Middle East Technical University Department of Computer Engineering

CENG 462

Artificial Intelligence

Fall '2019-2020

Homework 3

Due date: 15 December 2019, Sunday, 23:55

1 Objectives

This assignment aims to assist you to expand your knowledge on First Order Predicate Logic.

2 Problem Definition

In this assignment, you are going to implement a function called `theorem_prover` in python as a theorem prover for First Order Predicate Logic by using *Resolution Refutation* technique and *Set of Support* strategy with *Breadth-First* order. This function gets two lists of clauses, namely the list of base clauses and the list of clauses obtained from the negation of the theorem.

Your program has to eliminate

- tautologies
- subsumptions

The function detects whether the theorem is derivable or not. Then, it returns the result as a tuple. First element of this tuple will be **"yes"** or **"no"** according to derivability of the theorem. The second element will be:

- **list of resolutions** which contribute to the proof of the theorem, if it is derivable.
- **an empty list**, if the theorem is not derivable.

3 Specifications

- By the convention we follow in FOPL; variables, predicate and function names start with with a lower case letter, while the constants with an upper case letter.
- In the given clauses; "+" and "∼" signs will be used for disjunction and negation respectively.
- Each resolution in the return value must be given in "$< clause1 > \$ < clause2 > \$ < resolvent >$" form **without using any space character**.
- The "empty" string must be used for empty clause in the return value.

4 Sample Function Calls

```
>>> theorem_prover(["p(A,f(t))", "q(z)+~p(z,f(B))", "~q(y)+r(y)"],["~r(A)"])
('yes', ['~r(A)$~q(y)+r(y)$~q(A)', '~q(A)$q(z)+~p(z,f(B))$~p(A,f(B))', '~p(A,f(B))
    $p(A,f(t))$empty'])

>>> theorem_prover(["p(A,f(t))", "q(z)+~p(z,f(B))", "q(y)+r(y)"],["~r(A)"])
('no', [])
```

5 Regulations

1. **Programming Language:** You must code your program in Python **2.7**. Your submission will be tested in inek machines. So make sure that it can be executed on them as below.

   ```
   Python 2.7.15+ (default, Oct  7 2019, 17:39:04)
   [GCC 7.4.0] on linux2
   Type "help", "copyright", "credits" or "license" for more information.
   >>> import e1234567_hw3
   >>> e1234567_hw3.theorem_prover(...)
   ```

2. **Implementation:** You have to code your program by only using the functions in standard module of python. Namely, you **cannot** import any module in your program.

3. **Late Submission:** No late submission is allowed. No correction can be done in your codes after deadline.

4. **Cheating: We have zero tolerance policy for cheating**. People involved in cheating (any kind of code sharing and codes taken from internet included) will be punished according to the university regulations.

5. **Discussion:** You must follow the OdtuClass for discussions and possible updates on a daily basis.

6. **Evaluation:** Your program will be evaluated automatically using "black-box" technique so make sure to obey the specifications. A reasonable timeout will be applied according to the complexity of test cases in order to avoid infinite loops due to an erroneous code.

6 Submission

Submission will be done via OdtuClass system. You should upload a **single** python file named in the format **<your-student-id>_hw3.py** (i.e. e1234567_hw3.py).