Mathematical Foundations

Building Proficiency with the Tools We Need

Prof. Antonio Khalil Moretti

SCIS 313: Data Structures and Algorithm Analysis

Spelman College

A Word of Encouragement

"What one fool can understand, another can."

What is Mathematical Maturity?

Mathematical maturity means:

1. **Not disengaging when you see symbols**
   - $\sum_i$, $\binom{n}{k}$, $E[X]$ are just shorthand
2. **Breaking problems into smaller pieces**
   - Can't solve it all at once? Solve one part.
3. **Being comfortable saying "I don't know yet"**
   - Then figuring it out step by step

These are skills you build, not talents you're born with.

Why Do We Need Math in CS?

Computer Science vs Programming

Most programmers don't do mathematical derivations daily.

But **computer science** is different from **programming**:

Programming:

- Writing code that works
- Using tools and libraries
- Debugging and testing

Computer Science:

- Understanding *why* algorithms work
- Proving correctness and efficiency
- Designing new algorithms and data structures
- Understanding fundamental limits

Today's Roadmap

1. **Floor & Ceiling, Modular Arithmetic**: Essential operations
2. **Logarithm Properties**: Quick review of what matters
3. **Sequences and Series**: Patterns in numbers
4. **Proof by Induction**: How to prove recursive algorithms correct
5. **Combinatorics**: Counting, permutations, combinations
6. **Sets**: Set notation and operations
7. **Probability**: Expected value and basic distributions

We'll go slowly with lots of examples.

Part 1: Floor and Ceiling Functions

Rounding in algorithm analysis

Floor function: $\lfloor x \rfloor$ rounds DOWN to nearest integer

Ceiling function: $\lceil x \rceil$ rounds UP to nearest integer

Examples:

- $\lfloor 3.7 \rfloor = 3, \quad \lceil 3.7 \rceil = 4$
- $\lfloor 5.1 \rfloor = 5, \quad \lceil 5.1 \rceil = 6$
- $\lfloor 8 \rfloor = 8, \quad \lceil 8 \rceil = 8$ (already an integer)
- $\lfloor -2.3 \rfloor = -3, \quad \lceil -2.3 \rceil = -2$ (careful with negatives!)

Where they appear in CS:

- Array indices: splitting array in half $\rightarrow \lfloor n/2 \rfloor$
- Binary tree height: $\lceil \log_2 n \rceil$
- Division in algorithms: $\lfloor a/b \rfloor$ for integer division

Part 2: Modular Arithmetic

The remainder operation

Modulo operation: $a \bmod n$ gives the remainder when a is divided by n

Examples:

- $17 \bmod 5 = 2$ (because $17 = 3 \times 5 + 2$)
- $20 \bmod 4 = 0$ (divides evenly)
- $7 \bmod 10 = 7$ (when $a < n$, result is just a)
- $100 \bmod 7 = 2$ (because $100 = 14 \times 7 + 2$)

Where it appears in CS:

- **Hash functions:** $h(k) = k \bmod m$ maps keys to table slots
- **Circular arrays:** $(i + 1) \bmod n$ wraps around
- **Even/odd testing:** $x \bmod 2 = 0$ means x is even
- **Checking divisibility:** $n \bmod k = 0$ means k divides n

Part 3: Logarithm Properties (Quick Review)

Remember: $\log_b x = y$ means $b^y = x$

Key properties:

1. $\log_b(xy) = \log_b x + \log_b y$ (multiply inside = add outside)
2. $\log_b(x^k) = k \log_b x$ (power inside = multiply outside)
3. $\log_b 1 = 0$ for any base b (because $b^0 = 1$)
4. **Change of base:** $\log_a x = \frac{\log_b x}{\log_b a}$

Why base doesn't matter in Big-O:

$$\log_2 n = \frac{\log_{10} n}{\log_{10} 2} \approx 3.32 \cdot \log_{10} n$$

The 3.32 is just a constant! So $\log_2 n = O(\log_{10} n) = O(\log n)$

Part 4: Sequences

Patterns in numbers

Sequence: An ordered list of numbers

Examples:

- $1, 2, 3, 4, 5, \ldots$ (counting numbers)
- $2, 4, 6, 8, 10, \ldots$ (even numbers)
- $1, 4, 9, 16, 25, \ldots$ (perfect squares: n^2)
- $1, 2, 4, 8, 16, \ldots$ (powers of 2: 2^n)

Notation: $a_1, a_2, a_3, \ldots, a_n$

- a_1 is the first term
- a_n is the n-th term

Arithmetic Sequences

Adding the same amount each time

Arithmetic sequence: Add a constant d each time

Example: $3, 7, 11, 15, 19, \ldots$

- Start at 3
- Add 4 each time
- Formula: $a_n = 3 + 4(n - 1)$

General formula:

$$a_n = a_1 + d(n - 1)$$

where a_1 is the first term and d is the common difference.

In CS: Not super common, but useful for understanding patterns.

Geometric Sequences

Multiplying by the same amount each time

Geometric sequence: Multiply by a constant r each time **Example:**

$2, 6, 18, 54, 162, \ldots$

- Start at 2
- Multiply by 3 each time
- Formula: $a_n = 2 \cdot 3^{n-1}$

General formula:

$$a_n = a_1 \cdot r^{n-1}$$

where a_1 is the first term and r is the common ratio. **In CS:** Very common! Doubling, halving, divide-and-conquer algorithms.

