Ahsanullah University of Science and Technology

Department of Computer Science and Engineering

CSE4108: Artificial Intelligence Lab

Name: Rakib Hossain Rifat

ID: 16.02.04.099

Group: B2

Term Assignment # **01**

Resolution with Predicate Logic

Date of Submission: 01/09/2020

Resolution with Predicate Logic

Predicate Logic:

A predicate is an expression of one or more variables determined on some specific domain. A predicate with variables can be made a proposition by either authorizing a value to the variable or by quantifying the variable.

Resolution:

Resolution is a rule of inference leading to a refutation theorem-proving technique for sentences in propositional logic and first-order logic. In other words, iteratively applying the resolution rule in a suitable way allows for telling whether a propositional formula is satisfiable and for proving that a first-order formula is unsatisfiable. Attempting to prove a satisfactory first-order formula as unsatisfiable may result in a nonterminating computation; this problem doesn't occur in propositional logic. The clause produced by a resolution rule is sometimes called a resolvent.

Resolution rule:

The resolution rule in propositional logic is a single valid inference rule that produces a new clause implied by two clauses containing complementary literals. A literal is a propositional variable or the negation of a propositional variable. Two literals are said to be complements if one is the negation of the other (in the following, not c is taken to be the complement to c). The resulting clause contains all the literals that do not have complements. Formally:

where

all ai,bi and c are literals the dividing line stands for entails.

Steps for Resolution:

1. Conversion of facts into first-order logic.
2. Convert FOL statements into CNF
3. Negate the statement which needs to prove (proof by contradiction)
4. Draw resolution graph (unification).

Example:

Facts:

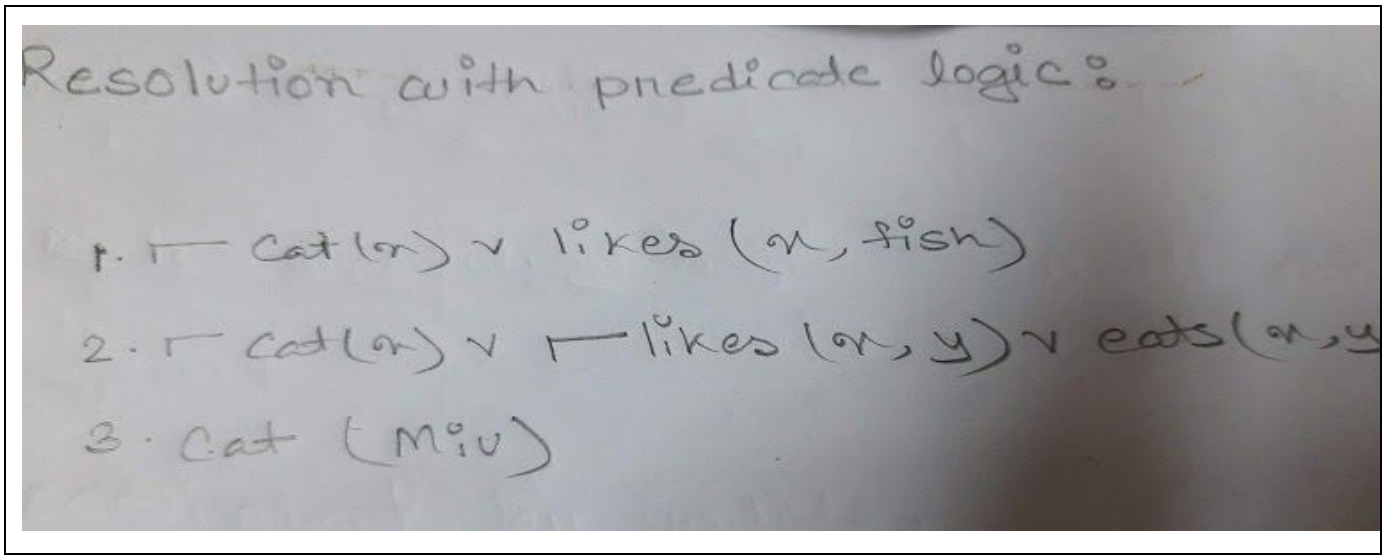
1. Cats like fish.
2. Cats eats everything they like.
3. Miu is a cat.

   does Miu east fish? eats(Miu,fish)

