CS 482/682 – AI: First-Order Logic

Fall 2021

The Language of Thought

- Natural languages are very expressive
- If we could uncover the rules for natural languages, we could use them in representation and reasoning systems
- However,
  - Natural languages are context-dependent and ambiguous
  - Pinker, 1995: When people think about spring, surely they are not confused as to whether they are thinking about a season or something that goes boing—and if one word can correspond to two thoughts, thoughts can't be words.
  - People interpret the words they read and form an internal nonverbal representation, and that the exact words are not consequential.

Pros and Cons of Propositional Logic

- Propositional logic is declarative
- Propositional logic allows partial/disjunctive/negated information
  - (unlike most data structures and databases)
- Propositional logic is compositional:
  - meaning of $B_{1,1} \wedge P_{1,2}$ is derived from meaning of $B_{1,1}$ and of $P_{1,2}$
- Meaning in propositional logic is context-independent
  - (unlike natural language, where meaning depends on context)
- Propositional logic has very limited expressive power
  - (unlike natural language)
  - E.g., cannot say "pits cause breezes in adjacent squares," except by writing one sentence for each square

Combining Formal and Natural Languages

- When we look at natural language, most obvious elements are
  - **Objects** (squares, pits, wumpuses)
  - Verbs with adjectives and adverbs that refer to **relations** among objects
  - Some relations are **functions**
- **Objects**: people, houses, numbers, theories, Ronald McDonald, colors, baseball games, wars, centuries ...
- **Relations**: these can be unary relations or properties such as red, round, bogus, prime, multistoried ..., or more general n-ary relations such as brother of, bigger than, inside, part of, has color, occurred after, owns, comes between, ...
- **Functions**: father of, best friend, one more than, beginning of ...

Combining Formal and Natural Languages

- Almost any assertion can be thought of as referring to objects and properties or relations
    - "One plus two equals three".
        - Objects: one, two, three, one plus two;
        - Relation: equals;
        - Function: plus. ("One plus two" is a name for the object that is obtained by applying the function "plus" to the objects "one" and "two." "Three" is another name for this object.)
    - "Squares neighboring the wumpus are smelly". Objects: wumpus, squares; Property: smelly; Relation: neighboring.
    - "Evil King John ruled England in 1200." Objects: John, England, 1200; Relation: ruled during; Properties: evil, king.

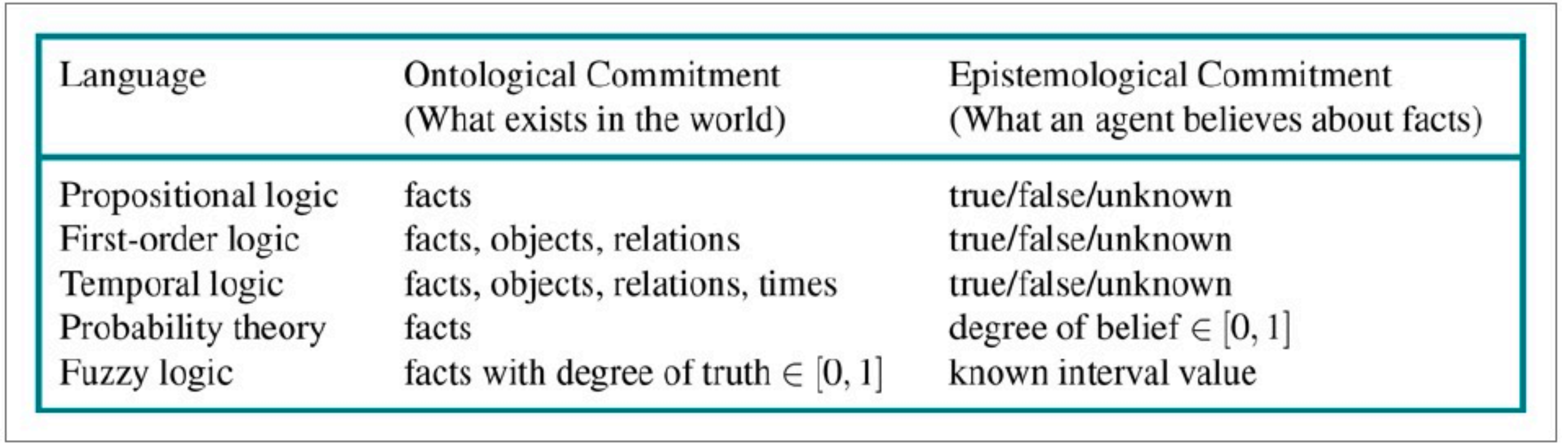
Ontological Commitment

- Ontological commitment
  - Propositional logic: there are facts that either hold or do not hold
  - First-order logic: the world consists of objects with certain relations among them that do or do not hold
  - This is powerful in domains where every proposition has clear boundaries (yes/no answer)
- Is Vienna a large city? Is that person tall?
  - Depends on who you ask, and the answer might be “kind of”
  - Fuzzy logic, in which propositions have a degree of truth between 0 and 1
  - However, it is harder to do $A \land B$, thus fuzzy logic needs different rules

