Contents

1 DIS 0B (Intro)

- My OH is Monday 1-2 and Tuesday 3-4 in Cory 212.
- Email is first.last@
- 3rd year cs + math major
- hobbies?

1.1 Some CS70 advice

- Goal: enhance problem solving techniques/approach
- Don't fall behind on content, catching up will not be fun
- problems, problems, more problems
- Ask lots of questions (imperative for strong foundation)
- Don't stress, we're in this ride together

1.2 Propositional Logic

Relevant notation:

- $\wedge$ = and
- $\vee$ = or
- $\neg$ = not
- $\implies$ = implies
- $\exists$ = there exists
- $\forall$ = forall
- $\mathbb{N}$ = natural numbers $\{0, 1, \ldots\}$
- $a|b$ = a divides b

$P \implies Q$ is an example of an implication. We can read this as "If P, then Q." An implication is false only when P is true and Q is false. If P is false, the implication is vacuously true.

Definition 1.1 (Contrapositive)
If $P \implies Q$ is an implication, then the implication $\neg Q \implies \neg P$ is known as the **contrapositve**.

An important identity is that $P \implies Q \equiv \neg Q \implies \neg P$.

1.3 Proofs

Induction will be in its own section.

Different methods.

1.3.1 Direct proof

Want to show $P \implies Q$ by assuming P and logically concluding Q.

1.3.2 Contraposition

Want to show $P \implies Q$ by equivalently proving $\neg Q \implies \neg P$.

1.3.3 Contradiction

Want to show P. We do this by assuming $\neg P$ and concluding $R \wedge \neg R$.

Why? Idea is that if we can show the implication $\neg P \implies (R \wedge \neg R)$ is True, this is the same as showing $\neg P \implies F$ is True. The contraposition gives $T \implies P$.

1.3.4 Cases

Break up a problem into multiple cases i.e. odd vs even.

2 DIS 1A (Induction)

Goal of induction is to show $\forall n P(n)$.

2.1 (Weak) Induction

- Prove $P(0)$ is true (or relevant base cases), then $\forall n \in \mathbb{N}\,(P(n) \implies P(n+1))$.
- Induction dominoes analogy!
- Sometimes you might have multiple base cases (Problem about $4x + 5y$ in Notes 3)

2.2 Strengthening the Hypothesis

Sometimes proving $P(n) \implies P(n+1)$ is not straightforward with induction. In such a scenario, we can try to introduce a (stronger) statement $Q(n)$. We want to construct Q such that $Q(n) \implies P(n)$. Inducting on Q proves P.

2.3 Strong Induction

- Prove $P(0)$ is true (or relevant base cases), then $\forall n\,((P(0) \wedge P(1) \wedge \cdots \wedge P(n)) \implies P(n+1))$.
- Dominoes analogy, but emphasis on the difference between weak and strong induction (assuming middle domino works vs everything from start to middle).

2.4 Weak vs Strong

A common point of confusion is when one should use strong induction in lieu of weak induction. Strong induction **always** works whenever weak induction works. However, there may be scenarios in which the induction hypothesis to prove $n = k + 1$ requires more information than just $n = k$. A scenario like this requires strong induction.

3 DIS 1B (Stable Matching)

Cool application of induction.

3.1 The Propose and Reject Algorithm

Suppose jobs proposes to candidates.

- both jobs and candidates have a list of preferences
- every day a job that doesn't have a deal with a candidate will propose to the next best candidate on its preference list
- every candidate will tentatively "waitlist" the offer from the job (put it on a string)
- if a candidate has multiple offers, they will choose the one they prefer the most
- the algorithm ends when every candidate has a job on their "waitlist" (all these WLs becomes acceptances)

(walk through q1 of dis as a class to visualize this)

3.2 Stability

Definition 3.1 (Rogue Couple)
A job-candidate pair (J,C) is denoted as a **rogue couple** if they prefer each other over their final assignment in a stable matching instance.

Definition 3.2 (Unstable)
A matching that has at least one rogue couple is considered **unstable**.

Conversely, a **stable** matching is one that has no rogue couples.

Some tricky vocab stuff like stable matching instance.

Lemma 1 (Improvement) *If a candidate has a job offer, then they will always have an offer from a job at least as good as the one they have right now.* □

Matchings produced by the algorithm are always **stable**.

3.3 Optimality

The propose and reject algorithm is proposer *optimal* and receiver *pessimal*.

Definition 3.3 (optimal)
A pairing is optimal for a group if each entity is paired with who it most prefers while maintaining stability.

Can be thought of a (well that's the best I could do) analogy.

Definition 3.4 (pessimal)
A pairing is pessimal for a group if each entity is paired with who it least prefers while maintaining stability.

Can be thought of a (well it can't get worse than this) analogy.

3.4 Potpourri

It is possible that there exists a stable matching instance that is neither job optimal nor candidate pessimal.

