Logical Agents

(AIMA - Chapter 7)

Outline

1. Knowledge-based agents
2. Wumpus world
3. An introduction to logic
   - Inference, validity, equivalence and satisfiability
4. Propositional logic

Next Time:
- Automated Propositional Theorem Provers
  - Resolution
  - A Practical Method: Walksat

1. Knowledge-based Agents

Logical Agents

- Most useful in non-episodic, partially observable environments
- Logic (Knowledge-Based) agents combine
  1. A knowledge base (KB): a list of facts that are known to the agent.
  2. Current percepts to infer hidden aspects of the current state using Rules of inference
- Logic provides a good formal language for both
  - Facts encoded as axioms
  - Rules of inference

The Knowledge Base

A set of sentences
- that encodes assertions about the world
- in a formal knowledge representation language.

Declarative approach to building an agent:
- Tell it what it needs to know – add fact to KB.
- Ask it what to do – compute using inference rules also in KB

Generic KB-Based Agent Pseudocode

function KB-Agent(percept) returns an action
  static: KB, a knowledge base
  t, a counter, initially 0, indicating time
  TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
  actions ← ANN(KB, MAKE-ACTION-QUERY(t))
  TELL(KB, MAKE-ACTION-SENTENCE(action, t))
  t ← t + 1
  return action

- Agent must be able to:
  1. Represent states and actions,
  2. Incorporate new percepts
  3. Update internal representation of the world
  4. Deduce hidden properties of the world
  5. Deduce appropriate actions – requires some logic of action
Running Example: The Wumpus World

A Hunt the Wumpus Flash version: http://www.flashrolls.com/puzzle-games/Hunt-The-Wumpus-Flash-Game.htm

Our PEAS description

- Performance measure
  - gold: +1000, death: -1000
  - -1 per step
- Environment
  - Squares adjacent to wumpus are smelly
  - Squares adjacent to pit are breezy
  - Glitter iff gold is in the same square
  - Gold is picked up by reflex, can’t be dropped
  - You bump if you walk into a wall
- Actuators: Move <dir>
- Sensors: Stench, Breeze, Glitter, Bump

Full PEAS description

- Performance measure
  - gold: +1000, death: -1000
  - -1 per step, -10 for using the arrow
- Environment
  - Squares adjacent to wumpus are smelly
  - Squares adjacent to pit are breezy
  - Glitter iff gold is in the same square
  - Shooting kills wumpus if you are facing it. It screams
  - Shooting uses up the only arrow
  - Grabbing picks up gold if in same square
  - Releasing drops the gold in same square
  - You bump if you walk into a wall
- Actuators: Face <direction>, Move, Grab, Release, Shoot
- Sensors: Stench, Breeze, Glitter, Bump, Scream

Wumpus world characterization

- Deterministic: Yes — outcomes exactly specified
- Static: Yes — Wumpus and Pits do not move
- Discrete: Yes
- Single-agent: Yes — Wumpus is essentially a natural feature
- Fully Observable: No — only local perception
- Episodic: No—What was observed before (breezes, pits, etc) is very useful.

Exploring the Wumpus World

1. The KB initially contains the rules of the environment.
2. Location: [1,1]
   - Percept: [¬Stench, ¬Breeze, ¬Glitter, ¬Bump]
   - Action: Move to safe cell e.g. 2,1
3. Location: [2,1]
   - Percept: [¬Stench, Breeze, ¬Glitter, ¬Bump]
   - Infer: Breeze indicates that there is a pit in [2,2] or [3,1]
4. Location: [1,2] (after going through [1,1])
   - Percept: [Stench, ¬Breeze, ¬Glitter, ¬Bump]
   - Infer: Wumpus is in [1,1] or [2,2] or [1,3]
   - Infer: ... stench not detected in [2,1], thus not in [2,2]
   - Remember: Wumpus not in [1,1]
   - Thus: ... Wumpus is in [1,3]
   - Therefore: Pit in [2,2] or [3,1]
   - Action: Move to [2,2]
   - Remember: Pit in [2,2] or [3,1]
   - Therefore: Pit in [3,1]!
LOGIC: What is a logic?

- A formal language with associated
  - Syntax — what expressions are legal (well-formed)
  - Semantics — what legal expressions mean
    - in logic, meaning is defined by the truth of each sentence with respect to a set of possible worlds
      - Possible world (simplified): an assignment of values to all logical variables
- e.g. the language of arithmetic
  - Syntax: \( X + 2 \geq y \) is a legal sentence, \( x^2 + y \) is not a legal sentence
  - Semantics: \( X + 2 \geq y \) is true in a world where \( x=7 \) and \( y=1 \)
  - Semantics: \( X + 2 \geq y \) is false in a world where \( x=0 \) and \( y=6 \)

4. Propositional Logic

- Propositional logic is the simplest logic — illustrates basic ideas
- Inference in propositional logic is also tractable with reasonable constraints — therefore very useful

Propositional logic: Syntax

Recursively defined:
  - Base case:
    - The atomic proposition symbols \( P_1, P_2 \) etc are sentences
  - Recursion:
    - If \( S \) is a sentence, \( \neg S \) is a sentence (negation)
    - If \( S_1 \) and \( S_2 \) are sentences, \( S_1 \land S_2 \) is a sentence (conjunction)
    - If \( S_1 \) and \( S_2 \) are sentences, \( S_1 \lor S_2 \) is a sentence (disjunction)
    - If \( S_1 \) and \( S_2 \) are sentences, \( S_1 \rightarrow S_2 \) is a sentence (implication)
    - If \( S_1 \) and \( S_2 \) are sentences, \( S_1 \leftrightarrow S_2 \) is a sentence (biconditional)

Propositional logic: Semantics

Each model/world specifies true or false for each proposition symbol

E.g., \( P_1, P_2, P_3 \)

true false false

With these symbols, 8 possible models (worlds) can be enumerated automatically.

