Logic: The Big Picture

Logic is a tool for formalizing reasoning. There are lots of different logics:

- probabilistic logic: for reasoning about probability
- temporal logic: for reasoning about time (and programs)
- epistemic logic: for reasoning about knowledge

The simplest logic (on which all the rest are based) is *propositional logic*. It is intended to capture features of arguments such as the following:

> *Borogroves are mimsy whenever it is brillig. It is now brillig and this thing is a borogrove. Hence this thing is mimsy.*

Propositional logic is good for reasoning about

- conjunction, negation, implication ("if . . . then . . . ")

Amazingly enough, it is also useful for

- circuit design
- program verification

Propositional Logic: Syntax

To formalize the reasoning process, we need to restrict the kinds of things we can say. Propositional logic is particularly restrictive. The *syntax* of propositional logic tells us what are legitimate formulas.

We start with *primitive propositions*, basic statements like

- It is now brillig
- This thing is mimsy
- It's raining in San Francisco
- n is even

We can then form more complicated *compound propositions* using connectives like:

- $\neg$: not
- $\wedge$: and
- $\vee$: or
- $\Rightarrow$: implies

MCS uses English (NOT, AND, OR, IMPLIES). I'll stick to the standard mathematical notation.

Examples:

- $\neg P$: it is not the case that P
- $P \wedge Q$: P and Q
- $P \vee Q$: P or Q
- $P \Rightarrow Q$: P implies Q (if P then Q)

Typical formula:

$$P \wedge (\neg P \Rightarrow (Q \Rightarrow (R \vee P)))$$

Wffs

Formally, we define *well-formed formulas* (*wffs* or just *formulas*) inductively:

1. Every primitive proposition $P, Q, R, \ldots$ is a wff
2. If A is a wff, so is $\neg A$
3. If A and B are wffs, so are $A \wedge B$, $A \vee B$, and $A \Rightarrow B$,

Disambiguating Wffs

We use parentheses to disambiguate wffs:

- $P \vee Q \wedge R$ can be either $(P \vee Q) \wedge R$ or $P \vee (Q \wedge R)$

Mathematicians are lazy, so there are standard rules to avoid putting in parentheses.

- In arithmetic expressions, $\times$ binds more tightly than $+$, so $3 + 2 \times 5$ means $3 + (2 \times 5)$
- In wffs, here is the precedence order:
  - $\neg$
  - $\wedge$
  - $\vee$
  - $\Rightarrow$
  - $\Leftrightarrow$

  Thus, $P \vee Q \wedge R$ is $P \vee (Q \wedge R)$;
  $P \vee \neg Q \wedge R$ is $P \vee ((\neg Q) \wedge R)$
  $P \vee \neg Q \Rightarrow R$ is $(P \vee (\neg Q)) \Rightarrow R$
- With two or more instances of the same binary connective, evaluate left to right:
  $P \Rightarrow Q \Rightarrow R$ is $(P \Rightarrow Q) \Rightarrow R$

Translating English to Wffs

To analyze reasoning, we have to be able to translate English to wffs.

Consider the following sentences:

1. Bob doesn't love Alice
2. Bob loves Alice and loves Ann
3. Bob loves Alice or Ann
4. Bob loves Alice but doesn't love Ann
5. If Bob loves Alice then he doesn't love Ann

First find appropriate primitive propositions:

- P: Bob loves Alice
- Q: Bob loves Ann

Then translate:

1. $\neg P$
2. $P \wedge Q$
3. $P \vee Q$
4. $P \wedge \neg Q$ (note: "but" becomes "and")
5. $P \Rightarrow \neg Q$

Evaluating Formulas

Given a formula, we want to decide if it is true or false.

How do we deal with a complicated formula like:

$$P \wedge (\neg P \Rightarrow (Q \Rightarrow (R \vee P)))$$

The truth or falsity of such a formula depends on the truth or falsity of the primitive propositions that appear in it. We use *truth tables* to describe how the basic connectives ($\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$) work.

