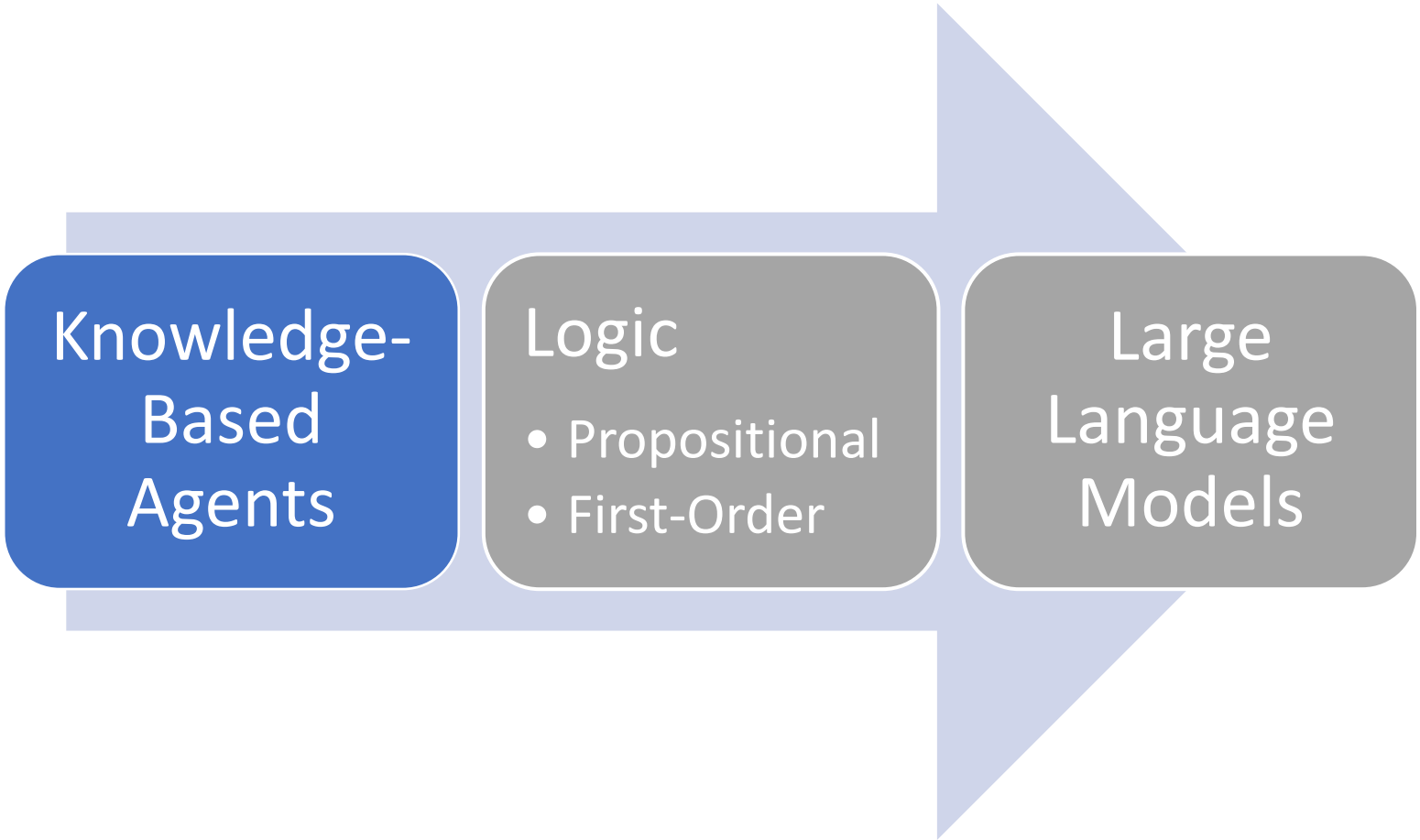
CS 5/7320
Artificial Intelligence

Knowledge-Based Agents
AIMA Chapters 7-9

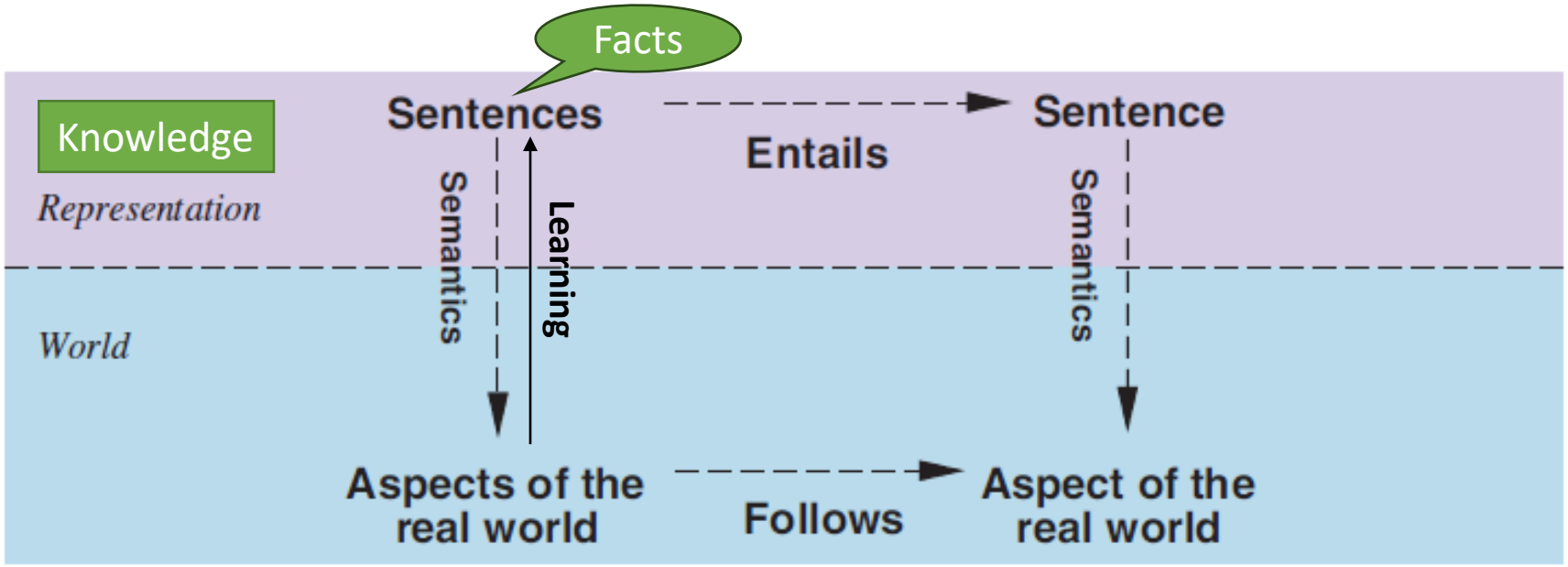
Slides by Michael Hahsler
based on slides by Svetlana Lazepnik
with figures from the AIMA textbook