Consider the following preferences

Jobs	Preferences
A	$1 > 2 > 3$
B	$2 > 3 > 1$
C	$3 > 1 > 2$

Candidates	Preferences
1	$B > C > A$
2	$C > A > B$
3	$A > B > C$

The matching above can generate (at least) 3 stable matching instances

$$S = \{(A,1), (B,2), (C,3)\}$$
$$T = \{(A,3), (B,1), (C,2)\}$$
$$U = \{(A,2), (B,3), (C,1)\}.$$

We see

- S is job-optimal/candidate-pessimal (result of running propose and reject with jobs proposing to candidates)
- T is candidate-optimal/job-pessimal (result of running propose and reject with candidates proposing to jobs)
- U is neither optimal nor pessimal for both candidates and jobs (S and T) corroborate that.

Also some other important facts that can be seen (from discussion worksheet questions):

- There is at least one candidate that will receive only one proposal (that too on the last day)
- We can upper bound the number of days needed by P&R algorithm to $(n-1)^2 + 1 = n^2 - 2n + 2$ (think about why)
- As a consequence of above, we can upper bound the number of rejections needed by P&R algorithm to $(n-1)^2 = n^2 - 2n + 1$ rejections.

4 DIS 2A (Graphs)

4.1 Notation

- V denotes set of vertices (points)
- E denotes set of edges (lines)
- $|V|$ denotes size of set of vertices i.e number of vertices; $|E|$ similarly
- Graph G with vertices V and edges E is denoted $G = (V,E)$.

4.2 Vocabulary

Definition 4.1 (Path)
A **path** is a sequence of edges. In CS70, we assume a path is *simple* which means no repeated vertices.

Definition 4.2 (Cycle)
A **cycle** is a simple path that starts and ends at the same vertex.

Definition 4.3 (Walk)
A **walk** is any arbitrary connected sequence of edges.

Definition 4.4 (Tour)
A **tour** is a walk that starts and end at the same vertex.

Definition 4.5 (Connected)
A graph is **connected** if there exists a path between any two distinct vertices.

Definition 4.6 (Eulerian Walk)
An **Eulerian walk** is a walk covering all edges without repeating any.

Definition 4.7 (Eulerian Tour)
An **Eulerian tour** is an Eulerian walk that starts and ends at the same vertex.

To summarize,

	no repeated vertices	no repeated edges	start = end	all edges	all vertices
Walk					
Path	✓	✓			
Tour			✓		
Cycle	✓*	✓	✓		
Eulerian Walk		✓		✓	
Eulerian Tour		✓	✓	✓	
Hamiltonian Tour	✓	✓	✓		✓

(*except for start and end vertices)

Theorem 4.1 (Euler's Theorem)
An undirected graph G has an Eulerian tour iff G is connected and all its vertices have even degree.

The requires condition for an Eulerian walk is that we have exactly 2 vertices of odd degree. (Of course, the case of 0 odd vertices trivially works since we claim from Euler's Theorem that we can find an Eulerian tour which is a stronger statement than an Eulerian walk)

Definition 4.8 (Bipartite)
A graph is considered bipartite if V can be partitioned into two sets L and R where $V = L \cup R$ such that there are no edges between vertices in L and no edges between vertices in R.

4.3 The holy grail for graph proofs

Induct, induct, induct, and induct.

- Think about what you want to induct on (edges or vertices???)
- Base case (read the problem carefully!)
- Prove for n by going from $n \to n-1 \to I.H. \to n$.
  - **DO NOT** go from $n-1 \to n$ directly.
  - Why? Build-up error!
  - Good example of build-up error when trying to prove "if every vertex of a graph has degree at least 2, then there exists a cycle of length 3." Any attempt at induction will give us a false proof but we cannot make square from triangle!
  - It's also a logistical nightmare lol (in the times it might accidentally work). Try generating all 5-vertex trees from all 4-vertex trees yikes.

4.4 Relevant Potpourri

Some other relevant information.

Definition 4.9 (Degree)
The **degree** of a vertex v denoted $\deg(v)$ is defined to be the number of incident edges to v.

Lemma 2 (Handshake)

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

□

The idea of a degree (with no adjective) is only well-defined for undirected graphs. We see for directed graphs it's a little funky; we need to introduce the concept of indegree and outdegree.

In a directed graph, the number of outgoing edges equals the number of ingoing edges.

We will discuss trees, planarity, coloring, and hypercubes in the next discussion.

5 DIS 2B (More Graphs)

5.1 Trees

A graph $G = (V,E)$ is a Tree if any of the statements below is true. TFAE (The following are equivalent):

- G is connected and has no cycles
- G is connected and $|E| = |V| - 1$
- G is connected and removing a single edge disconnects G
- G has no cycles and adding a single edge creates a cycle

Definition 5.1
A leaf is a node of degree 1.

