Last time: Logic and Reasoning

- Knowledge Base (KB): contains a set of <u>sentences</u> expressed using a **knowledge representation language**
  - TELL: operator to add a sentence to the KB
  - ASK: to query the KB
- Logics are KRLs where conclusions can be drawn
  - Syntax
  - Semantics
- Entailment: KB |= a iff a is true in all worlds where KB is true
- Inference: KB $|-_i$ a = sentence a can be derived from KB using procedure *i*
  - Sound: whenever KB $|-_i$ a then KB |= a is true
  - Complete: whenever KB |= a then KB $|-_i$ a

Last Time: Syntax of propositional logic

Propositional logic is the simplest logic

The proposition symbols P_1, P_2 etc are sentences

If S is a sentence, $\neg S$ is a sentence

If S_1 and S_2 is a sentence, $S_1 \wedge S_2$ is a sentence

If S_1 and S_2 is a sentence, $S_1 \vee S_2$ is a sentence

If S_1 and S_2 is a sentence, $S_1 \Rightarrow S_2$ is a sentence

If S_1 and S_2 is a sentence, $S_1 \Leftrightarrow S_2$ is a sentence

This time

- **First-order logic**
  - Syntax
  - Semantics
  - Wumpus world example

- **Ontology (ont = ‘to be’; logica = ‘word’): kinds of things one can talk about in the language**

Why first-order logic?

- **We saw that propositional logic is limited because it only makes the ontological commitment that the world consists of facts.**
- **Difficult to represent even simple worlds like the Wumpus world;**

e.g.,
"don't go forward if the Wumpus is in front of you" takes 64 rules

First-order logic (FOL)

- **Ontological commitments:**
  - **Objects**: wheel, door, body, engine, seat, car, passenger, driver
  - **Relations**: Inside(car, passenger), Beside(driver, passenger)
  - **Functions**: ColorOf(car)
  - **Properties**: Color(car), IsOpen(door), IsOn(engine)

- **Functions are relations with single value for each object**

Semantics

there is a correspondence between

- functions, which return values
- predicates, which are true or false

Function: father_of(Mary) = Bill

Predicate: father_of(Mary, Bill)

Examples:

- “One plus two equals three”

  Objects:

  Relations:

  Properties:

  Functions:

- “Squares neighboring the Wumpus are smelly”

  Objects:

  Relations:

  Properties:

  Functions:

Examples:

- **"One plus two equals three"**

  Objects: one, two, three, one plus two

  Relations: equals

  Properties: --

  Functions: plus ("one plus two" is the name of the object obtained by applying function plus to one and two; three is another name for this object)

- **"Squares neighboring the Wumpus are smelly"**

  Objects: Wumpus, square

  Relations: neighboring

  Properties: smelly

  Functions: --

FOL: Syntax of basic elements

- Constant symbols: 1, 5, A, B, USC, JPL, Alex, Manos, …
- Predicate symbols: >, Friend, Student, Colleague, …
- Function symbols: +, sqrt, SchoolOf, TeacherOf, ClassOf, …
- Variables: *x, y, z, next, first, last,* …
- Connectives: $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$
- Quantifiers: $\forall$, $\exists$
- Equality: =

FOL: Atomic sentences

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

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

- **Examples:**
  - SchoolOf(Jack)=USC
  - Colleague(TeacherOf(Ben), TeacherOf(Jack))
  - >((+ x y), x)

FOL: Complex sentences

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

- **Examples:**
  - S1 ∧ S2, S1 ∨ S2, (S1 ∧ S2) ∨ S3, S1 ⇒ S2, S1⇔ S3
  - Colleague(Paul, Max) ⇒ Colleague(Max, Paul)
    Student(Alex, Paul) ⇒ Teacher(Paul, Alex)

Semantics of atomic sentences

- **Sentences in FOL are interpreted with respect to a model**
- **Model contains objects and relations among them**
- **Terms: refer to objects (e.g., Door, Alex, StudentOf(Paolo))**
  - Constant symbols: refer to objects
  - Predicate symbols: refer to relations
  - Function symbols: refer to functional Relations