Epistemological Commitments

- Epistemological: relating to the theory of knowledge
- Propositional and first-order logic: yes/no
- Fuzzy logic: a degree of truth
- Temporal logic: facts hold at a particular times and those times are ordered
- Higher-order logic: relations and functions are objects themselves
- Probability theory: a degree of belief, or subjective likelihood, ranging from 0 (total disbelief) to 1 (total belief).
    - Some fuzzy systems allow uncertainty (degree of belief) about degrees of truth
    - A probabilistic wumpus-world agent might believe that the wumpus is in [1,3] with probability 0.75 and in [2,3] with p=0.25, although the wumpus is definitely in on particular square.

Figure 8.1



Language	Ontological Commitment (What exists in the world)	Epistemological Commitment (What an agent believes about facts)
Propositional logic	facts	true/false/unknown
First-order logic	facts, objects, relations	true/false/unknown
Temporal logic	facts, objects, relations, times	true/false/unknown
Probability theory	facts	degree of belief $\in [0, 1]$
Fuzzy logic	facts with degree of truth $\in [0, 1]$	known interval value

Formal languages and their ontological and epistemological commitments.

Syntax of FOL: Basic Elements

Constants	$KingJohn,\ 2,\ UCB, \ldots$
Predicates	$Brother,\ >, \ldots$
Functions	$Sqrt,\ LeftLegOf, \ldots$
Variables	$x,\ y,\ a,\ b, \ldots$
Connectives	$\wedge\ \vee\ \neg\ \Rightarrow\ \Leftrightarrow$
Equality	$=$
Quantifiers	$\forall\ \exists$

FOL: Atomic Sentences

$$\begin{array}{rcl} \text{Atomic sentence} & = & predicate(term_1, \ldots, term_n) \\ & & \text{or } term_1 = term_2 \\ \\ \text{Term} & = & function(term_1, \ldots, term_n) \\ & & \text{or } constant \text{ or } variable \end{array}$$

E.g., $Brother(KingJohn, RichardTheLionheart)$
$> (Length(LeftLegOf(Richard)), Length(LeftLegOf(KingJohn)))$

Complex Sentences

Complex sentences are made from atomic sentences using connectives

$$\neg S, \quad S_1 \wedge S_2, \quad S_1 \vee S_2, \quad S_1 \Rightarrow S_2, \quad S_1 \Leftrightarrow S_2$$

E.g. $Sibling(KingJohn, Richard) \Rightarrow Sibling(Richard, KingJohn)$

$>(1,2) \vee \leq(1,2)$

$>(1,2) \wedge \neg{>}(1,2)$

Symbols and Interpretations

$$
\begin{array}{rcl}
\mathit{Sentence} & \rightarrow & \mathit{AtomicSentence} \mid \mathit{ComplexSentence} \\
\mathit{AtomicSentence} & \rightarrow & \mathit{Predicate} \mid \mathit{Predicate}(\mathit{Term}, \ldots) \mid \mathit{Term} = \mathit{Term} \\
\mathit{ComplexSentence} & \rightarrow & (\ \mathit{Sentence}\ ) \\
 & \mid & \neg\, \mathit{Sentence} \\
 & \mid & \mathit{Sentence} \land \mathit{Sentence} \\
 & \mid & \mathit{Sentence} \lor \mathit{Sentence} \\
 & \mid & \mathit{Sentence} \ \Rightarrow\ \mathit{Sentence} \\
 & \mid & \mathit{Sentence} \ \Leftrightarrow\ \mathit{Sentence} \\
 & \mid & \mathit{Quantifier}\ \mathit{Variable}, \ldots\ \mathit{Sentence} \\
 & & \\
\mathit{Term} & \rightarrow & \mathit{Function}(\mathit{Term}, \ldots) \\
 & \mid & \mathit{Constant} \\
 & \mid & \mathit{Variable} \\
 & & \\
\mathit{Quantifier} & \rightarrow & \forall \mid \exists \\
\mathit{Constant} & \rightarrow & A \mid X_1 \mid \mathit{John} \mid \cdots \\
\mathit{Variable} & \rightarrow & a \mid x \mid s \mid \cdots \\
\mathit{Predicate} & \rightarrow & \mathit{True} \mid \mathit{False} \mid \mathit{After} \mid \mathit{Loves} \mid \mathit{Raining} \mid \cdots \\
\mathit{Function} & \rightarrow & \mathit{Mother} \mid \mathit{LeftLeg} \mid \cdots \\
\textsc{Operator Precedence} & : & \neg, =, \land, \lor, \Rightarrow, \Leftrightarrow
\end{array}
$$

The syntax of first-order logic with equality, specified in Backus–Naur form (see page 1030 if you are not familiar with this notation). Operator precedences are specified, from highest to lowest. The precedence of quantifiers is such that a quantifier holds over everything to the right of it.