Example: Compound Interest

How money grows exponentially

Scenario: You invest \$1,000 at 7% annual return (average stock market return)

This is a geometric sequence with $r = 1.07$:

Year	Amount	Formula
0	\$1,000	$1000 \cdot (1.07)^0$
1	\$1,070	$1000 \cdot (1.07)^1$
5	\$1,403	$1000 \cdot (1.07)^5$
10	\$1,967	$1000 \cdot (1.07)^{10}$
20	\$3,870	$1000 \cdot (1.07)^{20}$
30	\$7,612	$1000 \cdot (1.07)^{30}$
40	\$14,974	$1000 \cdot (1.07)^{40}$

Compound Interest: What Does Exponential Growth Mean for Society?

Thought experiment: Two people are born in the same year.

Person A is born into wealth: Parents invest **$100,000** at birth

Person B: Parents invest **$100** at birth

Question: At 7% annual return, how much *passive income* does each person make per year at age 40?

Compound Interest: What Does Exponential Growth Mean for Society?

Thought experiment: Two people are born in the same year.

Person A is born into wealth: Parents invest **$100,000** at birth

Person B: Parents invest **$100** at birth

Question: At 7% annual return, how much *passive income* does each person make per year at age 40? **Answer:**

- Person A at age 40: $100{,}000 \cdot (1.07)^{40} = \$1{,}497{,}400$
  Annual passive income (7% of balance): **$104,818/year**
- Person B at age 40: $100 \cdot (1.07)^{40} = \$1{,}497$
  Annual passive income: **$105/year**

Person A earns $100K+ per year without working. Person B earns $105.

Compound interest means *capital* (what you have) rewarded over *labor* (what you do). This is exponential privilege.

Exponential Growth at Extreme Scales: Millionare vs Billionaire

Thought experiment: Two wealthy people invest their fortunes.

Millionaire: Invests **$1,000,000**

Billionaire: Invests **$1,000,000,000**

Question: At 7% annual return, how much passive income per year?

Exponential Growth at Extreme Scales: Millionare vs Billionaire

Thought experiment: Two wealthy people invest their fortunes.

Millionaire: Invests **$1,000,000**

Billionaire: Invests **$1,000,000,000**

Question: At 7% annual return, how much passive income per year?

- **Millionaire:** $1{,}000{,}000 \cdot 0.07 = \$70{,}000/\text{year}$
  (A comfortable middle-class salary, without working)
- **Billionaire:** $1{,}000{,}000{,}000 \cdot 0.07 = \$70{,}000{,}000/\text{year}$
  (Seventy million dollars per year, without working)

The billionaire makes 1000 millionaires' worth of passive income.

Visualization: Scroll to see wealth inequality drawn to scale:
`https://eattherichtextformat.github.io/1-pixel-wealth/`

The Fibonacci Sequence

A famous sequence defined recursively

Fibonacci sequence: Each number is the sum of the previous two

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, \ldots$$

Pattern:

$$F_1 = 1$$
$$F_2 = 1$$
$$F_n = F_{n-1} + F_{n-2} \quad \text{for } n \geq 3$$

This is a recurrence relation! (More on this soon)

Part 5: Series and Summation

Adding up sequences

Series: The sum of a sequence

Example: $1 + 2 + 3 + 4 + 5 = 15$

Summation notation: $\sum$ (capital Greek letter sigma)

$$\sum_{i=1}^{5} i = 1 + 2 + 3 + 4 + 5 = 15$$

How to read it:

- $i = 1$ at the bottom: "Start with $i = 1$"
- 5 at the top: "Stop when $i = 5$"
- i to the right: "Add up all the i values"

Just fancy notation for "add these up."

Summation Notation Examples

Example 1:

$$\sum_{i=1}^{4} i^2 =$$

Summation Notation Examples

Example 1:

$$\sum_{i=1}^{4} i^2 = 1^2 + 2^2 + 3^2 + 4^2 = 1 + 4 + 9 + 16 = 30$$

Summation Notation Examples

Example 1:

$$\sum_{i=1}^{4} i^2 = 1^2 + 2^2 + 3^2 + 4^2 = 1 + 4 + 9 + 16 = 30$$

Example 2:

$$\sum_{k=0}^{3} 2^k =$$

Summation Notation Examples

Example 1:

$$\sum_{i=1}^{4} i^2 = 1^2 + 2^2 + 3^2 + 4^2 = 1 + 4 + 9 + 16 = 30$$

Example 2:

$$\sum_{k=0}^{3} 2^k = 2^0 + 2^1 + 2^2 + 2^3 = 1 + 2 + 4 + 8 = 15$$

Summation Notation Examples

Example 1:

$$\sum_{i=1}^{4} i^2 = 1^2 + 2^2 + 3^2 + 4^2 = 1 + 4 + 9 + 16 = 30$$

Example 2:

$$\sum_{k=0}^{3} 2^k = 2^0 + 2^1 + 2^2 + 2^3 = 1 + 2 + 4 + 8 = 15$$

Example 3:

$$\sum_{j=1}^{n} j = 1 + 2 + 3 + \cdots + n =$$

Summation Notation Examples

Example 1:

$$\sum_{i=1}^{4} i^2 = 1^2 + 2^2 + 3^2 + 4^2 = 1 + 4 + 9 + 16 = 30$$

Example 2:

$$\sum_{k=0}^{3} 2^k = 2^0 + 2^1 + 2^2 + 2^3 = 1 + 2 + 4 + 8 = 15$$

Example 3:

$$\sum_{j=1}^{n} j = 1 + 2 + 3 + \cdots + n = \frac{n(n+1)}{2}$$

This last one is very important! We'll use it all the time.

