Note

- Homework #2
  - On Lecture 4-5, Due in one week
  - Accessible in *Canvas's* "Assignment" section, click the link
  - Save your hand-written or typed answers into a single PDF file, combining all pages. Use any PDF scan app or multifunction printers.
  - Submit the PDF file to "Submission for HW #2" link by 3/4 4pm. **No late submission. Strictly applied**.
- Fast Prototyping Exercise starts today
  - First session with prep slides.
  - You must have read the reference paper: https://bidal.sfsu.edu/~kazokada/csc872/PD1.pdf

KB-Agent

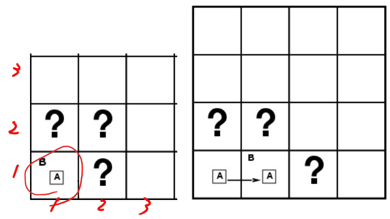
Knowledge-Based Agents with First-Order Predicate Logic

CSC 872

Pattern Analysis and Machine Intelligence

Review

- **Limitation of *Propositional Logic***



$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
$B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

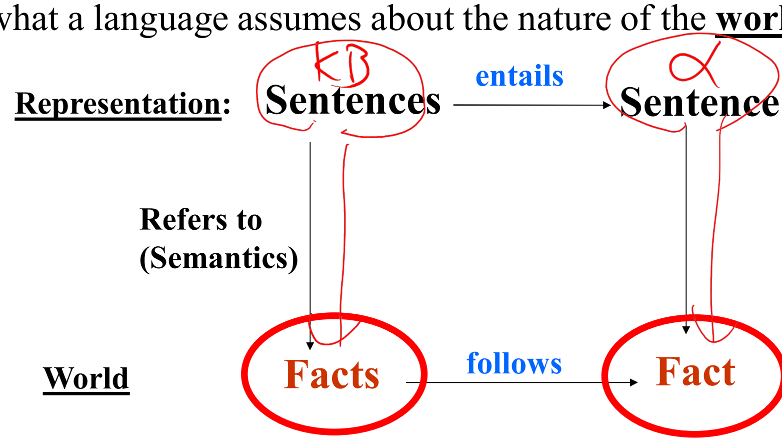
You need 16 rules
Imagine very big grids
…

- Even for a simple problem like the Wumpus world, KB gets quickly very large.
- This is because *Propositional Logic* only represents ***facts*** in the world

3

Propositional Logic Revisited

<u>Ontological commitments</u>:
what a language assumes about the nature of the **<u>world</u>**.



4

First Order Logic?

What is the ontological commitments for the FOL?

Representation: Sentences $\xrightarrow{\text{entails}}$ Sentence

Refers to (Semantics)

World: Facts, Objects, Relations $\xrightarrow{\text{follows}}$ Fact, Objects, Relations

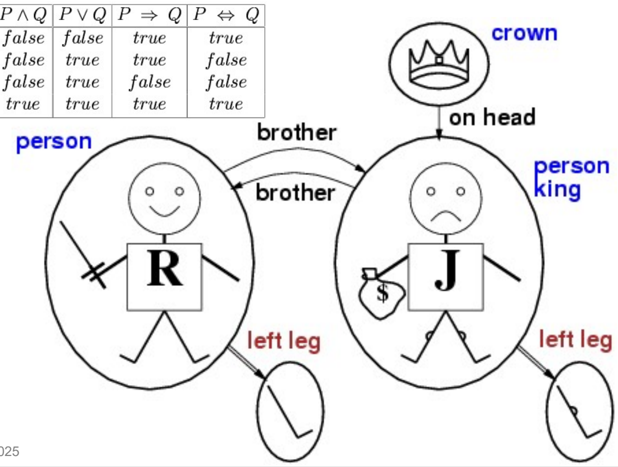
First Order Logic (FOL)

- First-order logic (like natural language) assumes the world contains:
  - **Objects**:
    - car, wheel, door, body, engine, seat, passenger, driver, Parth, Ishika
  - **Relations**:
    - **Properties (unary)**: Red(car), Healthy(body), IsOn(engine)
    - **N-ary relations**: Inside(car, Leela), Bigger(car, Ananyaa)
  - **Functions**:
    - ColorOf(car), MakeOf(car), SizeOf(engine)
- **Function** returns an object.
- **Relation** returns a truth value.

Models for FOL

- Propositional: sets of truth values for all prop. symbols
- FOL: same above + objects & relations!!!

