First-order logic

- First-order logic (FOL) models the world in terms of
  - **Objects,** which are things with individual identities
  - **Properties** of objects that distinguish them from other objects
  - **Relations** that hold among sets of objects
  - **Functions,** which are a subset of relations where there is only one “value” for any given “input”
- Examples:
  - Objects: Students, lectures, companies, cars ...
  - Relations: Brother-of, bigger-than, outside, part-of, has-color, occurs-after, owns, visits, precedes, ...
  - Properties: blue, oval, even, large, ...
  - Functions: father-of, best-friend, second-half, one-more-than ...

Syntax of FOL

```
S := <Sentence> ;
<Sentence> := <AtomicSentence> |
          <Sentence> <Connective> <Sentence> |
          <Quantifier> <Variable>,... <Sentence> |
          "NOT" <Sentence> |
          "(" <Sentence> ")";
<AtomicSentence> := <Predicate> "(" <Term>, ... ")" |
                    <Term> "=" <Term>;
<Term> := <Function> "(" <Term>, ... ")" |
          <Constant> |
          <Variable>;
<Connective> := "AND" | "OR" | "IMPLIES" | "EQUIVALENT";
<Quantifier> := "EXISTS" | "FORALL" ;
<Constant> := "A" | "X1" | "John" | ... ;
<Variable> := "a" | "x" | "s" | ... ;
<Predicate> := "Before" | "HasColor" | "Raining" | ... ;
<Function> := "Mother" | "LeftLegOf" | ... ;
```

Constants, Functions, Predicates

- **Constant symbols,** which represent individuals in the world
  - Mary
  - 3
  - Green
- **Function symbols,** which map individuals to individuals
  - father-of(Mary) = John
  - color-of(Sky) = Blue
- **Predicate symbols,** which map individuals to truth values
  - greater(5,3)
  - green(Grass)
  - color(Grass, Green)

Variables, Connectives, Quantifiers

- **Variable symbols**
  - E.g., x, y, foo
- **Connectives**
  - Same as in PL: not (¬), and (∧), or (∨), implies (→), if and only if (biconditional ↔)
- **Quantifiers**
  - Universal **∀x** or **(Ax)**
  - Existential **∃x** or **(Ex)**

Quantifiers

- **Universal quantification**
  - (∀x)P(x) means that P holds for **all** values of x in the domain associated with that variable
  - E.g., (∀x) dolphin(x) → mammal(x)
- **Existential quantification**
  - (∃ x)P(x) means that P holds for **some** value of x in the domain associated with that variable
  - E.g., (∃ x) mammal(x) ∧ lays-eggs(x)
  - Permits one to make a statement about some object without naming it

Sentences are built from terms and atoms

- A **term** (denoting a real-world individual) is a constant symbol, a variable symbol, or an n-place function of n terms.

  x and f(x_1, ..., x_n) are terms, where each x_i is a term.

  A term with no variables is a **ground term**
- An **atom** (which has value true or false) is either

  an n-place predicate of n terms, or,

  ¬P, P∨Q, P∧Q, P→Q, P↔Q where P and Q are atoms
- A **sentence** is an atom, or, if P is a sentence and x is a variable, then (∀x)P and (∃x)P are sentences
- A **well-formed formula (wff)** is a sentence containing no "free" variables. That is, all variables are "bound" by universal or existential quantifiers.

  (∀x)P(x,y) has x bound as a universally quantified variable, but y is free.

Translating English to FOL

Every gardener likes the sun.
(∀x) gardener(x) → likes(x,Sun)
You can fool some of the people all of the time.
(∃x)(∀t) (person(x) ^ time(t)) → can-fool(x,t)
You can fool all of the people some of the time.
(∀x)(∃t) (person(x) ^ time(t) → can-fool(x,t)
All purple mushrooms are poisonous.
(∀x) (mushroom(x) ^ purple(x)) → poisonous(x)
No purple mushroom is poisonous.
¬(∃x) purple(x) ^ mushroom(x) ^ poisonous(x)
(∀x) (mushroom(x) ^ purple(x)) → ¬poisonous(x)
There are exactly two purple mushrooms.
(∃x)(∃y) mushroom(x) ^ purple(x) ^ mushroom(y) ^ purple(y) ^ ¬(x=y) ^ (∀ z) (mushroom(z) ^ purple(z)) → ((x=z) v (y=z))
Clinton is not tall.
¬tall(Clinton)
X is above Y if X is on directly on top of Y or there is a pile of one or more other objects directly on top of one another starting with X and ending with Y.
(∀x)(∀y) above(x,y) _ (on(x,y) v (∃z) (on(x,z) ^ above(z,y)))