"Exercise Plays Vital Role Maintaining Brain Health" by A Health Blog

Outline

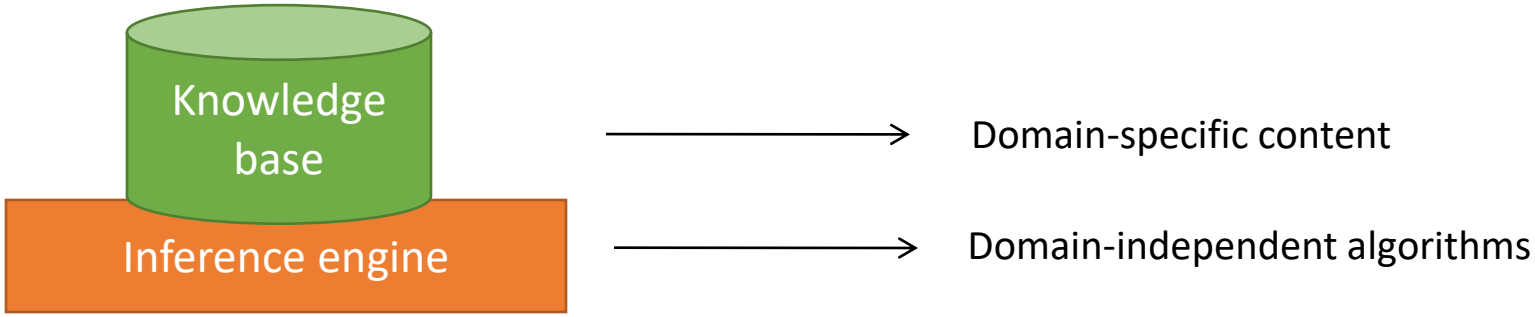

Knowledge-Based Agents
Logic
• Propositional
• First-Order
Large Language Models

Reality vs. Knowledge Representation



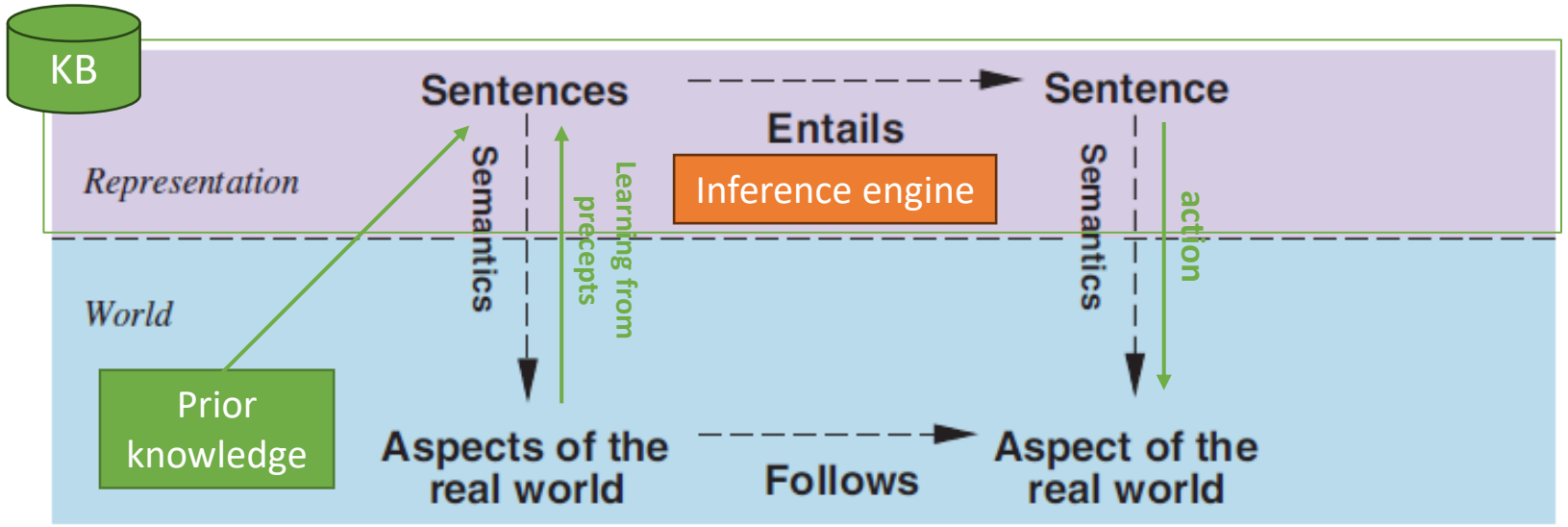
- **Facts:** Sentences we know to be true.
- **Possible worlds**: all worlds/models which are consistent with the facts we know (compare with belief state).
- **Learning** new facts reduces the number of possible worlds.
- **Entailment:** A new sentence logically follows from what we already know.

