Knowledge Representation Part 1

AI 109

Richard Kelley

All men naturally desire knowledge.

Aristotle, *Metaphysics*

What is Knowledge?

- What does it mean to *know* something? Someone?
- If you memorize a fact, do you truly know it?

Aristotle’s Approach

- Aristotle distinguished between “true belief” and *demonstrative knowledge* (ἐπιστήμη, episteme).
  - Reciting a truth didn’t count as knowledge.
  - Having demonstrative knowledge of a thing is knowing its *causes* – what makes it to be what it is.
  - So, knowledge for Aristotle involved *explanation*.
- The kind of “explanation” that Aristotle accepted is called a *demonstration*.

Example

- Argument:

“No human observed has a tail; therefore humans lack tails.”

- Is this a good argument?
  - Is it convincing?
- Alternatively:
  - All bipedal animals with upright posture lack tails, because a tail would not serve the functional balance appropriate to upright locomotion.
  - All humans are bipedal rational animals with upright posture.
  - **Therefore:** All humans lack tails.
- Is this better?

Demonstration

- A ***demonstration*** is a *logical argument* (Aristotle used the term *syllogism*) whose ***premises*** are true, primary, immediate, prior to, and better known than the ***conclusion***.
- The ***conclusion*** of an argument is the thing you want to prove.
- The ***premises*** are reasons given for accepting the conclusion.
- The process of moving from premises to a conclusion is called ***inference***.
- Aristotle's goal was to organize all knowledge according to logical arguments that explain things in terms of basic definitions and principles.
  - Look! Another example of a graph.

Knowledge Representation in AI

- ***Knowledge Representation*** (KR) studies
  - how to formally encode information about the world,
  - so that a machine can “reason” with it.
- Central question: What must be represented so that intelligent behavior becomes computationally possible?

Components of a KR System

- Representation
  - We need some kind of machine-like language for representing knowledge.
- Inference
  - We need some kind of system to enable the machine to “reason” from premises to conclusions.
  - Is this really necessary? Why do we need it?

Data vs Information vs Knowledge

- ***Data***
  - Raw symbols or measurements without interpretation (e.g., “42”, “Alice”, “3.7”).
- ***Information***
  - Data interpreted within a structure or ***schema*** (e.g., “Alice’s GPA is 3.7”).
- ***Knowledge***
  - Structured information organized so that new conclusions can be derived (e.g., “Students with GPA > 3.5 graduate with honors; therefore Alice graduates with honors”).
  - Knowledge implies the capacity for performing inference.

Computers Need Formal Language

- ***Natural language*** is expressive but ambiguous and context-dependent.
- A ***formal language*** removes ambiguity by specifying exact ***formation rules***.
- Meaning is defined mathematically, not by intuition.
- This lets us ask precise questions: What follows from what?

Propositional Logic

- ***Atomic propositions*** represent basic facts.
  - Example: Socrates is a man.
  - Can be either ***true*** or ***false***.
    - This rules out subjective statements (in this system).
  - We represent atomic propositions using ***symbols***: P, Q, R, …
  - We can also represent them using more informative names: is_hungry.
- We build logical ***formulas*** by combining atomic propositions using ***connectives*** and ***formation rules***.
  - Connectives: ¬ (not), ∧ (and), ∨ (or), → (implies).
  - Formation rules: If P and Q are formulas, so is (P ∧ Q)

Examples

- Example
  - P: “Alice is a student.”
  - Q: “Alice has GPA > 3.5.”
  - P $\wedge$ Q: “Alice is a student and has GPA > 3.5.”
- Example
  - P: “It is daytime.”
  - Q: “Eve is driving her car.”
  - P $\rightarrow$ Q: “If it is daytime, Eve is driving her car.”

Syntax

- We can follow rules to build up complicated formulas without knowing what they mean.
  - This is what allows computers to "do logic."
- The rules we follow to build up formulas from parts are called the ***syntax*** of our formal language.
- From P → Q and Q → R, we can form more complicated expressions like ((((P → Q) ∧ (Q → R)) ∧ P) → R).

Truth Values and Semantics

- Each atomic proposition is assigned a ***truth value***: True or False.
- The truth of compound formulas is determined ***compositionally***.
  - If you know the truth values of each part, you know the truth value of the whole.
- $P \wedge Q$ is true ***if and only if*** both P and Q are true.
- Semantics = mapping from formulas to truth values. What a formula “means.”
- Some statements are true no matter how you interpret them. These are called ***tautologies***.
  - $((((P \rightarrow Q) \wedge (Q \rightarrow R)) \wedge P) \rightarrow R)$ is a tautology. Why?

