19/11/24 First order Logic

Unification

~~John is parent of Alice.~~ Father
~~Amy is Mother of David.~~
~~x is Father.~~
~~All programmers write code for projects~~
$\forall x \forall y$ (Programmer(x) $\wedge$ Project(y) $\rightarrow$ WritesCodeFor(x,y))

Alice is a programmer
Programmer(Alice.)

Project X is a project
Project(ProjectX)

A programmer who writes code for a project is assigned to it
$\forall x \forall y$ (WritesCodeFor(x,y) $\rightarrow$ AssignedTo(x,y))

Only assigned programmers can access project details
$\forall x \forall y$ (AssignedTo(x,y) $\rightarrow$ CanAccess(x,y))

Bob is assigned to ProjectX
AssignedTo(Bob, ProjectX)
Unification
who can access ProjectX
$\exists x$ CanAccess(x, ProjectX)
AssignedTo(Bob, ProjectX)
$\forall x \forall y$ (AssignedTo(x,y) $\rightarrow$ CanAccess(x,y))
ProjectX is assigned To Bob
so Project x can be accessed By Bob

x = Bob
y = ProjectX

Program:

```
Knowledge base = [
  {"type": "rule", "rule": "∀x ∀y (Programmer(x) ∧ Project(y) → WritesCodeFor(x, y))"},
  {"type": "fact", "fact": "Programmer(Alice)"},
  {"type": "fact", "fact": "Project(Project X)"},
  {"type": "rule", "rule": "∀x ∀y (WritesCodeFor(x, y) → AssignedTo(x, y))"},
  {"type": "rule", "rule": "∀x ∀y (AssignedTo(x, y) → CanAccess(x, y))"},
  {"type": "fact", "fact": "AssignedTo(Bob, Project X)"},
]

query = {"predicate": "CanAccess", "arguments": ["?", "Project X"]}

def unify(kb, query):
    predicate = query["predicate"]
    target_project = query["arguments"][1]
    result = []

    for item in kb:
        if item["type"] == "rule" and predicate
        in item["rule"]:
        rule = item["rule"]

        if "AssignedTo(x, y)" in rule and
        "CanAccess(x, y)" in rule:

            for fact in kb:
                if fact["type"] == "fact" and "AssignedTo"
                in fact["fact"]:
                fact_parts = fact["fact"].split("(")[1].
                    strip(")").split(",")
```

```
        person, project = fact.parts

        if project == target_project:
            result.append(person)

if result:
    return f"The query '{query['predicate']}
    ({query['arguments'][0]}, {target
    project})' is unified: {', '.join(result)}
    can access {target_project}."
else:
    return f"The query '{query['predicate']}
    ({query['arguments'][0]}, {target_project})'
    could not be unified with the knowledge
    base."

result = unify(knowledge_base, query)
print(result)
```

Output

The query 'canAccess(?, ProjectX)' is unified: Bob can access ProjectX.

19/11/24