Knowledge-Based Agents



- Knowledge base (KB) = set of **sentences** in a **formal** language (knowledge representation) that are known to be true = **set of facts.**
- **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

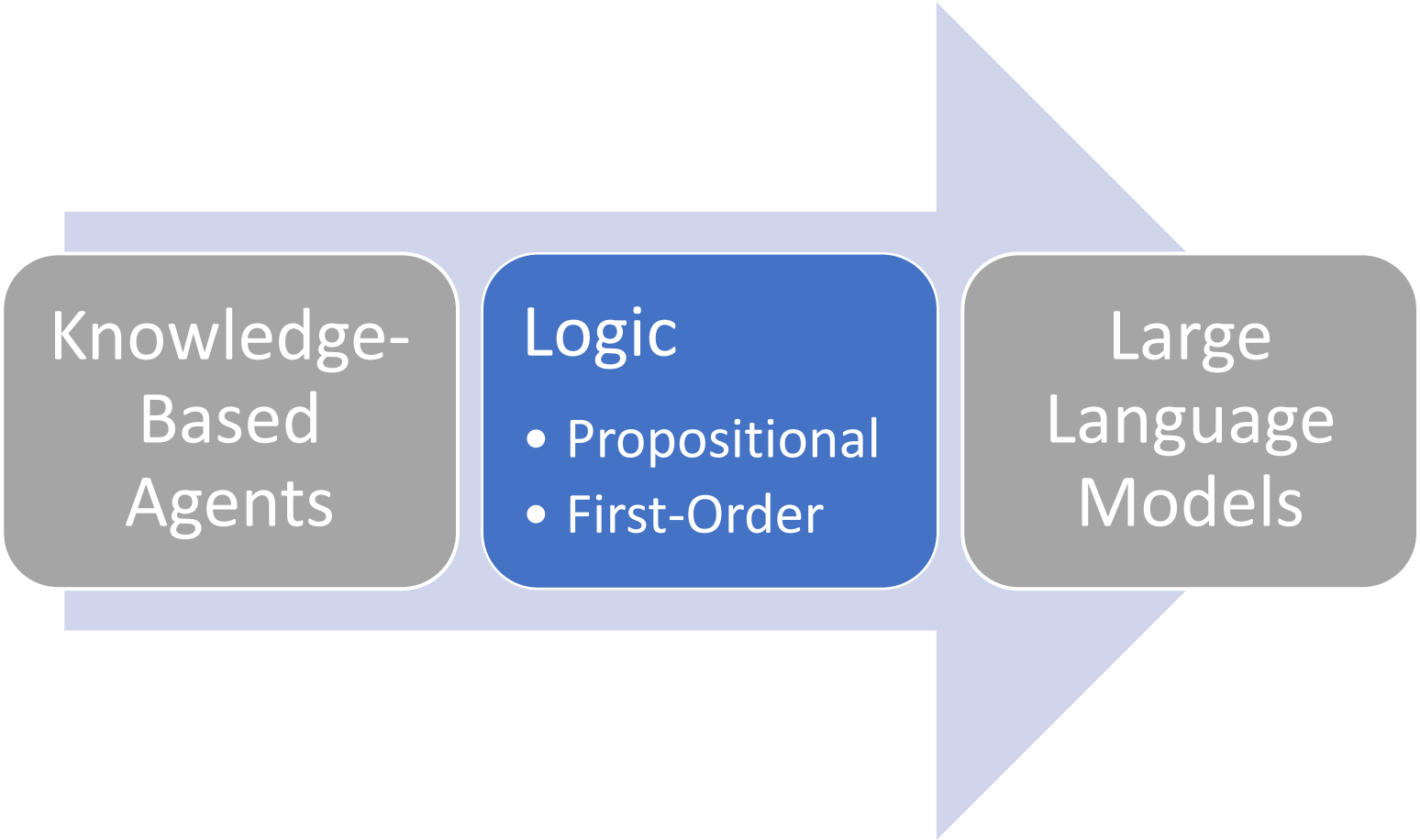
  TELL(*KB*, MAKE-PERCEPT-SENTENCE(*percept*, *t*))
  action ← ASK(*KB*, MAKE-ACTION-QUERY(*t*))
  TELL(*KB*, MAKE-ACTION-SENTENCE(*action*, *t*))
  t ← *t* + 1
  return *action*

