Propositional logic is limited

How could we express the following?

- All cats have more days than dogs.
- The length of one side of a triangle is less than the sum of the lengths of the other two sides.

First-order logic gives us ways of talking about:

- Infinite sets of objects
- Functions and relations over objects
- Quantified statements over objects
Logical Symbols:

- Propositional connectives: $\land, \lor, \neg, \rightarrow, \leftrightarrow$
- Variables: $v, y, z, x_1, x_2, \ldots$
- Quantifiers: $\exists, \forall$

Non-logical symbols (parameters):

- Constant symbols: $c_1, c_2, \ldots$
- Function symbols: $g, h, f, f_1, f_2, \ldots$
- Predicate symbols: $r, q, p, p_1, p_2, \ldots$
Predicate and function symbols are associated with an *arity*

Natural number that describes # of arguments

Examples:

- `=`: arity 2
- `f(a, b, c)`: arity 3
- Constants: can be seen as 0-arity functions
- Propositional variables: can be seen as 0-arity predicates
Terms in FOL evaluate to values other than truth values

- People
- Strings
- 64-bit integers

Terms are expressions that name objects

- Constants are terms
- Variables are terms
- For each function symbol $f$ of arity $n$, $f(t_1, \ldots, t_n)$ is a term if $t_1, \ldots, t_n$ are terms.
Atoms in FOL evaluate to either true or false

These generalize propositional assertions

- $\bot$, $\top$ are atoms
- Nullary predicates $p, q, \ldots$ are atoms
- For each predicate symbol $p$ of arity $n$,
  \[ p(t_1, \ldots, t_n) \]
  is an atom if $t_1, \ldots, t_n$ are atoms

As before, a literal is an atom or its negation
A first-order formula is:

- A literal
- The application of $\neg, \land, \lor, \rightarrow, \leftrightarrow$ to a formula
- The application of a quantifier to a formula

There are two quantifiers:

- $\forall x. F[x]$: “For all $x$, $F[x]$”
- $\exists x. F[x]$: “There exists an $x$ such that $F[x]$”

In each case,

- $x$ is the *quantified variable*
- $F[x]$ is the *scope* of the quantifier
- $x$ is *bound* in $F[x]$ by the quantifier
- $y$ is *free* in $F$ if it is not bound by any quantifier
A few distinctions on first-order formulas:

▶ A formula is *closed* if it contains no free variables
▶ Closed formulas are also called *sentences*
▶ A formula containing free variables is *open*
▶ A formula is *ground* if it contains no variables

Which are these examples

▶ \( \forall y.((\forall x.p(x)) \rightarrow q(x, y)) \)
▶ \( \forall y.((\forall x.p(x)) \rightarrow (\exists x.q(x, y))) \)
▶ \( p(a, f(b)) \rightarrow q(c) \)
Examples

How do we “pronounce” the formula? Which variables are free/bound?

\[ \forall x. p(f(x), y) \rightarrow \forall y. p(f(x), y) \]

\[ \forall x. g(x) \rightarrow \exists y. f(y) \land h(x, y) \]

\[ \forall x'. x' < x \rightarrow \forall y. y > 0 \rightarrow r(x', y) < y \]

\[ p(i, u) \rightarrow (f(a, i) = e \rightarrow \exists j. \ell \leq j \leq u \land f(a, j) = e) \]
“All cats have more days than dogs.”
\[ \forall x, y. \text{dog}(x) \land \text{cat}(y) \rightarrow \text{ndays}(y) > \text{ndays}(x) \]

The numeric array \( a \) is sorted
\[ \forall i. 0 \leq i < |a| \rightarrow a[i] \leq a[i + 1] \]

Graph \( G \) contains a triangle
\[ \exists v_1, v_2, v_3. e(v_1, v_2) \land e(v_2, v_3) \land e(v_3, v_1) \]

Graph \( G \) is connected
\[ \ldots \]
A first-order structure $S = (D, I)$:

- $D$: **universe of discourse**, non-empty set of objects we’d like to talk about

- $I$: Interpretation mapping parameters to objects, functions, and predicates in $D_I$:
  1. every constant symbol to a value in $D$
  2. every function symbol $f$ of arity $n$ to a function $f^I : D \rightarrow D$
  3. every relation symbol $p$ of arity $n$ to a relation $p^I \subseteq D^n$

Assignments map variables to values (not object constants!):
$\alpha : \text{Vars} \rightarrow D_I$
\[ x + y > z \rightarrow y > z - x \]

What’s the “standard” structure?

