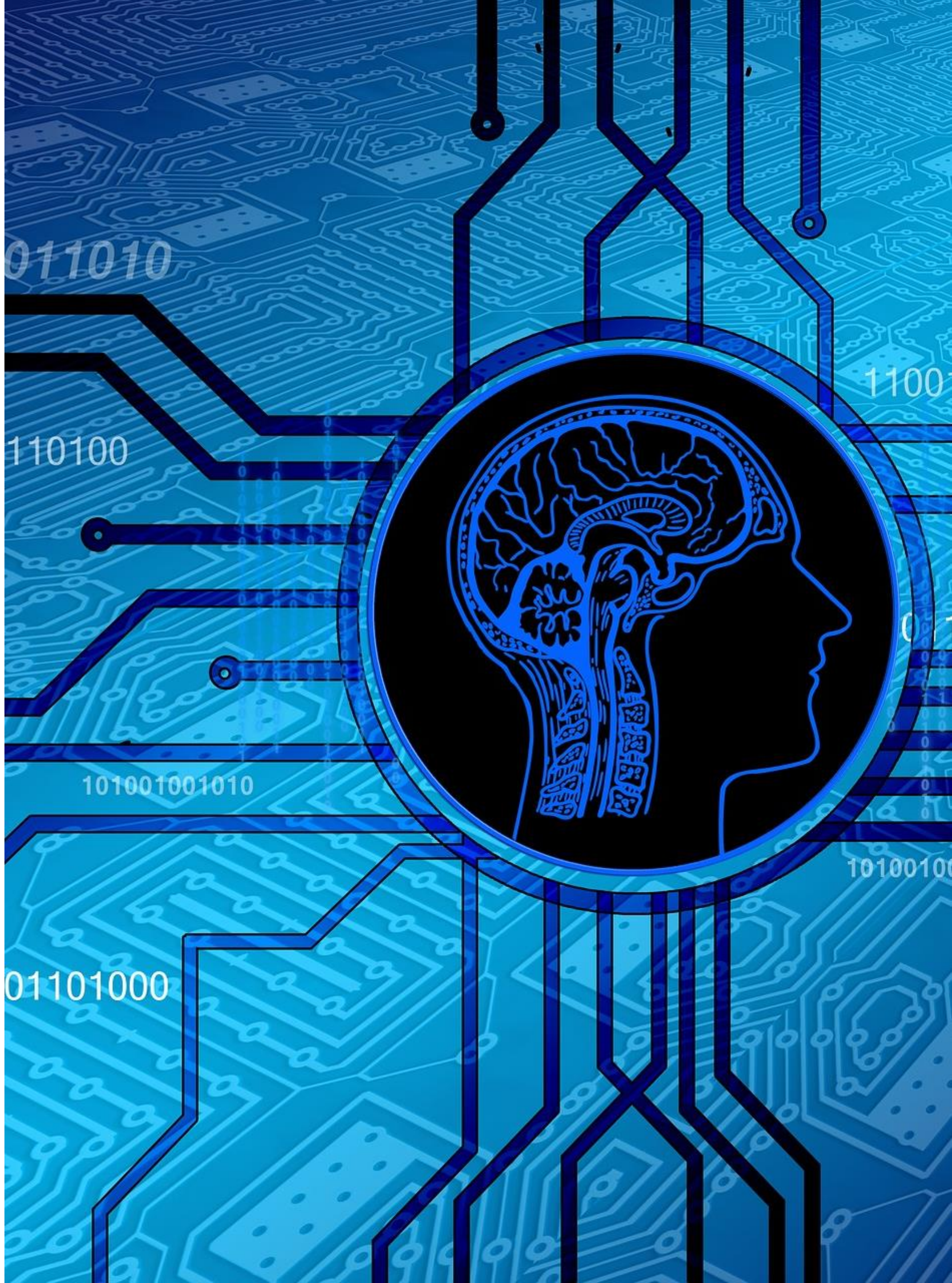
First-order Logic

Informatics 2D: Reasoning and Agents

Adapted from slides provided by Dr Petros Papapanagiotou



Pros and cons of Propositional Logic

✓ Declarative

✓ Partial/disjunctive/negated information

- (unlike most data structures and databases!)