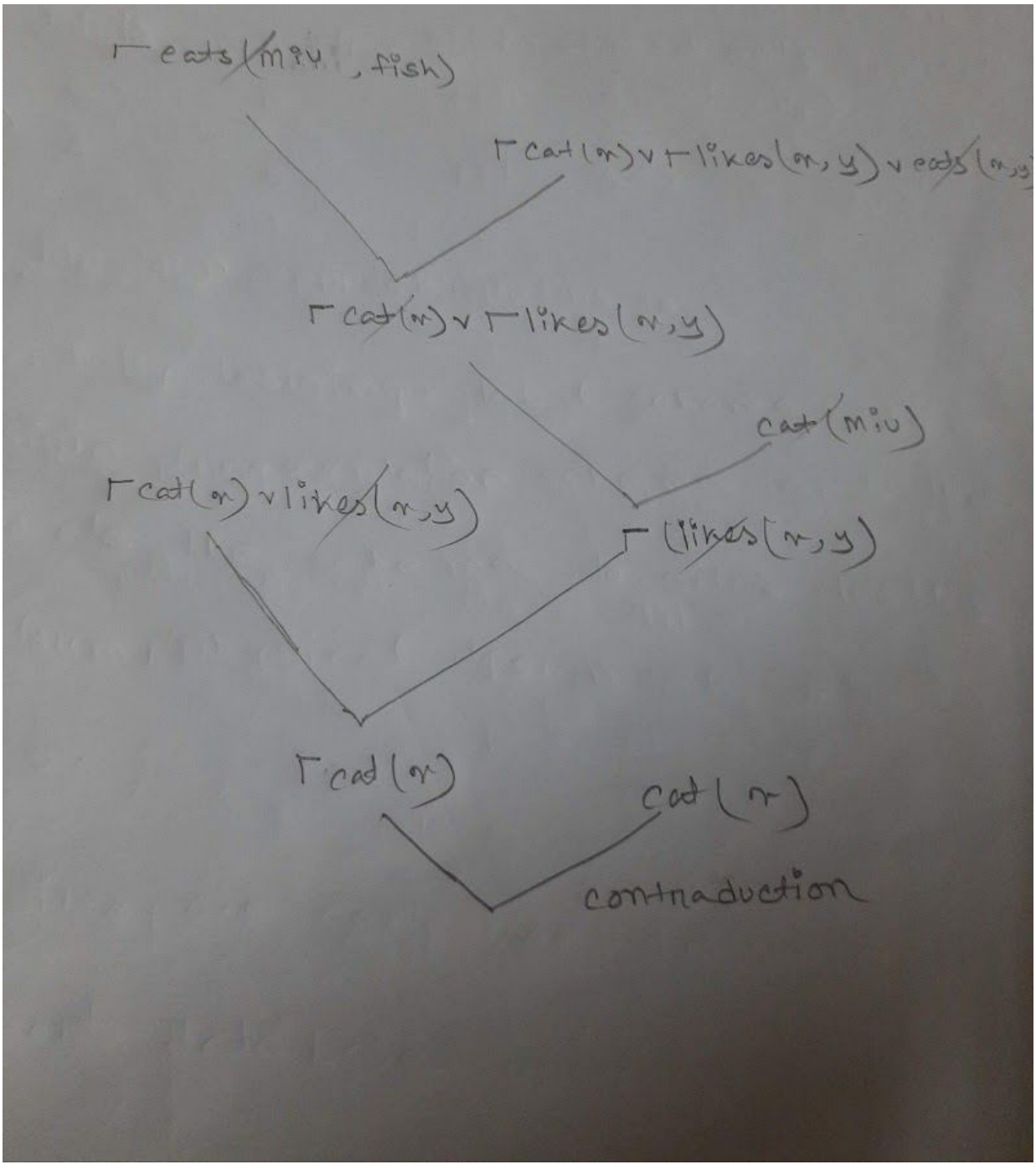
Step 1: Conversion of facts into first-order logic.

