Lec6: Knowledge and logic reasoning 2: First-Order Logic

Outline

- What is First-Order Logic (FOL)?
  - Syntax and semantics
- Using FOL
- Wumpus world in FOL
- Knowledge engineering in FOL

Limitations of propositional logic

- ☹ Propositional logic has limited expressive power
  - unlike natural language
  - E.g., cannot say "pits cause breezes in adjacent squares“
    - except by writing one sentence for each square

Wumpus World and propositional logic

- Find Pits in Wumpus world
  - $B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$ (Breeze next to Pit) 16 rules
- Find Wumpus
  - $S_{x,y} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee W_{x-1,y})$ (stench next to Wumpus) 16 rules
- At least one Wumpus in world
  - $W_{1,1} \vee W_{1,2} \vee \ldots \vee W_{4,4}$ (at least 1 Wumpus) 1 rule
- At most one Wumpus
  - $\neg W_{1,1} \vee \neg W_{1,2}$ (155 RULES)

First-Order Logic

- Propositional logic assumes that the world contains facts.

- First-order logic (like natural language) assumes the world contains

  - Objects: people, houses, numbers, colors, baseball games, wars, …

  - Relations: red, round, prime, brother of, bigger than, part of, comes between, …

Logics in General

- Ontological Commitment:
  - What exists in the world — TRUTH
  - PL : facts hold or do not hold.
  - FOL : objects with relations between them that hold or do not hold
- Epistemological Commitment:
  - What an agent believes about facts — BELIEF

Language	Ontological Commitment	Epistemological Commitment
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	degree of truth $\in [0,1]$	known interval value

Syntax of FOL: Basic elements

- Constant Symbols:
  - Stand for objects
  - e.g., KingJohn, 2, SA,...

- Predicate Symbols
  - Stand for relations
  - E.g., Brother(Richard, John), greater_than(3,2)...
  - Return true or false

- Function Symbols
  - Stand for functions
  - E.g., Sqrt(3), LeftLegOf(John),Father(John),...
  - Return what?

