}} \right) \times 9 + \left(\#\text{ of } \underbrace{\boxed{1\ldots9}\boxed{\text{any}}\boxed{\text{any}}}_{\substack{\text{no equal numbers}\\ \text{next to each other}}} \right) \times 8$$

But then we have to case again to solve these smaller problems! For this problem, this is overkill, but keep this strategy in mind.

Enough with this problem for now, let's move on to an even harder one! ☺

Example 1.4 (4-digit numbers with another twist).
How many 4-digit numbers are there with no equal digits next to each other which are also even?

Take a minute to try this yourself.

You'll quickly see that all options (left-to-right, right-to-left, some other order) lead to some messy casing. Unfortunately, there is no elementary counting method to solve this problem! Let's work through the cases. To stay organized, we'll work bottom up:

- How many 1-digit numbers (not starting with zero) are there where no side-by-side digits are the same? Clearly, there are 9. But, let's split them up into how many are even, and how many are odd:

$$\text{Even}_{\text{1-digit}} = 4 \qquad \text{and} \qquad \text{Odd}_{\text{1-digit}} = 5$$

- Now, how many 2-digit numbers, broken up by even and odd:
  - Let's find E_2 first. The 10's digit must be either even or odd. If it is even, then the number looks like

$$\underbrace{\boxed{}}_{\text{E}_1}\boxed{\text{even}} \implies \text{E}_{2\ (\text{even})} = 4 \times \text{E}_1$$

    since the last number can be any of $0, 2, 4, 6, 8$, but not the previous one. We know that one of these is ruled out, since we are considering the case when the leading box is even. Similarly, if we make the leading box odd:

$$\underbrace{\boxed{}}_{\text{O}_1}\boxed{\text{even}} \implies \text{E}_{2\ (\text{odd})} = 4 \times \text{O}_1$$

    since an odd number will never "collide" with an even number. Therefore, it total, we get $E_2 = \text{E}_{2\ (\text{even})} + \text{E}_{2\ (\text{odd})} = 4 \times E_1 + 5 \times O_1$.
  - Now, to find O_2. Again, case on the parity of the ten's digit to get:

$$\text{O}_{2\ (\text{even})} = 5 \times E_1 \qquad \text{and} \qquad \text{O}_{2\ (\text{odd})} = 4 \times O_1$$

    so in total, we get $O_2 = \text{O}_{2\ (\text{even})} + \text{O}_{2\ (\text{odd})} = 5 \times E_1 + 4 \times O_1$.

- However, note that the above argument generalizes. We can easily replace the squares with brackets under them by longer numbers with issues. Thus, we get the more general constraints:

$$E_1 = 4, O_1 = 5, E_{i+1} = 4E_i + 5O_i, \text{ and } O_{i+1} = 5E_i + 4O_i$$

These can be solved out by hand. Here's a table of the first few values:

i	E_i	O_i
1	4	5
2	41	40
3	364	365
4	3281	3280

The number we are looking for is $E_4 = 3281$ such numbers. ☺

1.2 More Counting and Induction

Counting Grids and Pascal's Triangle

Theorem 1.5 (Arrangements of N distinct objects).
N objects can be arranged in $N!$ ways.

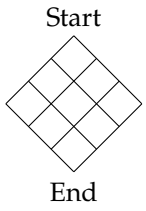
Proof. Working left-to-right, we have N options for the first object, $N-1$ for the second, and so on. We multiply to get $N!$. □

Example 1.6 (Paths Through a Grid).

Consider the network of roads pictured to the right.

Say we are only allowed to make downwards movements (either down-left or down-right).

Under these restrictions, how many ways are there to get from the top of the grid to the bottom?



Notice that we have to make three down-left movements (hereafter called an L), and three down-right movements (called an R). Thus, the number of paths is the number of ways to arrange 3 L's and 3 R's. How many ways are there to do this?

Theorem 1.7 (Arrangements with Duplicates).
Consider a group of N objects which include n_1 identical objects of type 1, n_2 identical objects of type 2, $\cdots$ and n_k identical objects of type k.

The number of permutations of these objects is

$$\frac{N!}{n_1! \times n_2! \times \cdots \times n_k!}$$

Proof. Index the elements in each group so that they are distinct. For example, for the type 1 objects, we might label them $n_1^{(1)}, n_1^{(2)}, \cdots, n_1^{(j_1)}$. Then, arrange these newly distinct objects. There are $N!$ ways to do this.

However, the original objects were not distinct, so we end up over-counting. By how much do we over-count? We'll deal with each type of object separately. Fix the positions of every other object:

$$\_\ \_\ A_1\ A_2\ \_\ A_3\ \_$$

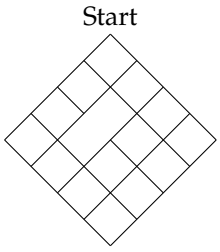
The above position is in a way "equivalent" to

$$\_\ \_\ A_2\ A_1\ \_\ A_3\ \_$$

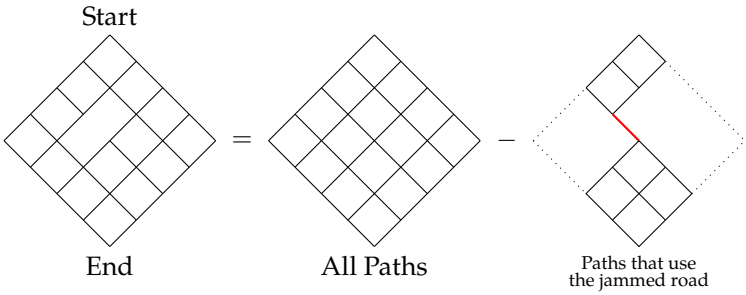
In fact, any arrangement of this form is equivalent if the only thing we do is permute the position of the A's. In this example, there are 3! ways to do this. In general, for objects of type-i, there are $n_i!$ ways to do this. Thus, we have to divide out by the amount we over-count by for each type of object, giving the above formula. □

Note 1.8 (Returning to the Grid Problem).
With this in hand, we see that there are $\frac{6!}{3!\times 3!}$ ways to arrange 3 L's and 3 R's, and thus that many paths through the grid.

Example 1.9 (Grid with a Traffic Jam).
Using the same rules, how many paths are there through the below grid?



Notice we can find the number of routes as follows:



Using the idea in the picture above, we have

- *Total Number of Paths*. As in the original grid example, there are $\frac{8!}{4!\times 4!}$ total paths through the grid. Here, we make the additional observation that we can equivalently find this by drawing 8 lines, and choosing the position of the 4, say L's.

  Thus, this is also equal to $\binom{8}{4}$. This is a convenient way of thinking about this.

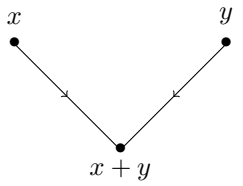
- *Number of Paths through the Jammed Road.* We have that

$$\begin{aligned}\text{Paths}_{(\text{total})} &= \text{Paths}_{(\text{Start}\to\text{Jammed Road})} \times \text{Paths}_{(\text{Jammed Road}\to\text{End})} \\ &= \binom{3}{1} \times \binom{4}{2}\end{aligned}$$

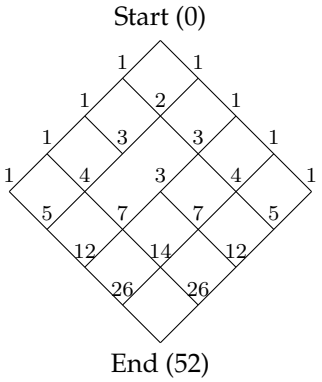
  The first equality is justified because every path uses the jammed road, and the top path is independent to the bottom path. The second uses the same logic as we've been using above.

Thus, in total, there are $\binom{8}{4} - \binom{3}{1}\binom{4}{2}$ paths through the original grid. ☺

There is another way to solve this problem: by "brute-forcing" the question. At each intersection in the grid, we will write the number of ways to get from the start to each point. The start point will have an implied number of 0, and every other intersection will be calculated by



If a path is missing (for example, on an edge), just ignore it. Proceeding in this way for the grid in the previous example, we get the following:



Fortunately, this agrees with our previous answer: $\binom{8}{4} - \binom{3}{1}\binom{4}{2} = 52$. ☺

Definition 1.10 (Pascal's Triangle).
Pascal's Triangle is based off of the constructions seen in this section, where the two edges are all 1's, and the rest of the numbers are calculated as the sum of the two numbers above it. It continues indefinitely. For example, here are the first 5 rows of Pascal's Triangle:

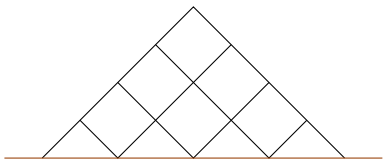
$$\begin{array}{ccccccccc} & & & & 1 & & & & \\ & & & 1 & & 1 & & & \\ & & 1 & & 2 & & 1 & & \\ & 1 & & 3 & & 3 & & 1 & \\ 1 & & 4 & & 6 & & 4 & & 1 \end{array}$$

Important 1.11 (Pascal's Triangle and Binomial Coefficients).
From our discussion above, we also know that Pascal's Triangle can also be written using binomial coefficients, as:

$$\begin{array}{ccccccccc} & & & & \binom{0}{0} & & & & \\ & & & \binom{1}{0} & & \binom{1}{1} & & & \\ & & \binom{2}{0} & & \binom{2}{1} & & \binom{2}{2} & & \\ & \binom{3}{3} & & \binom{3}{2} & & \binom{3}{1} & & \binom{3}{0} & \\ \binom{4}{0} & & \binom{4}{1} & & \binom{4}{2} & & \binom{4}{3} & & \binom{4}{4} \end{array}$$

Proposition 1.12 (Sum of the nth Row of Pascal's Triangle).
For any non-negative integer n, $\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n} = 2^n$, or $\sum_{k=0}^{n} \binom{n}{k} = 2^n$.

Proof. We prove this by *counting in two ways*. Consider half of an $n \times n$ grid without holes. We show the number of ways to get to the bottom is both of these expressions.



An Example with $n = 4$

- One method is to randomly choose n options of left and right (note, where on the blue line we end up doesn't matter, so after any n moves, we're done). At each step, we have 2 options, and we make n decisions. Thus, there are 2^n ways to get to the bottom.
- However, we could also count the number of ways to get to each point at the bottom, and add them up. From above, we know that the number of ways to get to the points are the numbers $\binom{n}{0}, \cdots, \binom{n}{n}$, so in total we have $\sum_{k=0}^{n} \binom{n}{k} = 2^n$ ways to get to the bottom.

Since these numbers count the same set, they must be equal. □

A (Brief) Review of Induction

Example 1.13 (Back to 4-digit Numbers).
Recall the problem in Exercise 1.4, where we found the relations

$$E_1 = 4, O_1 = 5, E_{i+1} = 4E_i + 5O_i, \text{ and } O_{i+1} = 5E_i + 4O_i$$

and some values of the sequence to be

i	E_i	O_i
1	4	5
2	41	40
3	364	365
4	3281	3280

However, it's kind of a pain to calculate these values with the recurrences. Can we find an explicit formula? Based on the values we calculated, we may guess that

$$E_n = \frac{9^n + (-1)^n}{2} \qquad \text{and} \qquad O_n = \frac{9^n - (-1)^n}{2}$$

Proof. First, note it is sufficient to prove only the expression for E_n, since then the other follows since $E_n + O_n = 9^n$. By induction on n.

- *Base Case.* When $n = 1$, $E_n = \frac{9^1+(-1)^1}{2} = 4$. Hooray!
- *Induction Step.* Assume the claim holds for n.

$$\begin{aligned} E_{n+1} &= 4E_n + 5O_n = 4E_n + 5(9^n - E_n) = 5 \cdot 9^n - E_n \\ &= 5 \cdot 9^n - \frac{9^n + (-1)^n}{2} = \frac{9^{n+1} + (-1)^{n+1}}{2} \end{aligned}$$

  which is what we wanted.

By induction, the claim holds for all n. □

1.3 Applications of Linear Algebra

Recall the problem of finding the number of n-digit numbers with no adjacent numbers equal that were even/odd. We had the relations:

$$\begin{aligned} E_{n+1} &= 4 \times E_n + 5 \times O_n \\ O_{n+1} &= 5 \times E_n + 4 \times O_n \end{aligned}$$

which solved to the explicit formulas $E_n = \frac{9^n+(-1)^n}{2}$ and $O_n = \frac{9^n-(-1)^n}{2}$. However, the last proof of this with induction was somewhat unsatisfying. Induction proofs typically are. Let's try and use matrices!

$$\begin{bmatrix} E_{n+1} \\ O_{n+1} \end{bmatrix} = \begin{bmatrix} 4 & 5 \\ 5 & 4 \end{bmatrix} \begin{bmatrix} E_n \\ O_n \end{bmatrix} \qquad \text{and} \qquad \begin{bmatrix} E_1 \\ O_1 \end{bmatrix} = \begin{bmatrix} 4 \\ 5 \end{bmatrix}$$

Notice that we can write a general term of the sequence by doing this multiplication many times. This gives an expression for E_n and O_n:

$$\begin{bmatrix} E_n \\ O_n \end{bmatrix} = \begin{bmatrix} 4 & 5 \\ 5 & 4 \end{bmatrix}^{n-1} \begin{bmatrix} 4 \\ 5 \end{bmatrix}$$

Definition 1.14 (Eigenvector/Eigenvalue).
From linear algebra, you should recall that powers of matrices are intimately related to eigenvalues and eigenvectors. Recall that if

$$A\vec{v} = \lambda\vec{v} \qquad \text{and} \qquad \vec{v} \neq \vec{0}$$

then we say $\vec{v}$ is an eigenvector of A with corresponding eigenvalue λ.

Proposition 1.15 (Eigenvector of Matrix with Fixed Row Sum).
The rows of a matrix sum to a value c if and only if $[1, \cdots, 1]$ is an eigenvector with corresponding eigenvalue c.

Proof. We won't give a formal proof, but rather an example which is illustrative. In our current problem

$$\begin{bmatrix} 4 & 5 \\ 5 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 9 \\ 9 \end{bmatrix} = 9 \times \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

so the relevant 2×2 matrix has fixed row sums of 9, so $\left[\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right]$ is an eigenvector with corresponding eigenvalue 9. □

Proposition 1.16 (Eigenvectors of a Real, Symmetric Matrix).
Given a matrix $A \in \mathbb{R}^n$, the eigenvectors of A are:

1. all real; and
2. orthogonal

Definition 1.17 (Trace of a Matrix).
The sum of the diagonal entries of a (square) matrix A is called the trace of A, and is denoted by $\operatorname{tr} A$.

Proposition 1.18 (Eigenvalues, Traces, and Determinants).
For a square matrix A, we have

1. $\det A =$ product of the eigenvalues of A
2. $\operatorname{tr} A =$ sum of the eigenvalues of A

The above facts can all be used without proof in this class (we won't prove them).

With this in mind, let's shift the 4-digit problem to looking at what fraction of 4-digit numbers are even/odd. Define:

$$e_n = \begin{array}{l}\text{fraction of all } n\text{-digit numbers which are even} \\ \text{with no same digit next to each other}\end{array} = \frac{E_n}{9^n}$$

$$o_n = \begin{array}{l}\text{fraction of all } n\text{-digit numbers which are odd} \\ \text{with no same digit next to each other}\end{array} = \frac{O_n}{9^n}$$

Then, our relations (in matrix form) become:

$$\begin{bmatrix} e_{n+1} \\ o_{n+1} \end{bmatrix} = \begin{bmatrix} 4/9 & 5/9 \\ 5/9 & 4/9 \end{bmatrix} \begin{bmatrix} e_n \\ o_n \end{bmatrix} \qquad \text{and} \qquad \begin{bmatrix} e_1 \\ o_1 \end{bmatrix} = \begin{bmatrix} 4/9 \\ 5/9 \end{bmatrix}$$

There are many ways to see this. One is to divide our original recurrences by 9^n. Then, as before, we combine these to get

$$\begin{bmatrix} e_n \\ o_n \end{bmatrix} = \begin{bmatrix} 4/9 & 5/9 \\ 5/9 & 4/9 \end{bmatrix}^{n-1} \begin{bmatrix} 4/9 \\ 5/9 \end{bmatrix}$$

Note 1.19 (Markov-Chains).

The matrix $\left[\begin{smallmatrix} 4/9 & 5/9 \\ 5/9 & 4/9 \end{smallmatrix}\right]$ is particularly nice because its rows and columns sum to one (we call such a matrix a *doubly stochastic matrix*). This allows us to think of the entries as probabilities.

For example, the top left entry can be thought of as the probability of becoming even after transition given that the current entry is even, and the bottom left entry can be thought of as probability of becoming even given currently odd, etc.

A system which behaves in this way (i.e. one where the chances of switching between states can be encapsulated in a matrix like this) is called a Markov Chain.

Now, all that's left to finish the problem is to do the algebra. Since the sums of the rows are all equal, 1 is an eigenvalue with corresponding eigenvector $\left[\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right]$. For the other eigenvector, we know it must be orthogonal, so try $\left[\begin{smallmatrix} 1 \\ -1 \end{smallmatrix}\right]$:

$$\begin{bmatrix} 4/9 & 5/9 \\ 5/9 & 4/9 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} -1/9 \\ 1/9 \end{bmatrix} = -\frac{1}{9} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \checkmark$$

so $\left[\begin{smallmatrix} 1 \\ -1 \end{smallmatrix}\right]$ is an eigenvector with eigenvalue $-1/9$. Using this as an eigenbasis, write:

$$\begin{bmatrix} 4/9 \\ 5/9 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \frac{1}{18} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

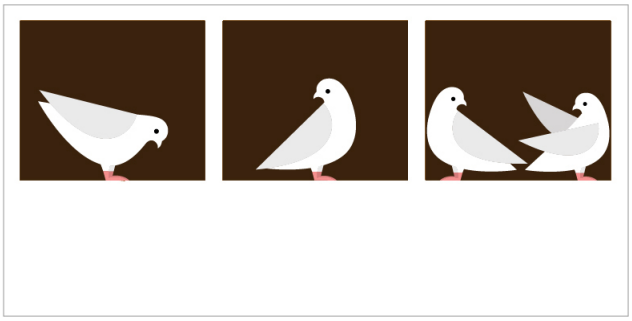
so that we get:

$$\begin{aligned} \begin{bmatrix} 4/9 & 5/9 \\ 5/9 & 4/9 \end{bmatrix}^{n-1} \begin{bmatrix} 4/9 \\ 5/9 \end{bmatrix} &= \begin{bmatrix} 4/9 & 5/9 \\ 5/9 & 4/9 \end{bmatrix}^{n-1} \left(\frac{1}{2} \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \frac{1}{18} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \right) \\ &= \frac{1}{2} \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \frac{1}{18} \left(-\frac{1}{9} \right)^{n-1} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \\ &= \begin{bmatrix} \frac{1}{2} + \frac{1}{2} \left(-\frac{1}{9} \right)^{n} \\ \frac{1}{2} - \frac{1}{2} \left(-\frac{1}{9} \right)^{n} \end{bmatrix} \end{aligned}$$

which when manipulated become $\frac{1}{9^n} \left(\frac{9^n + (-1)^n}{2} \right)$ and $\frac{1}{9^n} \left(\frac{9^n - (-1)^n}{2} \right)$, confirming our formulas for E_n and O_n. ☺

CHAPTER 2

Inclusion-Exclusion and the Pigeonhole Principle



The name "Pigeonhole Principle" actually comes from the fact that pigeonhole also means:

> **Pigeonhole.** (noun) a small open compartment (as in a desk or cabinet) for keeping letters or documents

It has nothing to do with pigeons or holes. It really ought to be called the "Mailbox Principle."

—Po-Shen Loh

Contents

2.1 Dirichlet's Theorem

π is well-known to be an irrational number. In particular,

$$\pi \approx 3.1415926...$$

However, there are some rational numbers which serve as a "good" approximation to pi. One better known one is $22/7 = 3.\overline{142857}$, and a perhaps lesser known one is $355/113 = 3.1415929...$ For how big the denominators of these fractions are, they do an excellent job of approximating π.

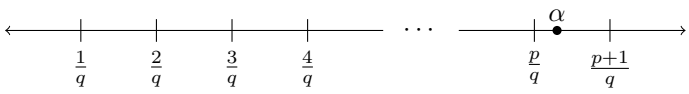
Note 2.1 (On small denominators).
Somehow these fractions are more "impressive" approximations than others. For example, the approximation $314/100$ is unremarkable.

We can make this more mathematical by observing that $314/100$ only gives 2 digits of accuracy for 3 digits in the denominator, while a fraction like $22/7$ gives the same accuracy (actually slightly better) with a third of the digits in its denominator.

Proposition 2.2 (Rational Approximation to a Real Number).
For any real number α, there are infinitely many ways to write $\alpha \approx \frac{p}{q}$ such that the approximation error is less than $\frac{1}{q}$, or more mathematically, such that

$$\left|\alpha - \frac{p}{q}\right| \leq \frac{1}{q}$$

Proof. For any $q \in \mathbb{Z}^+$, we can partition the number line as follows:



and since α has to fall somewhere, just pick the highest $\frac{p}{q}$ less than α. Since there are infinite positive integers, there are infinite of such rational approximations. □

Theorem 2.3 (Dirichlet's[a] Approximation Theorem).
For every real number $\alpha \in \mathbb{R}$, there are infinitely many rational approximations $\frac{p}{q} \approx \alpha$ such that

$$\left|\alpha - \frac{p}{q}\right| \leq \frac{1}{q^2}$$

[a] Pronounced "deer-eh-shleigh"

Example 2.4 (Dirichlet approximation to π).
The fractions $\frac{22}{7}$ and $\frac{355}{113}$ are Dirichlet approximations to π since

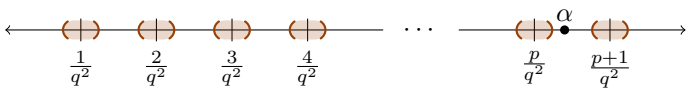
$$\left|\pi - \frac{22}{7}\right| \leq \frac{1}{7^2} \qquad \text{and} \qquad \left|\pi - \frac{355}{113}\right| \leq \frac{1}{113^2}$$

Note 2.5 (How much better is q^2?).
Notice that the only difference between Dirichlet's theorem is that we get q^2 error rather than q. This turns out to be really powerful! For example:

$$(0.01)^2 = 0.0001$$

so if we had an error of at most 0.01 with our original rough approximation, we can get an error of at most 0.0001 using Dirichlet's Theorem. This roughly *doubles* the number of correct digits! That's great!

Why is this harder to prove than our original proposition? Let's consider a modified number line for the new problem:



Notice that in order to be able to approximate alpha with our original strategy, it must lie in one of the shaded regions above, so this strategy will not be able to approximate the α pictured above for this given q. It follows that we can't do this for every q:

Example 2.6 (A (q, α) pair we can't approximate).
Take $\alpha = \sqrt{2}$ and $q = 10$. Since $\sqrt{2} \approx 1.414$, it is further away than 0.01 away from $14/10$ and $15/10$, so we cannot use $q = 10$ to approximate $\sqrt{2}$.

Now, to actually start to prove Dirichlet's Theorem, we'll come up with a stronger statement which we will show to be equivalent to the original:

Proposition 2.7 (Lemma for Dirichlet's Theorem).
For any $\alpha \in \mathbb{R}$ and $N \in \mathbb{Z}^+$, there are integers p and q with $1 \leq q \leq N$ and

$$\left|\alpha - \frac{p}{q}\right| \leq \frac{1}{qN}$$

Why does this imply Dirichlet's Theorem? We have to reconcile two things:

(1) *Where did our $\frac{1}{q^2}$ go?*

Our lemma gives that we get a $\frac{p}{q}$ with error at most $\frac{1}{qN}$. But then

$$\text{error} \leq \frac{1}{qN} \leq \frac{1}{q^2} \qquad (\text{since } N \geq q)$$

so we immediately get an approximation with error $\frac{1}{q^2}$.

(2) *Where did our infinitely many go?*

Idea: I can use the proposition on the same α with infinitely many $N \in \mathbb{Z}^+$. For example:

$$\left.\begin{array}{rcl} \alpha = \sqrt{2}, N = 1 & \longrightarrow & \boxed{\begin{array}{c} p_1/q_1 \\ p_2/q_2 \\ \vdots \end{array}} \begin{array}{l} \in \mathbb{Q} \\ \in \mathbb{Q} \\ \end{array} \\ \alpha = \sqrt{2}, N = 2 & \longrightarrow & \\ \vdots & & \end{array}\right\} \text{within } \pm\tfrac{1}{q_i^2} \text{ of } \alpha$$

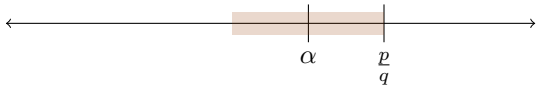
If we call the boxed set of numbers (⋆), then it suffices to show that some infinite subset of that set is pairwise distinct. Let's go ahead and prove this:

Proof. First note that Dirichlet's Theorem is easy if $\alpha \in \mathbb{Q}$, since

$$\alpha = \frac{a}{b} \quad \rightsquigarrow \quad \frac{a}{b}, \frac{2a}{2b}, \frac{3a}{3b}, \cdots \text{ are all perfect approximations}$$

with zero error. This gives our necessary infinite approximations. Therefore, for the rest of the proof, assume that α is irrational.

Assume towards a contradiction that there are only finitely many distinct approximations in (⋆). This implies that there is a closest approximation, say $\frac{p}{q}$



i.e. In the picture above, there are no elements of (⋆) which are in the shaded region. Since N can be any positive integer, there exists an N such that

$$\left|\alpha - \frac{p}{q}\right| \leq \frac{1}{N}$$

and by the proposition (which we have yet to prove, but we will), we get that

$$\left|\alpha - \frac{p'}{q'}\right| \leq \frac{1}{q'N} \leq \frac{1}{N}$$

and our new approximation $\frac{p'}{q'}$ will therefore be within the shaded region, which contradicts the assumption that $\frac{p}{q}$ is the closest approximation. □

Now that we've shown that proposition 2.7 implies Dirichlet's Theorem, all that's left now is to prove the proposition. The proof relies on the following "fact":

Definition 2.8 (Pigeonhole Principle).
If n items are put inside m containers, and $n > m$, then there must be at least one container with more than one item.

Because of this proof we are about to outline, the pigeonhole principle is also sometimes called Dirichlet's Box Theorem.

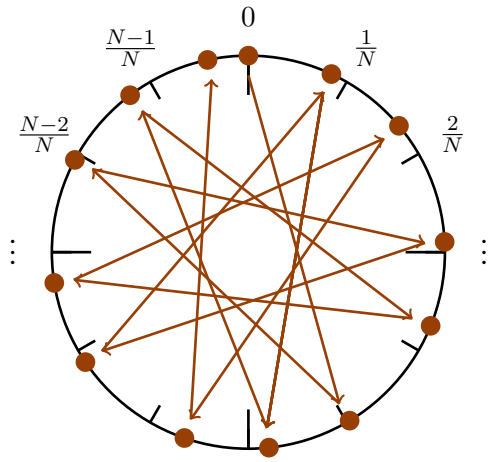
There's not really a proof of the PHP... it just kind of falls out of how we define the idea of "greater than." Hopefully it's believable.

Now, for the proof of our proposition.

Proof. To begin, since $q \neq 0$, we can multiply through by q and prove the equivalent statement:

$$|\alpha q - p| \leq \frac{1}{N}$$

We construct the "clock" $\mathbb{R}/\mathbb{Z}$, and add the numbers $0\alpha, 1\alpha, \cdots, N\alpha$ onto the clock, and subdivide it into intervals of length $1/N$:



There are N regions, and $N+1$ numbers, so there must be an interval with two numbers $x\alpha$ and $y\alpha$ in it. But notice that we can subtract 1 from x and y without changing the size of the gap between them, so we will "walk" the numbers down until one of them is zero. WLOG, assume $x > y$. Then

$$|x\alpha - y\alpha| \leq \frac{1}{N} \text{ and } |x\alpha - y\alpha| = |(x-y)\alpha - 0\alpha| \implies |(x-y)\alpha| \leq \frac{1}{N}$$

and so $(x-y)\alpha$ is within $1/N$ of some integer, which is what we wanted. □

2.2 More with the Pigeonhole Principle

Theorem 2.9 (Erdős-Szekeres Theorem).
In any sequence of N distinct real numbers, there is always a monotone subsequence of length $\geq \sqrt{N}$.

A *monotone subsequence* is just some subset of the numbers which keeps the original order and is either strictly increasing or strictly decreasing. Some examples are:

Example 2.10 (Monotone Subsequences).
Consider the sequence of numbers:

```
3  1  4  5  9  2  6
      4  5        6   is a monotone subsequence
3                 2   is a monotone subsequence
3  1                  is a monotone subsequence
```

Proof. The proof of this is actually surprisingly short! Think about how we might find a monotone subsequence:

	3	1	4	5	9	2	6
Length of the longest *increasing* subsequence ending here	1	1	2	3	(4)	2	(4)
Length of the longest *decreasing* subsequence ending here	1	2	1	1	1	2	2

Do you notice a pattern? Each cell in the *increasing* row is given by the largest length corresponding to a previous smaller number, plus 1 (and similarly for the *decreasing* row). This actually gives us an efficient way to find a monotone subsequence.

However, we will focus on another fact: all the columns in *any* table will be distinct! Why is this? Consider an arbitrary two rows:

	X	···	Y	

Since every number is distinct, either $\texttt{X} > \texttt{Y}$ or $\texttt{Y} > \texttt{X}$. In the first case, the number in the top row is guaranteed to go up by at least 1, and in the second case, the number in the bottom row is guaranteed to go up by at least 1. This comes from our algorithm for filling in the table. Thus, the two columns are distinct.

Let L denote the length of the longest monotone subsequence. Therefore

$$\#\text{ columns} \leq L^2 \implies N \leq L^2 \implies \sqrt{N} \leq L$$

which is what we wanted. □

Note 2.11 (Use of the Pigeonhole Principle).
Notice that we didn't ever explicitly mention the pigeonhole principle. However, we still used the philosophy of

"Too much stuff, not enough space."

in the proof. Often, this is how the PHP ends up being used.

Theorem 2.12 (Erdős-Szekeres Theorem, Again[a]).
For any positive integers s, t, any sequence of distinct real numbers with

$$\text{length} \geq (s-1)(t-1)+1$$

has a monotone increasing sequence of length s or a monotone decreasing sequence of length t.

[a] In other texts, the Erdős-Szekeres Theorem is typically phrased like this.

Note 2.13 (This statement implies the original one).
From this statement of the Erdős-Sekeres Theorem, we can recover the original one. To ensure there is *some* monotone subsequence of length L, this theorem only needs $(L-1)^2+1$ numbers.

This is even slightly better then the L^2 bound we had with the previous theorem!

Proof. We are given $(s-1)(t-1)+1$ numbers which are distinct. Assume towards a contradiction that all increasing subsequences have length $1, 2, \cdots (s-1)$ and all decreasing subsequences have length $1, 2, \cdots, (t-1)$.

In the table, any given column has

$$\begin{array}{|c|} \hline \\ \hline \\ \hline \end{array} \begin{array}{l} \longleftarrow \{1, 2, \cdots (s-1)\} \\ \longleftarrow \{1, 2, \cdots (t-1)\} \end{array} \implies (s-1)(t-1) \text{ options}$$

By the pigeonhole principle, we know that two of the columns will be the same, since there are $(s-1)(t-1)+1$ columns, but only $(s-1)(t-1)$ options. However, by the same logic as in the previous proof, we know that no two columns can be the same, a contradiction. □

2.3 Algorithms and Efficiency

The algorithm we used in the above proofs is actually an efficient one. But how good is it? That is, about how many steps does it take to find the length of the longest monotone subsequence in N distinct numbers? First, some terminology:

Definition 2.14 (Efficient Algorithm).
We say an algorithm is *efficient* if the number of steps it takes can be upper bounded by some polynomial.

