Worksheet 7: UPPAAL, Heart Model + Pacemaker + Model Checking

Due Date: 11/18/2019

Principles of Modeling for Cyber-Physical Systems
Madhur Behl | madhur.behl@virginia.edu

In this worksheet you will be using UPPAAL to model, simulate and model-check timed automata models of the heart and pacemaker.

For the worksheet problems give a good deal of explanations: why did you make some modeling choices? Why does your model lead to the desired effect? How do you interpret a given counter-example? It is not enough to demonstrate a correct `result', like a successful execution of the tool that yields the expected answer. It means very little if you follow some steps mechanically and get an expected result. You must demonstrate *understanding*, and you do so by explaining your reasoning.

A skeleton code with comments for the `System declarations' has been provided for you within HM1.xml. Fill out each section with the proper declarations as you progress from part to part, making each part as self-contained as possible. In other words, structure the declarations such that you comment out sections other than the one you are working on.

Part 1: Simulating heart conditions [35pts]

Load HM1.xml. Simulate the system made of HRA_1, RVA_1, Cond_1 and AVN_1. Change the parameters of the heart model to model the following conditions:

1a) Sinus Bradychardia: the HRA_1 automaton (which models the atrium) produces an event (an A_get! message) too infrequently. You only need to edit AtrialNode for this.

1b) Sinus Tachycardia: the HRA_1 automaton (which models the atrium) produces an event (an A_get! message) too frequently. There should be a 1:1 conduction from atrium to ventricle. You only need to edit AtrialNode for this.

1c) AV Block: because the atrium is firing too fast (is in sinus tachycardia), the AVN_1 automaton (which models the AV node) will only pass 1 out of every 2 atrial events to the ventricles (which are modeled by RVA_1 automaton). You only need to edit AVNode for this.

1d) AV delay: there is additional delay in the time it takes an event to go from atrium to ventricles.

1e) Premature Ventricular complex (PVC): tissue in the ventricles fires on its own, rather than wait to be excited by an event conducted from the atria. You can model this either by adding a new VentNode automaton, or by modifying parameters.

You can also review these conditions from the earlier lectures and from Worksheet 5 in Simulink.

Your answer will be a table with one column per condition. In each condition/column, you will write:

- A complete list of parameter values from `Declarations', namely: TAVI, TLRI, TVARP, TPVAB, TVRP, TURI, thresh.
- The `System declarations. You are only allowed to change the values of the constants that you see in there. E.g. in HRA_1, you can only change the values of the last 2 arguments, which indicate Aminwait and Amaxwait in template AtrialNode.
- An explanation for your choices: how do your modifications lead to the desired effect?

To grade this problem, we will use the HM1.xml model out-of-the-box, and plug in your parameter values. We will simulate the model for 1 or 2 cycles (i.e., 1 or 2 atrial beats), and expect to see the condition. If it doesn't manifest itself in 2 beats, I consider that the model is incorrect. So simulate your model in UPPAAL and make sure you see the condition before submitting. Partial credit will be given if the model is incorrect but the reasoning behind it still makes some sense.

Part 2: Multiple conditions in a single system [15pts]

You may have noticed that when you are simulating the heart model, there is only one enabled transition at any given moment. Change the parameter(s) of the AtrialNode such that the same model can now model both Sinus Brady and Sinus Tachy. Simulate the heart model and carefully choose enabled transitions to create simulation traces for both conditions.

> Your answer should explain what parameter(s) you changed and what you changed them to. You will also describe which transitions you now have to choose from, and why choosing one or the other gives you a different condition. Include a snapshot of the UPPAAL gui for each condition: the snapshot will show the choice between transitions, and the resulting events on the sequence chart.

Part 3: Creating a pacemaker and model checking [50 pts]

You will now create the last missing piece of a basic dual-chamber pacemaker, to handle some of the heart conditions that you have previously modeled.

3.1 One of the pacemaker timers we saw was the Ventricular Refractory Period (VRP). What is the purpose of the VRP?

3.2 Create a template VRP1 which models the Ventricular Refractory Period.

3.2.1 Call the initial state Idle. Declare a local clock t.

3.2.2 Starting from the Idle location, it listens for Vget?, V_act? and VP? If it receives Vget? or V_act? it transitions to a VRP location and issues a VS! action. If it receives VP, it transitions to VRP location without issuing anything.

3.2.3 In VRP location, it will wait until the local timer, starting at 0, exceeds the parameter value TVRP, then goes back to Idle.

3.3 Why does VRP issue VS! only when it receives Vget? or V_act?

3.4 Instantiate LRI, VRP and AVI. Call the instances pmLRI, pmVRP and pmAVI respectively. Add these instances to the Bradycardia heart model you created earlier in 1a. You now have a heart connected to a pacemaker. Verify if the following LRI property is satisfied by this closed loop:

> *For all paths, Always, if the monitor indicates two successive V events, then the time between them is less than LRI*

- Submit a screenshot of your simulator tab with the complete system (as required by 3.4)
- Submit a screenshot of the verifier after running the query.
- Submit a new file HM1_PM.xml which contains your VRP template.
- Submit a screenshot of your VRP template