First-Order Logic
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 others
  – **Relations** that hold among sets of objects
  – **Functions**, 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, more-than ...
User provides

- **Constant symbols** representing individuals in world
  - BarackObama, Green, John, 3, “John Smith”
- **Predicate symbols**, map individuals to truth values
  - greater(5,3)
  - green(Grass)
  - color(Grass, Green)
  - hasBrother(John, Robert)
- **Function symbols**, map individuals to individuals
  - father_of(SashaObama) = BarackObama
  - color_of(Sky) = Blue
FOL Provides

- **Variable symbols**
  - E.g., x, y, foo

- **Connectives**
  - Same as propositional logic: not (¬), and (∧), or (∨), implies (→), iff (↔)

- **Quantifiers**
  - Universal ∀x or (Ax)
  - Existential ∃x or (Ex)
Sentences: built from terms and atoms

- **term** (denoting a real-world individual) is a constant or variable symbol, or n-place function of n terms, e.g.:
  - Constants: john, umbc
  - Variables: x, y, z
  - Functions: mother_of(john), phone(mother(x))

- **Ground terms** have no variables in them
  - **Ground**: john, father_of(father_of(john))
  - **Not Ground**: father_of(X)
Sentences: built from terms and atoms

• **atomic sentences** (which are either true or false) are an n-place predicate of n terms, e.g.:
  – green(Kermit)
  – between(Philadelphia, Baltimore, DC)
  – loves(X, mother(X))

• **complex sentences** are formed from atomic sentences connected by logical connectives:
  \( \neg P, P \lor Q, P \land Q, P \rightarrow Q, P \leftrightarrow Q \)
  where P and Q are sentences
What do atomic sentences mean?

• Unary predicates typically encode a types
  – Dolphin(flipper): flipper is a kind of dolphin
  – Green(kermit): kermit is a kind of green thing
  – Integer(x): x is a kind of integer

• Non-unary predicates typically encode relations
  – Loves(john, mary)
  – Greater_than(2, 1)
  – Between(newYork, philadelphia, baltimore)
  – hasName(John, “John Smith”)
Ontology

- Designing a logic representation is similar to modeling in an object-oriented language.
- An **ontology** is a “formal naming and definition of the types, properties and relations of entities for a domain of discourse.”
- See [schema.org](http://schema.org) as for an ontology that’s used by search engines to add semantic data to web sites.
Sentences: built from terms and atoms

- **quantified sentences** adds quantifiers $\forall$ and $\exists$
  - $\forall x \text{ loves}(x, \text{mother}(x))$
  - $\exists x \text{ number}(x) \land \text{greater}(x, 100), \text{prime}(x)$

- A **well-formed formula (wff)** is a sentence with no *free* variables; all variables are *bound* by either a universal or existential *quantifier*.

  \[ \text{In } (\forall x)P(x, y) \quad x \text{ is bound and } y \text{ is free} \]
Quantifiers

• **Universal quantification**
  
  – $(\forall x)P(x)$ means $P$ holds for all values of $x$ in domain associated with variable
  
  – E.g., $(\forall x) \text{dolphin}(x) \rightarrow \text{mammal}(x)$

• **Existential quantification**
  
  – $(\exists x)P(x)$ means $P$ holds for some value of $x$ in domain associated with variable
  
  – E.g., $(\exists x) \text{mammal}(x) \land \text{lays_eggs}(x)$

  – This lets us make a statement about some object without identifying it
Quantifiers (1)

• Universal quantifiers typically used with implies to form rules:

  \[ \forall x \text{ student}(x) \rightarrow \text{smart}(x) \]

  Meaning: All students are smart

• Universal quantification rarely used to make statements about every individual in world:

  \[ \forall x \text{ student}(x) \land \text{smart}(x) \]

  Meaning: Everything in the world is a student and is smart
• Existential quantifiers usually used with **and** to specify a list of properties about an individual

*Logic:* $(\exists x) \text{ student}(x) \land \text{ smart}(x)$

*Meaning:* There is a student who is smart

• Common mistake: represent this in FOL as:

*Logic:* $(\exists x) \text{ student}(x) \rightarrow \text{ smart}(x)$

*Meaning:* ?
Quantifiers (2)

• Existential quantifiers usually used with and to specify a list of properties about an individual