- **An atomic sentence *predicate(term$_1$, …, term$_n$)* is true iff the relation referred to by *predicate* holds between the objects referred to by *term$_1$, …, term$_n$***

Example model

- **Objects:** **John, Jim, Mary, Alex, Dan, Joe, Anne, Rich**

- **Relation:** **sets of tuples of objects**
  {<John, Jim>, <Mary, Alex>, <Mary, James>, …}
  {<Dan, Joe>, <Anne, Mary>, <Mary, Joe>, …}

- **E.g.:**
  Parent relation -- {<John, Jim>, <Mary, Alex>, <Mary, James>}

  then Parent(John, Jim) is true
  Parent(John, Mary) is false

Quantifiers

- Expressing sentences about collections of objects without enumeration (naming individuals)
- E.g., All Trojans are clever

  Someone in the class is sleeping
- Universal quantification (for all): $\forall$
- Existential quantification (there exists): $\exists$

USC

Universal quantification (for all): ∀

∀ *<variables> <sentence>*

- ***"Every one in the cs561 class is smart"*:**
  $\forall x$ In(cs561, x) $\Rightarrow$ Smart(x)

- **∀ P corresponds to the conjunction of instantiations of P**
  In(cs561, Frank) $\Rightarrow$ Smart(Frank) $\wedge$
  In(cs561, Dan) $\Rightarrow$ Smart(Dan) $\wedge$
  …
  In(cs561, Ben) $\Rightarrow$ Smart(Ben)

Universal quantification (for all): $\forall$

- $\Rightarrow$ is a natural connective to use with $\forall$

- **Common mistake:** to use $\wedge$ in conjunction with $\forall$
  e.g: $\forall x \quad In(cs561, x) \wedge Smart(x)$
  means *"every one is in cs561 and everyone is smart"*

Existential quantification (there exists): ∃

$\exists$ *<variables> <sentence>*

- ***"Someone in the cs561 class is smart"*:**
  $\exists x \;\; \mathrm{In}(cs561, x) \wedge \mathrm{Smart}(x)$

- **$\exists$ P corresponds to the disjunction of instantiations of P**
  In(cs561, Frank) $\wedge$ Smart(Frank) $\vee$
  In(cs561, Dan) $\wedge$ Smart(Dan) $\vee$
  …
  In(cs561, Jack) $\wedge$ Smart(Jack)

USC

Existential quantification (there exists): $\exists$

- $\wedge$ **is a natural connective to use with** $\exists$

- **Common mistake: to use** $\Rightarrow$ **in conjunction with** $\exists$
  e.g: $\exists\, x \quad \mathrm{In}(cs561, x) \Rightarrow \mathrm{Smart}(x)$
  is true if there is anyone that is not in cs561!

  (remember, false $\Rightarrow$ **true is valid).**

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"

Not all by one person but each one at least by one

Quantifier duality: each can be expressed using the other

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

Proof?

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

Proof

- **In general we want to prove:**

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

- $\forall x \, P(x) = \neg(\neg(\forall x \, P(x))) = \neg(\neg(P(x1) \wedge P(x2) \wedge \ldots \wedge P(xn)))$
  $= \neg(\neg P(x1) \vee \neg P(x2) \vee \ldots \vee \neg P(xn)))$

- $\exists x \neg P(x) = \neg P(x1) \vee \neg P(x2) \vee \ldots \vee \neg P(xn)$

- $\neg\exists x \neg P(x) = \neg(\neg P(x1) \vee \neg P(x2) \vee \ldots \vee \neg P(xn))$

Example sentences

- Brothers are siblings
- Sibling is transitive
- One’s mother is one’s sibling’s mother
- A first cousin is a child of a parent’s sibling
-