Syntax and Semantics of First-Order Logic

Sentence → *AtomicSentence*
| *Sentence Connective Sentence*
| *Quantifier Variable , … , Sentence*
| ¬*Sentence*
| *(Sentence)*
AtomicSentence → *Predicate*(*Term*,...)
| *Term* = *Term*

Term → *Function* (*Term*,...)
| *Constant*
| *Variable*

Connective → | ⇒ | ∧ | ∨ | ⇔
Quantifier → ∀ | ∃
Variable → a | b | c | …

Function → *Mother* | *LeftLegOf* | ...
Predicate → *Before* | *HasColor* | *Raining* | ...
Constant → *A* | X_1 | *John* | ...

- Constant Symbols (A, B, John, ...)
  - A symbol names exactly one object
  - But each object might have multiple names (and some objects might not have a name)
- Predicate Symbols (Round, Brother,...)
  - Defined by a set of tuples of objects that satisfy the predicate
- Function Symbols (Cosine, FatherOf, ...)
  - A relation in which any given object is related to exactly one other object by this relation
  - Uniquely determines an object (without giving it a name)

Syntax and Semantics of First-Order Logic

Sentence → *AtomicSentence*
| *Sentence Connective Sentence*
| *Quantifier Variable , … , Sentence*
| ¬*Sentence*
| *(Sentence)*

AtomicSentence → *Predicate(Term,…)*
| *Term* = *Term*

Term → *Function (Term,…)*
| *Constant*
| *Variable*

Connective → | ⇒ | ∧ | ∨ | ⇔
Quantifier → ∀ | ∃
Variable → a | b | c | …

Function → *Mother* | *LeftLegOf* | …
Predicate → *Before* | *HasColor* | *Raining* | …
Constant → *A* | X_1 | *John* | …

- Variables (a, b, c, ...)
  - Stand for an object (without naming it)
- Terms (John, FatherOf(John), a,...)
  - A logical expression that refers to an object
  - Can be a constant or a variable
  - Can be a function of a list of terms

Syntax and Semantics of First-Order Logic

Sentence → *AtomicSentence*
| *Sentence Connective Sentence*
| *Quantifier Variable , … , Sentence*
| ¬*Sentence*
| *(Sentence)*

AtomicSentence → *Predicate*(*Term*,...)
| *Term* = *Term*

Term → *Function* (*Term*,...)
| *Constant*
| *Variable*

Connective → | ⇒ | ∧ | ∨ | ⇔
Quantifier → ∀ | ∃
Variable → a | b | c | …

Function → *Mother* | *LeftLegOf* | …
Predicate → *Before* | *HasColor* | *Raining* | …
Constant → *A* | X_1 | *John* | …

- Atomic Sentences via Predicates
- Examples:
  - Round(Coconut)
  - Brother(Cain, Abel)
  - Older(John, 35)
  - Square(Baseball)
- Assertions that represent a fact about the world
- Again, can be true or false given the state of the world

Syntax and Semantics of First-Order Logic

Sentence → *AtomicSentence*
| *Sentence Connective Sentence*
| *Quantifier Variable , … , Sentence*
| ¬*Sentence*
| *(Sentence)*
AtomicSentence → *Predicate*(*Term*,...)
| *Term* = *Term*

Term → *Function* (*Term*,...)
| *Constant*
| *Variable*

Connective → | ⇒ | ∧ | ∨ | ⇔
Quantifier → ∀ | ∃
Variable → a | b | c | …

Function → *Mother* | *LeftLegOf* | …
Predicate → *Before* | *HasColor* | *Raining* | …
Constant → *A* | X_1 | *John* | …

- Atomic Sentences via Equality
- Examples:
  - Father(John)=Henry
  - 1=Cosine(pi)
  - Three=Two
- Asserts that the two terms refer to the same real-world object

Syntax and Semantics of First-Order Logic

Sentence → *AtomicSentence*
| *Sentence Connective Sentence*
| *Quantifier Variable , … , Sentence*
| ¬*Sentence*
| *(Sentence)*

