Artificial Intelligence

First Order Logic

CS 444 – Spring 2019
Dr. Kevin Molloy
Department of Computer Science
James Madison University

JMU JAMES MADISON UNIVERSITY®

Inference-based Agent in Wumpus World

A wumpus-world agent using propositional logic:

$$\neg P_{1,1}$$
$$\neg W_{1,1}$$
$$B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$$
$$S_{x,y} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee W_{x-1,y})$$
$$W_{1,1} \vee W_{1,2} \vee \ldots \vee W_{4,4}$$
$$\neg W_{1,1} \vee \neg W_{1,2}$$
$$\neg W_{1,1} \vee \neg W_{1,3}$$
$$\ldots$$

64 distinct proposition symbols, 155 sentences.

Pros and Cons of Propositional Logic

PROS

- Propositional logic is declarative: pieces of syntax correspond to facts
- Propositional logic allows partial/disjunctive/negated information
- Propositional logic is compositional: meaning $B_{1,1} \wedge P_{1,2}$ is derived from the meaning of $B_{1,1}$ and $P_{1,2}$
- Meaning in propositional logic is context-independent (unlike natural language, where meaning depends on context).

CONS

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.

First-order logic

Whereas propositional logic assumes world contains **facts**, first-order logic (like natural language) assumes the world contains:

- **Objects**: people, houses, numbers, theories, Ronald McDonald, colors, baseball games, wars, …
- **Relations**: red, round, bogus, prime, brother of, isde, part of, has color,…
- **Functions**: father of, best friend of, third inning of, one more than, end of …

Logics in General

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, time	true/false/unknown
Probability theory	Facts	Degree of belief
Fuzzy logic	Facts + degree of truth	Known internal value

Syntax of FOL: Basic Elements

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

Atomic Sentences

Atomic Sentence = predicate $(term_1, \ldots, term_n)$

or $term_1, \ldots, term_n$

Term = $function(term_1, \ldots, term_n)$

or constant or variable

e.g., Brother (KingJohn, RichardTheLionheart)

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

Complex Sentences

Complex sentences are made from atomic sentences using connectives.

