Mathematical Logic

Part One

Question: How do we formalize the logic we've been using in our proofs?

Where We're Going

- **Propositional Logic** (Today)
  - Basic logical connectives.
  - Truth tables.
  - Logical equivalences.
- **First-Order Logic** (Wednesday/Friday)
  - Reasoning about properties of multiple objects.

Propositional Logic

A ***proposition*** is a statement that is, by itself, either true or false.

Some Sample Propositions

- Puppies are cuter than kittens.
- Kittens are cuter than puppies.
- Usain Bolt can outrun everyone in this room.
- CS103 is useful for cocktail parties.
- This is the last entry on this list.

More Propositions

- Got kiss myself
- I'm so pretty
- I'm too hot
- Called a policeman and a fireman
- Made a dragon want to retire, man

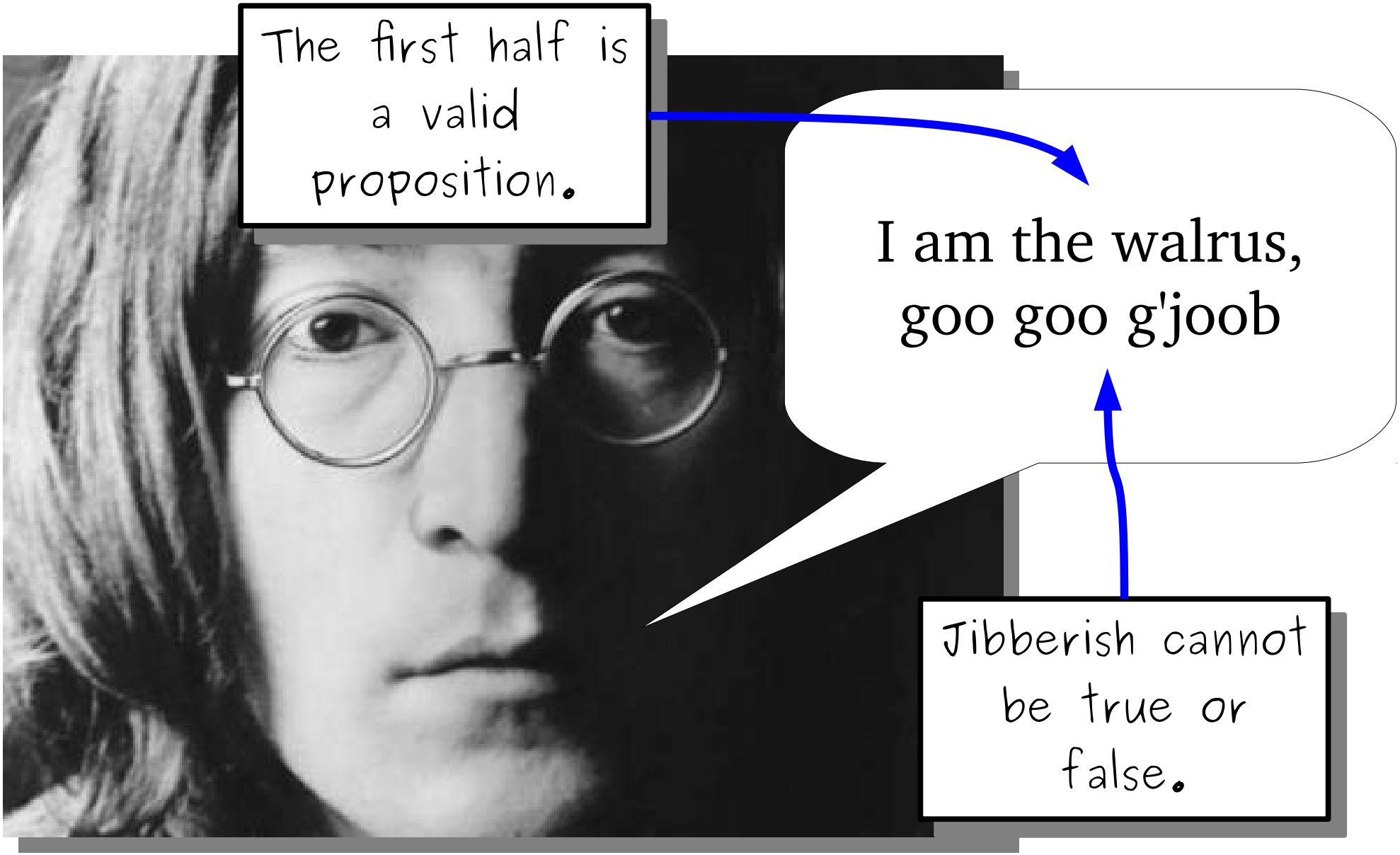
Things That Aren't Propositions

Commands cannot be true or false.