The variable in the polynomial is generally some notion of the "size" of the input. For example, in the problem we're currently discussing, we would want some bound in terms of N.

Note 2.15 (Big-O Notation).
This notation can be made precise with big-O notation. We say a function $f(x)$ is $O(g(x))$ if there exist constants $C > 0$ and $x_0 > 0$ such that

$$f(x) \leq Cg(x) \qquad \text{for all } x \geq x_0$$

and then that an algorithm is efficient if the number of steps it takes is $O(p(x))$ for some polynomial $p(x)$. We generally won't do this in this class.

Example 2.16 (Monotone-Subsequence is Efficient).
The algorithm discussed in the proof of the Erdős-Szekeres Theorem is an efficient algorithm for computing the longest monotone subsequence.

To show that this is true, we have to make more precise the procedure we are taking. We use the following algorithm to compute the longest monotone subsequence:

$\mathcal{S}$: a sequence of distinct real numbers
N: length of $\mathcal{S}$

LONGEST-MONOTONE($\mathcal{S}$):

1. Construct a $2 \times N$ table which looks like:

$\mathcal{S}_1$	$\mathcal{S}_2$	$\cdots$	$\mathcal{S}_{N-1}$	$\mathcal{S}_N$

2. Set the first column to $\begin{array}{|c|}\hline 1 \\ \hline 1 \\ \hline \end{array}$.
3. For every cell in the first row:
   a. Let $M =$ the max of the previous cells in the top row that have a *smaller* column label.
   b. Set the current cell to $M + 1$.
4. For every cell in the second row:
   a. Let $M =$ the max of the previous cells in the top row that have a *bigger* column label.
   b. Set the current cell to $M + 1$.
5. Scan through the table, and return the largest number.

Take a minute and convince yourself that this algorithm is exactly what we described in the proof. The advantage to this is that we can go back through and describe the cost of each step:

- [Step 1] We need to add a column with two cells for each number in the sequence, so this will take $\approx N$ steps.
- [Step 2] This will take ≈ 1 step
- [Step 3] For any given cell, step 3a will take up to N steps, since we could have to check every previous cell in the row. Step 3b will take ≈ 1 step. Since we have to do this for every cell in the row, step 3 will take $\approx N^2$ steps.
- [Step 4] This is basically just step 3 again, so this will also take $\approx N^2$ steps.
- [Step 5] We have to look at every cell in the table, so this will take $\approx N$ steps.

Note 2.17 (On the abundance of $\approx$'s).
Why are we only saying $\approx X$ number of steps? Well, we don't really have a good notion for what a "step" is, so we can't really afford to care deeply about constants. For example, we'll say $2N$ steps and N steps are about the same.

However, it feels like there should be a meaningful difference between N steps and N^2 steps, regardless of how we define a step.

If the argument for step 3 makes you feel a little uncomfortable, that's great! It is a bit hand-wavy. Let's make it more precise:

- The first cell takes 1 step to fill. Just write the number 1.
- The second cell takes ≈ 2 steps to fill. We have to check the previous 1 cell, then do some arithmetic.
- In general, cell i takes $\approx i$ steps to fill, since we have to check the previous $i-1$ cells, then do some arithmetic. Therefore, to fill the whole row, we will do

$$1 + 2 + \cdots + (N-1) + N = \frac{N(N-1)}{2} \approx N^2 \text{ steps}$$

In general, things that form a "triangle" like this will take $\approx N^2$ steps because of precisely this reasoning.

Now, it's easy to see that the quadratic steps will dominate the total number of steps, so the whole algorithm is N^2. Based on our definition of efficient, we conclude that this is in fact efficient. ☺

Example 2.18 (A Naïve Algorithm).
The "easy" algorithm to think of for the longest monotone subsequence problem is not efficient.

Without this clever approach, we may have thought to solve the problem as:

> $\mathcal{S}$: a sequence of distinct real numbers
> N: length of $\mathcal{S}$
>
> LONGEST-MONOTONE($\mathcal{S}$):
> 1 List all the subsequences of the N numbers.
> 2 Go through all of them and check if they are monotone.
> a If it is, compare it with the current largest, and keep the larger one.
> 3 Return the largest one found.

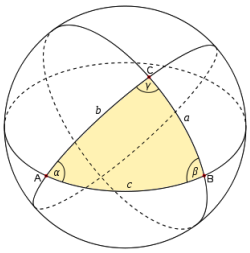
We notice right away that step 1 involves constructing 2^N subsequences, so the algorithm is immediately not efficient. Step 2 is even worse, since chceking if a subsequence is monotone takes $\approx N$ steps, so step 2 will take $\approx N \times 2^N$ steps.

We say that this algorithm takes *exponential time*, and it is not efficient. ☺

2.4 Spherical Geometry and Inclusion- Exclusion

Before the days of the SAT, the following question appeared on a CMU entry exam:

Example 2.19 (Spherical Area).
Find the area of the shaded triangle given that the angles are $100°$–$80°$–$60°$and that the radius of the sphere it is drawn on is 10.



This was intended as an easy problem, and was a problem type taught to high schoolers, so hopefully there's a straightforward answer! Let's try and build up some intuition for the problem:

Angles	Description	Surface Area
$90°$–$90°$–$90°$	An 8^{th} of the sphere	$1/8$
$60_+°$–$60_+°$–$60_+°$	A tiny, basically equilateral triangle	≈ 0
$300_+°$–$300_+°$–$300_+°$	Everything but a tiny, equilateral triangle	≈ 1
$180°$–$\theta°$–$\theta°$	A slice of the sphere	$\theta/360$

Using these (and you can try other ones too), we might guess:

$$\text{Area (Fraction)} = \frac{\text{sum of the angles} - 180}{720}$$

To do this, we'll use the *Inclusion-Exclusion Principle*:

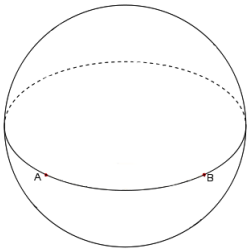
Theorem 2.20 (Inclusion-Exclusion, with 3 sets).
Let A, B, C be sets. Then

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |B \cap C| - |C \cap A| + |A \cap B \cap C|$$

Note 2.21 (A proof?).
We won't prove this right now, but we'll prove a more general version of Inclusion-Exclusion later.

First, some more info on spherical geometry:

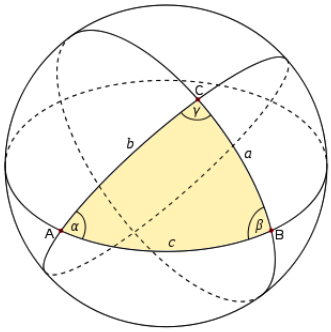
- The shortest path between two points is along what is called a "great circle":



  The great circle of two points can be found by considering the intersection of the sphere and the plane through the two points and the center of the sphere.

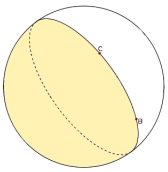
- The center of a great circle coincides with the center of the sphere.

Let's go ahead and prove this fact now:

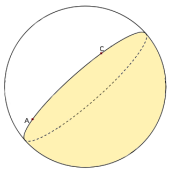


Divide the sphere's surface area as follows:

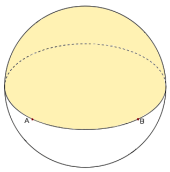
Set A:



Set B:



Set C:



We are interested in $|A \cap B \cap C|$. Let x be the fraction of the surface area of the sphere that the triangle covers. Now, with inclusion-exclusion, we get:

$$\underbrace{|A \cup B \cup C|}_{1-x} = \underbrace{|A|}_{1/2} + \underbrace{|B|}_{1/2} + \underbrace{|C|}_{1/2} - \underbrace{|A \cap B|}_{\gamma/360} - \underbrace{|B \cap C|}_{\alpha/360} - \underbrace{|C \cap A|}_{\beta/360} + \underbrace{|A \cap B \cap C|}_{x}$$

$$\implies \frac{\alpha+\beta+\gamma}{360} - \frac{1}{2} = 2x \implies \frac{\alpha+\beta+\gamma-180}{360} = 2x \implies x = \frac{\alpha+\beta+\gamma-180}{720} \checkmark$$

which is exactly what we wanted. ☺

Note 2.22 (Getting some of those areas).
Here's an explanation of where some of the areas in the proof came from:

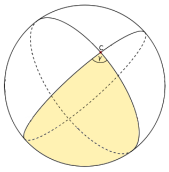
- $|A \cup B \cup C| = 1 - x$:

  Everything on the sphere is symmetric, so in the picture, we see a "mirror" of the triangle on the back of the sphere. $A \cup B \cup C$ involves shading everything but that triangle.

- $|A \cap B| = \frac{\gamma}{360}$:

  This forms a "slice" of the sphere, which goes γ° around the entire sphere.

  The other 2-sets follow from the same logic.



Note 2.23 (Solving the "easy" problem).
With this formula in hand, we see that the solution to the original problem is

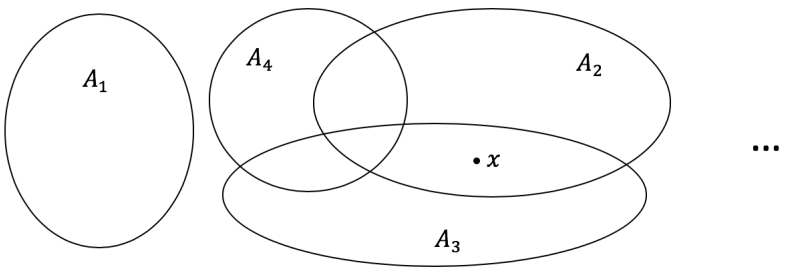
$$\frac{60}{720} \times 4\pi(10^2) = \frac{100\pi}{3}$$

Now, we'll prove the general form of the inclusion-exclusion principle:

Theorem 2.24 (Inclusion-Exclusion Principle).
Let $A_1, A_2, \cdots, A_n$ be sets. Then

$$\begin{aligned} |A_1 \cup A_2 \cup \cdots \cup A_n| = {} & |A_1| + |A_2| + \cdots + |A_n| \\ & - |A_1 \cap A_2| - |A_1 \cap A_3| - \cdots - |A_{n-1} \cap A_n| \\ & + |A_1 \cap A_2 \cap A_3| + \cdots + (\text{all 3-sets}) \\ & \vdots \\ & \pm |A_1 \cap A_2 \cap \cdots \cap A_n| \end{aligned}$$

Proof. We will show that everything in the left-hand side is counted exactly once. Consider an arbitrary element x. Maybe we have the following layout:



Observe that some of the sets will contain x, and others won't. It's easy to see which 1-sets x is in. For a 2-set, x is in it if and only if x is in the component 1-sets of the 2-set. Similarly for the 3-sets and so on.

Let's say that x occurs in m of the sets, with $1 \leq m \leq n$. Then x is counted:

$$\underbrace{\binom{m}{1}}_{\text{1-sets}} - \underbrace{\binom{m}{2}}_{\text{2-sets}} + \underbrace{\binom{m}{3}}_{\text{3-sets}} - \underbrace{\binom{m}{4}}_{\text{4-sets}} + \cdots \pm \underbrace{\binom{m}{m}}_{m\text{-sets}} \mp \underbrace{\binom{m+1}{m} \pm \cdots \pm \binom{m}{n}}_{\text{all} = 0}$$

so all we have to do is show this equals 1 (which implies x is counted exactly once).

Consider the expansion of $0 = (1-1)^m$:

$$\begin{aligned} 0 = (1-1)^m &= \binom{m}{0}(1)^m(-1)^0 + \binom{m}{1}(1)^{m-1}(-1)^1 + \cdots + \binom{m}{m}(1)^0(-1)^m \\ &= \binom{m}{0} - \binom{m}{1} + \binom{m}{2} - \binom{m}{3} + \cdots \pm \binom{m}{m} \end{aligned}$$

Moving all the terms other than $\binom{m}{0}$ to the other side gives:

$$\binom{m}{1} - \binom{m}{2} + \binom{m}{3} - \cdots \pm \binom{m}{m} = \binom{m}{0} = 1$$

which is what we wanted to show. □

2.5 Derangements

Definition 2.25 (Derangement).
A permutation with no fixed points is called a derangement.

Example 2.26 (The Party Hat Problem).
There are n people at a party, each with a different hat. In celebration, they all throw their hats into the air, and catch a random one from the lot (they can get their own again).

What is the probability that nobody gets their own hat back? Equivalently, find the probability that a uniformly random permutation is a derangement.

The probability that this happens is:

$$\text{Prob} = \frac{\#\text{ of derangements}}{\#\text{ of total permutations}} = 1 - \frac{\#\text{ where someone keeps hat}}{n!}$$

and so the problem reduces to finding the number of permutations where someone keeps their hat. Let's use inclusion-exclusion! Let

$$\begin{aligned} A_1 &= \text{set of permutations where person \#1 keeps their own hat} \\ A_2 &= \text{set of permutations where person \#2 keeps their own hat} \\ &\vdots \qquad\qquad\qquad\qquad \vdots \\ A_n &= \text{set of permutations where person \#}n\text{ keeps their own hat} \end{aligned}$$

Now, by inclusion exclusion, we get that

$$\begin{aligned}
&|A_1 \cup \cdots \cup A_n| \\
&\quad = \quad |A_1| + |A_2| + \cdots + |A_n| && \Rightarrow \tbinom{n}{1}(n-1)! = n! \text{ ways} \\
&\qquad - |A_1 \cap A_2| - \cdots - \text{(all 2-sets)} && \Rightarrow \tbinom{n}{2}(n-2)! = \tfrac{1}{2!}(n!) \text{ ways} \\
&\qquad + |A_1 \cap A_2 \cap A_3| + \cdots + \text{(all 3-sets)} && \Rightarrow \tbinom{n}{3}(n-3)! = \tfrac{1}{3!}(n!) \text{ ways} \\
&\qquad\qquad \vdots && \qquad\quad \vdots \\
&\qquad \pm |A_1 \cap \cdots \cap A_n| && \Rightarrow \tbinom{n}{n}(n-n)! = \frac{1}{2!}(n!) = 1 \text{ way}
\end{aligned}$$

How to get the numbers on the right? For the k-sets, pick the k people to keep their hats (in $\binom{n}{k}$ ways), then assign the rest in any order (in $(n-k)!$ ways). Thus

$$\begin{aligned}
|A_1 \cup \cdots \cup A_n| &= n! - \frac{1}{2!} \cdot n! + \frac{1}{3!} \cdot n! - \frac{1}{4!} \cdot n! + \cdots \\
&= n! \left(1 - \frac{1}{2!} + \frac{1}{3!} - \frac{1}{4!} + \cdots \pm \frac{1}{n!}\right)
\end{aligned}$$

Let's first consider the "easy" case: the limiting probability as $n \to \infty$. We get:

$$\text{Prob} = \frac{1}{n!} \left[\underbrace{n! - \left(\frac{n!}{1!} - \frac{n!}{2!} + \frac{n!}{3!} - \cdots \pm \frac{n!}{n!} \right)}_{\text{call this } (\star), \text{ we'll come back to it later}} \right] = 1 - \left(\frac{1}{1!} - \frac{1}{2!} + \frac{1}{3!} - \cdots \pm \frac{1}{n!} \right)$$

but recall from calculus that the Taylor series for e^x is given by

$$\begin{aligned}
e^x &= f(0) + f'(0)x + \frac{1}{2!} f''(0)x^2 + \cdots + \frac{1}{n!} f^{(n)}(0)x^n + \cdots \\
&= 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots
\end{aligned}$$

and so we see that our probability is $e^{-1} \approx 0.3679$ as $n \to \infty$. ☺

Example 2.27 (But wait, you cheated!).
That's all well and good, but what if we wanted the *exact* number of derangements on n people (or by extension, the probability)? Can we do that?

The short answer is: it's really close to $n!/e$. But how close is it?

$$\begin{aligned}
\text{Error} = \frac{n!}{e} - (\star) &= \frac{n!}{e} - \left(\frac{n!}{0!} - \frac{n!}{1!} + \frac{n!}{2!} - \frac{n!}{3!} + \cdots \pm \frac{n!}{n!} \right) \\
&= \left(\frac{n!}{0!} - \frac{n!}{1!} + \cdots \right) - \left(\frac{n!}{0!} - \frac{n!}{1!} + \frac{n!}{2!} - \frac{n!}{3!} + \cdots \pm \frac{n!}{n!} \right) \\
&= \mp \frac{n!}{(n+1)!} \pm \frac{n!}{(n+2)!} \mp \cdots
\end{aligned}$$

This is really tiny. In particular, we can do some algebra to get:

$$\begin{aligned}
\text{Error} &\leq \frac{1}{n+1} + \frac{1}{(n+1)(n+2)} + \frac{1}{(n+1)(n+2)(n+3)} + \cdots \quad \text{(by the } \triangle \text{ ineq.)} \\
&< \frac{1}{(n+1)} + \frac{1}{(n+1)^2} + \frac{1}{(n+1)^3} + \cdots
\end{aligned}$$

Proposition 2.28 (Geometric Sum Formula).
The sum of an infinite geometric series $(a, ar, ar^2, ar^3, \cdots)$ is given by

$$\Sigma = \frac{\text{first term}}{1 - \text{common ratio}} = \frac{a}{1-r}$$

We can use this fact to finish off our bound. We have a geometric sequence with initial term $\frac{1}{n+1}$ and a common ratio of $\frac{1}{n+1}$, so

$$\begin{aligned}\text{Error} &< \frac{1}{(n+1)} + \frac{1}{(n+1)^2} + \frac{1}{(n+1)^3} + \cdots \\ &= \frac{\frac{1}{n+1}}{1 - \frac{1}{n+1}} = \frac{1}{(n+1)-1} = \frac{1}{n}\end{aligned}$$

So now we know that the number of derangements is (strictly) within $\pm\frac{1}{n}$ of $n!/e$. For $n \geq 2$, there is only one integer within $\pm\frac{1}{n}$ of $n!/e$, since $\frac{1}{n}$ gets too small, so we can just use

$$D_n = \text{number of derangments on } n \text{ people} = \text{round}\left(\frac{n!}{e}\right)$$

It turns out that this formula holds for $n = 1$ as well (check it!), so we get the following general formula:

Proposition 2.29 (nth Derangement Number).
The nth derangement number is given by $D_n = \text{round}\left(\frac{n!}{e}\right)$.

Note 2.30 (A Consequence).
As a result of this, we also get the cool fact that $\frac{n!}{e}$ is always really close to an integer. Neat!

CHAPTER 3

Binomial Coefficients

Not all treasure's silver and gold, mate.

—Jack Sparrow

Contents

3.1 Pirates and Gold

We start off with an important counting technique in combinatorics:

Theorem 3.1 (The Pirates and Gold Theorem).
The number of ways to split n gold coins among k pirates is given by

$$\binom{n+k-1}{k-1} = \binom{n+k-1}{n}$$

Proof. We distribute coins in the following manner:

1. Draw $n+k-1$ empty slots.
2. Choose $k-1$ of the slots to be dividers. This partitions the lines into k regions, and leaves n slots empty (1 for each coin).
3. Pirate 1 gets the number of empty lines in region 1, pirate 2 the number in region 2, pirate 3 gets the number in region 3, etc.

This enumerates every possible distribution of coins. Thus, there are $\binom{n+k-1}{k-1}$ ways to distribute the gold. □

Note 3.2 (Stars and Bars). The above theorem is also sometimes called the stars and bars theorem since the arrangements of coins and dividers look like:

$$\star\star\star\star||\star|\star\star| \qquad \text{or} \qquad \star|\star|\star\star|\star|\star\star$$

This also illustrates that it is ok for more than one divider to be in between two stars, and for dividers to be at the ends of the stars.

Example 3.3 (No Pirate Left Out).
What is the number of ways to split n gold coins among k pirates such that each pirate is guaranteed to get at least 1 coin?

One way to do this is to just give 1 coin to each pirate, then to distribute the remaining $n-k$ coins among the pirates with the pirates and gold theorem. There are:

$$\binom{(n-k)+k-1}{k-1} = \binom{n-1}{k-1}$$

ways to do this. In general, this strategy can be used to ensure the pirates get various numbers of guaranteed coins. ☺

Example 3.4 (Multiple Kinds of Coins).
What is the number of ways for 5 pirates to split 20 gold and 40 silver coins?

We can first distribute the gold coins, then the silver. From the pirates and gold theorem, there are

$$\binom{5+20-1}{5-1} = \binom{24}{4}$$

ways to distribute the gold, and

$$\binom{5+40-1}{5-1} = \binom{44}{4}$$

ways to distribute the silver. Since the way we distribute one doesn't affect the other, we multiply the two to get $\binom{24}{4} \times \binom{44}{4}$ total ways. ☺

Example 3.5 (Greedy Pirates).
What is the number of ways for 6 pirates to split 20 gold coins such that exactly half of the pirates receive nothing, and every pirate in the other half receives something?

We first choose the 3 pirates who receive nothing; there are $\binom{6}{3}$ ways to do this. Then, we distribute the 20 gold coins amongst the remaining 3 pirates. From example (3.3), there are

$$\binom{n-1}{k-1} = \binom{20-1}{3-1} = \binom{19}{2}$$

ways to do this such that each gets at least 1. We can't allow any of these three pirates to get zero or else there would be more than 3 pirates who get none. Thus, in total there are $\binom{6}{3} \times \binom{19}{2}$ ways to distribute the coins. ☺

Example 3.6 (A Generous Captain).
What is the number of ways for 5 pirates to split 20000 gold coins if their captain refuses to accept a split which gives him more than 4000 gold coins?

The number of ways for the captain to receive less than or equal to 4000 coins is

$$\#\text{ total distributions} - \#\text{ distributions where the captain gets more than } 4000$$

The number of total distributions is given by the pirates and gold theorem, and is

$$\binom{20000+5-1}{5-1} = \binom{20004}{4}$$

and the number where the captain gets more than 4000 can be found by giving the captain 4001 gold coins to start, then distributing the rest. There are

$$\binom{(20000-4001)+5-1}{5-1} = \binom{16003}{4}$$

so subtracting gives that there are $\binom{20004}{4} - \binom{16003}{4}$ total ways. ☺

3.2 Lucas's Theorem

Example 3.7 (Last Digit Problem).
What is the last digit of the binomial coefficient $\binom{250}{125}$?

This is easy. Just bust out a calculator, and we find that

$$\binom{250}{125} = 91208366928185711600087718663295946582 84 \ddots$$

$$\ddots\, 7985411225264672245111235434562752$$

and so the answer is 2. We're done, right? Well, what if we had to do it without a calculator (or for *really* big numbers[1]). Could we still do it?

It turns out there is a "clean" way to solve this problem. To understand it, let's try an easier example: finding $\binom{17}{3} \pmod 3$ (or the last digit of $\binom{17}{3}$ in base-3).

Here's the algorithm (important caveat, it only works for prime powered number bases, we'll come back to base-10 later):

1. Convert 17 and 7 to base-3:

$$17_{10} = 122_3 \qquad \text{and} \qquad 7_{10} = 21_3$$

2. Write the numbers out in a table, padding the smaller one with zeroes so they are the same length:

$$\begin{matrix} 1 & 2 & 2 \\ 0 & 2 & 1 \end{matrix}$$

3. Put binomial coefficient parentheses around them, and multiply:

$$\binom{1}{0} \times \binom{2}{2} \times \binom{2}{1} \equiv 2 \pmod 3$$

That's it! The last digit of $\binom{17}{7}$ in base-3 is 2.

Theorem 3.8 (Lucas's Theorem).
For any prime p, to calculate $\binom{n}{k} \pmod p$:

- Write n in base-p as: $(n_1 n_2 \cdots n_t)_p$
- Write k in base-p as: $(k_1 k_2 \cdots k_t)_p$

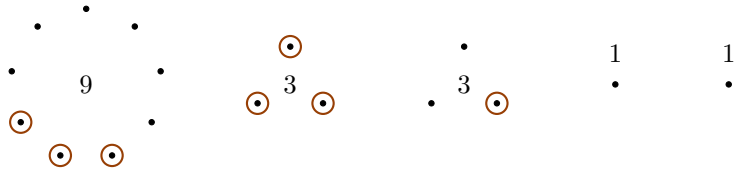
Importantly, we require n and k to have the same number of digits: t. Then

$$\binom{n}{k} \equiv \binom{n_1}{k_1}\binom{n_2}{k_2}\cdots\binom{n_t}{k_t} \pmod p$$

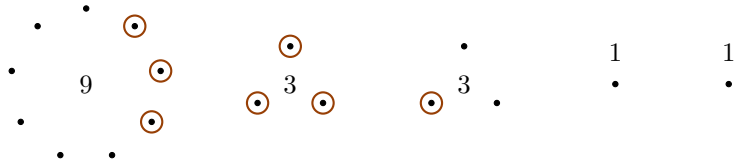
We will give an example below of how the proof proceeds for $\binom{17}{7}$, which can be generalized to an arbitrary binomial coefficient.

[1] Adding just 2 zeroes to the end of each of the numbers in the example gives us a 7523-digit number...

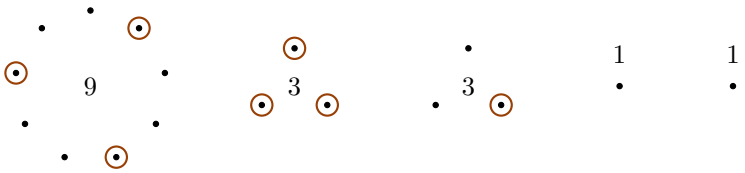
Proof for $\binom{17}{7}$. Remember that $\binom{17}{7}$ counts the number of 7 element subsets of a set of size 17. We can think of them as pictures which look like:



or

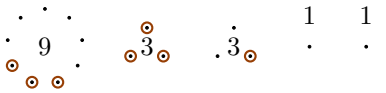


where the circled elements represent the dots we choose to include in our set of size 7. However, the two arrangements above seem to be "similar" in a way. We'll define similar in a second, but see if you can figure it out with the added information that the picture below is *not* similar to either of the above (and the two above are similar to each other):

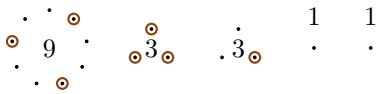


Can you guess? We'll define *similar* to mean that we can rotate the wheels of one picture to get to the second one. Our strategy will be to partition the sets into equivalence classes based on this notion of similarity, then count the size of each class.

For example, the sizes of the equivalence classes for the above drawings are:



$9 \times 1 \times 3 \times 1 \times 1$
$\Rightarrow$ 27 ways to spin



$3 \times 1 \times 3 \times 1 \times 1$
$\Rightarrow 9$ ways to spin

But notice that each of these is $\equiv 0 \pmod 3$, so they will not contribute to our count. Can we find one which is not zero?

We have to do all or nothing. The second we only choose part of a circle, it can spin and that gives us a multiple of three. This is because the number of rotational symmetries of a wheel is always a factor of the wheel size, and since our wheels are all powers of 3, the only possible factors are all multiples of 3.[2]

Therefore, we are interested in the number of ways to make (by adding) 7 from full wheels, that is, from the numbers 9, 3, 3, 1, and 1.[3]

[2] This is why we require p to be prime.

[3] Since we are working in base-3, we have at most 2 of each of these numbers. The setup wouldn't be useful if we just drew 17 single dots. In base p, we would have at most $p-1$ of each power of p.

The base-3 representation of 7 tells us exactly how to do this! Since

$$7_{10} = 021_3$$

we know we need zero 9's, two 3's, and one 1, and that this is the unique way to get to the number 7. How many ways can we do this? Consider the table:

	9's	3's	1's
Number we have	1	2	2
Number we need	0	2	1

so the number of ways to do this is just $\binom{1}{0} \times \binom{2}{2} \times \binom{2}{1}$, exactly what we wanted! □

Note 3.9 (The Original Problem).
So how do we find the last digit of $\binom{250}{125}$?

From Lucas's Theorem, we know that in base-5

$$\binom{250}{125} \equiv \binom{2}{1} \times \binom{0}{0} \times \binom{0}{0} \times \binom{0}{0} \equiv 2 \pmod 5$$

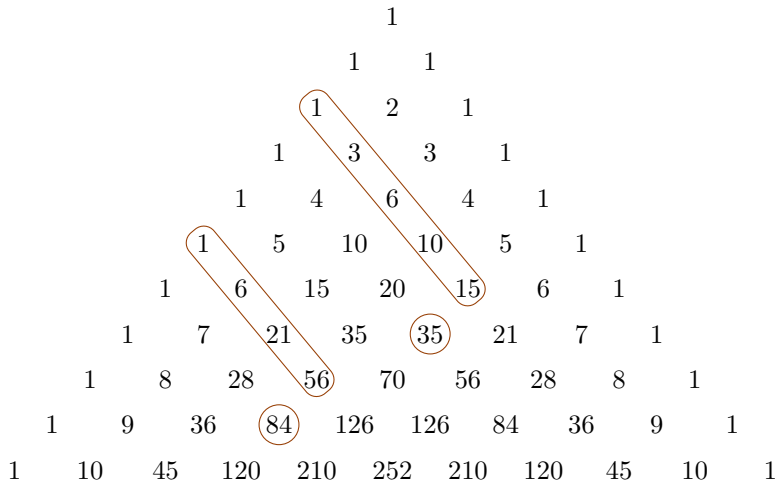
so the last digit is either a 2 or a 7. Then, doing the same in base-2 gives

$$\binom{250}{125} \equiv \binom{?}{?} \times \cdots \times \binom{?}{?} \times \binom{0}{1} \equiv 0 \pmod 2$$

(here, the last digits are all we need since we get a zero), so the binomial coefficient ends in an even number. Thus, it ends in a 2.

3.3 Pascal's Triangle and Related Identities

Consider the first 10 rows of Pascal's triangle. If you look at the circled parts below, you might notice an interesting pattern:



The sum of each long circled diagonal is equal to the single circled number to its southwest! Let's phrase this as a combinatorial identity.

Theorem 3.10 (Hockey Stick Identity).
For integers n and k, we have

$$\binom{n}{0} + \binom{n+1}{1} + \binom{n+2}{2} + \cdots + \binom{n+k}{k} = \binom{n+k+1}{k}$$

This is often written in summation notation as

$$\sum_{j=0}^{k} \binom{n+j}{j} = \binom{n+k+1}{k}$$

Proof. Using the fact that $\binom{n}{0} = \binom{n+1}{0}$, we can write

$$\begin{aligned}
&\binom{n}{0} + \binom{n+1}{1} + \binom{n+2}{2} + \cdots + \binom{n+k}{k} \\
&= \underbrace{\binom{n+1}{0} + \binom{n+1}{1}}_{\binom{n+2}{2} \text{ by Pascal's Identity}} + \binom{n+2}{2} + \cdots + \binom{n+k}{k} \\
&= \underbrace{\binom{n+2}{1} + \binom{n+2}{2}}_{\binom{n+3}{2} \text{ by Pascal's Identity}} + \cdots + \binom{n+k}{k} \\
&\qquad \vdots \\
&= \binom{n+k}{k-1} + \binom{n+k}{k} = \binom{n+k+1}{k}
\end{aligned}$$

which is what we wanted. □

Since we used Pascal's Identity in the above proof, let's go ahead and prove it as well:

Theorem 3.11 (Pascal's Identity).
For integers n and k, we have

$$\binom{n}{k} + \binom{n}{k+1} = \binom{n+1}{k+1}$$

Proof. We'll use counting in two ways:

- (RHS) This is the number of ways to pick $k+1$ objects from the set $\{1, \cdots, n+1\}$.
- (LHS) We count the same set by splitting into two cases:
  - CASE 1 If our objects include $n+1$, then we have to pick k more items from the set $\{1, 2, \cdots, n\}$. We can do this in $\binom{n}{k}$ ways.
  - CASE 2 If our objects don't include $n+1$, then we have to pick $k+1$ more from the set $\{1, 2, \cdots, n\}$. We can do this in $\binom{n}{k+1}$ ways.