Truth Tables

For $\neg$:

P	$\neg P$
T	F
F	T

For $\wedge$:

P	Q	$P \wedge Q$
T	T	T
T	F	F
F	T	F
F	F	F

For $\vee$:

P	Q	$P \vee Q$
T	T	T
T	F	T
F	T	T
F	F	F

Truth Tables

For $\neg$:

P	$\neg P$
T	F
F	T

For $\wedge$:

P	Q	$P \wedge Q$
T	T	T
T	F	F
F	T	F
F	F	F

For $\vee$:

P	Q	$P \vee Q$
T	T	T
T	F	T
F	T	T
F	F	F

This means $\vee$ is *inclusive* or, not *exclusive* or.

Exclusive Or

What's the truth table for "exclusive or"?

P	Q	$P \oplus Q$
T	T	F
T	F	T
F	T	T
F	F	F

$P \oplus Q$ is equivalent to $(P \wedge \neg Q) \vee (\neg P \wedge Q)$

P	Q	$\neg P$	$\neg Q$	$P \wedge \neg Q$	$Q \wedge \neg P$	$(P \wedge \neg Q) \vee (\neg P \wedge Q)$
T	T	F	F	F	F	F
T	F	F	T	T	F	T
F	T	T	F	F	T	T
F	F	T	T	F	F	F

Truth Table for Implication

For $\Rightarrow$:

P	Q	$P \Rightarrow Q$
T	T	
T	F	
F	T	
F	T	

Why is this right? What should the truth value of $P \Rightarrow Q$ be when P is false?

- Despite what the book says, implications with false hypotheses come up a lot:
  - If he hadn't been drunk (he was), he wouldn't have had the accident
  - This isn't vacuously true!
- This choice is mathematically convenient
- As long as Q is true when P is true, then $P \Rightarrow Q$ will be true no matter what.

How many possible truth tables are there with two primitive propositions?

P	Q	?
T	T	
T	F	
F	T	
F	F	

(a) 16?
(b) 32?
(c) no clue?

How many possible truth tables are there with two primitive propositions?

P	Q	?
T	T	
T	F	
F	T	
F	F	

(a) 16?
(b) 32?
(c) no clue?

By the product rule, there are 16.

- There are another two with only one primitive proposition.

We've just defined $\neg$, $\wedge$, $\vee$, $\Rightarrow$

- Why didn't we bother with the rest?
- They're definable!

E.g. $P \oplus Q$ is equivalent to $(P \wedge \neg Q) \vee (\neg P \wedge Q)$

How many possible truth tables are there with two primitive propositions?

P	Q	?
T	T	
T	F	
F	T	
F	F	

(a) 16?
(b) 32?
(c) no clue?

By the product rule, there are 16.

- There are another two with only one primitive proposition.

We've just defined $\neg$, $\wedge$, $\vee$, $\Rightarrow$

- Why didn't we bother with the rest?
- They're definable!

E.g. $P \oplus Q$ is equivalent to $(P \wedge \neg Q) \vee (\neg P \wedge Q)$

- Could get rid of $\Rightarrow$: $P \Rightarrow Q$ is equivalent to $\neg P \vee Q$.
- Could also get rid of $\vee$: $P \vee Q$ is equivalent to $\neg(\neg P \wedge \neg Q)$.

Tautologies

A *truth assignment* is an assignment of T or F to every proposition.

- How hard is it to check if a formula is true under a given truth assignment?
- Easy: just plug it in and evaluate.
  - Time linear in the length of the formula

A *tautology* (or *theorem*) is a formula that evaluates to T for *every* truth assignment.

Examples:

- $(P \vee Q) \Leftrightarrow \neg(\neg P \wedge \neg Q)$
- $P \vee Q \vee (\neg P \wedge \neg Q)$
- $(P \Rightarrow Q) \vee (Q \Rightarrow P)$
  - It's necessarily true that if elephants are pink then the moon is made of green cheese or if the moon is made of green cheese, then elephants are pink.

How hard is it to check if a formula is a tautology?

- How many truth assignments are there for a formula with n primitive propositions:
  (a) 2^n
  (b) n
  (c) ???

Checking tautologies

Are there better ways of telling if a formula is a tautology than trying all possible truth assignments.

- In the worst case, it appears not.
  - The problem is co-NP-complete.
  - The *satisfiability* problem—deciding if at least one truth assignment makes the formula true—is NP-complete.