Things That Aren't Propositions

Questions cannot be true or false.

Things That Aren't Propositions



Propositional Logic

- ***Propositional logic*** is a mathematical system for reasoning about propositions and how they relate to one another.
- Every statement in propositional logic consists of ***propositional variables*** combined via ***propositional connectives***.
  - Each variable represents some proposition, such as "You liked it" or "You should have put a ring on it."
  - Connectives encode how propositions are related, such as "If you liked it, then you should have put a ring on it."

Propositional Variables

- Each proposition will be represented by a ***propositional variable***.
- Propositional variables are usually represented as lower-case letters, such as *p, q, r, s,* etc.
- Each variable can take one one of two values: true or false.

Propositional Connectives

- **Logical NOT: $\neg p$**
  - Read "***not*** *p*"
  - $\neg p$ is true if and only if p is false.
  - Also called **logical negation**.
- **Logical AND: $p \wedge q$**
  - Read "*p* ***and*** *q*."
  - $p \wedge q$ is true if both p and q are true.
  - Also called **logical conjunction**.
- **Logical OR: $p \vee q$**
  - Read "*p* ***or*** *q*."
  - $p \vee q$ is true if at least one of *p* or *q* are true (inclusive OR)
  - Also called **logical disjunction**.

Truth Tables

- A ***truth table*** is a table showing the truth value of a propositional logic formula as a function of its inputs.
- Useful for several reasons:
  - Formally defining what a connective "means."
  - Deciphering what a complex propositional formula means.

The Truth Table Tool

Summary of Important Points

- The `∨` operator is an *inclusive* “or.” It's true if at least one of the operands is true.
  - Similar to the || operator in C, C++, Java and the `or` operator in Python.
- If we need an exclusive “or” operator, we can build it out of what we already have.

Mathematical Implication

Implication

- The → connective is used to represent implications.
  - Its technical name is the ***material conditional*** operator.
- What is its truth table?

Why This Truth Table?

- The truth values of the $\rightarrow$ are the way they are because they're *defined* that way.
- The intuition:
  - We want $p \rightarrow q$ to mean "whenever p is true, q is true as well."
  - The only way this *doesn't* happen is if p is true and q is false.
  - In other words, $p \rightarrow q$ should be true whenever $\neg(p \land \neg q)$ is true.
  - What's the truth table for $\neg(p \land \neg q)$?

Truth Table for Implication

p	q	$p \to q$
F	F	T
F	T	T
T	F	F
T	T	T

The <u>only way</u> for $p \to q$ to be false is for p to be true and q to be false. Otherwise, $p \to q$ is <u>by definition</u> true.

The Biconditional Operator

The Biconditional Connective

- The biconditional connective $\leftrightarrow$ is used to represent a two-directional implication.
- Specifically, $p \leftrightarrow q$ means that p implies q and q implies p.
- What should its truth table look like?

Biconditionals

- The ***biconditional*** connective $p \leftrightarrow q$ is read "p if and only if q."
- Here's its truth table:

p	q	$p \leftrightarrow q$
F	F	T
F	T	F
T	F	F
T	T	T

One interpretation of $\leftrightarrow$ is to think of it as equality: the two propositions must have equal truth values.

True and False

- There are two more “connectives” to speak of: true and false.
  - The symbol $\top$ is a value that is always true.
  - The symbol $\bot$ is value that is always false.
- These are often called connectives, though they don't connect anything.
  - (Or rather, they connect zero things.)

Operator Precedence

- How do we parse this statement?

$$\neg x \to y \lor z \to x \lor y \land z$$

- Operator precedence for propositional logic:

$$\neg$$

$$\land$$

$$\lor$$

$$\to$$

$$\leftrightarrow$$

- All operators are right-associative.
- We can use parentheses to disambiguate.

Operator Precedence

- How do we parse this statement?

$$\neg x \to y \lor z \to x \lor y \land z$$

- Operator precedence for propositional logic:

$$\neg$$

$$\land$$

$$\lor$$

$$\to$$

$$\leftrightarrow$$

- All operators are right-associative.
- We can use parentheses to disambiguate.

Operator Precedence

- How do we parse this statement?

$$(\neg x) \to ((y \lor z) \to (x \lor (y \land z)))$$

- Operator precedence for propositional logic:

$$\neg$$

$$\land$$

$$\lor$$

$$\to$$

$$\leftrightarrow$$

- All operators are right-associative.
- We can use parentheses to disambiguate.

Operator Precedence

- The main points to remember:
  - $\neg$ binds to whatever immediately follows it.
  - $\wedge$ and $\vee$ bind more tightly than $\rightarrow$.
- We will commonly write expressions like $p \wedge q \rightarrow r$ without adding parentheses.
- For more complex expressions, we'll try to add parentheses.
- Confused? Just ask!

