



# CS 5/7320 Artificial Intelligence

## Knowledge-Based Agents AIMA Chapters 7-9

---

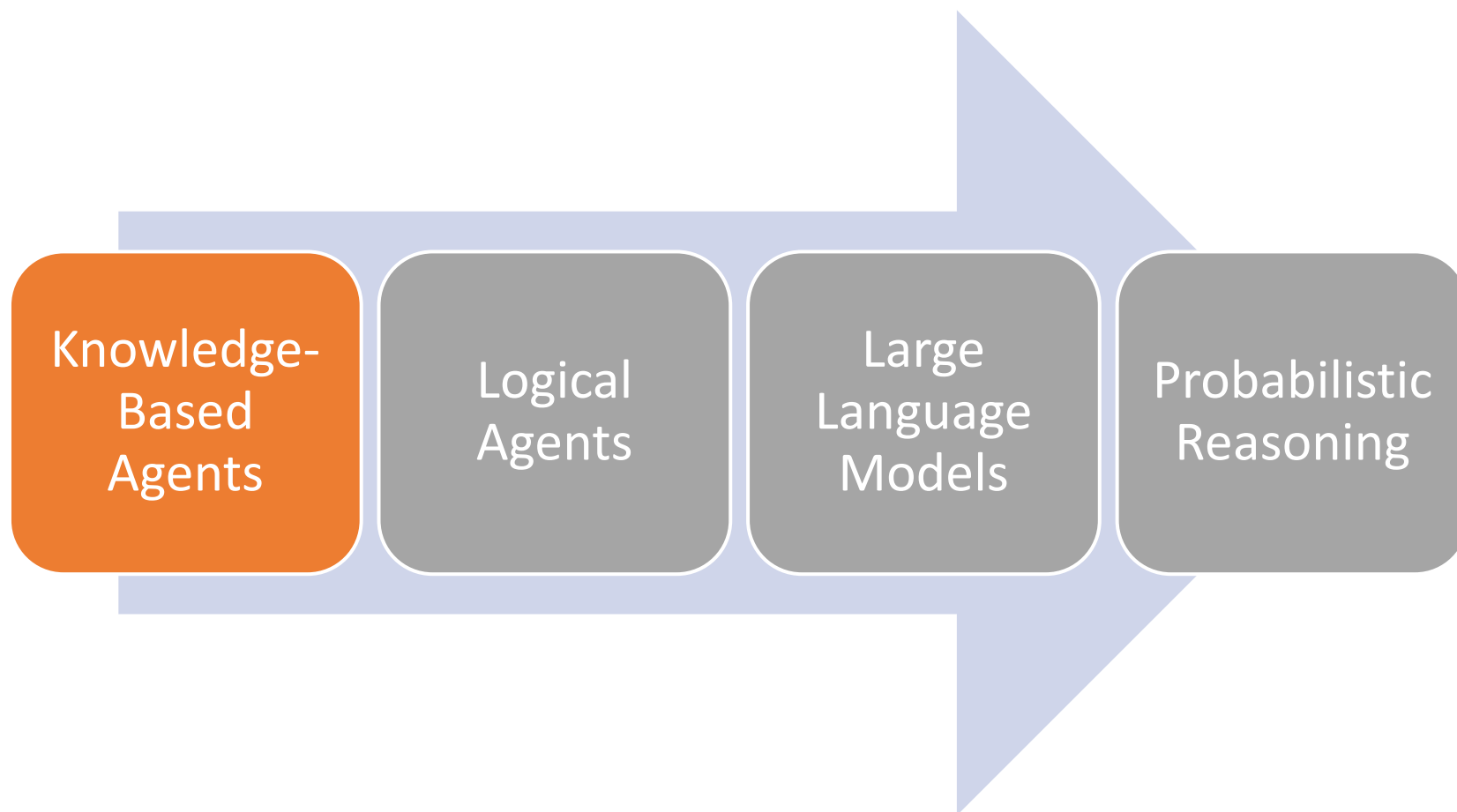
Slides by Michael Hahsler  
based on slides by Svetlana Lazepnik  
with figures from the AIMA textbook



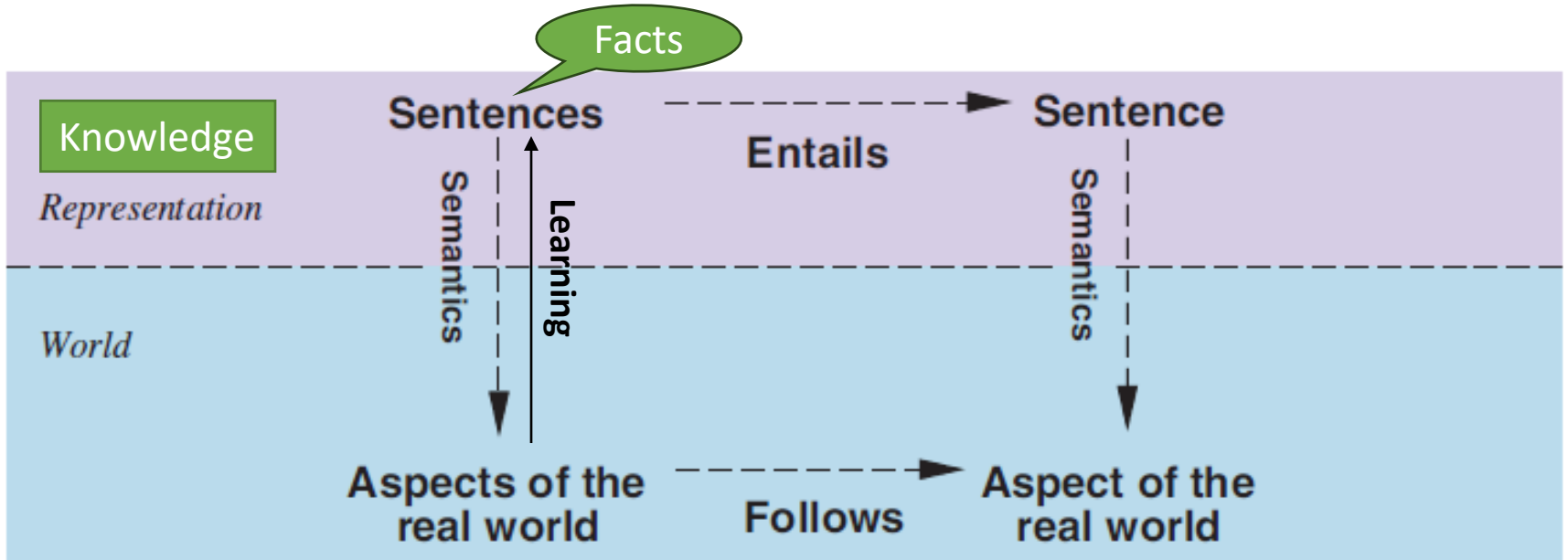
This work is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-sa/4.0/).