Nevertheless, it often seems that the reasoning is straightforward: Why is this true:

$$((P \Rightarrow Q) \wedge (Q \Rightarrow R)) \Rightarrow (P \Rightarrow R)$$

We want to show that if $P \Rightarrow Q$ and $Q \Rightarrow R$ is true, then $P \Rightarrow R$ is true.

So assume that $P \Rightarrow Q$ and $Q \Rightarrow R$ are both true. To show that $P \Rightarrow R$, assume that P is true. Since $P \Rightarrow Q$ is true, Q must be true. Since $Q \Rightarrow R$ is true, R must be true. Hence, $P \Rightarrow R$ is true.

We want to codify such reasoning.

Formal Deductive Systems

A *formal deductive system* (also known as an *axiom system*) consists of

- *axioms* (special formulas)
- *rules of inference*: ways of getting new formulas from other formulas. These have the form

  $$\begin{array}{l} A_1 \\ A_2 \\ \vdots \\ A_n \\ \hline B \end{array}$$

  Read this as "from $A_1, \ldots, A_n$, infer B."
  - Sometimes written "$A_1, \ldots, A_n \vdash B$"

Think of the axioms as tautologies, while the rules of inference give you a way to derive new tautologies from old ones.

Derivations

A *derivation* (or *proof*) in an axiom system AX is a sequence of formulas

$$C_1, \ldots, C_N;$$

each formula C_k is either an axiom in AX or follows from previous formulas using an inference rule in AX:

- i.e., there is an inference rule $A_1, \ldots, A_n \vdash B$ such that $A_i = C_{j_i}$ for some $j_i < N$ and $B = C_N$.

This is said to be a *derivation* or *proof* of C_N.

A derivation is a syntactic object: it's just a sequence of formulas that satisfy certain constraints.

- Whether a formula is derivable depends on the axiom system
- Different axioms $\rightarrow$ different formulas derivable
- Derivation has nothing to do with truth!
  - How can we connect derivability and truth?

Typical Axioms

- $P \Rightarrow \neg\neg P$
- $P \Rightarrow (Q \Rightarrow P)$

What makes an axiom "acceptable"?

- it's a tautology

Typical Rules of Inference

Modus Ponens

$$\frac{A \Rightarrow B \quad A}{B}$$

Modus Tollens

$$\frac{A \Rightarrow B \quad \neg B}{\neg A}$$

What makes a rule of inference "acceptable"?

- It preserves validity:
  - if the antecedents are valid, so is the conclusion
- Both modus ponens and modus tollens are acceptable

Sound and Complete Axiomatizations

Standard question in logic:

> *Can we come up with a nice* sound and complete axiomatization*: a (small, natural) collection of axioms and inference rules from which it is possible to derive all and only the tautologies?*

- *Soundness* says that only tautologies are derivable
- *Completeness* says you can derive all tautologies

If all the axioms are valid and all rules of inference preserve validity, then all formulas that are derivable must be valid.

- Proof: by induction on the length of the derivation

It's not so easy to find a complete axiomatization.

A Sound and Complete Axiomatization for Propositional Logic

Consider the following axiom schemes:

A1. $A \Rightarrow (B \Rightarrow A)$

A2. $(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$

A3. $((A \Rightarrow B) \Rightarrow ((A \Rightarrow \neg B) \Rightarrow \neg A)$

These are axioms schemes; each one encodes an infinite set of axioms:

- $P \Rightarrow (Q \Rightarrow P)$, $(P \Rightarrow R) \Rightarrow (Q \Rightarrow (P \Rightarrow R))$ are instances of A1.

Theorem: A1, A2, A3 + modus ponens give a sound and complete axiomatization for formulas in propositional logic involving only $\Rightarrow$ and $\neg$.

