Lecture 5b: First Order Logic

CSCI 360

Introduction to Artificial Intelligence

USC

Here is where we are…

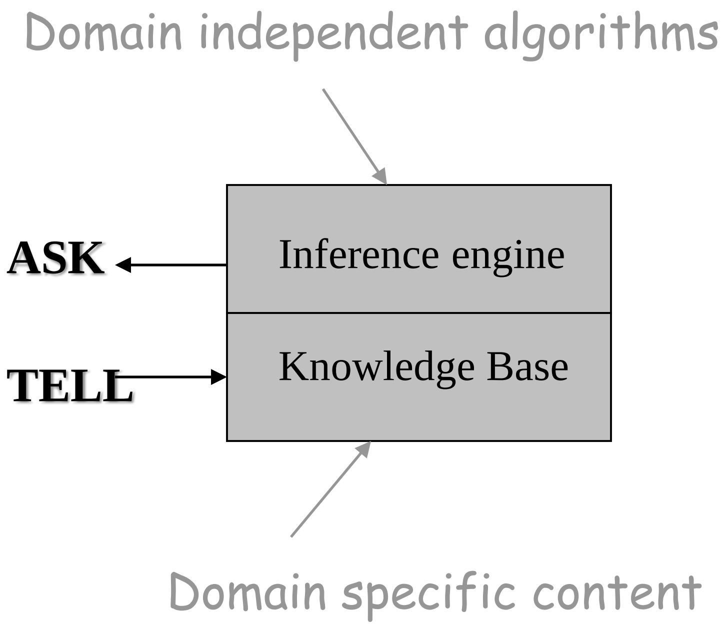
Week	30000D	30282R	Topics	Chapters
1	1/7	1/8	Intelligent Agents	[Ch 1.1-1.4 and 2.1-2.4]
	1/9	1/10	Problem Solving and Search	[Ch 3.1-3.3]
2	1/14	1/15	Uninformed Search	[Ch 3.3-3.4]
	1/16	1/17	Heuristic Search (A*)	[Ch 3.5]
3	1/21	1/22	Heuristic Functions	[Ch 3.6]
	1/23	1/24	Local Search	[Ch 4.1-4.2]
	1/25		Project 1 Out	
4	1/28	1/29	Adversarial Search	[Ch 5.1-5.3]
	1/30	1/31	Knowledge Based Agents	[Ch 7.1-7.3]
5	2/4	2/5	Propositional Logic Inference	[Ch 7.4-7.5]
	2/6	2/7	First-Order Logic	[Ch 8.1-8.4]
	2/8		Project 1 Due	
	2/8		Homework 1 Out	
6	2/11	2/12	Rule-Based Systems	[Ch 9.3-9.4]
	2/13	2/14	Search-Based Planning	[Ch 10.1-10.3]
	2/15		Homework 1 Due	
7	2/18	2/19	SAT-Based Planning	[Ch 10.4]
	2/20	2/21	Knowledge Representation	[Ch 12.1-12.5]
8	2/25	2/26	Midterm Review	
	2/27	2/28	**Midterm Exam**	

Outline

- What is AI?
- Problem-solving agent
  - Uninformed (DFS), informed (A*), and local search
  - Adversarial search (minimax, alpha-beta pruning)
- **Knowledge-based agent**
  - The Wumpus World
  - Propositional Logic
  - **First Order Logic**

Recap: *Logic for knowledge representation*

- **Logic** as a language for knowledge representation
  - Propositional logic (Boolean)
  - First-order logic (FOL)



- Advantage
  - Can combine and recombine information to suit many purposes

Recap: *Checking entailment*

- Two methods
  - Method#1: Based on enumeration (model checking)
  - Method#2: Based on inference rules (theorem proving)

- Enumerate all models and check if "α is true in all models in which KB is true"

$$\boxed{M(KB) \subseteq M(\alpha).}$$

Recap: *Checking entailment (example)*

$\boxed{KB \models \alpha_1}$

$\boxed{KB \not\models \alpha_2}$

α_1 = "There is no pit in [1,2]."