AtomicSentence → *Predicate*(*Term*,…)
| *Term* = *Term*

Term → *Function* (*Term*,…)
| *Constant*
| *Variable*

Connective → | ⇒ | ∧ | ∨ | ⇔
Quantifier → ∀ | ∃
Variable → a | b | c | …

Function → *Mother* | *LeftLegOf* | …
Predicate → *Before* | *HasColor* | *Raining* | …
Constant → *A* | X_1 | *John* | …

- Connectives (⇒ ∧ ∨ ⇔)
  - Work the same way as in predicate calculus
- Complex Sentence
  - Atomic Sentence
  - Connectives
  - Negated Sentence
  - Parentheses

Syntax and Semantics of First-Order Logic

Sentence → *AtomicSentence*
| *Sentence Connective Sentence*
| *Quantifier Variable , … , Sentence*
| ¬*Sentence*
| *(Sentence)*

AtomicSentence → *Predicate*(*Term*,...)
| *Term* = *Term*

Term → *Function* (*Term*,...)
| *Constant*
| *Variable*

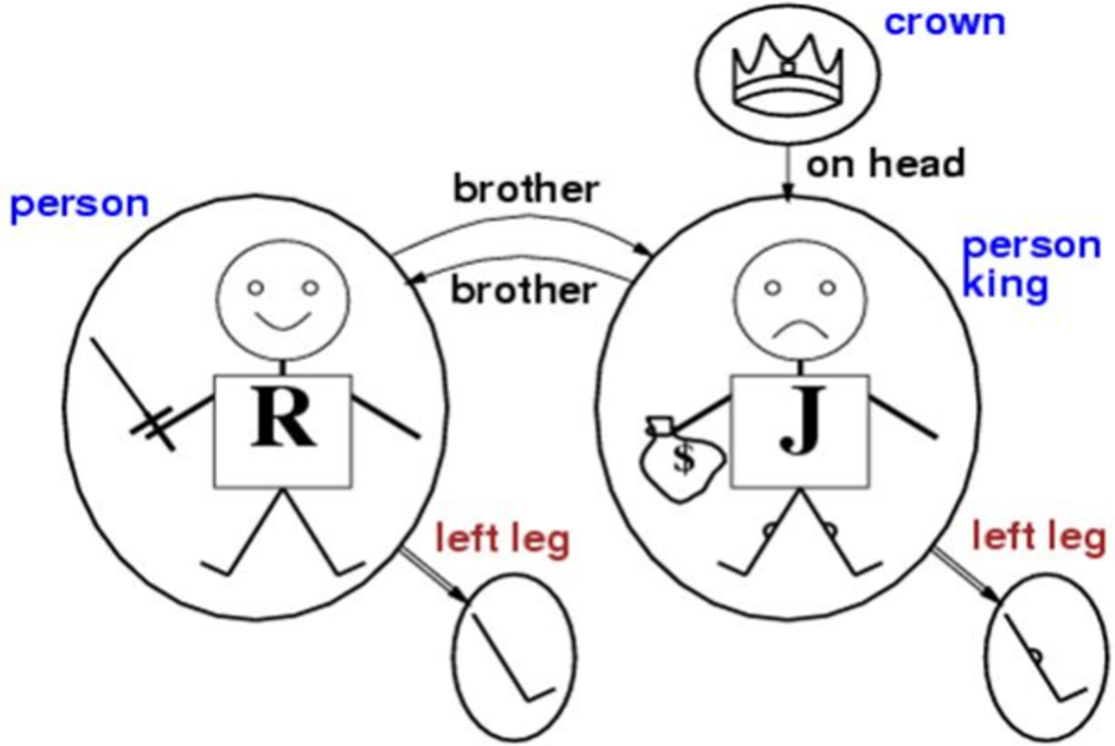
Connective → | ⇒ | ∧ | ∨ | ⇔
Quantifier → ∀ | ∃
Variable → a | b | c | …

Function → *Mother* | *LeftLegOf* | …
Predicate → *Before* | *HasColor* | *Raining* | …
Constant → *A* | X_1 | *John* | …

- Quantifiers (∃, ∀)
  - The real power of first-order logic
  - Express properties of entire collections of objects rather than having to enumerate all the objects by name
- Universal Quantifier (∀)
  - “all cats are mammals”
    ∀ x Cat(x) ⇒ Mammal(x)
- Existential Quantifier (∃)
  - “there exists a fish that can fly”
    ∃x Fish(x) ∧ CanFly(x)

