CS 181u Applied Logic HW 4
Due Wednesday March 11, 2020

Problem 1. Write CTL formulas that correspond to the following properties:

a. It is possible for p and q to hold at the same time.

b. From each reachable state it is possible to reach a state where p holds.

c. Whenever p holds, eventually q will hold.

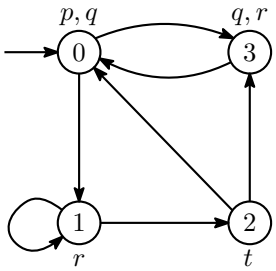
d. Whenever p holds, eventually r will hold and q will hold until r holds.

Problem 2. Show that each of the CTL operators, EG, AF, EF, EU, and AU, can be written by using only itself, EX, AX, and Boolean operations.

Problem 3. For each pair of LTL formula and CTL formula, give a transition system that satisfies one formula but not the other, and indicate which formula it satisfies.

a. $FG\ p$ and $AF\ AG\ p$ b. $FG\ p$ and $EF\ EG\ p$ c. $GF\ p$ and $AG\ EF\ p$

Problem 4. For the transition system, $\mathcal{M}$, write a νSMV model that captures the behavior of the $\mathcal{M}$. Then, for each of the following CTL formulas, ϕ, given below, use νSMV to (1) determine if $\mathcal{M} \models \phi$ and (2) exhibit a path that satisfies ϕ or show that none exist.



a. $AF\ q$

b. $AG\ (EG\ (p \vee r))$

c. $EX\ (EX\ r)$

d. $AG\ AF\ q$

e. $p\ AU\ q$

Problem 5. Consider the νSMV stack model that we covered in class. The code is given on the next page for your convenience, and available on the course Piazza Resources page as `stack.smv`. You may refer to the Lecture 7 slides for further explanation of the stack model.

First, download `stack.smv`. Then, for each of the following verification conditions, write a νSMV specification, add the specification to the provided `stack.smv` file, and confirm that running νSMV verifies the property.

i. LTL property: The stack pointer is never negative and never larger than the size of the stack buffer.

ii. LTL property: The stack is never simultaneously full and empty.

iii. CTL property: In all possible states, it is always the case that if the stack happens to be empty, then it is possible that the stack eventually becomes full.

iv. LTL property: If the user only ever pushes then the stack eventually fills up.

v. CTL property: At any time, if the user pops when the stack does not have any elements in it then the resulting pop value will be `NULL`.

vi. LTL property: At any time, if the user pops when the stack has at least one value in it, then the resulting pop value will not be `NULL`.

vii. LTL property: It is always the case that if x is pushed on the stack and the stack is not full, then if in the next state the user decides to pop then in the state after that the popped value is x.

viii. CTL property: Starting from any point in time, it is possible that the stack will eventually hold three items.

ix. CTL property: If the stack pointer is 1 and then pop is executed twice in a row, then the stack will be empty.

x. LTL property: If the user pushes an x followed directly by a y, and the stack is not full during either of those push actions, then if the user immediately pops, the resulting value will be y and if the user immediately pops again, the resulting value will be x.

```
#define SIZE 5

MODULE stack(action, push_val, pop_val)
  VAR
    top : 0 .. SIZE;
    buffer : array 0 .. SIZE - 1 of {x, y, z};

  DEFINE
    full := top = SIZE;
    empty := top = 0;

  ASSIGN
    init(top) := 0;

    next(top) :=
      case
        (action = push) & (top < SIZE)  : top + 1;
        (action = pop) & (top > 0) : top - 1;
        TRUE : top;
      esac;

    next(pop_val) :=
      case
        action = pop & !empty : buffer[top - 1];
        TRUE : NULL;
      esac;

    next(buffer[0]) := top = 0 & action = push ? push_val : buffer[0];
    next(buffer[1]) := top = 1 & action = push ? push_val : buffer[1];
    next(buffer[2]) := top = 2 & action = push ? push_val : buffer[2];
    next(buffer[3]) := top = 3 & action = push ? push_val : buffer[3];
    next(buffer[4]) := top = 4 & action = push ? push_val : buffer[4];

MODULE main
  VAR
    pop_val : {NULL, x, y, z};
    push_val : {x,y,z};
    action : {push, pop};
    s : stack(action, push_val, pop_val);
```