  Since our cases are disjoint, we have a total of $\binom{n}{k} + \binom{n}{k+1}$ ways.

However, both sides count the same set, so the two expressions are equal. □

Theorem 3.12 (Vandermonde's Convolution).
For any non-negative integers m, n, and k, we have

$$\sum_{j=0}^{k} \binom{m}{j}\binom{n}{k-j} = \binom{m+n}{k}$$

To understand this, it helps to see an example:

$$\underbrace{\binom{4}{0}\binom{3}{2}}_{\substack{\text{Pick 0 from } \{1,2,3,4\} \\ \text{and 2 from } \{5,6,7\}}} + \underbrace{\binom{4}{1}\binom{3}{1}}_{\substack{\text{Pick 1 from } \{1,2,3,4\} \\ \text{and 1 from } \{5,6,7\}}} + \underbrace{\binom{4}{2}\binom{3}{0}}_{\substack{\text{Pick 2 from } \{1,2,3,4\} \\ \text{and 0 from } \{5,6,7\}}} = \binom{7}{2}$$

and so we see that all Vandermonde's convolution is doing is splitting a set up into two parts, and casing on how many elements we take from the first part.

Proof. Again, we count in two ways.

- (RHS) This is the number of ways to select a committee of size k from a group of m men and n women.
- (LHS) We case on all the possible values of m: 0 to k. Given a j in that range, there are $\binom{m}{j}\binom{n}{k-j}$ ways to pick a group with j men, so in total, there are

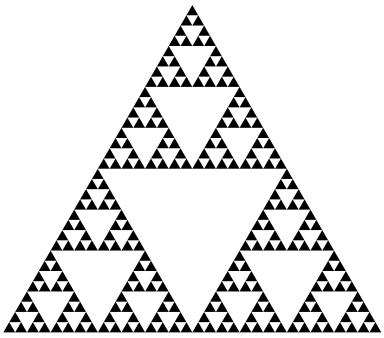
  $$\sum_{j=0}^{k} \binom{m}{j}\binom{n}{k-j}$$

  ways to build the committee.

Since both sides count the same set, the two expressions are equal. □

Note 3.13 (The Sierpinski Triangle).
We'll end this chapter with a cool fact about Pascal's triangle:

If we take Pascal's triangle, and erase all the even entries in it, we end up with a pattern which looks like:



This is called a Sierpinski Triangle. If you want to try and prove this is what you get, you can do so by induction.

(Hint: just write 1/0 for odd/even instead of the numbers. Then, apply Pascal's Identity!)

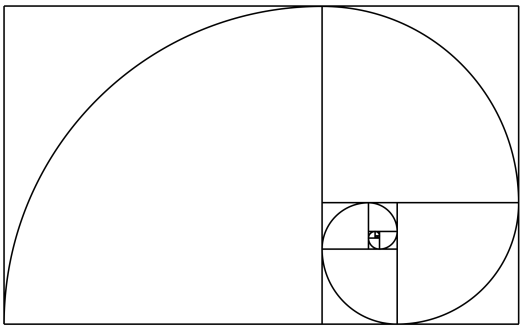
Part

II

Recurrence Relations

CHAPTER 4

Linear Algebra and Recurrences



The Fibonacci Sequence turns out to be the key to understanding how nature designs... and is... a part of the same ubiquitous music of the spheres that builds harmony into atoms, molecules, crystals, shells, suns and galaxies and makes the Universe sing.

—Guy Murchie

Contents

4.0 Diagonalization Primer

Throughout this chapter, we are going to be interested in taking the nth power of a matrix. Fortunately for us, there are techniques from linear algebra that allow us to do this quickly. This is a process called *diagonalization*.

Example 4.1 (Diagonalizing a Matrix).
Write the matrix

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

as PDP^{-1} for some diagonal matrix D and some invertible matrix P.

We want to find vectors $\mathbf{v}$ and corresponding scalars λ such that

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \mathbf{v} = \lambda \mathbf{v}$$

such that $\mathbf{v} \neq \mathbf{0}$ (why is this a useful condition?). We'll call $\mathbf{v}$ an eigenvector with corresponding eigenvalue λ. How might we find these? Note that our above condition is equivalent to

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \mathbf{v} = \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \mathbf{v} \iff \underbrace{\left(\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \right)}_{\text{need this to be singular}} = \mathbf{0}$$

That matrix is singular when $\det(A - \lambda I) = 0$. So we just have to solve that equation:

$$\begin{aligned} \det(A - \lambda I) = 0 &\iff (1-\lambda)(-\lambda) - 1(1) = 0 \\ &\iff \lambda^2 - \lambda - 1 = 0 \\ &\iff \lambda = \frac{1 \pm \sqrt{5}}{2} \end{aligned}$$

so we've found the eigenvalues.

Proposition 4.2 (Distinct Eigenvalues Implies Diagonalizable).
If the eigenvalues of A are all distinct, their corresponding eigenvectors are linearly independent and therefore A is diagonalizable.

Now, using the proposition (4.2), we know that there are linearly independent vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ which correspond to those eigenvalues. That is, there are vectors such that

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \mathbf{v}_1 = \underbrace{\frac{1+\sqrt{5}}{2}}_{\lambda_1} \mathbf{v}_1 \qquad \text{and} \qquad \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \mathbf{v}_2 = \underbrace{\frac{1-\sqrt{5}}{2}}_{\lambda_2} \mathbf{v}_2$$

Now, consider the effect of right multiplying by $\begin{bmatrix}\mathbf{v}_1 & \mathbf{v}_2\end{bmatrix}$. Let $\mathbf{v}_1 = \left[\begin{smallmatrix}a\\b\end{smallmatrix}\right]$ and $\mathbf{v}_2 = \left[\begin{smallmatrix}c\\d\end{smallmatrix}\right]$:

$$\begin{bmatrix}1 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}a & c\\b & d\end{bmatrix} = \left[\begin{array}{c:c} \begin{bmatrix}1 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}a\\b\end{bmatrix} & \begin{bmatrix}1 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}c\\d\end{bmatrix}\end{array}\right] = \left[\begin{array}{c:c} \frac{1+\sqrt{5}}{2}\begin{bmatrix}a\\b\end{bmatrix} & \frac{1-\sqrt{5}}{2}\begin{bmatrix}c\\d\end{bmatrix}\end{array}\right]$$
$$= \left[\begin{array}{c:c} \lambda_1 a & \lambda_2 c\\ \lambda_1 b & \lambda_2 d\end{array}\right] = \begin{bmatrix}a & c\\b & d\end{bmatrix}\begin{bmatrix}\lambda_1 & 0\\0 & \lambda_2\end{bmatrix}$$

But now, since $\mathbf{v}_1$ and $\mathbf{v}_2$ are linearly independent, we know that $\left[\begin{smallmatrix}a & c\\b & d\end{smallmatrix}\right]$ has an inverse, so we can write:

$$\begin{bmatrix}1 & 1\\1 & 0\end{bmatrix} = \underbrace{\begin{bmatrix}a & c\\b & d\end{bmatrix}}_{P}\underbrace{\begin{bmatrix}\lambda_1 & 0\\0 & \lambda_2\end{bmatrix}}_{D}\underbrace{\begin{bmatrix}a & c\\b & d\end{bmatrix}^{-1}}_{P^{-1}}$$

Now, when we take the nth power, we get

$$\begin{bmatrix}1 & 1\\1 & 0\end{bmatrix}^n = \underbrace{(PDP^{-1})(PDP^{-1})(PDP^{-1})\cdots(PDP^{-1})}_{n \text{ times}}$$
$$= PD^nP^{-1}$$

This is useful since taking the power of a diagonal matrix is easy: $\left[\begin{smallmatrix}\lambda_1 & 0\\0 & \lambda_2\end{smallmatrix}\right]^n = \left[\begin{smallmatrix}\lambda_1^n & 0\\0 & \lambda_2^n\end{smallmatrix}\right]$, so in total, we get

$$\begin{bmatrix}1 & 1\\1 & 0\end{bmatrix}^n = P\begin{bmatrix}\lambda_1^n & 0\\0 & \lambda_2^n\end{bmatrix}P^{-1}$$

Note 4.3 (What is P?).
If you wanted, you could calculate P and P^{-1} by finding the eigenvectors $\mathbf{v}_1$ and $\mathbf{v}_2$, but we won't need to for our purposes.

4.1 The Fibonacci Sequence

Example 4.4 (A Tiling Problem).
How many ways are there to tile a $1 \times n$ region:

image

using only 1×1 squares ($\square$) and 1×2 dominoes ($\square\!\square$)?

Let's try and work through some small cases:

$n = 1$: 1 way : $\square$
$n = 2$: 2 ways : $\square\square$ or $\square\!\square$
$n = 3$: 3 ways : $\square\square\!\square$ or $\square\!\square\square$ or $\square\square\square$

So it seems like we have our pattern! $1, 2, 3, 4, \cdots$. Unfortunately, this is wrong. Consider the case of $n = 4$. There are 5 ways:

$\square\square\square\square$ or $\square\square\square\!\square$ or $\square\!\square\square\square$
or $\square\!\square\!\square\!\square$ or $\square\square\!\square\square$

So in general, we need a more careful approach to count these tilings. Let's split it into cases based on the last tile in the sequence:

$$\text{number of tilings} = \text{number ending in a square} + \text{number ending in a domino}$$

$$= \boxed{n-2}\boxed{} + \boxed{n-1}\boxed{}$$

and so we get the relation $a_n = a_{n-1} + a_{n-2}$. The Fibonacci sequence! ☺

Definition 4.5 (The Fibonacci Sequence).
The Fibonacci sequence is the sequence of numbers:

$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, \cdots$$

defined by the rules

$$F_0 = 0, \quad F_1 = 1, \quad \text{and} \quad F_n = F_{n-1} + F_{n-2}$$

Almost every identity about Fibonacci numbers can be proven by appealing to a tiling argument based on the above:

Example 4.6 (A Fibonacci Identity).
Prove that $F_n^2 + F_{n-1}^2 = F_{2n-1}$.

Let's prove this by counting in two ways:

- (RHS) This is how many ways there are to tile a $1 \times 2n-2$ region using only dominoes and squares.
- (LHS) We split into two cases:

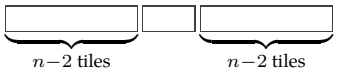
  CASE 1: The middle of the tile region falls along a split.

  Since the two halves are independent of each other, the number of ways to tile the whole region is

  $$(\#\text{ ways to tile } n-1)^2 = F_n^2$$

  CASE 2: The middle of the tile region falls on a tile.

  Since the two halves are independent of each other, the number of ways to tile the whole region is

  $$(\#\text{ ways to tile } n-2)^2 = F_{n-1}^2$$

  Thus, in total, there are $F_n^2 + F_{n-1}^2$ ways to tile the region.

Since both sides count the same thing, the two expressions are equal. ☺

Note 4.7 (Why is the Fibonacci sequence in nature so much?).
DNA needs to encode structures about how to grow something, and a natural way to do this is based solely on a few bits of information about previous states.

Some possible ways to describe growth are:

$$\begin{aligned} a_n &= a_{n-1} && \rightsquigarrow && \text{no growth, something like a rock} \\ a_n &= 2a_{n-1} && \rightsquigarrow && \text{rapid growth, such as bacteria} \\ a_n &= a_{n-1} + a_{n-2} && \rightsquigarrow && \text{next obvious choice for a growth pattern} \end{aligned}$$

and so natural selection settled on the Fibonacci sequence. QED?

Example 4.8 (Explicit Formula for Fibonacci).
Find a closed form formula for the nth Fibonacci number.

Notice that we can express the Fibonacci sequence in terms of a matrix equation as:

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} F_n \\ F_{n-1} \end{bmatrix} = \begin{bmatrix} F_{n+1} \\ F_n \end{bmatrix} \implies \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} F_{n+1} \\ F_n \end{bmatrix}$$

So this problem reduces down to finding the nth power of the matrix $\left[\begin{smallmatrix} 1 & 1 \\ 0 & 1 \end{smallmatrix}\right]$. We did this in the previous section! Let $\divideontimes$ denote some constant which doesn't depend on n. We have that

$$\begin{aligned} \begin{bmatrix} F_{n+1} \\ F_n \end{bmatrix} &= \begin{bmatrix} \divideontimes & \divideontimes \\ \divideontimes & \divideontimes \end{bmatrix} \begin{bmatrix} \lambda_1^n & 0 \\ 0 & \lambda_2^n \end{bmatrix} \begin{bmatrix} \divideontimes & \divideontimes \\ \divideontimes & \divideontimes \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \divideontimes & \divideontimes \\ \divideontimes & \divideontimes \end{bmatrix} \begin{bmatrix} \lambda_1^n & 0 \\ 0 & \lambda_2^n \end{bmatrix} \begin{bmatrix} \divideontimes \\ \divideontimes \end{bmatrix} \\ &= \begin{bmatrix} \divideontimes & \divideontimes \\ \divideontimes & \divideontimes \end{bmatrix} \begin{bmatrix} \lambda_1^n \cdot \divideontimes \\ \lambda_2^n \cdot \divideontimes \end{bmatrix} = \begin{bmatrix} \lambda_1^n \cdot \divideontimes + \lambda_2^n \cdot \divideontimes \\ \lambda_1^n \cdot \divideontimes + \lambda_2^n \cdot \divideontimes \end{bmatrix} \end{aligned}$$

and so equating the bottom rows, we get that there are constants (not depending on n) α and β such that

$$F_n = \alpha \left(\frac{1+\sqrt{5}}{2} \right)^n + \beta \left(\frac{1-\sqrt{5}}{2} \right)^n$$

Let's solve for the constants:

$$\begin{aligned} (n=0): \qquad & 0 = \alpha \left(\frac{1+\sqrt{5}}{2} \right)^0 + \beta \left(\frac{1-\sqrt{5}}{2} \right)^0 = \alpha + \beta \\ (n=1): \qquad & 1 = \alpha \left(\frac{1+\sqrt{5}}{2} \right)^1 + \beta \left(\frac{1-\sqrt{5}}{2} \right)^1 \end{aligned}$$

From here, we can just solve the system to get $\alpha = \frac{1}{\sqrt{5}}$ and $\beta = -\frac{1}{\sqrt{5}}$, so

$$F_n = \frac{1}{\sqrt{5}} \left(\frac{1+\sqrt{5}}{2} \right)^n - \frac{1}{\sqrt{5}} \left(\frac{1-\sqrt{5}}{2} \right)^n$$

☺

4.2 Solving Recurrences (Distinct Roots)

Example 4.9 (The Simplest Recurrence).
What is a general solution to the recurrence defined by the rules

$$\begin{cases} a_n = Aa_{n-1} \\ a_0 = a_0 \end{cases}$$

It's not hard to see that the recurrence is solved by $a_n = a_0 \cdot A^n$. You can prove this by induction if you'd like.

Example 4.10 (Generalizing Fibonacci).
What is a general solution to the recurrence defined by the rules

$$\begin{cases} a_n = Aa_{n-1} + Ba_{n-2} \\ a_0 = a_0 \\ a_1 = a_1 \end{cases}$$

The solution mirrors the solution we came up with to the Fibonacci sequence. We start by writing the recurrence as a matrix equation:

$$\begin{bmatrix} A & B \\ 1 & 0 \end{bmatrix} \begin{bmatrix} a_{n-1} \\ a_{n-2} \end{bmatrix} = \begin{bmatrix} a_n \\ a_{n-1} \end{bmatrix} \implies \begin{bmatrix} A & B \\ 1 & 0 \end{bmatrix}^n \begin{bmatrix} a_1 \\ a_0 \end{bmatrix} = \begin{bmatrix} a_{n+1} \\ a_n \end{bmatrix}$$

Assuming that our matrix $\left[\begin{smallmatrix} A & B \\ 1 & 0 \end{smallmatrix}\right]$ is diagonalizable, we can write

$$\begin{bmatrix} A & B \\ 1 & 0 \end{bmatrix}^n \begin{bmatrix} a_1 \\ a_0 \end{bmatrix} = \begin{bmatrix} ※ & ※ \\ ※ & ※ \end{bmatrix} \begin{bmatrix} \lambda_1^n & 0 \\ 0 & \lambda_2^n \end{bmatrix} \begin{bmatrix} ※ & ※ \\ ※ & ※ \end{bmatrix} \begin{bmatrix} ※ \\ ※ \end{bmatrix}$$

where ※ denotes some number independent of n, but which can be based on A, B, a_0, and a_1. Then

$$= \begin{bmatrix} ※ & ※ \\ ※ & ※ \end{bmatrix} \begin{bmatrix} \lambda_1^n & 0 \\ 0 & \lambda_2^n \end{bmatrix} \begin{bmatrix} ※ \\ ※ \end{bmatrix} = \begin{bmatrix} ※ & ※ \\ ※ & ※ \end{bmatrix} \begin{bmatrix} \lambda_1^n \cdot ※ \\ \lambda_2^n \cdot ※ \end{bmatrix}$$

$$= \begin{bmatrix} \lambda_1^n \cdot ※ + \lambda_2^n \cdot ※ \\ \lambda_1^n \cdot ※ + \lambda_2^n \cdot ※ \end{bmatrix} \overset{\text{BTW}}{=} \begin{bmatrix} a_{n+1} \\ a_n \end{bmatrix}$$

and so we can write $a_n = \alpha\lambda_1^n + \beta\lambda_2^n$, where α and β are functions of A, B, a_0, and a_1, and λ_1, λ_2 are the eigenvalues of $\left[\begin{smallmatrix} A & B \\ 1 & 0 \end{smallmatrix}\right]$.

Corollary 4.11 (Independence of Initial Values).
If you keep the same A and B, but change the values of a_0 and a_1, then the solution to the recurrence is still of the form

$$a_n = \alpha \cdot \lambda_1^n + \beta \cdot \lambda_2^n$$

Definition 4.12 (Characteristic Polynomial of a Recursion).
Let A be a matrix which represents a recursion (as above). Then, the characteristic polynomial of the recursion is

$$\det(A - \lambda I) = 0$$

In our example, we get the characteristic polynomial

$$\det\left(\begin{bmatrix} A & B \\ 1 & 0 \end{bmatrix} - \lambda I\right) = 0 \implies \det\left(\begin{bmatrix} A-\lambda & B \\ 1 & -\lambda \end{bmatrix}\right) = 0 \implies -\lambda(A-\lambda) - B = 0$$
$$\implies \lambda^2 - A\lambda - B = 0$$

and so we can find λ_1 and λ_2 by finding the roots of this polynomial.

How do we get α and β? Plug in some convenient values of n (such as 0 and 1):

$$\begin{aligned} n = 0: &\quad a_0 = \alpha + \beta \\ n = 1: &\quad a_1 = \alpha\lambda_1 + \beta\lambda_2 \end{aligned}$$

and then we can solve the resulting system to find α and β.

Note 4.13 (What issues can we run into?).
What if the system we get when solving for α and β gives us two of the same equations?

This happens when $\lambda_1 = \lambda_2$. In this case, our matrices are typically not even diagonalizable to begin with, so we'll have to deal with it later.

4.3 Solving Recurrences with More Variables

Can we extend these ideas to 3 or more variables? Sure!

Example 4.14 (Solving 3-Term Linear Recurrences).
What is the general solution to the recurrence $a_n = Aa_{n-1} + Ba_{n-2} + Ca_{n-3}$?

Start from the matrix equation:

$$\begin{bmatrix} A & B & C \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} a_n \\ a_{n-1} \\ a_{n-2} \end{bmatrix} = \begin{bmatrix} a_{n+1} \\ a_n \\ a_{n-1} \end{bmatrix} \implies \begin{bmatrix} A & B & C \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}^n \begin{bmatrix} a_2 \\ a_1 \\ a_0 \end{bmatrix} = \begin{bmatrix} \\ \\ a_n \end{bmatrix}$$

and then find eigenvalues to attempt to diagonalize:

$$\det\left(\begin{bmatrix} A-\lambda & B & C \\ 1 & -\lambda & 0 \\ 0 & 1 & -\lambda \end{bmatrix}\right) = 0 \implies -\begin{vmatrix} A-\lambda & C \\ 1 & 0 \end{vmatrix} - \lambda\begin{vmatrix} A-\lambda & C \\ 1 & -\lambda \end{vmatrix} = 0$$
$$\implies C - \lambda(\lambda^2 - A\lambda - B) = 0$$
$$\implies -\lambda^3 + A\lambda^2 + B\lambda + C = 0$$

Theorem 4.15 (Characteristic Polynomial of a Linear Recurrence).
The characteristic polynomial of the recurrence

$$a_n = C_1 a_{n-1} + C_2 a_{n-2} + \cdots + C_k a_{n-k}$$

is $\lambda^k - C_1\lambda^{k-1} - C_2\lambda^{k-2} - \cdots - C_{k-1}\lambda^1 - C_k$.

Proof. We won't prove the general case (you can do so with induction if you'd like), but will illustrate the approach for the $k = 3 \implies k = 4$ case.

For 4 terms, the matrix which represents the recurrence is

$$A = \begin{bmatrix} A & B & C & D \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

the characteristic polynomial is simply the equation obtained by taking the determinant of $A - \lambda I$ and setting it equal to zero. Let's do this through cofactor expansion on the last column:

$$\begin{aligned} \det\left(\begin{bmatrix} A-\lambda & B & C & D \\ 1 & -\lambda & 0 & 0 \\ 0 & 1 & -\lambda & 0 \\ 0 & 0 & 1 & -\lambda \end{bmatrix}\right) &= -D \underbrace{\begin{vmatrix} 1 & -\lambda & 0 \\ 0 & 1 & -\lambda \\ 0 & 0 & 1 \end{vmatrix}}_{\substack{=1 \text{ since upper} \\ \text{triangular}}} + (-\lambda)\left|\begin{array}{c}\text{previous } (n=3) \\ \text{determinant}\end{array}\right| \\ &= -D - \lambda\left(-\lambda^3 + A\lambda^2 + B\lambda + C\right) \\ &= \lambda^4 - A\lambda^3 - B\lambda^2 - C\lambda - D \end{aligned}$$

which is exactly what we wanted. □

Example 4.16 (Some concrete characteristic polynomials).
The general "algebra-ese" formula can be kind of hard to interpret, so here are some examples of recurrences and their characteristic polynomials:

$$\begin{aligned} a_n &= Aa_{n-1} + Ba_{n-2} + Ca_{n-3} + Da_{n-4} + Ea_{n-5} \\ &\Rightarrow \lambda^5 - A\lambda^4 - B\lambda^3 - C\lambda^2 - D\lambda - E = 0 \end{aligned}$$

$$\begin{aligned} a_n &= Aa_{n-2} + Ba_{n-4} \\ &\Rightarrow \lambda^4 - A\lambda^2 - B = 0 \end{aligned}$$

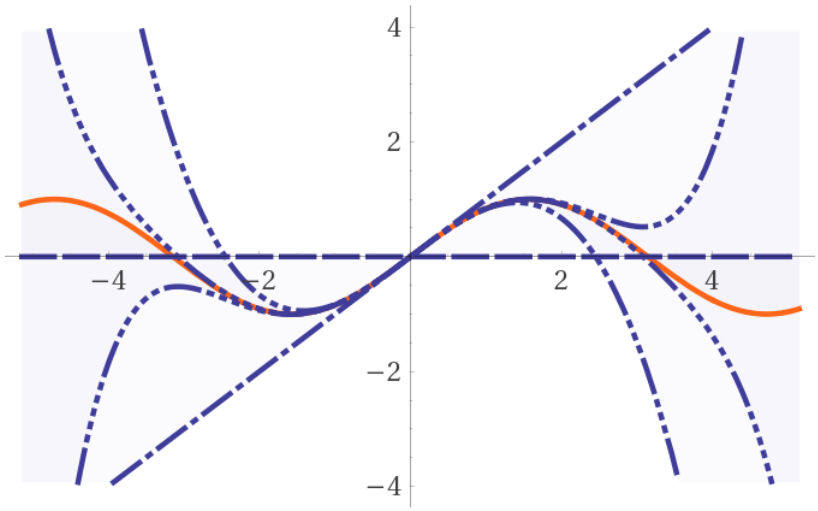
Ok, back to the $n = 3$ example. From here, we can solve the recurrence in three steps:

1. Solve the polynomial $\lambda^3 - A\lambda^2 - B\lambda - C = 0$. The roots are the eigenvalues.
2. If the roots are distinct, use the PDP^{-1} form to solve the matrix equation for a_n. You'll always get $a_n = \alpha\lambda_1^n + \beta\lambda_2^n + \gamma\lambda_3^n$, where α, β, and γ are unknown. (It's a good exercise to check this for yourself!)
3. Use three values of n (typically 0, 1, and 2) to solve for α, β, and γ.

That's it! ☺

CHAPTER 5

Generating Functions



A generating function is a clothesline on which we hang up a sequence of numbers for display.

—Herbert Wilf

Contents

5.1 Intro to Generating Functions

I claim that $1/89$ is a magic number. Let's look at it's base-10 expansion:

$$\frac{1}{89} = 0.011235...$$

It's the Fibonacci sequence! This is amazing... well, until you look at the next few digits in the decimal:

$$\frac{1}{89} = 0.0112359550...$$

What happened? Actually, nothing happened. $1/89$ is still a magical number. We just have to read it a little bit more carefully:

$$\begin{array}{cccccccccccccc} & 0 & . & 0 & 1 & 1 & 2 & 3 & 5 & 9 & 5 & 5 & 0 & \cdots \\ \hline + & & & & 1 & 1 & 2 & 3 & 5 & 8 & & & & \\ & & & & & & & & & 1 & 3 & & & \\ & & & & & & & & & & 2 & 1 & & \\ & & & & & & & & & & & 3 & 4 & \\ & & & & & & & & & & & & 5 & 5 \\ & & & & & & & & & & & & & 8 \quad 9 \end{array}$$

So this actually works (if you're willing to carry)! Let's prove it:

Example 5.1 (The Fibonacci Fraction).
Prove that

$$\sum_{i=0}^{\infty} \frac{F_i}{10^{i+1}} = \frac{1}{89}$$

We can do some clever algebra. Let $\text{RHS} = \sum_{i=0}^{\infty} \frac{F_i}{10^{i+1}}$

$$\begin{aligned} \text{RHS} &= \frac{F_0}{10^1} + \frac{F_1}{10^2} + \frac{F_2}{10^3} + \frac{F_3}{10^4} + \frac{F_4}{10^5} + \cdots && \text{expand sum} \\ &= \frac{F_0}{10^1} + \frac{F_1}{10^2} + \begin{cases} \dfrac{F_0}{10^3} + \dfrac{F_1}{10^4} + \dfrac{F_2}{10^5} + \cdots \\ \dfrac{F_1}{10^3} + \dfrac{F_2}{10^4} + \dfrac{F_3}{10^5} + \cdots \end{cases} && \text{Fibonacci definition (split up each term starting at } F_2) \\ &= \frac{F_0}{10^1} + \frac{F_1}{10^2} + \begin{cases} \dfrac{1}{10^2}\left(\dfrac{F_0}{10^1} + \dfrac{F_1}{10^2} + \dfrac{F_2}{10^3} + \cdots\right) \\ \dfrac{1}{10^1}\left(\underbrace{\dfrac{F_0}{10^1}}_{F_0=0} + \dfrac{F_1}{10^2} + \dfrac{F_2}{10^3} + \cdots\right) \end{cases} && \text{pull out common factors} \\ &= \frac{F_0}{10^1} + \frac{F_1}{10^2} + \frac{1}{10^2}(\text{RHS}) + \frac{1}{10^1}(\text{RHS}) && \text{this is our original sequence!} \end{aligned}$$

and so now we get that

$$-\frac{1}{100} = \text{RHS}\left(\frac{1}{10^2} + \frac{1}{10} - 1\right) \implies -\frac{1}{100} = \text{RHS}\left(-\frac{89}{100}\right) \implies \text{RHS} = \frac{1}{89}$$

Great! This is exactly what we wanted. ☺

Note 5.2 (On Convergence).
You may recall from calculus that infinite series are finicky. For example, consider

$$1 = 1 + (-1 + 1) + (-1 + 1) + \cdots$$

but then, just by regrouping the parentheses

$$0 = (1 - 1) + (1 - 1) + (1 - 1) + \cdots$$

so in general, we have to be very careful with infinite series.

However, in this class, we will primarily use them as a way to *guess* a formula for a recursion, which can always then be verified by induction. So we won't worry about convergence/divergence and the like here.

Note 5.3 (89 is a Fibonacci Number).
It might seem that something deep is going on here, since 89 is a Fibonacci number. Sadly, this is just a coincidence, it doesn't hold up in other bases.

However, this does motivate that it might be convenient to have a way to quickly figure out the fraction which does this in other bases. To do this, we will think of a function

$$f(z) = F_0 z^0 + F_1 z^1 + F_2 z^2 + F_3 z^3 + \cdots$$

where the coefficients are the terms of the Fibonacci sequence.

Definition 5.4 (Generating Function).
The generating function $f(z)$ of a recurrence with terms a_n is a function such that

$$f(z) = \sum_{i=0}^{\infty} a_n z^n = a_0 z^0 + a_1 z^1 + a_2 z^2 + \cdots$$

If we let $f(z)$ be the generating function for the Fibonacci numbers, our example showed that

$$0.011235 = \frac{1}{10} \cdot f\left(\frac{1}{10}\right)$$

Let's try and find $f(z)$ in general. We'll use the same trick as before:

$$\begin{aligned} f(z) &= F_0 z^0 + F_1 z^1 + F_2 z^2 + F_3 z^3 + \cdots \\ &= F_0 z^0 + F_1 z^1 + \begin{cases} F_0 z^2 + F_1 z^3 + F_4 z^4 + \cdots & = z^2 f(z) \\ F_1 z^2 + F_2 z^3 + F_3 z^4 + \cdots & = z f(z) \end{cases} \qquad \text{(because } F_0 = 0\text{)} \\ &= 0 + z + z f(z) + z^2 f(z) \end{aligned}$$

and so we get that

$$f(z) - z f(z) - z^2 f(z) = z \implies f(z)(1 - z - z^2) = z \implies f(z) = \frac{z}{1 - z - z^2}$$

and so we have a function which, when you look at the coefficients of the Taylor series of that function, will give you the Fibonacci numbers. Neat! ☺

Example 5.5 (Closed Form for Fibonacci with Generating Functions). Use the generating function for the Fibonacci sequence, $f(z)$, to find a closed form expression for F_n.

From calculus, we know that one approach could be to use the Taylor series:

$$f(z) = \sum_{k=0}^{\infty} \frac{f^{(k)}(0)}{k!} \cdot x^k \implies F_k = \frac{f^{(k)}(0)}{k!}$$

since F_k is the coefficient on x^k in $f(z)$. However, this is not particularly useful. Calculating the kth derivative of $f(z)$ gets painful very quickly. We need a better approach: we'll try and break $f(z)$ up into simpler functions.

One family of functions which are easy to work with are those of the form:

$$\frac{1}{1-cz} = 1 + cz + c^2z^2 + c^3z^3 + \cdots$$

by the geometric series formula. Note how easy it is to find the coefficient of z^k for these functions. So if we could break $f(z)$ up into a sum of functions of this form, that'd be great! We'll use a *partial fraction decomposition*[1]:

$$\begin{aligned} \frac{-z}{z^2+z-1} &= \frac{-z}{(z-\phi_1)(z-\phi_2)} && \text{where } \phi_1, \phi_2 = \text{roots of } z^2+z-1 \\ &= \frac{A}{z-\phi_1} + \frac{B}{z-\phi_2} && \text{for some constants } A \text{ and } B \end{aligned}$$

Now, multiplying through by $(z-\phi_1)(z-\phi_2)$ gives us the equation

$$-z = A(z-\phi_1) + B(z-\phi_2)$$

Now, we plug in values for z to get rid of one of the constants

- $(z = \phi_1)$: We get $-\phi_2 = B(\phi_2 - \phi_1) \implies B = \frac{\phi_2}{\phi_1 - \phi_2}$.
- $(z = \phi_1)$: We get $-\phi_1 = A(\phi_1 - \phi_2) \implies A = \frac{-\phi_1}{\phi_1 - \phi_2}$. (We won't put the $-$ in the denominator so that these fractions have a common denominator.)