  Logic: (\(\exists x\)) student(x) \(\land\) smart(x)

  Meaning: There is a student who is smart

• Common mistake: represent this in FOL as:

  Logic: (\(\exists x\)) student(x) \(\rightarrow\) smart(x)

  \(P \rightarrow Q = \sim P \lor Q\)

  \(\exists x\) student(x) \(\rightarrow\) smart(x) = \(\exists x\) \(\sim\)student(x) \(\lor\) smart(x)

  Meaning: There’s something that is not a student or is smart
Quantifier Scope

• FOL sentences have structure, like programs
• In particular, variables in a sentence have a **scope**
• For example, suppose we want to say
  – everyone who is alive loves someone
  – $(\forall x) \text{alive}(x) \rightarrow (\exists y) \text{loves}(x, y)$
• Here’s how we scope the variables

$$ (\forall x) \text{alive}(x) \rightarrow (\exists y) \text{loves}(x, y) $$

**Scope of x**

**Scope of y**
Quantifier Scope

• Switching order of universal quantifiers does not change the meaning
  - $(\forall x)(\forall y)P(x,y) \leftrightarrow (\forall y)(\forall x) P(x,y)$
  - Dogs hate cats (i.e., all dogs hate all cats)

• You can switch order of existential quantifiers
  - $(\exists x)(\exists y)P(x,y) \leftrightarrow (\exists y)(\exists x) P(x,y)$
  - A cat killed a dog

• Switching order of universal and existential quantifiers does change meaning:
  - Everyone likes someone: $(\forall x)(\exists y) \text{likes}(x,y)$
  - Someone is liked by everyone: $(\exists y)(\forall x) \text{likes}(x,y)$
def verify1():
    # Everyone likes someone: ( ∀x)(∃y) likes(x,y)
    for p1 in people():
        foundLike = False
        for p2 in people():
            if likes(p1, p2):
                foundLike = True
                break
    if not foundLike:
        print(p1, ‘does not like anyone 😞’)
        return False
    return True

Every person has at least one individual that they like.
def verify2():
    # Someone is liked by everyone: $(\exists y)(\forall x) \text{likes}(x,y)$
    for p2 in people():
        foundHater = False
        for p1 in people():
            if not likes(p1, p2):
                foundHater = True
                break
        if not foundHater:
            print(p2, 'is liked by everyone 😊')
    return True
return False
Connections between $\forall$ and $\exists$

- We can relate sentences involving $\forall$ and $\exists$ using extensions to **De Morgan’s laws**:

  1. $(\forall x) \neg P(x) \iff \neg (\exists x) P(x)$
  2. $\neg (\forall x) P(x) \iff (\exists x) \neg P(x)$
  3. $(\forall x) P(x) \iff \neg (\exists x) \neg P(x)$
  4. $(\exists x) P(x) \iff \neg (\forall x) \neg P(x)$

- Examples

  1. All dogs don’t like cats $\iff$ No dog likes cats
  2. Not all dogs bark $\iff$ There is a dog that doesn’t bark
  3. All dogs sleep $\iff$ There is no dog that doesn’t sleep
  4. There is a dog that talks $\iff$ Not all dogs can’t talk
Notational differences

- **Different symbols** for and, or, not, implies, ...
  - $\forall \exists \implies \iff \land \lor \neg \subseteq$
  - $-p \lor (q \land r)$
  - $-p + (q * r)$

- **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))`
Translating English to FOL

Every gardener likes the sun
\[ \forall x \text{ gardener}(x) \rightarrow \text{likes}(x, \text{Sun}) \]

All purple mushrooms are poisonous
\[ \forall x \ (\text{mushroom}(x) \land \text{purple}(x)) \rightarrow \text{poisonous}(x) \]

No purple mushroom is poisonous (two ways)
\[ \neg \exists x \ \text{purple}(x) \land \text{mushroom}(x) \land \text{poisonous}(x) \]
\[ \forall x \ (\text{mushroom}(x) \land \text{purple}(x)) \rightarrow \neg \text{poisonous}(x) \]
Translating English to FOL

There are (at least) two purple mushrooms

$$\exists x \ \exists y \ \text{mushroom}(x) \land \text{purple}(x) \land \text{mushroom}(y) \land \text{purple}(y) \land \neg (x=y)$$

There are exactly two purple mushrooms

$$\exists x \ \exists y \ \text{mushroom}(x) \land \text{purple}(x) \land \text{mushroom}(y) \land \text{purple}(y) \land \neg (x=y) \land$$

$$\forall z \ (\text{mushroom}(z) \land \text{purple}(z)) \rightarrow ((x=z) \lor (y=z))$$

Trump is not short

$$\neg \text{short(Trump)}$$
Translating English to FOL

What do these mean?

