Exam, TDDD08 Logic Programming

2017-08-18, 14:00 – 18:00

Means of assistance (hjälpmedel):
• A sheet of notes – 2 sided A5 or 1 sided A4. The notes should be signed in the same way as the exam sheets and returned together with the exam.
• English dictionary

You may answer in English or Swedish.

Grade limits: 3: 17 p, 4: 23 p, 5: 29 p (for total 35 p).

Remember to give explanations for all answers!
Unexplained answers may be granted 0 points.
For instance, when you write a program you should explain the relations defined by the predicates of the program, and the role of each clause.

GOOD LUCK!

---

1. Determine which of the following pairs of terms are unifiable, and provide a most general unifier (mgu) in case there is one.

   a) $p(X, f(Z), f(V))$ and $p(Y, f(X), Z)$

   b) $p(X, g(Y), X)$ and $p(f(Z), Z, f(g(a))$

   c) $p([X|Z], Z)$ and $p([f(Y), f(Y)], f(a))$

   d) $p(f(Z), X, f(Z))$ and $p(f(X), g(Y, V), V)$

   (4 points)

2. Consider a program P:

   $$p(X, f(Y)) \leftarrow r(a, Y). \qquad q(X, Y) \leftarrow r(X, f(Y)). \qquad r(f(Z), Z).$$

   Provide a query Q such that the result of executing P with Q under Prolog is different from what is theoretically described by SLD-resolution. (2 points)

3. Let us consider ordered binary trees, where each node contains a data item, which is an arbitrary term. Let us represent the trees as terms in the following way. Constant `e` represents the empty tree; a term $\texttt{tr}(l, t, r)$ represents a nonempty tree, where t is the term contained in the root, l is (the term representing) the left subtree of the root, and r is (the term representing) the right subtree of the root,

Write a logic program which checks whether a term is a representation of a non-empty tree as above. Write a logic program which produces the in-order list of the terms from the nodes of a given tree.

In-order means that the terms from the left subtree of a node n precede the term from n, and the latter is followed by the terms from the right subtree of n.

For the full score the program should efficiently work as a Prolog program; in particular it should avoid the inefficiency due to multiple traversing the same fragments of lists.

Hint: Begin with the obvious, inefficient version. Use difference lists for efficiency.
Your programs should be definite clause programs, without negation and any Prolog built-ins. (5 points)

4. Design a representation for non-empty ordered trees. In such a tree each node has zero or more children, and each node contains a term.

   Write a program which checks whether a term is contained in a tree.

   We do not specify how your program should behave on incorrect input data, i.e. a term which does not represent a tree instead of one which does.
   You are not allowed to use any built-ins of Prolog, this includes negation. (3 points)

5. Consider the following definite program P:

$$p(\, g(X), f(X)\,).$$
$$p(\, Y, f(Z)\,) \leftarrow p(\, g(Y), Z\,).$$
$$q(\, X\,) \leftarrow p(\, Z, Z\,).$$

   Assume that the vocabulary $\mathcal{A}$ contains one constant a and two one-argument function symbols f, g. What is the Herbrand universe $\mathbf{U}_\mathcal{A}$ corresponding to $\mathcal{A}$?

   Find the least Herbrand model $\mathbf{M}_P$ of the program. Alternatively, find the set $PTR(P)$ of atomic logical consequences of the program.

   Give an example of a ground atom which is a logical consequence of P; the atom should not be an instance of a unary clause of the program.

   Give an example of a ground atom which is not a logical consequence of P, its predicate symbol should be p.

   Give an example of a non-ground atom which (1) is a logical consequence of P, and (2) is not an instance of any unary clause of the program.
   (5 points)

6. For a chosen query Q and a chosen subset $P' \subseteq P$ of the previous program construct two SLD-trees (using different selection rules) – one finite and one infinite. (2 points)

7. Consider a program SUFFIX:

$$s(L, L).$$
$$s(S, [H|L]) \leftarrow s(S, L).$$

(a) Is the program correct with respect to the following specification?

$$S_1 = \{\, s([t_j, \ldots, t_n], [t_1, \ldots, t_n]) \in \mathbf{B}_\mathcal{A} \mid 1 \leq j \leq n+1 \,\}.$$

A brief explanation is sufficient here.
($\mathbf{B}_\mathcal{A}$ is the Herbrand base; by $[t_{n+1}, \ldots, t_n]$ we mean the empty list.)

(b) Using a standard method, prove that the program is correct w.r.t. a specification

$$S = \left\{\, p(u, t) \in \mathbf{B}_\mathcal{A} \;\middle|\; \text{if } u, t \text{ are lists then } |u| \leq |t| \,\right\}$$

(where $|l|$ is the length of a list l, e.g. $|[t_1, \ldots, t_n]| = n$). (4 points)

8. Consider the DCG:

```
p( [a] ) --> [a].        p( [a,b|L] ) --> [c], p( [b|L] ), [d].
p( [b] ) --> [].         p( [b,a|L] ) --> [c], p( [a|L] ), [d].
```

(a) Translate the DCG into a Prolog program (using a standard approach).

(b) Show that $[c, c, a, d, d]$ is a member of the language of $p([a, b, a])$ by sketching a proof tree or a successful SLD-derivation.

(c) What is the language of $p([a, b])$?

(3 points)

9. Consider the following general program P:

$$p(X) \leftarrow q(X).$$
$$p(X) \leftarrow \neg\, q(X).$$
$$q(a).$$
$$q(s^2(Y))) \leftarrow q(Y).$$

We abbreviate $s(s(X))$ by $s^2(X)$, $s(s(s(X)))$ by $s^3(X)$ and so on.

Draw SLDNF-forests for queries $p(s^3(a))$ and $p(X)$. Make it clear, which trees are finitely failed, which leaves are floundered, which branches are successful derivations, and what are their answers. (If a tree has many answers, make explicit at least two of them.)

Construct the completion $comp(P)$ of the program (except for the equality axioms CET). Explain whether $p(X)$ is a logical consequence of $comp(P)$. Compare this with the corresponding SLDNF-forest.

(5 points)

10. Choose two cases from the list below, and explain the notions.

    Your explanation should be short but precise, and should show that you understand the notions. The chosen notions should not be explained in your sheet of notes.

    (a) Model of a formula.

    (b) Unification.

    (c) Completeness of SLD-resolution

    (d) Specification (of a logic program), correctness of a program with respect to a specification.

    (e) Incompleteness diagnosis.

    (f) Constraint solver.

    (g) General program.

    (h) Closed world assumption (CWA).

    (2 points)