From here, it's just an algebra slog:

$$\begin{aligned} \frac{-z}{z^2+z-1} &= \frac{\frac{-\phi_1}{\phi_1-\phi_2}}{z-\phi_1} + \frac{\frac{\phi_2}{\phi_1-\phi_2}}{z-\phi_2} && \text{plug in for } A \text{ and } B \\ &= \frac{\frac{-\phi_1}{\phi_1-\phi_2} \cdot -\frac{1}{\phi_1}}{(z-\phi_1)\cdot -\frac{1}{\phi_1}} + \frac{\frac{\phi_2}{\phi_1-\phi_2} \cdot -\frac{1}{\phi_2}}{(z-\phi_2)\cdot -\frac{1}{\phi_1}} && \text{multiply by a clever 1} \end{aligned}$$

We do this with the goal of making the denominator look like $1-cz$, so that these fall into our "nice" family of functions.

$$= \frac{1}{\phi_1 - \phi_2}\left(\frac{1}{1-\frac{1}{\phi_1}z} - \frac{1}{1-\frac{1}{\phi_2}z}\right) \qquad \text{simplify}$$

[1] Hopefully you've seen this in calculus before; if not, we'll do a bunch of examples of it throughout this chapter. Also see: `http://tutorial.math.lamar.edu/Classes/CalcII/PartialFractions.aspx`

From here, using the geometric series formula, we can write

$$\frac{1}{1-\frac{1}{\phi_1}z} = 1 + \left(\frac{1}{\phi_1}\right)z + \left(\frac{1}{\phi_1}\right)^2 z^2 + \cdots + \left(\frac{1}{\phi_1}\right)^n z^n + \cdots$$
$$\frac{1}{1-\frac{1}{\phi_2}z} = 1 + \left(\frac{1}{\phi_2}\right)z + \left(\frac{1}{\phi_2}\right)^2 z^2 + \cdots + \left(\frac{1}{\phi_2}\right)^n z^n + \cdots$$

and therefore

$$\frac{1}{1-\frac{1}{\phi_1}z} - \frac{1}{1-\frac{1}{\phi_2}z} = (1-1)z^0 + \left(\frac{1}{\phi_1} - \frac{1}{\phi_2}\right)z^1 + \cdots + \left(\frac{1}{\phi_1^n} - \frac{1}{\phi_2^n}\right)z^n + \cdots$$

It follows that the coefficient on z^n in $f(z)$ is

$$\text{coefficient on } z^n = F_n = \frac{1}{\phi_1 - \phi_2}\left(\frac{1}{\phi_1^n} - \frac{1}{\phi_2^n}\right) = \cdots = \frac{1}{\sqrt{5}}\left[\left(\tfrac{1+\sqrt{5}}{2}\right)^n - \left(\tfrac{1-\sqrt{5}}{2}\right)^n\right]$$

which is a closed form expression for the nth Fibonacci number. From here, you can prove this expression is valid by induction on n. ☺

Note 5.6 (Why Generating Functions?).
You may ask why this is useful if we could already do this with matrices. In general, this method is more adaptable; we'll use it to solve recurrences whose characteristic polynomial has a repeated root, a situation we couldn't deal with using matrices.

5.2 Solving General Recurrences with Generating Functions

Theorem 5.7 (Solution to a 2-Term Linear Recurrence).
The linear recurrence $a_n = Aa_{n-1} + Ba_{n-2}$ with initial conditions a_0 and a_1 solves to

$$a_n = \alpha \cdot \lambda_1^n + \beta \cdot \lambda_2^n$$

where λ_1 and λ_2 are the roots of the characteristic polynomial of the recurrence, and α, β are some constants.

Proof. We'll walk through the general steps which go into solving this using generating functions:

(1) Write $f(z) = a_0 + a_1 z + a_2 z^2 + \cdots$.

(2) Use the recurrence to find a closed form for $f(z)$:

$$\begin{aligned} f(z) &= a_0 + a_1 z + \begin{cases} Aa_1 z^2 + Aa_2 z^3 + \cdots \\ Ba_0 z^2 + Ba_1 z^3 + \cdots \end{cases} \\ &= a_0 + a_1 z + Az(f(z) - a_0) + Bz^2 \cdot f(z) \end{aligned}$$

and so solving for $f(z)$ we get

$$f(z) = \frac{a_0 + z(a_1 - Aa_0)}{1 - Az - Bz^2}$$

(3) Break $f(z)$ into simpler parts. We want to write

$$f(z) = \frac{\alpha}{1-cz} + \frac{\beta}{1-dz} \qquad \text{where } c, d \text{ are the roots of the characteristic polynomial}$$

Why can we use the characteristic polynomial to get the values c and d? Well, we know we want

$$\begin{aligned} 1 - Az - Bz^2 &= (1-cz)(1-dz) \\ \implies 1 - A\tfrac{1}{x} - B\tfrac{1}{x^2} &= (1-c\tfrac{1}{x})(1-d\tfrac{1}{x}) && \text{let } z = \tfrac{1}{x} \\ \implies x^2 - Ax - B &= (x-c)(x-d) && \text{multiply by } x^2 \end{aligned}$$

and so c and d are just the roots of the characteristic polynomial.

(4) Now we get

$$\begin{aligned} f(z) = \frac{\alpha}{1-\lambda_1 z} + \frac{\beta}{1-\lambda_2 z} &= \sum_{n=0}^{\infty} \alpha \cdot \lambda_1^n z^n + \sum_{n=0}^{\infty} \beta \cdot \lambda_1^n z^n \\ &= \sum_{n=0}^{\infty} (\alpha \cdot \lambda_1^n + \beta \cdot \lambda_2^n) \cdot z^n \end{aligned}$$

and so $a_n = \alpha \cdot \lambda_1^n + \beta \cdot \lambda_2^n$.

From here, you can solve for α and β using two values for n, say $n = 0$ and $n = 1$. □

5.3 Dealing with Repeated Roots

Example 5.8 (Another Linear Recurrence).
Find a solution to $a_n = 6a_{n-1} - 9a_{n-2}$ with initial conditions $a_0 = 2$, $a_1 = 9$.

Recall from the previous section that the generating function for a recurrence of this form is

$$f(z) = \frac{a_0 + z(a_1 - Aa_0)}{1 - Az - Bz^2} = \frac{2 - 3z}{1 - 6z - 9z^2}$$

The characteristic polynomial is

$$x^2 - 6x - 9 = 0 \implies (x-3)^2 = 0 \implies \lambda_1, \lambda_2 = 3$$

and so we want to write

$$\frac{2-3z}{1-6z-9z^2} = \frac{\alpha}{1-3z} + \frac{\beta}{1-3z}$$

Uh-oh. If you try to do this, you will quickly see that you can't. One way to see this is that the LHS has a denominator of order 2, while the RHS has one of order 1. However, we can use a trick from partial fractions and try to write

$$\frac{2-3z}{1-6z-9z^2} = \frac{\alpha}{1-3z} + \frac{\beta}{(1-3z)^2}$$

This is more manageable. Multiply through by $(1-3z)^2$ to get

$$2 - 3z = \alpha(1-3z) + \beta \implies \begin{cases} \alpha + \beta = 2 \\ -3\alpha = -3 \end{cases} \implies \begin{cases} \alpha = 1 \\ \beta = 1 \end{cases}$$

so we can write

$$\frac{2-3z}{1-6z-9z^2} = \frac{1}{1-3z} + \frac{1}{(1-3z)^2}$$

We know how to deal with the first term from before, but how do we deal with a squared term in the denominator? We'll differentiate!

$$\begin{aligned}
\frac{1}{1-3z} = \sum_{n=0}^{\infty} 3^n z^n \implies \frac{3}{(1-3z)^2} &= \sum_{n=0}^{\infty} 3^n n \cdot z^{n-1} && \text{take derivative} \\
\implies \frac{1}{(1-3z)^2} &= \sum_{n=0}^{\infty} 3^{n-1} n \cdot z^{n-1} && \text{divide by 3} \\
&= \sum_{n=-1}^{\infty} 3^n (n+1) \cdot z^n && \text{reindex} \\
&= \sum_{n=0}^{\infty} 3^n (n+1) \cdot z^n && \text{first term is 0}
\end{aligned}$$

and so we can write $f(z)$ as

$$f(z) = \sum_{n=0}^{\infty} 3^n \cdot z^n + \sum_{n=0}^{\infty} 3^n (n+1) \cdot z^n = \sum_{n=0}^{\infty} (3^n + (n+1)3^n) \cdot z^n$$

and so $a_n = 3^n \cdot (n+2)$. ☺

Before we get to solving a general recurrence with repeated roots, we need to introduce some additional machinery:

Definition 5.9 (Generalized Binomial Coefficient).
If $\alpha \in \mathbb{R}$ and $n \in \mathbb{N}$, we define the generalized binomial coefficient $\binom{\alpha}{n}$ as

$$\binom{\alpha}{n} = \frac{(\alpha)_n}{n!} = \frac{\alpha(\alpha-1)(\alpha-2)\cdots(\alpha-n+1)}{n!}$$

Theorem 5.10 (Newton's Generalized Binomial Formula).
For all $x, y, r \in \mathbb{R}$, we have

$$(x+y)^r = \sum_{n=0}^{\infty} \binom{r}{n} x^k y^{r-k}$$

Proof. Recall from calculus that the Taylor series about 0 of a function $f(x)$ is

$$f(x) = f(0) + f'(0) \cdot x + \frac{f''(0)}{2!} \cdot x^2 + \frac{f'''(0)}{3!} \cdot x^3 + \cdots = \sum_{n=0}^{\infty} \frac{f^{(n)}(0)}{n!} \cdot x^n$$

and so we can express $(x+y)^r$ as an infinite series using its Taylor representation. It suffices to find an expression for the nth derivative of $(x+y)^r$. Let y be fixed and let x vary. Then

$$\begin{aligned}
\frac{d^n}{dx^n}(x+y)^r &= r(r-1)(r-2)\cdots(r-n+1)(x+y)^{r-n} \\
\implies f^{(n)}(0) &= r(r-1)(r-2)\cdots(r-n+1) \cdot y^{r-n}
\end{aligned}$$

so we can write

$$(x+y)^r = \sum_{n=0}^{\infty} \frac{r(r-1)(r-2)\cdots(r-n+1)}{n!} \cdot x^k y^{r-k} = \sum_{n=0}^{\infty} \binom{r}{n} \cdot x^k y^{r-k}$$

which is what we wanted. □

Theorem 5.11 (General Recurrences with Repeated Roots).
The recurrence $a_n = A_1 a_{n-1} + A_n a_{n-2} + \cdots + A_k a_{n-k}$ with initial conditions $a_0, a_1, \cdots, a_k$ and with characteristic polynomial $(x-\lambda)^k = 0$ has solution

$$a_n = \alpha \cdot \lambda^n + \beta \cdot n\lambda^n + \cdots + \omega \cdot n^{k-1}\lambda^n$$

for some constants $\alpha, \beta, \cdots, \omega$.

Proof. Following the pattern from the example, we can express our generating function $f(z)$ as

$$f(z) = \frac{\alpha}{1-\lambda z} + \frac{\beta}{(1-\lambda z)^2} + \cdots + \frac{\omega}{(1-\lambda z)^k}$$

so it suffices to find a way to write $(1-\lambda z)^{-r}$ as an infinite series for some general r. We'll use Newton's Generalized Binomial Formula:

$$\begin{aligned}
\frac{1}{(1-\lambda z)^r} &= \sum_{n=0}^{\infty} \binom{-r}{n} (-\lambda z)^n = \sum_{n=0}^{\infty} \binom{-r}{n} (-1)^n \lambda^n z^n \\
&= \sum_{n=0}^{\infty} \frac{(-r)(-r-1)(-r-2)\cdots(-r-n+1)}{n!} (-1)^n \lambda^n z^n \\
&= \sum_{n=0}^{\infty} \frac{r(r+1)(r+2)\cdots(r+n-1)}{n!} \cdot \lambda^n z^n \\
&= \sum_{n=0}^{\infty} \binom{n+r-1}{n} \cdot \lambda^n z^n
\end{aligned}$$

But notice that we can write

$$\binom{n+r-1}{n} = n^{r-1} + \text{Junk of lower order}$$

Let $LO(n^k)$ denote any collection of terms with order less than k. Then

$$\begin{aligned}
f(z) &= \sum_{n=0}^{\infty} \alpha\lambda^n z^n + \sum_{n=0}^{\infty} \beta(n + LO(n))\lambda^n z^n + \cdots + \sum_{n=0}^{\infty} \omega(n^{k-1} + LO(n^{k-1}))\lambda^n z^n \\
&= \sum_{n=0}^{\infty} (\alpha' + \beta' n + \gamma' n^2 + \cdots + \omega' n^{k-1}) \cdot \lambda^n z^n
\end{aligned}$$

for some new constants $\alpha', \beta', \gamma', \cdots, \omega'$. We get this by combining each of the $LO(-)$ terms into a single term under a different constant. Therefore, we can write

$$a_n = \alpha \cdot \lambda^n + \beta \cdot n\lambda^n + \cdots + \omega \cdot n^{k-1}\lambda^n$$

which is what we wanted to show. □

Note 5.12 (Pirates and Gold Again?).
You may have noticed that the pirates and gold formula showed up in the middle of this proof. What is it doing here? Well, we can think of

$$\frac{1}{(1-\lambda z)^k} = \underbrace{(1 + \lambda z + \lambda^2 z^2 + \cdots)\cdots(1 + \lambda z + \lambda^2 z^2 + \cdots)}_{k \text{ times}}$$

and take $\lambda^{x_i} z^{x_i}$ from the ith series. To get a coefficient of z^n, we need $x_1 + x_2 + \cdots + x_k = n$, which is the same as giving x_i gold to the ith pirate, out of n total. Hence, the pirates and gold formula!

5.4 Recurrence Recap

Example 5.13 (Finding the Form of a Solution).
Find the general form of a solution to the recurrence

$$a_n = 19a_{n-1} - 94a_{n-2} + 102a_{n-3} + 243a_{n-4} - 297a_{n-5} - 324a_{n-6}$$

with initial conditions $a_0, a_1, \cdots, a_5$.

The first step is to find our characteristic polynomial. We have

$$x^6 - 19x^5 + 94x^4 - 102x^3 - 243x^2 + 297x + 324 = 0$$

which factors to $(x-3)^3(x+1)^2(x-12) = 0$.[2] Therefore, we can write a general solution as

$$a_n = \underbrace{\alpha \cdot 3^n + \beta \cdot n3^n + \gamma \cdot n^2 3^n}_{\text{from } (x-3)^3} + \underbrace{\delta \cdot (-1)^n + \varepsilon \cdot n(-1)^n}_{\text{from } (x+1)^2} + \underbrace{\zeta \cdot 12^n}_{\text{from } (x-12)}$$

and then could painfully set up and solve a system of 6 equations using $a_0, a_1, \cdots, a_5$ to get the constants $\alpha, \beta, \gamma, \delta, \varepsilon$, and ζ. ☺

Important 5.14 (The Swiss Army Knife of Discrete Math).
For all the fanciness going on with generating functions and matrix equations, it's easy to forget the power of trying small examples, and guessing an answer based on the pattern.

Trying small examples is the single most powerful tool you have in this class. Be sure to not forget it!

Example 5.15 (A Nonlinear Recurrence).
Solve the recurrence $a_{n+1} = 2a_n + 1$ with $a_0 = 0$.

We'll illustrate 3 different approaches to the problem:

(1) *The Swiss Army Knife Method*

Let's try some small values:

n	0	1	2	3	4	5	$\cdots$
a_n	0	1	3	7	15	31	$\cdots$

It looks a lot like $a_n = 2^n - 1$. We can prove this by induction:

BC We have $a_0 = 0$ and $2^0 - 1 = 0$. Great!

IS Assume true for n. Then $a_{n+1} = 2a_n + 1 = 2(2^n - 1) + 1 = 2^{n+1} + 1$.

so the claim holds for all n by induction.

[2] You definitely don't need to know how to factor this. We just need an example with a bunch of repeated roots so we can illustrate the techniques in this chapter.

(2) *Generating Functions*

Write $f(z) = a_0 + a_1 z + a_2 z^2 + \cdots$, and use the recurrence definition to get

$$\begin{aligned}
f(z) &= a_0 + (2a_0 + 1)z + (2a_1 + 1)z^2 + (2a_2 + 1)z^3 + \cdots && \text{definition of } a_n \\
&= \begin{cases} 2a_0 z + 2a_1 z^2 + 2a_2 z^3 + \cdots \\ a_0 + z + z^2 + z^3 + \cdots \end{cases} && \text{collect terms} \\
&= 2zf(z) + z(1 + z + z^2 + \cdots) && \text{substitute for } f(z) \\
&= 2zf(z) + \frac{z}{1-z} && \text{geometric series}
\end{aligned}$$

and so we get that

$$f(z)(1-2z) = \frac{z}{1-z} \implies f(z) = \frac{z}{(1-z)(1-2z)} = \frac{A}{1-z} + \frac{B}{1-2z}$$

for some constants A and B. This implies

$$f(z) = A\sum_{i=0}^{\infty} z^n + B\sum_{i=0}^{\infty} 2^n z^n = \sum_{i=0}^{\infty}(A + B \cdot 2^n)z^n$$

and so $a_n = A + B \cdot 2^n$ for some A and B. Then, plugging in $n = 0, 1$ gives

$$\begin{cases} A + B = 0 \\ A + 2B = 1 \end{cases} \implies A = -1,\ B = 1$$

so $a_n = 2^n - 1$. We can then prove this by induction as above.

(3) *Define a New Sequence*

The sequence (a_n) is annoying because of the $+1$. To get rid of it, we can define a new sequence $b_n = a_n + 1$. Then

$$\begin{aligned}
a_0 = 0 \quad &\implies \quad b_0 = 1 \\
a_{n+1} = 2a_n + 1 \quad &\implies \quad b_{n+1} = a_{n+1} + 1 = 2a_n + 2 = 2(b_n - 1) + 2 = 2b_n
\end{aligned}$$

It's pretty clear that $b_n = 2^n$ (you don't even need to do induction to prove this... in 228 we can just say it's obvious). Therefore

$$b_n = 2^n \implies a_n + 1 = 2^n \implies a_n = 2^n - 1$$

Nicely, we get the same answer from all three methods. Math works! ☺

CHAPTER 6

The Catalan Numbers

In the modern mathematical literature, Catalan numbers are wonderfully ubiquitous. Althoughthey appear in a variety of disguises, we are so used to having them around, it is perhaps hard toimagine a time when they were either unknown or known but obscure and underappreciated

—Igor Pak

Contents

6.1 What are the Catalan Numbers?

Definition 6.1 (The Catalan Numbers).
The nth Catalan number is given by the relations:

$$C_0 = 1 \qquad \text{and} \qquad C_{n+1} = \sum_{i=0}^{n} C_i C_{n-i}$$

The first few Catalan numbers are: 1, 1, 2, 5, 14, 42, 132, 429, 1430, 4862...

Note 6.2 (Pattern Recognition).
If in your life you ever are working on a problem and the first few cases give you the numbers $1, 1, 2, 5, 14$, it will almost always be the Catalan numbers. Very few other meaningful sequences start this way.

Theorem 6.3 (Closed Form for C_n).
The nth Catalan number is given by the expression

$$C_n = \frac{1}{n+1}\binom{2n}{n}$$

Proof. We'll use generating functions. Let $f(z) = C_0 + C_1 z + C_2 z^2 + \cdots$. Then

$$\begin{aligned} f(z)^2 &= (C_0 + C_1 z + C_2 z^2 + \cdots)(C_0 + C_1 z + C_2 z^2 + \cdots) \\ &= \sum_{n=0}^{\infty} (C_0 C_n + C_1 C_{n-1} + \cdots + C_{n-1} C_1 + C_n C_0) z^n \end{aligned}$$

but note that this is the Catalan recurrence! So we can write

$$\begin{aligned} f(z)^2 = \sum_{n=0}^{\infty} C_{n+1} z^n &= C_1 + C_2 z + C_3 z^2 + \cdots \\ \implies z f(z)^2 &= C_1 z + C_2 z^2 + C_3 z^3 + \cdots && \text{multiply by } z \\ \implies z f(z)^2 &= f(z) - C_0 && \text{definition of } f(z) \\ \implies z f(z)^2 - f(z) + 1 &= 0 && C_0 = 1, \text{ rearrange} \\ \implies f(z) &= \frac{1 \pm \sqrt{1-4z}}{2z} && \text{solve for } f(z) \end{aligned}$$

Should we should use the $+$ or the $-$? We know that $f(0) = 1$ and

$$\lim_{z \to 0^+} \frac{1+\sqrt{1-4z}}{2z} = \infty \qquad \text{and} \qquad \lim_{z \to 0^+} \frac{1-\sqrt{1-4z}}{2z} = 1$$

so we want to use the negative root. Now, how do we deal with the $\sqrt{1-4z}$? We can use Newton's Generalized Binomial Theorem:

$$\sqrt{1-4z} = (1-4z)^{1/2} = \sum_{n=0}^{\infty} \binom{1/2}{n} (-4z)^n = \sum_{n=0}^{\infty} \frac{\frac{1}{2}\left(-\frac{1}{2}\right)\left(-\frac{3}{2}\right)\cdots\left(-\frac{2n-3}{2}\right)}{n!} (-4z)^n$$

$$= -\sum_{n=0}^{\infty} \frac{1 \cdot 3 \cdot 5 \cdots 2n-3}{2^n n!} \cdot 4^n z^n = -\sum_{n=0}^{\infty} \frac{1 \cdot 3 \cdot 5 \cdots 2n-3}{n!} \cdot 2^n z^n$$
$$= -\sum_{n=0}^{\infty} \frac{1 \cdot 3 \cdot 5 \cdots 2n-3}{n!} \cdot \frac{2 \cdot 4 \cdot 6 \cdots 2n-2}{2 \cdot 4 \cdot 6 \cdots 2n-2} \cdot 2^n z^n = -\sum_{n=0}^{\infty} \frac{(2n-2)!}{n!(n-1)!} \cdot 2z^n$$
$$= -\sum_{n=0}^{\infty} \frac{2}{n} \frac{(2n-2)!}{(n-1)!(n-1)!} \cdot 2z^n = -\sum_{n=0}^{\infty} \frac{2}{n} \binom{2(n-1)}{n-1} \cdot z^n$$

This algebra is almost legit; technically the term in the sum is undefined for $n = 0$. You can plug into the original equation to see that the $n = 0$ term should be 1. Therefore, we can write $f(z)$ as

$$f(z) = \frac{1 - (1 - \sum_{n=1}^{\infty} \frac{2}{n} \binom{2(n-1)}{n-1} \cdot z^n)}{2z} = \frac{\sum_{n=1}^{\infty} \frac{2}{n} \binom{2(n-1)}{n-1} \cdot z^n}{2z}$$
$$= \sum_{n=1}^{\infty} \frac{1}{n} \binom{2(n-1)}{n-1} \cdot z^{n-1} = \sum_{n=0}^{\infty} \frac{1}{n+1} \binom{2n}{n} \cdot z^n$$

and so $C_n = \frac{1}{n+1}\binom{2n}{n}$, as desired. □

The Catalan numbers show up in many interesting combinatorics problems. We'll go through a few of them in this chapter. Here's one:

Example 6.4 (Valid Sequences of Parentheses).
How many valid sequences of parentheses, with n each of `(` and `)` are there? For example,

`(())()` is valid `)()(()` is not valid

We'll call the answer P_n. Let's try some small cases:

$P_1 = 1$	`()`
$P_2 = 2$	`()()`, or `(())`
$P_3 = 5$	`()()()`, `((()))`, `(())()`, `()(())`, or `(()())`

This is looking awfully like the Catalan sequence. Let's try to do a general form: with n sets of parentheses, we get something that looks like:

$$(\underbrace{\text{some smaller case}}_{L_1 \text{ parens}})(\underbrace{\text{some smaller case}}_{L_2 \text{ parens}})$$

so to get the total, we can just add up all the locations of the `)(`. We know that $L_1 = 2k$ for some k, and so $L_2 = n - 2k - 2 = 2(n - k - 1)$, so in total

$$P_n = \sum_{\text{all breaks}} P_{L_1/2} \cdot P_{L_2/2} = \sum_{k=0}^{n-1} P_k P_{(n-1)-k}$$

but this is just the Catalan recurrence, so $P_n = C_n$. ☺

Note 6.5 (A Probabilistic Interpretation).
One way of viewing this is that if you randomly arrange n pairs of `()`, then the probability of it being valid is $\frac{1}{n+1}$, since

$$\Pr[\text{valid sequence}] = \frac{\#\text{ valid sequences}}{\#\text{ total sequences}} = \frac{\frac{1}{n+1}\binom{2n}{n}}{\binom{2n}{n}} = \frac{1}{n+1}$$

6.2 Triangulations

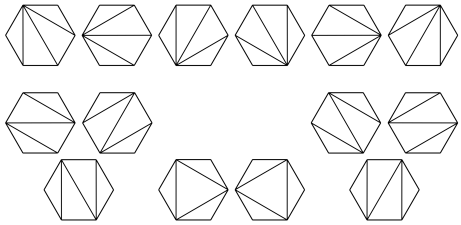
Example 6.6 (Triangulations of a Convex Polygon).
How many *triangulations*[a] are there of an n-sided convex polygon?

For example, when $n = 4$, the following are all the triangulations:

[a] We won't define triangulation formally, but you can think of it as a way to break a shape up into triangles while adding no new points.

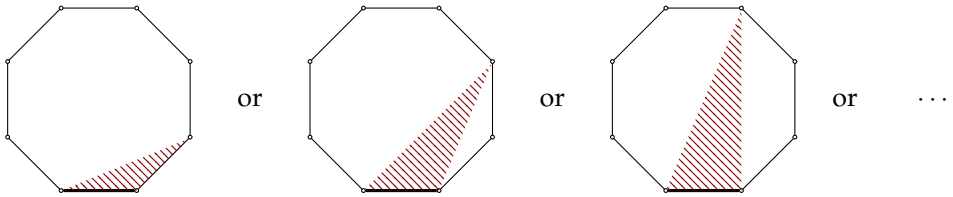
Let's try and work through some small examples. When $n = 3$, there is obviously only 1 triangulation (just the triangle itself). From the example above, we know there are 2 triangulations when $n = 4$. For $n = 5$, we get:

for a total of 5 triangulations, and when $n = 6$, we get



for a total of 14 triangulations. This certainly is beginning to look like the Catalan sequence. Let's see if we can prove it!

We claim that the number of triangulations of a convex n-gon is C_{n-2}, for $n \geq 3$. Our approach, as with the parentheses problem, will be to recover the Catalan recursion. Call T_n the number of triangulations of an n-gon. Now, consider some edge in the n-gon. It must be part of some triangle. Partition all the possible triangluations by the third point of this triangle. For example:



In general, doing this for an n-sided polygon splits it into a k-gon and a $(n - k + 1)$-gon, which we can recursively compute. If we sum over all the k's, we'll get that the total number of triangluations is

$$T_n = T_2T_{n-1} + T_3T_{n-2} + \cdots + T_{n-2}T_3 + T_{n-1}T_2$$

If we define $T_2 = 1$ for convenience. Since the question never asks about 2-sided polygons, we are allowed to choose whatever value we want for T_2.

Now, we know that

$$\begin{cases} T_2 = 1 \\ T_n = T_2T_{n-1} + T_3T_{n-2} + \cdots + T_{n-2}T_3 + T_{n-1}T_2 \end{cases}$$

which looks awfully like the Catalan recurrence. We conjecture that $T_n = C_{n-2}$. The proof is by induction on n:

BC When $n = 2$, we have $T_2 = 1$ and $C_{2-2} = C_0 = 1$, so the base case holds.

IS Assume the claim holds for all values smaller than or equal to n. Then

$$\begin{aligned} T_{n+1} &= T_2T_{n-1} + T_3T_{n-2} + \cdots + T_{n-2}T_3 + T_{n-1}T_2 \\ &= C_0C_{n-2} + C_1C_{n-3} + \cdots + C_{n-3}C_1 + C_{n-2}C_0 \\ &= C_{n-1} \end{aligned}$$

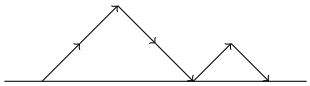
by the definition of C_n.

Therefore, by induction, the claim holds for all n. ☺

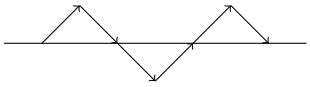
6.3 Mountain Pictures and More Grid Walks

Example 6.7 (Mountain Pictures).
How many mountain pictures with n ↗ and n ↘ are there?

Here is an example (and a non-example) of a mountain picture:



A valid mountain picture



And an invalid one

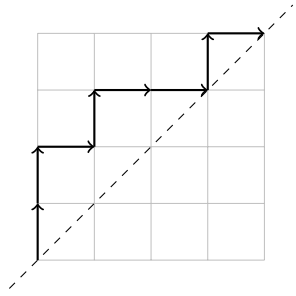
In words, a mountain picture can never go below the ground, and must start and end at ground level.

We can get a bijection between mountain pictures and valid sets of parentheses by mapping

$$\text{“}\nearrow\text{”} \mapsto \text{“(”} \qquad \text{and} \qquad \text{“}\searrow\text{”} \mapsto \text{“)”}$$

and so we see that there are exactly C_n mountain pictures with n ↗ and n ↘. ☺

Example 6.8 (Grid Walks, Again...).
How many grid walks are there in an $n \times n$ grid which stay in the upper triangle of the grid?



For example, this is a valid grid walk under these rules.

Notice that this is the same problem as the mountain pictures problem... just turn your head 45°to the left.

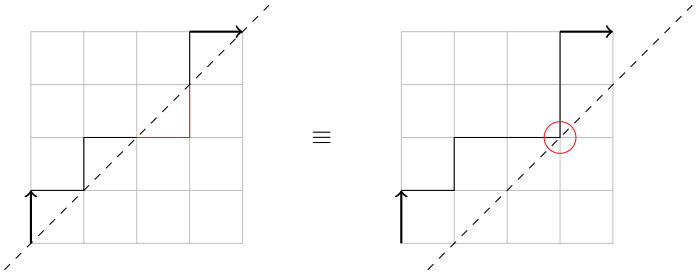
So we know that the number of these pictures is C_n. However, this time, we'll give a combinatorial proof of the closed form expression for C_n.

We know that all U/R walks must start with an $\uparrow$ and end with a $\rightarrow$, otherwise they will pass over the diagonal. (For the rest of the problem, we will only talk about grid pictures of this form.) Therefore, the answer is

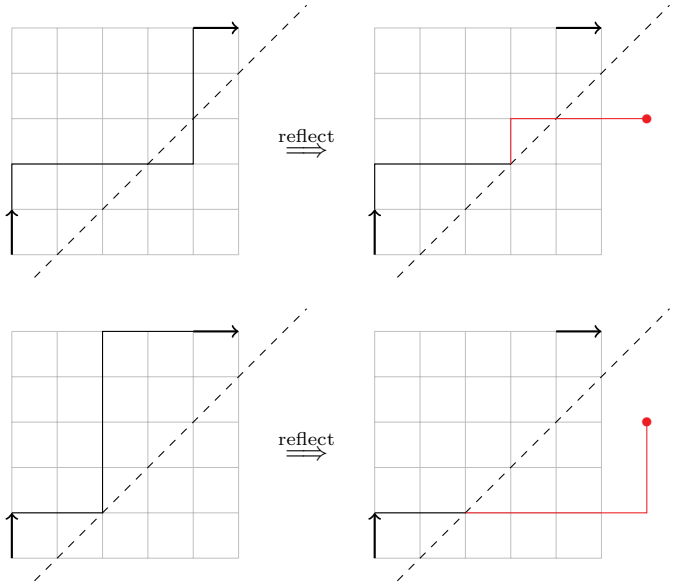
$$\text{ANS} = \left(\#\text{ of } \quad \text{total} \right) - \left(\#\text{ of } \quad \text{crossing the diagonal} \right)$$

$$= \binom{2(n-1)}{n-1} - \text{???}$$

so it suffices to find the number of U/R pictures which cross the diagonal. One way to do this is to notice that paths that *cross* the diagonal are the same as paths which *touch* the line shifted down/right by 1.



