CSU34011 Symbolic Programming

Second of Two Assessed Assignments

Submit to Blackboard by Thu, Nov 16 (23:59)

Problem 1 (20 points) Write a DCG that accepts strings of the form $u0v$ where u and v are strings over the alphabet $\{1, 2, 3\}$ such that u is v in reverse. For example,

```
?- s([2,1,3,1,0|L],[]).
L = [1,3,1,2] ;
false
```

Problem 2 (20 points) Exercise 6.6 in Learn Prolog Now describes a street with

(∗) three neighbouring houses that all have a different colour, namely red, blue, and green. People of different nationalities live in the different houses and they all have a different pet.

Leaving out all the other constraints mentioned in that exercise, write a DCG that outputs strings

```
[h(Col1,Nat1,Pet1), h(Col2,Nat2,Pet2), h(Col3,Nat3,Pet3)]
```

satisfying (∗), where the nationalities are

`english, spanish, japanese`

and the pets are

`jaguar, snail, zebra`.

To avoid confusion with the first problem, use different binary predicates for the difference lists, and, in particular, `nbd/2` for the 3 houses. For example,

```
?- nbd([h(red,english,snail), h(blue,japanese,jaguar),
        h(green,spanish,Z)], []).
Z = zebra ;
false.
```

Problem 3 (20 points) The nth *Fibonacci number* F_n is, for any integer $n \geq 0$, defined by

$$F_0 := 0$$
$$F_1 := 1$$
$$F_{n+2} := F_n + F_{n+1}$$

giving $F_2 = 1,\ F_3 = 2,\ F_4 = 3,\ F_5 = 5$, etc. Define a DCG that generates for every $n \geq 1$, lists $[F_0, F_1, \ldots, F_n]$ so that, for example,

```
?- fib(L,[]).
L = [0,1] ;
L = [0,1,1] ;
L = [0,1,1,2] ;
L = [0,1,1,2,3] ;
L = [0,1,1,2,3,5] ;
...
```

Problem 4 (40 points) The regular expression

$$(0+1)^* \, 1(0+1)(0+1)$$

denotes the set

$$L_3 := \{s \in \{0,1\}^* \mid s \text{ has length} \geq 3 \text{ and its third to the last bit is } 1\}$$

of bitstrings that end with one of the four strings 100, 101, 110, 111 from $1(0+1)(0+1)$. Recall from lecture that the predicate `accept/1` defined below is true of strings accepted by a finite automaton with transitions given by `tran/3` and final states given by `final/1`.

```
accept(L) :- steps(q0,L,F), final(F).
steps(Q,[],Q).
steps(Q,[H|T],Q2) :- tran(Q,H,Qn), steps(Qn,T,Q2).
```

Define the predicates `tran` and `final` to accept precisely the strings in L_3 so that, for example,

```
?- accept([0,0,Z,0,0]).
Z = 1 ;
false.
```

Turn your transitions into a DCG for L_3 so that, for example,

```
?- q0([0,0,Z,0,0],[]).
Z = 1 ;
false.
```

Finally, define a predicate `l3(String,Numeral)` that holds if `String` belongs to L_3 and has length `Numeral` and `numeral(Numeral)`, where

```
numeral(0).
numeral(succ(X)) :- numeral(X).
```

For example,

```
?- l3(String, succ(0)).
false.

?- l3(String, succ(succ(succ(succ(0))))).
String = [0, 1, 0, 0] ;
String = [0, 1, 0, 1] ;
String = [0, 1, 1, 0] ;
String = [0, 1, 1, 1] ;
String = [1, 1, 0, 0] ;
String = [1, 1, 0, 1] ;
String = [1, 1, 1, 0] ;
String = [1, 1, 1, 1] ;
false.
```