Universe of discourse:
- \( D = \mathbb{Z} = \{ \ldots, -1, 0, 1, \ldots \} \)

Interpretation:
- Function symbols: \(+ \mapsto +_\mathbb{Z}, - \mapsto -_\mathbb{Z}\)
- Predicate symbols: \(> \mapsto >_\mathbb{Z}\)

Possible assignment:
\[ \alpha = \{ x \mapsto 0, y \mapsto 1, z \mapsto -1 \} \]
Given $I$ and $\alpha$, we can evaluate terms to values in $D$

We write $\langle I, \alpha \rangle(a)$ to denote the evaluation of term $a$

Three cases for terms:

- **Object constants:** $\langle I, \alpha \rangle(a) = I(a)$
- **Variables:** $\langle I, \alpha \rangle(v) = \alpha(v)$
- **Function terms:**
  $$\langle I, \alpha \rangle(f(t_1, \ldots, t_n)) = I(f)(\langle I, \alpha \rangle(t_1), \ldots, \langle I, \alpha \rangle(t_n))$$
Given $D, I, \alpha$, we define the evaluation of a first-order formula $F$

We write:

- $D, I, \alpha \models F$ if $F$ evaluates to $true$
- $D, I, \alpha \not\models F$ if $F$ evaluates to $false$

Base cases:

- $D, I, \alpha \models \top$ and $D, I, \alpha \not\models \bot$
- $D, I, \alpha \models p(t_1, \ldots, t_n)$ iff $(\langle I, \alpha \rangle(t_1), \ldots, \langle I, \alpha \rangle(t_n)) \in I(p)$
Evaluating Formulas

Inductive case for connectives:

\[ D, I, \alpha \models \neg F \quad \text{iff} \quad D, I, \alpha \not\models F \]

\[ D, I, \alpha \models F_1 \land F_2 \quad \text{iff} \quad D, I, \alpha \models F_1 \text{ and } D, I, \alpha \models F_2 \]

\[ D, I, \alpha \models F_1 \lor F_2 \quad \text{iff} \quad D, I, \alpha \not\models F_1 \text{ or } D, I, \alpha \models F_2 \]

\[ D, I, \alpha \models F_1 \rightarrow F_2 \quad \text{iff} \quad D, I, \alpha \not\models F_1 \text{ or } I \models F_2 \]

\[ D, I, \alpha \models F_1 \leftrightarrow F_2 \quad \text{iff} \quad D, I, \alpha \not\models F_1 \text{ and } I \models F_2, \text{ or } \]

\[ D, I, \alpha \not\models F_1 \text{ and } D, I, \alpha \not\models F_2 \]

These are the same as they were for propositional logic.
Example

For universe $D = \{\circ, \bullet\}$, assignment $\alpha = \{x \mapsto \bullet, y \mapsto \circ\}$, and $I$:

$\begin{align*}
&\quad I(a) = \circ \\
&\quad I(f) = \{(\circ, \circ) \mapsto \circ, (\circ, \bullet) \mapsto \bullet, (\bullet, \circ) \mapsto \bullet, (\bullet, \bullet) \mapsto \circ\} \\
&\quad I(g) = \{\circ \mapsto \bullet, \bullet \mapsto \circ\} \\
&\quad I(p) = \{(\circ, \bullet), (\bullet, \bullet)\}
\end{align*}$

What do the following evaluate to?