["Exercise Plays Vital Role Maintaining Brain Health"](#)  
by [A Health Blog](#)

# Outline

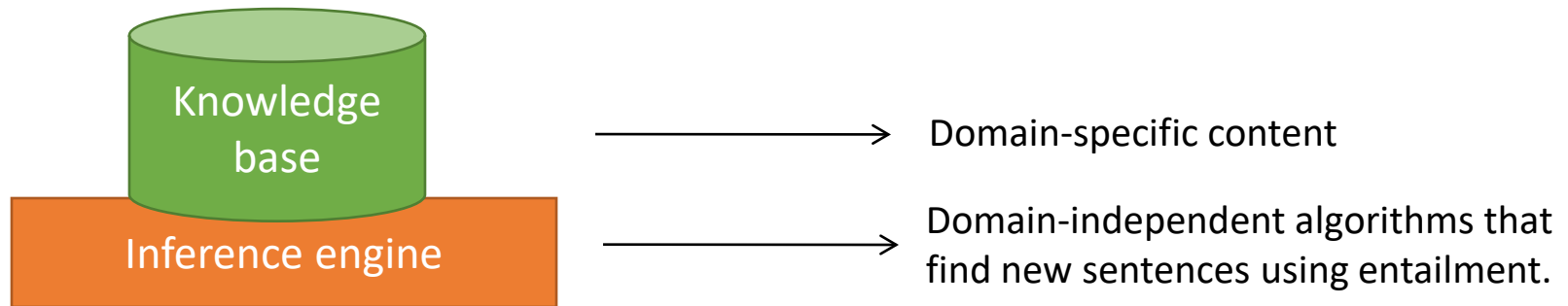


# Reality vs. Knowledge Representation



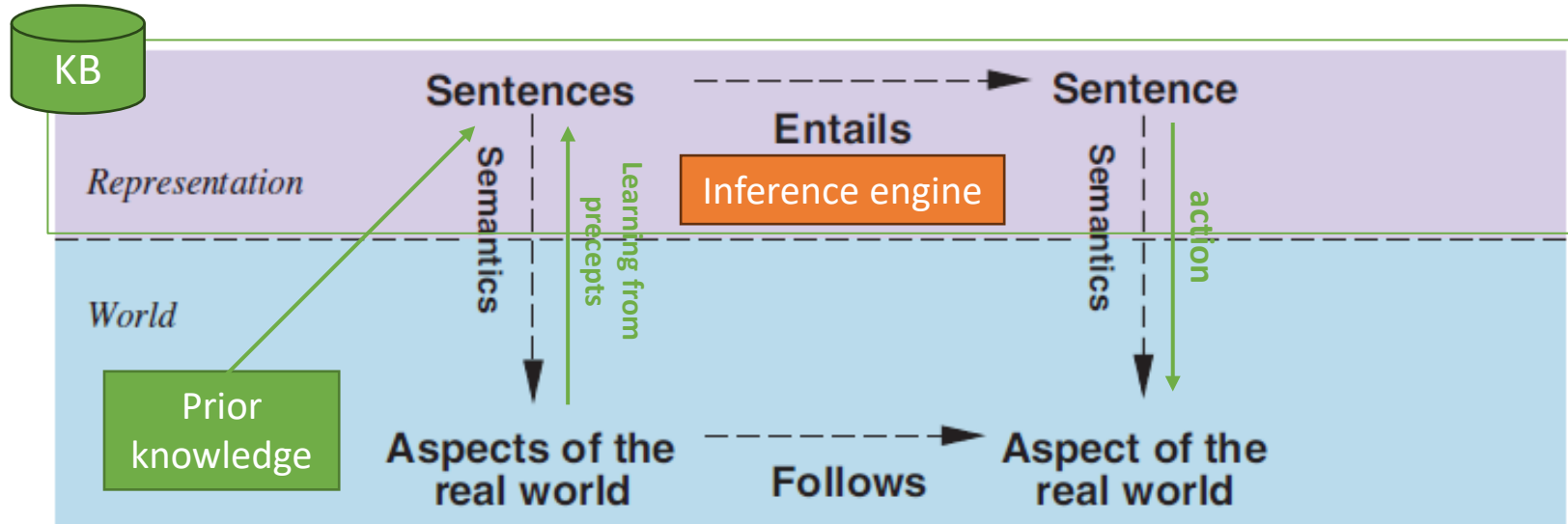
- **Facts:** Sentences we know to be true.
- **Possible worlds:** all worlds/models which are consistent with the facts we know (compare with belief state).
- **Learning** new facts reduces the number of possible worlds.
- **Entailment:** A new sentence logically follows from what we already know.

# Knowledge-Based Agents



- Knowledge base (KB) = **set of facts**. E.g., set of **sentences** in a **formal language** that are known to be true.
- **Declarative** approach to building an agent: Define what it needs to know in its KB.
- **Separation** between data (knowledge) and program (inference).
- Actions are based on knowledge (sentences + inferred sentences) + an **objective function**. E.g., the agent knows the effects of 5 possible actions and chooses the action with the largest utility.