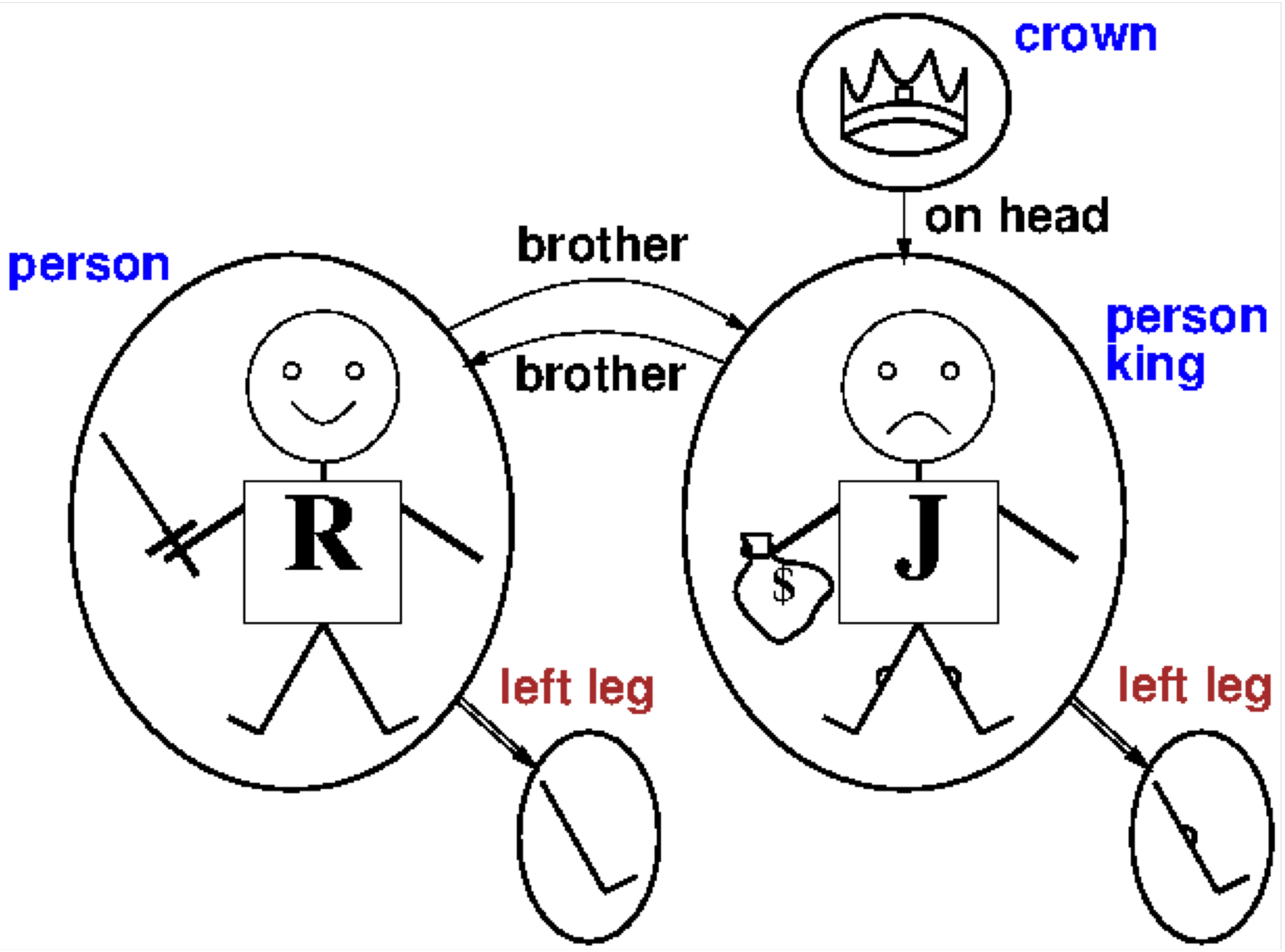
$\neg S$, $S_1 \land S_2$, $S_1 \lor S_2$ $S_1 \Rightarrow S_2$, $S_1 \Leftrightarrow S_2$

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

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

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

Models for FOL



Truth in First-order Logic

Sentences are true with respect to a model and an interpretation

Model contains ≥ 1 object (domain elements) and relations amongst them

Interpretation specifies referents for:

- Constant symbols → objects
- Predicate symbols → relations
- Function symbols → functional relations

An atomic sentence predicate ($term_1$, …, $term_n$) is true

iff the objects referred to by $term_1$, …, $term_n$ are in the relation referred to by the predicate

Models for FOL

Entailment in propositional logic can be computed by enumerating models

We can enumerate the FOL models for a given KB vocabulary:

For each number of domain elements n from 1 to ∞

For each k-ary predicate P_k in the vocabulary

For each possible k-ary relation on *n* objects

For each constant symbol C in the vocabulary

For each choice of referent for C from *n* objects

Computing entailment by enumerating FOL models is not easy!

Universal Quantification

$\forall$ <variables> <sentence>

Everyone at JMU is smart:

$\forall x \ At(x, JMU) \implies Smart(x)$

$\forall x \ P$ is true in a model m iff P is true with x being each possible object in the model

Roughly speaking, equivalent to the conjunction of instantiations of P

$$
\begin{array}{ll}
(& (At(KingJohn, JMU) \implies Smart(KingJohn)) \\
\land & (At(Richard, JMU) \implies Smart(Richard)) \\
\land & (At(Berkeley, JMU) \implies Smart(Berkeley)) \\
\land \ldots &
\end{array}
$$

A common Mistake to Avoid

Typically, $\Longrightarrow$ is the main connective with $\forall$

Common mistake: using $\land$ as the main connective with $\forall$:

$\forall x\ At(x, JMU) \land Smart(x)$ means

"Everyone is at JMU and everyone is smart".