α_2 = "There is no pit in [2,2]."

$$\left.\begin{array}{lll} R_1: & \neg P_{1,1}\,. & \\ R_2: & B_{1,1} & \Leftrightarrow\ (P_{1,2} \lor P_{2,1})\,. \\ R_3: & B_{2,1} & \Leftrightarrow\ (P_{1,1} \lor P_{2,2} \lor P_{3,1})\,. \\ R_4: & \neg B_{1,1}\,. & \\ R_5: & B_{2,1}\,. & \end{array}\right\} \textbf{KB}$$

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	KB
false	*false*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*false*	*false*
false	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*false*	*true*	*false*	*false*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
false	*true*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*false*	*true*	*true*	*false*
false	*true*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*false*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*true*	*false*	*false*	*true*	*false*	*false*	*true*	*true*	*false*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	*true*	*true*	*true*	*true*	*true*	*true*	*false*	*true*	*true*	*false*	*true*	*false*

Recap: *Checking entailment*

- Two methods
  - Method#1: Based on enumeration (model checking)
  - Method#2: Based on inference rules (theorem proving)

- Logical equivalences

$$
\begin{aligned}
(\alpha \wedge \beta) &\equiv (\beta \wedge \alpha) && \text{commutativity of } \wedge \\
(\alpha \vee \beta) &\equiv (\beta \vee \alpha) && \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) &\equiv (\alpha \wedge (\beta \wedge \gamma)) && \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) &\equiv (\alpha \vee (\beta \vee \gamma)) && \text{associativity of } \vee \\
\neg(\neg\alpha) &\equiv \alpha && \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\beta \Rightarrow \neg\alpha) && \text{contraposition} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\alpha \vee \beta) && \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) && \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) &\equiv (\neg\alpha \vee \neg\beta) && \text{De Morgan} \\
\neg(\alpha \vee \beta) &\equiv (\neg\alpha \wedge \neg\beta) && \text{De Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) &\equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) && \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) &\equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) && \text{distributivity of } \vee \text{ over } \wedge
\end{aligned}
$$

Recap: *Applying inference rules*

- Example: $\boxed{KB \models \alpha_1}$

$$\left.\begin{array}{llll} R_1: & \neg P_{1,1}\,. & & \\ R_2: & B_{1,1} & \Leftrightarrow & (P_{1,2} \vee P_{2,1})\,. \\ R_3: & B_{2,1} & \Leftrightarrow & (P_{1,1} \vee P_{2,2} \vee P_{3,1})\,. \\ R_4: & \neg B_{1,1}\,. & & \\ R_5: & B_{2,1}\,. & & \end{array}\right\} \textit{KB}$$

α_1 = "There is no pit in [1,2]."

- $(R_1 \wedge \mathbf{R_2} \wedge R_3 \wedge \mathbf{R_4} \wedge R_5) \Rightarrow (\neg \mathbf{P_{1,2}})$

- *From R_2:* $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})\,.$
- *And-Elimination:* $((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})\,.$
- *Contra-positive:* $(\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1}))\,.$
- *From R_4:* $\neg B_{1,1}$
- *Modus Ponens:* $\neg(P_{1,2} \vee P_{2,1})\,.$
- *De Morgan:* $\neg P_{1,2} \wedge \neg P_{2,1}$