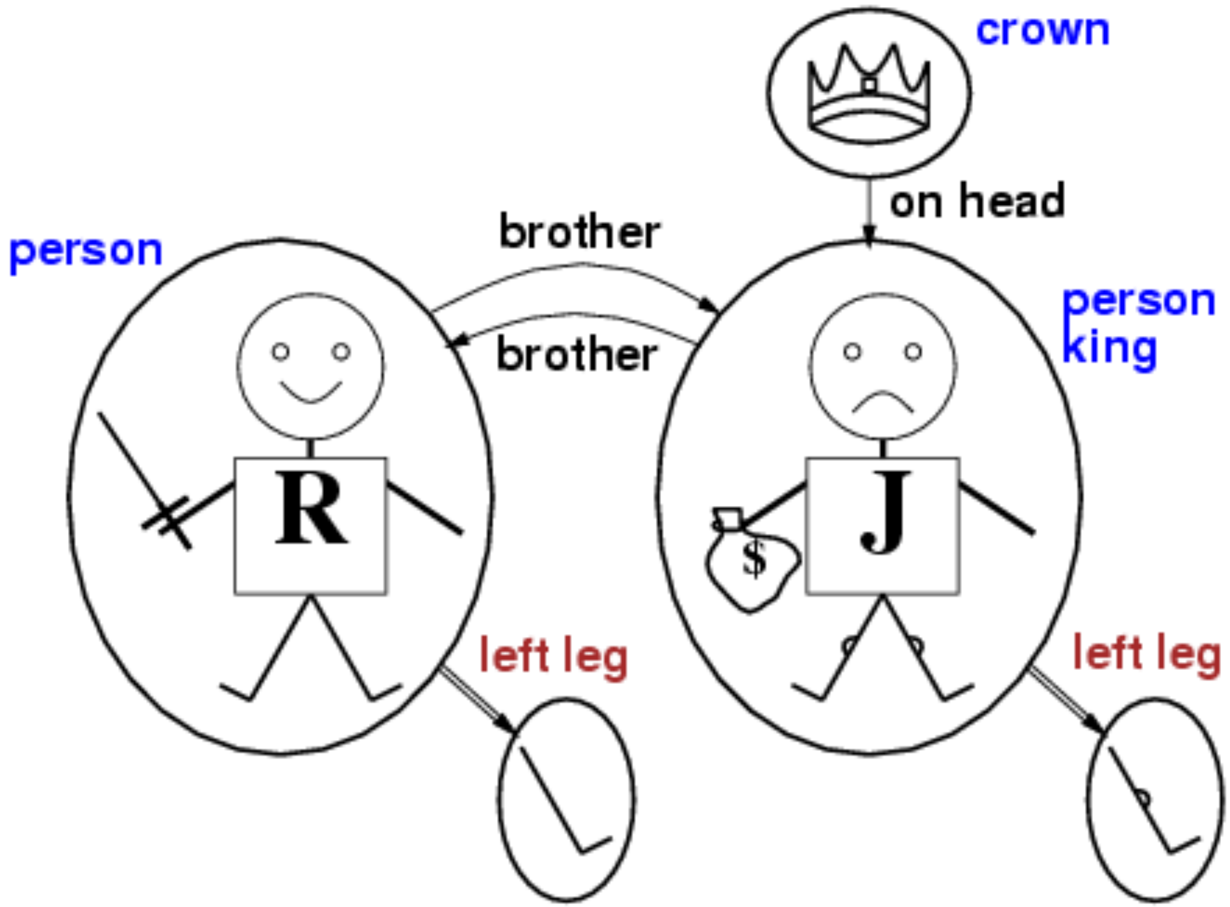
Syntax of FOL: Basic elements

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

Relations

- Some relations are properties: they state some fact about a single object: Round(ball), Prime(7).
- n-ary relations state facts about two or more objects: Married(John,Mary), LargerThan(3,2).
- Some relations are functions: their value is another object: Plus(2,3), Father(Dan).

Models for FOL: Graphical Example



Tabular Representation

- A FOL model is related to relational database.
- Historically, the relational data model comes from FOL.

Student

s-id	**Intelligence**	**Ranking**
Jack	3	1
Kim	2	1
Paul	1	2

Course

c-id	Rating	Difficulty
101	3	1
102	2	2

Professor

p-id	Popularity	Teaching-a
Oliver	3	1
Jim	2	1

RA

s-id	**p-id**	**Salary**	**Capability**
Jack	Oliver	High	3
Kim	Oliver	Low	1
Paul	Jim	Med	2

Registration

s-id	**c.id**	**Grade**	**Satisfaction**
Jack	101	A	1
Jack	102	B	2
Kim	102	A	1
Paul	101	B	1

Terms

- Term = logical expression that refers to an object.

- There are 2 kinds of terms:
  - constant symbols: Table, Computer
  - function symbols: LeftLeg(Pete), Sqrt(3), Plus(2,3) etc
- Functions can be nested:
  - Pat_Grandfather(x) = father(father(x))
- Terms can contain variables.
- No variables = **ground term**.

Atomic Sentences

- Atomic sentences state facts using terms and predicate symbols
  - P(x,y) interpreted as "x is P of y"

- Examples:

  LargerThan(2,3) is false.

  Brother_of(Mary,Pete) is false.

  Married(Father(Richard), Mother(John)) could be true or false

- Note: Functions do not state facts and form no sentence:
  - Brother(Pete) refers to John (his brother) and is neither true nor false.

- Brother_of(Pete,Brother(Pete)) is True.

Binary relation Function

Complex Sentences

- We make complex sentences with connectives (just like in propositional logic).

property

$$\neg \mathit{Brother}(\mathit{LeftLeg}(\mathit{Richard}), \mathit{John}) \vee (\mathit{Democrat}(\mathit{Bush}))$$

binary relation

function

objects

connectives

More Examples

- $Brother(Richard, John) \wedge Brother(John, Richard)$
- $King(Richard) \vee King(John)$
- $King(John) \Rightarrow \neg King(Richard)$
- $LessThan(Plus(1,2), 4) \wedge GreaterThan(1,2)$

(Semantics are the same as in propositional logic)

Variables

- Person(John) is true or false because we give it a single argument 'John'

- We can be much more flexible if we allow variables which can take on values in a domain. e.g., all persons x, all integers i, etc.
  - E.g., can state rules like Person(x) => HasHead(x)

    or Integer(i) => Integer(plus(i,1))

Universal Quantification ∀

- ∀ means "for all"
- Allows us to make statements about all objects that have certain properties
- Can now state general rules:

$\forall$ x King(x) => Person(x)

$\forall$ x Person(x) => HasHead(x)

$\forall$ i Integer(i) => Integer(plus(i,1))

Note that
$\forall$ x King(x) $\wedge$ Person(x) is not correct!
This would imply that all objects x are Kings and are People

$\forall$ x King(x) => Person(x) is the correct way to say

Existential Quantification ∃

- ∃ x means "there exists an x such that…." (at least one object x)
- Allows us to make statements about some object without naming it
- Examples:

  $\exists x$ King(x)

  $\exists x$ Lives_in(John, Castle(x))

  $\exists i$ Integer(i) $\wedge$ GreaterThan(i,0)

  Note that $\wedge$ is the natural connective to use with ∃

  (And => is the natural connective to use with ∀)

More examples

For all real x, x>2 implies x>3.

$$\forall x[(x > 2) \Rightarrow (x > 3)] \quad x \in R \quad (false)$$

$$\exists x[(x^2 = -1)] \quad x \in R \quad (false)$$

There exists some real x whose square is minus 1.

