Logical Agents

Russell and Norvig, chapter 7
Knowledge based agents

Agents need to be able to:

- Store information about their environment
- Update and reason about that information
Knowledge based agents

Agents need to be able to:

- Store information about their environment
- Update and reason about that information

To achieve that we will introduce:

- A knowledge base (KB): a list of facts that are known to the agent.
- Rules to infer new facts from old facts using rules of inference.
Knowledge based agents

Agents need to be able to:
- Store information about their environment
- Update and reason about that information

To achieve that we will introduce:
- A **knowledge base (KB)**: a list of facts that are known to the agent.
- Rules to infer new facts from old facts using rules of inference.
- Logic provides the natural language for this.
Knowledge Bases

- Knowledge base:
  - set of *sentences* in a *formal* language.

- **Declarative** approach to building an agent:
  - Tell it what it needs to know.
  - Ask it what to do → answers should follow from the KB.
The Wumpus World
The Wumpus World

- **Performance measure**
  - gold: +1000, death: -1000
  - -1 per step, -10 for using the arrow

- **Environment**
  - Squares adjacent to wumpus: smelly
  - Squares adjacent to pit: breezy
  - Glitter iff gold is in the same square
  - Shooting kills wumpus if you are facing it
  - Shooting uses up the only arrow
  - Grabbing picks up gold if in same square

- **Sensors:** Stench, Breeze, Glitter, Bump

- **Actuators:** Left turn, Right turn, Forward, Grab, Release, Shoot
The Wumpus World Environment

- Fully Observable? No, only local perception
- Deterministic? Yes, outcome exactly specified
- Static? Yes, Wumpus and pits do not move
- Discrete? Yes
- Single-agent? Yes
Exploring a wumpus world

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1,4</td>
<td>2,4</td>
<td>3,4</td>
<td>4,4</td>
</tr>
<tr>
<td>1,3</td>
<td>2,3</td>
<td>3,3</td>
<td>4,3</td>
</tr>
<tr>
<td>1,2</td>
<td>2,2</td>
<td>3,2</td>
<td>4,2</td>
</tr>
<tr>
<td>1,1</td>
<td>2,1</td>
<td>3,1</td>
<td>4,1</td>
</tr>
</tbody>
</table>

(a)

Initial state
Exploring a wumpus world

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>G</th>
<th>P</th>
<th>S</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent</td>
<td>Breeze</td>
<td>Glitter, Gold</td>
<td>Pit</td>
<td>Stench</td>
<td>Wumpus</td>
</tr>
<tr>
<td>OK</td>
<td>Safe square</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

After 1 move

Breezes are next to pits.
Which is the pit?
Exploring a wumpus world

After 3 moves

<table>
<thead>
<tr>
<th>1,4</th>
<th>2,4</th>
<th>3,4</th>
<th>4,4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1,3</th>
<th>2,3</th>
<th>3,3</th>
<th>4,3</th>
</tr>
</thead>
<tbody>
<tr>
<td>W!</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1,2</th>
<th>2,2</th>
<th>3,2</th>
<th>4,2</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1,1</th>
<th>2,1</th>
<th>3,1</th>
<th>4,1</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
<td>B</td>
<td>P!</td>
<td></td>
</tr>
</tbody>
</table>

Stench here
Means Wumpus here

**Key:**
- **A** = Agent
- **B** = Breeze
- **G** = Glitter, Gold
- **OK** = Safe square
- **P** = Pit
- **S** = Stench
- **V** = Visited
- **W** = Wumpus
Exploring a wumpus world

After 5 moves

Found gold, Avoided Wumpus
Logic

- We used logical reasoning to find the gold.
- Formal language for representing information such that conclusions can be drawn
- **Syntax** defines the sentences in the language
- **Semantics** define the "meaning" of sentences;
  - i.e., define truth of a sentence in a world
Logic

- **Syntax** defines the sentences in the language
- **Semantics** define the "meaning" of sentences;
  - i.e., define truth of a sentence in a world
- E.g., the language of arithmetic
  - Syntax: \( x+2 \geq y \) is a sentence; \( x^2+y > \emptyset \) is not a sentence
  - Semantics: \( x+2 \geq y \) is true in a world where \( x = 7, \ y = 1 \)
Entailment

- **Entailment** means that one thing follows from another:

  \[ \text{KB} \models \alpha \]

  Knowledge base \( KB \) **entails** sentence \( \alpha \) if and only if \( \alpha \) is true in all worlds where \( KB \) is true

- Example: in a knowledgebase of arithmetic \( x+y = 4 \) entails \( 4 = x+y \)
Models

- **Model**: a formally structured world with respect to which truth can be evaluated
  - Example: a model for $x+y=4$ is an assignment of values to $x$ and $y$
  - It is true in a world where $x$ is 2 and $y$ is 2
Entailment in the wumpus world

- Let’s consider possible models for wumpus KB, restricting our attention to pits, and focusing on a region of the wumpus world.