Example sentences

- Brothers are siblings
  $\forall$ x, y Brother(x, y) $\Rightarrow$ Sibling(x, y)
- Sibling is transitive

  $\forall$ x, y, z Sibling(x, y) $\wedge$ Sibling(y, z) $\Rightarrow$ Sibling(x, z)

- One's mother is one's sibling's mother

  $\forall$ m, c, d Mother(m, c) $\wedge$ Sibling(c, d) $\Rightarrow$ Mother(m, d)

- A first cousin is a child of a parent's sibling

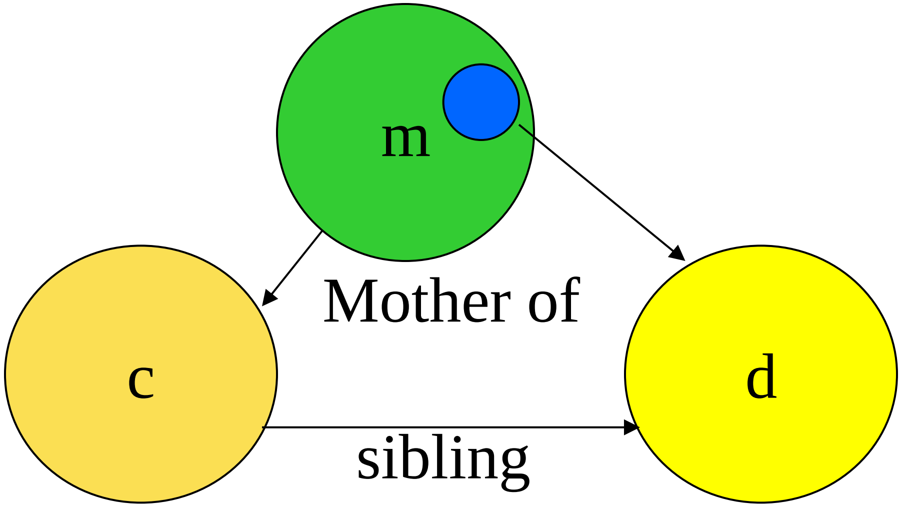
  $\forall$ c, d FirstCousin(c, d)
  $\Rightarrow$ $\exists$ p, q Parent(p, d) $\wedge$ Sibling(p, q) $\wedge$ Parent(q, c)

Example sentences

One's mother is one's sibling's mother

$\forall$ m, c,d   Mother(m, c) $\wedge$ Sibling(c, d) $\Rightarrow$ Mother(m, d)

- **$\forall$ c,d $\exists$m Mother(m, c) $\wedge$ Sibling(c, d) $\Rightarrow$ Mother(m, d)**



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)

Translating English to FOL

- **You can fool all of the people some of the time.**

  $\forall$ `x person(x) => `$\exists$` t time(t) ^ can-fool(x,t)`

- **All purple mushrooms are poisonous.**

  $\forall$ `x (mushroom(x) ^ purple(x)) => poisonous(x)`

Caution with nested quantifiers

- $\forall x \exists y P(x,y)$ is the same as $\forall x (\exists y P(x,y))$ which means "for every x, it is true that there exists y such that P(x,y)"

- $\exists y \forall x P(x,y)$ is the same as $\exists y (\forall x P(x,y))$ which means "there exists a y, such that it is true that for every x P(x,y)"

Translating English to FOL…

- **No purple mushroom is poisonous.**

$\neg (\exists\ x)\ \mathrm{purple}(x)\ \wedge\ \mathrm{mushroom}(x)\ \wedge\ \mathrm{poisonous}(x)$

or, equivalently,

$(\forall\ x)\ (\mathrm{mushroom}(x)\ \wedge\ \mathrm{purple}(x)) \Rightarrow \neg\mathrm{poisonous}(x)$

Translating English to FOL…

- **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))

- **Debbie is not tall.**

¬tall(Debbie)

Translating English to FOL…

- **X is above Y if X is on directly on top of Y or else 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)))

USC