- Cats(x) -> likes(x,fish)
- [cat(x) ^ likes(x,y)] -> eats(x,y)
- cat(Miu)

Step 2 & 3: Convert FOL statements into CNF and Negate the statement which needs to prove



Step 4: Draw resolution graph



So !eats(Miu, fish) is wrong . so Miu eats fish is true. [Proved by resolution]

CODE:

```
read= open("input.txt","r")
lines=read.readlines()
read.close()
facts=[]
prove=[]
print("INPUT")
for item in lines:
    facts.append(item.replace("\n",""))
    print(item.replace("\n",""))
for line in facts:
    temp_fact=line.split(" | ")
    for l in temp_fact:
        prove.append(l)

print("Graph")
while(len(prove)!=1):
    poped=prove.pop(0)

    if poped[0]=="!":
        item=poped[1:]
        if item in prove:
            print("{} {}".format(poped,item))
            prove.pop(prove.index(item))
        else:
            prove.append(poped)
    else:
        prove.append(poped)
print(prove)
```

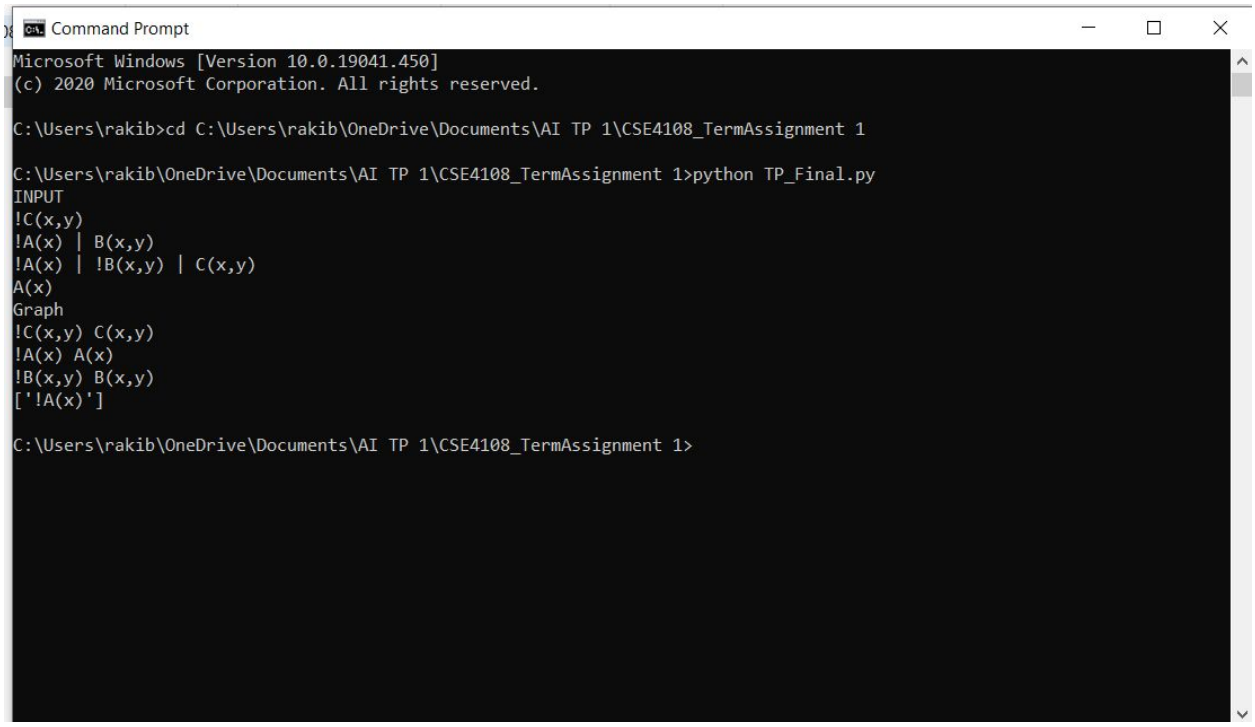
INPUT:

- A=Cat
- B=Likes
- C=Eats
- x= MIU
- y=Fish

This is taken as input in file

- !C(x,y)
- !A(x) | B(x,y)
- !A(x) | !B(x,y) | C(x,y)
- A(x)

Output:


Command Prompt
Microsoft Windows [Version 10.0.19041.450]
(c) 2020 Microsoft Corporation. All rights reserved.
C:\Users\rakib>cd C:\Users\rakib\OneDrive\Documents\AI TP 1\CSE4108_TermAssignment 1
C:\Users\rakib\OneDrive\Documents\AI TP 1\CSE4108_TermAssignment 1>python TP_Final.py
INPUT
!C(x,y)
!A(x) | B(x,y)
!A(x) | !B(x,y) | C(x,y)
A(x)
Graph
!C(x,y) C(x,y)
!A(x) A(x)
!B(x,y) B(x,y)
['!A(x)']
C:\Users\rakib\OneDrive\Documents\AI TP 1\CSE4108_TermAssignment 1>