Memorize percept at time t

Ask for logical action given an objective

Record action taken at time t

Outline


Knowledge-Based Agents
Logic
• Propositional
• First-Order
Large Language Models

Knowledge is Often Represented Using Logic

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

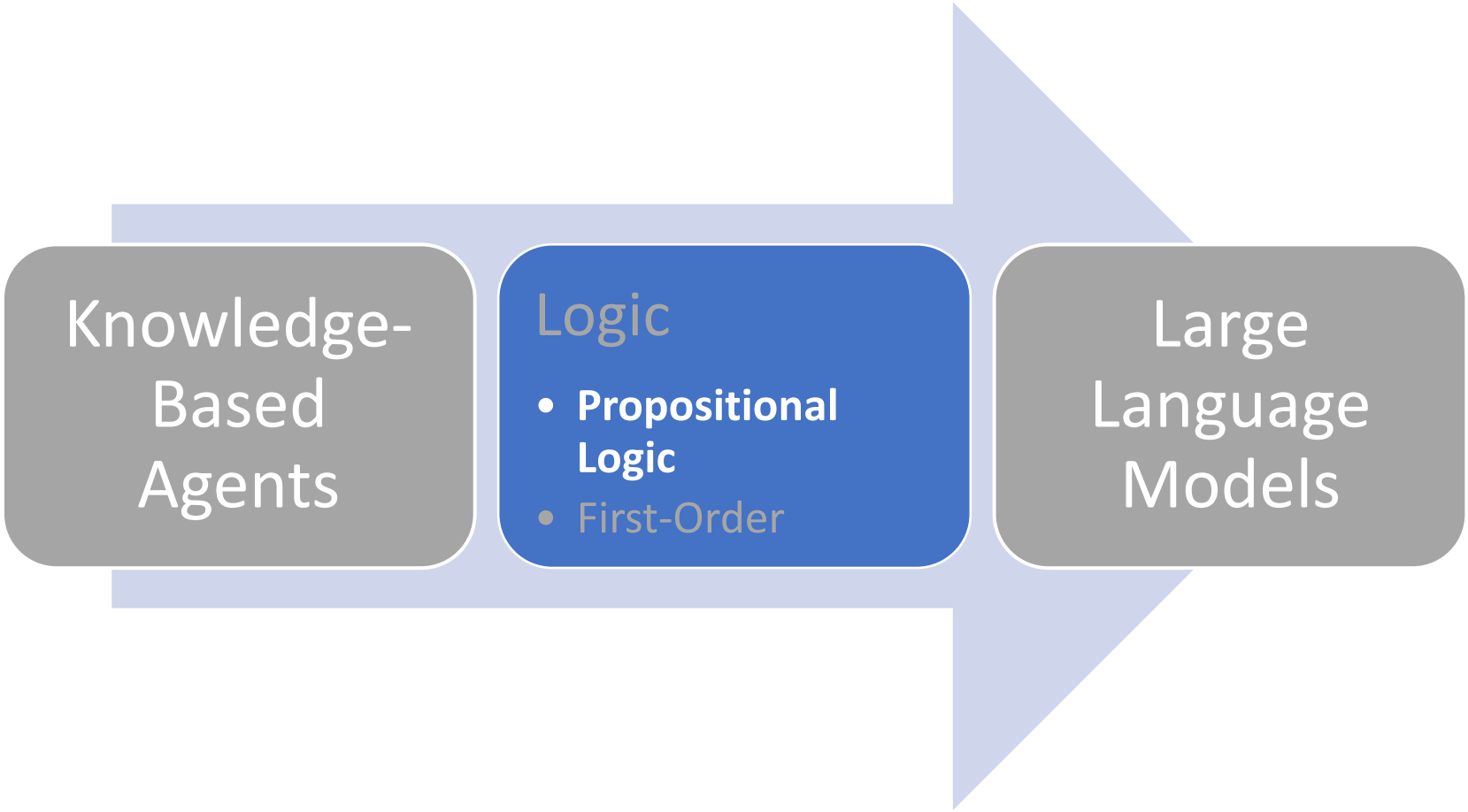
E.g., x + 2 ≥ y is a valid arithmetic sentence, ≥x2y + is not

Semantics: "meaning" of sentences, or relationship between logical sentences and the real world

Specifically, semantics defines truth of sentences

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

Outline


Knowledge-Based Agents
Logic
• Propositional Logic
• First-Order
Large Language Models

Propositional logic: Syntax in BN-Form

$$
\begin{aligned}
\mathit{Sentence} &\rightarrow \mathit{AtomicSentence} \mid \mathit{ComplexSentence} \\
\mathit{AtomicSentence} &\rightarrow \mathit{True} \mid \mathit{False} \mid P \mid Q \mid R \mid \ldots \\
\mathit{ComplexSentence} &\rightarrow (\ \mathit{Sentence}\ ) \\
&\mid \neg\ \mathit{Sentence} \\
&\mid \mathit{Sentence} \wedge \mathit{Sentence} \\
&\mid \mathit{Sentence} \vee \mathit{Sentence} \\
&\mid \mathit{Sentence} \Rightarrow \mathit{Sentence} \\
&\mid \mathit{Sentence} \Leftrightarrow \mathit{Sentence}
\end{aligned}
$$

= Symbols

Negation
Conjunction
Disjunction
Implication
Biconditional

$$
\text{OPERATOR PRECEDENCE} \quad : \quad \neg, \wedge, \vee, \Rightarrow, \Leftrightarrow
$$

Validity and Satisfiability

A sentence is **valid** if it is true in **all** models (called a tautology)

e.g., *True*, $A \vee \neg A$, $A \Rightarrow A$, $(A \wedge (A \Rightarrow B)) \Rightarrow B$
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 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 assignments of truth values to its atoms. Each row is a model.

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

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

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

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

Entailment

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

$$\boldsymbol{KB} \models \boldsymbol{\alpha}$$

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

Inference

- **Logical inference:** a procedure for generating sentences that follow from a knowledge base KB

- An inference procedure is **sound** if it derives a sentence α iff KB ⊨ α. I.e, it only derives **true sentences**.

- An inference procedure is **complete** if it can derive **all** α for which *KB* ⊨ α.

Inference

- How can we check whether a sentence α is entailed by KB?

- How about we **enumerate all possible models of the KB** (truth assignments of all its symbols), and check that α is true in every model in which KB is true?
  - Is this sound (if an answer is produced, it is correct)?
  - Is this complete (guaranteed to produce the correct answer)?
  - Problem: if KB contains *n* symbols, the truth table will be of size 2^n

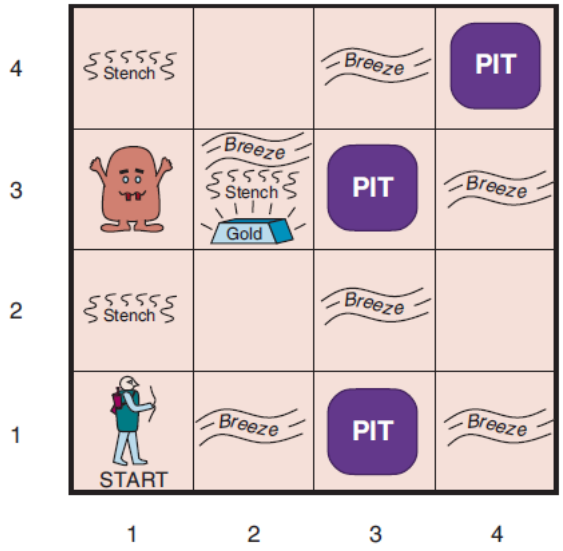
- Better idea: use ***inference rules***, or sound procedures to generate new sentences or *conclusions* given the *premises* in the KB

Complexity of inference

- Propositional inference is ***co-NP-complete***
  - *Complement* of the SAT problem: $\alpha \not\models \beta$ if and only if the sentence $\alpha \wedge \neg \beta$ is *unsatisfiable*
  - Every known inference algorithm has worst-case exponential running time

- Efficient inference 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 OK	2,2	3,2	4,2
1,1 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:

$Breeze(1,1) \Leftrightarrow Pit(1,2) \vee Pit(2,1)$
$Breeze(1,2) \Leftrightarrow Pit(1,1) \vee Pit(1,3) \vee Pit(2,2)$
$\text{Stench}(1,1) \Leftrightarrow W(1,2) \vee W(2,1)$
…

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.