Now, consider some path which touches the shifted line, and reflect it about the line after the first time it touches the line. Here are some examples:



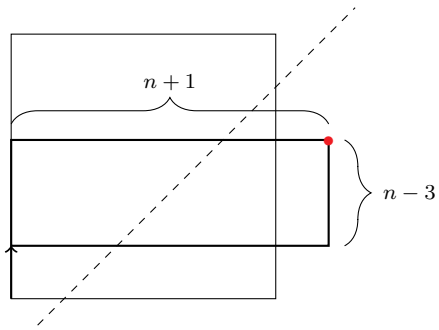
It looks like we are always ending up in the same place in the reflected pictures. Let's try and use this. We will try and set up a bijection between pictures which hit the shifted line and pictures which end at the red node. The function we will consider is

$$f : \left\langle \begin{matrix}\text{Grid Pictures} \\ \text{Touching Line}\end{matrix} \right\rangle \to \left\langle \begin{matrix}\text{Grid Pictures} \\ \text{Ending at } \bullet\end{matrix} \right\rangle \qquad \text{by} \qquad f(P) = P', \text{ where } P \overset{\text{reflect}}{\Longrightarrow} P'$$

We have to show that the function is well-defined, an injection, and a surjection:

- WELL-DEFINED. We are only considering grid pictures which touch the line, so our reflection operation is valid.
- INJECTIVE. We claim that you cannot get the same reflection twice. This is pretty clear, since we can establish an inverse—just reflect again at the first point which touches the line.
- SURJECTIVE. We claim that we can get every picture which starts at (0,1) and ends at the red dot. Since any such picture starts and ends on on opposite sides of the line, it must cross it. Then, we can reflect at the first such point to get a picture P which $f(P)$ gives us the picture we started with.

So we know that the number of pictures which cross the original diagonal is just the number of pictures which go from (0,1) to the red dot. This is easier to count. We have that:



The paths we are interested in are the ones which go from the bottom-left corner of the bold box to the top-right one. This is just

$$\binom{(n-3)+(n+1)}{n+1} = \binom{2n-2}{n+1} \text{ paths}$$

since we can, for example, pick the location of the up arrows. Thus, we get that our final answer is

$$\text{ANS} = \binom{2n-2}{n-1} - \binom{2n-2}{n+1}$$

This is good enough, but we're going to do some more algebra to get it into the familiar form we had previously:

$$\begin{aligned}
\binom{2n-2}{n-1} - \binom{2n-2}{n+1} &= \frac{(2n-2)!}{(n-1)!(n-1)!} - \frac{(2n-2)!}{(n+1)!(n-3)!} \\
&= \frac{(2n)!}{n!n!} \cdot \frac{n \cdot n}{2n(2n-1)} - \frac{(2n)!}{n!n!} \cdot \frac{n(n-1)(n-2)}{2n(2n-1)(n+1)} \\
&= \binom{2n}{n} \left[\frac{n}{2(2n-1)} - \frac{(n-1)(n-2)}{2(2n-1)(n+1)} \right] \\
&= \frac{1}{n+1}\binom{2n}{n}^{[1]}
\end{aligned}$$

which is the expression for C_n we had previously. ☺

Note 6.9 (Forcing Binomial Coefficients).
The above algebra illustrates a general technique of moving from one binomial coefficient to another by just writing it, and multiplying by the relevant fraction to compensate.

[1] I won't bore you with the algebra from here... just work away at the stuff in the $[\cdots]$.

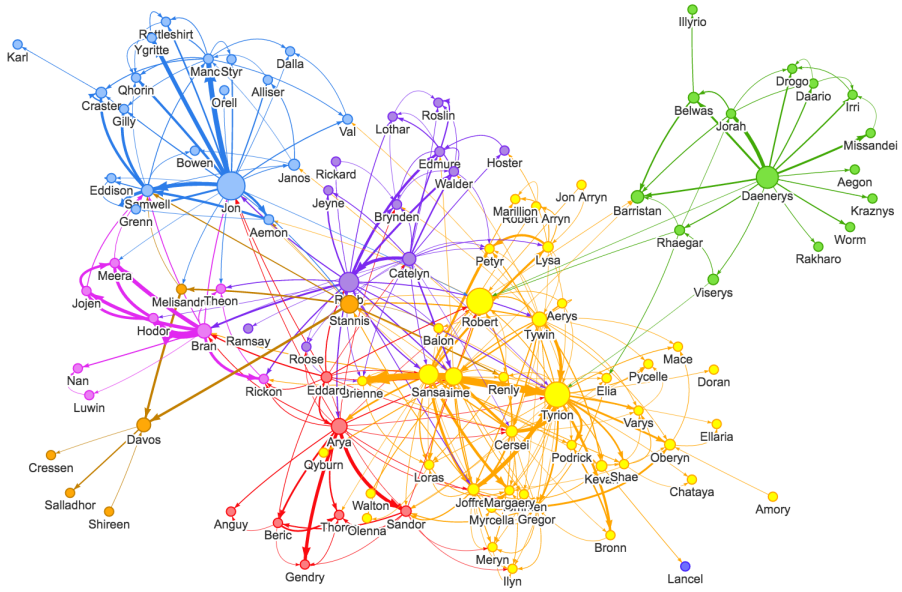
Part

III

Graph Theory

CHAPTER 7

Introduction to Graphs



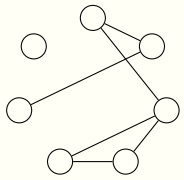
Graphs are what graphic designers use when they have to make smart looking math. Hence the name!

—Po-Shen Loh

Contents

7.1 What is a Graph?

Definition 7.1 (Graph).
Informally, a graph is any picture you can make with vertices and edges, such as the one to the right.



Formally, a graph G is a tuple $G = (V, E)$ where

- V is a nonempty set of *vertices*; and
- E is a set of *edges*, which are 2-element subsets of the vertex set V (i.e. $\{v_1, v_2\}$ s.t. $v_1, v_2 \in V$)

Note that our definition of a graph disallows the following:

- *Multiple Edges:*

  Since E is a set, it cannot have the same edge in it twice. Therefore, our graphs cannot have the construction seen to the right in them.

- *Self Loops:*

  Since the elements of R are sets, they cannot have the same vertex in them twice. Therefore, our graphs cannot have the construction seen to the right in them.

Sometimes, we call graphs without either of these features *simple graphs*. In this class, when we say graph, we mean a simple graph.

Note 7.2 (What Are Graphs Useful For?).
Graphs come up in many different fields, for example:

- In biology, we can represent proteins as vertices in a graph, and edges relate similar proteins.
- Facebook uses graph to analyze social relationships, with the vertices representing people, and edges representing friendships.
- Graphic designers use graphs as examples of smart looking math ☺

Example 7.3 (Shaking Hands).
Can there be a party with 7 people in which each person shakes exactly 3 other people's hands?

No, there can't. To see this, consider a count of total number of hands shaken. Whenever a person shakes another's hand, they add 1 to the count. In this way, every handshake between 2 people adds 2 to the count: 1 from each person.

If 7 people each shake 3 other people's hands, then the count should end at 21. However, each handshake adds 2 to the count, so it can never be odd. Therefore, this setup is impossible. ☺

Definition 7.4 (Lots of Graph Related Terminology).
Let $G = (V, E)$ be a graph. Then

- If $e = \{u, v\} \in E$ is an edge in the graph, we say that u and v are *neighbors* or *adjacent*. We also say u and v are *incident* to e.
- For $v \in V$, define the neighborhood of v, denoted $N(v)$, as the set of all neighbors of v, i.e. $N(v) = \{u \mid \{v, u\} \in E\}$.
- For $v \in V$, define the degree of v, denoted $\deg(v)$, as the number of edges incident to v. In a simple graph, $\deg(v) = |N(v)|$.

Lemma 7.5 (Handshake Lemma).
For any graph $G = (V, E)$, we have

$$\sum_{v \in V} \deg(v) = 2|E|$$

Proof. We prove the equality by double counting. For each vertex $v \in V$, put a token on all the edges it is incident to. We will count the total number of tokens.

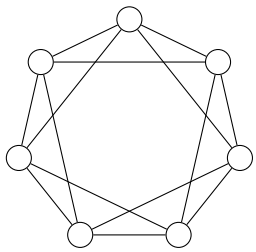
- Every vertex v is incident to $\deg(v)$ edges, so the total number of tokens put is $\sum_{v \in V} \deg(v)$.
- Each edge u, v in the graph will get two tokens, one from vertex u and one from vertex v. So the total number of tokens put is $2|E|$.

Since each of these expressions counts the same set, we get $\sum_{v \in V} \deg(v) = 2|E|$. □

Definition 7.6 (d-Regular Graph).
A graph $G = (V, E)$ is called d-regular if $\deg(v) = d$ for all $v \in V$.

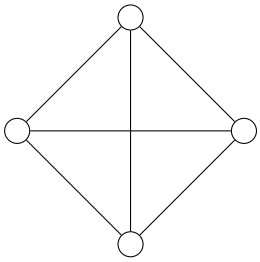
Example 7.7 (A 4-regular graph).
Can we construct a 4-regular graph on 7 vertices?

Yes. The intuition behind the construction is to start with a symmetric layout of 7 vertices, and try to keep the circular symmetry. The final graph is below:

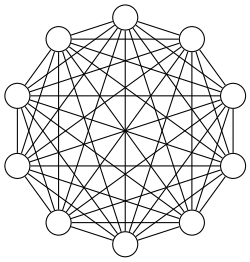


Definition 7.8 (Complete Graph on n Vertices).
The complete graph on n vertices, denoted K_n is the graph with n vertices and all $\binom{n}{2}$ edges. K_n is also often called a *clique* on n vertices.

For example, here are 2 complete graphs:



The complete graph K_4



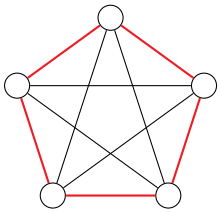
The complete graph K_{10}

Definition 7.9 (Walks, Paths and Cycles).
Let $G = (V, E)$ be a graph.

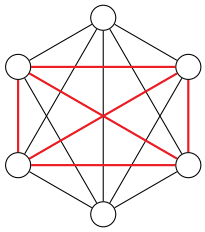
- A *walk* is a sequence of not-necessarily-distinct, pairwise-adjacent vertices
- A *path* is a sequence of distinct, pairwise-adjacent vertices
- A *cycle* is a path for which the first and last vertices are actually adjacent. If a cycle has k vertices, we say it is a k-cycle.
- Two vertices are *connected* if there is a path from one to the other.

Definition 7.10 (Graph Containment, Subgraph).
A graph $G = (V, E)$ contains a graph $G' = (V', E')$ if $V' \subseteq V$ and $E' \subseteq E$. If this is the case, we say that G' is a subgraph of G.

Here are some examples:



The complete graph K_5
contains a 5-cycle C_5



The complete graph K_6
contains a K_4

Note 7.11 (On Drawing Graphs).
Even though we are drawing the subgraphs on top of the original graph and thus not changing the shape of the subgraph, it is totally fine to draw subgraphs with a different shape to how it appears in the original.

How we draw a graph won't change the important properties of a graph.

Example 7.12 (Cycles in a K_n).
How many n-cycles are there in the complete graph on n vertices?

A good first guess might be $n!$, since we need to arrange n vertices. However, we're going to do a lot of overcounting. For example, these cycles are all the same:

$$(1,2,3,4,1) \qquad (2,3,4,1,2) \qquad (3,4,1,2,3) \qquad (4,1,2,3,4)$$

so we need to divide by n. However, we're not done yet, since the direction we write the cycle in doesn't matter. For example,

$$(1,2,3,4,1) \qquad \text{and} \qquad (1,4,3,2,1)$$

are the same cycle. So we actually divide by $2n$. In total, we get $n!/2n$ n-cycles. ☺

Definition 7.13 (Connected Graph, Connected Component).
A connected graph is a graph in which any two of its vertices are connected.

A connected component is a maximal connected subgraph. That is, a subgraph where adding any additional vertex breaks the property that it is connected.

For example, this graph is not connected. It has 3 connected components:

Adding two edges allows us to turn it into a connected graph.

7.2 Our First Big Graph Theorem

Theorem 7.14 (Conditions for a Cycle).
Every graph with n vertices and at least n edges contains a cycle (when $n \geq 3$). In addition, this bound is tight.

The idea is that if you start at a vertex, and keep "walking" around the graph avoiding previously visited vertices, you have to cross off a new vertex every step, so you can take at most $n-1$ steps. We'll make this argument formal below:

Proof. By induction on $n \geq 3$:

BC There is only one graph on 3 vertices with 3-edges: K_3. This graph clearly contains a cycle.

IS Assume the claim holds for $n-1$ and consider an arbitrary graph on n vertices with at least n edges. There are three cases:

C1 If the graph has a degree 0 vertex, remove it to get a graph on $n-1$ vertices with n edges. This graph contains a cycle by the IH. Adding the isolated vertex back doesn't disturb this cycle, so the original graph had a cycle as well.

c2 If the graph has a degree 1 vertex, remove it to get a graph on $n-1$ vertices with $n-1$ edges. This graph contains a cycle by the IH. Adding the removed vertex back doesn't disturb this cycle, so the original graph had a cycle as well.

c3 Now, we are in a case where the graph has no vertices of degree 0 or 1. We find a cycle as follows:

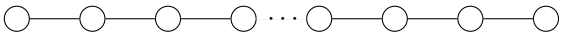
Start with an arbitrary vertex, and move to an adjacent one. Then, since every vertex has degree at least 2, we can exit that vertex through another edge and repeat.

Since the graph only has a finite number of vertices, eventually we will reach a vertex we previously visited, creating a cycle.

In addition, since we never backtrack, the cycle will have at least 3 vertices.

By induction, the claim holds for any n.

All we have left to do is to show that this bound is tight. To see this, consider the n-vertex path, denoted P_n:



This graph has $n-1$ edges, n vertices, and no cycles. □

Note 7.15 (Extremal Graph Theory).
This theorem is an example of a result from extremal graph theory, which is the branch of graph theory concerned with finding the biggest/smallest value for which a graph is forced to have a certain property.

Note 7.16 (Induction on Graphs).
Induction on graphs is an incredibly powerful tool, since so many graph problems end up reducing to a problem which looks suspiciously like the original one, but slightly smaller.

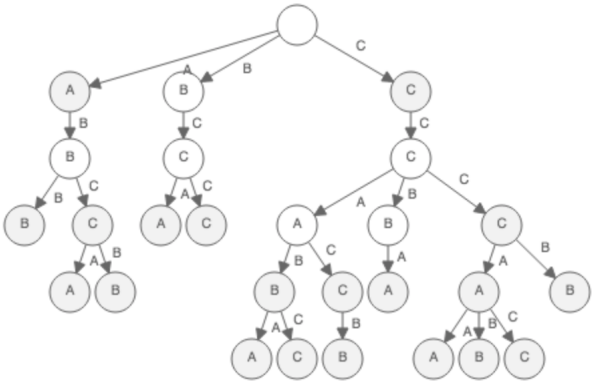
However, it is also very easy to get wrong. Here are the steps you should follow when doing graph induction (on vertices):

1. Prove the base case.
2. Assume the claim holds for $n-1$ vertices.
3. Consider an *arbitrary* graph on n vertices.
4. Remove a vertex to reduce the problem to one covered by the induction hypothesis.
5. Apply the induction hypothesis to argue for the existence of something.
6. Argue that the existence of your something is preserved when you add the removed vertex back into the graph.

Similar steps work for induction on edges/other features of graphs.

CHAPTER 8

Trees



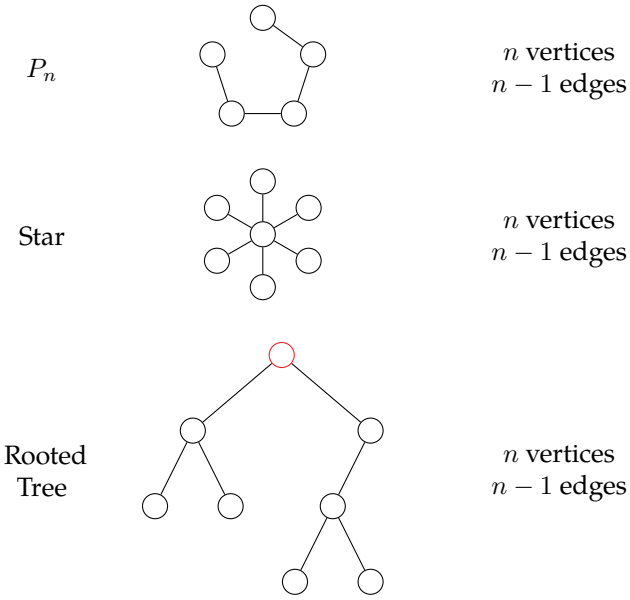
A picture like this is called a tree. If you want to know why the tree is growing upside down, ask the computer scientists who introduced this convention. (The conventional wisdom is that they never went out of the room, so they never saw a real tree.)

—László Lovász, József Pelikán, Katalin Vesztergombi

Contents

8.1 The Many Definitions of a Tree

In the last chapter, we were looking at graphs which had a lot of edges, but no cycles. In doing this, we found that if a graph had n vertices, the best we could do is to add $n-1$ edges. For example



In the last example, call the node in red the *root* of the tree. In fact all the above examples can be drawn as rooted trees (try and do so).

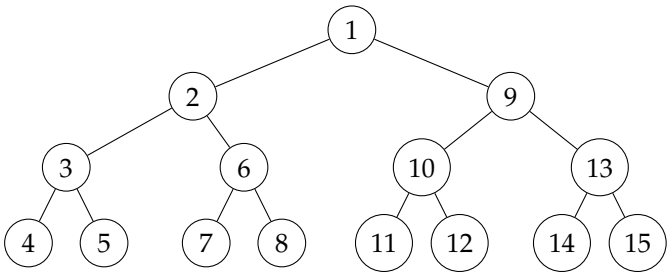
Can all high edge, but acyclic graphs be drawn as a "tree?" First, let's get a definition of a tree to work with:

Definition 8.1 (Tree [def 1]**).**
A tree is any structure which can be built recursively through the *tree growing procedure*, defined as:

- A single node is a tree.
- If $T = (V, E)$ is a tree, then $T' = (V \cup \{v_{\text{new}}\}, E \cup \{\{v_{\text{new}}, v\}\})$ is a tree for any $v \in V$ and $v_{\text{new}} \notin V$.

That is, a tree is anything that can be built up by repeatedly adding a new vertex and connecting it with exactly 1 edge to an existing vertex.

For example, this is a tree, which is called a *complete binary tree*



The numbers represent one possible order the vertices could have been added in. Can you think of others? There are a bunch.

However, often a recursive definition like [def 1] can be hard to work with... plus, it doesn't really tell us much about what a tree is! Where did our ideas of acyclicity and lots of edges go? Let's give a different definition of a tree more in line with this:

Definition 8.2 (Tree [def 2]).
A tree is a connected, acyclic graph.

Proof. It may be odd to see a proof of a definition, but in order to warrant calling this a definition, we have to show that it is consistent with our previous definition.

(Any tree satisfying [def 1] satisfies [def 2]):

We proceed by structural induction

BC Certainly a single node is a connected, acyclic graph.

IS Assume $T = (V, E)$ is a connected, acyclic graph which was built by the growing procedure. Let $T' = (V \cup \{v_{\text{new}}\}, E \cup \{v_{\text{new}}—v\})$ for some arbitrary $v \in V$.

We have to show that T' is still connected and acyclic:

CONNECTED Let $v_1, v_2 \in T'$ be arbitrary. If $v_1, v_1 \in T$, then they are connected by our inductive hypothesis. Otherwise, assume WLOG $v_1 \in T$ and $v_2 = v_{\text{new}}$.

Then $v_2—v_k$ is an edge for some $v_k \in T$ by the growing procedure, and v_k is connected to v_1 since T is connected and $v_1, v_k \in T$. Thus, we can concatenate the paths to get $v_2—v_k—v_1$, so $v_1—v_2$.

ACYCLIC We need to rule out that adding the new edge/vertex creates a cycle (this is the only way for there to be a cycle since T was acyclic by the IH). If it did, then v_{new} must be in a cycle, so $\deg(v_{\text{new}}) \geq 2$. But by the growing procedure, $\deg(v_{\text{new}}) = 1$, since it is connected to T by *exactly* one edge. Thus, adding v_{new} does not create a cycle.

So the claim holds by structural induction.

(Any tree satisfying [def 2] satisfies [def 1])

By induction on $|V|$

BC The only connected, acyclic graph with $|V| = 1$ is a single node, which is the base case of the growing procedure.

IS Assume any n vertex connected, acyclic graph can be grown, and let G be an arbitrary connected, acyclic graph on $n + 1$ vertices. We claim G contains a vertex with $\deg(v) = 1$.

Assume otherwise. Then, since G is connected, each vertex has degree ≥ 2. By the handshake lemma, we get

$$2|E| = \sum_{v \in V} \deg(V) \geq 2|V| \implies |E| \geq |V|$$

and so G must contain a cycle, a contradiction.

Now, since G contains a vertex of degree 1, remove it and it's incident edge to get G'. By the IH, G' can be grown. But then, adding the removed edge and vertex back is a valid growing operation, and so G can be grown.

Therefore, the claim holds by induction. □

Note 8.3 (Structural Induction).
In the above proof, we used a type of induction called structural induction, which is a technique for reasoning about recursively defined structures.

A proof by structural induction typically looks like:

- Base Case(s). Prove the property of interest for all base cases of your recursive definition.
- Inductive Step. Prove that application of the recursive rule(s) preserves the property of interest.

In fact, you've already been doing structural induction for your whole mathematical career. Regular induction on the naturals is just structural induction with the naturals defined by the rules:

$$0 \in \mathbb{N} \qquad \text{and} \qquad \text{if } n \in \mathbb{N}, \text{ then } n+1 \in N$$

so no stress. This isn't really anything different.

In proving those definitions are equivalent, we actually also proved another interesting result, and motivated another definition:

Definition 8.4 (Leaf).
A vertex of degree 1 in a tree is called a leaf.

Corollary 8.5 (Every Tree Has a Leaf).
Every tree $T = (V, E)$ with $|V| \geq 2$ contains a leaf.[a]

[a] We proved this as part of [def 2] $\implies$ [def 1]

You may be temped to think that we are done giving definitions of graphs. We aren't. There are actually *many* more ways we could define a tree; we'll cover one more here, which generalizes [def 2]. This will be our working definition of a tree going forward:

Definition 8.6 (Tree).
A tree is a graph $T = (V, E)$ which satisfies any two of the following properties:

(i) connected

(ii) acyclic

(iii) $|E| = |V| - 1$

In addition, if a graph has any two of these properties, it also has the third.

Buckle up... we've got a lot to prove here!

Proof. We know that any connected, acyclic graph is a tree, so it suffices to show that:

1. If $G = (V, E)$ is connected, then G acyclic $\iff |E| = |V| - 1$; and
2. If $G = (V, E)$ is acyclic, then G is connected $\iff |E| = |V| - 1$.

Before reading on, take a minute to convince yourself that this is enough to prove the claims in definition 8.6.

— (CLAIM 1) —

We'll first prove that $|E| \geq |V| - 1$. Remove all the edges of G; the resulting graph has $|V|$ connected components. Now, imagine adding the edges back one-by-one. Each time we do this, either:

(i) We connect two different connected components by putting an edge between two vertices that are not already connected (which does not create a cycle); or

(ii) We put an edge between two vertices that are already connected, creating a cycle

Now, note that (i) causes the number of connected components to go down by 1, while (ii) keeps it the same. Since we started with $|V|$ connected components, and ended with 1, we must have done (i) at least $|V| - 1$ times, so there are at least that many edges. Now, to prove the claim:

($\Rightarrow$) Assume G is connected and acyclic. From the above, we know $|E| \geq |V| - 1$. But we've previously shown that any acyclic graph satisfies $|E| \leq |V| - 1$. Thus, $|E| = |V| - 1$, as desired.

($\Leftarrow$) Assume G is connected and $|E| = |V| - 1$. Then, in the above procedure, to build G, we must have *only* done step (i), and thus could never have created a cycle. Thus, G is acyclic.

— (CLAIM 2) —

We'll prove each direction separately:

($\Rightarrow$) Assume $G = (V, E)$ is acyclic and connected. Then we're already done, since by claim 1 G also satisfies $|E| = |V| - 1$.

($\Leftarrow$) Assume $G = (V, E)$ is acyclic and satisfies $|E| = |V| - 1$. Then, in the above building procedure, we must have only done step (i), since doing step (ii) would create a cycle. Since $|E| = |V| - 1$, we do this exactly $|V| - 1$ times, so the number of connected components at the end of the procedure is:

$$|V| - (|V| - 1) = 1 \text{ connected component}$$

and so G is connected.

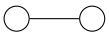
Therefore, this is an equivalent definition of a tree, and any having any two of the listed properties immediately implies we have the third. □

8.2 Encodings

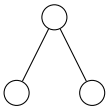
Example 8.7 (Counting Trees).
How many trees are there on n vertices?

I don't know about you, but I don't see an obvious way to approach this problem, so let's try some small examples. When $n = 1$, there's only 1 tree:

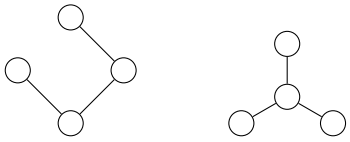
When $n = 2$, there just 1 tree:



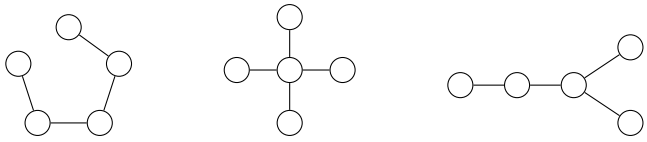
When $n = 3$, there is still just 1 tree:



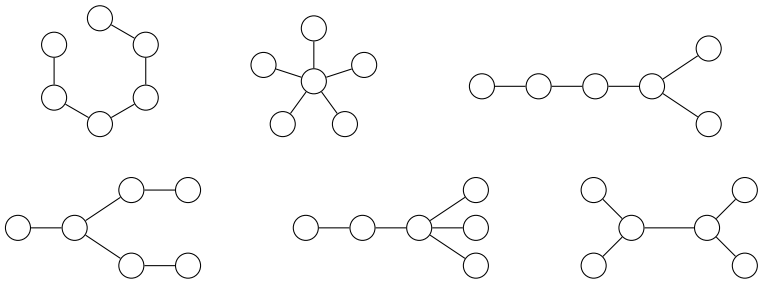
so the pattern is obvious, right? There's just 1 tree! Well, no, obviously that doesn't hold up going forward. When $n = 4$, there are 2 trees:



When $n = 5$ there are 3 trees:



And finally, when $n = 6$, there are 6 trees:



Great, so our pattern is $1, 1, 1, 2, 3, 6, \cdots$, which is, well, I still don't know. Plus, this is starting to get painful and there's no obvious recurrence or anything which would let us continue upwards with any ease.

For example, try and find all 11 trees with 7 vertices. If somehow you get through that, try and find all 23 with 8. It's a painful exercise.

However, we're not alone in not knowing where to go from here. In fact, no one does. The number of trees on n vertices is still not known. So our approach to this problem will be to call the answer T_n and just move on.

Note 8.8 (Bounds on T_n).
Although an exact formula for T_n is not known, it isn't *too* hard to show that

$$\frac{n^{n-2}}{n!} \leq T_n \leq 4^{n-1}$$

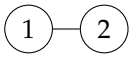
This is left as an exercise.

This problem may be very difficult, but there is a different, similar problem that we will be able to tackle:

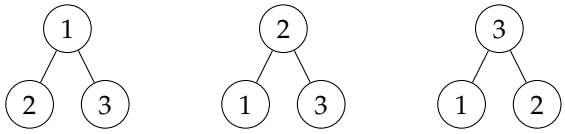
Example 8.9 (Number of Labeled Trees).
How many *labeled* trees are there on n vertices?

Let's run through our small examples again. When $n = 1$, there is still only 1 labeled tree:

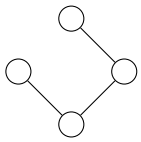
and when $n = 2$, there is still only 1 labeled tree. This is because we still allow graphs to be rotated, mirrored, etc.



But now, when $n = 3$, there are 3 labeled trees, in contrast with the 1 unlabeled tree we had before:

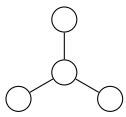


For $n = 4$, it will be painful to list all the labeled trees out (there will end up being 16 of them), but we can use a different trick. Instead, we'll count how many labeled trees there are for each kind of unlabeled tree:



There are $4!$ ways to assign labels $1, 2, 3, 4$ to the vertices in this graph. Of these, we double count by a factor of 2, since we can mirror the graph and not change it.

TOTAL: $\frac{4!}{2} = 12$ trees

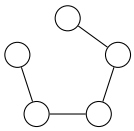


There are only 4 trees of this form. Once we pick the node in the middle, the placement of the others doesn't matter, since we can rotate the graph.

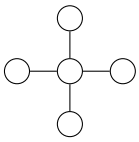
TOTAL: $\frac{4!}{3!} = 4$ trees

So in total for $n = 4$, we get 16 labeled trees.

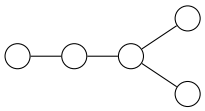
For the rest, we can use a similar logic. I'll simply provide the numbers of each kind of graph here. It's a good exercise to work through the counting arguments. When $n = 5$ there are 125 labeled trees:



$\frac{5!}{2!} = 60$ trees

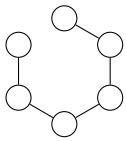


$\frac{5!}{4!} = 5$ trees

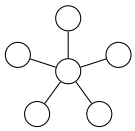


$\frac{5!}{2!} = 60$ trees

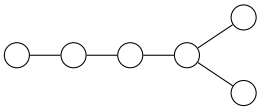
And finally, when $n = 6$, there are 1296 labeled trees:



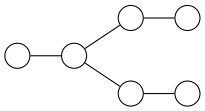
$\frac{6!}{2!} = 360$ trees



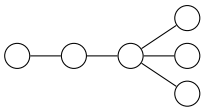
$\frac{6!}{5!} = 6$ trees



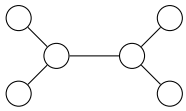
$\frac{6!}{2!} = 360$ trees



$\frac{6!}{2!} = 360$ trees



$\frac{6!}{2!} = 360$ trees



$\frac{6!}{2!} = 360$ trees

Alright, now our pattern is $1, 1, 3, 16, 125, 1296, \cdots$ which we can write more suggestively as $1, 2^0, 3^1, 4^2, 5^3, 6^4, \cdots$. At this point, we make our guess:

Theorem 8.10 (Cayley's Theorem).
There are n^{n-2} labeled trees on n vertices.

Note 8.11 (When $n = 1$...).
You may notice that Cayley's formula actually works for when $n = 1$ as well, since $1^{-1} = 1$. This is purely a coincidence, but is cool.