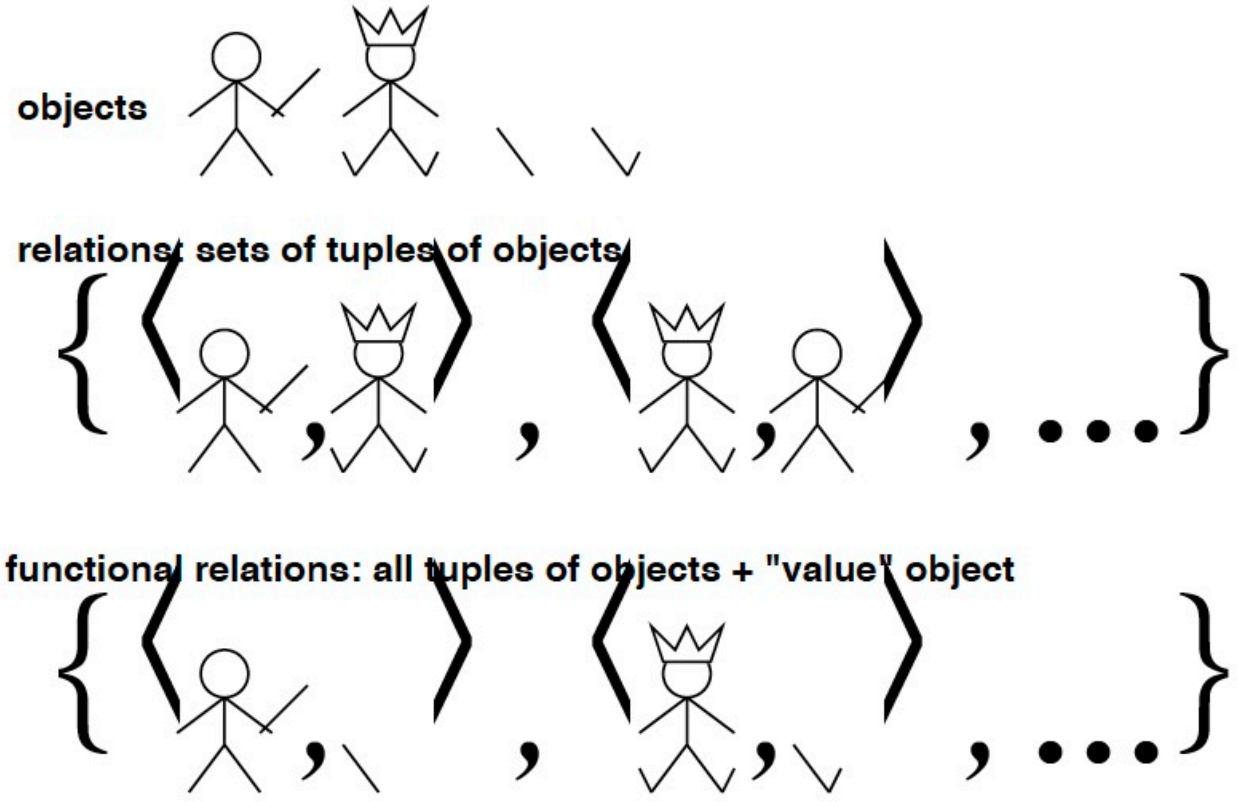
Models for FOL

- The models of a logical language are the formal structures that constitute the possible worlds under consideration
- Each model links the vocabulary of the logical sentences to elements of the possible world, so that the truth of any sentence can be determined
- Models of FOL:
    - There are objects inside
    - Domain is nonempty



5 objects, 2 binary relations (brother, on head), 3 unary relations (crown, king, person), 1 unary function (left leg)

Models for FOL



objects

relations: sets of tuples of objects

functional relations: all tuples of objects + "value" object

Universal Quantification

$\forall \langle variables \rangle \ \ \langle sentence \rangle$

Everyone at Berkeley is smart:
$\forall x \ \ At(x, Berkeley) \ \Rightarrow \ Smart(x)$

$\forall x \ \ P$ is equivalent to the <u>conjunction</u> of <u>instantiations</u> of P

$$\begin{aligned}
& At(KingJohn, Berkeley) \ \Rightarrow \ Smart(KingJohn) \\
\land \ & At(Richard, Berkeley) \ \Rightarrow \ Smart(Richard) \\
\land \ & At(Berkeley, Berkeley) \ \Rightarrow \ Smart(Berkeley) \\
\land \ & \ldots
\end{aligned}$$

Typically, $\Rightarrow$ is the main connective with $\forall$.
Common mistake: using $\land$ as the main connective with $\forall$:

$$\forall x \ \ At(x, Berkeley) \land Smart(x)$$

means "Everyone is at Berkeley and everyone is smart"

Universal Quantification (∀)

Makes a statement about all objects in the universe:

- $\forall x\ Cat(x) \Rightarrow Mammal(x)$ expands using conjunction:
  $Cat(Felix) \Rightarrow Mammal(Felix) \land$
  $Cat(Fluffy) \Rightarrow Mammal(Fluffy) \land$
  $Cat(Spot) \Rightarrow Mammal(Spot) \land$
  $Cat(Sylvester) \Rightarrow Mammal(Sylvester) \land \ldots$