Initial KB needs to contain rules like these for each square:
$Breeze(1,1) \Leftrightarrow Pit(1,2) \vee Pit(2,1)$
$Breeze(1,2) \Leftrightarrow Pit(1,1) \vee Pit(1,3) \vee Pit(2,2)$
$\text{Stench}(1,1) \Leftrightarrow W(1,2) \vee W(2,1)$
…

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.

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).

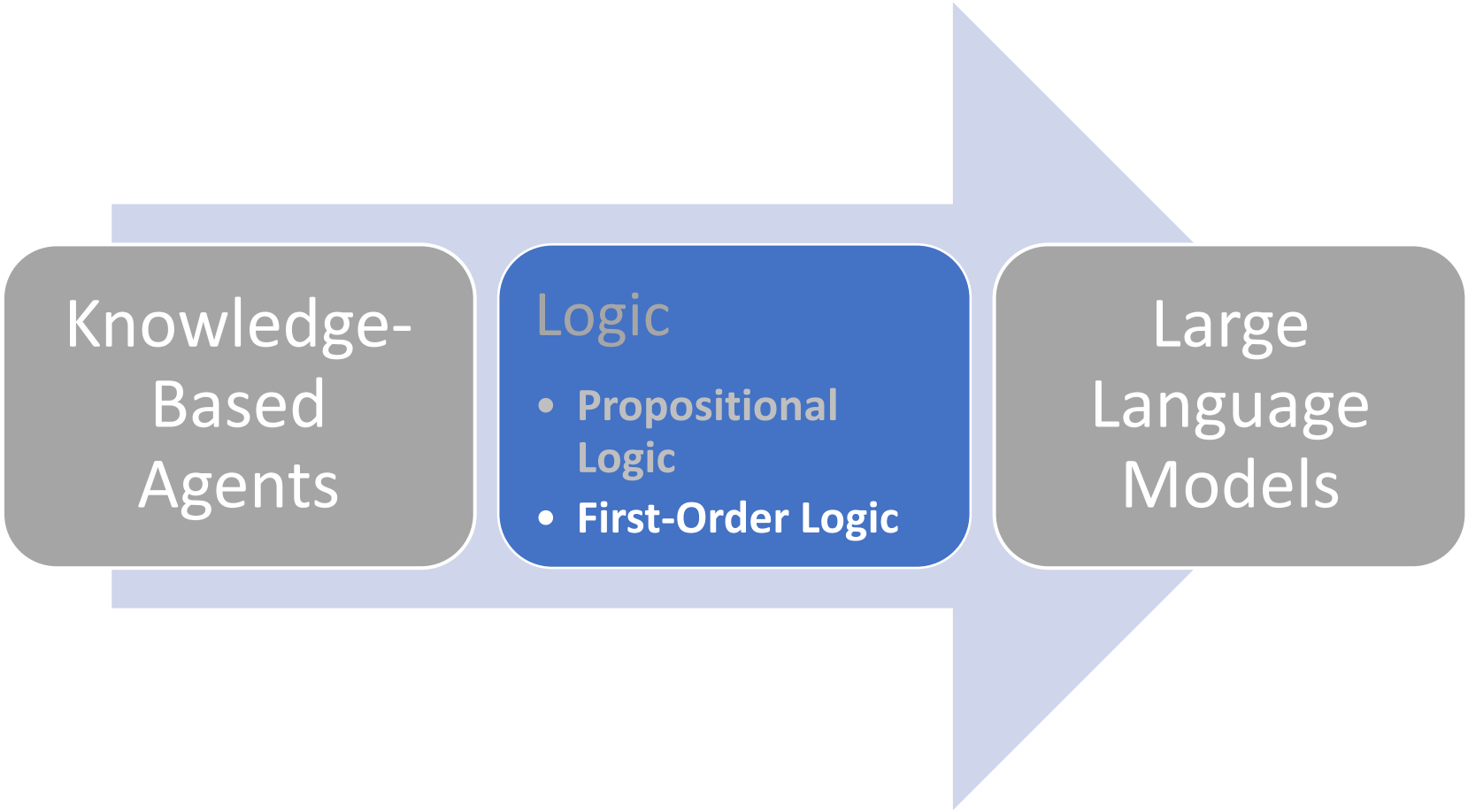
Limitations of propositional logic

- Suppose you want to say “All humans are mortal”
  - In propositional logic, you would need ~6.7 billion statements of the form:
    MichaelIsHuman and MichaelIsMortal,
    SarahIsHuman and SarahIsMortal, …
- Suppose you want to say “Some people can run a marathon”
  - You would need a disjunction of ~6.7 billion statements:

    MichaelcanRunAMarathon or … or SarahCanRunAMarathon

First-order logic can help with that by adding objects and relations to the facts represented by propositional logic.