✓ Compositional

The meaning of $B_{1,1} \wedge P_{1,2}$ is derived from that of $B_{1,1}$ and of $P_{1,2}$

⊗ Meaning is context-independent

- (unlike natural language, where meaning depends on context)

⊗ Very limited expressive power

- (unlike natural language)
- for example, we cannot say "*pits cause breezes in adjacent squares*", except by writing one sentence for each square

First-order logic (FOL)

- Propositional logic assumes the world contains atomic facts.
  - Non-structured propositional symbols, usually finitely many.
- FOL assumes the world contains:

Objects	• people, houses, numbers, colours, football games, wars, …
Relations	• red, round, prime, brother of, bigger than, part of, comes between, …
Functions	• father of, best friend, one more than, plus, …

Syntax of FOL: Basic elements

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

Syntax of FOL: Basic elements

Arity!

Constants	• *KingJohn*/0, 2 /0, *UoE* /0, …
Predicates	• *Brother*/2, >/2, …
Functions	• *Sqrt*/1, *LeftLegOf*/1, +/2, …
Variables	• x, y, a, b, …
Connectives	• $\neg$, $\Rightarrow$, $\wedge$, $\vee$, $\Leftrightarrow$
Equality	• $=$
Quantifiers	• $\forall$, $\exists$

Atomic formulae

Atomic formula = *predicate* (*term*$_1$,…,*term*$_n$)
or *term*$_1$ = *term*$_2$

Term = *function* (*term*$_1$,…,*term*$_n$)
or *constant* or *variable*

Examples:

- *Brother*(*KingJohn*,*Richard*)
- >(`Length`(*LeftLegOf*(*Richard*)), `Length`(*LeftLegOf*(*KingJohn*)))

predicate functions constants

Complex formulae

Complex formulae are made from atomic formulae using connectives

$\neg P \quad P \wedge Q \quad P \vee Q \quad P \Rightarrow Q \quad P \Leftrightarrow Q$

Examples:

$Sibling(KingJohn, Richard) \Rightarrow Sibling(Richard, KingJohn)$

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

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

Semantics of first-order logic

Formulae are mapped to an **interpretation.**

An interpretation is called a model of a set of formulae when all the formulae are **true** in the interpretation.

Semantics of first-order logic

- An interpretation contains objects (domain elements) and relations between them. Mapping is as follows :