Before we jump into the proof of this claim, let's take a moment to gain some intuition for how good a bound this actually is. Consider two more obvious ways to upper bound the number of labeled trees:

1. There are only $2^{\binom{n}{2}}$ labeled graphs, so certainly this is an upper bound for the number of labeled trees.

   For comparison, we have

   $$2^{\binom{n}{2}} \approx 2^{n^2} \qquad \text{and} \qquad n^{n-2} \approx \left(2^{\log n}\right)^{n-2} \approx 2^{n \log n}$$

   so Cayley's theorem is a real improvement.[1]

[1] $O(\log n)$ is *much, much* better than $O(n)$. For example, when $n = 1,000,000$, $\log n$ is just 6. So shaving one of the factors of n down to a $\log n$ is impressive. If you want more justification, try telling a computer scientist that $O(n^2)$ is "really about the same" as $O(n \log n)$.

2. Since n-vertex trees have only $n-1$ edges, we can get the bound:

$$\#\text{ labeled trees} \leq \binom{\binom{n}{2}}{n-1} \leq \binom{n^2}{n-1} \leq \left(n^2\right)^{n-1} = n^{2n-2}$$

   This is actually pretty close!

The last approach above is actually quite good. To motivate the proof of Cayley's Theorem, we'll present that idea in a slightly different form.

Definition 8.12 (Encoding).
An encoding of a set of objects X is an injective function

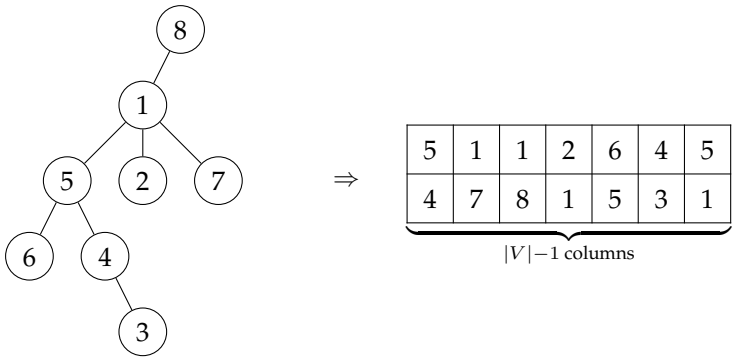
$$\text{Enc} : X \hookrightarrow \langle \text{set of encodings} \rangle$$

where our set of encodings is typically a simplified representation of an object, expressible with only a few symbols.

Important 8.13 (Using Encodings to get an Upper Bound).
Since an encoding is by definition an injective function, we get as a consequence that $|X| \leq |\langle \text{set of encodings} \rangle|$. If $\langle \text{set of encodings} \rangle$ is an easy set to count, then we can use this to establish an upper bound on $|X|$.

Further, if we can prove that our encoding is actually a *bijection*, then we can use this technique to determine exactly the size of X.

Turning back to counting trees, let's build an encoding for labeled trees, and use it to get an upper bound. Given a tree, we'll encode it by simply listing its edges in a table:



Certainly this mapping is injective (different trees will never end up with the same table, since they would have at least 1 different edge). Therefore, we get that

$$\#\text{ of } n\text{-vertex trees} \leq |\langle \text{set of encodings} \rangle| = \binom{n}{2}^{n-1} \approx n^{2n-2}$$

which is the result we got before with a counting argument.

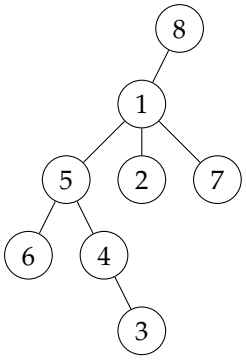
Note 8.14 (But this isn't a function...).
Yep, this mapping we defined above from trees to tables isn't actually a function. The same tree can have multiple tables associated with it, so it isn't well defined.

However, this isn't actually a problem for our counting arguments, since it only makes our bounds looser. However, if the function being ill-defined makes you uncomfortable, just think of a dictator arbitrarily defining which of the many possible outcomes is the "right" one for each possible tree.[a] All of our analysis will still stand either way.

[a] For this example, you could choose to, say, order the columns lexicographically.

This encoding is a good start, but it's wasteful, since so many tables represent the same tree. Can we do better?

One idea is to encode the tree as follows:



$\Rightarrow$

vertex	1	2	3	4	5	6	7
parent	8	1	4	5	1	5	1

$|V|-1$ columns

where we root the tree at the largest vertex, write the numbers $1, \cdots, n-1$ on the top row, and write the parent of the vertex below it.

But now, the top row is useless information, there's no need to actually write it since it is *always* $1, \cdots, n-1$, so we could have equivalently encoded the tree as:

```
8 1 4 5 1 5 1
```

if we assume the tree is rooted at 8 (one way to infer this is from the fact that `8` is the largest number present in the encoding... why is this enough?) So now:

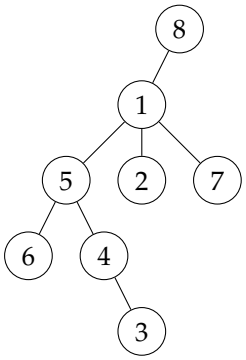
$$\#\text{ of } n\text{-vertex trees} \leq |\langle \text{set of encodings} \rangle| \leq n^{n-1}$$

which is *really* close to what we wanted. We could continue to refine this bound by, say, disallowing encodings which contain self-loops and cycles in our set of valid encodings, but that defeats the purpose of an encoding: to be an easier set to count. This leads us to our final encoding: the Prüfer code.

8.3 The Prüfer Code

Definition 8.15 (Extended Prüfer Code).
The Prüfer code of a labeled tree is the sequence of numbers generated by continually deleting the *smallest* leaf from the tree, and logging its parent.

For example, our tree is Extended Prüfer encoded as follows:



$\Rightarrow$

smallest leaf	2	3	4	6	5	7	1
parent	1	4	5	5	1	1	8

$|V|-1$ columns

At first you may wonder how this is possibly better than our previous encoding. After all, we still have just as many numbers, but now we've jumbled up the top row, so we can't even remove it anymore!

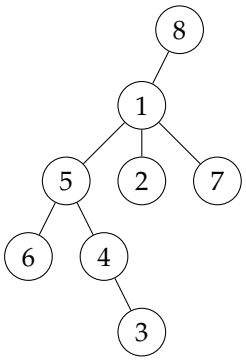
Indeed, that is why we call it the *Extended* Prüfer code. It turns out that we can not only remove the entire top row, but also the rightmost number in the bottom row!

Note 8.16 (Well-Definedness).
We've previously shown that every tree has at least 1 leaf, and that the graph that results from removing a leaf from a tree is still a tree.

These facts together allow this process to be well-defined.

Definition 8.17 (Prüfer Code).
The Prüfer code of a labeled tree is the first $n-2$ numbers in the bottom row of the Extended Prüfer code of the tree.

For example, the Prüfer code of our tree is:



$\Rightarrow$ `1 4 5 5 1 1`

Theorem 8.18 (Bijection between Prüfer Codes and Trees).
There is a bijective correspondence between Prüfer codes and labeled trees.

Proof. Consider the function

$$f : \langle \text{set of labeled trees} \rangle \to \langle \text{set of Prüfer codes} \rangle$$

defined by the following process:

1. From a labeled tree, construct its Extended Prüfer code. This is well-defined by the argument in note 8.16.
2. Chop off the top row and last column of this Extended Prüfer code.

We claim that this function is a bijection. From here, the proof will proceed in three parts. We will first introduce a function g, which we claim is the inverse. Then, we will show that it is both a left and right inverse to f. This will allow us to conclude that f is a bijection, which is what we wanted

1. Introducing the inverse function g

We present another process, which we will call g, where

$$g : \langle \text{set of Prüfer codes} \rangle \to \langle \text{set of labeled trees} \rangle$$

Define g as follows:

1. Build a table and fill the first $n-2$ columns of the table with the given Prüfer code. Fill the last column with n, where $n = (\#\text{ numbers in Prüfer code}) + 2$:

#	$\cdots$	#	n

2. Working left to right, fill the ? with the smallest number not in the shaded regions:

			?				
#	$\cdots$	#	#	#	$\cdots$	#	n

3. Construct the labeled tree by adding all the edges represented by the columns in the Extended Prüfer code.

To call g a function, we have to show that it is well-defined. Consider an arbitrary Prüfer code $p_1 p_2 \cdots p_k$. We have to show that (1) we never get stuck when executing g and (2) that the result of g is a valid labeled tree.

(1) The only real place we could get stuck is in step 2 of the process. Consider an arbitrary execution of step 2:

			?				
#	$\cdots$	#	#	#	$\cdots$	#	n

$n-1$ columns

The number that goes in the ? can be any of $1, 2, 3, \cdots, n-1$. It will never be n, since n can never be the *smallest* leaf. Thus, there are $n-1$ options for ?.

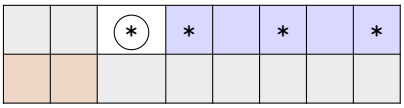
However, there are at most $n-2$ numbers in the shaded region, so we always have a valid number to put in the ? spot.

(2) We will show that $|E| = n - 1$ and acyclicity.

- ($|E| = n - 1$) Every column in the table corresponds to exactly one edge in the tree. Since there are $n - 1$ columns, there must also be $n - 1$ edges.
- (Acyclic) By part (1), we know that g generates a complete table where every cell has a number from 1 to n in it.

  First, note that no number is repeated in the top row, since when adding a new number to the top row, we disallow all previously added numbers (because of the shaded box to the left of the ?). Since there are $n-1$ columns in the table and only the numbers $1, \cdots, n-1$ can be added to the top row (n will never be the smallest leaf, and so will never be in the top row), we know that the numbers $1, \cdots, n-1$ each appear in the top row.

  Now, BWOC, assume there exists a cycle, and mark the vertices of the cycle with stars in the top row of finished table:



  We know that we can find at least one of the vertices in the cycle in the top row since from above, the numbers $1, \cdots, n-1$ each appear in the top row.

  Now consider the leftmost star (circled in the table):

  - First, note it cannot appear in any of the gray regions by construction.
  - If it appears in the red region, then there are two cases. First, if the square above it is also in the cycle, then we contradict the fact that (*) is the leftmost star in the table. If the square above it is not in the cycle, then this edge doesn't contribute to the cycle, and the rest of our argument holds.
  - Finally, since the numbers in the top row are all distinct, it cannot appear in the blue region.

  Thus, (*) appears only once in the columns of the table representing edges in the cycle. However, it was part of a cycle, so there must be at least two edges involving that vertex, a contradiction. Thus, the output of g is acyclic.

Now, since the output of g has $n - 1$ edges and is acyclic, it is a tree.

Together, (1) and (2) combine to show that g is a well-defined function, which is the result we wanted to show.

2. The function g is a left-inverse

We have to show that $(g \circ f) = \mathrm{id}_{\langle \text{labeled trees} \rangle}$. To do this, it suffices to show that if we know a Prüfer code came from a valid labeled tree, g will recover the tree.

Assume we have a Prüfer code. First note that the last column is easy to fill in; it is always the number of columns plus 2. It therefore suffices to show that we can recover the top row.

#	···	#	?					
#	···	#	#	#	···	#	n	

First note that since the top row contains every number from 1 to $n - 1$ exactly once, the ? cannot be any of the previous numbers in the top row. In addition, since once a number appears in the top row, we remove it from the tree. Therefore, it cannot appear in any cell in the table to its left. Finally, since self-loops are disallowed, it cannot appear below the ?.

Thus, all the numbers in the shaded region are ruled out:

				?				
#	···	#	#	#	···	#	n	

Let k be the smallest number not in the shaded region. To show our process is correct, it suffices to show that k is the smallest leaf in the original tree.

- Certainly k hasn't been removed from the tree, since it isn't to the left of the ? in the top row.
- AFSOC that k is not a leaf at this point. Then some other vertex has k as its parent. However, we know this isn't the case, since k is never listed in the bottom row after this column by assumption.

Therefore, k is the smallest leaf, so we chose the right number to put there.

3. The function g is a right inverse

First note that every Extended Prüfer code determines a unique tree (it is just an edge list), and that every tree determines a unique Extended Prüfer code by note 8.16.

But then, removing and then re-adding the other parts of the table certainly doesn't change the Prüfer code, so we get $f \circ g = \mathrm{id}_{\langle \text{Prüfer Codes} \rangle}$.

Now, as discussed before, we get that f is a bijection, which is what we wanted. □

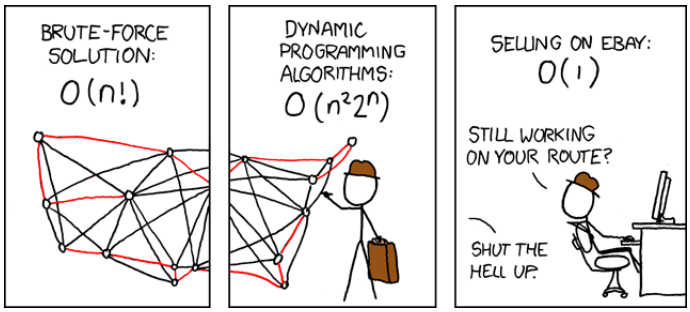
Note 8.19 (Take a Minute).
That was probably the most complicated proof we've seen in this class to date; it certainly isn't easy to follow the first time you see it. Take a minute and make sure all of it makes sense before reading on.

With all that out of the way, at long last, we can provide a (blessedly short) proof of Cayley's Theorem:

Proof (Thm 8.10). By Theorem 8.18, the number of labeled trees on n vertices is equal to the number of Prüfer codes with $n-2$ numbers. But this is just n^{n-2}. □

CHAPTER 9

The Traveling Salesman Problem



What's the complexity class of the best linear programming cutting-plane techniques? I couldn't find it anywhere. Man, the Garfield guy doesn't have these problems...

—Randall Munroe

Contents

9.1 Big O Notation

Often in graph theory, we are interested in the *asymptotic* behavior of certain phenomenon. That is, how certain properties of a graph change as the number of vertices/edges/other things change.

To do this, it is helpful to understand big-oh notation. Unfortunately, it is a law in at least 47 of the 50 states that we begin by giving the kinda hard to understand formal definition:

Definition 9.1 (Big-Oh).
We say that a function $f(n)$ is $O(g(n))$ (pronounced "oh of g") if there exists a constant c such that

$$|f(n)| \leq cg(n)$$

for all sufficiently large n.

With that out of the way, we'll pretty much never use the formal definition outside of this section. Rather, we'll focus more on an intuitive understanding of what it means. The "definition" I like to work with is:

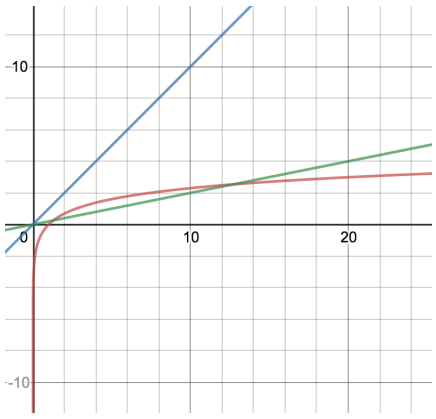
> *"We say $f(n)$ is $O(g(n))$ if f grows at most as fast as g, when we ignore any constants and lower order terms in the functions."*

It helps to see some examples:

Example 9.2 (Classifying Functions by Big-Oh).
Show the following:

- $\log n \in O(n)$
- $n + 1000 \in O(n)$
- $0 \in O(n)$ and $0 \in O(1)$
- $3n^2 + 10n - 400 \in O(n^2)$

It helps to see a plot when starting out:



Here we see that $f(n) = n$ (in blue) grows much faster than $f(n) = \log n$ (in red), so intuitively we should have $\log n \in O(n)$.

To see this from the definition, note that when $c = 1$:

$$|\log n| \leq n$$

for all $n > 1$.

It's also worth noting that our choice of c didn't really matter here, even $c = \frac{1}{5}$ (in green) eventually surpasses $\log n$.

It's fairly clear that $0 \in O(n)$—it definitely grows slower than $f(n) = n$. To see that $0 \in O(1)$, note that

$$|0| \leq c \cdot 1$$

for pretty much any constant c (except $c = 0$).

In both of the above examples, our choice of constant hasn't really mattered much. Does it ever? Well, yes. Otherwise it wouldn't be there. For an example of this, we'll prove $n + 1000 \in O(n)$. Using $c = 2$, we get

$$|n + 1000| \leq 2n$$

for all $n \geq 1000$, so indeed $n + 1000 \in O(n)$. Note that if we used a $c \leq 1$, we wouldn't have been able to prove this (why?).

Finally, to see an example of dropping lower order terms, consider $3n^2 + 10n - 400$. Here, we use $c = 4$ to get:

$$\left|3n^2 + 10n - 400\right| \leq 4n^2$$

since for large enough n, the n^2 terms will far outweigh the n terms, so we only need to beat $3n^2$. ☺

Example 9.3 (Using Your Gut).
Which of the following big-oh classes is $f(x) = 2^n$ a part of?

$$O(1), O(\log n), O(n), O(n^2), O(n^k), O(2^n), O(3^n), O(n!)$$

It's fairly clear that 2^n grows much faster than any polynomial, so the first 5:

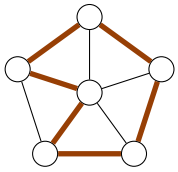
$$O(1), O(\log n), O(n), O(n^2), O(n^k) \Rightarrow \text{No.}$$

are all out. Definitely $2^n \in O(2^n)$. In addition, 3^n and $n!$ both seem to grow faster than 2^n, so $2^n \in O(3^n)$ and $2^n \in O(n!)$. ☺

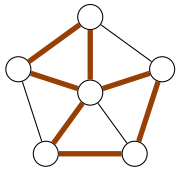
9.2 Hamiltonian Paths and Cycles

We'll start with a definition:

Definition 9.4 (Hamiltonian Cycle).
A Hamiltonian cycle is a cycle which contains every vertex (once).



This *is* a Hamiltonian cycle.

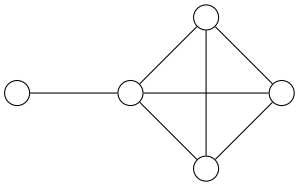


This is *not*.

We're going to try and explore the broad question of: what conditions will force a Hamiltonian cycle to exist in a graph?

Example 9.5 (Lots of Edges).
Will lots of edges force a Hamiltonian cycle to exist?

Certainly if every edge is present, then a Hamiltonian cycle exists. Sadly, we can't do much better than this. A K_{n-1} and an isolated vertex with 1 edge has a bunch of edges, but no Hamiltonian cycle. For example:



has no Hamiltonian cycle. But in general, it has:

$$\frac{\binom{n-1}{2}+1}{\binom{n}{2}} \text{ of the edges}$$

As $n \to \infty$, we get that

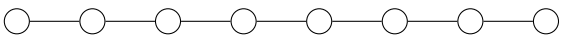
$$\frac{\binom{n-1}{2}+1}{\binom{n}{2}} = \frac{\frac{(n-1)(n-2)}{2}+1}{\frac{n(n-1)}{2}} = \frac{\frac{1}{2}n^2+O(n)}{\frac{1}{2}n^2+O(n)} \approx 1$$

so this isn't really much better than just requiring every edge to be present. Sadly, it seems that using edges to force a Hamiltonian cycle isn't very useful. However, there is another approach...

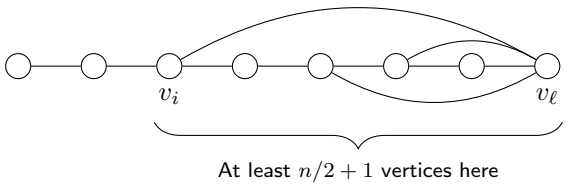
Note 9.6 (Abuse of Notation).
Note that we replaced a bunch of terms which collectively were $O(n)$ by just $O(n)$. This makes absolutely no sense mathematically—$O(n)$ is actually a set—but is so common we'll do it here too.

Theorem 9.7 (Dirac's Theorem).
Every graph with minimum degree $\geq \frac{n}{2}$ has a Hamiltonian cycle (for $n \geq 3$).

Before we prove this, let's start with something easier: let's try and find a cycle of length $\geq \frac{n}{2}+1$. To do this, consider the longest path in the graph:

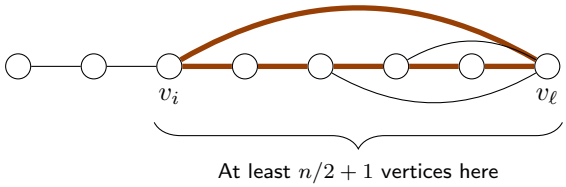


The last point in this graph must have all of it's neighbors in the graph, otherwise we could add one to make a longer path.



As the picture indicates, there are at least $\frac{n}{2} + 1$ vertices in between (and including) the leftmost neighbor of the last vertex and the last vertex itself. This is because the last vertex in the path has at least $\frac{n}{2}$ neighbors all of which are in the path.

But then, we get that



is a cycle with $\frac{n}{2} + 1$ vertices, which is what we wanted. We'll end up using a similar idea to prove the full version of Dirac's Theorem.

Proof (Dirac's Theorem). We'll do the proof in three parts:

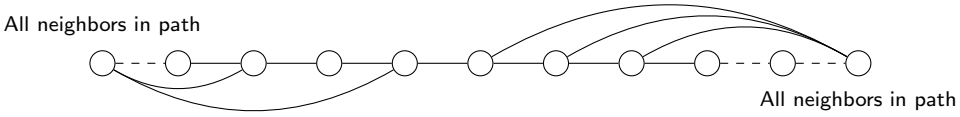
1. Prove that the graph is connected.
2. Prove that if there is a path with k vertices, then there is a cycle of length $\geq k$.
3. Using (1) and (2), prove the full theorem.

We begin by showing the graph is connected. Assume towards a contradiction that it isn't. Then there are ≥ 2 connected components. But since the minimum degree is $\geq \frac{n}{2}$, each must have at least $\frac{n}{2} + 1$ vertices. Thus, the graph has

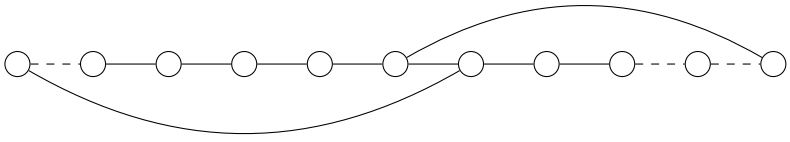
$$\left(\frac{n}{2} + 1\right) + \left(\frac{n}{2} + 1\right) = n + 2 \text{ vertices}$$

which is a contradiction. Therefore, the graph is connected.

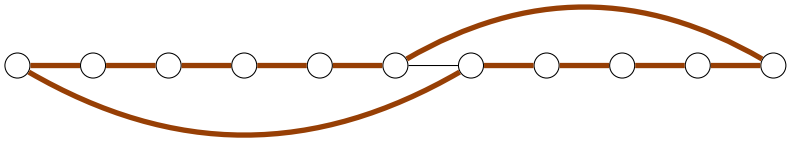
Now, assume we have a path with k vertices in it. From each edge, we'll extend it as far as we can—like we did in the warm up—until we get a maximal path, where each end has all its neighbors in the path:



Now, let's think about how we could end up with a k-vertex cycle. One way is if we end up with a setup like the following:

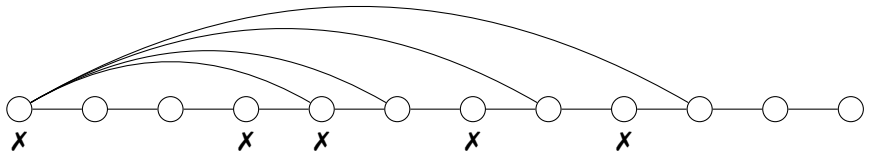


Since this yields the cycle:



and since there were at least k vertices in our path, then there must be at least k in our new cycle. We now turn our attention to proving that—given the conditions we have—this setup must exist in our graph.

Consider the edges coming from the leftmost vertex, and ✗ out the vertices which, if an edge from the rightmost vertex were to end there, would create a cycle with k vertices:



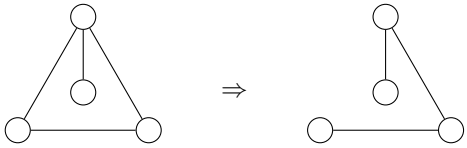
Notice that each edge coming from the right edge creates exactly 1 ✗, and since every vertex in our graph has degree $\geq \frac{n}{2}$, we will draw at least $\geq \frac{n}{2}$ ✗'s.

Now, consider again the edges from the rightmost vertex. There are only n vertices in the graph, so there are at most $n-1$ options for where these edges can end. But we've ✗'ed out at least $\frac{n}{2}$ of them, so there are only

$$(n-1) - \frac{n}{2} < \frac{n}{2}$$

of the vertices left. But we have $\geq \frac{n}{2}$ edges, so one must hit an ✗, and therefore we must get our cycle!

We can now wrap up the proof. There's definitely a path with 3 vertices in our graph: just take any vertex and 2 of its neighbors (this is ok since $n \geq 3$). Now we can use the above to get a cycle of length 3. But our graph is connected, so some vertex in this cycle connects to another vertex outside the cycle:



As in the picture above, we can just cut an edge to get a path with 4 vertices. Repeat this process until we can't, i.e.:

$$\text{path with 4 vertices} \implies \text{cycle with 4 vertices} \implies \text{path with 5 vertices} \implies \cdots$$

Eventually this chain of implications must end, since we only have a finite number of vertices. When it does, we'll have a cycle going through all n vertices: a Hamiltonian cycle. This is what we wanted. □

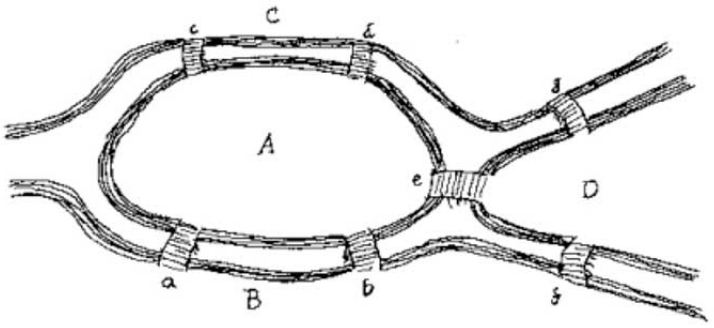
9.3 Eulerian Circuits

Definition 9.8 (Eulerian Circuit).
A Eulerian Circuit is a walk which contains every edge exactly once and which ends at the starting vertex.

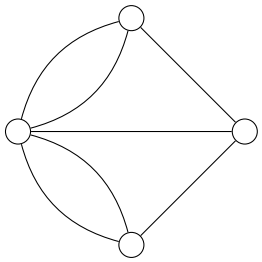
The idea of a Eulerian Circuit actually has historical roots. It—unsuprisingly—was first discussed by Euler in the context of touring the Bridges of Königsberg, which we'll use as our motivating example for this topic:

Example 9.9 (The Bridges of Königsberg).
Is it possible to tour all the bridges of Königsberg, passing over each exactly once, and return to where you started?

The bridges are laid out as follows:



The idea is to represent this as a *multigraph*, which is just a graph in which we allow there to be multiple edges. We'll represent each "chunk" of land as a vertex, and each of the bridges as an edge. Doing this yields the graph:



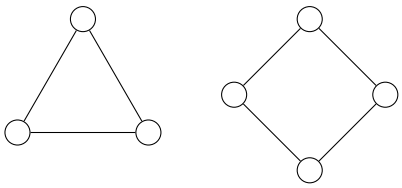
Briefly convince yourself that finding a Eulerian circuit in this graph exactly models the original problem.

One observation from here is that as you pass each vertex, you must go "in" then immediately "out" of the vertex. Thus, each pass through a vertex decreases the degree by two, and since at the end we must have used all the edges up, the degree of every vertex must be even.

There's a potential catch in the vertex we start with, since starting there doesn't relate to "passing through" it, but it turns out that our argument is still fine, since we have to start and end in the same place. Therefore, the first out is matched by the last in and so we're good. Summarizing, we get:

$$\text{Eulerian circuit} \implies \text{all degrees are even}$$

Can we go the other way? Sadly, no. Consider:



There's no way to even get from one part of the graph here to the other, nonetheless find a Eulerian cycle. However, your intuition might tell you that we kind of "cheated" by making the graph disconnected. What if we also require connectivity? Is that enough? Amazingly, it actually is! Let's prove it.

Theorem 9.10 (Necessary and Sufficient Conditions for a Eulerian Circuit). A multigraph G contains a Eulerian circuit if and only if all of its degrees are even and it is connected.[a]

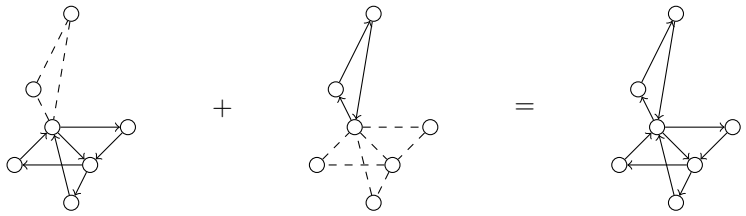
[a] Technically, degree 0 vertices are allowed, but are uninteresting.

Proof. We've actually already argued the forwards direction. For the backwards direction, we'll present an algorithm for finding a Eulerian circuit in a connected graph with all degrees even:

> $G = (V, E)$ is a graph
>
> FIND-EULER-CIRCUIT(G):
>
> 1 Discard all vertices of degree 0.
> 2 Start at any vertex and walk until you can't.
> 3 If not done, find an unused edge connected to our current circuit and recurse. Then, join the two resulting circuits.

Actually, this is almost a complete proof; we only have some small details to iron out:

- In step 2, we actually start and end at the same vertex. This is because we must end at a vertex with an odd degree "remaining," but the process of walking around the graph combined with the all degrees even condition only leaves even degree vertices. Thus, the only option for an endpoint is the original vertex.
- In step 3, while we aren't done, we can always find an edge connected to the current circuit because the graph is connected.
- Finally, to clarify what we mean by "join the resulting circuits," here's an example of what that looks like:



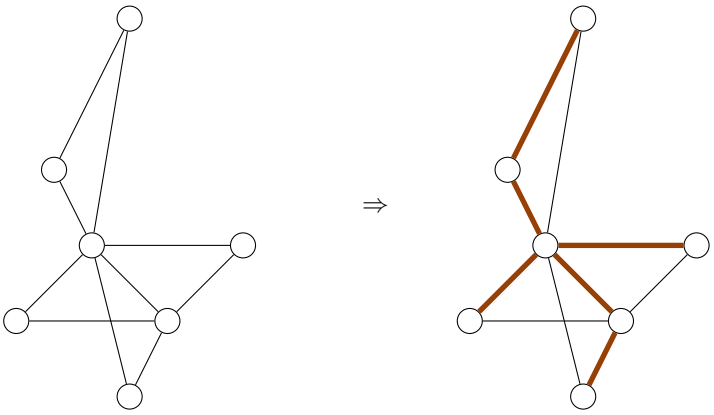
With all that, we're done. □

Note 9.11 (Multigraphs). Note that never did we actually need that there is a unique edge connecting two vertices, so this proof is totally valid for multigraphs as well.

9.4 Minimum Spanning Trees

Definition 9.12 (Spanning Tree).
A spanning tree of a graph is a subtree of the graph which contains every vertex.

For example (using the graph from our proof of the Eulerian circuit conditions), we can find the following spanning tree:



Proposition 9.13 (Existence of Spanning Trees).
Every connected graph has a spanning tree.

Proof. We'll present a surprisingly simple algorithm:

> $G = (V, E)$ is a graph
>
> FIND-MST(G):
>
> 1 While there is a cycle, remove an edge from it.

Certainly each step of this algorithm doesn't hurt connectivity, since we're removing an edge from a cycle. Any path that would have used that edge can go the long way around with the other edges in the cycle.

