CMSC 671 **Artificial Intelligence** Fall 2014

Project 2

Due November 4, 2014

In this assignment you will experience automatic theorem proving in first order logic (FOL). This is also an opportunity for you to practice more with Lisp. This project is divided into two parts as described in the following.

PART I (30 points):

In proj2.lisp you will find Lisp code for unification and resolution, two key components of a theorem prover. Your tasks are:

1. Read the code carefully and make sure you understand what each function does and how the functions call each other.
2. Test the code by trying to resolve the pairs of clauses given in Test Data Set below. Note that here any symbol preceded with a question mark is a variable, other symbols are predicates, functions or constants
3. Include the test results in the project report. For those pairs that are not resolved, find out and report why they failed.

Test Data Set:

```
1. '((p ?x) (q (f ?x))) '((~ p a) (r ?y))
2. '((p a) (~ q ?x)) '((~ p b) (q (f b)))
3. '((~ p a) (q (f ?x))) '((p (f ?y)) (~ q (g ?y)))
4. '((q ?x a b) (p (f ?x a) (g ?x b))) '((~ p ?y (g ?y b)) (r ?y))
5. '((p ?x (g ?x) (h b))) '((~ p (f ?u a) ?v ?u))
```

PART II (50 points):

Implement in Lisp a complete proof procedure for *refutation resolution* with *set of support* strategy on top of the functions defined in proj2.lisp. The proof procedure terminates if 1) a null clause is derived (the theorem is proved); or 2) no new resolvent can be generated (the theorem is not proved). You can implement the proof procedure either in the breadth first fashion or depth first fashion, but be careful to make sure that it does not terminate prematurely.

Your implementations should print out the following after each resolution:

The resolvent;
The two parent clauses;
The most general unifier (e.g., the substitution list generated by unification).

The printout should be in a form that is clear and easily understood. For example, the output for the first test case in PART I should look like

```
( ((q (f a)) (r y))        ;resolvent
  ((p x) (q (f x)))        ;parent1
  ((~ p a) (r y))          ;parent2
  ((x a))                  ;mgu
)
```

Test your implementation with the example "Did curiosity kill the cat" given in the lecture notes.

PART III (20 points):

Finally, you will use the theorem prover you develop to solve the following problem.

Prove that "Scrooge is not a child" is a theorem of the following set of axioms.

1. Every child loves Santa.
2. Everyone who loves Santa loves any reindeer.
3. Rudolph is a reindeer, and Rudolph has a red nose.
4. Anything which has a red nose is weird or is a clown.
5. No reindeer is a clown.
6. Scrooge does not love anything which is weird.

You should first encode the axioms and the theorem in FOL and then transform the axioms and the negated goal into clause form by hand before inputting them to the proof procedure.

SUBMIT. You should submit the following:

1. A cover page with your name, class, date, and a brief summary of what this project is about. You may also write comments and any points you wish to make.
2. Test results for PART I.
3. Lisp code for PART II and the running result of your code for "Did curiosity kill the cat" example.
4. The running result of your code with the problem in PART III. You also should include the FOL sentences and clauses for the axioms and goals.

Please zip your submission and email the zipped file to the grader (dorsaz1@umbc.edu) with the subject line "CMSC671 Project2 your first and last name".

Note: Please make sure that variables in different clauses have different names in PART II and PART III,