  constant symbols $\mapsto$ objects

  predicate symbols $\mapsto$ relation

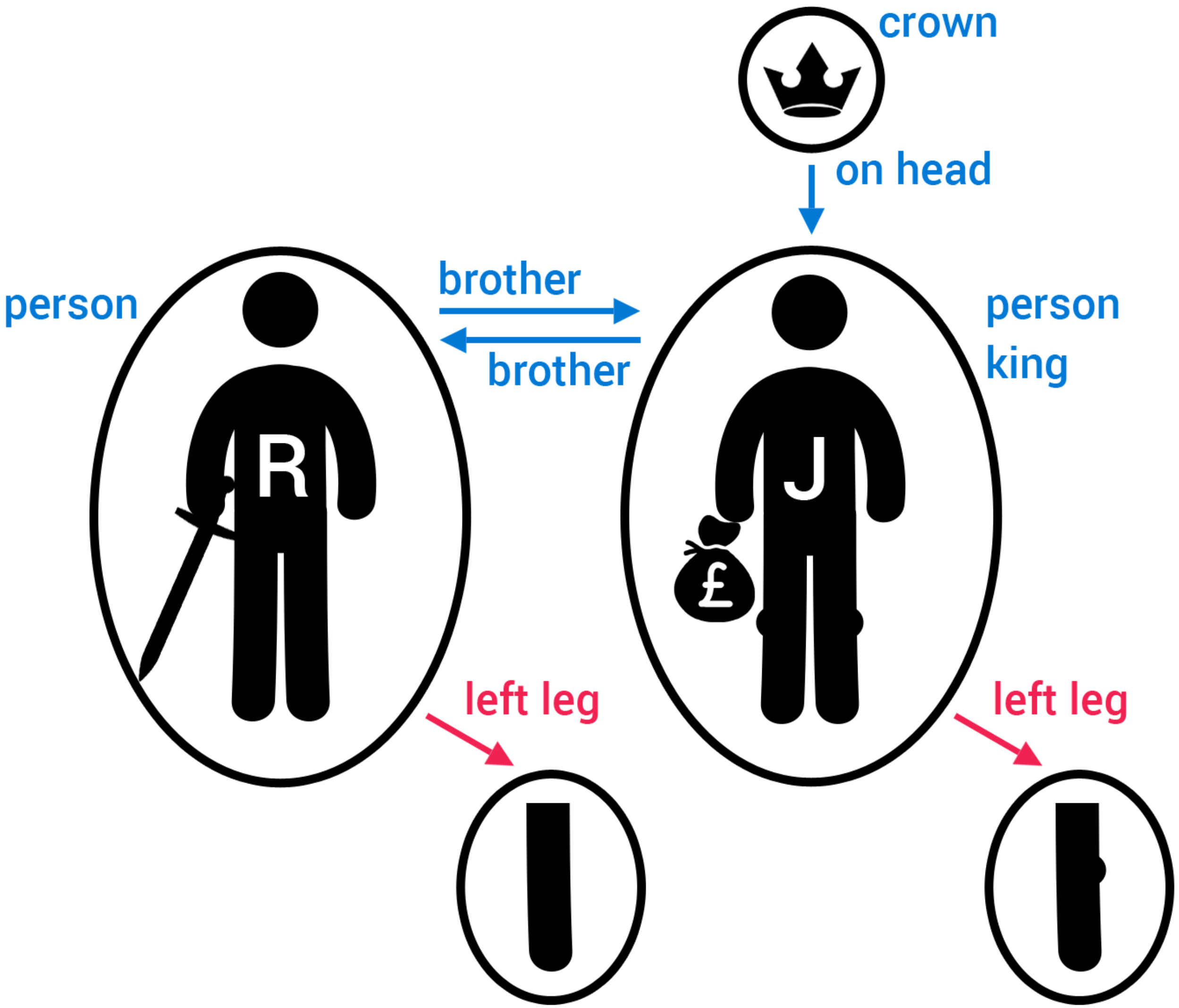
  function symbols $\mapsto$ functions

- An atomic formula *predicate*(*term*$_1$,...,*term*$_n$) is **true**

  iff the objects referred to by *term*$_1$,...,*term*$_n$

  are in the relation referred to by *predicate*.

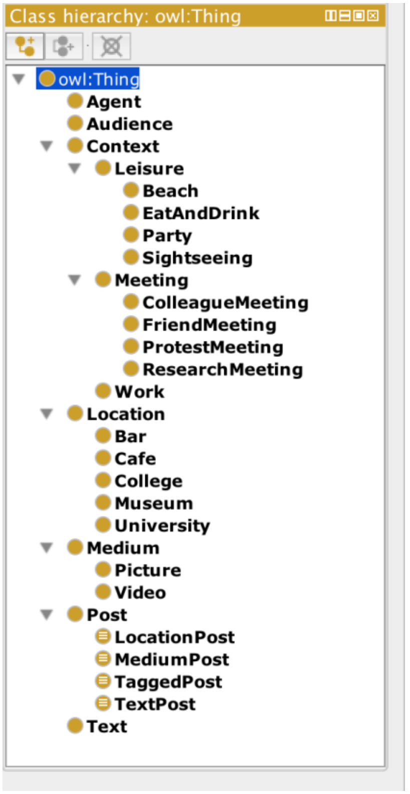
Interpretations for FOL: Example

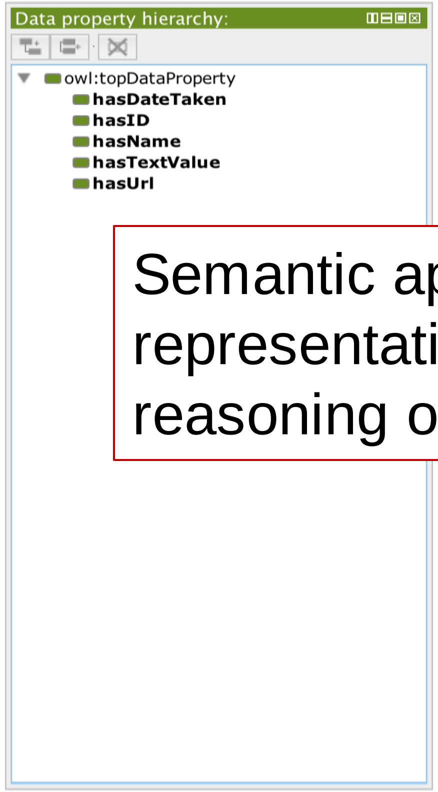


Brother(KingJohn,Richard)