Outline

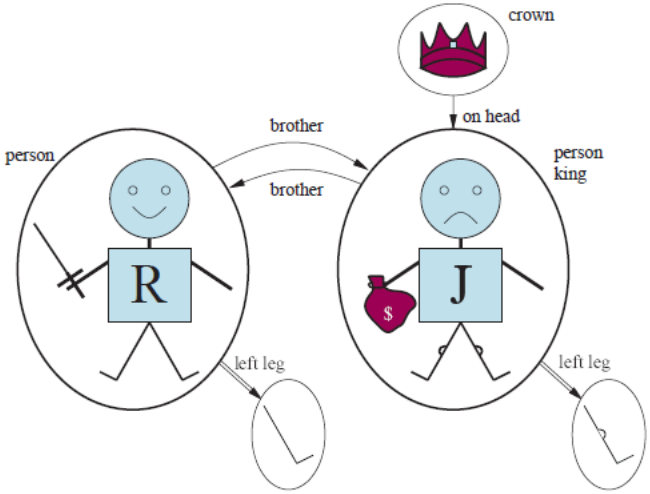

Knowledge-Based Agents
Logic
• Propositional Logic
• First-Order Logic
Large Language Models

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

First-order Logic adds **objects** and **relations.**

Syntax of FOL



$$
\begin{aligned}
\textit{Sentence} &\rightarrow \textit{AtomicSentence} \mid \textit{ComplexSentence} \\
\textit{AtomicSentence} &\rightarrow \textit{Predicate} \mid \textit{Predicate}(\textit{Term}, \ldots) \mid \textit{Term} = \textit{Term} \\
\textit{ComplexSentence} &\rightarrow (\ \textit{Sentence}\ ) \\
&\mid \neg\ \textit{Sentence} \\
&\mid \textit{Sentence} \wedge \textit{Sentence} \\
&\mid \textit{Sentence} \vee \textit{Sentence} \\
&\mid \textit{Sentence} \Rightarrow \textit{Sentence} \\
&\mid \textit{Sentence} \Leftrightarrow \textit{Sentence} \\
&\mid \textit{Quantifier}\ \textit{Variable}, \ldots\ \textit{Sentence} \\
\textit{Term} &\rightarrow \textit{Function}(\textit{Term}, \ldots) \\
&\mid \textit{Constant} \\
&\mid \textit{Variable} \\
\textit{Quantifier} &\rightarrow \forall \mid \exists \\
\textit{Constant} &\rightarrow A \mid X_1 \mid \textit{John} \mid \cdots \\
\textit{Variable} &\rightarrow a \mid x \mid s \mid \cdots \\
\textit{Predicate} &\rightarrow \textit{True} \mid \textit{False} \mid \textit{After} \mid \textit{Loves} \mid \textit{Raining} \mid \cdots \\
\textit{Function} &\rightarrow \textit{Mother} \mid \textit{LeftLeg} \mid \cdots \\
\textsc{Operator Precedence} &: \neg, =, \wedge, \vee, \Rightarrow, \Leftrightarrow
\end{aligned}
$$

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 At(x,SMU)** $\Rightarrow$ **Smart(x)**
  Why not $\forall$**x At(x,SMU)** $\wedge$ **Smart(x)**?
- Roughly speaking, equivalent to the conjunction of all possible instantiations of the variable:
  [At(John, SMU) $\Rightarrow$ **Smart(John)]** $\wedge$ **...**
  [At(Richard, SMU) $\Rightarrow$ **Smart(Richard)]** $\wedge$ **...**
- $\forall$**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

- **$\exists$x P(x)**

- Example: "Someone at SMU is smart"
  $\exists$x At(x,SMU) $\wedge$ Smart(x)
  Why not **$\exists$x At(x,SMU) $\Rightarrow$ Smart(x)**?

- Roughly speaking, equivalent to the disjunction of all possible instantiations:
  [At(John,SMU) $\wedge$ Smart(John)] $\vee$
  [At(Richard,SMU) $\wedge$ Smart(Richard)] $\vee$ **…**

- **$\exists$x P(x)** is true in a model m iff **P(x)** is true with **x** being some possible object in the model

Inference in FOL

- Inference is complicated.

1. **Reduction to propositional logic** and then use propositional logic inference.