• You can fool some of the people all of the time

• You can fool all of the people some of the time
Translating English to FOL

What do these mean?

Both English statements are ambiguous

• You can fool some of the people all of the time
  There is a nonempty subset of people so easily fooled that you can fool that subset every time*
  For any given time, there is a non-empty subset at that time that you can fool

• You can fool all of the people some of the time
  There are one or more times when it’s possible to fool everyone*
  Everybody can be fooled at some point in time

* Most common interpretation, I think
Some terms we will need

• \texttt{person(x)}: True iff \( x \) is a person

• \texttt{time(t)}: True iff \( t \) is a point in time

• \texttt{canFool(x, t)}: True iff \( x \) can be fooled at time \( t \)

Note: \texttt{iff} = \textit{if and only if} = \leftrightarrow
You can fool some of the people all of the time

There is a nonempty group of people so easily fooled that you can fool that group every time*

≡ There’s a person that you can fool every time

∃x ∀t  \( \text{person}(x) \land \text{time}(t) \rightarrow \text{canFool}(x, t) \)

For any given time, there is a non-empty group at that time that you can fool

≡ For every time, there is a person at that time that you can fool

∀t ∃x  \( \text{person}(x) \land \text{time}(t) \rightarrow \text{canFool}(x, t) \)

* Most common interpretation, I think
Translating English to FOL

You can fool all of the people some of the time

There are one or more times when it’s possible to fool everyone*

\[ \exists t \forall x \text{time}(t) \land \text{person}(x) \rightarrow \text{canFool}(x, t) \]

Everybody can be fooled at some point in time

\[ \forall x \exists t \text{person}(x) \land \text{time}(t) \rightarrow \text{canFool}(x, t) \]

* Most common interpretation, I think
Simple genealogy KB in FOL

Design a knowledge base using FOL that

• Has facts of immediate family relations, e.g., spouses, parents, etc.
• Defines of more complex relations (ancestors, relatives)
• Detect conflicts, e.g., you are your own parent
• Infers relations, e.g., grandparent from parent
• Answers queries about relationships between people
How do we approach this?

• Design an initial ontology of types, e.g.
  – e.g., person, man, woman, male, female
• Extend ontology by defining relations, e.g.
  – spouse, has_child, has_parent
• Add general constraints to relations, e.g.
  – spouse(X,Y) => ~ X = Y
  – spouse(X,Y) => person(X), person(Y)
• Add FOL sentences for inference, e.g.
  – spouse(X,Y) ⇔ spouse(Y,X)
  – man(X) ⇔ person(X) ∧ male(X)
Example: A simple genealogy KB by FOL

• Predicates:
  – parent(x, y), child(x, y), father(x, y), daughter(x, y), etc.
  – spouse(x, y), husband(x, y), wife(x, y)
  – ancestor(x, y), descendant(x, y)
  – male(x), female(y)
  – relative(x, y)

• Facts:
  – husband(Joe, Mary), son(Fred, Joe)
  – spouse(John, Nancy), male(John), son(Mark, Nancy)
  – father(Jack, Nancy), daughter(Linda, Jack)
  – daughter(Liz, Linda)
  – etc.
Example Axioms

(∀x,y) parent(x, y) ↔ child (y, x)
(∀x,y) father(x, y) ↔ parent(x, y) ∧ male(x) ; similar for mother(x, y)
(∀x,y) daughter(x, y) ↔ child(x, y) ∧ female(x) ; similar for son(x, y)
(∀x,y) husband(x, y) ↔ spouse(x, y) ∧ male(x) ; similar for wife(x, y)
(∀x,y) spouse(x, y) ↔ spouse(y, x) ; spouse relation is symmetric
(∀x,y) parent(x, y) → ancestor(x, y)
(∀x,y)(∃z) parent(x, z) ∧ ancestor(z, y) → ancestor(x, y)
(∀x,y) descendant(x, y) ↔ ancestor(y, x)
(∀x,y)(∃z) ancestor(z, x) ∧ ancestor(z, y) → relative(x, y)
(∀x,y) spouse(x, y) → relative(x, y) ; related by marriage
(∀x,y)(∃z) relative(z, x) ∧ relative(z, y) → relative(x, y) ; transitive
(∀x,y) relative(x, y) ↔ relative(y, x) ; symmetric
Axioms, definitions and theorems

