CSE 4/586: Project 2

Murat Demirbas

[2015-11-24 Tue]

1 Dijkstra's 4-state token ring program

Consider Dijkstra's 4-state token ring algorithm described in the class. There are N+1 nodes numbered 0..N. Each node has two boolean variables, *up* and *c*. By definition, 0.up=True & N.up=False always hold. [1]

At process 0:
0.c = 1.c $\wedge$ $\neg$ up.1 $\longrightarrow$ 0.c:=$\neg$ 0.c

At process N:
N.c $\neq$ (N-1).c $\longrightarrow$ N.c := (N-1).c

At process j, j $\neq$ 0 $\wedge$ j $\neq$ N:
j.c $\neq$ (j-1).c $\longrightarrow$ j.c := (j-1).c; j.up:=True
j.c = (j+1).c $\wedge$ j.up $\wedge$ $\neg$ (j+1).up $\longrightarrow$ j.up:=False

1.1 Write a Pluscal program to represent this algorithm (20 points)

Write the algorithm in Pluscal and let TLA+ tool to translate this to TLA+. (You can study and learn from the ProcessTokenRing.tla for Dijkstra's classic token ring algorithm. See the Piazza post.)

1.2 Model check for invariant properties (10 points)

Provide a tight invariant property "InvProp== ???" for the program and model check the invariant.

[1] The original description of the algorithm appears here `https://www.cs.utexas.edu/users/EWD/ewd04xx/EWD426.PDF`

1.3 Model check for stabilization (10 points)

Model check the program for stabilizing to the invariant. For this you can define the "Stabilization == $\Diamond$ InvProp" temporal property and add it in the "Properties" section of TLA+ model. In order to simulate arbitrarily corrupted initial state of the program, you can initialize the variables arbitrarily. [2]

1.4 Determine a suitable variant function to prove stabilization (10 points)

Provide a suitable variant property "sVariant==???" for the program and use model checking to show these properties:

- It never increases,
- It always eventually, $\Box\Diamond$, decreases (unless it hits the lowerbound)

[2]While model-checking for stabilization, you should uncheck the "InvProp" from the Invariants of TLA+ model, otherwise you will get an invariant violation error, duh!

2 Dijkstra's 3-state token ring program

Consider Dijkstra's 3-state token ring algorithm described in the class. There are N+1 nodes numbered 0..N. Each node has a counter c with three possible values: 0,1,2. Let a $+_3$ 1 $\equiv$ (a + 1) modulo 3

At process 0:
0.c $+_3$ 1 = 1.c $\longrightarrow$ 0.c:= 1.c $+_3$ 1

At process N:
(N-1).c= 0.c $\wedge$ N.c$\neq$ (N-1).c $+_3$ 1 $\longrightarrow$ N.c:= (N-1).c $+_3$ 1

At process j, j $\neq$ 0 $\wedge$ j $\neq$ N:
j.c (+) 1 = (j-1).c $\longrightarrow$ j.c := (j-1).c

j.c (+) 1 = (j+1).c $\longrightarrow$ j.c := (j+1).c

2.1 Write a Pluscal program to represent this algorithm (20 points)

Write the algorithm in Pluscal and let TLA+ tool to translate this to TLA+.

2.2 Model check for invariant properties (10 points)

Provide a tight invariant property "InvProp== ???" for the program and model check the invariant.

2.3 Model check for stabilization (10 points)

Model check the program for stabilizing to the invariant. For this you can define the "Stabilization == $\diamondsuit$ InvProp" temporal property and add it in the "Properties" section of TLA+ model. In order to simulate arbitrarily corrupted initial state of the program, you can initialize the variables arbitrarily. [3]

2.4 Determine a suitable variant function to prove stabilization (10 points)

Provide a suitable variant property "sVariant==???" for the program and use model checking to show the three properties: It never increases, it always eventually, $\Box\diamondsuit$, decreases (unless it hits the lowerbound).

[3] While model-checking for stabilization, you should uncheck the "InvProp" from the Invariants of TLA+ model, otherwise you will get an invariant violation error, duh!

3 Submission

Your TLA+ files should be named "4state.tla" and "3state.tla". Your model's name should be the default name $Model\_1$ (do not name your model file differently).

Generate a pdf print of your two TLA+ programs using the "Produce Pdf version" from the TLA+ menu. (This will get included in your submission automatically.)

Now create a zip file from the two ".tla" files and the corresponding ".toolbox" directories. **Name the zipfile as: proj2.zip**

Not following these directions will cause you to lose points.

You will use the submit command ($submit\_cse486$ or $submit\_cse586$ respectively) to submit your work. The submit command instructions are here: `https://wiki.cse.buffalo.edu/services/content/submit-script`