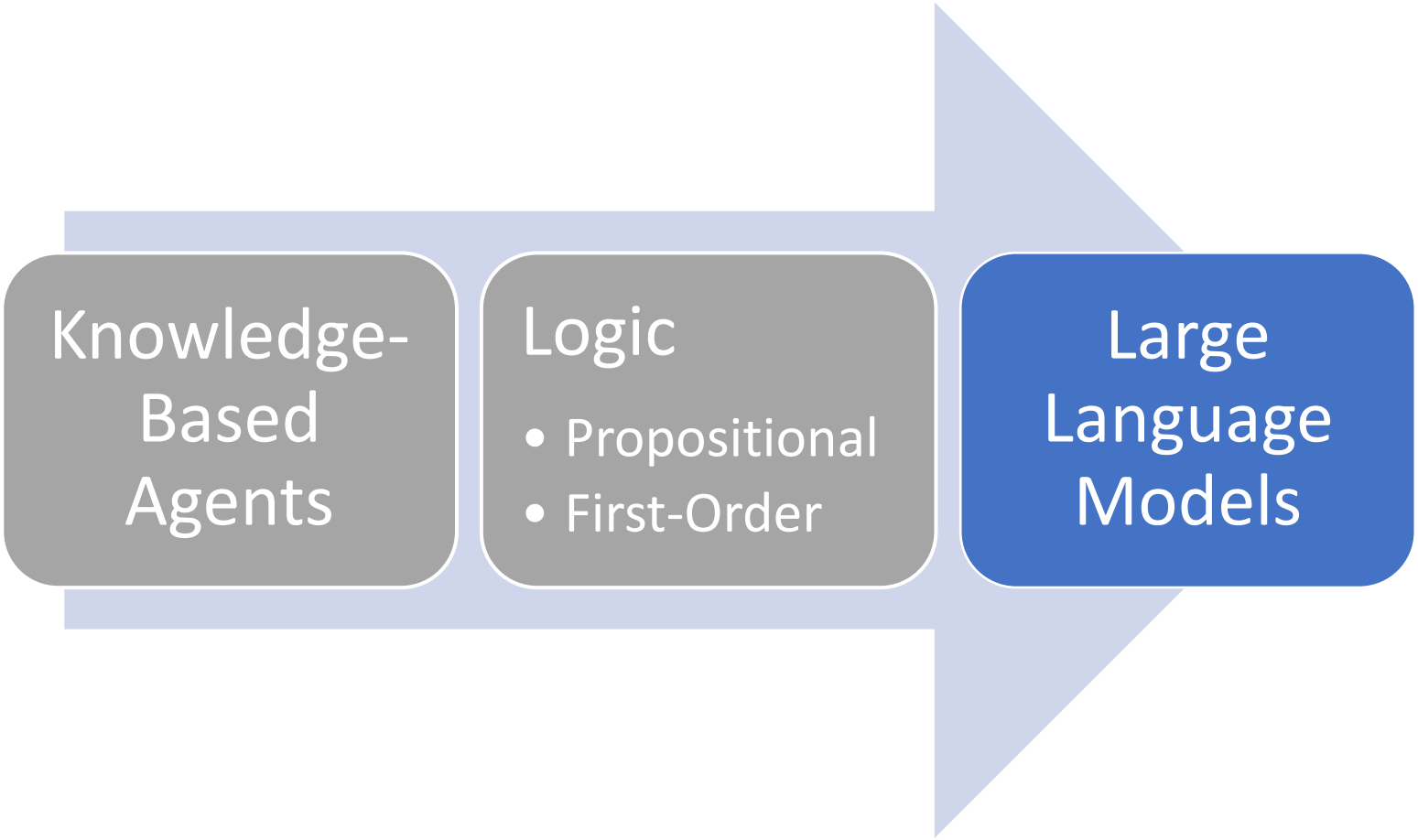
2. **Directly do inference on FOL (or a subset like definite clauses)**
   - Unification: Combine two sentences into one.
   - Forward Chaining for FOL
   - Backward Chaining for FOL
   - Logical programming (e.g., Prolog)

Limitations of Logic

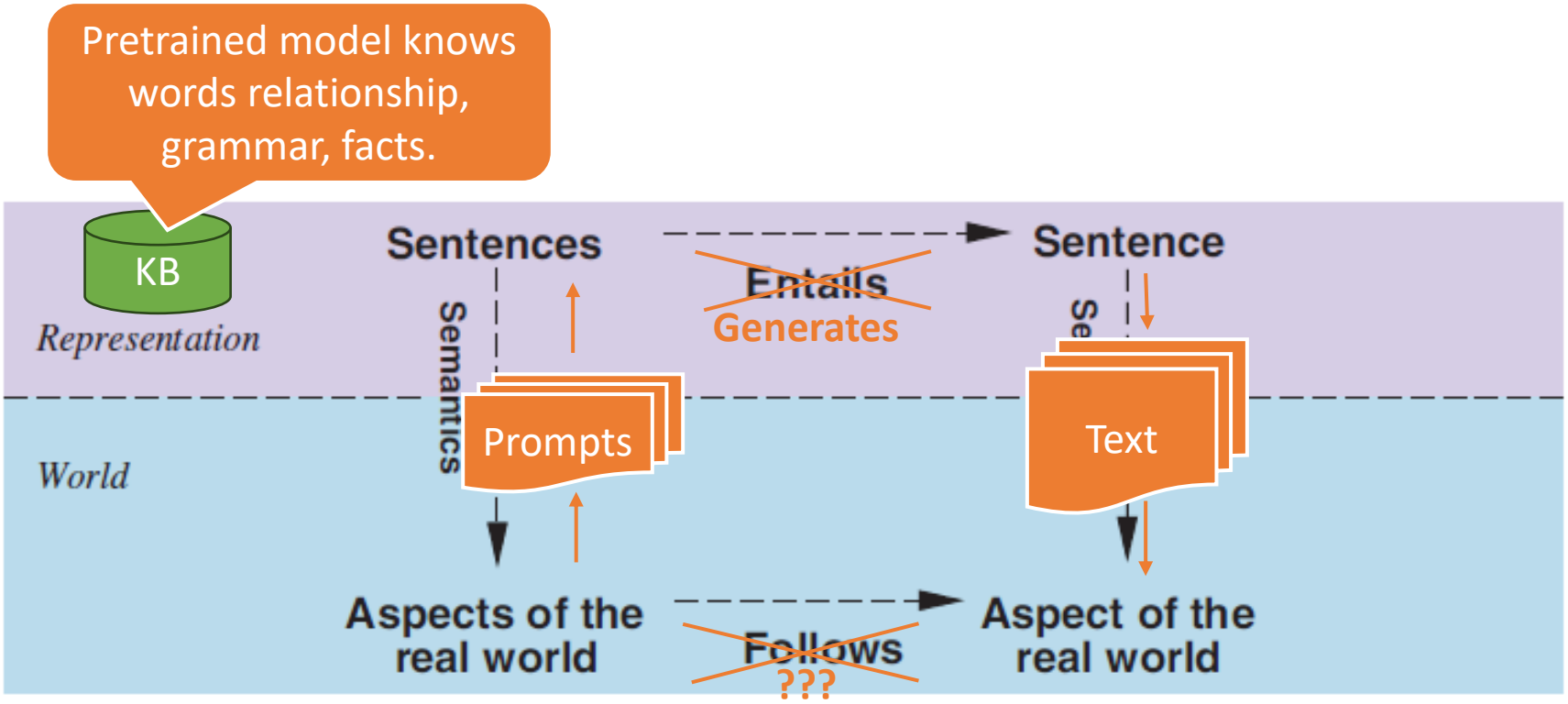
- What if we cannot collect enough knowledge to make a decision
  e.g., the environment if only partially observable?
- What about randomness?
- What about facts about the future?
  grade(Michael, AI, this_semester)