Fun with sentences

Brothers are siblings

Fun with sentences

Brothers are siblings

$\forall\, x, y \;\; Brother(x, y) \;\Rightarrow\; Sibling(x, y).$

"Sibling" is symmetric

Fun with sentences

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

Fun with sentences

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

Fun with sentences

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

$\forall\, x, y \;\; Mother(x, y) \;\Leftrightarrow\; (Female(x) \land Parent(x, y)).$

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

$\forall\, x, y \;\; FirstCousin(x, y) \;\Leftrightarrow\; \exists\, p, ps \;\; Parent(p, x) \land Sibling(ps, p) \land Parent(ps, y)$

Combining Quantifiers

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

- For everyone ("all x") there is someone ("y") that they love.

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

- there is someone ("y") who is loved by everyone

Clearer with parentheses: $\exists y (\forall x \quad Loves(x,y))$

Duality: Connections between Quantifiers

- Asserting that all x have property P is the same as asserting that
  there does not exist any x that does't have the property P

$\forall$ x Likes(x, 271 class) $\Leftrightarrow$ $\neg$ $\exists$ x $\neg$ Likes(x, 271 class)

In effect:
- $\forall$ is a conjunction over the universe of objects
- $\exists$ is a disjunction over the universe of objects

Thus, DeMorgan's rules can be applied

De Morgan's Law for Quantifiers

De Morgan's Rule

$$P \wedge Q \equiv \neg(\neg P \vee \neg Q)$$

$$P \vee Q \equiv \neg(\neg P \wedge \neg Q)$$

$$\neg(P \wedge Q) \equiv \neg P \vee \neg Q$$

$$\neg(P \vee Q) \equiv \neg P \wedge \neg Q$$

Generalized De Morgan's Rule

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

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

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

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

Rule is simple: if you bring a negation inside a disjunction or a conjunction, always switch between them (or →and, and → or).

Exercise

- Formalize the sentence
  “Jack has reserved all red boats.”
- Apply De Morgan’s duality laws to this sentence.

Using FOL

- We want to TELL things to the KB, e.g.
  TELL(KB, $\forall x, King(x) \Rightarrow Person(x)$)
  TELL(KB, *King(John)*)

  These sentences are assertions

- We also want to ASK things to the KB,
  ASK(KB, $\exists x, Person(x)$)

  these are queries or goals

  The KB should output x where Person(x) is true: {x/John,x/Richard,...}

Deducing hidden properties

Environment definition:

$\forall x,y,a,b\ \mathit{Adjacent}([x,y],[a,b]) \Leftrightarrow [a,b] \in \{[x+1,y], [x\text{-},y],[x,y+1],[x,y-1]\}$

Properties of locations:

$\forall s,t\ \ \mathit{At}(\text{Agent},s,t) \wedge \text{Breeze}(t) \Rightarrow \text{Breezy}(s)$

Location *s* and time *t*

Squares are breezy near a pit:

- Diagnostic rule---infer cause from effect

  $\forall s\ \text{Breezy}(s) \Leftrightarrow \exists\ r\ \text{Adjacent}(r,s) \wedge \text{Pit}(r)$

- Causal rule---infer effect from cause.

  $\forall r\ \text{Pit}(r) \Rightarrow [\forall s\ \text{Adjacent}(r,s) \Rightarrow \text{Breezy}(s)]$

Knowledge engineering in FOL

1. Identify the task
2. Assemble the relevant knowledge
3. Decide on a vocabulary of predicates, functions, and constants
4. Encode general knowledge about the domain
5. Encode a description of the specific problem instance
6. Pose queries to the inference procedure and get answers
7. Debug the knowledge base.
8. See text for full example: electric circuit knowledge base.

Summary

- First-order logic:
  - Much more expressive than propositional logic
  - Allows objects and relations as semantic primitives
  - Universal and existential quantifiers
  - syntax: constants, functions, predicates, equality, quantifiers
- Knowledge engineering using FOL
  - Capturing domain knowledge in logical form
- Inference and reasoning in FOL
  - Next lecture.
- FOL is more expressive.

32