Rules for evaluating truth with respect to a model \( m \):

- \( \neg S \) is true iff \( S \) is false
- \( S_1 \land S_2 \) is true iff \( S_1 \) is true and \( S_2 \) is true
- \( S_1 \lor S_2 \) is true iff \( S_1 \) is true or \( S_2 \) is true
- \( S_1 \rightarrow S_2 \) is true iff \( S_1 \) is false or \( S_2 \) is true
  - i.e., is false iff \( S_1 \) is true and \( S_2 \) is false
- \( S_1 \leftrightarrow S_2 \) is true iff \( S_1 \) and \( S_2 \) are true

Simple recursive process evaluates an arbitrary sentence, e.g.,

\[
\neg P_1 \land (P_1 \lor P_2) = \text{true} \quad (\text{true} \lor \text{false}) = \text{true} \quad \text{true} = \text{true}
\]

Wumpus world sentences

Let \( P_{i,j} \) be true if there is a pit in \([i, j]\).
Let \( B_{i,j} \) be true if there is a breeze in \([i, j]\).

start:

\[
\neg P_{1,1} \\
\neg B_{1,1} \\

"Pits cause breezes in adjacent squares"
\]

\[
\begin{align*}
B_{1,2} & \equiv (P_{1,2} \lor P_{2,2}) \\
B_{2,2} & \equiv (P_{1,2} \lor P_{2,2} \lor P_{3,2})
\end{align*}
\]

Models

- Logicians typically think in terms of models
  - Formally structured “worlds” with respect to which truth can be evaluated
  - We say \( m \) is a model of a sentence \( \alpha \) if \( \alpha \) is true in \( m \)
  - \( M(\alpha) \) is the set of all models of \( \alpha \)
A key semantic relation: **Entailment**

- **Entailment** means that the truth of one sentence follows from the truth of another:
  \[ KB \vDash \alpha \]

Knowledge base \( KB \) entails sentence \( \alpha \) if and only if

- \( \alpha \) is true in all worlds where \( KB \) is true
  - e.g., the KB containing *the Giants won and the Reds lost* entails *The Giants won*
  - e.g., the KB containing \( x+y = 4 \) entails \( 4 = x+y \)

- **Entailment is a relationship between sentences based on semantics**

---

**Models II**

- **Review:**
  - \( m \) is a model of a sentence \( \alpha \) if \( \alpha \) is true in \( m \)
  - \( M(\alpha) \) is the set of all models of \( \alpha \)

- **Entailment:**
  - \( KB \vDash \alpha \) iff \( M(KB) \subseteq M(\alpha) \)
  - **Example:**
    - \( KB = \) Giants won and Reds lost
    - \( \alpha = \) Giants won
    - \( M(\alpha) \)

---

**Entailment in the wumpus world**

- Consider possible models for \( KB \) assuming only pits and a reduced Wumpus world with only 5 squares and pits

- Situation after
  - A. detecting nothing in \([1,1]\)
  - B. moving right, breeze in \([2,1]\):

  ![Wumpus World Diagram](image)

- **Wumpus models I**

  ![Wumpus Models](image)

- **Wumpus models II**

  ![Wumpus Models](image)

- **Deciding what to do by model checking I**

  \[ \alpha_1 = \text{"[1,2] is safe"}, \ KB \vDash \alpha_1, \text{proved by model checking} \]
Deciding what to do by model checking II

α₂ = "[2,2] is safe", KB ∤ α₂

Truth tables for connectives

- Truth tables enumerate all possible propositional models
- Thus, truth tables are a simple form of model checking

<table>
<thead>
<tr>
<th>P</th>
<th>Q</th>
<th>¬P</th>
<th>P ∧ Q</th>
<th>P ∨ Q</th>
<th>P ⊃ Q</th>
<th>P ⇔ Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>false</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
</tbody>
</table>

OR: P or Q is true or both are true.

XOR: P or Q is true but not both.

Implication is always true when the premises are False!

Inference Procedures

- KB ⊢ a: sentence a can be derived from KB by procedure i
- Soundness: i is sound if whenever KB ⊢ a, it is also true that KB ⊭ a
  - No false inferences
  - (but all true statements might not be derived)
- Completeness: i is complete if whenever KB ⊬ a, it is also true that KB ⊢ a
  - All true sentences can be derived,
  - (but some false statements might be derived)
- Desirable: sound and complete

Inference by enumeration

- Enumeration of all models (truth tables) is sound and complete.
- For n symbols, time complexity is O(2ⁿ)...
- We need a smarter way to do inference!
- One approach: infer new logical sentences from the data-base and see if they match a query.

Logical equivalence

- To manipulate logical sentences we need some rewrite rules.
- Two sentences are logically equivalent iff they are true in same models: α ≡ β iff α ⊨ β and β ⊨ α

Validity and satisfiability

A sentence is valid if it is true in all models, e.g. True, A ∨ ¬A, A → (A ∧ (A ⊃ B)) ⊃ B

Validity is connected to inference via the Deduction Theorem:

KB ⊢ α if and only if (KB ⊢ α) is valid

A sentence is satisfiable if it is true in some model e.g. A ¬A

A sentence is unsatisfiable if it is false in all models e.g. A ⊃ A

Satisfiability is connected to inference via the following:

KB ⊬ α if and only if (KB ⊢ ¬α) is unsatisfiable

(there is no model for which KB=true and α is false)