Knowledge-based agents

Chapter 7

Outline

♦ Knowledge-based agents
♦ Wumpus world
♦ Logic in general—models and entailment
♦ Propositional (Boolean) logic
♦ Equivalence, validity, satisfiability
♦ Inference rules and theorem proving
  – forward chaining
  – backward chaining
  – resolution

Chapter 7 2

Knowledge bases

Knowledge base = set of sentences in a formal language

Declarative approach to building an agent (or other system):
Knowledge base = set of statements in a formal language

Knowledge-based agent

Chapter 7 3

A simple knowledge-based agent

Knowledge bases

Wumpus world characterization

Wumpus world PEAS description

Performance measure

Gold +1000, death -1000

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
Shooting uses up the only arrow
Grabbing picks up gold if in same square
Releasing drops the gold in same square

Actuators

Left turn, Right turn, Forward, Grab, Release, Shoot

Sensors

Breeze, Glitter, Stench

Chapter 7 4

Wumpus world PEAS description

Performance measure

Gold +1000, death -1000

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
Shooting uses up the only arrow
Grabbing picks up gold if in same square
Releasing drops the gold in same square

Actuators

Left turn, Right turn, Forward, Grab, Release, Shoot

Sensors

Breeze, Glitter, Stench

Chapter 7 5

Wumpus world characterization

Observable

Chapter 7 6
Wumpus world characterization

<table>
<thead>
<tr>
<th>Observable</th>
<th>No—only local perception</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deterministic</td>
<td>Yes—outcomes exactly specified</td>
</tr>
<tr>
<td>Episodic</td>
<td>No—sequential at the level of actions</td>
</tr>
<tr>
<td>Static</td>
<td>Yes—Wumpus and Pits do not move</td>
</tr>
<tr>
<td>Discrete</td>
<td>Yes</td>
</tr>
<tr>
<td>Single-agent</td>
<td>Yes—Wumpus is essentially a natural feature</td>
</tr>
</tbody>
</table>

Wumpus world characterization

<table>
<thead>
<tr>
<th>Observable</th>
<th>No—only local perception</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deterministic</td>
<td>Yes—outcomes exactly specified</td>
</tr>
<tr>
<td>Episodic</td>
<td>No—sequential at the level of actions</td>
</tr>
<tr>
<td>Static</td>
<td>Yes—Wumpus and Pits do not move</td>
</tr>
<tr>
<td>Discrete</td>
<td>Yes</td>
</tr>
<tr>
<td>Single-agent</td>
<td>Yes—Wumpus is essentially a natural feature</td>
</tr>
</tbody>
</table>
Exploring a wumpus world
Other Tight Spots

Breeze in (1,2) and (2,1) ⇒ no safe actions

Assuming pits uniformly distributed:

(2,2) has pit w/ prob 0.36 vs. 0.31

Smell in (1,1) ⇒ cannot move

Can use a strategy of coercion:

Shoot straight ahead wumpus was there ⇒ dead ⇒ safe
wumpus wasn't there ⇒ safe

Note: logic focuses syntax (of some sort)

Entailment is a relationship between sentences (syntax)

Entailment is a relationship between sentences (syntax)

Entailment is a relationship between sentences (syntax)

Entailment means that one thing follows from another:

Entailment means that one thing follows from another:

Entailment means that one thing follows from another:

Note: brains process syntax (of some sort)

Entailment means that one thing follows from another:

Entailment means that one thing follows from another:

Entailment means that one thing follows from another:

Entailment means that one thing follows from another:

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General

Logic in General
Logicians typically think in terms of models, which are 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 $\alpha$. Then $KB \models \alpha$ if and only if $M(KB) \subseteq M(\alpha)$.

E.g. $KB = \text{Giants won and Reds won}$, $\alpha = \text{Giants won}$. Then $M(\alpha)$ contains exactly the model that saysGiants won.
Propositional logic: Syntax

Propositional logic is the simplest logic—illustrates basic ideas.

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).

Propositional logic: Semantics

Each model specifies true/false for each proposition symbol.

For example, \( P_1, P_2 \) are true, \( P_3 \) is false.

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

Rules for evaluating truth with respect to a model:

- \( \neg S \) 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.
- \( S_1 \Leftrightarrow S_2 \) is true iff \( S_1 \Rightarrow S_2 \) is true and \( S_2 \Rightarrow S_1 \) is true.

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

\[
\neg P_1 \land (P_2 \lor P_3) = \text{true} \land (\text{false} \lor \text{true}) = \text{true} \land \text{true} = \text{true}
\]

Truth tables for connectives:

\[
\begin{array}{c|c|c|c|c}
P & Q & \neg P & P \land Q & P \lor Q & P \Rightarrow Q & P \Leftrightarrow Q \\
\hline
T & T & F & T & T & T & T \\
T & F & F & F & T & T & F \\
F & T & T & F & T & T & F \\
F & F & T & F & T & T & T \\
\end{array}
\]

Wumpus world sentences

Let \( P_{1,j} \) be true if there is a pit in \([1, j]\).