>(`Length`(LeftLegOf(Richard)), `Length`(LeftLegOf(KingJohn)))

A Social Network Ontology based on Description Logics





Semantic approaches rely on a knowledge representation, such as an ontology, for reasoning on the content.

PriGuard: A Semantic Approach to Detect Privacy Violations in Online Social Networks. Nadin Kökciyan and PInar Yolum. IEEE Transactions on Knowledge and Data Engineering (TKDE), vol. 28, no. 10, pp. 2724-2737. 2016.

Adding further inference: Rules

I_1: isFriendOf(*?user*, *?tp*) → inFRContext(*?user*, *?tp*)
I_2: workAt(*?user*, *?office*), workAt(*?tp*, *?office*), hasRole(*?tp*, *?role*), isColleagueOf(*?user*, *?tp*) → inPRContext(*?user*, *?tp*)
I_3: Emergency(*?em*), isInEmergency(*?user*, *?em*), hasRole(*?tp*, *:em-responder*) → inEMContext(*?user*, *?tp*)
P_1: owns(*:bob*, *:bob-loc*), inPRContext(*:bob*, *?tp*) → allow(*?tp*, *:bob-loc*)
P_2: owns(*:bob*, *:bob-loc*), inFRContext(*:bob*, *?tp*) → disallow(*?tp*, *:bob-loc*)
A_1: owns(*:bob*, *:bob-loc*), inPRContext(*:bob*, *:alice*), hasAR(*:alice*, *:bob-loc*) → allow(*:alice*, *:bob-loc*)
A_2: owns(*:bob*, *:bob-mobile*), inEMContext(*:bob*, *:alice*), hasAR(*:alice*, *:bob-mobile*) → allow(*:alice*, *:bob-mobile*)
A_3: owns(*:bob*, *:bob-loc*), inFRContext(*:bob*, *:alice*), hasAR(*:alice*, *:bob-loc*) → disallow(*:alice*, *:bob-loc*)

Contextual Integrity for Argumentation-based Privacy Reasoning. Gideon Ogunniye and Nadin Kökciyan.
The International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2023, Accepted.

Universal quantification

∀*<variables>*. *<formula>*

- But will often write $\forall\, x,y.\ P$ for $\forall x.\ \forall y.\ P$
- Example: *Everyone at UoE is smart*: $\forall x.\ At(x, UoE) \Rightarrow Smart(x)$

➢ $\forall x.\ P$ is true in an interpretation *m* iff *P* is true with *x* being **each** possible object in the interpretation.

➢ Roughly speaking, equivalent to the conjunction of instantiations of *P*

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

Existential quantification

∃<variables>. <formula>

- But will often write ∃ *x*,*y*. *P* for ∃ x. ∃ *y*. *P*
- Example: *Someone at UoE is smart*: ∃x. At(x, UoE) ∧ Smart(x)

➢ ∃*x*. *P* is true in an interpretation *m* iff *P* is true with *x* being **some** possible object in the interpretation.

➢ Roughly speaking, equivalent to the disjunction of instantiations of *P*

At(KingJohn, UoE) ∧ Smart(KingJohn)

∨ At(Richard, UoE) ∧ Smart(Richard)

∨ At(UoE, UoE) ∧ Smart(UoE) ∨ ...