# Generic Knowledge-based Agent



**function** KB-AGENT(*percept*) **returns** an *action*

**persistent:** *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*

Memorize percept at time *t*

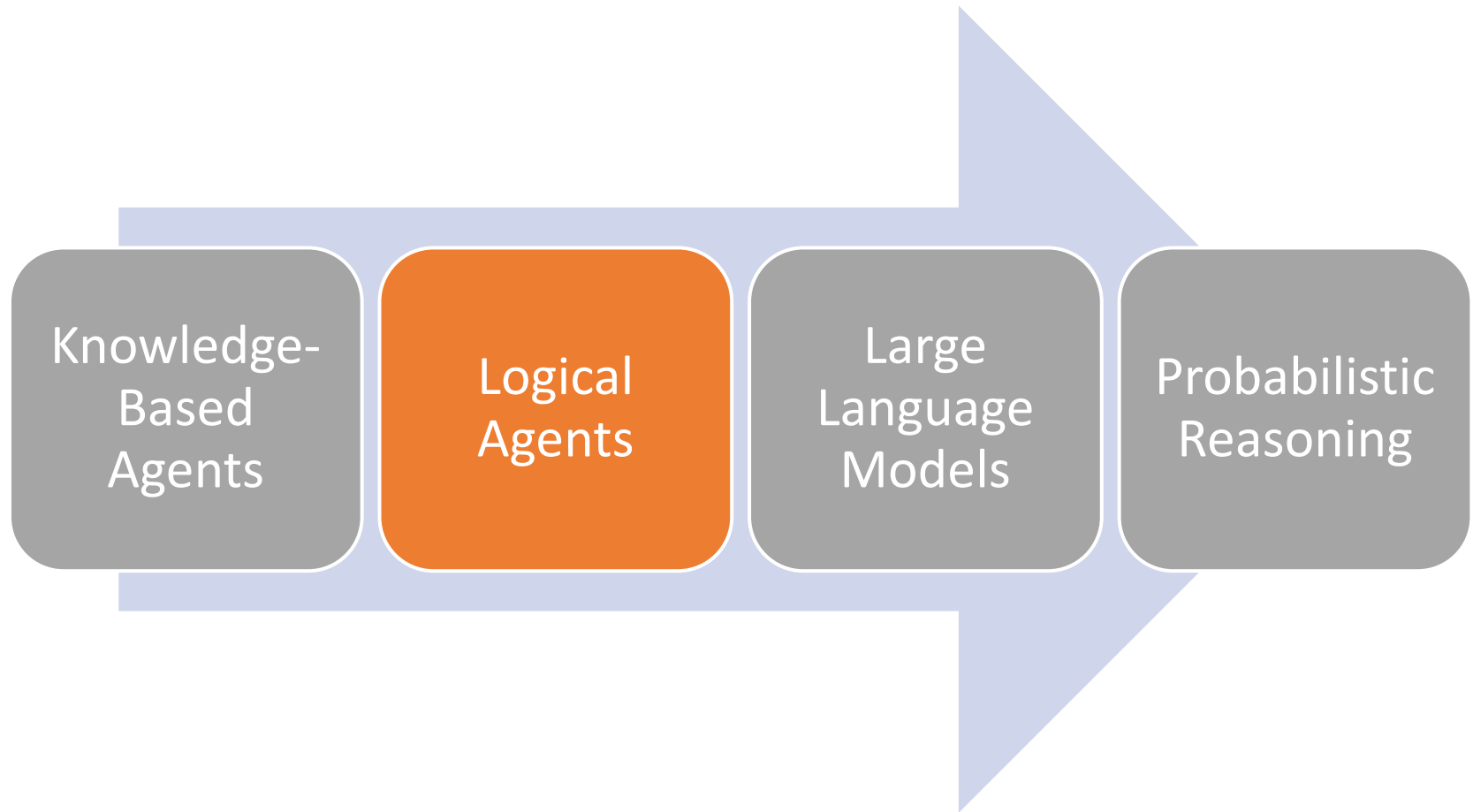
Ask for logical action given an objective

Record action taken at time *t*

# Different Languages to Represent Knowledge

Language	Ontological Commitment (What exists in the world)	Epistemological Commitment (What an agent believes about facts)
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	facts with degree of truth $\in [0, 1]$	known interval value
+ Natural Language	facts, objects, relations, ...	???

# Outline



# Logical Agents

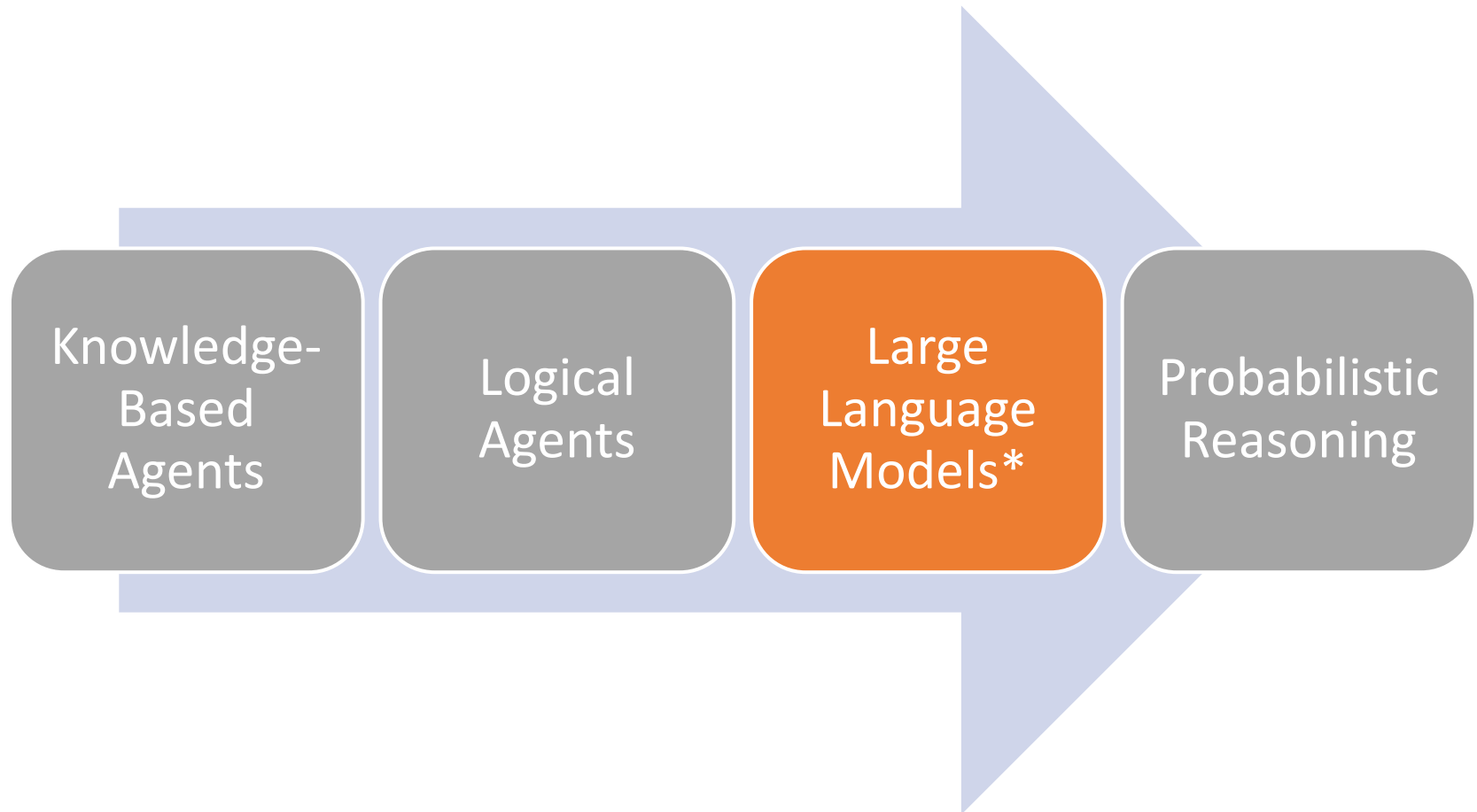
Language	Ontological Commitment (What exists in the world)	Epistemological Commitment (What an agent believes about facts)
Propositional logic	facts	true/false/unknown
First-order logic	facts, objects, relations	true/false/unknown
Temporal logic	facts, objects, relations, times	true/false/unknown
Probability theory	facts	degree of belief $\in [0, 1]$
Fuzzy logic	facts with degree of truth $\in [0, 1]$	known interval value

+ Natural Language      facts, objects, relations, ...      ???

- Facts are logical sentences that are known to be true.
- Inference: Generate new sentences that are entailed by all known sentences.
- Issues:
  - Inference is computationally very expensive.
  - Logic cannot deal with uncertainty.