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

"Pits cause breezes in adjacent squares."

\[
\begin{align*}
\neg P_{1,1} & \land \neg B_{1,1} & \land B_{2,1} & \land \neg P_{2,1} & \land \neg B_{2,1} \\
\end{align*}
\]
Chapter 7

38

Model checking

Proof methods differ into (roughly) two kinds:

Inference by enumeration

Validity and satisfiability

Truth-tables for inference

Wumpus world sentences

Logical equivalence

Truth tables (different assignments to symbols)

Exercise: Suggest a heuristic search in model space, search for a model where $\alpha \iff \beta$.

Exercise: What is a breeze in the Wumpus world? (Real sentences in a propositional logic.)

Let $P'$ be true if there is a breeze in $P'$. If $P, P'$ are in the same model, then $P \iff P'$:
A Horn Clause is a disjunction of literals of which at most one is positive.

Useful, as they are equivalent to:

- → proposition symbol; or
- (conjunction of symbols) ⇒ false

A natural way to describe facts

Forward and backward chaining

A KB is in Horn Form if

KB = conjunction of (restricted) Horn clauses

Restricted to exactly one positive literal

Modus Ponens (for Horn Form): complete for Horn KBs

\[\alpha_1, \ldots, \alpha_n, \alpha_1 \land \cdots \land \alpha_n \Rightarrow \beta\]

Can be used with forward chaining or backward chaining.

These algorithms are very natural and run in linear time.

Forward chaining algorithm

```plaintext
function PL-FC-Entails?(KB, q)
returns true or false
inputs: KB, the knowledge base, a set of propositional Horn clauses
        q, the query, a proposition symbol
local variables:
    count, a table, indexed by clause, initially the number of premises
    inferred, a table, indexed by symbol, each entry initially false
    agenda, a list of symbols, initially the symbols known in KB
while agenda is not empty do
    p ← Pop(agenda)
    unless inferred[p] do
        inferred[p] ← true
        for each Horn clause c in whose premise p appears do
            decrement count[c]
            if count[c] = 0 then do
                if Head[c] = q then return true
            end
        end
        Push(Head[c], agenda)
    end
return false
```

Forward chaining example

```
Q
P
M
L
B
A
```

Forward and backward chaining

```
Q
P
M
L
B
A
```
Proof of completeness

1. FC reaches a fixed point where no new atomic sentences are derived.
2. Consider the final state as a model $m$, assigning true/false to symbols.
3. Every clause in the original $KB$ is true in $m$.

Proof:

Suppose a clause $a_1 \land \ldots \land a_k \Rightarrow b$ is false in $m$.
Then $a_1 \land \ldots \land a_k$ is true in $m$ and $b$ is false in $m$.

However, by BC all premises of some rule concluding $b$ are known to be true.
Therefore the algorithm has not reached a fixed point!

Hence $m$ is a model of $KB$.

If $KB \models q$, $q$ is true in every model of $KB$, including $m$.

Backward chaining example

Idea: work backwards from the query $q$ to prove it by BC, checking if $q$ is known already, or proving by BC all premises of some rule concluding $q$.

Avoid loops: check if new subgoal is already on the goal stack.
Avoid repeated work: check if new subgoal is already proved true, or already failed.

Chapter 7 55

Chapter 7 56

Chapter 7 57

Chapter 7 58

Chapter 7 59

Chapter 7 60
Resolution is sound and complete for propositional logic.

\[
\frac{\neg p \land 
eg q}{\neg (p \lor q)}
\]

where \( p \) and \( q \) are conjunctions.

Resolution inference rule (for CNF): complete for propositional logic.

Resolution is sound and complete for propositional logic.

Resolution of CNF—universal.

Resolution is sound and complete for propositional logic.

Resolution of CNF—universal.

Resolution is sound and complete for propositional logic.

Resolution of CNF—universal.

Resolution is sound and complete for propositional logic.

Resolution of CNF—universal.

Resolution is sound and complete for propositional logic.

Resolution of CNF—universal.

Resolution is sound and complete for propositional logic.

Resolution of CNF—universal.

Resolution is sound and complete for propositional logic.

Resolution of CNF—universal.

Resolution is sound and complete for propositional logic.

Resolution of CNF—universal.

Resolution is sound and complete for propositional logic.

Resolution of CNF—universal.

Resolution is sound and complete for propositional logic.

Resolution of CNF—universal.
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 with respect to models
- **Entailment**: necessary truth of one sentence given another
- **Soundness**: derivations produce only entailed sentences
- **Completeness**: derivations can produce all entailed sentences
- **Resolution**: a method for deriving sentences from other sentences
- **Forward and backward chaining**: linear-time, complete for Horn clauses

The Wumpus world requires the ability to represent partial and negated information.

Propositional logic lacks expressive power.

Resolution is complete for propositional logic.

Forward, backward chaining are linear-time complete for Horn clauses.

Resolution is complete for propositional logic.

Propositional logic lacks expressive power.