- What if the universe includes non-cats?
  $Cat(Dave) \Rightarrow Mammal(Dave) \land Cat(Tree) \Rightarrow Mammal(Tree)\ \ldots$
  - Still OK… because if Cat(Dave) is false, then $Cat(Dave) \Rightarrow Mammal(Dave)$ is true

- Can we express "all cats are mammals" as
  $\forall x\ Cat(x) \land Mammal(x)$
  - No… requires that all objects are both cats and mammals

Existential Quantification

$\exists \langle variables \rangle \;\; \langle sentence \rangle$

Someone at Stanford is smart:
$\exists x \;\; At(x, Stanford) \land Smart(x)$

$\exists x \;\; P$ is equivalent to the disjunction of instantiations of P

$$\begin{aligned} & At(KingJohn, Stanford) \land Smart(KingJohn) \\ \lor \; & At(Richard, Stanford) \land Smart(Richard) \\ \lor \; & At(Stanford, Stanford) \land Smart(Stanford) \\ \lor \; & \ldots \end{aligned}$$

Typically, $\land$ is the main connective with $\exists$.
Common mistake: using $\Rightarrow$ as the main connective with $\exists$:

$$\exists x \;\; At(x, Stanford) \Rightarrow Smart(x)$$

is true if there is anyone who is not at Stanford!

Existential Quantification (∃)

Makes a statement about some object in the universe

- "Spot has a sister that is a cat" is expressed as
  $\exists x\ Sister(x,Spot) \land Cat(x)$

- Expands using disjunction:
  $(Sister(Fluffy, Spot) \land Cat(Fluffy)) \lor$
  $(Sister(Richard, Spot) \land Cat(Richard)) \lor$
  $(Sister(BigRock, Spot) \land Cat(BigRock)) \lor \ldots$

- What if multiple objects fulfill the requirements?
  - Still ok... True or True is still True

- Can you express this with an implication?
  $\exists x\ Sister(x,Spot) \Rightarrow Cat(x)$
  - Results in nonsense... if any object is not Spot's sister, then this relation is true

Nested Quantifiers

"If x is the parent of y, then y is the child of x"

$\forall x\ \forall y\ Parent(x,y) \Rightarrow Child(y,x)$

Syntactic sugar:

$\forall x,y\ Parent(x,y) \Rightarrow Child(y,x)$

"Everybody loves somebody"

$\forall x\ \exists y\ Loves(x,y)$

Is this the same as $\exists y\ \forall x\ Loves(x,y)$?

No... this sentence states that "there exists someone who is loved by everyone"

Connections between ∀ and ∃

- "Everyone dislikes parsnips" is equivalent to "there does not exist someone who likes parsnips":

  $\forall x\, \neg \text{Likes}(x,\text{Parsnips})$ is equivalent to
  $\neg \exists x\, \text{Likes}(x,\text{Parsnips})$

- Similarly, "Everyone likes Scheme" is equivalent to "there is no one who does not like Scheme"

  $\forall x\, \text{Likes}(x,\text{Scheme})$ is equivalent to
  $\neg \exists x\, \neg \text{Likes}(x,\text{Scheme})$

These quantifiers obey De Morgan's rules

Properties of Quantifiers

$\forall x\ \forall y$ is the same as $\forall y\ \forall x$ (why??)

$\exists x\ \exists y$ is the same as $\exists y\ \exists x$ (why??)

$\exists x\ \forall y$ is not the same as $\forall y\ \exists x$

$\exists x\ \forall y\ Loves(x, y)$
"There is a person who loves everyone in the world"

$\forall y\ \exists x\ Loves(x, y)$
"Everyone in the world is loved by at least one person"

Quantifier duality: each can be expressed using the other

$\forall x\ Likes(x, IceCream)$ $\neg\exists x\ \neg Likes(x, IceCream)$

$\exists x\ Likes(x, Broccoli)$ $\neg\forall x\ \neg Likes(x, Broccoli)$

Properties of Quantifiers

Because ∀ is really a conjunction over the universe of objects and ∃ is a disjunction, it should not be surprising that they obey De Morgan's rules. The De Morgan rules for quantified and unquantified sentences are as follows:

