CS 538

Final Exam

Monday, December 20, 2004
2:25 PM — 4:25 PM
103 Psychology

Instructions

Answer any four questions. (If you answer more, only the first four will count.) Point values are as indicated. Please try to make your answers neat and coherent. Remember, if we can't read it, it's wrong. Partial credit will be given, so try to put something down for each question (a blank answer always gets 0 points!).

1. (a) (9 points)
Write facts and rules that define a Prolog relation `notIn(V,L)`. `V` is an atomic value and `L` is a list. The relation is satisfied (answers yes) if `V` is not a member of `L`. Thus

```
notIn(1,[]) ⇒ yes
notIn(1,[3,2,1]) ⇒ no
notIn(1,[3,2]) ⇒ yes
```

(b) (8 points)
Explain how Prolog, using your definition of `notIn`, will solve the query

```
notIn(1,[2,1]).
```

(c) (8 points)
What will happen if, using your definition of `notIn`, we pose the following query (where `X` is a free variable)

```
notIn(X,[2,1]).
```

2. (a) (15 points)
What is the type of the following ML function? How did you infer the type you selected?

```
fun xx f []  []   = []
 |  xx f [a] [b]  = [f(a,b)]
 |  xx f (i::t) (j::s) = f(i,j) :: (xx f t s);
```

(b) (10 points)
What does the ML function `f`, defined below, compute? (`g` is used as a subroutine).

```
fun g [x] = []
 |  g(h::t) = h::g t;

fun f [x] = x
 |  f(h::t) = f(g t);
```

3. (25 points)
A well-known children's game is "tic-tac-toe." A three by three grid contains x's and o's. Three x's is a row, horizontally, vertically, or diagonally is a winner. Assume we represent, in Prolog, a tac-tac-toe board by a list containing three sublists. Each sublist contains 3 elements, which can be an x, an o, or a b (representing a blank position). Thus the grid

x		o
o	x	x
o		x

would be represented as `[[x,b,o],[o,x,x],[o,b,x]]`.

Write Prolog rules that define the relation `winner(L)`. `L` is a list of lists representing a tic-tac-toe board as defined above. Given that `L` is ground (already bound to a value), `winner` should succeed if `L` represents a winning position for `x` *or* if `L` can be transformed into a winning position for `x` by transforming a single `b` into an `x`. That is, `winner` should recognize boards that `x` has already won or can win on his next move.

4. (25 points)
A *deque* is a double-ended queue; that is, a queue that allows elements to be added or removed on either end. Define an ML abstract type definition (an `abstype`) for a polymorphic deque. It should provide the following:

(i) `null`
A null deque containing no elements.

(ii) `empty(d)`
A boolean function that tests whether deque `d` is empty.

(iii) `enterleft(v,d)`
Return a deque with a new element `v` added to the left end of deque `d`.

(iv) `enterright(v,d)`
Return a deque with a new element `v` added to the right end of deque `d`.

(v) `rmleft(d)`
Return a deque with the leftmost element of deque `d` removed.

(vi) `rmright(d)`
Return a deque with the rightmost element of deque `d` removed.

(vii) `left(d)`
Return the leftmost element of deque `d`.

(viii) `right(d)`
Return the rightmost element of deque `d`.

5. (a) (18 points)
Let `L` be a list of distinct integers. Write a Python function `perm(L)` that computes a list of sublits. Each sublist is a different permutation of the values in `L`. For example,

```
perm([]) ⇒ [[]]
perm([1]) ⇒ [[1]]
perm([1,2]) ⇒ [[1, 2], [2, 1]]
```

(b) (7 points)
If list `L` contains duplicate values, the output of `perm` will contain duplicate sublists. Create a version of `perm`, `perm1`, that produces only one copy of each permutation. For example,

```
perm1([1,2,1]) ⇒ [[1, 2, 1], [2, 1, 1], [1, 1, 2]]
```

6. What do each of the following Python program fragments compute? In each case explain why.

(a) (6 points)

```
L=[3,2,1]
M=L*2
   for i in L:
      M = M[1:i]+M[0:-i]
print M
```

(b) (6 points)

```
def g(a=3,b=2,c=1):
   return a+b-c
print g(c=g(), a=g(1), b=g(2,1))
```

(c) (6 points)

```
ff = map((lambda x: (lambda y:(y - x))),[1,2,3])
   for f in ff:
      print f(10),
```

(d) (7 points)

```
L1=FL=range(2,25)
for i in L1:
   L2=FL;FL=()
   for j in L2:
      if i==j or j%i:
         FL=FL+(j,)
print FL
```