CS 337 **Quiz 2** **11/3/04**

Open book and notes.

Max points = 50 Time = 50 min Do all questions.

1. (Recursive Programming; 28 points) **Every function that you are asked to write should be complete by itself, i.e., do not use functions from the Haskell prelude, such as map, foldr, etc.**

   (a) (10 points) Define function `tie` that takes a pair of lists of equal lengths as argument and returns a list of pairs of corresponding elements. So,

   ```
   tie ([1,2,3], ['a','b','c']) = [(1,'a'), (2,'b'), (3,'c')]
   ```

   Define function `untie` that is the inverse of `tie`:

   ```
   untie (tie (xs,ys)) = (xs,ys)
   ```

   (b) (10 points) Define function *pos* which takes an integer n, $n \geq 0$, as argument. It produces a list of the first n integers. So,

   ```
   pos 0 = []
   pos 4 = [1,2,3,4]
   ```

   Note: Do not use `++` in your definition. Your program should take only linear time to implement.

   Hint: Define a more general function with two arguments.

   (c) (8 points) Function *check* accepts three arguments: (1) function f, (2) function g, (3) list zs. Functions f and g applied to zs each produce a list of integers; the output list lengths may be different for f and g. Function *check* determines if $(f\ zs) \succeq (g\ zs)$. Definition of $\succeq$ over lists of integers is:

   $$\begin{array}{ll} X \succeq [\,], & \text{where } X \text{ is any list of integers,} \\ [\,] \not\succeq X, & \text{where } X \text{ is a non-empty list of integers,} \\ ((x : xs) \succeq (y : ys)) & \equiv\ (x \geq y\ \wedge\ xs \succeq ys) \end{array}$$

   Define *check*.

2. (Finite State Machines; 22 points)

   (a) (8 points) Draw the diagram of a finite state machine which accepts a binary string if it has even number of zeroes and it ends in "11".

   Hint: Draw two machines and combine them to create a machine which accepts when both machines do.

   (b) (10 points) A block in a binary string is a substring consisting of the same symbol. So, 011100 has two 0 blocks and one 1 block.

   The following machine accepts binary strings which have equal number 0 and 1 blocks. The machine accepts 0111001, for example.

We would like to prove that the given machine meets the specification. But you will only do a part of the proving.

Annotate the states with appropriate predicates. Use symbols (such as p and q which I have used in the Notes, Page 69) for the predicates and explain their meanings.

Specify what needs to be proved for the transitions incoming and outgoing from state E. You don't have to specify for *all* transitions in the machine.

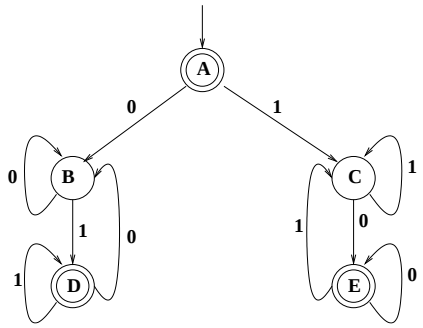

Figure 1: Machine to accept binary strings with equal number of blocks

(c) (4 points) Write a regular expression for the set of binary strings which start with a 0 and end with a 1. Examples are 01, 0011, 010101.