Existential Quantification

∃ <variables><sentence>

Someone at Stanford is smart:

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

∃ x P is true in a model *m* iff P is true with *x* being *some* possible object in the model.

Roughly speaking, equivalent to the disjunction of instantiations of P

(At(KingJohn, Stanford) ∧ Smart(KingJohn))

∨ (At(Richard, Stanford) ∧ Smart(Richard))

∨ (At(Stanford, Stanford) ∧ Smart(Stanford))

Another Common Mistake to Avoid

Typically, $\wedge$ is the main connective with $\exists$

Common mistake: using $\Longrightarrow$ as the main connective with $\exists$:

$\exists x\ At(x, Stanford) \Longrightarrow Smart(x)$ is true if

there is anyone who is not at Stanford.

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 is loves everyone in the world).
$\forall y \exists x$	Loves(x, y)	"Everyone 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)

Fun with Sentences

Brothers are siblings

$\forall$ x, y Brother(x,y) $\Longrightarrow$ Sibling (x, y)

“Sibling” is symmetric

$\forall$ x, y Sibling(x,y) $\Longrightarrow$ 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)

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., $1 = 2$ and **$\forall$ x   X(Sqrt(x), Sqrt(x)) = x** are satisfiable

$2 = 2$ is valid

e.g. definition of (full) Sibling in terms of Parent:

$\forall$**x, y Sibling(x, y) $\Leftrightarrow$ [¬(x =y) ∧ ∃ m, f ¬(m = f)   ∧**

Parent(m, x) ∧ Parent(f, x) ∧ Parent(m, y) ∧ Parent (f,y)]

Interacting with FOL KBs

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

- Tell (KB, Percept([Smell, Breeze, None], 5))
- Ask(KB, ∃ a Action (a, 5))

i.e., does KB entail any particular actions at t = 5.

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

Given a sentence S and a substitution σ,

S_σ = denotes the result of plugging σ into S, e.g.

S = Smarter(x, y)

σ = {x/Liz, y/Kevin}

Sσ = Smarter(Liz, Kevin)

Ask(KB, S) returns some/all σ such that KB ⊨ Sσ

Knowledge Base for the Wumpus World

"Perception"

$\forall$ b, g, t Percept([Smell, b, g], t) $\Longrightarrow$ Smelt(t)

$\forall$ s, b, t Percept ([s, b, Glitter], t) $\Longrightarrow$ AtGold(t)

"Reflex" $\forall$ t AtGold(t) $\Longrightarrow$ Action(Grab, t)

Reflex with internal state: do we have the gold already?

$\forall$ t At Gold(t) $\land$ $\neg$Holding(Gold, t) $\Longrightarrow$ Action(Grab, t)

Holding(Gold, t) cannot be observed

Keeping track of change is essential.

Deciding Hidden Properties

Properties of locations:

$\forall$ x, t At(Agent, x, t) $\land$ Smelt(t) $\Longrightarrow$ Smelly(x)

$\forall$ x t At(Agent, x, t) $\land$ Breeze(t) $\Longrightarrow$ Breezy(x)

Squares are breezy near a pit (Diagnostic rule – infer cause from effect):

$\forall$ y Breezy(y) $\Longrightarrow$ $\exists$ x Pit(x) $\land$ Adjacent(x, y)

Causal rule – infer effect from cause):

$\forall$ x, y Pit(x) $\land$ Adjacent(x, y) $\Longrightarrow$ Breezy(y)

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

Definition for the Breezy predicate:

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

Keeping Track of Change

Facts hold in situations, rather than eternally.

e.g., Holding(Gold, Now) rather than just Holding(Gold)