Important Summation Formulas

These will save you time!

Memorize these (or at least know they exist):

1. **Sum of first n numbers:**

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} \approx \frac{n^2}{2}$$

2. **Sum of first n squares:**

$$\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6} \approx \frac{n^3}{3}$$

3. **Geometric series:**

$$\sum_{i=0}^{n} r^i = \frac{r^{n+1} - 1}{r - 1} \quad \text{for } r \neq 1$$

   Special case: $\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$

Part 6: Proofs - What Does Mean to Prove Something?

Proof: A *verification* of a *proposition* by a chain of *logical deductions*

Example proposition: "The sum $1 + 2 + 3 + \cdots + n = \frac{n(n+1)}{2}$"

How do we *know* this is true for **all** values of n?

- Can't test every value (infinite possibilities!)
- Need a logical argument that works for any n

Mathematical induction is a proof technique perfectly suited for statements about all natural numbers.

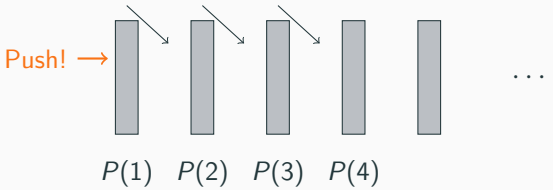
Key idea: Instead of proving infinitely many cases separately, we prove:

1. It works for the first case
2. *If* it works for case k, *then* it works for case $k + 1$

This creates a chain: $P(1) \Rightarrow P(2) \Rightarrow P(3) \Rightarrow \cdots$

Proof by Induction: The Domino Analogy

Mathematical induction is like dominoes:



To knock down all dominoes:

1. The first domino falls (**base case**)
2. Each domino knocks down the next (**inductive step**)

Why does this work?

The inductive step gives us:

- $P(1) \Rightarrow P(2)$
- $P(2) \Rightarrow P(3)$
- $P(3) \Rightarrow P(4)$
- ...

Combined with $P(1)$ true:

$$P(1) \xrightarrow{\text{step}} P(2) \xrightarrow{\text{step}} P(3) \xrightarrow{\text{step}} \cdots$$

This is our chain of logical deductions!

The Structure of an Induction Proof

To prove statement $P(n)$ is true for all $n \geq n_0$:

1. **Base Case:** Prove $P(n_0)$ is true
   "Knock down the first domino"
2. **Inductive Hypothesis:** Assume $P(k)$ is true for some arbitrary $k \geq n_0$
   "Suppose domino k has fallen..."
3. **Inductive Step:** Prove $P(k+1)$ is true using the assumption that $P(k)$ is true
   "...show that this knocks down domino $k+1$"
4. **Conclusion:** By mathematical induction, $P(n)$ is true for all $n \geq n_0$

Key insight: We're not assuming what we're trying to prove! We're proving the *implication*: "IF $P(k)$ is true THEN $P(k+1)$ is true."

Induction Example: Sum of First n Numbers

Claim: $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ for all $n \geq 1$

Proof: Base case ($n = 1$):

$$\sum_{i=1}^{1} i = 1 \quad \text{and} \quad \frac{1(1+1)}{2} = 1 \quad \checkmark$$

Induction Example: Sum of First n Numbers

Claim: $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ for all $n \geq 1$

Proof: Base case ($n = 1$):

$$\sum_{i=1}^{1} i = 1 \quad \text{and} \quad \frac{1(1+1)}{2} = 1 \quad \checkmark$$

Inductive hypothesis: Assume true for $n = k$: $\sum_{i=1}^{k} i = \frac{k(k+1)}{2}$

Induction Example: Sum of First n Numbers

Claim: $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ for all $n \geq 1$

Proof: Base case ($n = 1$):

$$\sum_{i=1}^{1} i = 1 \quad \text{and} \quad \frac{1(1+1)}{2} = 1 \quad \checkmark$$

Inductive hypothesis: Assume true for $n = k$: $\sum_{i=1}^{k} i = \frac{k(k+1)}{2}$

Inductive step: Show true for $n = k + 1$.

Need to verify that: $\sum_{i=1}^{k+1} i \stackrel{?}{=} \frac{(k+1)(k+2)}{2}$

Induction Example: Sum of First n Numbers

Claim: $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ for all $n \geq 1$

Proof: Base case ($n = 1$):

$$\sum_{i=1}^{1} i = 1 \quad \text{and} \quad \frac{1(1+1)}{2} = 1 \quad \checkmark$$

Inductive hypothesis: Assume true for $n = k$: $\sum_{i=1}^{k} i = \frac{k(k+1)}{2}$

Inductive step: Show true for $n = k + 1$.

Need to verify that: $\sum_{i=1}^{k+1} i \stackrel{?}{=} \frac{(k+1)(k+2)}{2}$

$$\begin{aligned}
\text{LHS: } \sum_{i=1}^{k+1} i &= \left(\sum_{i=1}^{k} i\right) + (k+1) \\
&= \frac{k(k+1)}{2} + (k+1) \quad \text{(by IH)} \\
&= \frac{k(k+1) + 2(k+1)}{2} \\
&= \frac{(k+1)(k+2)}{2} = \text{RHS} \quad \checkmark
\end{aligned}$$

Induction Example: Geometric Series

Claim: $\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$ for all $n \geq 0$

Proof:

Base case ($n = 0$):

$$\sum_{i=0}^{0} 2^i = 2^0 = 1 \quad \text{and} \quad 2^{0+1} - 1 = 2 - 1 = 1 \quad \checkmark$$

