INTRODUCTION *to* PROPOSITIONAL & PREDICATE CALCULUS

RECAP

- According to Newell and Simon, intelligent activity, in either human or machine, is achieved through the use of:
  - Symbol patterns to represent significant aspects of a problem domain.
  - Operations on these patterns to generate potential solutions to problems.
  - Search to select a solution among these possibilities

RECAP

- Physical symbol system hypothesis justifies our efforts to build intelligent machines
- The physical symbol system hypothesis implicitly distinguishes between the *patterns* formed by the arrangement of symbols and the *medium* used to implement them
    - If intelligence derives only from the structure of a symbol system, then any medium that successfully implements correct patterns and processes will achieve intelligence, regardless of whether it is composed of neurons, logic circuits, etc.

KNOWLEDGE REPRESENTATION

- AI is concerned with
  - Qualitative rather than quantitative problem solving
  - Reasoning rather than calculation
  - Organizing large amounts of knowledge rather than implementing single, well defined algorithm
- Hence, an AI representation language must:
  - Handle qualitative knowledge
  - Allow new knowledge to be inferred from existing knowledge
  - Allow generalization and specialization
  - Capture complex semantics
  - Allow meta-level reasoning

PROPOSITIONAL CALCULUS

PROPOSITIONAL CALCULUS

- Proposition
- Propositional Calculus Symbols
  - P, Q, R,….
  - *True,False*
  - Connectives: $\wedge$, $\vee$, $\neg$, $\equiv$, $\rightarrow$
- Propositional Calculus Sentences
  - Every propositional symbol and truth symbol
  - Negation, Conjunction, Disjunction, Implication, Equivalence
- Propositional Calculus Semantics
  - An *interpretation* is the assignment of a truth value from the set {T , F} to each propositional symbol
  - True is assigned **T** and False **F**

PROPOSITIONAL CALCULUS

  - Negation of P i.e, ¬P, is assigned **T** if P is False and vice versa
  - Conjunction and Disjunction
  - Implication and Equivalence
- Equivalence Laws
  - Contrapositive
  - De Morgan's
  - Commutative
  - Associative
  - Distributive

PREDICATE CALCULUS

PREDICATE CALCULUS

- Predicates
- Predicate Calculus Symbols & Terms
  - Truth symbols: true and false
  - Constant symbols: having first character small
  - Variable symbols: having first character bold
  - Function symbols: firsts symbol small and having an arity
  - Predicate calculus terms are: either constant, variable, or function expression
- Predicate Calculus Sentences
  - We may combine atomic sentences using logical operators to form sentences in predicate calculus (same logical symbols as discussed earlier)

PREDICATE CALCULUS

  - Atomic Sentence
  - First Order Predicate Calculus
  - Every atomic sentence is a sentence
  - Its negation, conjunction, disjunction, implication, equivalence, universal/existential quantification

- Predicate Calculus Semantics
  - Interpretation
    - Assignment of each of constant, variable, function, predicate
- Problem of Universal/Existential Quantification for determining *truth value* of a sentence
  - *Undecidable*

Quantifiers

- Universal Quantifier "

  “Sentence holds true for all values of x in the domain of variable x” in other words “Express properties of collection of objects”

  - Syntax: " <variable> <sentence>
  - e.g; “Everyone studying at PU is smart”

    " ∀ x (StudiesAt(x; PU)) ⇒ Smart(x)) equivalent to:

    StudiesAt(Ahmed; PU)) ⇒ Smart(Ahmed)

    ^ StudiesAt(Rizwan; PU)) ⇒ Smart(Rizwan)

    ^ StudiesAt(Khalid; PU)) ⇒ Smart(Khalid)

    ^ …

Quantifiers

- Existential Quantifier ∃

  “Sentence holds true ***for some*** *value of* x *in the domain of variable* x” in other words “Express properties of some particular objects”

  – Syntax: ∃ <variable> <sentence>

  – e.g; “Someone studying at PU is smart”

  $\exists\, x\ \mathit{StudiesAt}(x; PU) \wedge \mathit{Smart}(x)$ equivalent to:

  $\mathit{StudiesAt}(\mathit{Ahmed}; PU)) \wedge \mathit{Smart}(\mathit{Ahmed})$

  $\vee\ \mathit{StudiesAt}(\mathit{Rizwan}; PU)) \wedge \mathit{Smart}(\mathit{Rizwan})$