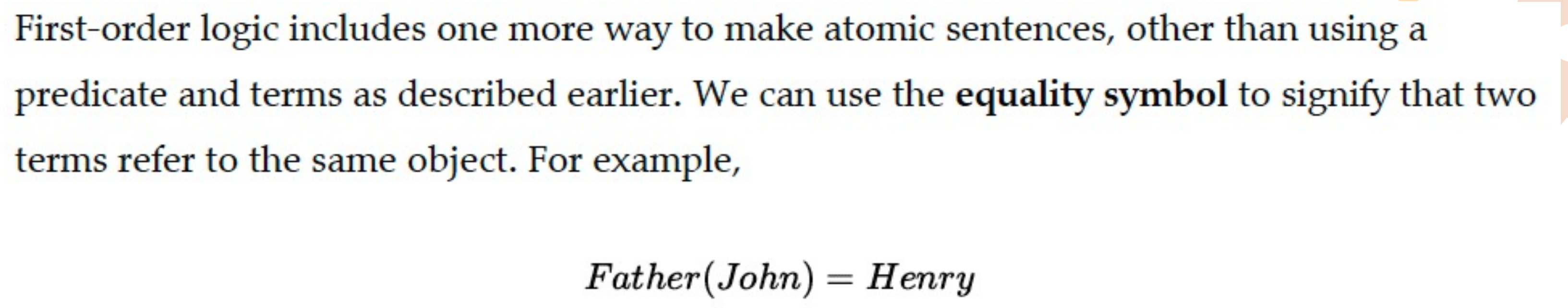
$$
\begin{array}{llllll}
\neg\exists x \quad P & \equiv & \forall x \quad \neg P & \qquad \neg(P \lor Q) & \equiv & \neg P \land \neg Q \\
\neg\forall x \quad P & \equiv & \exists x \quad \neg P & \qquad \neg(P \land Q) & \equiv & \neg P \lor \neg Q \\
\forall x \quad P & \equiv & \neg\exists x \quad \neg P & \qquad P \land Q & \equiv & \neg(\neg P \land \neg Q) \\
\exists x \quad P & \equiv & \neg\forall x \quad \neg P & \qquad P \lor Q & \equiv & \neg(\neg P \land \neg Q)\,.
\end{array}
$$

Thus, we do not really need both ∀ and ∃, just as we do not really need both ∧ and ∨. Still, readability is more important than parsimony, so we will keep both of the quantifiers.

Exercise

- Brothers are siblings
  - $\forall x, y\ Brother(x, y) \Rightarrow Sibling(x, y)$
- "Sibling" is symmetric
  - $\forall x, y\ Sibling(x, y) \Leftrightarrow Sibling(y, x)$
- One's mother is one's female parent
  - $\forall x, y\ Mother(x, y) \Leftrightarrow (Female(x) \wedge Parent(x, y))$
- A first cousin is a child of a parent's sibling
  - $\forall x, y\ FirstCousin(x, y) \Leftrightarrow \exists p, ps\ Parent(p, x) \wedge Sibling(ps, p) \wedge Parent(ps, y)$

Equality Symbol



First-order logic includes one more way to make atomic sentences, other than using a predicate and terms as described earlier. We can use the **equality symbol** to signify that two terms refer to the same object. For example,

$$Father(John) = Henry$$

- How to say that Richard has at least two brothers?
  - $\exists x, y \; Brother(x, Richard) \wedge Brother(y, Richard)$?
  - $\exists x, y \; Brother(x, Richard) \wedge Brother(y, Richard) \wedge \neg(x = y)$

$term_1 = term_2$ is true under a given interpretation
if and only if $term_1$ and $term_2$ refer to the same object

E.g., definition of (full) $Sibling$ in terms of $Parent$:

$$\forall\, x,y \;\; Sibling(x,y) \;\Leftrightarrow\; [\neg(x=y) \land \exists\, m,f \;\; \neg(m=f) \land Parent(m,x) \land Parent(f,x) \land Parent(m,y) \land Parent(f,y)]$$

Database Semantics

- Richard has two brothers: John and Geoffrey
    - $Brother(John, Richard) \land Brother(Geoffrey, Richard)$?
    - This does not completely capture the state of affairs
    - This one is true in a model where Richard has only one brother
    - This does not rule out model in which Richard has many more brothers
    - $Brother(John, Richard) \land Brother(Geoffrey, Richard) \land John \neq Geoffrey \land \forall x Brother(x, Richard) \Rightarrow (x = John \lor x = Geoffrey)$
- Database semantics assumptions:
    - Unique-names : every constant symbol refers to a distinct object
    - Closed-world : atomic sentences not known to be true are in fact false
    - Domain closure: each model contains no more domain elements than those named by the constant symbols