Induction Example: Geometric Series

Claim: $\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$ for all $n \geq 0$

Proof:

Base case ($n = 0$):

$$\sum_{i=0}^{0} 2^i = 2^0 = 1 \quad \text{and} \quad 2^{0+1} - 1 = 2 - 1 = 1 \quad \checkmark$$

Inductive hypothesis: Assume the formula holds for $n = k$:

$$\sum_{i=0}^{k} 2^i = 2^{k+1} - 1$$

Induction Example: Geometric Series

Claim: $\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$ for all $n \geq 0$

Proof:

Base case ($n = 0$):

$$\sum_{i=0}^{0} 2^i = 2^0 = 1 \quad \text{and} \quad 2^{0+1} - 1 = 2 - 1 = 1 \quad \checkmark$$

Inductive hypothesis: Assume the formula holds for $n = k$:

$$\sum_{i=0}^{k} 2^i = 2^{k+1} - 1$$

Inductive step: Show the formula holds for $n = k + 1$:

Need to show: $\sum_{i=0}^{k+1} 2^i \stackrel{?}{=} 2^{k+2} - 1$

Induction Example: Geometric Series

Claim: $\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$ for all $n \geq 0$

Proof:

Base case ($n = 0$):

$$\sum_{i=0}^{0} 2^i = 2^0 = 1 \quad \text{and} \quad 2^{0+1} - 1 = 2 - 1 = 1 \quad \checkmark$$

Inductive hypothesis: Assume the formula holds for $n = k$:

$$\sum_{i=0}^{k} 2^i = 2^{k+1} - 1$$

Inductive step: Show the formula holds for $n = k + 1$:

Need to show: $\sum_{i=0}^{k+1} 2^i \stackrel{?}{=} 2^{k+2} - 1$

$$\begin{aligned} \text{LHS: } \sum_{i=0}^{k+1} 2^i &= \left(\sum_{i=0}^{k} 2^i \right) + 2^{k+1} \\ &= (2^{k+1} - 1) + 2^{k+1} \quad \text{(by IH)} \\ &= 2 \cdot 2^{k+1} - 1 = 2^{k+2} - 1 = \text{RHS} \quad \checkmark \end{aligned}$$

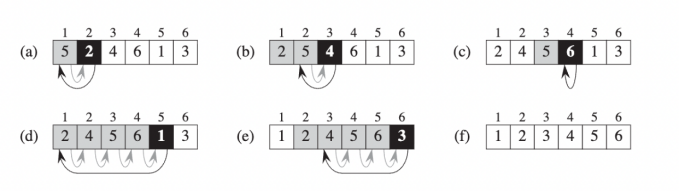
Application: Worst-Case Analysis of Insertion Sort

What is Insertion Sort?

Think of sorting playing cards:

- Start with one card in hand
- Pick up the next card
- Insert it in the correct position
- Repeat until all cards are sorted

At each step, the cards in your hand stay sorted.



Worst-Case Analysis: How Many Comparisons?

Step 1: Count comparisons per iteration

On iteration k, we insert the k-th element into $k-1$ already-sorted elements.
Worst case: element travels all the way to the front $\Rightarrow$ makes $\mathbf{k-1}$ comparisons.

Step 2: Sum across all n iterations

$$\text{Total comparisons} = 0 + 1 + 2 + \cdots + (n-1) = \sum_{i=0}^{n-1} i = \sum_{i=1}^{n-1} i$$

Worst-Case Analysis: How Many Comparisons?

Step 1: Count comparisons per iteration

On iteration k, we insert the k-th element into $k-1$ already-sorted elements.
Worst case: element travels all the way to the front $\Rightarrow$ makes $\mathbf{k-1}$ comparisons.

Step 2: Sum across all n iterations

$$\text{Total comparisons} = 0 + 1 + 2 + \cdots + (n-1) = \sum_{i=0}^{n-1} i = \sum_{i=1}^{n-1} i$$

Step 3: Apply our summation formula

We just proved $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$. Substituting $n-1$ for n:

$$\sum_{i=1}^{n-1} i = \frac{(n-1)n}{2}$$

Worst-Case Analysis: How Many Comparisons?

Step 1: Count comparisons per iteration

On iteration k, we insert the k-th element into $k-1$ already-sorted elements.
Worst case: element travels all the way to the front $\Rightarrow$ makes $\mathbf{k-1}$ comparisons.

Step 2: Sum across all n iterations

$$\text{Total comparisons} = 0 + 1 + 2 + \cdots + (n-1) = \sum_{i=0}^{n-1} i = \sum_{i=1}^{n-1} i$$

Step 3: Apply our summation formula

We just proved $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$. Substituting $n-1$ for n:

$$\sum_{i=1}^{n-1} i = \frac{(n-1)n}{2}$$

Insertion sort makes $\frac{(n-1)n}{2}$ comparisons in the worst case $\Rightarrow$ **insertion sort is** $O(n^2)$

Putting It All Together

Two proofs one insight:

1. **Proved by induction:** $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$
2. **Observed:** Insertion sort makes exactly $0 + 1 + \cdots + (n-1)$ comparisons in the worst case
3. **Concluded:** Worst-case cost $= \frac{(n-1)n}{2} \Rightarrow O(n^2)$

Key Insight: Algorithms are computational processes. Computation is inherently mathematical. Analyzing an algorithm means reasoning about what it *does*, and what it does is arithmetic.

This is why algorithm analysis requires mathematical proof. An algorithm's behavior is not empirical, it is computational.