Time-Out for Announcements!

Major Career Fair

- The Computer Forum Career Fair is this **Wednesday, September 30** from **11:00AM – 4:00PM** on the lawn between Gates and Packard.
- One of the biggest career fairs of the year – highly recommended if you're looking for a full-time job or an internship for the summer.
- Bring lots of copies of your resumes, wear nice but not super formal clothes, and have fun!
- Freshmen, sophomores: sorry this is so early in the year!

WiCS Welcome Back Dinner

- Stanford WiCS (Women in Computer Science) is holding a welcome back dinner tonight at 5:30PM in the Women's Community Center.
  - For freshmen – that's right behind Tresidder and the Old Union.
- Highly recommended, and everyone is welcome!

A Note on Problem Sets

- Whenever possible, we try to avoid problem set questions that involve Cruel and Unusual Math.
- As a hint for the Piano Tuning problem on Problem Set One: if you find yourself needing to simplify the expression $(2k + 1)^{12}$, you're missing a *much* simpler proof approach.
- Try solving this without modifying the proof that the square root of two is irrational.

Your Questions

"If we only started CS at Stanford (with the 106 classes), what is the best way to gain more experience to obtain a tech internship? What are good classes to take or programming languages to know to adequately prepare for such internships?"

"I have an upcoming technical interview and I have no idea how to prepare - what should I do?"

There are a lot of programs out there specifically aimed at freshmen and sophomores. Check out Explore Microsoft, Google Summer Engineering Practicum, and Facebook U as starting points.

Attend the big career fair this Wednesday. Even with just CS106, you're actually qualified for a lot of the internships out there. You'd be surprised!

We're currently offering a course, CS9, about navigating the job market. It meets on Tuesdays right before CS103A, so feel free to sign up.

Don't worry too much about specific programming languages. Quite honestly, that's the easy part. You can pick that up on your own time.

Finally, for good prep, pick up a copy of "Cracking the Coding Interview."

“If we put in effort into the psets but are terrible at writing proofs, is there any way we can still make an A in the class? (Basically how I'm feeling on pset 1)”

Don't worry! We're just getting started and this is a skill you'll pick up with practice. This stuff is tricky, but it's a skill that you can improve with practice. We're here to help out, so please feel free to stop by office hours or ask questions on Piazza. Even better, do that and get a group to work on the problem sets with.

“What do you think of the Symbolic Systems major?”

It's cool! You get less depth in each of the areas you explore, but you'll get a much more cohesive picture of the overlap between those areas.

Also consider exploring CS+X, possibly CS+Linguistics or CS+Philosophy. Those are less structured options, but might give you more of what you're looking for.

Back to CS103!

Recap So Far

- A ***propositional variable*** is a variable that is either true or false.
- The ***propositional connectives*** are
  - Negation: $\neg p$
  - Conjunction: $p \land q$
  - Disjunction: $p \lor q$
  - Implication: $p \to q$
  - Biconditional: $p \leftrightarrow q$
  - True: $\top$
  - False: $\bot$

Translating into Propositional Logic

Some Sample Propositions

a: I will be awake this evening

b: There is a lunar eclipse this evening

c: There are no clouds in the sky this evening

d: I will see the lunar eclipse

"I won't see the lunar eclipse if I'm not awake this evening."

$$\neg a \rightarrow \neg d$$

“p if q”

translates to

$$q \rightarrow p$$

It does *not* translate to

$$p \rightarrow q$$

Some Sample Propositions

a: I will be awake this evening

b: There is a lunar eclipse this evening

c: There are no clouds in the sky this evening

d: I will see the lunar eclipse

"If I will be awake this evening, but it's cloudy outside, I won't see the lunar eclipse.

$$a \land \neg c \to \neg d$$

“p, but q”

translates to

$$p \wedge q$$

The Takeaway Point

- When translating into or out of propositional logic, be very careful not to get tripped up by nuances of the English language.
  - In fact, this is one of the reasons we have a symbolic notation in the first place!
- Many prepositional phrases lead to counterintuitive translations; make sure to double-check yourself!

Propositional Equivalences

Quick Question

What would I have to show you to convince you that the statement $p \wedge q$ is false?

Quick Question

What would I have to show you to convince you that the statement $p \vee q$ is false?

De Morgan's Laws

- Using truth tables, we concluded that

$$\neg(p \wedge q)$$

  is equivalent to

$$\neg p \vee \neg q$$

- We also saw that

$$\neg(p \vee q)$$

  is equivalent to

$$\neg p \wedge \neg q$$

- These two equivalences are called ***De Morgan's Laws***.