However, at the end, we end up with an acyclic graph. Thus, the resulting graph is a connected acyclic graph; i.e. a tree. □

This problem on its own isn't particularly interesting. But what if we add weights (to keep it simple, in $\mathbb{R}^+$) to each of the edges? That is, what if we make some more "expensive" to add to our tree?

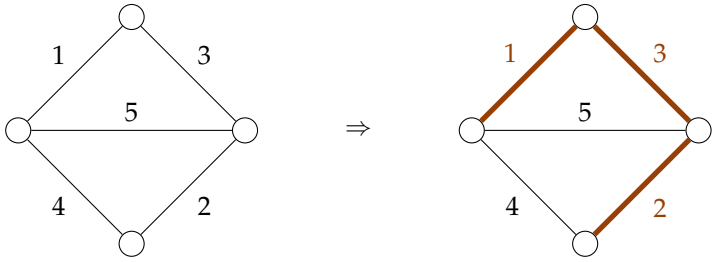
Is there a good way to find the cheapest spanning tree in this new system?

Note 9.14 (Interpreting Weights).
In applications, these weights are typically thought of as distances (such as in road networks) or as costs/expenses (such as the amount of fuel needed for a flight path).

This whole section is exceptionally useful in the real world.

Definition 9.15 (Minimum Spanning Tree).
The minimum spanning tree (or MST) of a graph is the spanning tree which minimizes the sum of the weights of the edges.

For example, here is a weighted graph and its MST:



Note 9.16 (MST's in K_n).
Sometimes the task of finding an MST is presented in the context of the complete graph, since often in the real world we are allowed to travel between any two points we wish, but perhaps at great cost.

These two presentations are equivalent actually. Just replace missing edges by edges of weight ∞ to get a complete graph if necessary.

We'll now examine some algorithms for finding MST's.

Prim's Algorithm

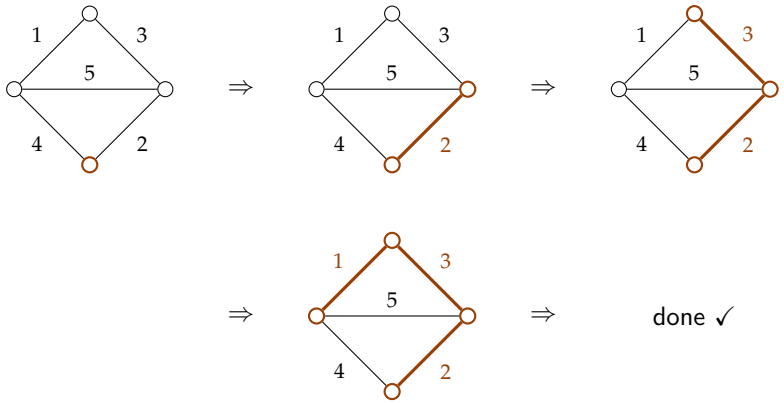
The algorithm is as follows:

$G = (V, E, c)$ is a graph with a cost function $c : E \to \mathbb{R}^+$

PRIM-MST(G):

1. Start at any vertex v. It is your "starting" connected component.
2. Keep adding the smallest edge extending from your current connected component to a new vertex.
3. Stop after we've added $n - 1$ edges.

Here's how Prim's algorithm would find the MST from the above graph:

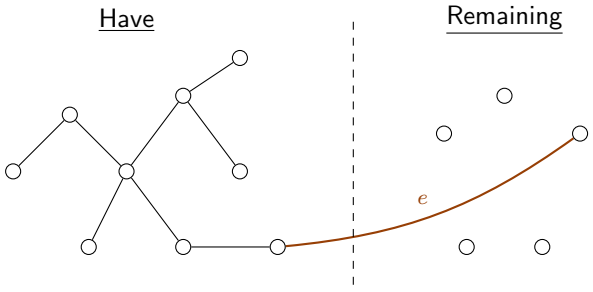


Proof. We're actually mimicking the tree growing procedure, so we're definitely going to end up with a tree at the end.

To show that we get the *minimum* spanning tree, we'll show that at each point in the algorithm, we are on a "right track." That is, no matter what we've built so far, there exists an MST which contains what we have. This better be true, since Prim's algorithm never backtracks. To do this, we'll use induction:

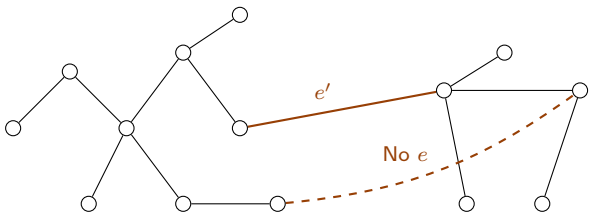
BC Any MST contains every vertex, so whatever vertex we choose to start at must be in an MST.

IS Assume that the current tree we have can be extended to an MST. Our setup will look something like the following:



Where the IH guarantees there exists an MST containing the "have," and the algorithm chose to add e at this stage.

Assume towards a contradiction that no MST contains this edge. Now, consider the MST guaranteed by the induction hypothesis:



Note that this MST does not contain e, but contains some other edge e' which crosses from our "have" to our "remaining." By the assumption of Prim's algorithm, we have

$$\text{cost}(e) \leq \text{cost}(e')$$

But then, removing e' and adding e also makes a tree, and from that inequality, it must have $\leq$ cost, so it is also a minimum spanning tree. But then, some MST contains e, a contradiction.

By induction, Prim's algorithm will terminate with an MST. □

Note 9.17 (MST Cut Property).
In doing this proof, we ended up proving something called the MST cut property. This states that:

> Between any set $S \subseteq V$ and $V \backslash S$, the least cost edge is in a minimum spanning tree.

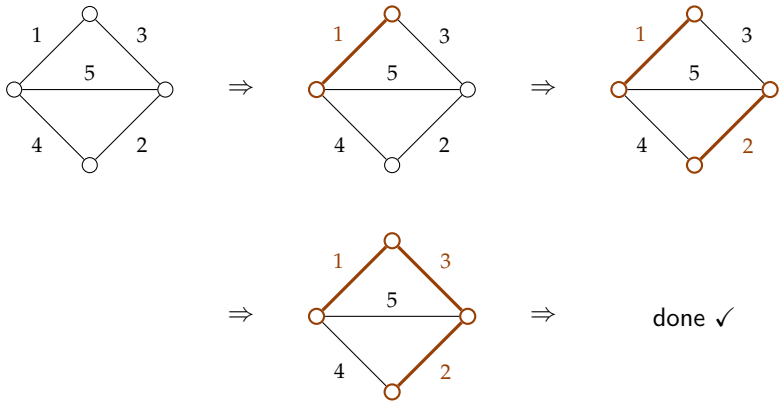
Take a second and make sure you see how this fact can be derived with almost identical logic to that found in the proof above.

Kruskal's Algorithm

The algorithm is as follows:

> $G = (V, E, c)$ is a graph with a cost function $c : E \to \mathbb{R}^+$
>
> KRUSKAL-MST(G):
>
> 1 Keep adding the lowest cost edge (of any edge in the graph) which doesn't create a cycle.
>
> 2 Stop after we've added $n - 1$ edges.

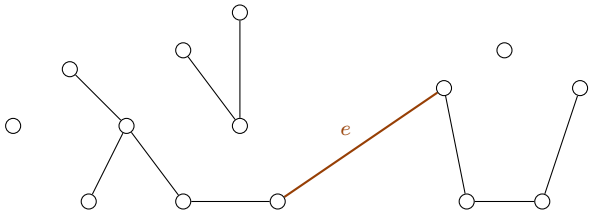
Here's how Kruskal's algorithm would find the MST from the above graph:



Proof. Again, it's fairly clear that this actually builds a tree. We'll use the same induction approach to show that this yields an MST:

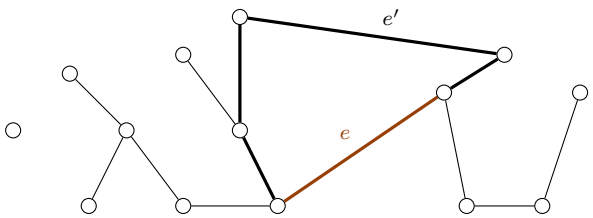
BC We start with no edges, and since $\varnothing \subseteq$ anything, the claim holds.

IS Assume that the current edges we have can be extended to an MST. Our setup will look something like the following:



Where the IH guarantees there exists an MST containing these edges and the algorithm chose to add e at this stage.

Assume towards a contradiction that no MST contains e, and consider the MST guaranteed by the IH. Add the edge e to this MST:



Doing so must necessarily create a cycle (in bold). But then, there must exist an edge e' with $\text{cost}(e) \leq \text{cost}(e')$ in the cycle, otherwise we wouldn't have added e at this point in the algorithm.

This is because the graph ends up connected, so there must be some edge[1] which connects the two connected components connected by e which wasn't in our graph at this stage in the algorithm. None of these would have created a cycle, so they must have $\geq$ cost since our algorithm didn't choose them.

But then, removing e' and adding e gives a spanning tree with $\leq$ cost, hence there is an MST with e after all, a contradiction.

By induction, Kruskal's algorithm will terminate with an MST. □

9.5 The Traveling Salesman Problem

Definition 9.18 (The Traveling Salesman Problem, Metric TSP).
The traveling salesman problem (or TSP), is the problem of finding a route through every vertex in a graph which returns to the starting point and has the minimum possible cost.

The metric TSP problem is a variant where we are restricted to K_n, and the distances between vertices satisfy the triangle inequality:

$$c(v_1, v_2) + c(v_2, v_3) \leq c(v_1, v_3)$$

Here, our goal is to find a minimum cost Hamiltonian cycle.

Note 9.19 (A Million Dollar Question).
This is literally a million dollar question. If you find an "efficient" algorithm to solve this, the Clay Institute will literally give you $1,000,000. See

`http://www.claymath.org/millennium-problems/p-vs-np-problem`

As you may expect from this, there is known solution.

Note 9.20 (Metric TSP).
The conditions of the metric variant of the TSP may be hard to wrap your head around at first. They are designed to *exactly* model the application where the vertices are cities and the costs are the distances between the cities. It is often useful to think of it in terms of this.

Just because getting an exact solution to this problem is hard doesn't mean we will give up though. Instead, we'll show how to find an approximate solution. First we give a definition:

Definition 9.21 (Factor-n-Approximation Algorithm).
An algorithm is a factor-n-approximation to a problem if the cost of it's output is always within a factor of n to the optimal output.

[1] Or series of edges, at least one of which isn't in the current graph.

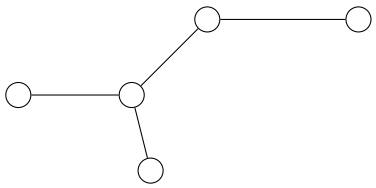
Theorem 9.22 (Metric TSP Approximation).
There is a factor-2-approximation algorithm to the metric TSP problem.

Proof. The algorithm is as follows:

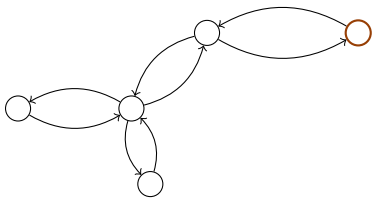
> $G = (V, E, c)$ is a complete graph with cost function c that obeys the triangle inequality
>
> METRIC-TSP-APPROX(G):
>
> 1 Using Prim's or Kruskal's algorithm, find a minimum spanning tree.
> 2 Walk around the tree, taking every edge twice.
> 3 If we would have to return to a previously visited vertex, jump directly to the next one on our path.

For visual clarity, we won't draw all the edges in our K_n, and assume the cost of an edge is the Euclidean distance between them. The proof, however, works for arbitrary metric spaces.[2]

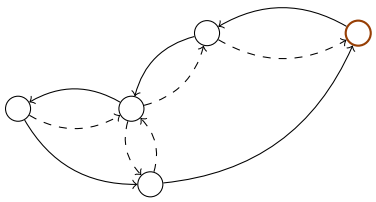
For an example, consider the following complete graph, with the following MST:



We'd first walk around the tree as follows (we'll assume we start at the rightmost vertex, in red):



But notice this is not a Hamiltonian cycle, since we visit some vertices multiple times. We fix this with step (3), jumping immediately to the next vertex:



From this example, it's not hard to see that the algorithm actually will generate a Hamiltonian cycle. Therefore, we turn to showing that it is a factor-2-approximation.

[2] Formally, a metric space is a pair (M, d) where M is a set of points, and $d : M \times M \to \mathbb{R}^+$ is a metric which is: (1) symmetric; (2) positive definite; and (3) obeys the triangle inequality.

First, consider the cost of the walk generated by step (2), before doing step (3). This walk has cost:

$$\text{cost}(\text{walk}) = 2 \times \text{cost}(\text{MST})$$

since we take every edge in the MST exactly twice. But note that we can't possibly hope to do better than the MST, since finding a TSP tour requires connecting the graph and the MST is the cheapest way to connect the graph. Thus $\text{cost}(\text{MST}) \leq \text{cost}(\text{OPT})$, which gives us that

$$\text{cost}(\text{walk}) = 2 \times \text{cost}(\text{MST}) \leq 2 \times \text{cost}(\text{OPT})$$

and so the walk from step (2) is already a factor-2-approximation. However, we still have to show that step (3) doesn't screw this up. Luckily, this is easy: the triangle inequality guarantees that jumping straight to the next vertex is better than taking two or more edges to get there, so (3) actually will reduce the cost, if anything.

Thus, we've found a factor-2-approximation to the metric TSP problem. □

CHAPTER 10

Ramsey Theory

Imagine an alien force, vastly more powerful than us, landing on Earth and demanding the value of $R(5,5)$ or they will destroy our planet. In that case, we should marshal all our computers and our mathematicians and attempt to find the value. Suppose, instead, that they ask for $R(6,6)$. In that case, we should attempt to destroy the aliens.

—Paul Erdős

Contents

10.0 Probability Primer

In general, probability is an extremely difficult concept to define rigorously; however, this task becomes much easier when in a discrete setting. And luckily for us, this is a discrete math course. So let's do it!

Definition 10.1 (Probability Space, Event).
A probability space is a pair $(\Omega, \Pr)$, where

- Ω is a set[a] called the sample space of all possible outcomes of some process
- $\Pr : \mathcal{P}(\Omega) \to [0, 1]$ is a probability measure, which indicates how likely an outcome in the sample space is.

In addition, we require common sense checks, such as:

- $\Pr[\Omega] = 1$; that is, something always happens.
- If A, B disjoint, then $\Pr[A \cup B] = \Pr[A] + \Pr[B]$. This actually extends to countable unions as well, but we won't use this fact much.

Intuitively, a probability is just an assignment of how likely any subset of a larger set of outcomes is to happen.

[a] which for our purposes, we'll restrict to be countable

Note 10.2 (Probably can ignore all that...).
The above is a balance between a very rigorous definition and a common sense definition. But honestly, if you just run with your intuition about what probability is, you'll be able to to everything in this chapter just fine.

Definition 10.3 (Event).
We call a subset of Ω an event. Intuitively, an event is just a collection of things that can happen.

For our purposes, we'll mainly be focusing on coming up with clever way to upper bound probabilities of things happening. The first bound we'll look at is straightforward, but powerful:

Proposition 10.4 (Union Bound).
Let $A_1, A_2, \cdots, A_n, \cdots$ be a countable collections of events (subsets of Ω). Then

$$\Pr[A_1 \cup A_2 \cup \cdots \cup A_n \cup \cdots] \leq \Pr[A_1] + \Pr[A_2] + \cdots + \Pr[A_n] + \cdots$$

Notice that we don't require the A_i to be disjoint.

Proof. The idea is to use the inclusion exclusion principle to get it for two events, then use induction to extend it to n. It's a good exercise if you've got time.

It's a bit more of a pain for the countably infinite case. We won't do it here. □

Definition 10.5 (Random Variable).
A random variable is a function $X : \Omega \to \mathbb{R}$. Intuitively, a random variable is a way to assign real number to outcomes in the sample space.

Note, it is extremely common to use capital letters towards the end of the alphabet X, Y, Z to denote random variables.

Example 10.6 (Flipping Coins).
If we flip 5 fair coins, what's the probability we get more than 3 heads?

First, let's say Ω is the set of all possible sequences of 5 coin flips, eg.

$$\Omega = \{\text{HHHHH}, \text{HTHHT}, \cdots\}$$

Then a random variable might be number of heads flipped. We're often interested in the probability that a random variable takes on a given value.

For example, if we let X be a random variable denoting the number of heads flipped, we may be interested in $\Pr[X > 3]$. Well, what is this probability (assuming a fair coin)? For this to happen, one of two things must happen:

1. We flip 4 heads. There are 5 ways to to this, since once we pick the location of the tail, the heads are fixed.

   $$\{\text{HHHHT}, \text{HHHTH}, \text{HHTHH}, \text{HTHHH}, \text{THHHH}\}$$

   To get any one of these sequences, we must hit five 50/50's in a row, so our probability is $5 \cdot \left(\frac{1}{2}\right)^5 = \frac{5}{32}$.

2. We flip 5 heads. There's only 1 way to do this, and again, we must hit five 50/50's in a row. Thus, our probability of this is $\left(\frac{1}{2}\right)^5 = \frac{1}{32}$.

But only one of those two things can happen, so we can add the probabilities to get $\Pr[X > 3] = \frac{5}{32} + \frac{1}{32} \approx 0.1875$, so this happens about 18% of the time. ☺

This idea of finding the probability of some number of "successes" in n "tries" comes up enough that we'll give it a special name:

Definition 10.7 (Binomial Random Variable).
A binomial random variable models the number of "successes" in n "tries," where a success happens with probability p.

When a random variable X is binomial, we say $X \sim \text{Binomial}(n, p)$.

Note 10.8 (Bernoulli Random Variables).
Sometimes when $n = 1$, we call a binomial random variable a Bernoulli random variable.

Proposition 10.9 (Probability of a Binomial Random Variable).
When $X \sim \text{Binomial}(n, p)$, we have

$$\Pr[X = x] = \binom{n}{x} p^x (1-p)^{n-x}$$

Proof. This uses a similar argument to the coin flipping example. We choose the position of the x successes in $\binom{n}{x}$ ways. Each of these sequences has probability $p^x(1-p)^{n-x}$ of happening. Thus, we get

$$\begin{aligned}\Pr[X = x] &= \underbrace{p^x(1-p)^{n-x} + \cdots + p^x(1-p)^{n-x}}_{\binom{n}{x} \text{ times}} \\ &= \binom{n}{x} p^x (1-p)^{n-x}\end{aligned}$$

which is what we wanted. □

Definition 10.10 (Expected Value).
The expected value of a random variable is just a weighted average. Mathematically, we say that

$$E[X] = \sum_x x \cdot \Pr[X = x]$$

In addition, we extend the expected value to allow expected values of functions of random variables:

$$E[f(X)] = \sum_x f(x) \cdot \Pr[X = x]$$

With this, we introduce maybe the most important theorem in probability theory:

Theorem 10.11 (Linearity of Expectation).
For any random variables X, Y and constants a, b, then

$$E[aX + bY] = a \cdot E[X] + b \cdot E[Y]$$

This extends to arbitrarily large sums by induction.

Example 10.12 (Using Linearity).
What is the average total when rolling two dice?

Let X_1 denote the number rolled on the first die, and X_2 the number on the second. We are interested in

$$E[X_1 + X_2] = E[X_1] + E[X_2] \qquad \text{(by linearity of expectation)}$$

The dice are the same, so $E[X_1] = E[X_2]$. Then

$$E[X_1] = \sum_{x=1}^{6} x \cdot \Pr[x] = 1 \cdot \frac{1}{6} + 2 \cdot \frac{1}{6} + 3 \cdot \frac{1}{6} + 4 \cdot \frac{1}{6} + 5 \cdot \frac{1}{6} + 6 \cdot \frac{1}{6} = 3.5$$

so our expected total is $E[X_1] + E[X_2] = 3.5 + 3.5 = 7$. ☺

Definition 10.13 (Variance).
The variance of a random variable X is a measure of how spread out it is, and is given by

$$\text{Var}[X] = E\left[(X - E[X])^2\right]$$

i.e. take the average of the squared distance from the mean.

Theorem 10.14 (Easier Variance).
For any random variable X, we also have that

$$\text{Var}[X] = E[X^2] - E[X]^2$$

Proof. This is largely an exercise in algebra and applying previous theorems:

$$\begin{aligned}
\text{Var}[X] &= E\left[(X - E[X])^2\right] = E\left[X^2 - 2XE[X] + E[X]^2\right] \\
&= E[X^2] - 2E[X] \cdot E[X] + E[X]^2 && \text{by linearity} \\
&= E[X^2] - E[X]^2
\end{aligned}$$

which is what we wanted to show. □

Proposition 10.15 (Back to Binomials).
If $X \sim \text{Binomial}(n, p)$, then $E[X] = np$ and $\text{Var}[X] = np(1-p)$.

We'll conclude this section by coming back to our goal of bounding probabilities of things, and introduce two more useful inequalities:

Theorem 10.16 (Markov's Inequality).
Let X be a non-negative random variable (most random variables we care about in this class are). Then, for any $c > 0$:

$$\Pr[X \geq c] \leq \frac{E[X]}{c}$$

Theorem 10.17 (Chebyshev's Inequality).
Let X be a non-negative random variable with variance $\text{Var}[X] = \sigma^2$. Then, for any $c > 0$:

$$\Pr\left[|X - E[X]| \geq k\sigma\right] \leq \frac{1}{k^2}$$

Note 10.18 (Using all of this).
Do note that you probably won't end up needing all of these facts to get through the remaining sections. However, these are all useful ideas. Find one or two tricks that work well for you, and stick to those.

10.1 Ramsey Numbers

Once upon a time, a Hungarian sociologist made a remarkable observation about behavior of elementary school students in classrooms. It seemed that in classes of about 20 students, he could always find either:

- 4 kids, all of whom were friends with one another; or
- 4 kids, all of whom were not friends with one another.

But fortunately, the sociologist was Hungarian, and thus good at graph theory.[1] After some work, he realized that this phenomenon was hardly unique to humans, it was actually just a graph theoretic fact!

Definition 10.19 (Independent Set).
An independent set of size n, denoted $\overline{K}_n$, is a collection of n vertices, where there are no edges between any two of the vertices.

With this definition, we can state the sociologist's claim in graph theoretic language:

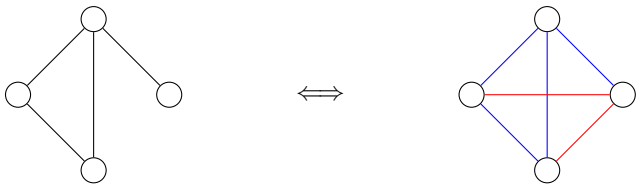
Every graph on 20 vertices contains a K_4 or a $\overline{K}_4$.

In fact, if you check Wikipedia, you'll find that we actually can pull this off with just 18 vertices—this somehow feels better. It feels more impressive to pull this off with less vertices. Let's generalize this idea:

Definition 10.20 (Ramsey Number).
The t^{th} Ramsey number, denoted $R(t,t)$ is the minimum number n such that every n vertex graph has a K_t or a $\overline{K}_t$.

Alternatively, $R(t,t)$ is the smallest number n such that coloring the edges of K_n one of two colors yields a monochromatic K_t.

Why are these two definitions the same? Just consider the absence of an edge to be a different color:

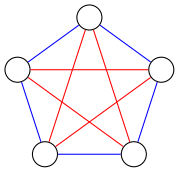


Note 10.21 (Choosing Colors).
When coloring graphs with two colors, we typically use red and blue to avoid colorblindness issues. We'll stick with this here.

[1] It's a well-known fact that basically every Hungarian is good at discrete math for some reason...

Proposition 10.22 (Value of $R(3,3)$).
The third Ramsey number is $R(3,3) = 6$.

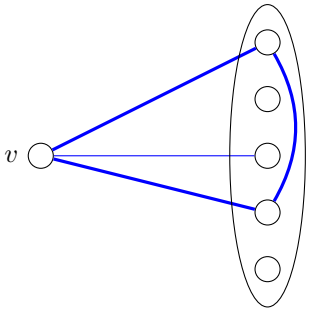
Proof. First, we'll exhibit a graph on 5 vertices which has no monochromatic K_3:

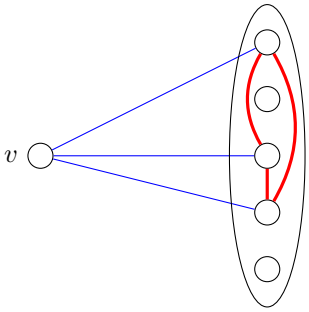


Notice that any three vertices we pick have 2 edges of one color, and 1 of the other. Thus, we know that $R(3,3) > 5$.

Now, consider K_6, and fix an *arbitrary* 2-edge-coloring of it. Consider an arbitrary vertex v. Since it connects to 5 other vertices, at least 3 red edges or at least 3 blue edges. WLOG, assume it has 3 blue edges. Now, consider the vertices which are connected to v by these 3 blue edges. There are 2 cases:

- If any of the edges between these vertices are blue, then we're done.
- If all of the edges are red, then we can use those three as our K_3!





In either case, we've found a monochromatic K_3, so we're done. □

Note 10.23 (Proving the value of a Ramsey number).
To show that $R(t,t) = n$, we have to do two things:

- Show that there is an $n-1$ vertex graph which can be 2-edge-colored so that it has no monochromatic K_t. This establishes $R(t,t) > n-1$.
- Show that *any* 2-edge-coloring of an n vertex graph has a monochromatic K_t. This establishes $R(t,t) \leq n$.

Together, these facts give $R(t,t) = n$.

From here, you might be tempted to start proving the value of all kinds of Ramsey numbers. Some clever argument might be able to get you that $R(4,4) = 18$, but at $R(5,5)$, you'll probably get stuck.

It turns out you wouldn't be alone in that. No one knows the value of $R(5,5)$. In fact, if you can find it, you'll get an immediate A in this class (and probably a PhD).

10.2 Random Graphs and the Probabilistic Method

Going forward, we'll need to apply probabilistic ideas from the beginning of this chapter to graphs. To do this, we need a way to talk about random graphs:

Definition 10.24 (The Erdős-Réyni Model).
The Erdős-Réyni random graph model, denoted $\mathcal{G}(n,p)$, constructs a random graph as follows:

- Construct an independent set with n vertices.
- For every pair of vertices, add an edge with probability p.

In fact, this definition is about all we'll need. However, let's do some examples to make sure you're comfortable with using it:

Example 10.25 (Isolated Vertex in $\mathcal{G}(n,1/2)$).
Show that as $n \to \infty$, a graph generated with the $\mathcal{G}(n,1/2)$ model almost surely will have no vertices of degree zero.

Let n be arbitrary and let A_n denote the event that this happens in a graph with n vertices (i.e. that the graph contains a vertex of degree 0).

Let E_i denote the event that vertex i in the graph is isolated after the procedure. Now, note that $A_n = \bigcup_{i=1}^{n} B_i$. We'll use the union bound:

$$\Pr[A_n] = \Pr\left[\bigcup_{i=1}^{n} B_i\right] \leq \sum_{i=1}^{n} \Pr[B_i] = n \cdot \Pr[B_1]$$

where the last equality holds since by symmetry $\Pr[B_i] = \Pr[B_j]$ for any i, j. However, B_1 happens iff the edges to vertices $2, 3, \cdots, n$ are all missing. This happens with probability $(1/2)^{n-1}$. Thus,

$$\Pr[A_n] \leq n \cdot \left(\frac{1}{2}\right)^{n-1} \implies \lim_{n\to\infty} \Pr[A_n] = 0$$

which is what we wanted to show. ☺

Example 10.26 (Triangles in $\mathcal{G}(n,p)$).
What is the expected number of triangles in a $\mathcal{G}(n,p)$ graph?

Let T be a random variable denoting the number of triangles in the graph. Index all 3-sets of vertices $1, 2, \cdots, \binom{n}{3}$, and define random variables

$$T_i = \begin{cases} 1 & \text{if the 3-set } i \text{ has all edges present (i.e. is a triangle)} \\ 0 & \text{if there's an edge missing} \end{cases}$$

Then $T = T_1 + T_2 + \cdots + T_{\binom{n}{3}}$. Now, using linearity of expectation, we get

$$E[T] = E[T_1] + E[T_2] + \cdots + E[T_{\binom{n}{3}}] = \binom{n}{3} \cdot E[T_1]$$

since by symmetry $E[T_1] = E[T_i]$ for any i.

Now, to find $E[T_1]$. We have that

$$\begin{aligned} E[T_1] &= 1 \cdot \Pr[T_1 = 1] + 0 \cdot \Pr[T_1 = 0] = \Pr[T_1 = 1] \\ &= \Pr[\text{the 3-set 1 is a triangle}] \end{aligned}$$

In order for this to be the case, we must have chosen each of the 3 edges when generating the graph. This happens with probability $p \cdot p \cdot p = p^3$. Thus, $E[T_1] = p^3$, and so we get

$$E[T] = \binom{n}{3} \cdot p^3$$

☺

Important 10.27 (The Probabilistic Method).
The probabilistic method is a technique pioneered by Erdős for proving the existence of something. It proceeds as follows:

- Generate a candidate object randomly.
- Prove that your random procedure generates the class of object you want with non-zero probability.

Then, since we can make such an object with non-zero probability, one must exist. Neat!

Example 10.28 (Round-Robin Tournaments).
Let n teams play each other in a round-robin tournament, where each plays every other team. Prove that if

$$\binom{n}{k}(1 - 2^{-k})^{n-k} < 1$$

then we can choose the outcomes of the $\binom{n}{2}$ matches such that for every set of k teams, some other team beat every team in the set.

This seems like a nearly impossible problem... how on earth do you even begin to tackle such a question? Luckily, it becomes easy when we use the probabilistic method!

Let's just choose the outcomes of each match randomly. For every subset K of k teams, let A_K be the event that none of the teams in the set beat all the others. We want to show that:

$$\Pr\left[\bigcup_{\substack{K \subseteq [n] \\ |K| = k}} A_K\right] < 1$$

i.e. The A_K's are bad, and there is a chance that all of them don't happen. First, let's try and find $\Pr[A_K]$.

- For some team not in our set K, they will beat each of the teams in our set with probability:
$$\underbrace{\frac{1}{2} \times \frac{1}{2} \times \cdots \times \frac{1}{2}}_{k \text{ times}} = 2^{-k}$$
so the probability that we fail to have that team beat every team in K is $1 - 2^{-k}$.
- There are $n - k$ other teams to try with, so we multiply to get the probability that we fail each time is $(1 - 2^k)^{n-k}$.

Therefore, we get that

$$\Pr[A_K] = (1 - 2^{-k})^{n-k}$$

Now, we take a union bound:

$$\Pr\left[\bigcup_{\substack{K \subseteq [n] \\ |K|=k}} A_K\right] \leq \sum_{\substack{K \subseteq [n] \\ |K|=k}} \Pr[A_K] = \binom{n}{k}(1 - 2^{-k})^{n-k} < 1$$

and so with positive probability, none of the events A_K occurs; i.e. there is a tournament such that for each set K, a team beats every team in the set. ☺

Note 10.29 (Non-Constructive Proofs).
Notice that in our proof above, we never actually managed to construct a graph that had the desired property. The power of the probabilistic method is precisely that we *don't* have to do this.

10.3 Finding Bounds on Ramsey Numbers

Our ultimate goal for this section will be to find bounds on the value $R(t, t)$. However, to do this, we need to introduce a more general definition:

Definition 10.30 (Generalized Ramsey Number).
$R(s, t)$ is the minimum n such that every 2-edge-coloring of K_n has a red K_s or a blue K_t (again, we use red and blue WLOG).

Note 10.31 (Sanity Checks).
Take a minute to convince yourself that the following things are true:

- $R(s, t) = R(t, s)$
- $R(1, t) = 1$
- $R(2, t) = t$

For example, we'll give a brief justification of the last one, that $R(2, t) = t$:

- *t vertices is enough.*

  If any edges are red, then we've found a red K_2 and we're done. Otherwise, all edges are blue, and thus we get a blue K_t.