Quantifiers

- Universal quantifiers are often used with "implies" to form "rules":

  (∀x) student(x) → smart(x) means "All students are smart"
- Universal quantification is *rarely* used to make blanket statements about every individual in the world:

  (∀x)student(x)∧smart(x) means "Everyone in the world is a student and is smart"
- Existential quantifiers are usually used with "and" to specify a list of properties about an individual:

  (∃x) student(x) ∧ smart(x) means "There is a student who is smart"
- A common mistake is to represent this English sentence as the FOL sentence:

  (∃x) student(x) → smart(x)
  - But what happens when there is a person who is *not* a student?

Quantifier Scope

- Switching the order of universal quantifiers *does not* change the meaning:
  - $(\forall x)(\forall y)P(x,y) \Rightarrow (\forall y)(\forall x)\ P(x,y)$
- Similarly, you can switch the order of existential quantifiers:
  - $(\exists x)(\exists y)P(x,y) \Rightarrow (\exists y)(\exists x)\ P(x,y)$
- Switching the order of universals and existentials *does* change meaning:
  - Everyone likes someone: $(\forall x)(\exists y)$ likes(x,y)
  - Someone is liked by everyone: $(\exists y)(\forall x)$ likes(x,y)

Connections between All and Exists

We can relate sentences involving $\forall$ and $\exists$ using De Morgan's laws:

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

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

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

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

Quantified inference rules

- Universal instantiation
  - $\forall x\ P(x)\ \therefore\ P(A)$
- Universal generalization
  - $P(A) \wedge P(B) \ldots\ \therefore\ \forall x\ P(x)$
- Existential instantiation
  - $\exists x\ P(x)\ \therefore P(F)$ ← **skolem constant F**
- Existential generalization
  - $P(A)\ \therefore\ \exists x\ P(x)$

An example from Monty Python

- **FIRST VILLAGER:** We have found a witch. May we burn her?
- **ALL:** A witch! Burn her!
- **BEDEVERE:** Why do you think she is a witch?
- **SECOND VILLAGER:** She turned *me* into a newt.
- **B:** A newt?
- **V2** *(after looking at himself for some time)*: I got better.
- **ALL:** Burn her anyway.
- **B:** Quiet! Quiet! There are ways of telling whether she is a witch.

Monty Python cont.

- **B:** Tell me… what do you do with witches?
- **ALL:** Burn them!
- **B:** And what do you burn, apart from witches?
- **V4:** …wood?
- **B:** So **why do witches burn**?
- **V2** *(pianissimo)*: **because they're made of wood?**
- **B:** Good.
- **ALL:** I see. Yes, of course.

Monty Python cont.

- **B: So how can we tell if she is made of wood?**
- **V1: Make a bridge out of her.**
- B: Ah… but can you not also make bridges out of stone?
- **ALL:** Yes, of course… um… er…
- **B:** Does wood sink in water?
- **ALL:** No, no, it floats. Throw her in the pond.
- **B:** Wait. Wait… tell me, what also floats on water?
- **ALL:** Bread? No, no no. Apples… gravy… very small rocks…
- **B:** No, no, no,

Monty Python cont.

- **KING ARTHUR:** A duck!
- *(They all turn and look at Arthur. Bedevere looks up, very impressed.)*
- **B:** Exactly. So… logically…
- **V1** *(beginning to pick up the thread)*: **If she… weighs the same as a duck… she's made of wood**.
- **B:** And therefore?
- **ALL:** **A witch!**

Monty Python Fallacy #1

- $\forall x$ witch(x) → burns(x)
- $\forall x$ wood(x) → burns(x)
- -------------------------------
- ∴ $\forall z$ witch(x) → wood(x)

- $p \rightarrow q$
- $r \rightarrow q$
- ---------
- $p \rightarrow r$ Fallacy: Affirming the conclusion

Monty Python Near-Fallacy #2

- wood(x) → bridge(x)
- ------------------------------
- ∴ bridge(x) → wood(x)

- B: Ah… but can you not also make bridges out of stone?