De Morgan's Laws in Code

- Pro tip: Don't write this:

```
if (!(p() && q())) {
    /* … */
}
```

- Write this instead:

```
if (!p() || !q()) {
    /* … */
}
```

Logical Equivalence

- Because $\neg(p \land q)$ and $\neg p \lor \neg q$ have the same truth tables, we say that they're ***equivalent*** to one another.
- We denote this by writing

$$\neg(p \land q) \equiv \neg p \lor \neg q$$

- The $\equiv$ symbol is not a connective.
  - The statement $\neg(p \land q) \leftrightarrow (\neg p \lor \neg q)$ is a propositional formula. If you plug in different values of p and q, it will evaluate to a truth value. It just happens to evaluate to true every time.
  - The statement $\neg(p \land q) \equiv \neg p \lor \neg q$ means "these two formulas have exactly the same truth table."
- In other words, the notation $\varphi \equiv \psi$ means "φ and ψ always have the same truth values, regardless of how the variables are assigned."

An Important Equivalence

- Earlier, we talked about the truth table for $p \to q$. We chose it so that

$$\boldsymbol{p \to q \quad \equiv \quad \neg(p \land \neg q)}$$

- Later on, this equivalence will be incredibly useful:

$$\boldsymbol{\neg(p \to q) \quad \equiv \quad p \land \neg q}$$

Another Important Equivalence

- Here's a useful equivalence. Start with

$$p \to q \equiv \neg(p \land \neg q)$$

- By De Morgan's laws:

$$\begin{aligned} p \to q &\equiv \neg(p \land \neg q) \\ &\equiv \neg p \lor \neg\neg q \\ &\equiv \neg p \lor q \end{aligned}$$

- Thus $p \to q \equiv \neg p \lor q$

Another Important Equivalence

- Here's a useful equivalence. Start with

$$p \to q \equiv \neg(p \land \neg q)$$

- By De Morgan's laws:

$$\begin{aligned} p \to q &\equiv \neg(p \land \neg q) \\ &\equiv \neg p \lor \neg\neg q \\ &\equiv \neg p \lor \end{aligned}$$

- Thus $p \to q \equiv \neg p \lor q$

If p is false, then $\neg p \lor q$ is true. If p is true, then q has to be true for the whole expression to be true.

One Last Equivalence

The Contrapositive

- The contrapositive of the statement

$$p \rightarrow q$$

is the statement

$$\neg q \rightarrow \neg p$$

- These are logically equivalent, which is why proof by contradiction works:

$$\boldsymbol{p \rightarrow q \quad \equiv \quad \neg q \rightarrow \neg p}$$

Why All This Matters

Why All This Matters

- Suppose we want to prove the following statement:

  "If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

Why All This Matters

- Suppose we want to prove the following statement:

  "If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$x + y = 16 \rightarrow x \geq 8 \lor y \geq 8$$

Why All This Matters

- Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$x < 8 \land y < 8 \rightarrow x + y \neq 16$$

"If $x < 8$ and $y < 8$, then $x + y \neq 16$"

Theorem: If $x + y = 16$, then $x \geq 8$ or $y \geq 8$.

Proof: By contrapositive. We prove that if $x < 8$ and $y < 8$, then $x + y \neq 16$. To see this, note that

$$\begin{aligned} x + y &< 8 + y \\ &< 8 + 8 \\ &= 16 \end{aligned}$$

This means that $x + y < 16$, so $x + y \neq 16$, which is what we needed to show. ■

Why All This Matters

- Suppose we want to prove the following statement:

  "If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$x + y = 16 \rightarrow x \geq 8 \lor y \geq 8$$

Why All This Matters

- Suppose we want to prove the following statement:

  "If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$\neg(x + y = 16 \rightarrow x \geq 8 \lor y \geq 8)$$

Why All This Matters

- Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$x + y = 16 \land x < 8 \land y < 8$$

"$x + y = 16$, but $x < 8$ and $y < 8$."

Theorem: If $x + y = 16$, then $x \geq 8$ or $y \geq 8$.

Proof: Assume for the sake of contradiction that $x + y = 16$, but $x < 8$ and $y < 8$. Then

$$\begin{aligned} x + y &< 8 + y \\ &< 8 + 8 \\ &= 16 \end{aligned}$$

So $x + y < 16$, contradicting that $x + y = 16$. We have reached a contradiction, so our assumption must have been wrong. Therefore if $x + y = 16$, then $x \geq 8$ or $y \geq 8$. ■

Why This Matters

- Propositional logic is a tool for reasoning about how various statements affect one another.
- To better understand how to prove a result, it often helps to translate what you're trying to prove into propositional logic first.
- That said, propositional logic isn't expressive enough to capture all statements. For that, we need something more powerful.