$\begin{align*}
&\quad p(a, g(\circ)) = \text{true} \\
&\quad p(x, f(g(x), y)) \rightarrow p(y, g(x)) = \text{true}
\end{align*}$
Evaluating Quantifiers

Let $x$ be a variable assigned by $\alpha$

An $x$-variant of $\alpha$ is an interpretation that:
- Agrees with $\alpha$ on all variables except $x$
- Assigns $x$ to some given value $c \in D$

We write an $x$-variant of $\alpha$ as $\alpha[x \mapsto c]$

Universal quantifier:
$$D, I, \alpha \models \forall x. F \text{ iff for all } c \in D, D, I, \alpha[x \mapsto c] \models F$$

Existential quantifier
$$D, I, \alpha \models \exists x. F \text{ iff there exists } c \in D, D, I, \alpha[x \mapsto c] \models F$$
Example

For universe $D = \{\circ, \bullet\}$, assignment $\alpha = \{x \mapsto \bullet, y \mapsto \circ\}$, and $I$:

- $I(a) = \circ$
- $I(f) = \{(\circ, \circ) \mapsto \circ, (\circ, \bullet) \mapsto \bullet, (\bullet, \circ) \mapsto \bullet, (\bullet, \bullet) \mapsto \circ\}$
- $I(g) = \{\circ \mapsto \bullet, \bullet \mapsto \circ\}$
- $I(p) = \{(\circ, \bullet), (\bullet, \bullet)\}$

What do the following evaluate to?

- $\exists x. \neg p(x, g(a)) = \text{false}$
- $\forall w. \exists z. p(w, f(g(w), z)) = \text{true}$
- $(\exists x. p(x, x)) \rightarrow p(y, g(x)) = \text{false}$
A first-order formula $F$ is satisfiable if and only if:

There exists $S = (D, I)$ and assignment $\alpha$ where $D, I, \alpha \models F$

A first-order formula $F$ is valid if and only if:

For all $S = (D, I)$ and assignments $\alpha, D, I, \alpha \models F$

We write $\models F$ when $F$ is valid

As before, these are dual to each other
Is the formula $\exists x. f(x) = g(x)$ satisfiable?

**Satisfying $S, \alpha$:**

- $D = \{0, 1\}$
- $I(f) = \{0 \mapsto 1, 1 \mapsto 1\}$
- $I(g) = \{0 \mapsto 0, 1 \mapsto 1\}$

Is it valid?

**Falsifying $S, \alpha$:**

- $D = \{0, 1\}$
- $I(f) = \{0 \mapsto 1, 1 \mapsto 1\}$
- $I(g) = \{0 \mapsto 1, 1 \mapsto 0\}$
We’ll extend the semantic argument method from earlier

1. Assume \( F \) is not valid: there exists \( I \) such that \( I \not\models F \)
2. Apply proof rules (more on this shortly)
3. **If:** no contradiction, no applicable rules, conclude that \( F \) is invalid
4. **If:** every branch reaches contradiction, conclude that \( F \) is valid

But, we need new proof rules for quantifiers
Recall the semantics for implication:

- From $D, I, \alpha \models F_1 \rightarrow F_2$:
  \[
  \frac{D, I, \alpha \models F_1 \rightarrow F_2}{D, I, \alpha \not\models F_1 \quad D, I, \alpha \models F_2}
  \]

- From $D, I, \alpha \not\models F_1 \rightarrow F_2$:
  \[
  \frac{D, I, \alpha \not\models F_1 \rightarrow F_2}{D, I, \alpha \models F_1 \quad D, I, \alpha \not\models F_2}
  \]
Proof Rules: Universal Quantification

- From $D, I, \alpha \models \forall x.F$:
  \[
  \frac{D, I, \alpha \models \forall x.F}{D, I, \alpha[x \mapsto c] \models F}
  \]
  For any $c \in D$

For example, if we know $D, I, \alpha \models \forall x.p(x, a)$

Then we can conclude $D, I, \alpha[x \mapsto b] \models p(x, a)$
Proof Rules: Universal Quantification