# Outline



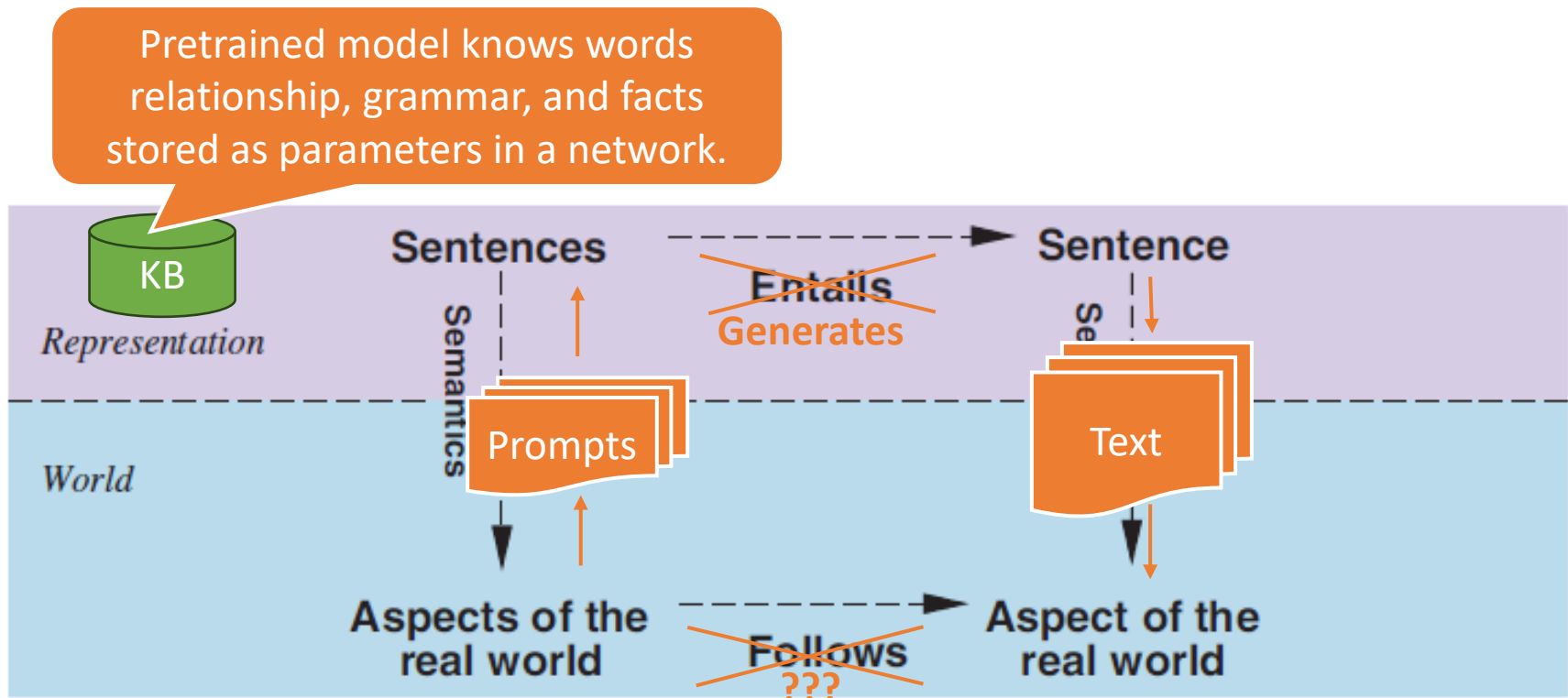
**\* This is not in the AIMA textbook!**

# LLMs - Large Language Models

Language	Ontological Commitment (What exists in the world)	Epistemological Commitment (What an agent believes about facts)
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	facts with degree of truth $\in [0, 1]$	known interval value
+ Natural Language	facts, objects, relations, ...	???

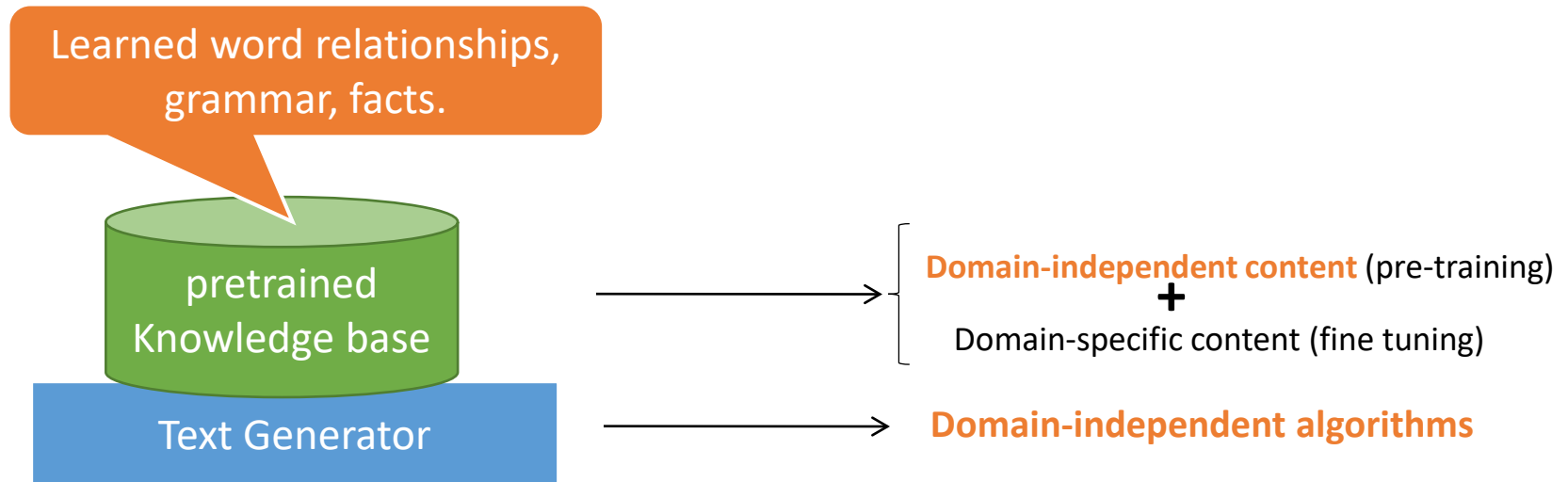
- Store knowledge as parameters in neural networks.

# Using Natural Language for Knowledge Representation



- The user formulates a question about the real world as a natural language prompt (a sequence of tokens).
- The LLM generates text using a model representing its knowledge base.
- The text (hopefully) is useful in the real world. The **objective function** is not clear. Maybe it is implied in the prompt?

# LLM as a Knowledge-Based Agents



Current text generators are:

- Pretrained decoder-only transformer models (e.g., GPT stands for Generative Pre-trained Transformer). The knowledge base is not updated during interactions.
- Tokens are created autoregressively. One token is generated at a time based on all the previous tokens using the transformer attention mechanism.