- Recall: can define $\vee$ and $\wedge$ using $\Rightarrow$ and $\neg$
  - $P \vee Q$ is equivalent to $\neg P \Rightarrow Q$
  - $P \wedge Q$ is equivalent to $\neg(P \Rightarrow \neg Q)$

A Sample Proof

Derivation of $P \Rightarrow P$:

1. $P \Rightarrow ((P \Rightarrow P) \Rightarrow P)$
   [instance of A1: take $A = P$, $B = P \Rightarrow P$]
2. $(P \Rightarrow ((P \Rightarrow P) \Rightarrow P)) \Rightarrow ((P \Rightarrow (P \Rightarrow P)) \Rightarrow (P \Rightarrow P))$
   [instance of A2: take $A = C = P$, $B = P \Rightarrow P$]
3. $(P \Rightarrow (P \Rightarrow P)) \Rightarrow (P \Rightarrow P)$
   [applying modus ponens to 1, 2]
4. $P \Rightarrow (P \Rightarrow P)$ [instance of A1: take $A = B = P$]
5. $P \Rightarrow P$ [applying modus ponens to 3, 4]

Try deriving $P \Rightarrow \neg\neg P$ from these axioms

- it's hard!

Algorithm Verification

This is (yet another) hot area of computer science.

- How do you prove that your program is correct?
  - You could test it on a bunch of instances. That runs the risk of not exercising all the features of the program.

In general, this is an intractable problem.

- For small program fragments, formal verification using logic is useful
- It also leads to insights into program design.

Syntax of First-Order Logic

We have:

- *constant symbols*: *Alice*, *Bob*
- variables: $x, y, z, \ldots$
- *predicate symbols* of each arity: P, Q, R, $\ldots$
  - A *unary* predicate symbol takes one argument: $P(Alice)$, $Q(z)$
  - A *binary* predicate symbol takes two arguments: *Loves(Bob,Alice)*, *Taller(Alice,Bob)*.

An *atomic expression* is a predicate symbol together with the appropriate number of arguments.

- Atomic expressions act like primitive propositions in propositional logic
  - we can apply $\wedge$, $\vee$, $\neg$ to them
  - we can also quantify the variables that appear in them

Typical formula:

$$\forall x \exists y (P(x, y) \Rightarrow \exists z Q(x, z))$$

Semantics of First-Order Logic

Assume we have some domain D.

- The domain could be finite:
  - $\{1, 2, 3, 4, 5\}$
  - the people in this room
- The domain could be infinite
  - N, R, ...

A statement like $\forall x P(x)$ means that $P(d)$ is true for each d in the domain.

- If the domain is N, then $\forall x P(x)$ is equivalent to

$$P(0) \wedge P(1) \wedge P(2) \wedge \ldots$$

Similarly, $\exists x P(x)$ means that $P(d)$ is true for some d in the domain.

- If the domain is N, then $\exists x P(x)$ is equivalent to

$$P(0) \vee P(1) \vee P(2) \vee \ldots$$

Is $\exists x(x^2 = 2)$ true?

(a) Yes

(b) No

(c) It depends

Yes if the domain is R; no if the domain is N.

How about $\forall x \forall y((x < y) \Rightarrow \exists z(x < z < y))$?

Is $\exists x(x^2 = 2)$ true?

(a) Yes

(b) No

(c) It depends

Yes if the domain is R; no if the domain is N.

How about $\forall x \forall y((x < y) \Rightarrow \exists z(x < z < y))$?

We'll skip the formal semantics of first-order logic here.

- If you want to know more, take a logic course!

Translating from English to First-Order Logic

All men are mortal
Socrates is a man
Therefore Socrates is mortal

There is two unary predicates: *Mortal* and *Man*
There is one constant: *Socrates*
The domain is the set of all people

$\forall x(Man(x) \Rightarrow Mortal(x))$
$Man(Socrates)$

$Mortal(Socrates)$

More on Quantifiers

$\forall x \forall y P(x, y)$ is equivalent to $\forall y \forall x P(x, y)$

- P is true for every choice of x and y

Similarly $\exists x \exists y P(x, y)$ is equivalent to $\exists y \exists x P(x, y)$

- P is true for some choice of (x, y).

What about $\forall x \exists y P(x, y)$? Is it equivalent to $\exists y \forall x P(x, y)$?

(a) Yes

(b) $\exists y \forall x P(x, y)$ implies $\forall x \exists y P(x, y)$, but the converse isn't true

(c) $\forall x \exists y P(x, y)$ implies $\exists y \forall x P(x, y)$, but the converse isn't true

(d) ???

More on Quantifiers

$\forall x \forall y P(x, y)$ is equivalent to $\forall y \forall x P(x, y)$

- P is true for every choice of x and y

Similarly $\exists x \exists y P(x, y)$ is equivalent to $\exists y \exists x P(x, y)$

- P is true for some choice of (x, y).

