Logical Agents

Burr H. Settles
CS-540, UW-Madison
www.cs.wisc.edu/~cs540-1
Summer 2003

Announcements

- Homework #1 is due today
  - You have up to 3 “late days”
  - Weekends only count as 1 late day
- Read Chapter 8 in AI: A Modern Approach for Monday
- Project proposals are due Monday, too

Review of Agent Architecture

Recap of Agent Properties

- The agent must be able to:
  - Represent states, actions, etc.
  - Incorporate new percepts
  - Update internal representation of world
  - Deduce possibly unobservable properties of world
  - Decide on appropriate actions, etc…
- One of the core issues in developing intelligent agents is that of knowledge:
  - How to represent knowledge
  - How to reason using that knowledge

Knowledge Bases

- A knowledge base is:
  - The domain-specific content for an agent
  - A set of representations of facts about the world
  - A set of sentences in a formal language
- Building a knowledge base:
  - Learning: agent discovers what it knows
  - Telling: agent is given what it knows (declarative)

Knowledge-Based Agents

- Main actions of knowledge-based agents:
  - Tell information to the KB in the form of percept
  - Ask the KB what to do in the form of action
- An inference engine is composed of domain-independent algorithms that are used to determine what follows from the knowledge base
  - Answers should follow from KB… the Agent shouldn’t just make things up!
Knowledge-Based Agents

- Views of a knowledge-based agent:
  - **Knowledge level**: what agent knows at high level
  - **Logic level**: level of sentence encoding
  - **Implementation level**: level that runs on the architecture, detail of data structures and algorithms

General Logic

- Logics are formal languages for representing knowledge from which conclusions can be drawn
- Syntax specifies symbols and how they are combined to form sentences in the language
  - e.g. arithmetic: $2 \times x < y$ is a sentence, $2x < y$ is not
- Semantics specifies what world facts a sentence refers to, and how to assign truth value to sentence
  - e.g. $2 \times x < y$ means:
    - Is true if & only if the number $2 \times x$ is less than the number $y$
    - Is true in a world where $x = 11$, $y = 33$
    - Is false in a world where $x = 3$, $y = 4$

General Logic

- Logics are characterized by what they consider to be "primitives"

<table>
<thead>
<tr>
<th>Logic</th>
<th>Primitives</th>
<th>Available Knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Propositional</td>
<td>facts (propositions)</td>
<td>true/false/unknown</td>
</tr>
<tr>
<td>First-Order</td>
<td>facts, objects, relations</td>
<td>true/false/unknown</td>
</tr>
<tr>
<td>Temporal</td>
<td>facts, objects, relations,</td>
<td>true/false/unknown</td>
</tr>
<tr>
<td></td>
<td>times</td>
<td></td>
</tr>
<tr>
<td>Probability</td>
<td>facts</td>
<td>degree of belief 0…1</td>
</tr>
<tr>
<td>Theory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fuzzy</td>
<td>degree of truth</td>
<td>degree of belief 0…1</td>
</tr>
</tbody>
</table>

General Logic

- Recall that the agent internally represents its world/environment in its knowledge base
- Sentences are representations in some language
- Facts are claims about the world that are true/false

General Logic

- Sentences represent facts in the world
- Semantics connect sentences with facts
- A sentence is true if what it represents is actually the case in the real world

- In human reasoning, we try to take known facts and deduce new facts from them, to arrive at logical conclusions that are also facts
- The agent, however, only knows sentences, which are *representations* of facts… so it must generate new sentences from old ones
- We must be careful that the sentences generated by the agent actually follow from the KB!
General Logic

- Proper reasoning ensures that conclusions inferred from the KB are consistent with reality
  - That is, conclusions represent facts that actually follow from the facts in the KB

Entailment

- $\text{KB} \models \alpha$
  - The knowledge base $\text{KB}$ is said to entail $\alpha$ if and only if $\alpha$ is true in all worlds where $\text{KB}$ is true

- $\alpha$ is true no matter what $\text{KB}$’s interpretation is: meaning is now meaningless!