Follow-Up Question

We showed insertion sort is $O(n^2)$ in the worst case.

Question

What is the **best case** for insertion sort, and can you use a similar argument to show it is $O(n)$?

Hint: Think about what happens when the input array is already sorted. How many comparisons does each iteration make?

Answer: Best Case is $O(n)$

Best case: Input array is already sorted in ascending order.

Step 1: Count comparisons per iteration

On iteration k: the element being inserted is already $\geq$ all elements to its left $\Rightarrow$ only **1 comparison** needed (we check the neighbor and immediately stop).

Step 2: Sum across all iterations

$$\text{Total comparisons} = \underbrace{1 + 1 + \cdots + 1}_{n-1 \text{ times}} = n - 1$$

Answer: Best Case is $O(n)$

Best case: Input array is already sorted in ascending order.

Step 1: Count comparisons per iteration

On iteration k: the element being inserted is already $\geq$ all elements to its left $\Rightarrow$ only **1 comparison** needed (we check the neighbor and immediately stop).

Step 2: Sum across all iterations

$$\text{Total comparisons} = \underbrace{1 + 1 + \cdots + 1}_{n-1 \text{ times}} = n - 1$$

Step 3: Conclude

No induction needed this time: the sum is trivially $n - 1$.

Insertion sort makes $n - 1$ comparisons in the best case $\Rightarrow$ **insertion sort is $\Theta(n)$ in the best case**

Answer: Best Case is $O(n)$

Best case: Input array is already sorted in ascending order.

Step 1: Count comparisons per iteration

On iteration k: the element being inserted is already $\geq$ all elements to its left $\Rightarrow$ only **1 comparison** needed (we check the neighbor and immediately stop).

Step 2: Sum across all iterations

$$\text{Total comparisons} = \underbrace{1 + 1 + \cdots + 1}_{n-1 \text{ times}} = n - 1$$

Step 3: Conclude

No induction needed this time: the sum is trivially $n - 1$.

Insertion sort makes $n - 1$ comparisons in the best case $\Rightarrow$ **insertion sort is $\Theta(n)$ in the best case**

Takeaway: Insertion sort is $\Theta(n)$ best case and $\Theta(n^2)$ worst case: performance depends heavily on input order.

Induction Beyond Formulas: Tiling with Trominoes

Can we tile a board with one square missing?

Tromino: An L-shaped piece made of 3 squares

Question: Can we tile a $2^n \times 2^n$ board with one square removed using L-trominoes?

Claim: For any $n \geq 1$, any $2^n \times 2^n$ board with one square missing can be perfectly tiled with L-trominoes.

Why induction?

The board size depends on n, and we can think recursively:

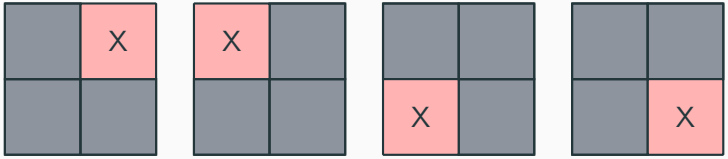
- A $2^{n+1} \times 2^{n+1}$ board is made of four $2^n \times 2^n$ boards
- If we can tile smaller boards, can we tile the larger one?

This is a perfect setup for induction!

Tromino Tiling: Base Case

Claim: Any $2^n \times 2^n$ board with one square missing can be tiled with L-trominoes.

Base case ($n = 1$): A 2×2 board with one square missing



A 2×2 board with one square missing **is exactly one L-tromino!** ✓

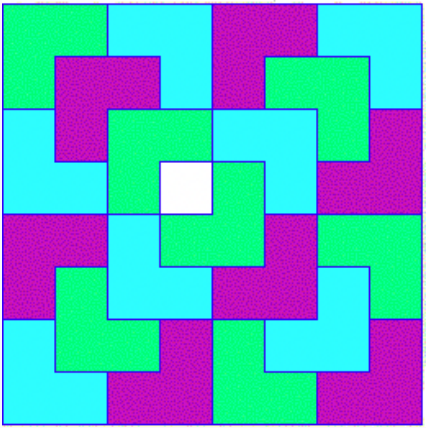
Tromino Tiling: Inductive Step

IH: Assume any $2^k \times 2^k$ board with one square missing can be tiled.

Inductive step: Consider a $2^{k+1} \times 2^{k+1}$ board with one square missing.

Strategy:

1. Divide into four $2^k \times 2^k$ quadrants
2. One quadrant (Q1) has the missing square
3. **Trick:** Place one tromino at center, covering one square from Q2, Q3, Q4!
4. Now each quadrant has exactly one square missing
5. By IH, tile each quadrant ✓



Strong Induction

A more powerful variant

Regular induction:
Assume $P(k)$ true, prove $P(k+1)$

Strong induction:
Assume $P(1), P(2), \ldots, P(k)$ all true, prove $P(k+1)$

Structure:

1. Base case(s)
2. IH: Assume $P(1), \ldots, P(k)$ all true
3. Prove $P(k+1)$

When to use:

- $P(k+1)$ depends on multiple previous cases
- Fibonacci: $F_{k+1} = F_k + F_{k-1}$
- Divide-and-conquer algorithms

Note: The tromino proof is strong induction. We used the hypothesis for $n = k$ to tile four quadrants!

When Do We Use Induction in CS?