A consequence of above is that every tree has at least 2 leaves.

5.2 Planarity

Definition 5.2 (planar)
A graph is **planar** if it can be drawn without any edge crossings.

Theorem 5.1 (Euler)
For every connected planar graph, $f + v = e + 2$.

Corollary 1 *If a graph is planar, then* $e \leq 3v - 6$. □

Theorem 5.2 (Kuratowski)
A graph is non-planar iff it contains K_5 or $K_{3,3}$.

(draw the two above graphs on the board)

The notation K_x denotes a complete graph with x vertices.

Definition 5.3 (complete graph)
A **complete graph** is a graph where all possible edges exist. Formally, in graph $G = (V,E)$, for any distinct $u,v \in V$, then $\{u,v\} \in E$.

5.3 Coloring

Two types: edge and vertex

- edge: color edges so that no two adjacent edges have the same color
- vertices: color vertices so that no two adjacent vertices have the same color

Theorem 5.3 (4 color theorem)
If a graph is planar, then it can be colored with 4 (or less) colors.

5.4 Hypercubes

A hypercube of dimension n is a graph whose vertices are bitstrings of length n. An edge between two vertices exists iff the two vertices differ at exactly 1 bit.

(draw $n = 1, 2, 3$ on the board)

We can see that $|V| = 2^n$ and $|E| = n2^{n-1}$.

Give some motivation on induction on hypercubes.

6 DIS 3A (Modular Arithmetic)

The relevant notation we'll be using for this section is expressions of the form

$$a \equiv b \pmod{x}$$

reads "a is equivalent to b mod x". It means that the remainder of a when divided by x equals the remainder of b when divided by x.

An important identity is that

$$a \equiv b \pmod{x} \iff (\exists k \in \mathbb{Z})(a = b + kx).$$

Talk about the "clock analogy".

Example 6.1
We can see a display of some of the properties:

- Addition: $7 + 4 \equiv 1 \pmod 5$
- Subtraction: $7 - 4 \equiv 1 \pmod 2$
- Multiplication: $2 \cdot 3 \equiv 0 \pmod 6$.
- Division??

In modular arithmetic, division is not well-defined. The opposite of multiplication is multiplying by the modular inverse.

Definition 6.1 (modular inverse)
The value a is the **modular inverse** of x with respect to mod m if

$$ax \equiv 1 \pmod{m}.$$

Does an inverse always exist? No.

Theorem 6.1
Let x and m be positive integers. Then $x^{-1} \pmod m$ exists and is unique only if $\gcd(x,m) = 1$.

Definition 6.2 (Greatest Common Divisor)
The **greatest common divisor** (gcd) of two integers a, b is the greatest $d \in \mathbb{Z}$ such that $d|a$ and $d|b$.

How does one efficiently calculate the GCD?

Algorithm 6.1 (Euclidean Algorithm)
function GCD(a,b)
  if $b = 0$ **then**
    return a
  return GCD($b, a \mod b$)

7 DIS 3B (More Modular Arithmetic)

7.1 Modular inverse

Lemma 3 (Bézout) *For integers x,y such that* $\gcd(x,y) = d$*, there exist integers a and b that obey*

$$ax + by = d.$$

□

We care about the case when $\gcd(x,y) = d = 1$.

Why? This is how we can find the modular inverse.

If $ax + by = 1$, taking $\mod x$ gives us

$$by \equiv 1 \pmod{x} \implies b \equiv y^{-1} \pmod{x}.$$

Similarly, taking $\mod y$ gives us

$$ax \equiv 1 \pmod{y} \implies a \equiv x^{-1} \pmod{y}.$$

Takeaway: the values of a and b we will solve for (Q1 on discussion) give us the inverse of x with respect to y and vice versa.

7.2 Chinese Remainder Theorem (CRT)

Theorem 7.1 (CRT)
For pairwise relatively prime integers $m_1, m_2, \ldots, m_n$, the modular system

$$\begin{aligned} x &\equiv a_1 \pmod{m_1} \\ x &\equiv a_2 \pmod{m_2} \\ &\vdots \\ x &\equiv a_n \pmod{m_n} \end{aligned}$$

has a unique solution $x \pmod{m_1 m_2 \cdots m_n}$.
To clarify, the term pairwise relatively prime means for any distinct i, j, it follows $\gcd(m_i, m_j) = 1$.

How do we solve the system above? Discussion Q2...

...or we can solve them a faster way (not taught in the course lol)

Example 7.1
Suppose we take the first two systems from Q2 on discussion.

$$\begin{aligned} x &\equiv 1 \pmod{3} \\ x &\equiv 3 \pmod{7}. \end{aligned}$$

Since $\gcd(3,7) = 1$, CRT tells us x has a unique solution mod 21. The first equation tells us there exists some integer k such that $x = 1 + 3k$. Plugging this into the second equation we have

$$1 + 3k \equiv 3 \pmod{7} \implies k \equiv 3 \pmod{7}.$$

Plugging in $k = 3$ gives $x \equiv 10 \pmod{21}$.