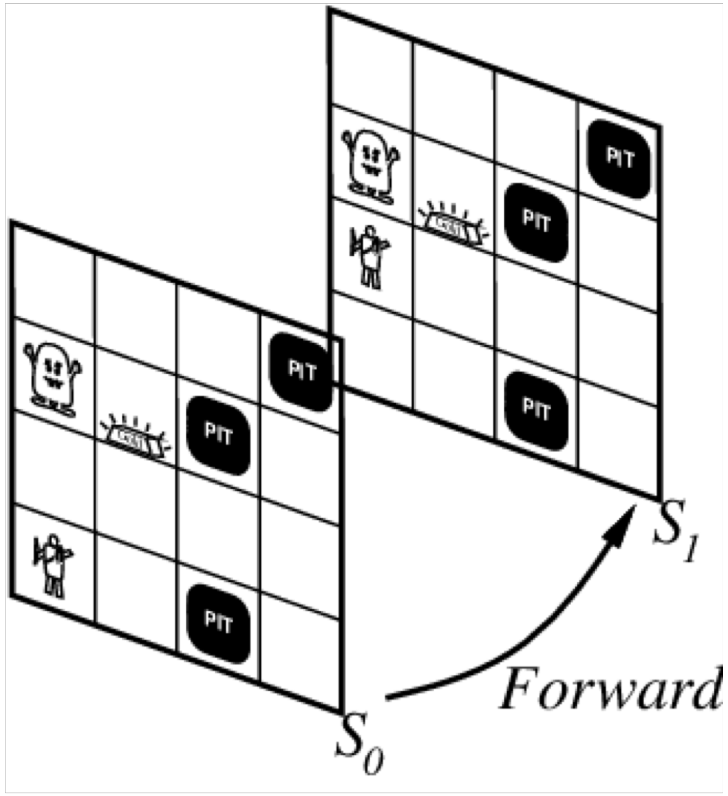
Situational calculus is one way to represent change in FOL:

Adds a situation argument to each non-eternal predicate.

e.g., now in Holding(Gold, Now) denotes a situation

Situations are connected by the Result function

Result(a, s) is the situation that results from doing a in s



Preliminaries on Situation Calculus

Situation calculus is a logic formalism designed for representing and reasoning about dynamical domains.

a dynamic world is modeled as progressing through a series of situations as a result of various actions being performed within the world

Introduced by John McCarthy in 1963. McCarthy described a situation as a state. Ray Reiter corrected this (1991):

> "A situation is a finite sequence of actions. Period. It's not a state, its not a snapshot, it's a history".

Describing Actions I

"Effect" axiom – describe changes due to action

$\forall$ s AtHold(s) $\Longrightarrow$ Holding(Gold, Result(Grab,s))

"Frame" axiom – describe non-changes due to action

$\forall$ s HaveArrow(s) $\Longrightarrow$ HaveArrow(Result(Grab,s))

Frame problem: find an elegant way to handle non-change:

a) Representation – avoid frame axioms

b) Inference – avoid repeated "copy-vers" to keep track of state

Qualification problem: true descriptions of real actions require endless caveats – what if gold is slippery or nailed down or …

Ramification problem: real actions have many secondary consequences – what about the dust on the gold, wear and tear on gloves, …

Describing Actions II

Successor-state axioms solve the representational frame problem

Each axiom is "about" a predicate (not an action per se):

P true afterwards ⇔ [an action made P true ∨ P true already and no action made P false]

For holding the gold:

$\forall$ a, s Holding (Gold, Result(a, s)) ⇔

[(a = Grab ∧ AtHold(s)) ∨ (Holding(Gold, s) ∨ a ≠ Release)]

Making Plans (a better way)

Represents plans as action sequences $[a_1, a_2, \ldots, a_n]$

PlanResult(p, s) is the result of executing p in s

Query: Ask (KB, $\exists$ p Holding(Gold, PlanResult(p, S_0))

Has the solution: s/ Result(Grab, Result(Forward, S_0))

Definition of PlanResult in terms of Result:

$\forall$ s PlanResult([], s) = s

$\forall$ a, p, s PlanResult([a|p], s) = PlanResult(p, Result(a,s))

Planning systems are special-purpose reasoners designed to do this type of inference more efficiently than a general-purpose reasoner.