Induction is perfect for proving:

- **Correctness of recursive algorithms**
  Example: Prove merge sort correctly sorts any array
- **Properties of recursive data structures**
  Example: A binary tree with n nodes has $n - 1$ edges
- **Summation formulas**
  Example: Runtime of nested loops with triangular pattern
- **Loop invariants**
  Example: At iteration i, array $[0..i - 1]$ is sorted

If it's recursive or involves n, consider induction!

Part 7: Combinatorics

The mathematics of counting

Combinatorics: The study of counting, arrangement, and selection

Why we care:

- Analyzing average-case complexity
- Understanding algorithm behavior on different inputs
- Probability calculations
- Understanding hash collisions

Key questions:

- How many ways can we arrange things? (**permutations**)
- How many ways can we select things? (**combinations**)
- If we have n items and k slots, what are the possibilities?

The Multiplication Principle

Fundamental counting rule

If you have:

- n_1 ways to do task 1
- n_2 ways to do task 2
- n_3 ways to do task 3

Then: Total number of ways $= n_1 \times n_2 \times n_3$

Example: How many outfits can you make?

- 5 shirts
- 3 pairs of pants
- 2 pairs of shoes

Answer: $5 \times 3 \times 2 = 30$ different outfits

In CS: Counting possible inputs, passwords, hash values, etc.

Permutations (Order Matters)

Permutation: An arrangement of objects where **order matters**

Example: How many ways can 3 people (A, B, C) stand in line?

ABC, ACB, BAC, BCA, CAB, CBA

Answer: $3 \times 2 \times 1 = 6$ ways

General formula: n objects in n positions

$$P(n) = n! = n \times (n-1) \times (n-2) \times \cdots \times 2 \times 1$$

Partial permutations: n objects, choose r positions

$$P(n, r) = \frac{n!}{(n-r)!} = n \times (n-1) \times \cdots \times (n-r+1)$$

Combinations (Order Doesn't Matter)

Combination: A selection of objects where **order doesn't matter**

Example: Choose 2 people from {A, B, C} for a committee

AB, AC, BC

Answer: 3 ways (note: AB = BA, they're the same committee)

General formula: Choose r objects from n objects

$$C(n, r) = \binom{n}{r} = \frac{n!}{r!(n-r)!}$$

Read as "n choose r" or "binomial coefficient"

Permutations vs Combinations

When to use which?

Ask yourself: Does order matter?

PERMUTATIONS (order matters)

Use when:

- Arranging people in line
- Assigning roles (president, VP, secretary)
- Creating passwords
- Race finishing positions

Formula: $P(n, r) = \frac{n!}{(n-r)!}$

COMBINATIONS (order doesn't)

Use when:

- Selecting committee members
- Choosing pizza toppings
- Forming teams
- Selecting subset of items

Formula: $C(n, r) = \binom{n}{r} = \frac{n!}{r!(n-r)!}$

Rule of thumb: If you can rearrange and it's still the same thing, use combinations!

The Binomial Coefficient

Definition: $\binom{n}{r} = \frac{n!}{r!(n-r)!}$ = number of ways to choose r items from n

Properties:

- $\binom{n}{0} = \binom{n}{n} = 1$ (one way to choose nothing or everything)
- $\binom{n}{1} = n$ (n ways to choose 1 item)
- $\binom{n}{r} = \binom{n}{n-r}$ (symmetry: choosing r = choosing which $n-r$ to exclude)
- $\binom{n}{r} = \binom{n-1}{r-1} + \binom{n-1}{r}$ (Pascal's identity)

Examples:

- $\binom{5}{2} = \frac{5!}{2!3!} = \frac{120}{2 \cdot 6} = 10$
- $\binom{10}{3} = \frac{10!}{3!7!} = \frac{10 \times 9 \times 8}{3 \times 2 \times 1} = 120$

Pascal's Triangle

Visualizing binomial coefficients

1	$n = 0$: $\binom{0}{0}$
1 1	$n = 1$: $\binom{1}{0}$, $\binom{1}{1}$
1 2 1	$n = 2$: $\binom{2}{0}$, $\binom{2}{1}$, $\binom{2}{2}$
1 3 3 1	$n = 3$
1 4 6 4 1	$n = 4$
1 5 10 10 5 1	$n = 5$

Pattern: Each number is the sum of the two numbers above it

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

Use: Quick reference for small values, basis for dynamic programming

The Binomial Theorem

A pattern for expanding polynomials $(x+y)^n$

Binomial Theorem:

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

Examples:

$$(x+y)^2 = \binom{2}{0}x^2 + \binom{2}{1}xy + \binom{2}{2}y^2 = x^2 + 2xy + y^2$$

$$\begin{aligned}(x+y)^3 &= \binom{3}{0}x^3 + \binom{3}{1}x^2y + \binom{3}{2}xy^2 + \binom{3}{3}y^3 \\ &= x^3 + 3x^2y + 3xy^2 + y^3\end{aligned}$$

Special case ($x = y = 1$):

$$(1+1)^n = 2^n = \sum_{k=0}^{n} \binom{n}{k}$$

This says: sum of all ways to choose subsets of n items $= 2^n$ (each item: in or out)

The Pigeonhole Principle

A simple but powerful idea

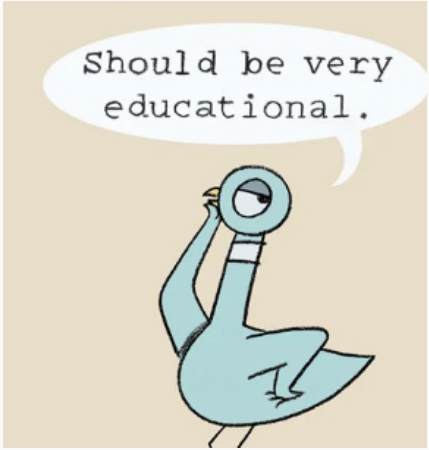
Pigeonhole Principle:
If you put $n+1$ pigeons into n holes, at least one hole must contain 2 or more pigeons.