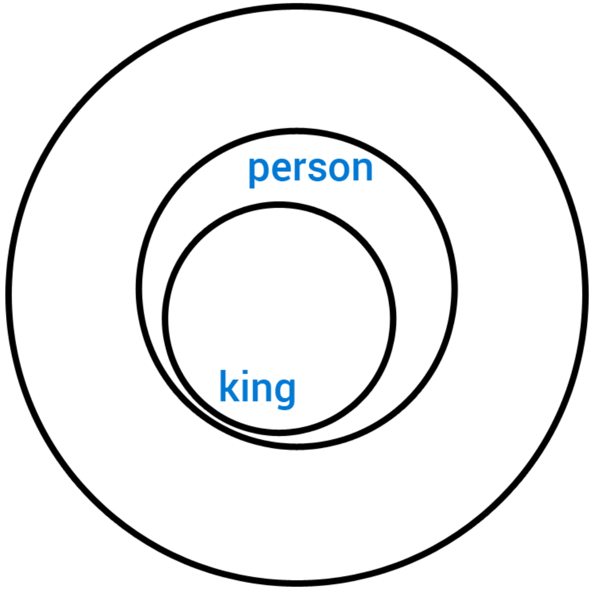
Rule of thumb

$\forall \quad \Rightarrow$

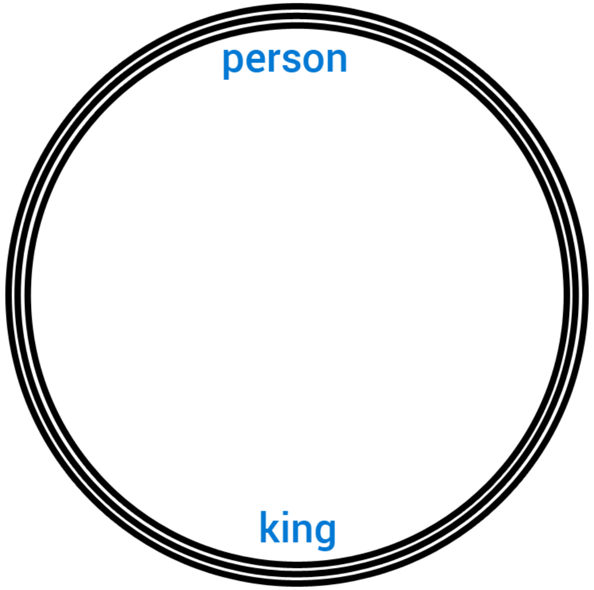
$\exists \quad \wedge$

Common mistakes

$\forall x.\ \text{King}(x) \Rightarrow \text{Person}(x)$

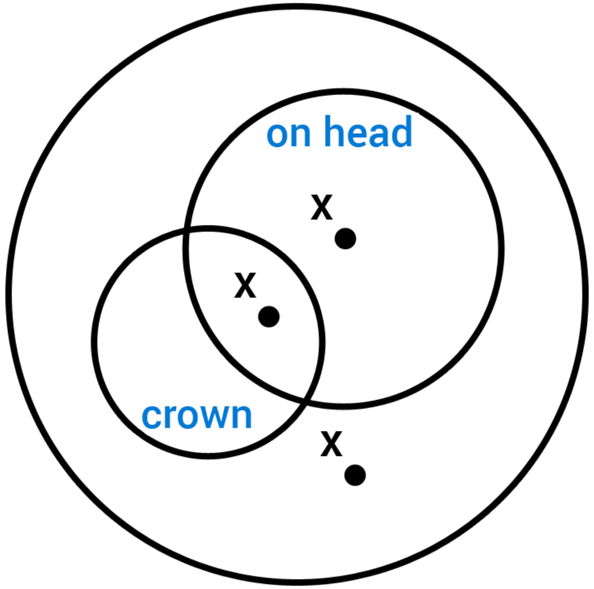


$\forall x.\ \text{King}(x) \wedge \text{Person}(x)$

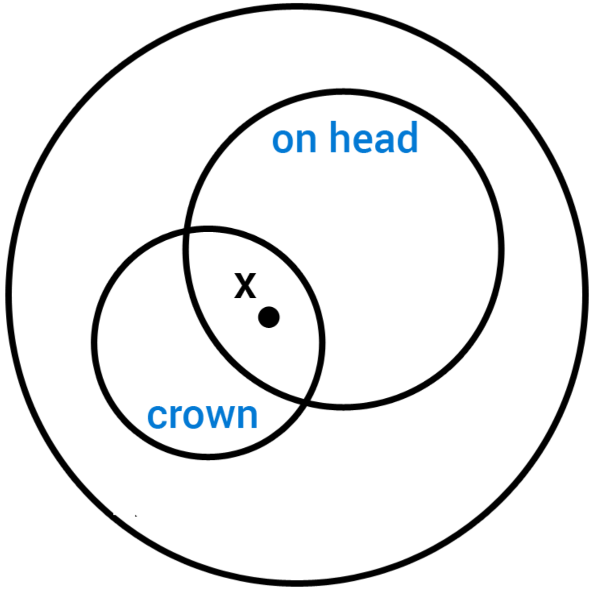


Common mistakes

$\exists x.\ \mathrm{Crown}(x) \Rightarrow \mathrm{OnHead}(x, John)$



$\exists x.\ \mathrm{Crown}(x) \wedge \mathrm{OnHead}(x, John)$



Properties of quantifiers

- $\forall x.\forall y.$ is the same as $\forall y.\forall x.$ $\exists x.\exists y.$ is the same as $\exists y.\exists x.$
- $\exists x.\forall y.$ 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) $\equiv$ $\neg\exists x.\ \neg$Likes(x, IceCream)
  - $\exists x.$ Likes(x, Broccoli) $\equiv$ $\neg\forall x.\ \neg$Likes(x, Broccoli)

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.

- Example: Definition of *Sibling* in terms of *Parent*:

$$\forall x, y.\ Sibling(x, y) \Leftrightarrow (\neg(x = y) \wedge$$
$$\exists m, f.\ \neg(m = f) \wedge$$
$$\mathrm{Parent}(m, x) \wedge \mathrm{Parent}(f, x) \wedge \mathrm{Parent}(m, y) \wedge \mathrm{Parent}(f, y))$$

Example: Kinship domain

Brothers are siblings.

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

