Siddhanta Ghosh
1BM21CS106

A1

```
import re
def isVariable(x):
    return len(x) == 1 and x.islower() and x.isalpha()

def getAttributes(string):
    expr = '\([^)]+\)'
    matches = re.findall(expr, string)
    return matches

def getPredicates(string):
    expr = '([a-zA-Z~]+)\([^&|]+\)'
    return re.findall(expr, string)

class Fact:
    def __init__(self, expression):
        self.expression = expression
        predicate, params = self.splitExpression(expression)
        self.predicate = predicate
        self.params = params
        self.result = any(self.getConstants())

    def splitExpression(self, expression):
        predicate = getPredicates(expression)[0]
        params = getAttributes(expression)[0].strip('()').split(',')
        return [predicate, params]

    def getResult(self):
        return self.result

    def getConstants(self):
        return [None if isVariable(c) else c for c in self.params]
```

Scanned with CamScanner

Date: / /

```
def getVariables(self):
    return [v if isVariable(v) else None for v in
                                    self.params]

def substitute(self, constants):
    c = constants.copy()
    f = f"{self.predicate}({','.join([constants.pop(0)
        if isVariable(p) else p for p in self.params])}"
    return Fact(f)

Class Implication:
def __init__(self, expression):
    self.expression = expression
    l = expression.split('=>')
    self.lhs = [Fact(f) for f in l[0].split('&')]
    self.rhs = Fact(l[1])

def evaluate(self, facts):
    constants = {}
    new_lhs = []
    for fact in facts:
        for val in self.lhs:
            if val.predicate == fact.predicate:
                for i,v in enumerate(val.getVariables(
                    if v:
                        constants[v] = fact.getConstants()
                new_lhs.append(fact)
    predicate, attributes = getPredicate(self.rhs.expression)
    str(getAttributes(self.rhs.expression)[0])
    for key in constants:
        if constants[key]:
            attributes = attributes.replace(key, constants
                                            [key])
```

Signature:

Scanned with CamScanner

papergrid
Date: / /

```
expr = f'{predicate}({attributes})'
return Fact(expr) if res(res.lhs) and all([f.getResult() for f in res.rhs]) else None

class KB:
    def __init__(self):
        self.facts = set()
        self.implications = set()

    def tell(self, e):
        if '=>' in e:
            self.implications.add(Implication(e))
        else:
            self.facts.add(Fact(e))
        for i in self.implications:
            res = i.evaluate(self.facts)
            if res:
                self.facts.add(res)
    def query(self, e):
        facts = set([f.expression for f in self.facts])
        i = 1
        print(f'Querying {e}:')
        for f in facts:
            if Fact(f).predicate == Fact(e).predicate:
                print(f'\t{i}. {f}')
                i += 1
    def display(self):
        print("All facts:")
        for i, f in enumerate(set([f.expression for f in
                                   self.facts])):
            print(f'\t{i+1} {f}')
```

Scanned with CamScanner