

CSE-505 Computing with Logic

Fall '04

Final Exam

(Take Home)

Due: Dec 14, 2004 (by midnight)

Max: 100 points

Instructions:

- If you are submitting answers written on paper, please hand the exam to me personally on the due date, or put your exam in a clearly marked envelope and slide it under my office door. You may also submit electronically. Put your answers, including programs and program fragments, in a **single** file. I prefer to get answers in a plain, ascii, text file. If you want, however, you may submit answers in a PDF file. Please do not submit Word or other editor-specific files. Send the file to me by email (to cram@cs.sunysb.edu).
 - Clearly mark the question number corresponding to each answer.
 - For programs, you may use XSB's library predicates *unless expressly prohibited*. You may also use predicates from Bratko's book, but for each predicate, please remember to cite the page number in the text where the predicate is defined.
 - For programs, please make sure you *indent* the programs to make them readable, to avoid grading problems.
-

1. [12 points] Write a Prolog predicate `incsub/2` such that, given a list of integers `L1`, `incsub(L1, L2)` returns in `L2` an increasing subsequence of `L1`. (You may assume that `L1` contains distinct elements).

L_2 is a subsequence of L_1 if all elements of L_2 also occur in L_1 , and if a occurs before b in L_2 then a occurs before b in L_1 also.

L is an increasing sequence if the elements are in ascending order: i.e. if a occurs before b in L , then $a < b$.

For instance, `incsub([2,4,1,7,3,8], L2)` should succeed with answers `L2 = []`, `L2 = [2]`, `L2 = [4]`, `L2 = [2,4]`, `L2 = [1]`, ... (upon backtracking; a total of 26 distinct answers).

2. [12 points total (6 points ea.)]

- (a) Write a predicate `indivisible/2` that, given a list of positive integers L and a positive integer I succeeds if and only if I is not divisible by any integer in L .

For instance, `indivisible([2,3,5], 7)` succeeds while `indivisible([2,3,5], 9)` fails.

Note: both arguments will be bound in any query to `indivisible`.

Hint: Use the `mod` operation (e.g. `X is Y mod Z`) to check for divisibility.

- (b) Consider the following predicate `gen/2`:

```
gen(N, N).
gen(N, M) :- gen(N, I), M is I+1.
```

`gen(n, X)` generates all integers (upon backtracking) that are greater than or equal to n .

Let L be a list of integers in descending order. Using `gen` and `indivisible`, write a predicate `next/2` such that `next(L, N)` succeeds with the N bound to the smallest number that is (a) larger than any number in L and (b) is not divisible by any number in L .

For example `next([7,5,3,2], X)` will succeed with `X=11`.

For full credit, the predicate `next(L,X)` must terminate with exactly one answer as long as all integers in L are ≥ 2 .

Note: first argument of `next` will always be bound to a list of positive integers in descending order (no need to check for this).

3. [10 points] Consider the following definite logic program:

$$\begin{aligned} p(X) &\leftarrow q(Y, X), r(Y). \\ q(s(X), Y) &\leftarrow q(X, Y). \\ r(0). \end{aligned}$$

Show that there is one computation rule such that the query $\leftarrow p(0)$ has a finitely failed SLD tree and another computation rule such that $\leftarrow p(0)$ has an infinite SLD-tree.

4. [20 points (5 points ea.)] Let P be a propositional general logic program, and A be an arbitrary literal. State whether each of the following statements are true or false. If a statement is true, argue why; if false, give a counter example.

- (a) If A is in *some* stable model of P then A is also in the well-founded model of P .
 (b) If A is in *all* stable models of P then A is also in the well-founded model of P .

- (c) If A is in the well-founded model of P then A is also in some stable model of P .
- (d) If P is a stratified program then its well-founded model is 2-valued (i.e., for every proposition p in the program, either p or $\neg p$ is in the well-founded model)

5. [12 points total] Consider the following program:

```
r(X,Y) :- e(X,Y).
r(X,Y) :- e(X,Z), r(Z,Y).
```

where $e/2$ represents the edge relation of some graph.

- (a) [6 points] Consider the complete graph of four vertices, with vertices labelled 1, 2, 3, 4, and the tabled resolution of query $r(1, A)$.
What will be the entries in the call table made when resolving the above query?
What will be the entries in the answer table for call $r(1, A)$ when resolving the above query?
- (b) [6 points] Consider an arbitrary graph with n vertices and m edges, and a top-level query of the form $r(v, A)$ for some vertex v in the graph.
What will be the number of call table entries (in the worst case) in terms of n and m ?
What will be the total number of answer table entries (over all call tables, in the worst case) in terms of n and m ?

6. [12 points total] Let $v/1$, $p/2$ and $q/1$ be three predicates defined in a program P .

- (a) [4 points] Extend P to define a new predicate $r/1$ such that $r(X)$ holds whenever

$$\exists Y.p(X, Y) \wedge q(Y)$$

The extended program must be a definite logic program.

For example, if the following is least Herbrand model of P :

```
v(1).    p(1,2).    q(2).
v(2).    p(1,3).    q(4).
v(3).    p(2,3)
v(4).    p(3,4).
```

then $r(1)$ and $r(3)$ are logical consequences of the extended program.

- (b) [8 points] Extend P to define a new predicate $s/1$ such that $s(X)$ holds whenever

$$v(X) \wedge (\forall Y.p(X, Y) \Rightarrow q(Y))$$

The extended program may be a general logic program.

For example, if the following is the least Herbrand model of P :

```
v(1).    p(1,2).    q(2).
v(2).    p(1,3).    q(4).
v(3).    p(2,3)
v(4).    p(3,4).
```

then $s(3)$ and $s(4)$ are logical consequences of the extended program (and $s(1)$ and $s(2)$ are *not* logical consequences).

Do not use any Prolog built-ins!

7. [10 points total] Consider the following set of statements:
- Penguins are birds
 - Ducks are birds
 - Canaries are birds
 - Generally, all birds other than penguins can fly
 - Donald Duck is a duck
 - Donald Duck cannot fly
 - Tweety is a canary
 - Willy is a penguin
- (a) [8 points] Represent the above knowledge using F-logic. For full credit, represent the above knowledge using only facts i.e. without using rules of the form $a : -b \dots$
- (b) [4 points] Write an F-logic query to find who can fly.
8. [10 points total]
- (a) [6 points] What term is constructed in the heap when the following sequence of WAM instructions is executed:
- ```

put_structure a/0 X4
put_structure f/2, X2
set_value X4
set_variable X5
put_structure g/1, X3
set_value X5
put_structure h/2, X1
set_value X2
set_value X3

```
- (b) [4 points] Give the sequence of WAM instructions to unify a query term (in register X1) with the program term  $p(f(X), g(a, X))$ .

---

END OF EXAM