What about $\forall x \exists y P(x, y)$? Is it equivalent to $\exists y \forall x P(x, y)$?

(a) Yes

(b) $\exists y \forall x P(x, y)$ implies $\forall x \exists y P(x, y)$, but the converse isn't true

(c) $\forall x \exists y P(x, y)$ implies $\exists y \forall x P(x, y)$, but the converse isn't true

(d) ???

Suppose the domain is the natural numbers. Compare:

- $\forall x \exists y (y \geq x)$
- $\exists y \forall x (y \geq x)$

More on Quantifiers

$\forall x \forall y P(x, y)$ is equivalent to $\forall y \forall x P(x, y)$

- P is true for every choice of x and y

Similarly $\exists x \exists y P(x, y)$ is equivalent to $\exists y \exists x P(x, y)$

- P is true for some choice of (x, y).

What about $\forall x \exists y P(x, y)$? Is it equivalent to $\exists y \forall x P(x, y)$?

(a) Yes

(b) $\exists y \forall x P(x, y)$ implies $\forall x \exists y P(x, y)$, but the converse isn't true

(c) $\forall x \exists y P(x, y)$ implies $\exists y \forall x P(x, y)$, but the converse isn't true

(d) ???

Suppose the domain is the natural numbers. Compare:

- $\forall x \exists y (y \geq x)$
- $\exists y \forall x (y \geq x)$

In general, $\exists y \forall x P(x, y) \Rightarrow \forall x \exists y P(x, y)$ is *logically valid*.

- A logically valid formula in first-order logic is the analogue of a tautology in propositional logic.
- A formula is logically valid if it's true in every domain and for every *interpretation* of the predicate symbols.

More valid formulas involving quantifiers:

- $\neg\forall x P(x) \Leftrightarrow \exists x \neg P(x)$
- Replacing P by $\neg P$, we get:

$$\neg\forall x \neg P(x) \Leftrightarrow \exists x \neg\neg P(x)$$

- Therefore

$$\neg\forall x \neg P(x) \Leftrightarrow \exists x P(x)$$

- Similarly, we have

$$\neg\exists x P(x) \Leftrightarrow \forall x \neg P(x)$$

$$\neg\exists x \neg P(x) \Leftrightarrow \forall x P(x)$$

Axiomatizing First-Order Logic

Just as in propositional logic, there are axioms and rules of inference that provide a sound and complete axiomatization for first-order logic, independent of the domain.

A typical axiom:

- $\forall x(P(x) \Rightarrow Q(x)) \Rightarrow (\forall x P(x) \Rightarrow \forall x Q(x))$.

A typical rule of inference is *Universal Generalization*:

$$\varphi(x) \vdash \forall x \varphi(x)$$

Gödel provided a sound and complete axioms system for first-order logic in 1930.

Is Everything Provable?

If we ask you to prove something from homework which happens true, is it necessarily provable?

Is Everything Provable?

If we ask you to prove something from homework which happens true, is it necessarily provable?

- Of course, if we ask you to prove it, then it should be provable.
- But what about if a computer scientist is trying to prove a theorem that she is almost certain is true.
  - Can she be confident that it has a proof?
  - Can something be true without being provable?
    - Yes
    - No
    - It depends.

Is Everything Provable?

If we ask you to prove something from homework which happens true, is it necessarily provable?

- Of course, if we ask you to prove it, then it should be provable.
- But what about if a computer scientist is trying to prove a theorem that she is almost certain is true.
  - Can she be confident that it has a proof?
  - Can something be true without being provable?
    - Yes
    - No
    - It depends.

Remember, whether something is provable depends on the rules of the game:

- the axioms and inference rules

Obviously, you can't prove much if you don't have a good selection of axioms and inference rules to work with.

- In a remarkable result, Gödel proved that, *no matter what axiom system AX you used*, there were statements that were true about arithmetic that could not be proved in AX.

