d) Likes (Johhy, Books) ∧ Likes (Johny, music)

Here, Likes (x, y) represents that x likes y.
so, likes (Johhy, Books) ∧ likes (Johhy, music) represents that Johny likes both music & books.

Name - Aditya Nandwana
Reg :- 22BCG10121

Q.1 Explain Unification Algorithm

⇒ The unification algorithm is a process used in logic programming & automated theorem proving to determine if two expressions can be made identical by substituting variables with terms. It is the process of finding a substitution that can make two terms equal.

The algorithm takes two terms & tries to match them by recursively comparing their sub-terms. The comparision starts with top level symbols of each term, and if they match, the algorithm moves on to compare the subterms. If the top level symbols do not match, the algorithm tries to unify the terms by creating a variable to stand for one of the terms & then proceeding with the unification process.

1) Predicate logic representation

i) Marcus was a Pompeian

P(Marcus) ∧ P(Pompeian)

ii) All Pompeian were Romans

∀x: (P(Pompeian, x) → P(Roman, x))

2) Clause form representation

i) Marcus was a Pompeian

• ¬P(Marcus) ∨ P(Pompeian)

ii) All Pompeian were Romans

• ¬P(Pompeian, x) ∨ P(Roman, x) (for all x)

3) Resolution proof

Using the resolution method, we can prove "Marcus was a Roman" from the given premises

1) Assume ¬P(Roman, Marcus) (Negation of the statement to be proved)

2) From premise i, we have P(Marcus) and P(Pompeian)

3) From premise ii, we have ¬P(Pompeian, Marcus) ∨ P(Roman, Marcus)

4) Resolving clause 2 & 3 with literal P(Pompeian), we get P(Roman, Marcus)

Req-

5) Since we have derived the statement we were trying to prove, we can conclude that "Marcus was a Roman" is true

Q3 ->

=> i) ∃x (student(x) ∧ Took(x, AI∩L) ∧ Summer(2023))

Here took(x, AI∩L) represents that x took AI∩L & summer(2023) represent that it uses Summer 2023.

(ii) ∀x (student(x) ∧ Takes(x, AI∩L) -> passes(x, AI∩L)

Here takes(x, AI∩L) represents that x takes AIML & passes(x, AI∩L) represents that x passes AIML.

(iii) Man(Jack)

Here, man(x) represnt that x is a man.

(iv) ∃x (Doctor(x) ∧ Father of(x, mark))

Here, Doctor(x) represents that x is a doctor, & father of (x, mark) represents that x is the father of mark.

Utkarsh Srivastava
22BCG10144

Fundamentals of AI and ML (CSA2001)

Ans.1 Unification Algorithm is a key concept in automated reasoning and AI that is used to find a common substitution that makes two logical expression equivalent. In simple terms, it is an algorithm that takes two logical expression equivalent. It is an algorithm that takes two logical expressions as input tries to find a way to make them equal.

The most basic form of unification involves finding a substitution that makes two atomic terms equal. for example- if we have the expression "f(a, b)" and "f(x, y)" the algorithm can find a substitution that makes them equal by mapping the variable "x" to "a" and the variable "y" to "b".

It is a key tool for reasoning about complex expressions and finding solutions to problems that involve logical reasoning. Unification Algorithm is used in many areas of AI, including natural language processing.

ans 2 In predicate logic:

i. $P(m)$ (where $P(x)$ means "x is a Pompeian" and m is a constant symbol for "Marcus")

ii. $\forall x (P(x) \rightarrow R(x))$ (where $(R(x)$ means "x" is a Roman"))

In clause form

i. $\{-P(m)\}$

ii. $\{P(x), \neg R(x)\}$ (where x is a universally quantified variable).

To prove that "Marcus was a Roman" using resolution:

1. Add the negation of the conclusion "$\neg R(m)$" to the set of clauses

2. Apply resolution to the clauses:

$\{\neg P(m)\}, \{P(x), \neg R(x)\}, \neg R(m)$

$\{\neg P(m)\}, \{\neg R(m)\}$ (resolving on $P(m)$ and $P(x)$)

3. The resulting clause $\{\neg P(m), \neg R(m)\}$ is the empty clause which means the original set of clauses is unsatisfiable.

4. Since original set of clauses is unsatisfiable, the negation of conclusion must be false which means that "Marcus was a Roman" is true. Therefore we proved that Marcus was a roman.

ans 3. i. $\exists x$ (Students (x) $\wedge$ Took (x, AIML) $\wedge$ Summer (2023))

Here, Took (x, AIML) represents that x took AIML and Summer (2023) represent that it was summer 2023.

ii. $\forall x$ (Student (x) $\wedge$ Takes (x, AIML) $\rightarrow$ Passes (x, AIML))

Here, Takes (x, AIML) represents that x takes AIML, and Passes (x, AIML) represents that x passes AIML.

iii. Man (JACK)

Here, Man (x) represents that x is a man.

iv. $\exists x$ (Doctor (x) $\wedge$ Father Of (x, Mark))

Here, Doctor (x) represents that x is a doctor, and Father Of (x, Mark) represents that x is the father of Mark.

v. Likes (Johny, Books) $\wedge$ Likes (Johny, Music)

Here, Likes (x, y) represents that x likes y. So, Likes (Johny, Books) $\wedge$ Likes (Johny, Music) represents that Johny likes both music and books.

ans.4 INF - Logical expression are represented in form of an implication or a logical consequence it means that if some propositions are true, then some other proposition must also be true. It is way of making inferences or conclusions based on the given information.

INF is also called Horn form.

for example :- If it is raining then the ground is wet. The INF representation of this statement will be -

Raining ⇒ Wet Ground

Here, Raining is the antecedent or the condition and Wet Ground is the consequent or the logical consequence that follows from the condition.

CNF - Logical expression are represented as a conjuction (AND) of clause where each clause is a disjunction (OR) of literals. A literal is a variable or its negatiation.

CNF is used in various applications such as automated theorem proving logical programming and model checking.

Example of a CNF representation -

Consider the logical expression -

$(P \vee Q) \wedge (\neg P \vee R)$

This can be represented in CNF as a conjuction of two clauses:

$$(P \vee Q) \wedge (\neg P \vee R) = (P \wedge \neg P) \vee (P \wedge R) \vee (Q \wedge \neg P) \vee (Q \wedge R)$$

Here, each clause is a disjunction of literals and the entire expression is a conjuction of these clauses.

ans. 5. Forward Reasoning and backward reasoning are two methods used in logic to draw conclusions or make inferences from a given set of premises.

Forward Reasoning -

It starts with a set of premises and applies logical rules to derive new conclusion or statements. It moves from the given premises to a conclusion, using rules of inference to combine statements in order to reduce information.

for example -

All cats are animals
Tom is a cat

From these premises we can infer that tom is an animal.

Backward Reasoning - It starts with a goal or conclusion and works backward through the promises to determine what must be true in order for the conclusion to be valid. It involves applying rules of inference in reverse order to determine what promises are necessary to support the given conclusion.

Example of backward reasoning -

Goal: Tom is an animal.

From this goal, we can work backward through the promises to determine what must be true for the goal to be valid. In this case, we can use the promise "All cats are animal" and fact that Tom is a cat to conclude that Tom must be an animal.

In both forward and backward reasoning, the goal is to derive new statements or conclusions based on the given promises or goals. The difference is direction of the inference - forward reasoning moves from promises to conclusion, while backward reasoning moves from conclusions to promises.