# LLM as a Generic Knowledge-based Agent

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

~~TELL(*KB*, MAKE-PERCEPT-SENTENCE(*percept*, *t*))~~

*action*  $\leftarrow$  ASK(*KB*, MAKE-ACTION-QUERY(*t*))

~~TELL(*KB*, MAKE-ACTION-SENTENCE(*action*, *t*))~~

*t*  $\leftarrow$  *t* + 1

**return** *action*

Ask for generating text (generate new tokens using autoregression)

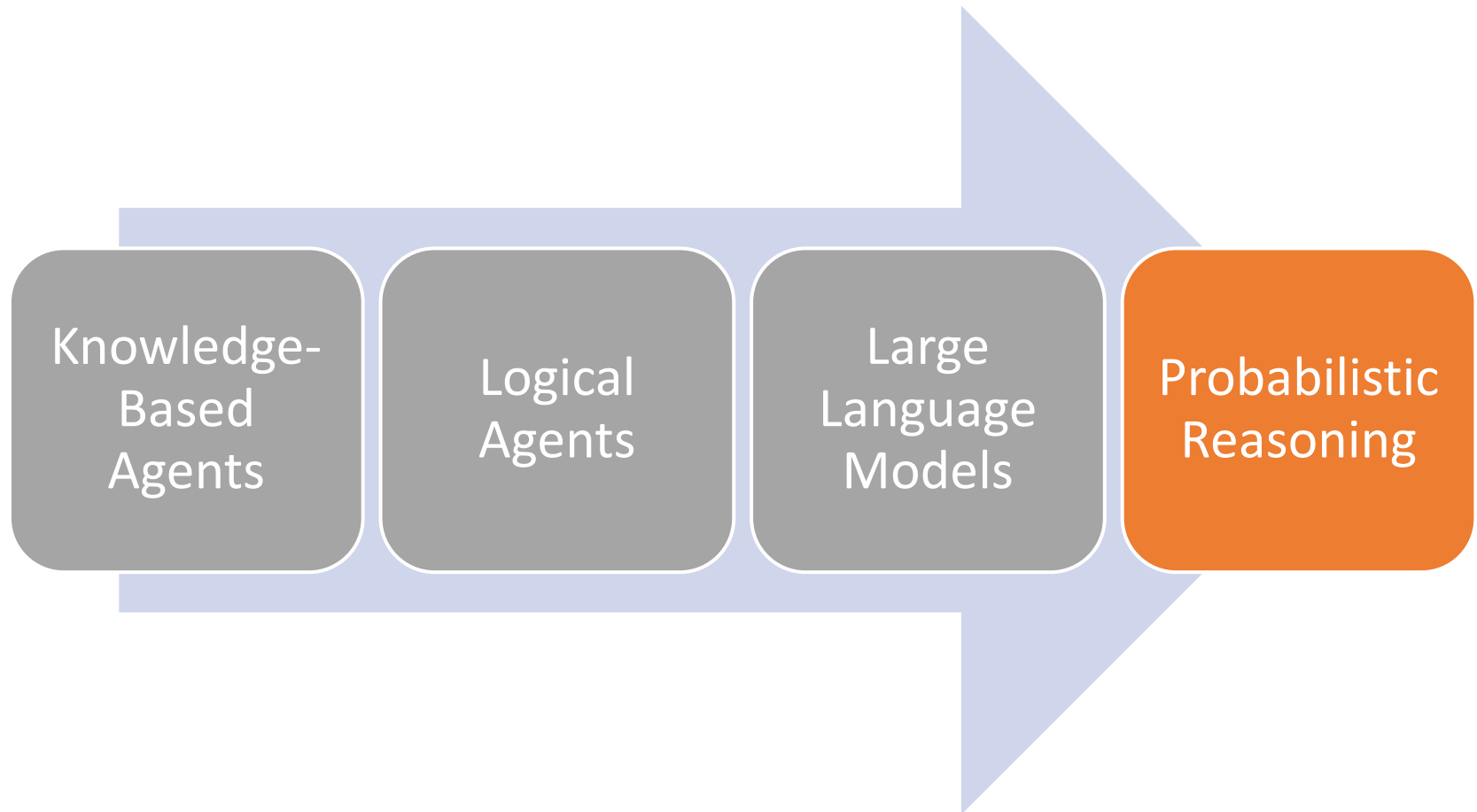
# Many Open Questions about LLMs

- Correlation is not causation: **Can LLMs reason** to solve problems?
- Generative stochasticity leads to **hallucinations**: LLM makes up facts.
- The training data contains **biases**, nonsense and harmful content.
- **Security**: LLM reveals sensitive information.
- Rights-laundering: **Copyrighted or licensed material** in the training data
- Leaky data makes it hard to evaluate true **performance**.

Reading: [2307.04821] Amplifying Limitations, Harms and Risks of Large Language Models (arxiv.org)



# Outline



# Probabilistic Reasoning

Language	Ontological Commitment (What exists in the world)	Epistemological Commitment (What an agent believes about facts)
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	facts with degree of truth $\in [0, 1]$	known interval value

+ Natural Language      facts, objects, relations, ...      ???