Seems obvious, but very useful!

Example 1: In any group of 13 people, at least 2 were born in the same month.
(13 people, 12 months)

Example 2: In any group of 367 people, at least 2 share a birthday.
(367 people, 366 possible birthdays including leap day)

In CS: Hash collisions are unavoidable! If you have n slots and $n+1$ keys, you MUST have a collision.



Part 8: Sets

Collections of distinct objects

Set: An unordered collection of distinct elements

Notation:

- $A = \{1, 2, 3, 4\}$ (set with 4 elements)
- $\emptyset$ or $\{\}$ (empty set)
- $x \in A$ ("x is an element of A")
- $x \notin A$ ("x is not an element of A")
- $|A|$ (cardinality = number of elements in A)

Important: Order doesn't matter, duplicates don't count

- $\{1, 2, 3\} = \{3, 2, 1\} = \{1, 1, 2, 3\}$

Set Operations

Combining and comparing sets

Let $A = \{1, 2, 3, 4\}$ and $B = \{3, 4, 5, 6\}$

Union ($A \cup B$): All elements in A or B or both

$$A \cup B = \{1, 2, 3, 4, 5, 6\}$$

Intersection ($A \cap B$): Elements in both A and B

$$A \cap B = \{3, 4\}$$

Difference ($A - B$ or $A \setminus B$): Elements in A but not in B

$$A - B = \{1, 2\}$$

Subset ($A \subseteq B$): Every element of A is in B

$$\{1, 2\} \subseteq \{1, 2, 3, 4\}$$

Important Set Identities

Useful properties:

- **Commutative:** $A \cup B = B \cup A, \quad A \cap B = B \cap A$
- **Associative:** $(A \cup B) \cup C = A \cup (B \cup C)$
- **Distributive:** $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
- **De Morgan's Laws:**
  - $(A \cup B)^c = A^c \cap B^c$
  - $(A \cap B)^c = A^c \cup B^c$
- **Cardinality:** $|A \cup B| = |A| + |B| - |A \cap B|$

In CS: Sets are fundamental data structures. Understanding set operations helps with algorithms on collections.

Part 9: Probability Basics

Probability: Measures likelihood of events, ranges from 0 to 1

Key concepts:

- **Sample space** (S or Ω): Set of all possible outcomes
- **Event** (E): A subset of the sample space
- **Probability of event** E:

$$P(E) = \frac{\text{number of favorable outcomes}}{\text{total number of outcomes}}$$

Example: Rolling a six-sided die

- Sample space: $S = \{1, 2, 3, 4, 5, 6\}, \quad |S| = 6$
- Event "roll even": $E = \{2, 4, 6\}, \quad |E| = 3$
- Probability: $P(E) = \frac{3}{6} = \frac{1}{2}$

Random Variables

Random variable: A function that assigns a number to each outcome

Example: Rolling two dice

- Sample space: All pairs (i, j) where $i, j \in \{1, 2, 3, 4, 5, 6\}$, so $|S| = 36$
- Random variable X = sum of the two dice
- X can take values $2, 3, 4, \ldots, 12$

Probability distribution:

X	2	3	4	5	6	$\cdots$
$P(X)$	$\frac{1}{36}$	$\frac{2}{36}$	$\frac{3}{36}$	$\frac{4}{36}$	$\frac{5}{36}$	$\cdots$

Note: $\sum_x P(X = x) = 1$ (probabilities must sum to 1)

Expected Value

The "average" outcome

Expected value (or **expectation**): The average value you'd get if you repeated the experiment many times

Formula:

$$E[X] = \sum_x x \cdot P(X = x)$$

Sum over all possible values: (value) $\times$ (probability of that value)

Example 1: Rolling a fair die

$$E[X] = 1 \cdot \frac{1}{6} + 2 \cdot \frac{1}{6} + 3 \cdot \frac{1}{6} + 4 \cdot \frac{1}{6} + 5 \cdot \frac{1}{6} + 6 \cdot \frac{1}{6} = \frac{21}{6} = 3.5$$

Expected Value Examples

Example 2: Flipping a fair coin (heads = 1, tails = 0)

$$E[X] = 1 \cdot \frac{1}{2} + 0 \cdot \frac{1}{2} = \frac{1}{2}$$

Example 3: Sum of two dice

- We could compute: $E[X] = 2 \cdot P(X = 2) + 3 \cdot P(X = 3) + \cdots + 12 \cdot P(X = 12)$
- Or use **linearity of expectation:** $E[D_1 + D_2] = E[D_1] + E[D_2]$
- Each die has expected value 3.5, so $E[X] = 3.5 + 3.5 = 7$

Linearity of expectation: $E[aX + bY] = aE[X] + bE[Y]$

This works even if X and Y are not independent!

Expected Value: Two Dice

Sum of two fair dice

Two dice $\rightarrow 6 \times 6 = 36$ equally likely pairs.

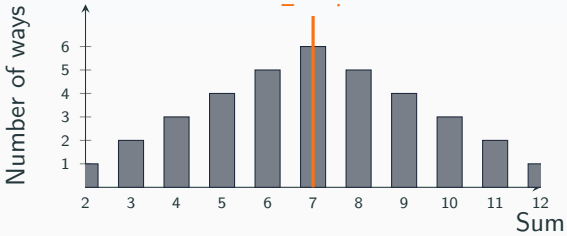
Each cell is the sum of its row and column.