If we wanted to solve entirety of Q2 this way, we then apply the same trick above to the systems

$$x \equiv 10 \pmod{21}$$
$$x \equiv 4 \pmod{11}.$$

8 DIS 4A (FLT, RSA)

8.1 Fermat's Little Theorem

A relevant theorem in modular arithmetic that will help us with RSA is Fermat's Little Theorem (FLT).

Theorem 8.1 (Fermat's Little Theorem (FLT))
For prime p and $a \in \{1, 2, \ldots, p-1\}$, it follows

$$a^p \equiv a \pmod{p}.$$

Special case, if a is not divisible by p, then

$$a^{p-1} \equiv 1 \pmod{p}.$$

8.2 RSA

Objective: Alice transfers info to Bob without Eve cracking it.

8.2.1 The algorithm

Here's a detailed outline of how the scheme works for RSA with 2 primes:

1. Entire world knows about a public key (N, e) where $N = pq$ for primes p and q such that $\gcd(e, (p-1)(q-1)) = 1$.
2. Alice and Bob meet in private, and Alice tells Bob what p and q are.
3. On his own time, Bob computes $(p-1)(q-1)$ and then calculates
$$d = e^{-1} \pmod{(p-1)(q-1)}.$$
(Think about why we know such a d must exist)
4. To encrypt her message x, Alice sends $E(x)$ to Bob where
$$E(x) = x^e \pmod{N}.$$
5. To decrypt the message received y, Bob calculate $D(y)$ where
$$D(y) = y^d \pmod{N}.$$
High level idea of why this works:
$$\begin{aligned} D(E(x)) &= D(x^e) \pmod{N} \\ &= x^{ed} \pmod{N} \\ &= x \pmod{N}. \end{aligned}$$
More detailed proof by cases in page 3? of Note 7.

8.3 Why does RSA work?

- N is too large to brute force solve x where $y = x^e \pmod{N}$.
- N is too large to factor into $p \cdot q$. Factorization is an intractable problem!

9 DIS 4B (Polynomials)

A single variable expression of the form

$$p(x) = a_n x^n + a_{n-1}x^{n-1} + \ldots + a_1 x + a_0$$

for reals a_i and x is denoted a polynomial.

Definition 9.1 (degree)
The degree of a polynomial $p(x)$, often denoted $\deg(p)$, is the value of the largest exponent of $p(x)$.

For example, any quadratic function has degree 2.

We mainly explore two relevant properties in this section.

Note 9.1 (Proprety 1)
If $\deg(p) = d$, then $p(x)$ has at most d roots.

Note 9.2 (Property 2)
Given $d+1$ distinct (x,y) points, we can find/compute a unique degree d polynomial.

The concept of secret sharing follows directly from property 2.

9.1 Finite Fields

We will be using notation $GF(p)$ which represents a finite field (aka Galois Field) with respect to modulo p. All operations in this field are done in $\mod p$. We want to convert all fractions to their modular inverse equivalents.

Example 9.1
If we're working in $GF(5)$, we remark

$$7x^2 \equiv 2x^2 \pmod 5$$

and

$$\frac{1}{8} \equiv 8^{-1} \equiv 3^{-1} \equiv 2 \pmod 5.$$

9.2 Lagrange Interpolation

For $d+1$ points of the form $(x_1, y_1), \ldots, (x_{d+1}, y_{d+1})$, we can construct a unique degree d polynomial

$$p(x) = \sum_{i=1}^{d+1} y_i p_i(x)$$

where

$$p_i(x) = \frac{\prod_{j \neq i} x - x_j}{\prod_{j \neq i} x_i - x_j}.$$

10 DIS 5A (Error Correcting Codes)

Objective: transmit n packets of data (integers).

Two problems may arise.

1. Packets get erased/lost (erasure errors)! If we know we have up to k packet erasures, we fix this by sending $n+k$ packets.
2. Packets get corrupted (general errors)! If we know we have up to k packets corrupted , we fix this by sending $n+2k$ packets.

If we run into erasure errors, we simply use interpolation to recover the lost packets.

If we run into general errors on the other hand, we need a more powerful tool.

10.1 Berlekamp-Welch Algorithm

We need to identify which indices the error occurs at. Messages are encoded by some polynomial $P(x)$. Our goal is to retrieve $P(x)$.

