Rheinische Friedrich-Wilhelms-Universität Bonn
Institut für Informatik III
Prof. Dr. A. B. Cremers

Advanced Logic Programming
Summer Semester 2010
Dr. Günter Kniesel

Assignment 3

Due: Sunday, 09.05.2010, 23:59:59 via SVN

For help, contact alp-staff@lists.iai.uni-bonn.de (staff only) or alp-course@lists.iai.uni-bonn.de (staff and participants).

Please start working on the exercises early enough so that you can contact us in time in case of problems. Don't expect us to be available during weekend!

Task 1. *Resolution of prolog programs* (2 Points)

Consider the following program and write down the resolution steps that lead to the first result for the query

```
?- big(X), dark(X).
```

Use the notation introduced in the lecture on slide 2-42: (goal, clause, mgu, resolvent).

```
big(bear).                % #1
big(elephant).            % #2
brown(bear).              % #3
black(cat).               % #4
gray(elephant).           % #5
dark(Z):-                 % #6
      black(Z).
dark(Z):-                 % #7
      brown(Z).
```

Task 2. *Clause normalization* (5 Points)

The transformation of a predicate logic formula into clause form through normalization was presented in the lecture. Transform following formula into clause form:

$$\exists Z: [\neg\exists X: \Big(P(X,Z) \vee \forall Y: Q\big(X, f(Y)\big)\Big) \vee \forall Y: P(g(Z,Y),Z)]$$

1. Clean the formula by systematically renaming bound variables so that they are unique. The resulting formula is called F1.
2. Bring F1 into the equivalent formula F2 in Prenex Normal Form, meaning all quantifier are at the beginning of the formula.
3. Create the Skolem Normal Form of F2 by replacing every existential variable with a unique constant. The resulting formula F3 only preserves the satisfiability of F2. Finally remove all universal quantifier from the formula F3.
4. Transform the formula F3 into a formula F4 which is in Conjunctive Normal Form, a conjunction of disjunctions of atoms or negated atoms.
5. Write F4 in clause form by replacing the logical conjunctions "∧" with ",". The resulting set of clauses is called F5.
6. Notate the set of clauses F5 as Horn clauses by writing negated literals on the left side and positive literals on the right side of the implication.
7. Write the clauses in Prolog syntax.

Task 3. *Resolution* (3 Points)

Given the following set of clauses

$$F = \{\, [\, loves(hobbes, calvin)], [loves(hobbes, susi)]\,, [loves(calvin, X) \lor \neg loves(X, susi)]\, \}$$

Formulate the query "Is there someone calvin loves?" as a first order formula. Deduct with the resolution principle the answer to this query. Which additional information is generated during this process and from which mechanism?

Task 4. *Declarative semantics* (4 Points)

```
    likes(a,b).
    likes(c,d).
    likes(X,Y) :-
        likes(Y,X).
```

Given the above program:

a) Describe the Herbrand Universe.

b) Describe the Herbrand Base.

c) Give a Herbrand Interpretation.

Task 5. *Operation semantics* (4 Points)

a) For the program given in task 4, which answers will Prolog derive for the goal `likes(X,Y)`?

b) How often will Prolog succeed for the goal in a)?

c) Construct a non-terminating query for the program given in task 4. A non-terminating query is a query that does not return any answer, and will need infinity to find out that there is no answer.

Bonus Task *Depth of Derivation Tree* (4 Points)

Bonus tasks are optional. They are intended for advanced students to give them a challenge and also a chance to get the points needed for exam admission more quickly.

For the following task take the following definition of append/3:

```
    append([X|Xs], Ys, [X|Zs]) :-
          append(Xs,Ys,Zs).
    append([],Ys,Ys).
```

a) (1 Point) Describe the Herbrand Universe and the Herbrand Base of the program consisting of the append predicate.

b) (3 Points) Show that the depth of the derivation tree of append has m+1 nodes. Hence that it has linear complexity.