ITCS 6150
Intelligence Systems

Lecture 10

Logical Agents

Chapter 7

Logical Agents

What are we talking about, "logical?"

- Aren't search-based chess programs logical
  - Yes, but knowledge is used in a very specific way
    - Win the game
    - Not useful for extracting strategies or understanding other aspects of chess
- We want to develop more general-purpose knowledge systems that support a variety of logical analyses

Why study knowledge-based agents

Partially observable environments

- combine available information (percepts) with general knowledge to select actions

Natural Language

- Language is too complex and ambiguous. Problem-solving agents are impeded by high branching factor.

Flexibility

- Knowledge can be reused for novel tasks. New knowledge can be added to improve future performance.

Components of knowledge-based agent

Knowledge Base

- Store information
  - knowledge representation language
- Add information (Tell)
- Retrieve information (Ask)
- Perform inference
  - derive new sentences (knowledge) from existing sentences

Knowledge Representation

Must be syntactically and semantically correct

Syntax

- the formal specification of how information is stored
  - a + 2 = c   (typical mathematical syntax)
  - a2y += (not legal syntax)

Semantics

- the meaning of the information
  - a + 2 = c   (c must be 2 more than a)

Logical Reasoning

Entailment

- one sentence follows logically from another
  - a->b
  - the sentence a entails the sentence b
- a->b if and only if
  - in every model in which a is true, b is also true

Inference Algorithms

Sound

- only entailed sentences are inferred
- always true

Complete

- inference algorithm can derive any sentence that is entailed

Propositional (Boolean) Logic

Syntax of allowable sentences

- atomic sentences
  - indivisible syntactic elements (can not be divided)
  - Use uppercase letters as representation
  - True and False are predefined proposition symbols

Complex sentences

Formed from symbols using connectives

- ~ (not): the negation
- ^ (and): the conjunction
- V (or): the disjunction
- => (implies): the implication
- ⇔ (if and only if): the biconditional

Backus-Naur Form (BNF)

$$
\begin{array}{rcl}
\mathit{Sentence} & \rightarrow & \mathit{AtomicSentence} \mid \mathit{ComplexSentence} \\
\mathit{AtomicSentence} & \rightarrow & \mathbf{True} \mid \mathbf{False} \mid \mathit{Symbol} \\
\mathit{Symbol} & \rightarrow & \mathbf{P} \mid \mathbf{Q} \mid \mathbf{R} \mid \ldots \\
\mathit{ComplexSentence} & \rightarrow & \neg\, \mathit{Sentence} \\
 & \mid & (\, \mathit{Sentence} \wedge \mathit{Sentence}\,) \\
 & \mid & (\, \mathit{Sentence} \vee \mathit{Sentence}\,) \\
 & \mid & (\, \mathit{Sentence} \Rightarrow \mathit{Sentence}\,) \\
 & \mid & (\, \mathit{Sentence} \Leftrightarrow \mathit{Sentence}\,)
\end{array}
$$

Propositional (Boolean) Logic

Semantics

- given a particular model (situation), what are the rules that determine the truth of a sentence?
- use a truth table to compute the value of any sentence with respect to a model by recursive evaluation
- model – valuation assigning values {T,F} to every atomic statement

Truth table

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	*false*	*true*	*false*	*false*	*true*	*true*
false	*true*	*true*	*false*	*true*	*true*	*false*
true	*false*	*false*	*false*	*true*	*false*	*false*
true	*true*	*false*	*true*	*true*	*true*	*true*

Figure 7.8 Truth tables for the five logical connectives. To use the table to compute, for example, the value of $P \vee Q$ when P is true and Q is false, first look on the left for the row where P is *true* and Q is *false* (the third row). Then look in that row under the $P \vee Q$ column to see the result: *true*. Another way to look at this is to think of each row as a model, and the entries under each column for that row as saying whether the corresponding sentence is true in that model.

Concepts related to entailment

logical equivalence

- a and b are logically equivalent if they are true in the same set of models…

  a⇔ b

validity (or tautology)

- a sentence that is valid in all models
  - P V ~P
  - deduction theorem: a entails b if and only if a implies b

satisfiability

- a sentence that is true in *some* model
- a entails b ⇔ (a ^ ~b) is unsatisfiable

Something to work on

ID	B	C	D	E	F
t1	b1	c1	d1	e1	f1
t2	b1	c1	d2	e2	f2
t3	b1	c2	d3	e3	f3
t4	b2	c3	d1	e4	f4
t5	b2	c1	d2	e5	f1
t6	b1	c3	d3	e1	f1
t7	b1	c2	d1	e2	f2
t8	b1	c2	d2	e3	f2
t9	b1	c2	d3	e4	f3
t10	b2	c3	d1	e5	f3
t11	b2	c3	d2	e5	f4
t12	b2	c4	d1	e5	f4

Attributes = {B, C, D, E, F}

Dom(B) = {b1, b2}
Dom(C) = {c1, c2, c3, c4}
Dom(D) = {d1, d2, d3}
Dom(E) = {e1, e2, e3, e4, e5}
Dom(F) = {f1, f2, f3, f4}

Query Language

1) Syntax
2) Semantics