CSC421 Intro to Artificial Intelligence

UNIT 12:
First-Order Logic

Using FOL

- Assertions
- Queries or goals
- Substitution or binding list
- Axioms
- Theorems (entailed by axioms)
- Should a KB only contain axioms ?

Interacting with FOL KBs

- Suppose a wumpus-world agent is using a FOL KB and perceives a smell and a breeze (but no glitter) at t = 5.
  - Tell(KB, Percept([Smell, Breeze, None], 5))
  - Ask(KB, ∃ a Action(a, 5))
  - i.e does KB entail any particular action at t = 5
- Answer: yes, {a/Shoot} - substitution
  - Binding list
  - Typically what we really want is the substitution for which there is an answer not just yes/no

Substitutions

- Given a sentence S and a substitution σ
  $S\sigma$ denotes the result of plugging σ into S
- For example
  - $S = Smarter(x,y)$
  - $\sigma = \{x/Foo, y/Bar\}$
  - $S\sigma = Smarter(Foo,Bar)$
- ASK(KB, S) returns some/all σ such that KB $\models S\sigma$

KB for the Wumpus world

- "Perception"
  - $\forall b,g,t$ Percept([Smell, b,g], t) => Smell(t)
  - $\forall s,b,t$ Percept([s, b,Glitter], t) => AtGold(t)
- "Reflex"
  - $\forall t$ AtGold(t) => Action(Grab, t)
- "Reflex with internal state"
  - $\forall t$ AtGold(t) $\wedge \neg$ Holding(Gold, t) => Action (Grab,t)
- Holding(Gold, t) can not be observed => keeping track of change is essential

Deducing Hidden Properties

- Properties of locations:
  - $\forall x,t$ At(Agent,x,t) $\wedge$ Smelt(t) => Smelly(x)
  - $\forall x,t$ At(Agent,x,t) $\wedge$ Breeze(t) => Breezy(x)
- Squares are breezy near a pit:
  - Diagnostic rule – infer cause from effect
    - $\forall y$ Breezy(y) => $\exists x$ Pit(x) $\wedge$ Adjacent(x,y)
  - Causal rule – infer effect from cause
    - $\forall x,y$ Pit(x) $\wedge$ Adjacent(x,y) => Breezy(y)
  - Neither of these is complete – e.g. The causal rul e doesn't say whether squares away from pits can be breezy
  - Definition of Breezy Predicate:
    - $\forall y$ Breezy(y) <=> $\exists x$ Pit(x) $\wedge$ Adjacent(x,y)

Knowledge Engineering

- Identify task
- Assemble relevant knowledge
- Decide on a vocabulary on predicates, functions and constants
- Encode general information about the domain
- Encode a description of the specific problem instance
  - “Disembodied” knowledge vs sensors
- Pose queries to inference procedure
- Debug KB

Circuits I

- Composed of wires and gaes
- Signals flow along wires to the input terminals of gates, and each gate produces signal on the output terminal that flow along another wire. AND, OR, XOR, NOT
- Modeling depends on what you want
  - Functionality and connectivity
  - Faulty circuits (inlcude wires in ontology)
  - Timing faults (include gate delays)

Circuits II

- Decide on vocabulary
- Gates
  - Distinguish gate from other gates (X1,X2...)
  - Type(X1) = XOR (use function)
  - Alternatively Type(X1,XOR) or XOR(X1)
- Terminals
  - In(1, X1)
- Connectivity
  - Connected(Out(1,X1), In(1,X2))
- Signal on/off
  - 1, 0

Encode General Knowledge of the domain

- If two terminals are connected, they have same signal
  - $\forall$ t1, t2 Connected(t1, t2) => Signal(t1) = Signal(t2)
- The signal at every terminal is either 1 or 0 (but not both)
  - $\forall$ t Signal(t) = 1 $\vee$ Signal(t) = 0 1 $\neq$ 0
- Connected is commutative
- OR gate ouput is 1 iff any of it's inputs is 1
  - $\forall$ g Type(g) = OR => Signal(Out(1,g)) = 1 <=> $\exists$ n Signal(In(n,g)) = 1
- AND similarly
- XOR
- NOT

Encode specific problem instance

- Gates
  - Type(X1) = XOR Type(X2) = XOR
  - Type(A1) = AND Type(A2) = AND
  - Type(O1) = OR
- Connections:
  - Connected(Out(1,X1), In(1,X2))
  - Connected(Out(1,X1), In(2, A2))
  - Connected(Out(1,A2), In(1,O1))
  - ..
  - ..
  -

Pose queries to inference procedure

- What combinations of inputs would cause the output of C1 to be 0 and the second output of C1 (the carry bit) to be 1
  - Answers: Substitutions for variables
- What are the possible sets of values of all the terminals in the adder circuit ?
  - Complete input-ouput table for device
  - Simple example of circuit verification
- Stuctured knowledge-base development