CS 181u Applied Logic HW 3
Due Wednesday Feb 26, 2020

Problem 1. Consider the following transition system, $\mathcal{M} = (S, \rightarrow, L)$, where

- $S = \{0, 1, 2, 3, 4\}$ is the set of states,
- $I = \{0, 1\}$ is the set of initial states,
- $\rightarrow = \{(0,2), (0,3), (1,3), (2,3), (3,1), (3,2), (3,4), (4,3), (4,4)\}$ defines the transition relation,
- $AP = \{p, q, r\}$ is the set of atomic propositions, and
- $L : S \rightarrow 2^{AP}$ is the labeling function with
  - $L(0) = \{p\}$
  - $L(1) = \{r\}$
  - $L(2) = \{q, r\}$
  - $L(3) = \{q\}$
  - $L(4) = \{p, q, r\}$

a. Draw the transition diagram for $\mathcal{M}$.

b. For each of the following LTL formulas, ϕ, determine if $\mathcal{M} \models \phi$. If $\mathcal{M} \not\models \phi$, give a counterexample execution path.

i. Xq
ii. $G(p \vee q)$
iii. Fr
iv. pUr
v. FGr
vi. GFr
vii. $X(\neg r) \Rightarrow XXr$
viii. $G(q \vee Xq)$
ix. $pU(G(q \vee Xq))$
x. $(XXq)U(q \vee r)$

Problem 2. Assume that p, q and r are atomic properties. Write LTL formulas that correspond to the following statments.

a. Whenever p holds, q must hold in the next state.

b. q holds infinitely many times.

c. p holds only a finite number of times.

d. If p holds sometime, then q must hold at least once before it.

e. p and q become true in strictly alternating fashion.

Problem 3. Consider the two processed defined below where `a` and `b` are shared Boolean variables and initially `a = FALSE` and `b = FALSE`. The assignments are atomic and each process loops between two program points (1 and 2).

```
P0 :: while(TRUE){
1:      a := true;
2:      b := !a;
3:    }
```

```
P1 :: while(TRUE){
1:      b := true;
2:      a := !b;
3:    }
```

A. Define the composed state of the two concurrently running programs, $P_0||P_1$, to be the tuple (pc_0, pc_1, a, b) where $pc_i \in \{1, 2\}$ is the program counter of P_i and $a, b \in \{T, F\}$. Draw the transition system $\mathcal{M}$ for $P_0||P_1$.

(Hint: There are 12 states. The initial state should be $(1, 1, F, F)$.)

B. For each of the following LTL formulas, ϕ, determine if $\mathcal{M} \models \phi$.

If $\mathcal{M} \not\models \phi$, give a counterexample execution path.

i. $G(\neg a \wedge \neg b)$

ii. $F(a = b)$

iii. $F(a \neq b)$

iv. $G(pc_0 = 2 \wedge pc_1 = 2 \Rightarrow X(a \neq b))$

v. $GF(a = b)$

vi. $GF(a \neq b)$

Problem 4. Consider Dekker's two process mutual exclusion algorithm, shown below. Each relevant line is labeled with a letter and number, where `n` indicates non-critical section, `w` indicates waiting to enter the critical section, `c` indicates critical section code, and `e` indicates exiting the critical section. Each process has an `id`, is able to look at the `other` process, and has a variable called `turn`, indicating which process is allowed to use the critical section if both processes concurrently want to enter. Each process also has a `flag` which is used to signal its status. Initially, `flag` is `FALSE` for both processes.

```
         proc(id, other, turn){
           while(TRUE){
n1:           flag := TRUE;
w2:           while(other.flag){
w3:            if(turn != id){
w4:               flag := FALSE;
w5:               wait until(turn = id)
w6:               flag := TRUE;
w7:             }
w8:           }
c9:           turn = other.id;
e10:          flag := FALSE;
       }
     }
```

a. Write a νSMV model for Dekker's two-process mutual exclusion algorithm with process `id`'s 0 and 1. Remember to include the `FAIRNESS running` constraint.

b. Write a νSMV LTL specification for the mutual exclusion (safety) property that the two processes are never in the critical section at the same time. Verify this property with NuSMV.

c. Write a νSMV LTL specification for the starvation freedom property: if either process is waiting to enter the critical section then it will eventually enter the critical section. Verify this property with NuSMV.

d. Write a νSMV LTL specification for the liveness property that if neither process is in the critical section, then eventually one of the processes will enter the critical section. Verify this property with NuSMV.