1. Suppose we know error at k bits. Define the error polynomial
$$E(x) = (x - e_1)(x - e_2)\cdots(x - e_k).$$
2. Denote the ith packet info we **see** as r_i. Note, r_i may not be the actual value (might be a corrupted value).
3. Solve the equations $P(i)E(i) = r_i E(i)$.
4. Define polynomial $Q(x) := P(x)E(x)$.
5. Substituting, we have
$$Q(i) = P(i)E(i) = r_i E(i).$$
6. We solve linear equations generated by $Q(i) = r_i E(i)$ in step 5 to find the polynomials $E(x), Q(x)$.
7. Once we have that, we can calculate
$$P(x) = Q(x)/E(x).$$

11 DIS 5B (Countability)

11.1 Terminology

For all definitions, we use a function $f : A \to B$. A is called the **domain** and B is called the **codomain**.

Definition 11.1 (Injection (one-to-one))
A function f is *injective* or *one-to-one* if no two points in the domain map to the same point in the codomain. Mathematically for all $a \in A$ and $b \in A$,

$$f(a) = f(b) \implies a = b.$$

Definition 11.2 (Surjective (onto))
A function f is *surjective* or *onto* if every point in the codomain has a point in the domain that maps to it. Mathematically, for all $b \in B$ there exists an $a \in A$ such that $f(a) = b$.

Definition 11.3 (Bijective (one-to-one correspondence))
A function f is *bijective* or has a *one-to-one correspondence* if it is both injective (one-to-one) and surjective (onto).

Definition 11.4 (Cardinality)
The **cardinality** of a set A, denoted $|A|$, is equal to the number of elements in the set.

Two sets A and B have the same cardinality (size) if there exists a bijection between A and B. Another way is to show $|A| \leq |B|$ and $|B| \leq |A|$ (this is how we prove $|\mathbb{N}| = |\mathbb{Q}|$).

11.2 The Countable

A set S is **countable** if there exists a bijection between S and $\mathbb{N}$ or another countable set. The main idea is this concept of enumeration. If we can find a way to "enumerate" or number a set, we say it's countable.

Note: countable sets may be infinite!

Some examples of common countable sets: $\mathbb{N}$, $\mathbb{Q}$, $\mathbb{Z}$, $\mathbb{Z} \times \mathbb{Z}$, set of all finite length bit strings, set of all polynomials with coefficients in $\mathbb{N}$.

11.3 The Uncountable

Effectively, the sets that aren't countable are considered **uncountable**.

Common uncountable sets: power set, $\mathbb{R}$, set of infinite length bit strings

How do we prove a set S is uncountable. In this class, either we show $|S| > |\mathbb{N}|$ or...

11.4 Cantor Diagonalization

...Cantor Diagonalization. The main idea of Cantor is to show that we can always create a new number that belongs in S that was not originally in S. As a result, we cannot possibly fathom how large S is and enumerate all of its entries since we can always create new entries based on all the ones already in S.

(Lot of words, walk through visual example on board).

12 DIS 6A (Computability)

Just think of a program like a piece of text (code).

12.1 The Halting Problem

Definition 12.1 (The Halting Problem)
We claim that the ability to determine if a program P will terminate on input x is uncomputable. In other words, there does not exist computer program (code) that can determine this.

In this class, the way we will prove a problem is uncomputable is by reducing the Halting Problem to this problem. You will explore in further detail in classes like CS170, that if problem A reduces to problem B, then problem B is at least as computationally hard as problem A.

In our case, if we can display that the halting problem reduces to our problem (i.e. if we can solve our problem, we can solve the halting problem), then this shows that our problem is uncomputable.

A basic template to prove some program Other is uncomputable

```
def TestHalt(P, x):
    def Q(y):
        run P(x)
        return <whatever makes TestOther true>
    return TestOther(Q, y)
```

12.2 Foreshadowing

We will cover counting more in depth next discussion.

13 DIS 6B (Counting)

We introduce a topic in this class called **the first rule of counting**. This effectively says if I have k boxes with $n_1, n_2, \ldots, n_k$ items per respective box, the number of ways to choose 1 item per box is

$$n_1 \cdot n_2 \cdots n_k.$$

Example 13.1
How many ways can we arrange n books on a bookshelf?

Definition 13.1 (factorial)
The factorial function of n denoted $n!$ represents the quantity

$$n! = \prod_{k=1}^{n} k.$$

If you want to check your understanding for small values of n,

n	$n!$
0	1
1	1
2	2
3	6
4	24
5	120
6	720
7	5040

To shorthand future notation, we will introduce the binomial coefficients.

How many ways can we choose k objects from a total of n objects?

Definition 13.2 (Binomial Coefficient)
The binomial coefficient

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

represents the total number of ways we can choose k objects from a total of n objects for $k \leq n$.

13.1 A sampling synopsis

The number of ways I can pick k objects from a total of n objects is...

	sampling with replacement	sampling without replacement
order matters	n^k	$n(n-1)\cdots(n-k+1)$
order doesn't matter	$\binom{n+k-1}{k-1}$	$\binom{n}{k}$

13.2 Stars and Bars

The case when order doesn't matter and we are sampling with replacement is coined *stars and bars*.