$\neg P_{1,2}$

Recap: *Problem related to completeness*

- Soundness:
  - obviously sound

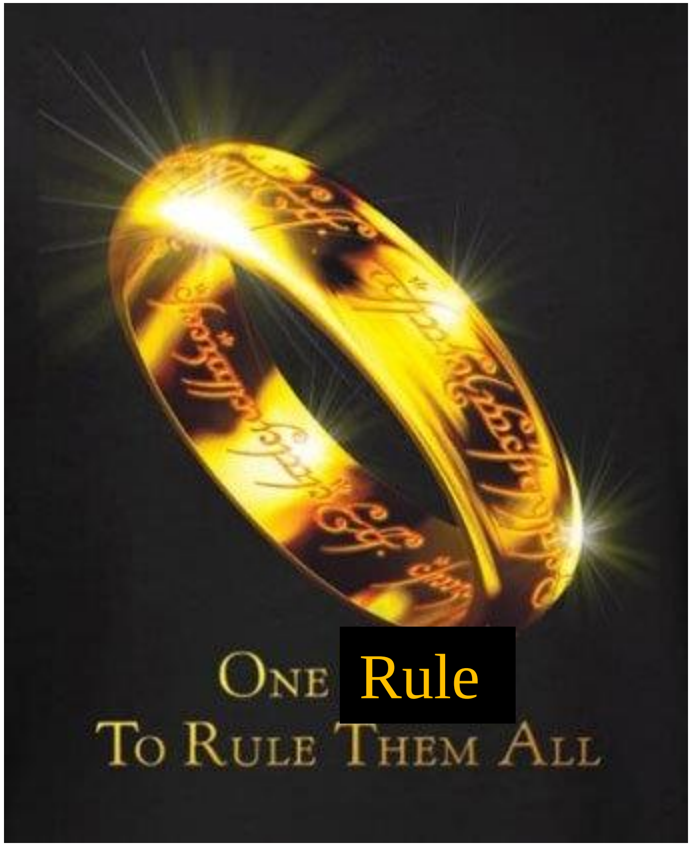
- Completeness:
  - if the inference rules are inadequate, proof may not be reachable

$$
\begin{array}{rcll}
(\alpha \wedge \beta) & \equiv & (\beta \wedge \alpha) & \text{commutativity of } \wedge \\
(\alpha \vee \beta) & \equiv & (\beta \vee \alpha) & \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) & \equiv & (\alpha \wedge (\beta \wedge \gamma)) & \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) & \equiv & (\alpha \vee (\beta \vee \gamma)) & \text{associativity of } \vee \\
\neg(\neg\alpha) & \equiv & \alpha & \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\beta \Rightarrow \neg\alpha) & \text{contraposition} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\alpha \vee \beta) & \text{implication elimination} \\
\cancel{(\alpha \Leftrightarrow \beta)} & \cancel{\equiv} & \cancel{((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))} & \cancel{\text{biconditional elimination}} \\
\neg(\alpha \wedge \beta) & \equiv & (\neg\alpha \vee \neg\beta) & \text{De Morgan} \\
\neg(\alpha \vee \beta) & \equiv & (\neg\alpha \wedge \neg\beta) & \text{De Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) & \equiv & ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) & \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) & \equiv & ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) & \text{distributivity of } \vee \text{ over } \wedge
\end{array}
$$

If we delete this rule, the proof on previous slide won't go through

Recap: *Resolution rule*

- Single rule that yields a **complete** inference algorithm



Recap: *Intuition behind resolution*

- Resolvent
- Pivot variable (*x*)

$$\boxed{(C \vee x) \wedge (D \vee \neg x) \rightarrow (C \vee D)}$$

- Resolution can be understood as "either… or…":
  - When (x = false), the formula equals (C)
  - When (x = true), the formula equals (D)
  - So, regardless, the formula is either (C) or (D)

Recap: *Intuition behind resolution*

- Resolvent
- Pivot variable (x)

$$\boxed{(C \vee x) \wedge (D \vee \neg x) \rightarrow (C \vee D)}$$

- Example: simple case (unit resolution = ***modus ponens***)

$\boxed{P_{1,1} \vee P_{2,2} \vee P_{3,1}}$ $\boxed{\neg P_{2,2}}$

$\boxed{P_{1,1} \vee P_{3,1}}$

$$\frac{\alpha \Rightarrow \beta, \quad \alpha}{\beta}$$

Recap: *Intuition behind resolution*

- Resolvent
- Pivot variable (x)

$$(C \vee x) \wedge (D \vee \neg x) \rightarrow (C \vee D)$$

- Example: general case (= *transitivity of implication*)

$$\frac{P_{1,1} \vee P_{3,1}, \qquad \neg P_{1,1} \vee \neg P_{2,2}}{P_{3,1} \vee \neg P_{2,2}}$$

$$\begin{aligned} & (\neg C \rightarrow x) \wedge (x \rightarrow D) \\ = & (\neg C \rightarrow D) \\ = & (C \vee D) \end{aligned}$$