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	*false*	*true*	*false*	*false*	*true*	*true*
false	*true*	*true*	*false*	*true*	*true*	*false*
true	*false*	*false*	*false*	*true*	*false*	*false*
true	*true*	*false*	*true*	*true*	*true*	*true*



Syntax: Basic Elements

Example Symbols

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

Syntax: Terms FOL

- **Term**: logical expression referring to an object

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

- Examples:
  - TeacherOf(*Abdoulfatah*), StudentOf(*Kaz*)
  - *Kaz*, *Harsh*, *Juve*, *Sai*, *Dhvanil*, *Matthew, Duy*…
  - **x**, referring to {*Sai*, *Adrian*,…| all students in csc872}

Syntax: Atomic Sentences

- **Atomic sentence**: a sentence that states a fact (indicating a proposition)

AtomicSentence → **Predicate(Term, …)**
| Term1 = Term2

- Examples:
  - Classmate(*Alexander*, *Zoe*),
  - TeacherOf(*Kenji*) = TeacherOf(*Mohammad*) (= *Kaz*)

Syntax: Complex Sentences

Sentence → AtomicSentence / ∨ ∧ ⊕ → ↔
| (Sentence Connective Sentence)
| Quantifier Variable, … Sentence
∀ ∃
| ¬ Sentence

- Examples:
  - $((S1 \wedge S2) \vee S3)$, $((S1 \vee S3) \Rightarrow S2)$, $(S1 \Leftrightarrow S3)$
  - Student(*Kaz*, *Anurag*) ⇒ Teacher(*Anurag*, *Kaz*)
  - ∀ *x Smart(x, CSC872)*
  - ¬ ∀ *x Perfect(x, CSC872)*

Syntax: Predicate & Function

- Function := Term
  - Returns an object
- Predicate := Sentence
  - Either true or false
- Correspondence between them
  - Function: **friend_of(*Divya*) = *Gabrielle***
  - Predicate: **friend_of(*Diviya*, *Gabrielle*)**

Semantics: Models & Interpretations

- **Model** + **Interpretation** → truth of sentences
- **Model** contains (**objects**=**domain elements**, **relations**)
- **Interpretation** specifies referents for symbols
  - constant symbols → objects
  - predicate symbols → relations
  - function symbols → functional relations

Semantics: Truth Assignments

- ***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***
- ***term$_1$ = term$_2$*** is **true** under an interpretation *iff* ***term$_1$*** and ***term$_2$*** refer to the same object
- For complex sentences: Use the same rules for propositional logic

Quantifiers

- Goal: expressing sentences about ***collections*** of objects without enumeration
- Aim: avoid naming every constants (propositional logic must do this: e.g. Wumpus)

- **Variable: x,y,a,b**
- **Universal quantification (*for all*):** $\forall$
  - e.g., All students in CS872 are smart $\forall x\ Smart(x, CS872)$
- **Existential quantification (*there exists*):** $\exists$
  - e.g., Someone in the class is sleeping $\exists x\ Sleeping(x, CS872)$

Universal Quantification (*for all*): $\forall$

Syntax: $\forall$ ***<variables> <sentence>***

- English: *"All students in CS872 are smart"*:
- FOL: $\forall x \quad In872(x) \Rightarrow Smart(x)$

- $\forall$ **x** corresponds to the **conjunction** of all instantiations of **x**

$$
\begin{array}{ll}
\quad In872(Andre) & \Rightarrow Smart(Andre) \\
\wedge\ In872(Banaz) & \Rightarrow Smart(Banaz) \\
\wedge\ \ldots & \\
\wedge\ In872(Atharva) & \Rightarrow Smart(Atharva)
\end{array}
$$

Universal Quantification (*for all*): $\forall$

- **$\Rightarrow$ is a natural connective to use with $\forall$**
- *Common mistake*: <u>to use $\wedge$ together with $\forall$</u>
  - e.g: $\forall$ *x* In(CS872, *x*) $\wedge$ Smart(*x*)
  - means *"every one is in CS872 and everyone is smart"*

Existential quantification (*there exists*): $\exists$

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

- English: *"Someone in the class is sleeping"*:
- FOL: $\exists$ *x* In(CS872, *x*) $\wedge$ Sleeping(*x*)

- $\exists$ **x** corresponds to the **disjunction** of all instantiations of **x**

(In(CS872, Brandon) $\wedge$ Sleeping(Brandon))
$\vee$ (In(CS872, Veronica) $\wedge$ Sleeping(Veronica))
$\vee$ …
$\vee$ (In(CS872, Mohanad) $\wedge$ Sleeping(Mohanad))

Existential quantification (*there exists*): ∃

- ∧ **is a natural connective to use with ∃**

- *Common mistake*: to use ⇒ together with ∃
  - e.g: ∃ *x* In(CS872, *x*) ⇒ Sleeping(*x*)
  - True even for someone not in CS872!
  - (False ⇒ True/False) are Valid!