The number of ways to throw n balls into k distinguishable bins is

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

Tip: When doing stars and bars problems, if the question says the bins must have a minimum number of balls, add the minimum # of balls to each bin and do stars and bars with the remaining balls.

Example 13.2
If a problem effectively says that each bin has a positive number of balls, we can add 1 ball to each bin and figure out how to distribute the remaining $n-k$ balls.

13.3 Inclusion-Exclusion

Think of a venn diagram!

For two set case:

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

For three set case:

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

13.4 Combinatorial Proofs

Focuses on proving mathematical expressions in words with a story.

Example 13.3
Prove that

$$\binom{n}{r} = \binom{n}{n-r}.$$

Pick the side that looks easier and think about what it means. Now create a story that's equivalently portrayed by the other side.

To read on your own time: `https://drive.google.com/file/d/1Nzbdno6c_6n-T3A7rmqhqdIUWsYck6bO/view?usp=share_link`

14 DIS 7B (Intro to Probability)

Definition 14.1 (sample space)
The *sample space* denoted Ω is the set of all possible outcomes.

Two main properties of probability:

- $0 \leq \mathbb{P}[x] \leq 1$ for all $x \in \Omega$.
- $\sum_{x \in \Omega} \mathbb{P}[x] = 1$. In words, the sum of the probabilities of all outcomes is 1.

What really is $\mathbb{P}[x]$?

$$\mathbb{P}[x] = \frac{\text{\# of outcomes satisfying x}}{\text{total \# of outcomes}} = \frac{|X|}{|\Omega|}.$$

Definition 14.2 (Complement)
The *complement* of an event X is denoted $\overline{X}$ where $\overline{X} = \Omega \setminus X$.

Example 14.1
I flip a fair coin 10 times. What's the probability I get at least 1 head?

14.1 Inclusion-Exclusion

We saw them for counting. We now see them for probability.

$$\mathbb{P}[A \cup B] = \mathbb{P}[A] + \mathbb{P}[B] - \mathbb{P}[A \cap B]$$

and

$$\mathbb{P}[A \cup B \cup C] = \mathbb{P}[A] + \mathbb{P}[B] + \mathbb{P}[C] - \mathbb{P}[A \cap B] - \mathbb{P}[A \cap C] - \mathbb{P}[B \cap C] + \mathbb{P}[A \cap B \cap C].$$

15 DIS 8A (Conditional Probability)

So far we've looked at the likelihood of some event A occurring. What if I want to look at the likelihood of some even A occurring given that some event B occurred?

This is what spurs the insight into conditional probability. The notation "$A|B$" should be read as "A given B".

Theorem 15.1 (Bayes Rule)

$$\mathbb{P}[A|B] = \frac{\mathbb{P}[A \cap B]}{\mathbb{P}[B]}.$$

A nice corollary of Bayes Rule is

$$\mathbb{P}[A|B] = \frac{\mathbb{P}[B|A]\,\mathbb{P}[A]}{\mathbb{P}[B]}.$$

15.1 Total Probability

We explore an idea called the law of total probability.

Theorem 15.2 (Law of Total Probability)

$$\begin{aligned}\mathbb{P}[B] &= \mathbb{P}[A \cap B] + \mathbb{P}\left[\overline{A} \cap B\right] \\ &= \mathbb{P}[B|A]\,\mathbb{P}[A] + \mathbb{P}\left[B|\overline{A}\right]\mathbb{P}\left[\overline{A}\right].\end{aligned}$$

15.2 Independence

Two events A and B are independent if the occurrence of one does not affect the likelihood of the other. Concretely, for independence TFAE:

1. $\mathbb{P}[A \cap B] = \mathbb{P}[A]\,\mathbb{P}[B]$
2. $\mathbb{P}[A|B] = \mathbb{P}[A]$.

16 DIS 8B

Discussion 8B does not cover any new content topic-wise. Please refer to DIS 8A mini lecture notes.

17 DIS 9A (Random Variables)

Recall from last week, we looked at the concept of independence: two events A and B are independent if knowing whether B occurred tells us nothing about whether A occurred.

17.1 Unionized Events

Definition 17.1 (mutually exclusive)
Events A and B are **mutually exclusive** if $\mathbb{P}[A \cap B] = 0$.

- Generally, mutually exclusive events are (almost) never independent.
- Never want to assume a sequence of events are exclusive or independent for that matter.

Definition 17.2 (Union bound)
For events $A_1, A_2 \ldots, A_n$, **union bound** approximation claims

$$\mathbb{P}[A_1 \cup A_2 \cup \cdots \cup A_n] \leq \mathbb{P}[A_1] + \mathbb{P}[A_2] + \ldots + \mathbb{P}[A_n].$$

The intuition of above should follow from Inclusion-Exclusion.