Monty Python Fallacy #3

- ∀x wood(x) → floats(x)
- ∀x duck-weight (x) → floats(x)
- ------------------------------
- ∴ ∀x duck-weight(x) → wood(x)

- p → q
- r → q
- -----------
- ∴ r → p

Monty Python Fallacy #4

- ∀z light(z) → wood(z)
- light(W)
- ------------------------------
- ∴ wood(W) ok…………..

- witch(W) → wood(W) applying universal instan. to fallacious conclusion #1

- wood(W)
- ---------------------------------
- ∴ witch(z)

Axioms for Set Theory in FOL

1. The only sets are the empty set and those made by adjoining something to a set:
   ∀s set(s) <=> (s=EmptySet) v (∃x,r Set(r) ^ s=Adjoin(s,r))
2. The empty set has no elements adjoined to it:
   ~ ∃x,s Adjoin(x,s)=EmptySet
3. Adjoining an element already in the set has no effect:
   ∀x,s Member(x,s) <=> s=Adjoin(x,s)
4. The only members of a set are the elements that were adjoined into it:
   ∀x,s Member(x,s) <=> ∃y,r (s=Adjoin(y,r) ^ (x=y v Member(x,r)))
5. A set is a subset of another iff all of the 1st set's members are members of the 2^{nd}:
   ∀s,r Subset(s,r) <=> (∀x Member(x,s) => Member(x,r))
6. Two sets are equal iff each is a subset of the other:
   ∀s,r (s=r) <=> (subset(s,r) ^ subset(r,s))
7. Intersection
   ∀x,s1,s2 member(X,intersection(S1,S2)) <=> member(X,s1) ^ member(X,s2)
8. Union
   ∃x,s1,s2 member(X,union(s1,s2)) <=> member(X,s1) v member(X,s2)

Axioms, definitions and theorems

- **Axioms** are facts and rules that attempt to capture all of the (important) facts and concepts about a domain; axioms can be used to prove **theorems**
  - Mathematicians don't want any unnecessary (dependent) axioms –ones that can be derived from other axioms
  - Dependent axioms can make reasoning faster, however
  - Choosing a good set of axioms for a domain is a kind of design problem
- A **definition** of a predicate is of the form "p(X) __ …" and can be decomposed into two parts
  - **Necessary** description: "$p(x) \rightarrow \ldots$"
  - **Sufficient** description "$p(x) \leftarrow \ldots$"
  - Some concepts don't have complete definitions (e.g., person(x))

Extensions to FOL

- **Higher-order logic**
  - Quantify over relations
- **Representing functions with the lambda operator (λ)**
- **Expressing uniqueness $\exists!$, ι**
- **Sorted logic**

Higher-order logic

- In FOL, variables can only range over objects
- HOL allows us to quantify over relations
- More expressive, but undecidable
- Example:

  "two functions are equal iff they produce the same value for all arguments"
  - $\forall f \forall g\ (f = g) \Rightarrow (\forall x\ f(x) = g(x))$
- Example:

  $\forall r\ transitive(r) \Rightarrow (\forall x \forall y \forall z\ r(x,y) \wedge r(y,z) \rightarrow r(x,z))$

Expressing uniqueness

- Sometimes we want to say that there is a single, unique object that satisfies a certain condition
- "There exists a unique x such that king(x) is true"
  - $\exists x\ king(x) \wedge \forall y\ (king(y) \Rightarrow x=y)$
  - $\exists x\ king(x) \wedge not(\exists y\ (king(y) \wedge x \neq y)$
  - $\exists! x\ king(x)$
- "Every country has exactly one ruler"
  - $\forall c\ country(c) \Rightarrow \exists! r\ ruler(c,r)$
- Iota operator: "ι x P(x)" means "the unique x such that p(x) is true"
  - "The unique ruler of Freedonia is dead"
  - dead(ι x ruler(freedonia,x))

Notational differences

- **Different symbols** for *and, or, not, implies, ...*
  - ∀ ∃ ⇒ ⇔ ∧ ∨ ¬ • ⊃
  - p v (q ^ r)
  - p + (q * r)
  - etc
- **Prolog**

  cat(X) :- furry(X), meows (X), has(X, claws)
- **Lispy notations**

```
(forall ?x (implies (and (furry ?x)
                         (meows ?x)
                         (has ?x claws))
                    (cat ?x)))
```