Recap: *Limitations of Propositional Logic*

1. It can be too weak, i.e., with limited expressiveness:

- Each rule has to be represented for each situation:
  e.g., "don't go forward if the wumpus is in front of you" takes 64 rules

2. It cannot keep track of changes:

- If one needs to track changes of environment, e.g., where the agent has been before, we need a timed-version of each rule.
- To track 100 steps we'll then need 6400 rules for the previous example.

Its **hard to write and maintain** such a huge rule-base and **inference also becomes intractable**

Outline

- What is AI?
- Problem-solving agent
  - Uninformed (DFS, IDS, …), informed (A*, …), and local search (…)
  - Adversarial search (minimax, alpha-beta pruning)
- **Knowledge-based agent**
  - Propositional Logic
  - **First Order Logic**

Why first-order logic?

- Propositional logic is limited because it only makes the **ontological commitment** that a world consists of **facts**
  - Facts: propositions that are either *true* or *false*

Language	Ontological Commitment (What exists in the world)
Propositional logic	facts
First-order logic	facts, objects, relations
Temporal logic	facts, objects, relations, times
Probability theory	facts
Fuzzy logic	facts with degree of truth $\in [0, 1]$

- Difficult to represent even simple Wumpus world
  - "*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
  - **Functions**: ColorOf(car)
  - **Relations**: Inside(car, passenger), Beside(driver, passenger)
  - **Properties**: IsOpen(door), IsOn(engine)

- **Functions** return **objects**
- **Relations** return ***true*** or ***false*** *(they're predicates)*

First-order logic (FOL)

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

CarA

CarB

	CarA	CarB
Function:	ColorOf(CarA) = BLACK	ColorOf(CarB) = BLUE
Relation:	ColorOfCar(CarA,BLACK) = True	ColorOfCar(CarB,BLACK) = False
	ColorOfCar(CarA,BLUE) = False	ColorOfCar(CarB,BLUE) = True
Property:	IsBlackCar(CarA) = True	IsBlackCar(CarB) = False
	IsBlueCar(CarA) = False	IsBlueCar(CarB) = True

Examples:

- "One plus two equals three"

Objects:

Relations:

Properties:

Functions:

Examples:

- “One plus two equals three”

Objects: one, two, one plus two, three

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)

Examples:

- "Squares neighboring the Wumpus are smelly"

  Objects:

  Relations:

  Properties:

  Functions:

Examples:

- "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, Markus, …
- **Function symbols:** +, sqrt, SchoolOf, TeacherOf, ClassOf, …
- **Equality:** =
- **Predicate symbols:** >, Friend, Student, Colleague, …
- **Variables:** *x, y, z, next, first, …*
- **Connectives:** $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$
- **Quantifiers:** $\forall$, $\exists$

FOL: Syntax

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

FOL: Atomic sentences

$$\boxed{AtomicSentence \;\rightarrow\; Predicate \mid Predicate(Term, \ldots) \mid Term = Term}$$

$$\begin{array}{rcl} Term & \rightarrow & Function(Term, \ldots) \\ & \mid & Constant \\ & \mid & Variable \end{array}$$

- Examples:
  - SchoolOf (Bob)
  - Colleague (TeacherOf (Alice), TeacherOf (Bob))
  - >(+(x y), x)

FOL: Complex sentences

$$
\begin{array}{lcl}
ComplexSentence & \rightarrow & (\ Sentence\ ) \mid [\ Sentence\ ] \\
& \mid & \neg\ Sentence \\
& \mid & Sentence \land Sentence \\
& \mid & Sentence \lor Sentence \\
& \mid & Sentence \ \Rightarrow\ Sentence \\
& \mid & Sentence \ \Leftrightarrow\ Sentence
\end{array}
$$

