CS 364 – Assignment 3 – due 10/17

Deliverables: Hand in a directory containing `assign3.pdf`, your LaTeX'ed solutions to problems 1 - 5a. The program for 5b should be the file `jabber.py` and should be runnable. The program for problem 6 will be in the file `predres.py` distributed with the assignment, and for problem 7 use an appropriately named file with ".pl" extension.

You should work independently on problems 1 - 5a, but may collaborate as with earlier assignments on 5b, 6 and 7. Only 1 member of a team needs to hand in solutions to 5b, 6 and 7. Make sure your `readme.txt` identifies any team efforts.

1. (15) Represent the following sentences by predicate calculus wffs. Use as many predicates and terms as seems reasonable. Identify the predicates and terms before writing the wff.

   Example: If a program cannot be told a fact, then it cannot learn that fact.
   Solution:

$\text{program}(X)$	X is a program
$\text{fact}(Y)$	Y is a fact
$\text{told}(X, Y)$	X can be told Y
$\text{learn}(X, Y)$	X can learn Y

   $$\forall X \forall Y [\text{program}(X) \wedge \text{fact}(Y)] \rightarrow [\neg \text{told}(X, Y) \rightarrow \neg \text{learn}(X, Y)]$$

   (a) A computer system is intelligent if it can perform a task which, if performed by a human, requires intelligence.

   (b) If the input to the unification algorithm is a set of unifiable expressions, then the output is the mgu; if the input is a set of non-unifiable expressions, the output is *FAIL*.

   (c) If a production system is commutative, then, for any database, D, each member of the set of rules applicable to D is also applicable to any database produced by applying an applicable rule to D.

2. (15) Convert each of the following expressions to a set of clauses:

   (a) $A \vee (B \wedge C) \vee (D \wedge E \wedge \neg(A \vee B))$

   (b) $(\forall X)(P(X) \rightarrow (A(X) \wedge B(X) \vee \neg C(X, a)))$

   (c) $(\exists Y)(Q(Y, a) \wedge ((\forall Z)A(Z) \rightarrow \neg B(Y)))$

3. (10) Determine whether the following is a valid argument (without using a computer) by

   (a) Negating the conclusion and converting to clause form

   (b) Performing resolution refutation on the the resulting clause set.

   $$(\forall X)(A(X) \rightarrow B(X)), (\exists Y)(\neg B(Y)) \vdash (\exists Z)(\neg A(Z))$$

4. (18) Attempt to unify by hand the following sets of expressions. Either show their mgu's or explain why they won't unify.

   (a) $p(X, Y)$ and $p(a, Z)$.

   (b) $p(X, X)$ and $p(a, b)$.

(c) ancestor(X, Y) and ancestor(bill, father(bill)).

(d) $p(X)$ and $\neg p(a)$.

(e) $p(X, Z, Y)$, $p(W, U, W)$, and $p(a, U, U)$.

(f) $p(f(a), X)$ and $p(X, a)$.

5. (20) Use resolution to prove that the following logical argument, devised by Lewis Carroll, is valid.

   (a) Do it by hand, first converting to clause form.

   (b) Use your resolution program from (6) to prove it electronically.

No shark ever doubts that it is well fitted out A fish that cannot dance a minuet is contemptible No fish is quite certain that it is well fitted out, unless it has three rows of teeth All fishes, except sharks, are kind to children No heavy fish can dance a minuet A fish with three rows of teeth is not to be despised (i.e. not contemptible) *Conclusion*: No heavy fish is unkind to children

   Hints: i) the first line becomes "shark$(X) \rightarrow$ certainWellFittedOut(X)", which in clause form is "$\neg$shark$(X) \vee$ certainWellFittedOut(X)". ii) Negating the conclusion should produce a Skolem constant. iii) The resolution proof will use every clause to reach a contradiction.

6. (10) Using the predicate calculus representation and unifier contained in `predunify.py`, extend the resolution theorem prover in `propres.py` to handle predicate calculus. Put the solution in the file `predres.py` distributed with the assignment.

   This problem should only require changes to less than 5 lines of code from the propositional calculus version.

   The file `predres.py` included with the assignment is set up showing the functions that may need modification. It also contains some example data and prettyprint functions to display the answer.

7. (12) Write a Prolog deduction system similar to the ones contained either in `clue.pl` or `animal.pl`, which employ user queries to obtain facts not declared in the program. Be creative but also careful that your program is capable of performing the deductions.

   The Blackboard Documents folder contains resources for downloading and learning how to use SWI Prolog.