- *Can't do it with $t - 1$.*

  Just build a blue K_{t-1}.

Therefore, we get that $R(2, t) = t$, which is what we wanted. ☺

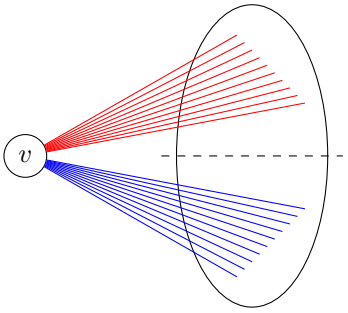
Theorem 10.32 (Ramsey Number Upper Bound).
For all $s, t \in \mathbb{Z}^+$, we have that

$$R(s,t) \leq \binom{s+t-2}{s-1}$$

Proof. We'll begin by showing that

$$R(s,t) \leq R(s-1,t) + R(s,t-1)$$

To do this, fix an arbitrary 2-edge coloring of K_n. Pick any vertex v, and split up the rest of the graph by the color of v's edges:



Now, consider the top half. If there's a red K_{s-1}, then we can add v to get a red K_s. Also, if there's a blue K_t, then we're done as well. Thus, the upper half can have at most $R(s-1,t)-1$ vertices. Similarly, the bottom half can have at most $R(s,t-1)-1$ vertices. Therefore, the entire graph has at most

$$\big(R(s-1,t)-1\big) + \big(R(s,t-1)-1\big) + 1 \text{ vertices}$$

so any n greater than $R(s-1,t) + R(s,t-1) - 1$ works; i.e.

$$R(s,t) \leq R(s-1,t) + R(s,t-1)$$

Now, we can prove the main claim by joint induction on s and t:

BC Our base cases are when one of s or t is 1:

$$R(1,t) = 1 \leq \binom{1+t-2}{1-1} \qquad \text{and} \qquad R(s,1) = 1 = \binom{s+1-2}{s-1}$$

so we're good.

IS We'll show that the bound for $R(s-1,t)$ and $R(s,t-1)$ implies it for $R(s,t)$. From above, we get:

$$\begin{aligned}
R(s,t) &\leq R(s-1,t) + R(s,t-1) && \text{from the above} \\
&\leq \binom{(s-1)+t-2}{(s-1)-1} + \binom{s+(t-1)-2}{s-1} && \text{by the IH} \\
&= \binom{s+t-3}{s-2} + \binom{s+t-3}{s-1} \\
&= \binom{s+t-2}{s-1} && \text{by Pacal's Identity}
\end{aligned}$$

which is what we wanted.

Therefore, the claim holds for any s and t by joint induction. □

Corollary 10.33 (Exponential Bound on $R(t,t)$).
For any $t \in \mathbb{Z}^+$, we have $R(t,t) \leq 4^t$.

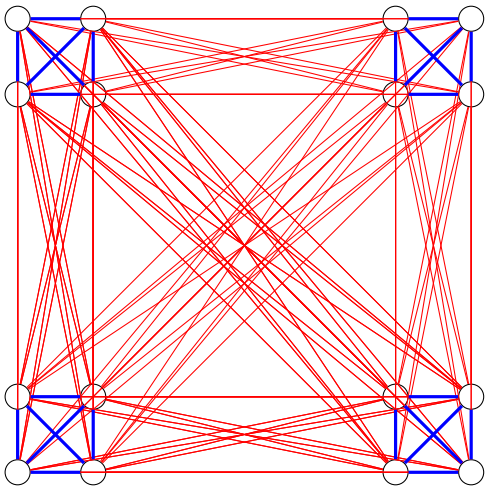
Proof.

$$R(t,t) \leq \binom{2t-2}{t-1} \leq \binom{2t}{t-1} \leq 2^{2t} = 4^t$$

□

Proposition 10.34 (Quadratic Lower Bound for $R(t,t)$).
For any $t \in \mathbb{Z}^+$, we have $R(t,t) > (t-1)^2$.

Proof. We just have to show a graph such that there are no monochromatic K_t's. The following works:



Where here the graph is drawn for $R(5,5)$. Each smaller K_4 is blue, and the rest of the graph is connected by red edges.

This process generalizes as follows. Begin by making $t-1$ disjoint, blue K_{t-1}'s. Then, fill all the missing edges as red.

- There's definitely no blue K_t, since each blue component has only $t-1$ vertices, and there's no blue edges between the K_{t-1}'s.
- There's also no red K_t, since we can't pick two vertices in the same K_{t-1}, otherwise we get a blue edge. However, there's only $t-1$ K_{t-1}'s, so we're good.

Therefore, we get that $R(t,t) > (t-1)^2$. □

Note 10.35 (Not so great sadly...).
Unfortunately, this is only a polynomial bound. We would ideally like an exponential lower bound to match the exponential upper bound we got earlier.

For a long while, this was the best we could do. However, eventually Erdős managed to get an exponential lower bound...

Theorem 10.36 (Erdős's Lower Bound).
There is a coloring of the edges of K_n where $n = 2^{t/2}$ with no monochromatic K_t; i.e. $R(t,t) > 2^{t/2}$.

Proof. We'll use the probabilistic method. For each of the $\binom{n}{t}$ sets of t vertices, define the event

$$E_S = \text{event that all edges in } S \text{ are the same color}$$

Now, randomly color the edges of our graph. We can take a union bound to get:

$$\begin{aligned}\Pr[\text{random process fails}] &= \Pr[\text{some } E_S \text{ happens}] \\ &\leq \Pr[B_{S_1}] + \cdots + \Pr[B_{S_{\binom{n}{t}}}]\end{aligned}$$

but $\Pr[B_{S_i}] = 2 \times \left(\frac{1}{2}\right)^{\binom{t}{2}}$, since we have to pick the color (2 ways), then hit $\binom{t}{2}$ 50/50's to get them all to be that color. Thus

$$\begin{aligned}&= \underbrace{2 \times \left(\frac{1}{2}\right)^{\binom{t}{2}} + \cdots + 2 \times \left(\frac{1}{2}\right)^{\binom{t}{2}}}_{\binom{n}{t} \text{ times}} \\ &= \binom{n}{t} \times 2 \times \left(\frac{1}{2}\right)^{\binom{t}{2}} = \binom{n}{t} \times 2 \times \left(\frac{1}{2}\right)^{\frac{t^2+t}{2}} \\ &= \binom{2^{t/2}}{t} \times \frac{2}{2^{\frac{t^2+t}{2}}} < \frac{\overbrace{2^{t/2} \times \cdots \times 2^{t/2}}^{t \text{ times}}}{t!} \times \frac{2}{2^{\frac{t^2+t}{2}}} \\ &= \frac{2^{t^2/2}}{t!} \times \frac{2}{2^{t^2/2 - t/2}} = \frac{1}{t!} \times \frac{2}{2^{-t/2}} = \frac{2^{t/2+1}}{t!} \\ &< 1\end{aligned}$$

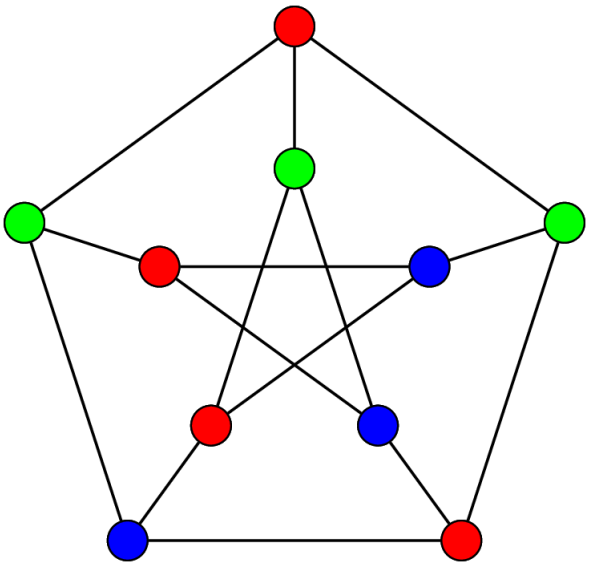
since $t!$ grows much faster than $2^{t/2+1}$. Thus, the probability of our process failing to generate a valid graph is less than 1, so such a graph must exist. □

Note 10.37 (Summarizing our bounds).
For any $t \in \mathbb{Z}^+$, we have that

$$\sqrt{2}^t < R(t,t) \leq 4^t$$

CHAPTER 11

Colorability and Planarity



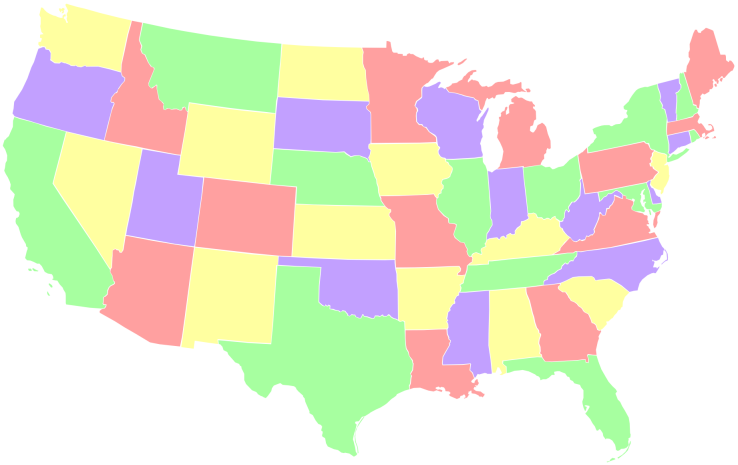
If you prove something about 3-coloring graphs, you want to prove there isn't an easy way to check it. If you somehow find efficient conditions to check, then some big governmnet guys will show up one day and you'll just disappear.

—Po-Shen Loh

Contents

11.1 Colorability

You may have noticed that when you look at maps in geography books, they are colored with just 4 colors. For example



Somehow, even for complicated maps, it seems that 4-colors are all that is necessary to color each state/country/etc. such that adjacent ones don't use the same color.[1]

Let's introduce some graph theoretic terminology related to this.

Definition 11.1 (k-Colorable, Chromatic Number).
A graph is k-colorable if you can assign one of k colors to each vertex such that no adjacent vertices have the same color.

The chromatic number of a graph, denoted $\chi(G)$, is the smallest k such that the graph is k-colorable.

When building and coloring a graph, it is often useful to think of the edges as conflicts, such as shared boarders in a map, or overlapping time slots, or something similar.

Note 11.2 (1-colorable?).
What does it mean for a graph to be 1-colorable? Well, if we have any edge, we can't pick colors for its endpoints so that they're different, so 1-colorable means there are no edges.

Note 11.3 (Increasing k).
If a graph is, say, 4-colorable, then it is also 5-colorable, 6-colorable, etc. That is, coloring a graph is easier with more colors.

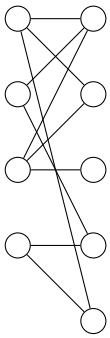
We've managed to classify 1-colorable graphs. How about 2-colorable ones? For that, we need a definition:

[1] More formally, we have that every "planar" graph is 4-colorable. Intuitively, that's just every graph that represents a map. We'll define planarity in the next section.

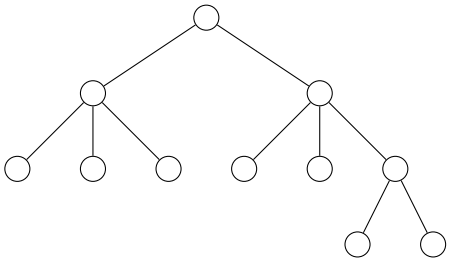
Definition 11.4 (Bipartite Graph).
A graph $G = (V, E)$ is said to be bipartite if there is a partition of $V = X \sqcup Y$ such that every edge in E has one endpoint in X and the other in Y.

We will often write $G = (X, Y, E)$ to emphasize the partition.

Here are some examples of bipartite graphs:



Bipartite graphs are often drawn like this



Any tree is also a bipartite graph

Theorem 11.5 (Classifying 2-colorable graphs).
The following are equivalent:

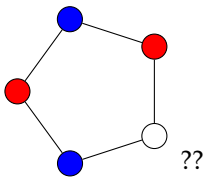
(1) G is 2-colorable

(2) G is bipartite

(3) G contains no odd cycles

Proof. We'll show that $(1) \iff (2)$ and $(1) \iff (3)$.

- $[(1) \Rightarrow (2)]$ If a graph is 2-colorable, then it has a 2-coloring. Put all the red vertices in a set X, and all the blue ones in a set Y. Then $X \sqcup Y$ is our bipartition.
- $[(2) \Rightarrow (1)]$ If a graph is bipartite, then color all the vertices in X blue and the ones in Y red. This is a 2-coloring.
- $[(1) \Rightarrow (3)]$ We'll show the contrapositive. If an odd cycle exists, it looks like:

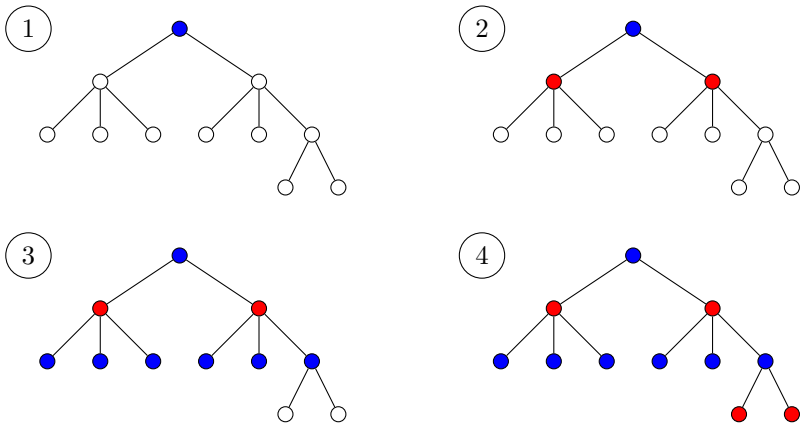


But then, either color we choose for the last vertex violates our 2-coloring. Hence, the graph is not 2-colorable.

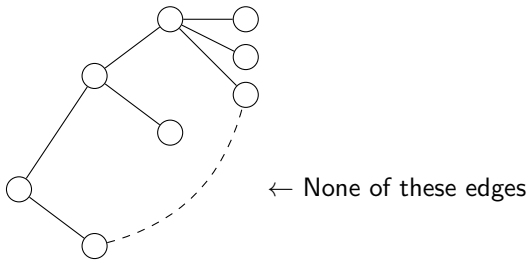
- [$(3) \Rightarrow (1)$] We'll present an algorithm to 2-color the graph:

> Breadth-First-Coloring(G)
> 1. Start at any vertex, give it a color (say blue).
> 2. Color each adjacent vertex red.
> 3. Color each each vertex adjacent to those red vertices blue; repeat 2 and 3
> 4. Repeat for each connected component.

Here's an example of a breadth first coloring:



Note that if we color the graph in a breadth first manor, we never will have:



Since then we wouldn't have actually explored the graph in this order. That is, the vertex at the end of the dotted line should be in an earlier layer.

So what could break our 2-colorability? Any edge between two vertices on the same level. But then, there must be a cycle of length

$$2 \times (\text{distance from common ancestor}) + 1$$

which is an odd cycle, a contradiction.

□

Note 11.6 (3-colorable?).
Can we classify all the 3-colorable graphs? Well, probably not. This is another one of those million dollar, NP-complete questions.

If you somehow manage to find easy to check conditions for when a graph is 3-colorable, then go claim your prize!

Theorem 11.7 (Bound on the Chromatic Number).
Every graph with maximum degree Δ is $(\Delta + 1)$-colorable.

Proof. Fix some value of Δ. We'll show that all graphs with maximum degree Δ are $(\Delta + 1)$-colorable by induction on $|V|$.

BC A graph with a single vertex is 1 colorable. And since $\Delta > 0$, we know it is certainly $\Delta + 1$ colorable.

IS Assume any graph with $n-1$ vertices is $\Delta+1$ colorable, and consider an arbitrary graph on n vertices. Remove an arbitrary vertex v. The resulting graph has $n-1$ vertices, and so is $\Delta + 1$ colorable by the IH.

Then, add v back into the graph. Since the maximum degree is Δ, v can have at most Δ neighbors. Thus, there is always a color left over to color v with, so our original graph can be $\Delta + 1$ colored.

By induction, the claim holds for any graph with maximum degree Δ. □

Example 11.8 (A Max-Cut).
There is a way to color any graph with 2 colors such that at least half of the edges have different colored endpoints.

To see this, we'll present an algorithm:

Max-Cut-Approx(G):
1. Color all the vertices red.
2. While there is a vertex where flipping its color increases the number of edges with different colored endpoints, flip its color.

With this strategy, every vertex has at least half of its edges having different colored endpoints, otherwise we would flip its color. Thus, at least half of all the edges in the graph must have different colored endpoints. In addition, this procedure must terminate, since each step strictly increases the number of edges with different colored endpoints, and this quantity is bounded by $|E|$. ☺

11.2 Planar Graphs

We've already kind of stumbled across planar graphs when we discussed maps. Let's give a more formal definition:

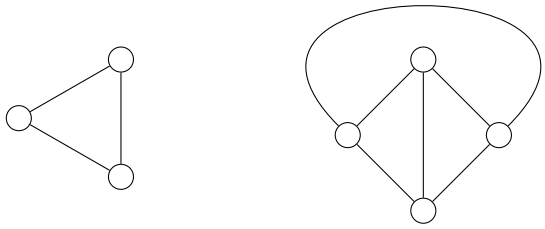
Definition 11.9 (Planar Graph).
A graph is planar if it can be drawn in the plane with no crossing edges.

To tie it back to colorability, we'll state a cool theorem (which we sadly won't be able to prove in this class).

Theorem 11.10 (4-Color Theorem).
Every planar graph is 4-colorable.

However, even though we can't prove this here, there are plenty of other things about planar graphs we can prove. Let's start with some examples.

K_3 and K_4 are both planar:



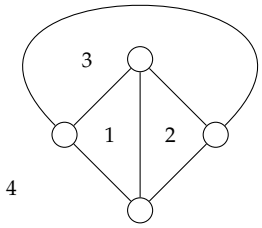
If you try to draw K_5, you'll see it is hard to do so without crossing edges. It actually isn't planar... we'll prove it later. Notice that this implies that K_6 is also not planar, since it contains a K_5.

How about $K_{3,3}$ (we use $K_{3,3}$ to denote the *complete bipartite graph*, where we make a bipartition of size 3 and 3, and add every possible edge)? This is actually the famous 3-utilities problem:

Where the goal is to connect every house to every utility without crossing lines. From childhood, you may recall that this is also impossible to do; i.e., $K_{3,3}$ is not planar.

Theorem 11.11 (Euler's Formula).
In any connected planar graph, we have $V - E + F = 2$; where V is the number of vertices, E is the number of edges, and F is the number of faces.

For example, in our K_4, we have that



$$V = 4$$
$$E = 6$$
$$F = 4$$

so indeed $V - E + F = 2$. Let's prove it for a general planar graph!

Proof. We'll go by induction.

BC Our base cases will be trees. In this case, we have that

$$V - E + F = V - (V - 1) + 1 = 2 \checkmark$$

15 Now, any other graph will have a spanning tree. Take an edge not in the spanning tree, and remove it. Call the resulting graph G'. By the IH, we know that this graph satisfies

$$V' - E' + F' = 2$$

When we add the edge back, the number of vertices doesn't change, the number of edges increases by 1, and the number of faces also increases by 1. Therefore:

$$V - E + F = V' - (E' + 1) + (F' + 1) = V' - E' + F' = 2$$

so the claim holds.

By induction, $V - E + F = 2$ for any planar graph. □

Proposition 11.12 (Another Planar Graph Relationship).
Any planar graph satisfies $E \leq 3V - 6$.

Proof. Each face has perimeter ≥ 3, so double counting the edges of the graph gives that

$$2E \geq 3F \implies \frac{2}{3}E \geq F$$

and then by Euler's formula we get that

$$\begin{aligned} 2 = V - E + F \leq V - E + \frac{2}{3}E = V - \frac{1}{3}E &\implies 2 \leq V - \frac{1}{3}E \\ &\implies E \leq 3V - 6 \end{aligned}$$

which is what we wanted. □

Corollary 11.13 (Some non-planar graphs).
The graphs K_5 and $K_{3,3}$ are not planar.

Proof. In K_5, we have $V = 5$ and $E = 10$, so from the above we have

$$10 \leq 3(5) - 6 = 9$$

which is a contradiction. Getting $K_{3,3}$ is slightly trickier. Note that in the above proof, we got $2E \geq 3F$. But $K_{3,3}$ has no triangles (it's bipartite, thus no odd cycles), so actually $2E \geq 4F$. But from Euler's formula, we get that

$$F = 2 - V + E = 2 - 6 + 9 = 5$$

and so $2(9) \geq 4(5) \implies 18 \geq 20$, a contradiction. □

CHAPTER 12

Matchings

She is tolerable; but not handsome enough to tempt me; I am in no humour at present to give consequence to young ladies who are slighted by other men.
—Mr. Darcy, Pride and Prejudice

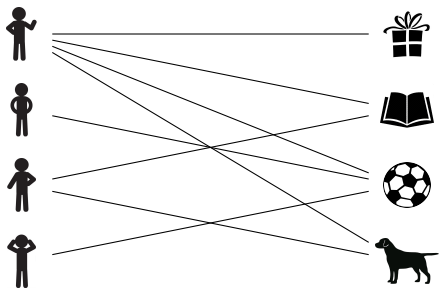
Contents

12.1 Hall's Theorem

The following scenario is common in the real world:

> People need to get stuff. However, people are tricky and they only like certain stuff. Can we give stuff to people in a way that makes them happy?

We can represent this setup using a graph:



Notice that this is just a bipartite graph! Let's define the notion of "give stuff to people in a way that makes them happy."

Definition 12.1 (Matching, Perfect Matching).
Consider a bipartite graph $G = (V, E)$ with partition $V = A \cup B$. A matching is a collection of edges which have no endpoints in common.

We say there is a perfect matching from A to B if there is a matching which hits every vertex in A.

Note 12.2 (Perfect Matchings).
Sometimes the term "perfect matching" is used to mean a matching where *every* vertex (in A or B) is hit exactly once.

However, it is sometimes practically useful to allow unused vertices on one side of our bipartition, which is why we use the terminology

> "Perfect matching from A to B"

If this matters, we will be explicit. However, we'll almost always be talking about perfect matchings from the left side of the graph to the right side of the graph.

Is there a perfect matching in our original graph from above? If you stare at it for a bit, you'll notice that no, there isn't. This is because person 2 and person 4 will only settle for the ball and nothing else. However, they can't both have it. Therefore, no perfect matching can exist.

Our goal moving forward is to extend this thinking to an arbitrary bipartite graph, and come up with conditions that allow us completely classify the bipartite graphs which have a perfect matching.

Theorem 12.3 (Hall's Marriage Theorem).
In any bipartite graph $G = (V, E)$ with bipartition $V = (X, Y)$, G admits a perfect matching from X to Y if and only if

$$\forall S \subseteq X, |S| \leq |N(S)|^a \qquad (\star)$$

The starred expression is called the Hall condition.

[a] Recall that $N(S)$ denotes the *union* of all the neighborhoods of the vertices in S. That is, all the vertices reachable from S by following only 1 edge.

Proof. The forward direction hopefully makes sense. If we have a perfect matching, then any set S we choose connects to at least $|S|$ vertices in the other side by just following the edges in the matching. Hence, $|N(S)| \geq |S|$.

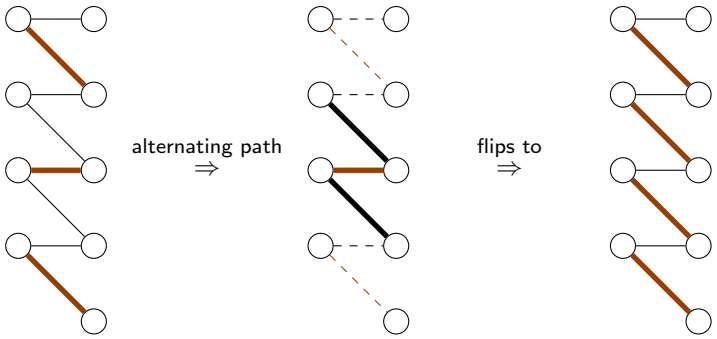
However, it's a miracle that such a simple condition ends up being sufficient as well. To show this, we'll present an algorithm to find a perfect matching in a graph satisfying the Hall condition. It's even more amazing that the algorithm is also extraordinarily simple:

> Find-Perfect-Matching(G):
> 1 While there is an alternating path from left to right, flip the edges along it.

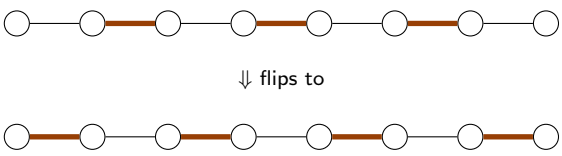
However, to make sense of this, we ought to define what an alternating path is. We'll call a path an alternating path from left to right if:

1. It starts at a vertex in the left, and ends at a vertex on the right.
2. Alternates between edges not in the matching, and edges in the matching, starting with an edge not in the matching.

For example, consider the following bipartite graph, with colored edges representing our matching so far:



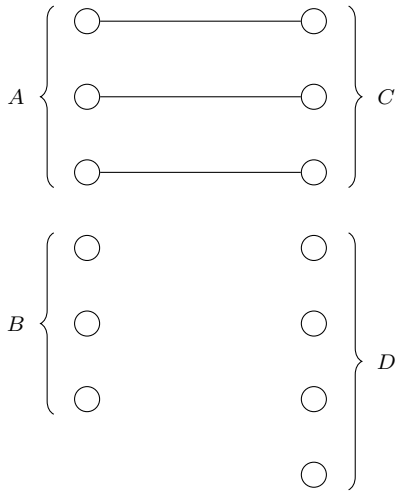
In general, we do the following:



But notice that doing this adds $+1$ to the number of edges in our matching. Therefore, every time we run step (1) in the algorithm, our matching gets 1 bigger. Therefore, the process will definitely terminate, since there are only so many edges in the graph.

Thus, it suffices to show that when we terminate (i.e. when there are no more alternating paths in the graph), then we've found a perfect matching.

By way of contradiction, assume that we don't get a perfect matching from left to right. Our graph will look something like this:

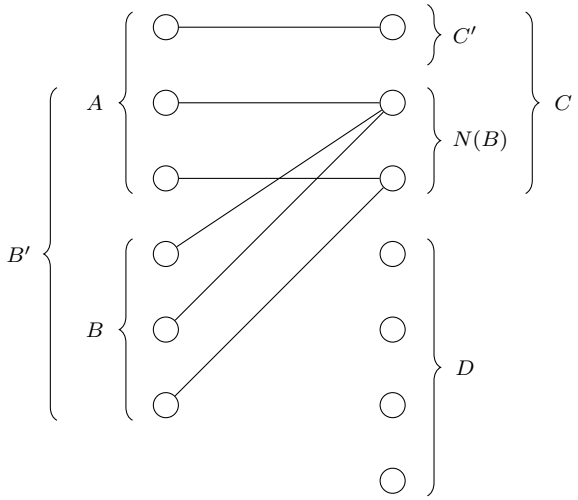


These are the ones that our algorithm has matched so far. We've moved them to the top so it's easier to follow.

First, note that our set B is non-empty, since otherwise we got a perfect matching from left to right. Now:

- $\Rightarrow$ There can be no edges from $B \to D$, otherwise, there is a one edge alternating path, of which we assume there are none.
- $\Rightarrow$ There are edges from $B \to C$. In fact, all the edges coming out of B go to C.

However, there is no need for every vertex in C to be the neighbor of a vertex in B, only some of them. Let's redraw our picture slightly:



Now, let's examine the set B'. There are no edges from $B \to D$ as discussed before, but also there are no edges from $(B' \backslash B) \to D$, since otherwise we would have an alternating path by following:

$$B \to N(B) \to (B' \backslash B) \to D$$

We also can't have any edges to C', since then we didn't make B' big enough; that is, we should have included another vertex in B: the pair of the vertex in C'.

Therefore, all of the edges from B' must go to $N(B)$; i.e. $N(B') = N(B)$. But since $B \neq \varnothing$ and everything in $A \cap B'$ has a match in $N(B)$, we know that

$$|N(B')| < |B'|$$

which contradicts Hall's condition. $\square$

12.2 Applications of Hall's Theorem

Proposition 12.4 (Matchings in Regular Graphs).
Every d-regular[a] bipartite graph has a perfect matching both from left to right and from right to left.

[a] Recall that a graph is d-regular if every vertex has degree exactly d.

Proof. Hall's Theorem allows us to just check the Hall condition. Let S be an arbitrary subset of the left. Certainly it is true that

$$\#\text{ edges out of } S \leq \#\text{ edges into } N(S)$$

But then, since the graph is d-regular, we get:

$$d|S| = \#\text{ edges out of } S \leq \#\text{ edges into } N(S) = d|N(S)|$$

and so $|S| \leq |N(S)|$. Thus, Hall's condition is true, and so the graph has a perfect matching from left to right by Hall's Theorem.

This proof also has a nice "story" version to it. We can think of every vertex in S as throwing exactly d balls at the people on the right. If people can only catch d balls, then certainly you need at least the same number of people to catch all the thrown balls. Hence, $|S| \leq |N(S)|$.

But now, notice that there is nothing special about left and right. We can just flip them and use the same argument. Therefore, there is also a perfect matching from right to left. Notice that this implies that $|\text{Right}| = |\text{Left}|$, since every vertex on the left and right are involved in exactly 1 edge in the matching. □

Corollary 12.5 (Partitioning Regular Bipartite Graphs).
Any d-regular bipartite graph can be partitioned into d perfect matchings.

Proof. Use the above to find a perfect matching. Then remove it. The result will be a $(d-1)$-regular bipartite graph. Repeat to get d perfect matchings. □

Hall's Theorem ends up being one of the most useful theorems in graph theory. It often shows up in odd ways in other problems and fields of math. We'll give one example of this:

Example 12.6 (One Last Grid Problem).
An $n \times n$ grid has some of its cells shaded in, such that every row and column has exactly $k \leq n$ shaded cells. Prove that it is possible to pick n shaded cells such that no two are in the same row or column.

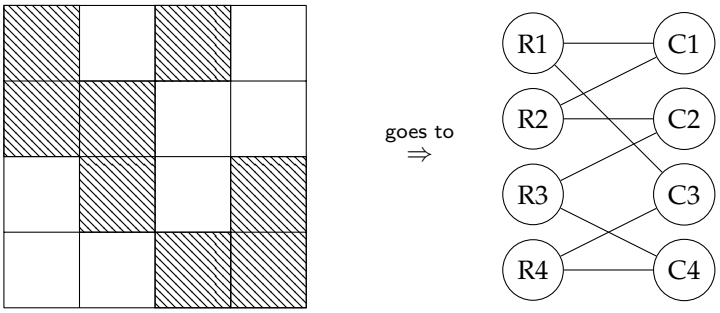
At first, it may not be obvious how graphs apply to this question, nonetheless how Hall's Theorem does. The intuition behind the construction we present is:

> We have two sets, rows and columns, and we want to somehow "pair them up" (i.e. pick cells, which involves specifying a row, column pair)

This seems more like Hall's Theorem.

We'll represent the grid as a bipartite graph. The left hand set will be the rows, and the right hand set will be the columns. We'll add an edge between a row r and a column c if the cell at (r, c) is shaded.

An example of this construction is given below:



But since there are k shaded cells in each row and column, the resulting graph will be k-regular, and thus have a perfect matching.

If we pick the row, column pairs in this matching, then we will pick a shaded square in each row and column, which is what we wanted. ☺