- **Axioms**: facts and rules that capture (important) facts & concepts in a domain; axioms are used to prove theorems
  - Mathematicians dislike unnecessary (dependent) axioms, i.e. ones that can be derived from others
  - Dependent axioms can make reasoning faster, however
  - Choosing a good set of axioms is a design problem

- A **definition** of a predicate is of the form “p(X) ↔ ...” and can be decomposed into two parts
  - **Necessary** description: “p(x) → ...”
  - **Sufficient** description “p(x) ← ...”
  - Some concepts have definitions (e.g., triangle) and some don’t (e.g., person)
More on definitions

Example: define father(x, y) by parent(x, y) and male(x)

• parent(x, y) is a necessary (but not sufficient) description of father(x, y)
  \[ \text{father}(x, y) \rightarrow \text{parent}(x, y) \]

• parent(x, y) ^ male(x) ^ age(x, 35) is a sufficient (but not necessary) description of father(x, y):
  \[ \text{father}(x, y) \leftarrow \text{parent}(x, y) ^ \text{male}(x) ^ \text{age}(x, 35) \]

• parent(x, y) ^ male(x) is a necessary and sufficient description of father(x, y)
  \[ \text{parent}(x, y) ^ \text{male}(x) \leftrightarrow \text{father}(x, y) \]
More on definitions

S(x) is a necessary condition of P(x)

\[ \forall x \ P(x) \implies S(x) \]

# all Ps are Ss

S(x) is a sufficient condition of P(x)

\[ \forall x \ P(x) \leq S(x) \]

# all Ps are Ss

S(x) is a necessary and sufficient condition of P(x)

\[ \forall x \ P(x) \iff S(x) \]

# all Ps are Ss

# all Ss are Ps
Higher-order logic

• FOL only lets us quantify over variables, and variables can only range over objects

• HOL allows us to quantify over relations, e.g.
  “two functions are equal iff they produce the same value for all arguments”

\[ \forall f \forall g \ (f = g) \iff (\forall x \ f(x) = g(x)) \]

• E.g.: (quantify over predicates)

\[ \forall r \ \text{transitive}(r) \rightarrow (\forall xyz \ r(x,y) \land r(y,z) \rightarrow r(x,z)) \]

• More expressive, but reasoning is undecidable, in general
Expressing uniqueness

• Often want to say that there is a single, unique object that satisfies a condition

• There exists a unique x such that king(x) is true
  – $\exists x \text{ king}(x) \land \forall y (\text{king}(y) \rightarrow x=y)$
  – $\exists x \text{ king}(x) \land \neg \exists y (\text{king}(y) \land x \neq y)$
  – $\exists! x \text{ king}(x)$

• Every country has exactly one ruler
  – $\forall c \text{ country}(c) \rightarrow \exists! r \text{ ruler}(c,r)$

• Iota operator: $\iota x P(x)$ means “the unique x such that p(x) is true”
  – The unique ruler of Freedonia is dead
  – dead($\iota x \text{ ruler}(\text{freedonia},x)$)
Examples of FOL in use

• Semantics of W3C’s **Semantic Web** stack (RDF, RDFS, OWL) is defined in FOL

• **OWL** Full is equivalent to FOL

• Other OWL profiles support a subset of FOL and are more efficient

• The semantics of **schema.org** is only defined in natural language text

• **Wikidata**’s knowledge graph (and Google’s) has a richer schema
FOL Summary

• First order logic (FOL) introduces predicates, functions and quantifiers

• More expressive, but reasoning more complex
  – Reasoning in propositional logic is NP hard, FOL is semi-decidable

• Common AI knowledge representation language
  – Other KR languages (e.g., OWL) are often defined by mapping them to FOL

• FOL variables range over objects
  – HOL variables range over functions, predicates or sentences