17.2 Intro to Random Variables

For a sample space Ω, a *random variable* X is a function $X : \Omega \to \mathbb{R}$; it maps $X(\omega) \to \mathbb{R}$ for every $\omega \in \Omega$.

17.2.1 Distribution of R.V.

There are two important things for any R.V.

- The set of all values it can take (the ω values)
- Probabilities with which it takes on each of those values.

For example, $X = a$ is an *event* that is a set modeled by $S = \{\omega \in \Omega \mid X(\omega) = a\}$. As a result, if I wanted to compute the probability of an event it would follow the form $\mathbb{P}[X = a] = \frac{|S|}{|\Omega|}$.
We will work with some well known distributions in discussion today, but I'll formally define them on Thursday.

18 DIS 9B (Distributions and Expectations)

We will be looking at discrete random variables for the time being.

18.1 Well known distributions

18.1.1 Bernoulli

Bernoulli RVs either output 0 or 1. If $X \sim \text{Bernoulli}(p)$ then

$$\mathbb{P}[X = i] = \begin{cases} p & i = 1 \\ 1 - p & i = 0 \end{cases}.$$

18.1.2 Binomial

Binomial RVs take in a fixed number of trials n and probability of success p and calculate the probability of i successes for all $i \leq n$. If $X \sim \text{Binomial}(n, p)$, then

$$\mathbb{P}[X = i] = \binom{n}{i} p^i (1 - p)^{n-i}.$$

18.1.3 Geometric

Geometric RVs for a fixed probability of success p determine the probability that it takes a certain number of trials i until we see our **first** success. If $X \sim \text{Geometric}(p)$, then

$$\mathbb{P}[X = i] = (1 - p)^{i-1} p.$$

These variables are also cool because they are memoryless!

18.2 Joint Distributions

For two RVs X and Y, their *joint distribution* is denoted by the values $\mathbb{P}[X = a, Y = b]$ for all possible a that X can output and all possible b that Y can output.

Suppose from a joint distribution we want to retrieve a distribution of a single RV. How?

Definition 18.1 (Marginal distribution)
From a joint distribution, the distribution of a single RV is defined as its *marginal distribution*. To calculate it,

$$\mathbb{P}[X = a] = \sum_b \mathbb{P}[X = a, Y = b]$$

Definition 18.2 (Independence)
For *independence* in a joint distribution setting, we must have

$$\mathbb{P}[X = a, Y = b] = \mathbb{P}[X = a]\,\mathbb{P}[Y = b].$$

18.3 Expectation

Definition 18.3 (Expected Value)
The *expected value* of a random variable X, denoted $\mathbb{E}[X]$, is the anticipated average value of X. Mathematically,

$$\mathbb{E}[X] = \sum_{i \in \Omega} i \cdot \mathbb{P}[X = i].$$

Example 18.1
The expected value of a single fair dice roll is

$$\begin{aligned}\mathbb{E}[X] = \sum_{i \in \Omega} i \cdot \mathbb{P}[X = i] &= 1 \cdot \mathbb{P}[X = 1] + 2 \cdot \mathbb{P}[X = 2] + \ldots + 6 \cdot \mathbb{P}[X = 6] \\ &= 1 \cdot \frac{1}{6} + 2 \cdot \frac{1}{6} + \ldots + 6 \cdot \frac{1}{6} \\ &= \frac{7}{2} = 3.5\end{aligned}$$

18.3.1 Linearity of Expectation

For any random variables X and Y, the property

$$\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y]$$

is always true and is often denoted *linearity of expectation*.

A result of above is that for RV X and constant c, we have $\mathbb{E}[cX] = c\mathbb{E}[X]$. Additionally, by definition $\mathbb{E}[c] = c$.

19 DIS 10A (Variance)

19.1 Recall

Last time we looked at

Definition 19.1 (Expected Value)
The *expected value* of a random variable X, denoted $\mathbb{E}[X]$, is the anticipated average value of X. Mathematically,

$$\mathbb{E}[X] = \sum_{i \in \Omega} i \cdot \mathbb{P}[X = i].$$

We extend this definition slightly to incorporate for a random variable defined by a function $g(\cdot)$. If X is a RV, then so is $g(X)$.

Theorem 19.1 (Law of the Unconscious Statistician (LOTUS))

$$\mathbb{E}[g(X)] = \sum_{i \in \Omega} g(i)\mathbb{P}[X = i].$$

19.2 Variance

Today we introduce a new topic variance.

Definition 19.2 (Variance)
The *variance* of a random variable X measures how much on average the variable deviates from its expectation (the mean).