► From $D, I, \alpha \not\models \forall x. F$:

\[
\begin{aligned}
D, I, \alpha \not\models \forall x. F \\
\hline
\quad D, I, \alpha[x \mapsto c] \not\models F
\end{aligned}
\]

For a fresh $c \in D$

Here, “fresh” means “not used previously in the proof”

If $D, I, \alpha \not\models \forall x. F$, all we know is that $F$ doesn’t hold for some object

We don’t know which object

Hence, we pick a new one, making no assumptions
Proof Rules: Existential Quantification

- From $D, I, \alpha \models \exists x. F$:

\[
\begin{align*}
D, I, \alpha \models \exists x. F \\
\frac{}{D, I, \alpha[x \mapsto c] \models F}
\end{align*}
\]

For a fresh $c \in D$

Again “fresh” means “not used previously in the proof”

Notice the similarity to the previous rule
Proof Rules: Existential Quantification

- From \( D, I, \alpha \not\models \exists x. F \):

\[
\frac{D, I, \alpha \not\models \exists x. F}{D, I, \alpha[x \mapsto c] \not\models F}
\]

For any \( c \in D \)

There does not exist any object \( c \) for which \( F \) holds

No matter what \( x \) maps to, \( F \) won’t hold

Even if its something we used before
Contradiction rule:

\[ D, I, \alpha[\cdots] \models p(s_1, \ldots, s_n) \]
\[ D, I, \alpha[\cdots] \not\models p(t_1, \ldots, t_n) \]
\[ \langle I, \alpha[\cdots] \rangle(s_i) = \langle I, \alpha[\cdots] \rangle(t_i) \text{ for all } 1 \leq i \leq n \]

\[ \bot \]

In the top two lines, the assignments are both variants of \( \alpha \).

Contradiction exists whenever disagreement on value of \( p \).

For example, if 
\[ D, I, \{x \mapsto a\} \models p(x) \]
\[ D, I, \{y \mapsto a\} \not\models p(y) \]
then \( \bot \)
Example

Prove the following valid:

\[ F : (\forall x.p(x)) \rightarrow (\forall y.p(y)) \]

1. \( D, I, \alpha \not\models F \)
2. \( D, I, \alpha \models \forall x.p(x) \quad 1 \text{ and } \rightarrow \)
3. \( D, I, \alpha \not\models \forall y.p(y) \quad 1 \text{ and } \rightarrow \)
4. \( D, I, \alpha[x \rightarrow c] \models p(x) \quad 2 \text{ and } \forall \)
5. \( D, I, \alpha[y \rightarrow c] \not\models p(x) \quad 3 \text{ and } \forall \)
6. \( \bot \quad 4 \text{ and } 5 \)
Prove the following valid:

\[ F : (\forall x. p(x)) \leftrightarrow (\neg \exists x. \neg p(x)) \]
Soundness and Completeness

These proof rules satisfy important properties

**Soundness**

If every branch of the semantic argument proof derives a contradiction, then $F$ is invalid. In other words, the proof rules don’t reach wrong conclusions.

**Completeness**

If $F$ is valid, then there exists a finite-length proof in which every branch derives a contradiction. In other words, there are no valid formulas we can’t prove to be valid.

This is called *refutational completeness*
Decidability of First-Order Logic

Decidable Problem

A decision problem is *decidable* if and only if there exists a procedure $P$ such that, for any input, either

1. Halts with “yes” when the answer is positive
2. Halts with “no” when the answer is negative

**Important result** (Church & Turing): The problem of deciding validity for first-order logic is not decidable.

Is there a problem here?
Semidecidability

Semidecidable Problem

A decision problem is *semidecidable* if and only if there exists a procedure $P$ such that, for any input, either

1. Halts with “yes” when the answer is positive
2. Halts with “no” when the answer is negative
3. or does not halt when the answer is negative

The first-order validity decision problem is semidecidable

- Procedure always halts with **yes** when $F$ is valid
- Might not terminate when $F$ is not valid
Next Lecture

Topic: First-order theories

Read: Chapter 3, up to (not including) 3.5

Start the homework!

Come to office hours with questions