Devanshu Surana
PC-23, 1032210755
Panel C, Batch C1

AIES Lab Assignment 4

Aim: To implement Unification Algorithm

Objective: To study and implement Unification algorithm.

Theory:

1) **Unification Algorithm:**
   1. It is a computational method used in symbolic reasoning and AI.
   2. Used for finding a common substitute for variable in logical expressions.
   3. This algo. plays a crucial role in various AI application such as NLP, automated theorem proving, etc.

   Condition for Unification:
   1. Predicate symbols must be same
   2. No. of args must be same for both literals.
   3. Unification fails if two similar variables appear in same expression.

2) **Resolution as Proof Procedure:**
   - A technique in automated theorem proving
   - Assumes negation of the statement to be proved and attempt to derive a contradiction.
   - Uses the resolution rule to ~~determine~~ combine clauses, aiming to prove the original statement true.

Page No.
Date

Input: Two literals L1 and L2
Output: A set of substitutions

Algo: Unification Algorithm

FAQ's

1. Why resolution is required?

→ It is a fundamental technique in automated theorem proving which is crucial in various fields of comp. Sci, AI and formal logic. It enables the automatic derivation of new logical conclusions from a set of existing premises.

Resolution is a complete inference rule, meaning that if there is a valid logical deduction to be made, resolution will eventually find it.

2. What are prerequisites for applying unification algorithm?

→ 1. Logical Statements: You need a set of logical statements / predicates typically represented in first-order logic.
2. Variable and constants: These statements should contain variables and constants, which are symbols representing variables.
3. Unification Terms: The literals should follow the conditions for unification.

3. What are the applications of Unification Algorithm?

→ Automated theorem proving in AI.
NLP for sentence parsing
Type Inference in programming languages
Knowledge representation in expert systems.

```python
def unify(e1, e2, theta={}):
    if theta is None:
        return None
    elif e1 == e2:
        return theta
    elif isinstance(e1, str):
        return unify_var(e1, e2, theta)
    elif isinstance(e2, str):
        return unify_var(e2, e1, theta)
    elif isinstance(e1, list) and isinstance(e2, list):
        if len(e1) != len(e2):
            return None
        else:
            for i in range(len(e1)):
                theta = unify(e1[i], e2[i], theta)
                if theta is None:
                    return None
            return theta
    else:
        return None

def unify_var(var, x, theta):
    if var in theta:
        return unify(theta[var], x, theta)
    elif x in theta:
        return unify(var, theta[x], theta)
    elif occurs_check(var, x, theta):
        return None
    else:
        theta[var] = x
        return theta

def occurs_check(var, x, theta):
    if var == x:
        return True
    elif isinstance(x, str) and x in theta:
        return occurs_check(var, theta[x], theta)
    elif isinstance(x, list):
        for e in x:
            if occurs_check(var, e, theta):
                return True
        return False

e1 = ["likes", "Parimal", "Kolhe","p"]
e2 = ["likes", "x", "y","q"]
theta = unify(e1, e2)
print(theta)
```

```
{'Parimal': 'x', 'Kolhe': 'y', 'p': 'q'}
```