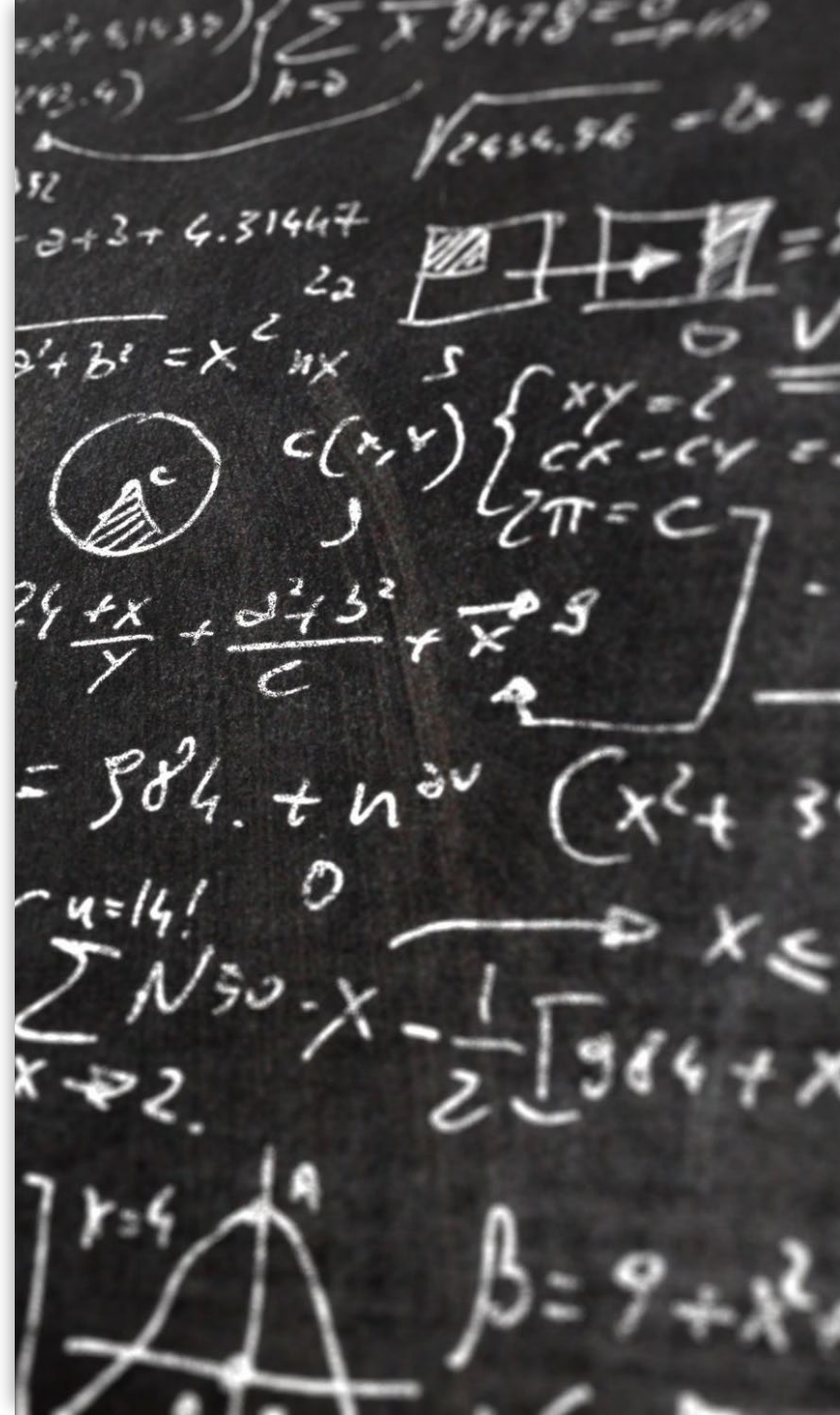
- Replaces true/false with a probability.
- Is the basis for
  - Probabilistic reasoning under uncertainty
  - Decision theory
  - Machine Learning

We will talk about these topics a lot more



# Conclusion

- The **clear separation between knowledge and inference engine** is very useful.
- **Pure logic** is often not flexible enough. The fullest realization of knowledge-based agents using logic was in the field of expert systems or knowledge-based systems in the 1970s and 1980s.
- **Pretrained Large Language Models** are an interesting new application of knowledge-based agents based on natural language.
- Next, we will talk about **probability theory** which is the standard language to reason under uncertainty and forms the basis of machine learning.





# Appendix: Logic

Details on Propositional and First-Order Logic



# Logic to Represent Knowledge



**Logic** is a formal system for representing and manipulating facts (i.e., knowledge) so that true conclusions may be drawn



**Syntax:** rules for constructing valid sentences

E.g.,  $x + 2 \geq y$  is a valid arithmetic sentence,  $\geq x^2y +$  is not



**Semantics:** “meaning” of sentences, or relationship between logical sentences and the real world

Specifically, semantics defines truth of sentences

E.g.,  $x + 2 \geq y$  is true in a world where  $x = 5$  and  $y = 7$

# Propositional Logic

Language	Ontological Commitment (What exists in the world)	Epistemological Commitment (What an agent believes about facts)
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	facts with degree of truth $\in [0, 1]$	known interval value

# Propositional Logic: Syntax in Backus-Naur Form

$Sentence \rightarrow AtomicSentence \mid ComplexSentence$

$AtomicSentence \rightarrow True \mid False \mid P \mid Q \mid R \mid \dots$  = Symbols

$ComplexSentence \rightarrow ( Sentence )$

$\mid \neg Sentence$

Negation

$\mid Sentence \wedge Sentence$

Conjunction

$\mid Sentence \vee Sentence$

Disjunction

$\mid Sentence \Rightarrow Sentence$

Implication

$\mid Sentence \Leftrightarrow Sentence$

Biconditional

OPERATOR PRECEDENCE :  $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$



# Validity and Satisfiability

A sentence is **valid** if it is true in **all** models/worlds

e.g., *True*,  $A \vee \neg A$ ,  $A \Rightarrow A$ ,  $(A \wedge (A \Rightarrow B)) \Rightarrow B$  are called tautologies and are useful to deduct new sentences.

A sentence is **satisfiable** if it is true in **some** model

e.g.,  $A \vee B$ ,  $C$   
useful to find new facts that satisfy all current possible worlds.

A sentence is **unsatisfiable** if it is true in no models

e.g.,  $A \wedge \neg A$

# Possible Worlds, Models and Truth Tables

A **model** specifies a “possible world” with the true/false status of each proposition symbol in the knowledge base

- E.g., **P** is true and **Q** is true
- With two symbols, there are  $2^2 = 4$  possible worlds/models, and they can be enumerated exhaustively using:

A **truth table** specifies the truth value of a composite sentence for each possible assignment of truth values to its atoms. Each row is a model.

$P$	$Q$	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>true</i>
<i>false</i>	<i>true</i>	<i>true</i>	<i>false</i>	<i>true</i>	<i>true</i>	<i>false</i>
<i>true</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>
<i>true</i>	<i>true</i>	<i>false</i>	<i>true</i>	<i>true</i>	<i>true</i>	<i>true</i>

We have 3 possible worlds for  $P \Rightarrow Q = \text{true}$

# Propositional Logic: Semantics

Rules for evaluating truth with respect to a model:

- $\neg P$  is true iff  $P$  is false
- $P \wedge Q$  is true iff  $P$  is true and  $Q$  is true
- $P \vee Q$  is true iff  $P$  is true or  $Q$  is true
- $P \Rightarrow Q$  is true iff  $P$  is false or  $Q$  is true

Sentence

Model



# Logical Equivalence

Two sentences are **logically equivalent** iff (read if, and only if) they are true in same models

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

# Entailment

- **Entailment** means that a sentence **follows from** the premises contained in the knowledge base:

$$KB \models \alpha$$

- The knowledge base  $KB$  entails sentence  $\alpha$  iff  $\alpha$  is true in all models where  $KB$  is true
  - E.g., KB with  $x = 0$  entails sentence  $x * y = 0$
- Tests for entailment
  - $KB \models \alpha$  iff  $(KB \Rightarrow \alpha)$  is *valid*
  - $KB \models \alpha$  iff  $(KB \wedge \neg \alpha)$  is *unsatisfiable*

# Inference

- **Logical inference:** a procedure for generating sentences that follow from (or entailed by) a knowledge base KB.
- An inference procedure is **sound** if it derives a sentence  $\alpha$  iff  $KB \models \alpha$ . I.e, it only derives **true sentences**.
- An inference procedure is **complete** if it can derive **all**  $\alpha$  for which  $KB \models \alpha$ .