A	B	A→B
T	T	T
T	F	F
F	T	T
F	F	T

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 $\forall y \ \exists x$

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

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

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

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

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

Translating FOL to English

The kinship domain: ¬ ∧ ∨ → ↔

- $\forall$ x,y *Brother(x,y)* $\Rightarrow$ *Sibling(x,y)*
  - Brothers are siblings
- $\forall$ m,c *Mother(c) = m* $\overset{\text{if}}{\Rightarrow}$ *(Female(m)∧Parent(m,c))*
  - One's mother is one's female parent
- $\forall$x,y *Sibling(x,y)* $\overset{\text{iff}}{\Leftrightarrow}$ *Sibling(y,x)*
  - "Sibling" is a symmetric relationship

Translating English to FOL

gardener(x)
likes(x,y)
person(x)
fool(x,y)
purple(x)
mushroom(x)
poisonous(x)

- Every gardener likes the sun.
  $\forall$ x *gardener*(x) $\Rightarrow$ *likes*(x,Sun)
- You can fool some of the people all of the time.
  $\exists$ x $\forall$ t (*person*(x) ^ *time*(t)) $\Rightarrow$ *can-fool*(x,t)
- You can fool everyone some of the time.
  $\forall$ x $\exists$ t (*person*(x) ^ *time*(t)) $\Rightarrow$ *can-fool*(x,t)
- All purple mushrooms are poisonous.
  $\forall$ x (*mushroom*(x) ^ *purple*(x)) $\Rightarrow$ *poisonous*(x)
- No purple mushroom is poisonous.
  ¬($\exists$ x) *mushroom*(x) ^ *purple* (x) ^ *poisonous*(x)
  ($\forall$ x) ¬(*mushroom*(x) ^ *purple*(x)) v ¬*poisonous*(x)
  ($\forall$ x) (*mushroom*(x) ^ *purple*(x)) $\Rightarrow$ ¬*poisonous*(x)
  All purple mushroom is not poisonous

STOP: Higher-Order Logic

- First-order logic **allows quantification over objects** (= the first-order entities that exist in the world).
- Higher-order logic also **allows quantification over relations and functions**.

  e.g., "two objects are equal *iff* all properties applied to them are equivalent":
  $\forall$ x,y   (x=y) $\Leftrightarrow$ ($\forall$*p*, *p*(x) $\Leftrightarrow$ *p*(y))

- Higher-order logics are more expressive than first-order; however, so far we have little understanding on how to effectively reason with sentences in higher-order logic (**no sound and complete inference procedure known**)

23

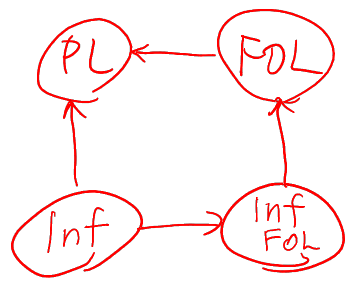
Using FOL in KB Agent

- Ground Terms: a term without variables Func (terms) Constant
- Substitution/Binding List: {variable/GTerm}
  - Given a sentence *S*   e.g., Faster(x,y)
  - Given a **substitution** σ e.g., {x/Car},{y/Turtle}
  - Plugging σ to *S*: ***S*σ**   e.g., Faster(Car, Turtle)
- Knowledge-Based Agent
  - ASK(*KB,S*)
  - **KB Agent answers if KB entails S with any σ; and also return some/all σ such that *KB* $\models$ *S*σ**
- Example
  - TELL(*KB*, *Percept*([Smell,Breeze,…], t=5)
  - ASK(*KB*, $\exists$ a *Action*(a,t=5))
  - Answers: Yes (True) {a/Shoot} {a/ more ops}

24

Inference for FOL

- Propositionalization
  – Translate KB in FOL into one in PL then apply propositional inference
- Lifting
  – Generalized Modus Ponens
  – Unification
  – Forward chaining
  – Backward chaining
  – Resolution



Universal Instantiation: UI

- **A new inference rule for FOL for eliminating** $\forall$
  for any sentence α, variable x and ground term τ,

$$\frac{\forall x \quad \alpha(x)}{\alpha\{x/\tau\}} \Leftrightarrow \frac{\alpha(x_1)\wedge\alpha(x_2)\wedge\cdots\wedge\alpha(x_n)}{\alpha\{x/k\},\ k\in U}$$