Higher-Order Logics

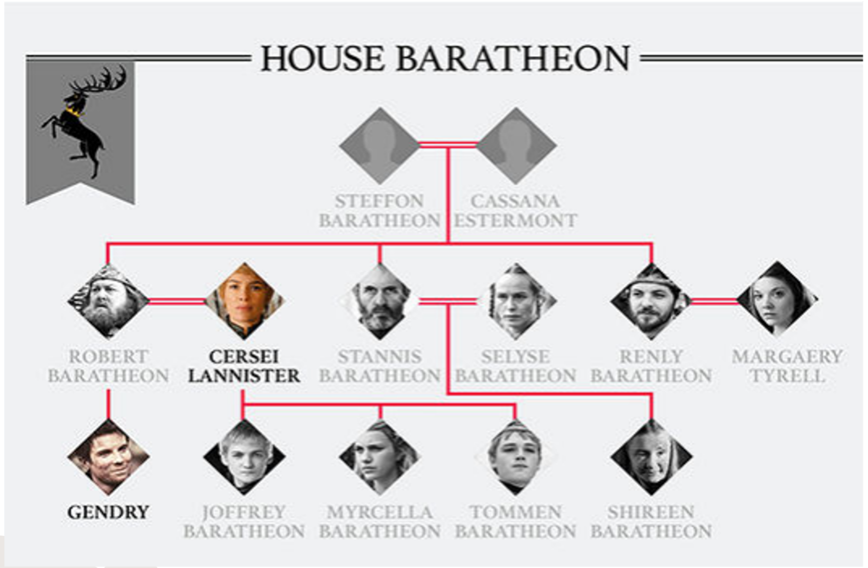
- “First-Order” Logic implies that you can quantify over ***objects***, but not over ***relations***
- Higher order logics allow quantification over relations and functions
    - Define equality as objects that have the same properties
      $\forall x,y\ (x=y) \Leftrightarrow (\forall p\ p(x) \Leftrightarrow p(y))$
    - or the equality of functions that give the same value for all arguments
      $\forall f,g\ (f=g) \Leftrightarrow (\forall x\ f(x)=g(x))$

Other Notations

- Notational variations exist (especially within other fields that use logic)
- Some other operators are also useful
  - Uniqueness quantifier
  - "Every student has exactly one advisor"
    $\forall x\ \text{Student}(x)\ \exists!\ y\ \text{Advisor}(y, x)$

- Uniqueness operator: $\exists!$

Advantages of Using First-Order Logic: Kinship Example



Define relationships in first-order logic:

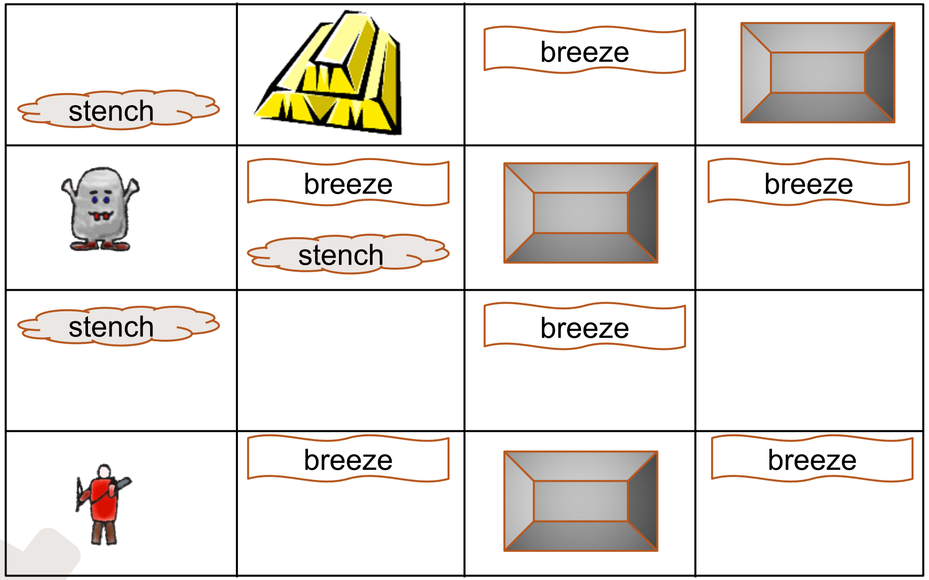
Your mother is your female parent
$\forall m,c\ Mother(c)=m \Leftrightarrow Female(m) \land Parent(m,c)$

Male and female are disjoint categories
$\forall x\ Male(x) \Leftrightarrow \neg Female(x)$

A grandparent is the parent of one's parent
$\forall g,c\ Grandparent(g,c) \Leftrightarrow \exists p\ Parent(g,p) \land Parent(p,c)$

Advantage: you can use these rules for any family tree, not just this example

Advantages of Using First-Order Logic: Wumpus Example



Consider an infinite or unknown board configuration

How many propositional rules are required?

First-order logic can handle this

Interaction with FOL KBs