- For example:
  - $\text{KB}$: “sky is blue,” “grass is green”
  - Entails: “sky is blue and grass is green”

Logical Inference

- An inference procedure can:
  - Generate new sentences $\alpha$ entailed by $\text{KB}$
  - Determine whether or not a given sentence $\alpha$ is entailed by $\text{KB}$ (i.e. “prove” $\alpha$)

- $\text{KB} \models \alpha$
  - Sentence $\alpha$ can be derived from $\text{KB}$ by some inference procedure $\models$

Computers don’t know the semantics (meaning)!

- So we need a mechanical inference procedure that derives conclusion sentences without needing to know the meanings of sentences

Interpretations and Models

- An interpretation is a formally structured world from which a sentence’s truth can be determined
  - Assign truth values to symbols in sentence

- A model for a sentence is any world under a particular interpretation where that sentence is true
  - $m$ is a model of sentence $\alpha$ if $\alpha$ is true in $m$
  - $M(\alpha)$ is the set of all models of $\alpha$
  - Then $\text{KB} \models \alpha$ if and only if $M(\text{KB})$ are also $M(\alpha)$

Logical Inference

- **Soundness**:
  - Inference procedure $\models$ is sound if whenever $\text{KB} \models \alpha$, it is also true that $\text{KB} \models \alpha$

- An inference procedure that derives only entailed sentences is called sound or truth-preserving

- Inference produces only real entailments, i.e. any sentence that follows deductively from true premises is itself true
**Logical Inference**

*Completeness:* Inference procedure 1 is complete if whenever $\text{KB} \models \alpha$, it is also true that $\text{KB} \models_1 \alpha$.

- Inference should produce all entailments, i.e., all true sentences can be derived from the true premises.

**Propositional Logic (PL)**

- A very simple but useful logic
- Syntax of PL:
  - Proposition symbols $P_1, P_2, \ldots$ are sentences
  - If $S$ is a sentence $\neg S$ is a sentence
  - If $S_1$ and $S_2$ are sentences, $S_1 \land S_2$ is a sentence
  - If $S_1$ and $S_2$ are sentences, $S_1 \lor S_2$ is a sentence
  - If $S_1$ and $S_2$ are sentences, $S_1 \implies S_2$ is a sentence
  - If $S_1$ and $S_2$ are sentences, $S_1 \equiv S_2$ is a sentence

**Our Goal**

- We want to define a simple logic that is expressive enough to say anything of interest, but also has a sound and complete inference procedure
- This will allow an agent to answer questions whose answers follow from the KB
- *The fundamental problem of designing logical languages is the tradeoff between their expressiveness (power) and its tractability (efficiency) for knowledge and reasoning*
Inference Proof Methods

- Heuristic search in model space (i.e. complete search)
  - Sound but incomplete
- Inference by enumeration (i.e. using truth tables)
  - Sound and complete for propositional logic
- Application of syntactic operations (i.e. inference rules):
  - Sound generation of new sentences from old
  - Could use inference rules as operators for a search

Inference by Enumeration

- The computer doesn’t know the interpretation for the propositional symbols in the real world
- So all logically distinct cases must be checked to prove that a sentence can be derived from a KB

Inference by Enumeration

- Given n symbols, $2^n$ possible combinations of truth value assignments
- Each combination can be considered an interpretation
- Rows where all of sentences in KB are true are the models of KB

Inference by Enumeration

- A sentence is valid if and only if it is true under all possible interpretations
  - i.e. Its entire column in a truth table is true
- To determine if a sentence $\alpha$ is entailed by KB, all models of KB must also be models of $\alpha$
  - i.e. All rows where KB is true, $\alpha$ is true
- In other words: KB $\Rightarrow$ $\alpha$ is valid

Inference by Enumeration

- Let’s write out truth tables for the following:

\[
\begin{array}{c|c|c|c|c|c}
(P \lor \neg P) & \neg P & (P \lor \neg P) & P \\
T & F & T & T \\
F & T & T & F \\
\end{array}
\]