- e.g.,

$$\frac{\forall x\ King(x) \wedge Greedy(x) \Rightarrow Evil(x)}{\begin{array}{l} King(John) \wedge Greedy(John) \Rightarrow Evil(John) \quad \{x/John\} \\ King(Richard) \wedge Greedy(Richard) \Rightarrow Evil(Richard) \quad \{x/Richard\} \end{array}}$$

Existential Instantiation: EI

- **A new inference rule for FOL for eliminating ∃**

  for any sentence α, variable x and constant symbol k **not** in KB,

  $x_i \in U$

$$\frac{\exists x \;\; \alpha(x)}{\alpha\{x/k\}}$$

$\exists x \; \alpha(x) \Leftrightarrow \alpha(x_1) \vee \alpha(x_2) \vee \cdots \vee \alpha(x_n)$

$\alpha\{x/k\}$ $k \notin U$

- e.g.,

$$\frac{\exists x \; Crown(x) \wedge OnHead(x, John)}{Crown(C_1) \wedge OnHead(C_1, John)}$$

$C_1 \notin U$

- C_1 Skolem constant

Propositionalization

- **Every KB in FOL can be propostionalized so as to preserve entailment**
  - Reducing KB in FOL to the new one in PL using UI and EI rules
  - Perform propositional inference on the new KB
- **Herbrand (1930)** If a sentence α is entailed by an FOL KB, it is entailed by a finite subset of the propositionalized KB

Func(Func(Func(Term)))

Terms → Constant
| Variable
| Func(Term, ...)

- Idea: For n = 0 to ∞ do
  - create a propositional KB by instantiating with depth-n terms
  - see if α is entailed by this KB

Father of( )

Problems of Propositionalization

- Infinitely many ground terms
  - e.g., *Father*(*Father*(*Father*(*John*)))
- Generate lots of irrelevant sentences
- The approach based on **Herbrand's theorem** works only for entailed sentences
- **Turing-Church Theorem** (1936):
  - Entailment for FOL is semidecidable
  - algorithms exist that say yes to every entailed sentence, but no algorithm exists that also says no to every nonentailed sentence.)

Lifting

- Raising an inference rule from propositional to first-order logic and apply them directly to FOL KB
- Generalized Modus Ponens (GMP)

$$\frac{P \quad P \to Q}{Q}$$

$$\frac{p'_1,\ p'_2,\ \ldots,\ p'_n,\quad (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{q\theta} \qquad \text{where } p'_i\,\theta = p_i\,\theta \text{ for all } i$$

$$\frac{King(John),\ Greedy(y),\quad (King(x) \wedge Greedy(x) \Rightarrow Evil(x))}{Evil(John)}$$

with $\theta = \{x/John\}$
$\{y/John\}$

- GMP used with KB of definite clauses (Horn clause with exactly one positive literal)
- All variables assumed universally quantified

Unification

- **Unify(α,β) = θ if $\alpha\theta$ = $\beta\theta$**
- θ is called unifier

α	β	θ
Knows(*John*,*x*)	Knows(*John*,*Jane*)	{x/Jane}
Knows(*John*,*x*)	Knows(*y*,*OJ*)	{x/OJ,y/John}
Knows(*John*,*x*)	Knows(*y*,*Mother(y)*)	{x/Mother(John),y/John}
Knows(*John*,*x*)	Knows(*x*,*OJ*)	{fail}
	Knows(z,OJ)	{x/OJ, z/John}

- **Standardizing apart**
  - Changing *x* in Knows(*x*,*OJ*) to *z* that does not clash
- **Most General Unifier**
  - Unique unifier that is most general when more than one unifier possible

CSC872: PAMI – Kazunori Okada (C) 2025 31

31

Forward & Backward Chaining

- Transform KB to Definite Clauses
- Apply Generalized Modus Ponens
  - From premises: Forward Chaining
  - From conclusion: Backward Chaining
- Sound and Complete for FOL in definite clauses

IBM Watson
→ Jeoparady

- Forward Chaining
  - Basis for Deductive Databases (e.g., expert systems)
- Backward Chaining
  - Basis for Logical Programming (e.g., Prolog)

CSC872: PAMI – Kazunori Okada (C) 2025 32

32

Resolution

- Forward & backward chaining are NOT complete in general FOL
- Complete theorem (Gödel,1930)
  - any sentence entailed by a set of sentences can be proven from that set.
  - shows that it is possible to find **sound** and **complete** inference rules.
- Resolution (Robinson, 1965)
  - Proof by contradiction: ($KB \land \neg \boldsymbol{\alpha}$) is unsatisfiable → $KB \models \alpha$
  - Conjunctive Normal Form
  - Apply the resolution inference rule sequentially until finding unsatisfiable sentence (empty clause).