- Situation after:
  nothing in [1,1], moving right, breeze in [2,1]
Wumpus models

All possible models
Exploring a wumpus world

$\alpha_1 = \text{“no pit in } [1,2]\text{”}$

$KB = \text{all possible wumpus-worlds consistent with the observations and behavior of Wumpus world.}$

$KB \models \alpha_1$ can be proved by model checking
Exploring a wumpus world

$\alpha_2 = \text{"no pit in [2,2]'"}, \ KB \models \alpha_2$
Logical inference

- The notion of entailment can be used for logic inference.
  - Model checking: check all possible models
  - Is this a good inference method?
- $KB \models_i \alpha$ - sentence $\alpha$ can be derived from $KB$ by procedure $i$
- If an algorithm only derives entailed sentences it is called *sound* or *truth preserving*.
  - Otherwise it just makes things up.

  * $i$ is sound if whenever $KB \models_i \alpha$ it is also true that $KB \models \alpha$

- Completeness: the algorithm can derive any sentence that is entailed.

  * $i$ is complete if whenever $KB \models \alpha$ it is also true that $KB \models_i \alpha$
Logic

Needed:

**Representation**: formalism for storing knowledge.

**Reasoning**: mechanism for deriving new knowledge from old: *deduction*
Propositional logic

- Propositional logic is a simple logic based on propositions:
  - Propositions are either *true* or *false*

- Examples of propositions
  - Your textbook is green
  - The sky is falling
Propositional logic

- **Propositions**: statements of fact
  - “It is raining” becomes **RAINING**

- **Connectives**: operators on propositions
  - “If it is raining, then it is not sunny.” becomes **RAINING \( \rightarrow \neg \text{SUNNY} \)**
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 $[i, j]$.

\[
\text{start: } \quad \neg P_{1,1} \\
\neg B_{1,1}
\]

"Pits cause breezes in adjacent squares"

\[
B_{1,1} \iff (P_{1,2} \lor P_{2,1}) \\
B_{2,1} \iff (P_{1,1} \lor P_{2,2} \lor P_{3,1})
\]
Propositional logic: syntax

- The proposition symbols $P_1$, $P_2$ etc are sentences
- 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)
# Truth Values

<table>
<thead>
<tr>
<th>This sentence...</th>
<th>... is true when (iff)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$</td>
<td>$S$ is true</td>
</tr>
<tr>
<td>$\neg S$</td>
<td>$S$ is false</td>
</tr>
<tr>
<td>$(S_1 \wedge S_2)$</td>
<td>$S_1$ is true and $S_2$ is true</td>
</tr>
<tr>
<td>$(S_1 \lor S_2)$</td>
<td>At least one of $S_1$ or $S_2$ is true</td>
</tr>
<tr>
<td>$(S_1 \implies S_2)$</td>
<td>$S_1$ is false or $S_2$ is true</td>
</tr>
<tr>
<td>$(S_1 \iff S_2)$</td>
<td>$S_1$ &amp; $S_2$ have the same truth value</td>
</tr>
</tbody>
</table>
### Truth tables for connectives

**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!**
Evaluating truth value

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

\[ \neg P_{1,2} \land (P_{2,2} \lor P_{3,1}) = true \land (true \lor false) = true \land true = true \]
Question

Does \{A\Rightarrow B, B\} entail \{A\}?

- Written as \{A\Rightarrow B, B\} \vdash \{A\}
- How do you know if this is true?

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>( (A\Rightarrow B)^\land B )</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>
Question

- Does \{A \Rightarrow B, B\} entail \{A\}?
  - Written as \{A \Rightarrow B, B\} \models \{A\}
  - How do you know if this is true?

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>(A \Rightarrow B)^B</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

No, \{A\} is not entailed
Inference by enumeration

- Enumeration of all models is sound and complete.
- For n symbols, time complexity is $O(2^n)$.
- Need a smarter way to do inference!