- Examples:
  - $S1 \land S2$, $S1 \lor S2$, $(S1 \land S2) \lor S3$, $S1 \Rightarrow S2$, $S1 \Leftrightarrow S3$
  - Colleague (Paolo, Maja) $\Rightarrow$ Colleague (Maja, Paolo)

    Student (Alex, Paolo) $\Rightarrow$ Teacher (Paolo, 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 functions

- An atomic sentence *predicate*$(term_1, \ldots, term_n)$ is **true** iff the relation holds between objects $term_1, \ldots, term_n$

Example model

- **Objects**: John, James, Marry, Alex, Dan, Joe, Anne, Rich
- **Relations**: Parent(X,Y)

  Parent(John, James) is true
  Parent(John, Mary) is false

Example model

- **Objects**: John, James, Marry, Alex, Dan, Joe, Anne, Rich

- **Relations**: Parent(X,Y)

  Parent(John, James) is true
  Parent(John, Mary) is false

- A relation == a set of tuples of objects

  Parent := {<John, James>, <Mary, Alex>, <Mary, James>, …}

Quantifiers

- Representing **collections** of objects without enumeration (naming individuals)
  - E.g., All Trojans are clever
  - E.g., Someone in this class is sleeping
- Universal quantification (**for all**): $\forall$
- Existential quantification (**there exists**): $\exists$

Universal quantification (for all): ∀

∀ *<variables>* <***sentence***>

Example

- *"Every one in the cs360 class is smart"*:

$\forall\ x \quad \text{In}\ (\text{cs360}, x) \Rightarrow \text{Smart}\ (x)$

- **∀ P means the conjunction of all instantiations of P**

(In (cs360, Markus) ⇒ Smart (Markus)) ∧
(In (cs360, Dora) ⇒ Smart (Dora)) ∧
…
(In (cs360, Hao) ⇒ Smart (Hao))

Universal quantification (for all): $\forall$

- **Implication ($\Rightarrow$)** is a natural connective to use with **($\forall$)**
- Common mistake: to use $\wedge$ in conjunction with $\forall$

$$\forall x \quad In(cs360, x) \wedge Smart(x)$$

It means *"every one is in cs360 and everyone is smart"*

Existential quantification (there exists): ∃

∃ *<variables>* *<sentence>*

Example

- *"Someone in the cs360 class is smart"*:

  ∃ *x* In (cs360, *x*) ∧ Smart (*x*)

- **∃ P represents the disjunction of all instantiations of P**

  In (cs360, Markus) ∧ Smart (Markus) ∨
  In (cs360, Dora) ∧ Smart (Dora) ∨
  …
  In (cs360, Hao) ∧ Smart (Hao)

Existential quantification (there exists): $\exists$

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

- Common mistake: to use $\Rightarrow$ in conjunction with $\exists$

$$\exists x \quad In(cs360, x) \Rightarrow Smart(x)$$

This formula is actually is true if *there exists someone NOT in cs360*!

(Recall false $\Rightarrow \beta$ is always 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 $\forall y \;\exists x$

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

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

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"

Properties of quantifiers

Quantifier duality: each can be expressed using the other

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

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

Proof of duality

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

Proof of duality

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

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

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

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

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) $\Leftrightarrow$
  $\exists$ p, ps Parent(p, d) $\wedge$ Sibling(p, ps) $\wedge$ Child(c, ps)

Translating English to FOL

- Every gardener likes the sun.

  $\forall$ **x gardener(x) => likes(x,Sun)**

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

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

Translating English to FOL

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

  ∀ x person(x) => ∃ t time(t) ^ can-fool(x,t)

- All purple mushrooms are poisonous.

  ∀ x (mushroom(x) ^ purple(x)) => poisonous(x)

Caution with nested quantifiers

- $\forall x \exists y P(x,y)$ is the same as $\forall x (\exists y P(x,y))$

  "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 $\exists y (\forall x P(x,y))$

  "there exists y such that it is true that for every x P(x,y)"

Translating English to FOL…

- No purple mushroom is poisonous.

$$\neg(\exists\ x)\ purple(x) \wedge mushroom(x) \wedge poisonous(x)$$

or, equivalently,

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

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))
```

- Deb is not tall.