Lifted Resolution Rule

$$\frac{\alpha \lor \beta, \quad \neg\beta \lor \gamma}{\alpha \lor \gamma}$$

$$\frac{\begin{array}{c} p_1 \lor \cdot p_{j-1}\ p_j\ p_{j+1} \cdot \lor p_m, \\ q_1 \lor \cdot q_{k-1}\ q_k\ q_{k+1} \cdot \lor q_n \end{array}}{(p_1 \lor \ldots\ p_{j-1} \lor p_{j+1}\ \ldots p_m \lor q_1 \ldots\ q_{k-1} \lor q_{k+1}\ \ldots \lor q_n)\sigma}$$

where $p_j\sigma = \neg q_k\sigma$ $\quad$ $Unify(P_j, \neg q_k) = \sigma$

For example,

$$\frac{\begin{array}{l} \neg Rich(x) \lor Unhappy(x) \\ Rich(Me) \end{array}}{Unhappy(Me)}$$

with $\sigma = \{x/Me\}$

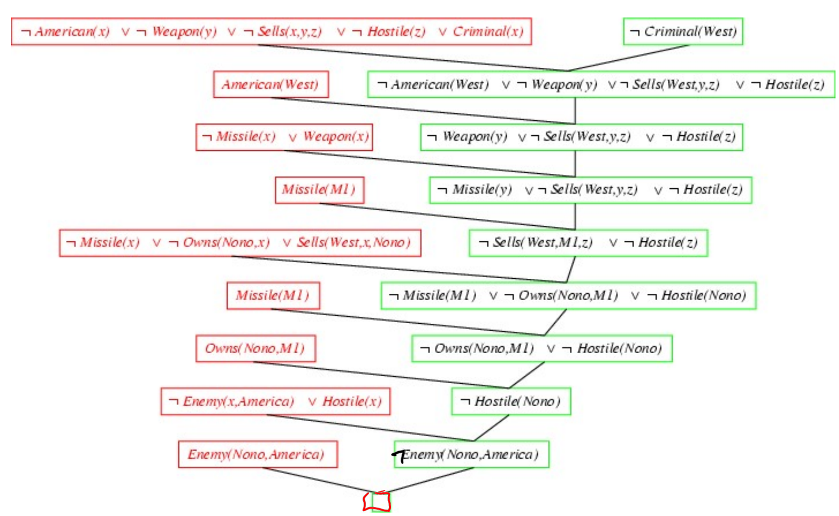
CNF Conversion for FOL

Any FOL KB can be converted to CNF as follows:

1. Replace $P \Rightarrow Q$ by $\neg P \vee Q$
2. Move $\neg$ inwards, e.g., $\neg \forall x\, P$ becomes $\exists x\, \neg P$
3. Standardize variables apart, e.g., $\forall x\, P \vee \exists x\, Q$ becomes $\forall x\, P \vee \exists y\, Q$
4. Move quantifiers left in order, e.g., $\forall x\, P \vee \exists x\, Q$ becomes $\forall x \exists y\, P \vee Q$
5. Eliminate $\exists$ by Skolemization (~~next slide~~)
6. Drop universal quantifiers
7. Distribute $\wedge$ over $\vee$, e.g., $(P \wedge Q) \vee R$ becomes $(P \vee Q) \wedge (P \vee R)$

35

Resolution Example



36

More Complex Applications

- Ontological Engineering (Ch.10)
  - Objects, Categories, Inheritance, Taxonomic hierarchy
- Situation Calculus (Ch.10)
  - Conventions for describing actions and changes
- Planning (Ch.11)
  - Deriving a sequence of actions that will achieve a goal
  - Can formulate planning as inference on a situation calculus KB

Logic/Knowledge-Based Agents?

- You are a self-driving agent
- You are now at an intersection
- You check the traffic signal
- Signal turns to green
- You move forward
- A car suddenly run over and clash you and you die.
- What went wrong as an agent-based picture???

We did it right according to what we have learned

- Multi-agent (different agents different models)
- Stochastic/Dynamic environment
- Probability Theory (Facts + Uncertainty)

Summary

- FOL is more powerful
  - Objects + Relations as Ontological Commitments
  - Quantifiers introduced
- Sound and Complete Inference available
  - Propositionalization
  - Forward & backward chaining
  - Resolution
- Next
  - Ontological Commitment: Facts + Uncertainty?
  - Probability Theory
  - Bayesian Framework
- Fast Prototyping Exercise #1: 1st-session after the break.