Faculty of Engineering, Mathematics and Science
School of Computer Science & Statistics

Integrated Computer Science Programme
B.A. (Mod.) Computer Science & Business
B.A. (Mod.) Computer Science & Language
Mathematics

Michaelmas Term 2018

Symbolic Programming

Thu, 13 Dec 2018
RDS-SIM COURT
14:00 – 16:00

Dr Tim Fernando

Instructions to Candidates:
Attempt two questions. All questions carry equal marks. Each question is scored out of a total of 50 marks.

You may not start this examination until you are instructed to do so by the Invigilator.

Exam paper is not to be removed from the venue.

Materials permitted for this examination:
Non-programmable calculators are permitted for this examination — please indicate the make and model of your calculator on each answer book used.
1. (a) State how a Prolog interpreter responds to the following queries, assuming no Prolog program has been consulted.

(i) \(?- [1,2] = [X|Y].\)

(ii) \(?- X \text{ is } 0+Y.\)

(iii) \(?- X = 0+Y\)

(iv) \(?- <(0,1).\)

(v) \(?- \text{assert}(r(a)), r(a).\)

(vi) \(?- \text{retract}(r(a)), r(a), \text{assert}(r(a)).\)

(vii) \(?- (X=1;X=2), Y = X+1.\)

(viii) \(?- (X=1;X=2),! ,X=Y.\)

(ix) \(?- \text{findall}(X,X=X=Y,L).\)

(x) \(?- \text{setof}(X,X\|X|1,L).\)

[20 marks]

(b) The predicate \texttt{memb} below is like \texttt{member}, except the recursive (or inductive) clause is declared before the base case.

\[
memb(X,[\_|T]) :- memb(X,T).

memb(X,[X|\_]).
\]

How does Prolog respond to the query

\(?- \text{memb}(X,[1,2,3]).\)

[10 marks]

(c) Recall that the non-negative integers can be encoded as follows

\[
\text{numeral}(0).
\text{numeral(succ}(X)) :- \text{numeral}(X).
\]

Define the 3-ary predicates \texttt{add}(X,Y,Z) and \texttt{multiply}(X,Y,Z) to encode addition and multiplication so that, for example,

\(?- \text{add}(\text{succ}(0), Y, \text{succ}(\text{succ}(0))).\)

\(Y = \text{succ}(0);\)

\text{no.}
?- multiply(succ(0), Y, succ(succ(0))).
Y = succ(succ(0)) ;
no.

[10 marks]

(d) A binary predicate edge/2 is **symmetric** if whenever edge holds of a pair \((a, b)\), it also holds of \((b, a)\). An obvious way to make edge symmetric is by adding the rule

\[
\text{edge}(A, B) :- \text{edge}(B, A).
\]

Why is this problematic, and how can we get around this problem whilst turning edge into a symmetric predicate symEdge?

[10 marks]
2. The \textbf{nth derangement number} $D_n$ is the integer defined inductively as follows

\begin{align*}
D_1 & := 0 \\
D_2 & := 1 \\
D_n & := (n - 1)(D_{n-1} + D_{n-2}) \quad \text{for } n > 2
\end{align*}

so that, for example,

\begin{align*}
D_3 & = 2(1 + 0) = 2 \\
D_4 & = 3(2 + 1) = 9.
\end{align*}

This question is about defining a binary predicate \texttt{d} computing derangement numbers, leading the Prolog interpreter to respond, for example, as follows.

?- \texttt{d(3,D)}.
D=2 ? ;
no

?- \texttt{d(4,D)}.
D=9 ? ;
no.

(a) A simple attempt to define \texttt{d} is given below, with question marks


that must be replaced by suitable Prolog variables.

\texttt{d(1,0).}
\texttt{d(2,1).}

\texttt{d(N,D) :- N>2,}
\hspace{1em} ?1 is N-1,
\hspace{1em} ?2 is N-2,
\hspace{1em} d(N1,?3),
\hspace{1em} d(N2,?4),
\hspace{1em} D is ?5*(D1+D2).}

Replace the question marks ?1,?2,?3,?4,?5 so that the code above computes derangement numbers.
(b) What is tail recursion and why is the definition of $d/2$ in part (a) not tail recursive?

(c) Revise the definition of $d/2$ above for a tail-recursive program computing the $n$th derangement number.

(d) What is memoization and what is it good for?

(e) Use memoization to compute the $n$th derangement number, revising the definition of $d/2$ in part (a) above.
3. (a) What are difference lists and how are they useful? [5 marks]

(b) A simple DCG for generating bit strings (i.e., lists of 0 and 1) is

\[
\begin{align*}
s &\rightarrow \text{[]} . \\
s &\rightarrow s, b . \\
b &\rightarrow \text{[0]} . \\
b &\rightarrow \text{[1]} .
\end{align*}
\]

For example,

\[
\begin{align*}
?\text{-} s(L, \text{[]}). \\
L = \text{[]} ? ; \\
L = \text{[0]} ? ; \\
L = \text{[1]} ? ; \\
L = \text{[0,0]} ? ; \\
L = \text{[0,1]} ? ; \\
L = \text{[1,0]} ? ; \\
L = \text{[1,1]} ? ; \\
L = \text{[0,0,0]} ? ; \\
\ldots
\end{align*}
\]

But what is problematic about the query

\[
?\text{-} s(\text{[a]}, \text{[]}). 
\]

and how can we fix the DCG above so that the Prolog interpreter answers no to this query. [10 marks]

(c) Write out the DCG given in part (b) as ordinary Prolog clauses, making the difference lists explicit. [5 marks]

(d) Write a DCG for a predicate \( s \) that outputs lists of the form

\[
[b_1, b_2, \ldots, b_n, k]
\]

for every integer \( n \geq 0 \) such that each \( b_i \) is either 0 or 1 for \( 1 \leq i \leq n \), and \( k \) is the integer that the bitstring \( b_1 b_2 \ldots b_n \) encodes.
For example,

?- s([1,1,0,X],[]).
X=6 ? ;
no

?- s([1,B,0,4],[]).
B=0 ? ;
no

since 110 is the binary encoding of 6, and 100 is the binary encoding of 4.

(e) Three students, no two of the same age, play three different sports, and come from three different counties.  

In alphabetic order (which may or may not correspond with age),

(i) the students are: chris, pat, sandy  
(ii) the sports are: boxing, football, tennis  
(iii) the counties are: cork, dublin, mayo.

Write a DCG that outputs strings of the form

[Stu1, Spo1, Cou1, Stu2, Spo2, Cou2, Stu3, Spo3, Cou3]  

(of length 9) representing all the possibilities such that

(i) Stui plays Spoi and comes from Coui for 1 ≤ i ≤ 3, and
(ii) Stui is older than Stuj for 1 ≤ i < j ≤ 3.

For example,

?- s([chris,football,mayo, pat,PatSport,cork, Stu,boxing,Cou],[]).
PatSport = tennis,
Stu = sandy,
Cou = dublin ? ;
no.

[15 marks]