  $\vee\ \mathit{StudiesAt}(\mathit{Khalid}; PU)) \wedge \mathit{Smart}(\mathit{Khalid})$

  $\vee\ \ldots$

Quantifiers

- Properties of Quantifiers

  $\forall x\ \forall y$ is the same as $\forall y\ \forall x$

  $\exists x\ \exists y$ is the same as $\exists y\ \exists x$

  – e.g

    $\forall x\ \forall y$ Likes(x,y)

    “Everyone likes everyone”

Quantifiers

- Properties of Quantifiers
  - □ $\forall x\ \exists y$ is *not* the same as $\exists y\ \forall x$
  - □ $\exists x\ \forall y$ is *not* the same as $\forall y\ \exists\ x$
  - – e.g;
    - □ $\exists x\ \forall y\ Loves(x; y)$

      “There is a person who loves everyone in the world”
    - □ $\forall y\ \exists x\ Loves(x; y)$

      “Everyone in the world is loved by at least one person”

Quantifiers

- Properties of Quantifiers

  $\forall x\ P(x)$ is the same as $\neg\exists\ x\ \neg P(x)$

  $\exists x\ P(x)$ is the same as $\neg\forall\ x\ \neg P(x)$

  – e.g

    $\forall x\ \mathit{sleep}(x)$

    “Everybody sleeps”

    $\neg\exists x\ \neg \mathit{sleep}(x)$

    double negative: “Nobody never sleeps”

General Definitions

- First order predicate calculus allows quantified variables to refer to objects in the domain and not to predicates or functions
- An atomic sentence is a predicate of arity *n*, followed by *n* terms: $t_1, t_2, \ldots. t_n$
- A free variable is a variable that isn't bound by a quantifier
  - *i.e.* ∃y Likes(x,y): x is free, y is bound by ∃ quantifier
- A well-formed formula is a sentence where all variables, if exist, are quantified (none are free)
- A ground expression has no variables at all

General Definitions

- Satisfy
  - An interpretation (I) that makes an expression (X) true is said to satisfy that sentence
- Model
  - If ‘I’ satisfies ‘X’ for all variable assignments then it is model for ‘X’
- Logically Follows
  - ‘X’ logically follows from a set ‘S’ of expressions if every interpretation that satisfies ‘S’ also satisfies ‘X’

Inference Rules

- Define a means of producing new predicate calculus sentences from other sentences.
- Provide a computationally feasible way to determine when an expression *logically follows* from other expression/s.
- Soundness & Completeness of Inference Rules
  - An inference rule is sound if every predicate calculus expression produced from a set S logically follows from S
  - Complete inference rule can infer every expression that logically follows from S

Inference Rules

- **Modus Ponens**
- **Modus Tollens**
- **And Elimination**
- **And Introduction**
- **Universal Instantiation**

Unification

- It is an algorithm for determining the substitutions needed to make two predicate calculus expressions match

  $\forall\ x\ (man(x) \Rightarrow mortal(x))$

  man(socrates)

  Unification: socrates/*x*

- The logical database must be expressed in an appropriate form

UNIFICATION ALGORITHM

- E1 and E2 are two expressions to be unified. The algorithm returns NIL to indicate an empty list, FAIL to indicate failure of unification process, and SUB to indicate the required list of substitutions
- Unify(E1,E2)
  - If E1 or E2 are both variables or constants then:
    - If E1 and E2 are identical, then return NIL
    - Else if E1 is a variable, then if E1 occurs in E2 then return FAIL, else return SUB = {E2/E1}
    - Else if E2 is a variable, then if E2 occurs in E1 then return FAIL, else return SUB = {E1/E2}
    - Else return FAIL
  - If the initial predicate symbols in E1 and E2 are not identical, then return FAIL
  - If E1 and E2 have a different no. of arguments, then return FAIL
  - Set SUB to NIL

UNIFICATION ALGORITHM

- For i = 1 to the arity of E1:
  - Call Unify with the i^th^ argument of E1 and E2, putting result in TEMP //a temporary list
  - If TEMP contains FAIL then return FAIL
  - If TEMP is not equal to NIL then:
    - Apply TEMP to the remainder of both E1 and E2
    - SUB = *append*(TEMP, SUB)
- Return SUB