```
¬tall(Deb)
```

Translating English to FOL…

- X is above Y *if and only if*
  X is 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)))
```

Equality

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

Equality

$term_1 = term_2$ 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 $Parent$:

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

Higher-order logic?

- First-order logic allows us to quantify over **objects**.

- Higher-order logic allows us to quantify over **objects**, **relations,** and **functions**.

  e.g., “two objects are equal if and only if all **properties** applied to them are equivalent”:

  $$\forall\, x,y \quad (x=y) \Leftrightarrow (\forall\, \mathbf{p},\ \mathbf{p}(x) \Leftrightarrow \mathbf{p}(y))$$

- Higher-order logics are **more expressive**; however, it is not clear (yet) *how to effectively reason about* sentences in higher-order logic.

Knowledge-based agent

```
function KB-AGENT(percept) returns an action
    static: KB, a knowledge base
            t, a counter, initially 0, indicating time

    TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
    action ← ASK(KB, MAKE-ACTION-QUERY(t))
    TELL(KB, MAKE-ACTION-SENTENCE(action, t))
    t ← t + 1
    return action
```

1. TELL KB what was perceived
   Uses a KRL to insert new sentences, representations of facts, into KB

2. ASK KB what to do.
   Uses logical reasoning to examine actions and select best.

Using FOL Knowledge Base *(King John)*

$\text{TELL}(KB,\ King(John))$.
$\text{TELL}(KB,\ Person(Richard))$.
$\text{TELL}(KB,\ \forall x\ \ King(x)\ \Rightarrow\ Person(x))$.

$\text{ASK}(KB,\ King(John))$

$\text{ASK}(KB,\ Person(John))$

$\text{ASK}(KB,\ \exists x\ \ Person(x))$

$\text{ASKVARS}(KB, Person(x))$

$\{x/John\}$ and $\{x/Richard\}$.

Using FOL Knowledge Base *(kinship)*

Parent and child are inverse relations:

A grandparent is a parent of one's parent:

A sibling is another child of one's parents:

Using FOL Knowledge Base *(kinship)*

These are called "**axioms**" since they are facts given in the KB

Parent and child are inverse relations:

$$\forall\, p, c \;\; Parent(p, c) \;\Leftrightarrow\; Child(c, p) \,.$$

A grandparent is a parent of one's parent:

$$\forall\, g, c \;\; Grandparent(g, c) \;\Leftrightarrow\; \exists\, p \;\; Parent(g, p) \wedge Parent(p, c) \,.$$

A sibling is another child of one's parents:

$$\forall\, x, y \;\; Sibling(x, y) \;\Leftrightarrow\; x \neq y \wedge \exists\, p \;\; Parent(p, x) \wedge Parent(p, y) \,.$$

$$\boxed{\forall\, x, y \;\; Sibling(x, y) \;\Leftrightarrow\; Sibling(y, x)}$$

This is called a "**theorem**" since it can be inferred from the KB

Using FOL Knowledge Base *(Wumpus)*

$$Percept([Stench, Breeze, Glitter, None, None], 5)\ .$$

$$\forall\, t, s, g, m, c \;\; Percept([s, Breeze, g, m, c], t) \;\Rightarrow\; Breeze(t)\ ,$$
$$\forall\, t, s, b, m, c \;\; Percept([s, b, Glitter, m, c], t) \;\Rightarrow\; Glitter(t)\ ,$$

$$\forall\, t \;\; Glitter(t) \;\Rightarrow\; BestAction(Grab, t)\ .$$

$$\textsc{AskVars}(\exists\, a \;\; BestAction(a, 5))$$

$$\boxed{BestAction(Grab, 5)}$$

Using FOL Knowledge Base *(Wumpus)*

Suppose a wumpus-world agent is using an FOL KB
and perceives a smell and a breeze (but no glitter) at $t = 5$:

$\textsc{Tell}(KB, Percept([Smell, Breeze, None], 5))$
$\textsc{Ask}(KB, \exists\, a \;\; Action(a, 5))$

I.e., does the KB entail any particular actions at $t = 5$?

