MCS Rodin project

Due before Friday May 29th 23:59 on Toledo. Make sure you understand the exercises of session 8 and 9.

1 Introduction

For this project you will have to model a train system very much alike the train problem in the IDP project. However, some details have changed to make the exercise appropriate for Event-B. The project must be completed individually and the same remarks hold as for the IDP project.

2 Modelling the train system in Event-B

You are going to make two contexts and four machines from scratch. Start by making a new Event-B project in Rodin named `LastName_FirstName_StudentNumber`

Context tracks Create a context called `tracks` and create two enumerated sets called `stations` and `trains` containing the elements as in the tracks example in the IDP project together with the element `no_train`. This means that trains is equal to {no_train, IC1, IC2, IC3, IC4, L1, L2} and stations is equal to {A, B, C, D, E, F}. Besides the two enumerated sets, add a binary relation `tracks` between stations.

Machine machine0 Create a machine `machine0` that uses the context `tracks`. Add a variable `current_station`, which is a station (so don't forget the invariant). Initialise `current_station` with a random station (check the Event-B cheat sheet). Add a new event `go_to_next` that takes a station as paramater and which sets the current station to this station if there is a direct track from the current station to this station.

Machine machine1 Refine `machine0` to a machine `machine1`. Add a boolean variable `in_train` that is initially false. Add two events `hop_on` and `hop_off` that change the variable `in_train` appropiately. Of course you can only hop on a train if you're not already in a train. Likewise for the hop off event. Update the `go_next_to` event so that you can only go to the next station if you are in a train.

Machine machine2 Refine `machine1` to a machine `machine2`. Add a variable `current_train` that refines `in_train`. `current_train` is an element of the set of trains and is initialised to `no_train`, which designates that you're not in a train. Remove the variable `in_train` from `machine2`. Do not forget to add the gluing invariant.

Refine the `hop_on` event to have a train as parameter and update the hop on and off events accordingly. Also update the `go_to_next` event.

Context trains Extend the context `tracks` to a context `trains`. Add two constants `next_pass` and `stops` and appropiate axioms described below. `next_pass` is a partial function[1] from $T \times$ `stations` to `stations` where T is the set of trains without the `no_train` element, and where $\times$ denotes the cartesian product[2], i.e. tuples. `stops` is a total function from T to the set of non-empty subsets of `stations`.

Additionally, we have the following axioms.

- Any two stations that have the same next pass for the train t, are equal.
- For each train $t \in T$, there is exactly one station such that it is a first pass of t. A station s is a first pass of t if (t, s) is in the domain of `next_pass` and there is no station s_2 such that `next_pass`$(t, s_2) = s$.
- For any two consecutive passes, there is a direct track between them.
- Any stop of t is also a pass of t.

Machine machine3 Refine `machine2` to a machine `machine3` that uses context `trains`. Update the `go_to_next` event so that it goes to the next pass from the current station, i.e. does not have a parameter anymore. However, make sure that it cannot be activated when the current station is the last pass of the current train. Do not forget the witness for the removed parameter. Update the hop on and off events so that you can only hop on or off at stops of the current train.

Normally, all proof obligations are satisfied.

2.1 Perform LTL/CTL checking on the `Machine3` machine using ProB.

To do LTL/CTL checking on the `Machine3` machine follow the following steps:

1. Right-click the `Machine3` machine and under the `Prob Classic...` submenu, select `Open in Prob classic`. A prompt will appear saying the file exists and ask whether you want to overwrite it. Select Yes.

[1] The set of partial functions from A to B is denoted in event-be as `+->`. For a partial function `f`, `dom(f)` is the set of elements in which `f` is defined.

[2] Cartesian product in event-B is `**`

- If, you get a prompt saying ProB can not be found, go to the windows/preferences menu and under ProB/Prob Classic, select the path to your ProB installation. This is not the ProB plugin, but the actual application, as used in the exercise session on ProB.

2. Your machine will open in ProB in read-only mode, allowing you to specify and check LTL and CTL formulas as seen in the exercise sessions on ProB.
3. Use LTL and/or CTL to specify the following statements.

LTL/CTL

Check the following statements on your machine using LTL or CTL statements. They do not need to be true. **Add these statements to your report so we can verify them!** Remember that B syntax can be used in {curly brackets} in an LTL and CTL formula. The occurrence of an event is expressed as [E] for event E. Use the `ProB Classic ... - Open In ProB Classic` functionality in Rodin to open your final machine in ProB.

- It is always possible to arrive in a situation where you're inside a train, where you cannot hop off and cannot go to a next station.
- Once you are in a train, you keep driving until you hop off.
- Eventually you only hop on or off a train.
- When you are in station A, you will never reach station F unless you pass station C.

3 Practical

The output of this part of the project consists of

1. A *concise*(at most 1 page) report in which you discuss design decisions.
   - The report should contain the time you spent on this part of the project. The expected time needed is 8 hours, this depends on your preparation during the exercise sessions.
2. A zip-file `LastName_FirstName_StudentNumber.zip` containing all files in your Rodin project produced as explained below.
3. Use `http://wiki.event-b.org/images/EventB-Summary.pdf` for a cheat sheet on the event-B syntax.

3.1 Exporting your project

To export your project from Rodin, follow the following steps:

1. Right-click your project and select `Export...`
2. Select `Archive File`

3. Select `Create only selected directories`
4. Select **only** your project on the left.
5. Inspect the zip file for yourself, that everything is there.

You are allowed to discuss this project with other people, but are not allowed to copy any code from others. This is not an open source project, i.e., you are also not allowed to put your code openly available on the web. If you want to use git, do not use public GitHub repositories (we will find them).

For any questions, do not hesitate to contact `simon.marynissen@kuleuven.be`.

This project is again made individually. This project is expected to be completable in 8 hours or less. The deadline for submitting your solution on Toledo is the 29th of May 2020, 23.59.

Good luck!