One's mother is one's female parent.

- $\forall m, c.\ Mother(c) = \mathrm{m} \Leftrightarrow (Female(m) \wedge Parent(m, c))$

"Sibling" is symmetric.

- $\forall x, y.\ Sibling(x, y) \Leftrightarrow Sibling(y, x)$

"Parent" and "Child" are inverse relations.

- $\forall x, y.\ Parent(x, y) \Leftrightarrow Child(y, x)$

Example: Set domain

$\forall s.\ Set(s) \Leftrightarrow (s = \{\}) \vee (\exists x, s_2.\ Set(s_2) \wedge s = \{x|s_2\})$

$\neg\exists x, s.\ \{x|s\} = \{\}$

$\forall x, s.\ x \in s \Leftrightarrow s = \{x|s\}$

$\forall x, s.\ x \in s \Leftrightarrow [\ \exists y, s_{2.}\ (s = \{y|s_2\} \wedge (x = y \vee x \in s_2))]$

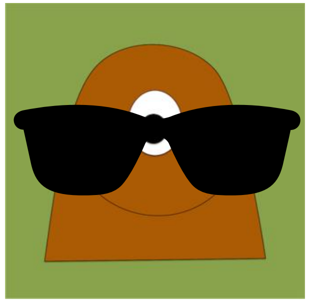
$\forall s_1, s_{2.}\ s_1 \subseteq s_2 \Leftrightarrow (\forall x.\ x \in s_1 \Rightarrow x \in s_2)$

$\forall s_1, s_{2.}\ (s_1 = s_2) \Leftrightarrow (s_1 \subseteq s_2 \wedge s_2 \subseteq s_1)$

$\forall x, s_1, s_{2.}\ x \in (s_1 \cap s_2) \Leftrightarrow (x \in s_1 \wedge x \in s_2)$

$\forall x, s_1, s_{2.}\ x \in (s_1 \cup s_2) \Leftrightarrow (x \in s_1 \vee x \in s_2)$

Interacting with FOL KBs



- Suppose a Wumpus-world agent using a FOL KB perceives: a smell and a breeze (but no glitter) at *t=5*:

`Tell`(KB, Percept([Smell, Breeze, None], 5))

`Ask`(KB, ∃a. BestAction(a, 5))

i.e., does the KB entail some best action at t=5?