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

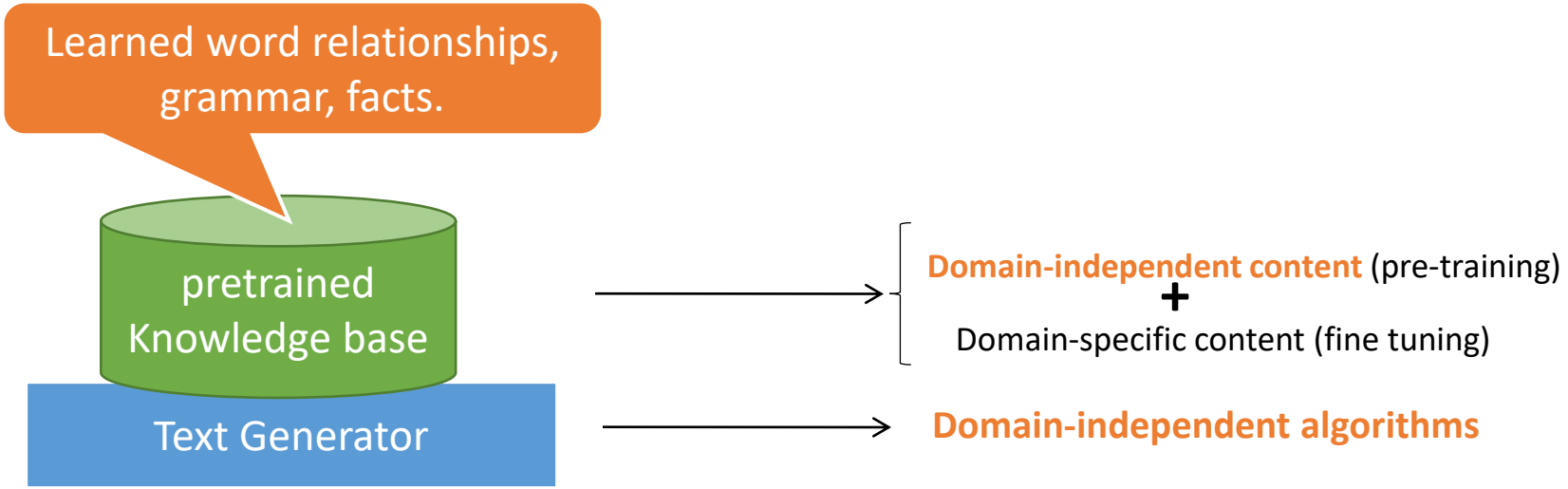

Knowledge-Based Agents
Logic
• Propositional
• First-Order
Large Language Models

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?

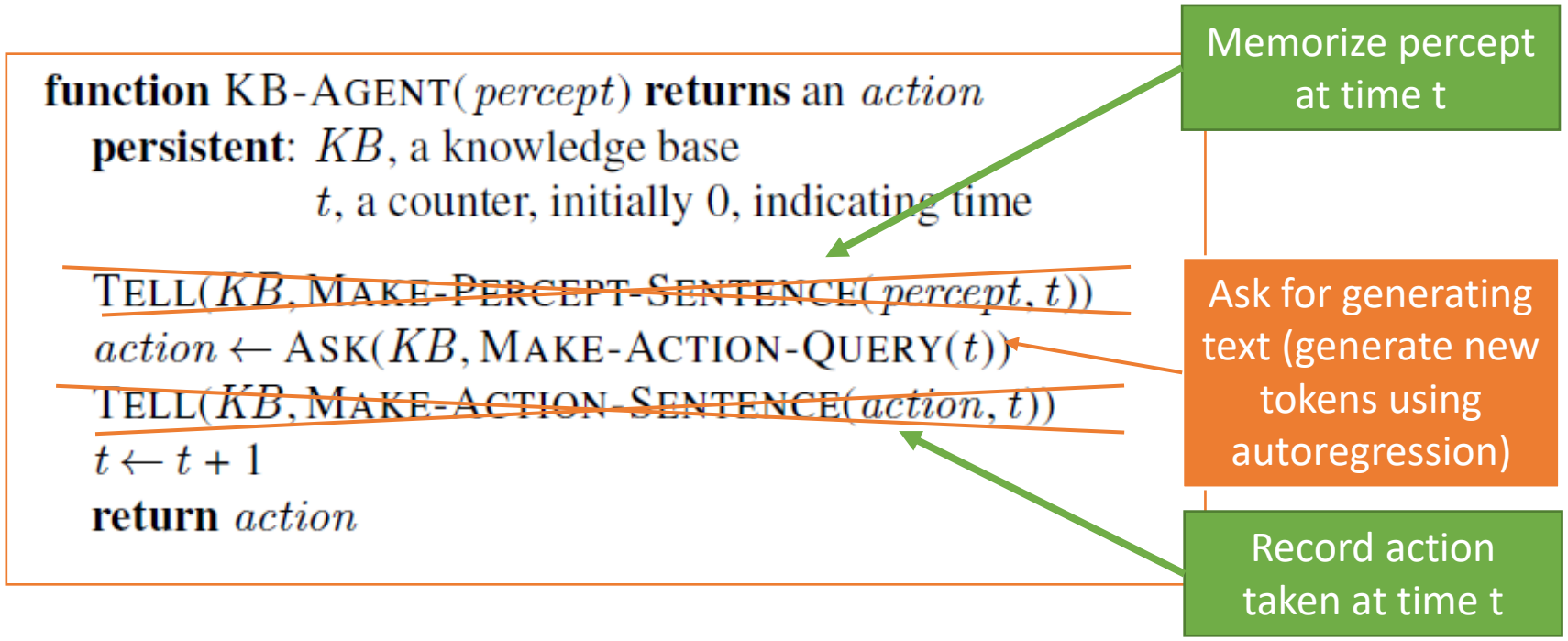
LLM as a Knowledge-Based Agents



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.

Generic Knowledge-based Agent



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.