Using FOL Knowledge Base *(Wumpus)*

Suppose a wumpus-world agent is using an FOL KB
and perceives a smell and a breeze (but no glitter) at $t = 5$:

$\textsc{Tell}(KB, Percept([Smell, Breeze, None], 5))$
$\textsc{Ask}(KB, \exists\, a \;\; Action(a, 5))$

I.e., does the KB entail any particular actions at $t = 5$?

Answer: Yes, $\{a/Shoot\}$ $\leftarrow$ <u>substitution</u> (binding list)

Using FOL Knowledge Base *(Wumpus)*

$$\forall\, x, y, a, b \;\; Adjacent([x, y], [a, b]) \;\Leftrightarrow$$
$$(x = a \wedge (y = b - 1 \vee y = b + 1)) \vee (y = b \wedge (x = a - 1 \vee x = a + 1))$$

Squares are breezy near a pit:

<u>Diagnostic</u> rule—infer cause from effect

$$\forall y \;\; Breezy(y) \;\Rightarrow\; \exists x \;\; Pit(x) \wedge Adjacent(x, y)$$

Using FOL Knowledge Base *(Wumpus)*

$$\forall\, x, y, a, b \;\; Adjacent([x,y],[a,b]) \;\Leftrightarrow$$
$$(x = a \wedge (y = b - 1 \vee y = b + 1)) \vee (y = b \wedge (x = a - 1 \vee x = a + 1))$$

Squares are breezy near a pit:

Diagnostic rule—infer cause from effect

$$\forall\, y \;\; Breezy(y) \;\Rightarrow\; \exists\, x \;\; Pit(x) \wedge Adjacent(x,y)$$

Causal rule—infer effect from cause

$$\forall\, x, y \;\; Pit(x) \wedge Adjacent(x,y) \;\Rightarrow\; Breezy(y)$$

Using FOL Knowledge Base *(Wumpus)*

$$\forall\, x, y, a, b \;\; Adjacent([x, y], [a, b]) \;\Leftrightarrow$$
$$(x = a \wedge (y = b - 1 \vee y = b + 1)) \vee (y = b \wedge (x = a - 1 \vee x = a + 1))$$

Squares are breezy near a pit:

Diagnostic rule—infer cause from effect

$$\forall\, y \;\; Breezy(y) \;\Rightarrow\; \exists\, x \;\; Pit(x) \wedge Adjacent(x, y)$$

Causal rule—infer effect from cause

$$\forall\, x, y \;\; Pit(x) \wedge Adjacent(x, y) \;\Rightarrow\; Breezy(y)$$

Neither of these is complete—e.g., the causal rule doesn't say whether squares far away from pits can be breezy

Using FOL Knowledge Base *(Wumpus)*

$$\forall\, x, y, a, b \;\; Adjacent([x, y], [a, b]) \;\Leftrightarrow$$
$$(x = a \wedge (y = b - 1 \vee y = b + 1)) \vee (y = b \wedge (x = a - 1 \vee x = a + 1))$$

Squares are breezy near a pit:

Diagnostic rule—infer cause from effect

$$\forall y \;\; Breezy(y) \;\Rightarrow\; \exists x \;\; Pit(x) \wedge Adjacent(x, y)$$

Causal rule—infer effect from cause

$$\forall x, y \;\; Pit(x) \wedge Adjacent(x, y) \;\Rightarrow\; Breezy(y)$$

Neither of these is complete—e.g., the causal rule doesn't say whether squares far away from pits can be breezy

Definition for the $Breezy$ predicate:

$$\forall y \;\; Breezy(y) \;\Leftrightarrow\; [\exists x \;\; Pit(x) \wedge Adjacent(x, y)]$$

Outline

- What is AI?
- Problem-solving agent
  - Uninformed (DFS), informed (A*), and local search
  - Adversarial search (minimax, alpha-beta pruning)
- **Knowledge-based agent**
  - The Wumpus World
  - Propositional Logic
  - **First Order Logic**