Syntax vs Semantics

- Syntax
  - formal structure of expressions (symbols, formation rules).
  - Like grammar.
- Semantics
  - what those expressions mean.
- Syntax and semantics are different, but work together
  - **Inference is correct when syntactic derivations preserve semantic truth.**
- KR requires an explicit semantic foundation.

Inference Rules

- ***Inference rules*** tell us how to reach conclusions from premises.
- Example: ***Modus Ponens***
  - Given: $P \rightarrow Q$ and P
  - Conclude: Q

Limitations of Propositional Logic

- Propositional logic treats statements as indivisible atoms.
- No internal structure: cannot express “All students have GPA > 3.5.”
- Cannot quantify over individuals or describe relations.
- Useful for reasoning about fixed, finite sets of facts.

- The above lead to more sophisticated logical systems.

KR as Modeling

- Representation is an abstraction: it omits detail to focus on relevant structure.
- Ontology: specification of what kinds of entities and relations exist in the model.
- Ontological commitments determine what can and cannot be expressed.
- Modeling choices constrain possible inferences.
- Good KR balances fidelity to the domain with computational feasibility.

A Computer System for Logic

- Called ***Prolog***.
- Consists of ***facts*** and ***rules***.
  - Rules are written “backwards”
    - if X is human, then X is mortal.
  - commas mean “and.”
- Upper case symbols denote ***variables***.
- We can ***query*** the system to make Prolog perform inference.

```
% --- Facts ---
human(socrates).
human(plato).

% --- Rule ---
mortal(X) :-
    human(X).
```

More Complicated Example

- Paths in Graphs
  - path(X,Y) :- edge(X,Y).
  - path(X,Y) :- edge(X,Z), path(Z,Y).

edge(a,b). edge(a,c). edge(a,d).

edge(b,e). edge(b,f).

edge(c,f). edge(c,g).

edge(d,g). edge(d,h).

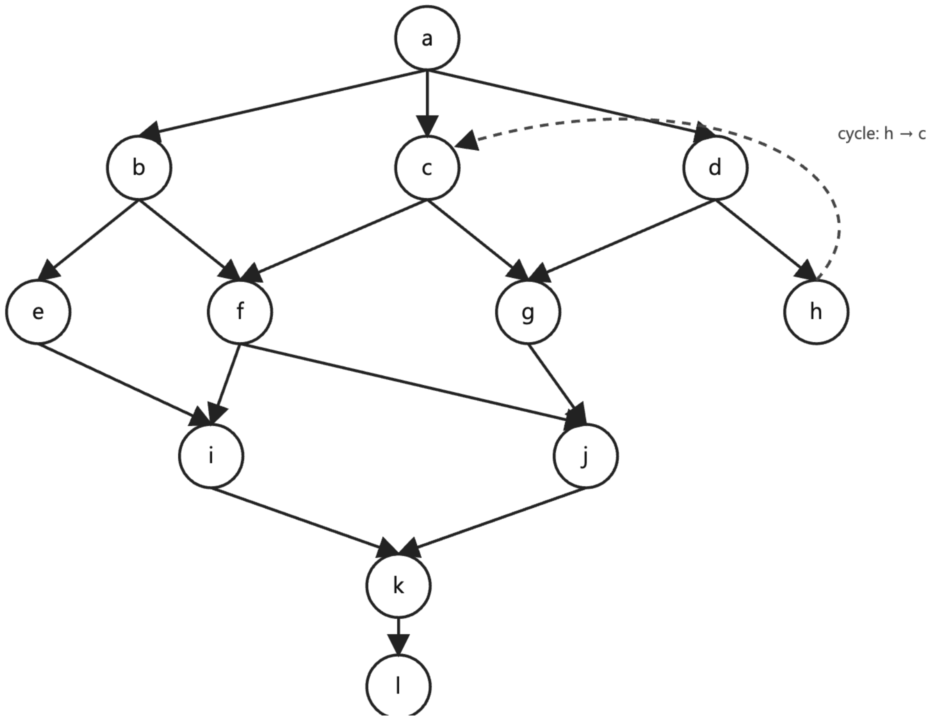
edge(e,i).

edge(f,i). edge(f,j).

edge(g,j). edge(h,c). % cycle

edge(i,k). edge(j,k).

edge(k,l).



Strengths? Weaknesses?

- Strengths
  - Fast
  - Systematic
- Weaknesses
  - Rules and facts are too rigid.
  - Doesn’t handle synonymy, nuance.
  - Doesn’t seem to match human reasoning.