# Inference

- How can we check whether a sentence  $\alpha$  is entailed by KB?
- How about we **enumerate all possible models of the KB** (truth assignments of all its symbols), and check that  $\alpha$  is true in every model in which KB is true?
  - This is sound: All produced answer are correct.
  - This is complete: It will produce all correct answers.
  - **Problem:** if KB contains  $n$  symbols, the truth table will be of size  $2^n$
- Better idea: use ***inference rules***, or sound procedures to generate new sentences or *conclusions* given the *premises* in the KB.
- Look at the textbook for inference rules and resolution.

# Inference Rules

- Modus Ponens

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

← premises  
← conclusion

This means: If the KB contains the sentences  $\alpha \Rightarrow \beta$  and  $\alpha$  then  $\beta$  is true.

- And-elimination

$$\frac{\alpha \wedge \beta}{\alpha}$$

# Inference Rules

- And-introduction

$$\frac{\alpha, \beta}{\alpha \wedge \beta}$$

- Or-introduction

$$\frac{\alpha}{\alpha \vee \beta}$$

# Inference Rules

- Double negative elimination

$$\frac{\neg\neg\neg\alpha}{\alpha}$$

- Unit resolution

$$\frac{\alpha \vee \beta, \neg\beta}{\alpha}$$

# Resolution

$$\frac{\alpha \vee \beta, \neg \beta \vee \gamma}{\alpha \vee \gamma}$$

or

$$\frac{\alpha \vee \beta, \beta \Rightarrow \gamma}{\alpha \vee \gamma}$$

- Example:

$\alpha$ : "The weather is dry"

$\beta$ : "The weather is rainy"

$\gamma$ : "I carry an umbrella"



# Resolution is Complete

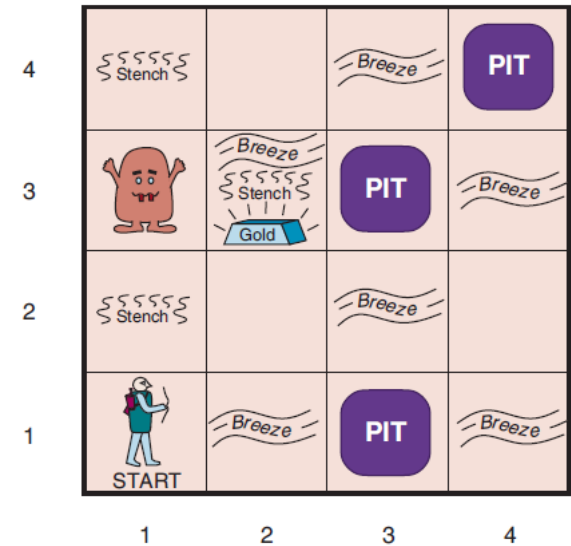
$$\frac{\alpha \vee \beta, \neg \beta \vee \gamma}{\alpha \vee \gamma}$$

- To prove  $KB \models \alpha$ , assume  $KB \wedge \neg \alpha$  and derive a contradiction
- Rewrite  $KB \wedge \neg \alpha$  as a conjunction of *clauses*, or disjunctions of *literals*
  - *Conjunctive normal form* (CNF)
- Keep applying resolution to clauses that contain *complementary literals* and adding resulting clauses to the list
  - If there are no new clauses to be added, then KB does not entail  $\alpha$
  - If two clauses resolve to form an *empty clause*, we have a contradiction and  $KB \models \alpha$

# Complexity of Inference

- Propositional inference is ***co-NP-complete***
  - *Complement* of the SAT problem:  $\alpha \models \beta$  if and only if the sentence  $\alpha \wedge \neg \beta$  is *unsatisfiable*
  - Every known inference algorithm has worst-case exponential run time complexity.
- Efficient inference is only possible for restricted cases
  - e.g., Horn clauses are disjunctions of literals with at most one positive literal.

# Example: Wumpus World



1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2	2,2	3,2	4,2
1,1 OK A OK	2,1 OK	3,1	4,1

(a)

**A** = Agent  
**B** = Breeze  
**G** = Glitter, Gold  
**OK** = Safe square  
**P** = Pit  
**S** = Stench  
**V** = Visited  
**W** = Wumpus

1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 A B OK	3,1 P?	4,1

(b)

# Example: Wumpus World

**Initial KB** needs to contain rules like these for each square:

$$Breeze(1,1) \Leftrightarrow Pit(1,2) \vee Pit(2,1)$$

$$Breeze(1,2) \Leftrightarrow Pit(1,1) \vee Pit(1,3) \vee Pit(2,2)$$

$$Stench(1,1) \Leftrightarrow W(1,2) \vee W(2,1)$$

...

**Percepts** at (1,1) are no breeze or stench. Add the following facts to the KB:

$$\neg Breeze(1,1)$$

$$\neg Stench(1,1)$$

**Inference** will tell us that the following facts are entailed:

$$\neg Pit(1,2), \neg Pit(2,1), \neg W(1,2), \neg W(2,1)$$

This means that (1,2) and (2,1) are safe.

We have to enumerate all possible scenarios in propositional logic! First-order logic can help.

# Summary

- Logical agents apply **inference** to a **knowledge base** to derive new information and make decisions.
- Basic concepts of logic:
  - **syntax**: formal structure of sentences
  - **semantics**: truth of sentences in models
  - **entailment**: necessary truth of one sentence given another
  - **inference**: deriving sentences from other sentences
  - **soundness**: derivations produce only entailed sentences
  - **completeness**: derivations can produce all entailed sentences
- Resolution is complete for propositional logic.
- Algorithms use forward, backward chaining, are linear in time, and complete for special clauses (definite clauses).

# Limitations of Propositional Logic

Suppose you want to say “All humans are mortal”

- In propositional logic, you would need ~6.7 billion statements of the form:

Michael\_Is\_Human and Michael\_Is\_Mortal,  
Sarah\_Is\_Human and Sarah\_Is\_Mortal, ...

Suppose you want to say “Some people can run a marathon”

- You would need a disjunction of ~6.7 billion statements:

Michael\_Can\_Run\_A\_Marathon or ... or Sarah\_Can\_Run\_A\_Marathon

# First-Order Logic

Language	Ontological Commitment (What exists in the world)	Epistemological Commitment (What an agent believes about facts)
Propositional logic	facts	true/false/unknown
First-order logic	facts, <span>objects, relations</span>	true/false/unknown
Temporal logic	facts, objects, relations, times	true/false/unknown
Probability theory	facts	degree of belief $\in [0, 1]$
Fuzzy logic	facts with degree of truth $\in [0, 1]$	known interval value

First-order Logic adds **objects** and **relations** to the facts of propositional logic.

This addresses the issues of propositional logic, which needs to store a fact for each instance of an object individually.