Substitution

- Given a sentence S and a substitution σ,
  - $S\sigma$ denotes the result of "plugging" σ into S; e.g.,

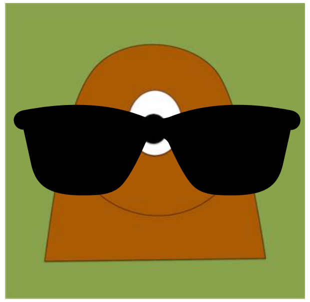
$$S = \text{Smarter}(x, y)$$

$$\sigma = \{x/\text{Agent}_1, y/\text{Wumpus}_1\}$$

$$S\sigma = \text{Smarter}(\text{Agent}_1, \text{Wumpus}_1)$$

- `Ask`(KB, S) returns some/all σ such that KB $\models S\sigma$

Interacting with FOL KBs



- Suppose a Wumpus-world agent using a FOL KB perceives:

a smell and a breeze (but no glitter) at *t=5*:

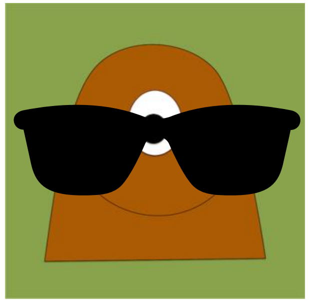
`Tell`(KB, Percept([Smell, Breeze, None], 5))

`Ask`(KB, ∃a. BestAction(a, 5))

i.e., does the KB entail some best action at t=5?

Answer: *Yes*, {*a*/`Shoot`} ← substitution (binding list)

KB for the Wumpus world

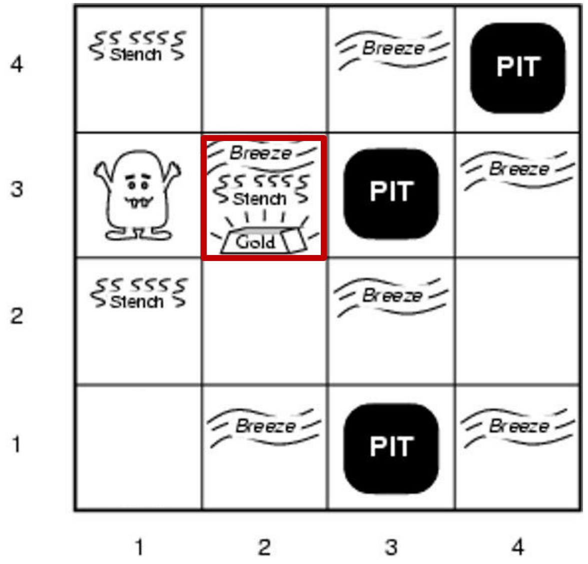


Perception

$\forall\, t,s,b.\ \mathrm{Percept}(\ [s,\ b,\ \mathrm{Glitter}],\ t) \Rightarrow \mathrm{Glitter}(t)$

Reflex

$\forall\, t.\ \mathrm{Glitter}(t) \Rightarrow \mathrm{BestAction}(\mathrm{Grab},\ t)$



Deducing hidden properties

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

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

- Squares are breezy near a pit:
  - Diagnostic rule: infer cause from effect

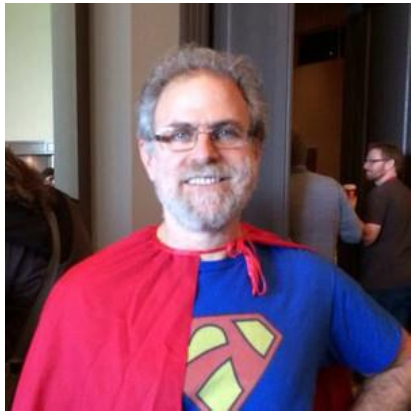
$$\forall s.\ \text{Breezy}(s) \Rightarrow \exists r.\ \text{Adjacent}(r, s) \land \text{Pit}(r)$$

  - Causal rule: infer effect from cause

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

Why?

- Universal ontology language.
  - e.g., databases, semantic web, knowledge graphs
- At the core of:
  - programming language semantics and type theory.
  - formal verification and advanced (> propositional) automated reasoning.
  - theorem proving, including in mathematics, physics, cryptography, and beyond.
  - logic programming and its derivations, expert systems, rule-based systems.
- Renewed interest in the context of explainable AI (XAI) and the "third-wave of AI".



Phil Wadler "*What does logic have to do with Java?*" 2009