Orange = sum of 7 (6 ways out of 36).

D1 \ D2	1	2	3	4	5	6
1	2	3	4	5	6	**7**
2	3	4	5	6	**7**	8
3	4	5	6	**7**	8	9
4	5	6	**7**	8	9	10
5	6	**7**	8	9	10	11
6	**7**	8	9	10	11	12

$$E[\text{sum}] = \frac{1}{36} \sum_{\text{all 36 cells}} \text{value} = \frac{252}{36} = 7$$

Alternatively: $E[\text{sum}] = \sum_{s=2}^{12} s \cdot P(\text{sum} = s)$



7 is the most likely sum *and* the expected value.

The distribution is **symmetric** around 7.

Again: $E = 7$ is not guaranteed on any single roll.

Why List All 36 Outcomes?

Key insight: Not all sums are equally likely!

- **Sum = 2:** only **one** way

$$(1,1) \qquad P(\text{sum} = 2) = \frac{1}{36}$$

- **Sum = 7:** **six** ways

$$(1,6),(2,5),(3,4),(4,3),(5,2),(6,1) \qquad P(\text{sum} = 7) = \frac{6}{36} = \frac{1}{6}$$

- **Sum = 12:** only **one** way

$$(6,6) \qquad P(\text{sum} = 12) = \frac{1}{36}$$

Different sums $\Rightarrow$ different probabilities.

Two Ways to Compute Expected Value

Both methods give the same answer — but one scales better.

1. **Enumerate the full sample space** (all 36 outcomes):

$$E[\text{sum}] = \frac{1}{36} \sum_{\text{all 36 pairs}} (\text{sum}) = 7$$

2. **Group by sum value** (compress probabilities):

$$E[\text{sum}] = 2 \cdot \frac{1}{36} + 3 \cdot \frac{2}{36} + \cdots + 7 \cdot \frac{6}{36} + \cdots + 12 \cdot \frac{1}{36} = 7$$

Why avoid approach 1 in general?

Sample spaces can grow **exponentially** (e.g., 10 dice $\rightarrow 6^{10} \approx 60$ million outcomes).

Expected Value vs. Median vs. Mode (Three Measures of "Center")

- **Mean (Expected Value):** $E[X] = \sum_i x_i \cdot P(X = x_i)$
  - Weighted average of all possible values
  - **Sensitive to outliers/extreme values**
- **Median:** The middle value when outcomes are ordered
  - 50% of probability mass below, 50% above
  - **Robust to outliers**
- **Mode:** The most likely outcome (highest probability)
  - Can have multiple modes (bimodal, multimodal distributions)

When are they all equal?

- In a **symmetric, unimodal** distribution (e.g., normal dist, symmetric dice sum)
- For the two-dice example: Mean = Median = Mode = 7

Example where they differ:

- Income distribution: Mean > Median (billionaires pull mean up)
- Median income is often more representative than mean income

Common Probability Distributions

Uniform distribution: All outcomes equally likely

- Example: Fair die, random array shuffle
- $P(X = k) = \frac{1}{n}$ for each of n outcomes

Geometric distribution: Number of trials until first success

- Example: Flipping coins until you get heads
- $P(X = k) = (1 - p)^{k-1}p$ where p is success probability
- Expected value: $E[X] = \frac{1}{p}$

Binomial distribution: Number of successes in n trials

- Example: Number of heads in 10 coin flips
- $P(X = k) = \binom{n}{k} p^k (1 - p)^{n-k}$
- Expected value: $E[X] = np$

Why Probabilities Sum to 1

Connecting combinatorics and probability

Key fact: For any probability distribution, $\sum_x P(X = x) = 1$

Example 1: Binomial distribution Sum of all probabilities:

$$\sum_{k=0}^{n} \binom{n}{k} p^k (1-p)^{n-k} = (p + (1-p))^n = 1^n = 1 \quad \checkmark$$

(By the binomial theorem!)

Example 2: Geometric distribution

$$\sum_{k=1}^{\infty} (1-p)^{k-1} p = p \sum_{k=0}^{\infty} (1-p)^k = p \cdot \frac{1}{1-(1-p)} = p \cdot \frac{1}{p} = 1 \quad \checkmark$$

(By the geometric series formula!)

What We Covered

Today's mathematical toolkit:

1. **Floor/Ceiling & Modular Arithmetic** — Rounding and remainders
2. **Logarithm properties** — Key facts for algorithm analysis
3. **Sequences and Series** — Patterns and summations
4. **Proof by Induction** — Proving recursive statements
5. **Combinatorics** — Counting, permutations, combinations
6. **Sets** — Collections and operations
7. **Probability** — Expected value and distributions

Key formulas:

- $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}, \quad \sum_{i=0}^{n} 2^i = 2^{n+1} - 1$
- $P(n, r) = \frac{n!}{(n-r)!}, \quad \binom{n}{r} = \frac{n!}{r!(n-r)!}$
- $E[X] = \sum_x x \cdot P(X = x)$

What's Next

We'll use this to do the following:

- Analyze recursive algorithms
- Solve recurrence relations
- Examples: merge sort, binary search, dynamic programming

Homework: Practice problems on Canvas

- Induction proofs
- Combinatorics calculations
- Expected value problems

Office hours: Come by if anything is confusing!