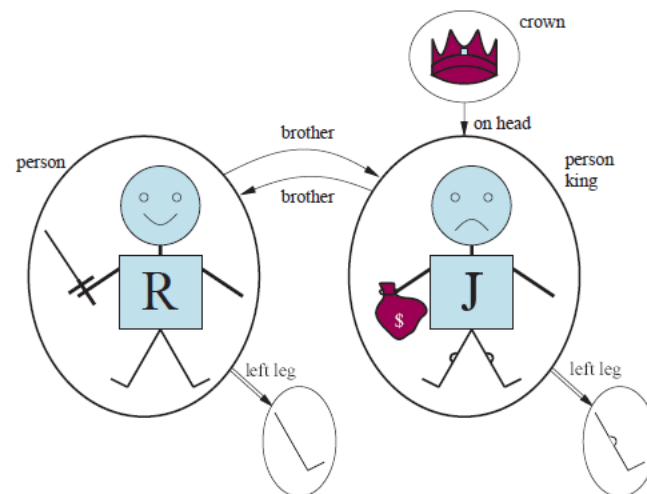
# Syntax of FOL

$Sentence \rightarrow AtomicSentence \mid ComplexSentence$   
 $AtomicSentence \rightarrow Predicate \mid Predicate(Term, \dots) \mid Term = Term$   
 $ComplexSentence \rightarrow (Sentence)$   
 $\quad \mid \neg Sentence$   
 $\quad \mid Sentence \wedge Sentence$   
 $\quad \mid Sentence \vee Sentence$   
 $\quad \mid Sentence \Rightarrow Sentence$   
 $\quad \mid Sentence \Leftrightarrow Sentence$   
 $\quad \mid Quantifier Variable, \dots Sentence$

$Term \rightarrow Function(Term, \dots)$   
 $\quad \mid Constant$   
 $\quad \mid Variable$

$Quantifier \rightarrow \forall \mid \exists$   
 $Constant \rightarrow A \mid X_1 \mid John \mid \dots$   
 $Variable \rightarrow a \mid x \mid s \mid \dots$   
 $Predicate \rightarrow True \mid False \mid After \mid Loves \mid Raining \mid \dots$   
 $Function \rightarrow Mother \mid LeftLeg \mid \dots$

OPERATOR PRECEDENCE :  $\neg, =, \wedge, \vee, \Rightarrow, \Leftrightarrow$



Objects

Relations. Predicate  
is/returns True or False

Function returns an object



# Universal Quantification

- $\forall x P(x)$
- Example: “Everyone at SMU is smart”  
 $\forall x \text{ At}(x, \text{SMU}) \Rightarrow \text{Smart}(x)$   
Why not  $\forall x \text{ At}(x, \text{SMU}) \wedge \text{Smart}(x)$ ?
- Roughly speaking, equivalent to the **conjunction** of all possible instantiations of the variable:  
 $[\text{At}(\text{John}, \text{SMU}) \Rightarrow \text{Smart}(\text{John})] \wedge \dots$   
 $[\text{At}(\text{Richard}, \text{SMU}) \Rightarrow \text{Smart}(\text{Richard})] \wedge \dots$
- $\forall x P(x)$  is true in a model  $m$  iff  $P(x)$  is true with  $x$  being each possible object in the model

# Existential Quantification

- $\exists \mathbf{x} \mathbf{P}(\mathbf{x})$
- Example: “Someone at SMU is smart”  
 $\exists \mathbf{x} \text{At}(\mathbf{x}, \text{SMU}) \wedge \text{Smart}(\mathbf{x})$   
Why not  $\exists \mathbf{x} \text{At}(\mathbf{x}, \text{SMU}) \Rightarrow \text{Smart}(\mathbf{x})$ ?
- Roughly speaking, equivalent to the **disjunction** of all possible instantiations:  
 $[\text{At}(\text{John}, \text{SMU}) \wedge \text{Smart}(\text{John})] \vee$   
 $[\text{At}(\text{Richard}, \text{SMU}) \wedge \text{Smart}(\text{Richard})] \vee \dots$
- $\exists \mathbf{x} \mathbf{P}(\mathbf{x})$  is true in a model  $m$  iff  $\mathbf{P}(\mathbf{x})$  is true with  $\mathbf{x}$  being some possible object in the model

# 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  $\forall y \exists x$

$\exists x \forall y \text{ Loves}(x,y)$

“There is a person who loves everyone”

$\forall y \exists x \text{ Loves}(x,y)$

“Everyone is loved by at least one person”

- **Quantifier duality:** each quantifier can be expressed using the other with the help of negation

$\forall x \text{ Likes}(x, \text{IceCream}) \quad \neg \exists x$

$\exists x \text{ Likes}(x, \text{Broccoli}) \quad \neg \forall x$

# Equality

- **Term<sub>1</sub> = Term<sub>2</sub>** is true under a given model if and only if **Term<sub>1</sub>** and **Term<sub>2</sub>** refer to the same object
- E.g., definition of **Sibling** in terms of **Parent**:  
$$\forall x,y \text{ Sibling}(x,y) \Leftrightarrow$$
$$[\neg(x = y) \wedge \exists m,f \neg (m = f) \wedge \text{Parent}(m,x) \wedge \text{Parent}(f,x) \wedge$$
$$\text{Parent}(m,y) \wedge \text{Parent}(f,y)]$$

## Example: The Kinship Domain

- Brothers are siblings

$$\forall x,y \text{ Brother}(x,y) \Rightarrow \text{Sibling}(x,y)$$

- “Sibling” is symmetric

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

- One's mother is one's female parent

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

## Example: The Set Domain

- $\forall s \text{ Set}(s) \Leftrightarrow (s = \{\}) \vee (\exists x, s_2 \text{ Set}(s_2) \wedge s = \{x | s_2\})$
- $\neg \exists x, s \{x | s\} = \{\}$
- $\forall x, s \ x \in s \Leftrightarrow s = \{x | s\}$
- $\forall x, s \ x \in s \Leftrightarrow [\exists y, s_2 (s = \{y | s_2\} \wedge (x = y \vee x \in s_2))]$
- $\forall s_1, s_2 \ s_1 \subseteq s_2 \Leftrightarrow (\forall x \ x \in s_1 \Rightarrow x \in s_2)$
- $\forall s_1, s_2 \ (s_1 = s_2) \Leftrightarrow (s_1 \subseteq s_2 \wedge s_2 \subseteq s_1)$
- $\forall x, s_1, s_2 \ x \in (s_1 \cap s_2) \Leftrightarrow (x \in s_1 \wedge x \in s_2)$
- $\forall x, s_1, s_2 \ x \in (s_1 \cup s_2) \Leftrightarrow (x \in s_1 \vee x \in s_2)$

# Inference in FOL

Inference in FOL is complicated!

1. **Reduction to propositional logic** and then use propositional logic inference.
2. **Directly do inference on FOL (or a subset like definite clauses)**
  - Unification: Combine two sentences into one.
  - Forward Chaining for FOL
  - Backward Chaining for FOL
  - Logical programming (e.g., Prolog)