Axiomatizing Arithmetic

Suppose we restrict the domain to the natural numbers, and allow only the standard symbols of arithmetic ($+$, $\times$, $=$, $>$, 0, 1).
Typical true formulas include:

- $\forall x \exists y (x \times y = x)$
- $\forall x \exists y (x = y + y \vee x = y + y + 1)$

Let $Prime(x)$ be an abbreviation of

$$\forall y \forall z ((x = y \times z) \Rightarrow ((y = 1) \vee (y = x)))$$

When is $Prime(x)$ true?

Axiomatizing Arithmetic

Suppose we restrict the domain to the natural numbers, and allow only the standard symbols of arithmetic ($+$, $\times$, $=$, $>$, 0, 1).
Typical true formulas include:

- $\forall x \exists y (x \times y = x)$
- $\forall x \exists y (x = y + y \vee x = y + y + 1)$

Let $Prime(x)$ be an abbreviation of

$$\forall y \forall z ((x = y \times z) \Rightarrow ((y = 1) \vee (y = x)))$$

When is $Prime(x)$ true? If x is prime!

Axiomatizing Arithmetic

Suppose we restrict the domain to the natural numbers, and allow only the standard symbols of arithmetic ($+$, $\times$, $=$, $>$, 0, 1).
Typical true formulas include:

- $\forall x \exists y (x \times y = x)$
- $\forall x \exists y (x = y + y \vee x = y + y + 1)$

Let $Prime(x)$ be an abbreviation of

$$\forall y \forall z ((x = y \times z) \Rightarrow ((y = 1) \vee (y = x)))$$

When is $Prime(x)$ true? If x is prime!

What does the following formula say?

- $\forall x (\exists y (y > 1 \wedge x = y + y) \Rightarrow$
  $\exists z_1 \exists z_2 (Prime(z_1) \wedge Prime(z_2) \wedge x = z_1 + z_2))$

Axiomatizing Arithmetic

Suppose we restrict the domain to the natural numbers, and allow only the standard symbols of arithmetic ($+$, $\times$, $=$, $>$, 0, 1).
Typical true formulas include:

- $\forall x \exists y (x \times y = x)$
- $\forall x \exists y (x = y + y \vee x = y + y + 1)$

Let $Prime(x)$ be an abbreviation of

$$\forall y \forall z ((x = y \times z) \Rightarrow ((y = 1) \vee (y = x)))$$

When is $Prime(x)$ true? If x is prime!

What does the following formula say?

- $\forall x (\exists y (y > 1 \wedge x = y + y) \Rightarrow$
  $\exists z_1 \exists z_2 (Prime(z_1) \wedge Prime(z_2) \wedge x = z_1 + z_2))$
- This is *Goldbach's conjecture*: every even number other than 2 is the sum of two primes.
  - Is it true? We don't know.

Gödel's Incompleteness Theorem

Is there an axiom system from which you can prove all and only true statements about arithmetic?

- that is, you want the axiom system to be *sound*
  - The axioms must be valid arithmetic facts, and the rules of inference must preserve validity
  - otherwise you could prove statements that are false

  and *complete*
  - This means that you can prove *all* true statements

This is easy!

- Just take the axioms to consist of all true statements.

Gödel's Incompleteness Theorem

Is there an axiom system from which you can prove all and only true statements about arithmetic?

- that is, you want the axiom system to be *sound*
  - The axioms must be valid arithmetic facts, and the rules of inference must preserve validity
  - otherwise you could prove statements that are false

  and *complete*
  - This means that you can prove *all* true statements

This is easy!

- Just take the axioms to consist of all true statements.

That's cheating! To make this interesting, we need a restriction:

- The set of axioms must be "nice"
  - technically: *recursive*, so that a program can check whether a formula is an axiom

Gödel's Incompleteness Theorem

Is there an axiom system from which you can prove all and only true statements about arithmetic?

- that is, you want the axiom system to be *sound*
  - The axioms must be valid arithmetic facts, and the rules of inference must preserve validity
  - otherwise you could prove statements that are false

  and *complete*
  - This means that you can prove *all* true statements

This is easy!

- Just take the axioms to consist of all true statements.

That's cheating! To make this interesting, we need a restriction:

- The set of axioms must be "nice"
  - technically: *recursive*, so that a program can check whether a formula is an axiom

Gödel's Incompleteness Theorem: There is no sound and complete recursive axiomatization of arithmetic.

- This is arguably the most important result in mathematics of the 20th century.