This is not valid, since not all interpretations of the sentence are true

\[
\begin{array}{c|c|c|c|c|c}
(P \land \neg P) & \neg P & (P \land \neg P) & P \\
T & F & F & T \\
F & T & F & T \\
\end{array}
\]

This is valid, since all interpretations of the sentence are true

- Though sound and complete for PL, the proofs using this technique grow exponentially in length as the number of symbols increases
  - There must be a better way!
- Natural deduction is an inference procedure that uses sound inference rules to derive new sentences from the KB, and any previously derived sentences, until the conclusion sentence is derived
Using Inference Rules

- Implication-Elimination, or Modus Ponens (MP):
  \[ \alpha \rightarrow \beta, \alpha \Rightarrow \beta \]

- Format of Inference Rules:
  - true premise sentence(s)
  - sound conclusion sentence(s)

- Sentences are separated by commas

- Rules can be shown to be sound by using truth table enumeration

Using Inference Rules

- And-Elimination (AE):
  \[ \alpha \land \beta \Rightarrow \gamma \]

- And-Introduction (AI):
  \[ \alpha \land \beta \Rightarrow \gamma \]

- Or-Introduction (OI):
  \[ \alpha \lor \beta \Rightarrow \gamma \]

- Double-Negation Elimination (DNN):
  \[ \neg \neg \neg \neg \gamma \Rightarrow \gamma \]

Using Inference Rules

- Unit Resolution (UR):
  \[ \alpha \lor \beta, \neg \beta \Rightarrow \alpha \]

- Resolution (R):
  \[ \alpha \lor \beta, \neg \beta \lor \gamma \Rightarrow \alpha \lor \gamma \]

...or, equivalently

(transitivity of implication):

\[ \neg \alpha \land \beta, \beta \land \gamma \Rightarrow \neg \alpha \lor \gamma \]

Using Inference Rules

- These inference rules can be applied in sequence to derive some sentence \( \alpha \) from the KB, thus showing that \( \text{KB} \models \alpha \)

- The ability of inference rules to be used for proofs in PL relies on the monotonicity property
  - The number of entailed sentences can only increase as information is added to the KB
  - If \( \text{KB} \models \alpha \) then \( \text{KB} \models \beta \Rightarrow \alpha \)

- Non-monotonic logics capture a common property of human reasoning: changing your mind!

Using Inference Rules

- We want to prove that \text{shouldReadCh8} follows logically from these statements in our KB:

1. isCStudent \( \Rightarrow \neg \text{isIlliterate} \)
2. isIlliterate \( \lor \text{canRead} \Rightarrow \text{shouldReadCh8} \)
3. canRead \( \land \text{takingAI} \Rightarrow \text{shouldReadCh8} \)
4. isCStudent
5. takingAI
6. \( \neg \text{isIlliterate} \) (Modus Ponens: 1, 4)
7. canRead (Unit Resolution: 2, 6)
8. canRead \( \land \text{takingAI} \) (And-Introduction: 5, 7)
9. \text{shouldReadCh8} (Modus Ponens: 3, 8)

Proofs Using Inference Rules
Summary

- Logical agents apply inference to a knowledge base to derive new information and make decisions
- Base concepts:
  - Syntax: formal structure of sentences
  - Semantics: truth of sentences based on interpretations
  - Entailment: necessary truth of one sentence given another
  - Inference: deriving conclusions from premises (sentences)
  - Soundness: deriving true conclusions from true premises
  - Completeness: deriving all true conclusions from a set of true premises

Summary

- Propositional logic (PL) commits only to the existence of facts having a true/false state
- Truth-table enumeration method is a sound and complete proving method for PL
  - Time complexity: $O(2^n)$ where $n$ is the # of symbols
  - In practice: only a subset of KB is needed for proof
- Natural deduction through the application of inference rules is a sound proving method for PL
  - Could design as search with inference rules being operators

Summary

- We have to enumerate all possible facts with PL, and cannot identify specific individuals, or generalize using variables
- PL Can’t directly express properties of individuals, objects or relations between them
  - sword(Excalibur)
  - weapon(Arthur,Excalibur)

- Next time we’ll discuss first-order logic (FOL), a more expressive logic language that can remedy some of these problems