The main way we will compute variance is

$$\mathrm{Var}(X) = \mathbb{E}\left[X^2\right] - \mathbb{E}[X]^2.$$

If you have taken a statistics class before, you may remark that the standard deviation σ is defined as

$$\sigma(X) = \sqrt{\mathrm{Var}(X)}.$$

We looked at how scalars behaved with expectation. For variance, with a constant c,

$$\mathrm{Var}(cX) = c^2\mathrm{Var}(X)$$

and

$$\mathrm{Var}(X + c) = \mathrm{Var}(X).$$

19.3 Independence

If two RVs X and Y are independent, then

$$\mathrm{Var}(X + Y) = \mathrm{Var}(X) + \mathrm{Var}(Y).$$

20 DIS 10B (Covariance, More Distributions)

Last time we looked at variance, today we look at

20.1 Covariance

Covariance measures the association between two (or more) RVs X and Y.

Mathematically,

$$\text{Cov}(X,Y) = \mathbb{E}[XY] - \mathbb{E}[X]\,\mathbb{E}[Y].$$

We see that if

$$\text{Cov}(X,Y) : \begin{cases} < 0 & \implies \text{X and Y are inversely correlated} \\ = 0 & \implies \text{no correlation (does not imply independence)} \\ > 0 & \implies \text{X and Y are directly correlated} \end{cases}$$

20.2 Some general principles for Var and Cov

1. $\text{Cov}(X,Y) = \text{Cov}(Y,X)$
2. $\text{Cov}(X+Y,Z) = \text{Cov}(X,Z) + \text{Cov}(Y,Z)$
3. $\text{Cov}(aX,Y) = a\text{Cov}(X,Y)$
4. $\text{Var}(X) = \text{Cov}(X,X)$
5. $\text{Cov}(X,Y) = \mathbb{E}[XY] - \mathbb{E}[X]\,\mathbb{E}[Y]$
6. $\text{Var}(X+Y) = \text{Var}(X) + \text{Var}(Y) + 2\text{Cov}(X,Y)$

   A consequence of above is **when X and Y are independent**:

   - $\text{Cov}(X,Y) = 0$
   - $\mathbb{E}[XY] = \mathbb{E}[X]\,\mathbb{E}[Y]$
   - $\text{Var}(X+Y) = \text{Var}(X) + \text{Var}(Y)$

   README: When two variables are independent their covariance is 0 but just because their covariance is 0 doesn't mean they're independent

20.3 Poisson Distribution

We use this distribution when the data tends to fluctuate around some rate or "average". We denote this distribution as $X \sim \text{Poisson}(\lambda)$ where λ is the rate.

We claim

$$\mathbb{P}[X = i] = \frac{\lambda^i}{i!}e^{-\lambda}.$$

An interesting application is that Binomial distribution as $n \to \infty$ approaches Poisson.

Another interesting fact is that for independent X and Y where $X \sim \text{Poisson}(\lambda)$ and $Y \sim \text{Poisson}(\mu)$, then $X + Y \sim \text{Poisson}(\lambda + \mu)$.

20.4 EV/Variance Recap

X	$\mathbb{E}[X]$	$\text{Var}(X)$
Bernoulli(p)	p	$p(1-p)$
Binomial(n,p)	np	$np(1-p)$
Geometric(p)	$\frac{1}{p}$	$\frac{1-p}{p^2}$
Poisson(λ)	λ	λ

21 DIS 11A (Conditional PMFs and Expectation)

Recall that in a joint setting, the principal of marginal distribution revolved around the idea of

$$\mathbb{P}[X = a] = \sum_b \mathbb{P}[X = a|Y = b]\,\mathbb{P}[Y = b]\,.$$

We extend this idea (of total probability) now to expectation. For conditional expectation, we first require the conditional PMF.

Definition 21.1 (Conditional PMF)
The *conditional PMF* of a variable X describes how it behaves conditioned with respect to Y. Mathematically,

$$\mathbb{P}[X = x|Y = y] = \frac{\mathbb{P}[X = x, Y = y]}{\mathbb{P}[Y = y]}.$$

Definition 21.2 (Conditional Expectation)
The *conditional expectation* of X given $Y = y$ is defined as

$$\mathbb{E}[X|Y = y] = \sum_x x \cdot \mathbb{P}[X = x|Y = y]$$

If we apply the principal of marginal distribution to the conditional expectation above, we get

$$\sum_y \mathbb{E}[X|Y = y]\,\mathbb{P}[Y = y] = \mathbb{E}[X]\,.$$

The result above is known as **law of iterated expectation**. Formally, it's defined as

$$\mathbb{E}[X] = \mathbb{E}[\mathbb{E}[X|Y]]\,.$$

Remark: $\mathbb{E}[X|Y]$ is a function in terms of Y.

21.1 EV/Variance Recap

X	$\mathbb{E}[X]$	$\mathrm{Var}(X)$
Bernoulli(p)	p	$p(1-p)$
Binomial(n,p)	np	$np(1-p)$
Geometric(p)	$\frac{1}{p}$	$\frac{1-p}{p^2}$
Poisson(λ)	λ	λ
Uniform(a,b)	$\frac{a+b}{2}$	$\frac{b^2-a^2}{12}$