Week 1

Thursday, June 18, 2020 3:57 PM

1

Knowledge-based Agents :
    Agents that reason by operating on internal representations of knowledge

Sentence :
    An assertion about the world in a knowledge representation language

Model :
    Assignment of a truth value to every propositional symbol (a "possible world")

Knowledge base :
    A set of sentences Known by a knowledge-based agent

Entailment (a |= b) :
    In every model a is true b is also true

inference :
    The process of driving a new sentences from old ones

    Usage of inference : Does A |= B ?

    Example for inference algorithm : Model Checking

Propositional logic

Propositional symbols :
    Representation of a fact about the world

Logical connectives :
    Not $\neg$
    And $\wedge$ implication $\rightarrow$
    Or $\vee$ biconditional $\leftrightarrow$

Truth table for :
Implication :

P	Q	P ---> Q
true	true	true
true	false	false
false	true	true
false	false	true

Biconditional :

P	Q	P <---->Q
true	true	true
true	false	false
false	false	true
false	true	false

Knowledge Engineering

To solve a problem using propositional logic :
1- start by asking what **propositional symbols** you are going to need
2- using the symbols create logical sentences (knowledge)
3- using the sentences create the KB

Model Checking

To determine if KB |= a :

-Enumerate all possible models
-if in every model where KB is true , a is true
    Then KB entail a
-Otherwise KB does not entail a

2

Theorem Proving

Initial State : starting Knowledge base
Actions : inference rules
Transition model : New KB after inference
Goal test : check statement we are trying to prove
Path cost function : number of steps in proof

Inference Rules

:: Set of rulers that take knowledge that already exists and translate it to a new form of knowledge

- Modus Ponens :
  - If we have :
    - A --> B
    - A
  - Then we have :
    - B
- And Elimination :
  - If we have :
    - A $\wedge$ B
  - Then we have :
    - A
    - B
- Double Negation Elimination :
  - If we have :
    - $\neg(\neg$ A)
  - Then we have :
    - A
- Implication Elimination :
  - If we have :
    - A --> B
  - Then we have :
    - $\neg$ A V B
- Biconditional Elimination :
  - If we have :
    - A <-->B
  - Then we have :
    - (A --> B) ^ (B -->A)
- De Morgan`s law :
  - If we have :
    - Not(A ^ B)
  - Then we have :
    - Not(A) V not(B)
  - The reverse is also right **
    - If : Not(A V B) then we have not(A) ^ not(B)
- Distributive Property :
  - A ^ (B V C) give us : (A ^ B) V (A ^ C)
  - A V (B ^ C) give us : (A V B) ^ (A V C)

3

Inference by resolution
To determine if KB |= A :
    Check if (KB ^ not(A)) is a contradiction?
        If so, then KB |= A
        Otherwise , no entailment
And we do the check by :
    Convert (KB ^ not(A)) to conjunction Normal form(CNF)
    Keep checking if we can use resolution to produce new clause
        If we ever produce the empty clause ()
            Then we have a contradiction and
                KB |= A
        Otherwise, if we can not add new clause
            Then : no entailment

Clause :

A disjunction of literals

P V Q V R

Conjunction Normal form (CNF):

Logical sentence that is a conjunction of clauses

(A V B V C) ^ (D V E V F) ^ P

Conversion to CNF

- Eliminate biconditionals
  - Turn (A <--> B) into (A --> B) ^ (B -->A)
- Eliminate implications
  - Turn (A --> B) into not(A) V B
- Move not inward using De Morgan`s Law
  - Turn not(A ^ B) into not(A) V not(B)
- Use distribution law to distribute V whenever possible
-

Resolution rule

If we have :

P V Q

Not(p)

Then we have :

Q

If we have :

P V Q

Not(p) V R

Then we have :

Q V R

If we have :

A

Not(A)

Then we have :

()[false]

4

First-Order Logic

We have :

Constant Symbols : represent objects

Predicate Symbols : relations or functions (properties that might hold true or false)

Quantifiers

We have :

Universal Quantification : something is going to be true for all value of a variable

Existential Quantification : something going to be true for some value of a variable(at least one)