

Assignment 4

Evacuation Analysis of Bmod

Cláudio Gomes
claudio.gomes@uantwerp.be

November 21, 2019

1 Practical Information

The goal of this assignment is to implement a translation that takes a Bmod floor plan model, together with a specification of dangerous properties, and outputs a petri net model, that represents the movement of people in case of an emergency.

Please download the `generate_pn.egl` from here: http://msdl.cs.mcgill.ca/people/hv/teaching/MSBDesign/assignments/generate_pn.zip

Pipe2 (v2.5) is available from here: <https://sourceforge.net/projects/pipe2/files/Pipe%20/PIPE2%20V2.5/>. It is ok to use a different version, but if so, please state it in the report and provide a download link for it.

1.1 Task Overview

Task 1 *Extend petri-net formalism with Inhibitor Arcs.*

Task 2 *Implement a model transformation that creates a petri net model representing the movement of people in an emergency situation, and the dangerous conditions evaluation.*

Task 3 *Extend the `generate_pn.egl` script to generate inhibitor arcs.*

Task 4 *Analyse a non-trivial building floor model.*

Task 5 *Write a report.*

1.2 Deadline, Logistics, and Plagiarism

Complete this assignment in **groups of 2**.

One, and only one, person in the group must submit the solution on blackboard before the deadline announced on the course web page: <http://msdl.cs.mcgill.ca/people/hv/teaching/MSBDesign/>.

Discussion on the following topics with your classmates *is encouraged*:

- Interpretation of the assignment requirements;
- Technical difficulties with the tools;

Discussion on the main concepts of the domain that may lead to similar solutions is *discouraged*. Creating a domain model is a highly creative process, and therefore it is unlikely that two groups will have a similar solution.

Contact Cláudio Gomes (claudio.gomes@uantwerp.be) if you have questions and/or need some help.

2 Requirements

2.1 Task 1

Extend the petri-net formalism with Inhibitor Arcs, with minimal modifications on the current metamodel (that is, find the simplest way to do it).

2.2 Task 2

The goal of the resulting petri-net is to evaluate an evacuation scenario, and signal whether any dangerous condition has been observed in any of these.

In order to make this assignment feasible, several simplifications will be made to the original operational semantics, described in the first assignment:

- Ignore the dynamics of fire;
- Assume that every person knows that there is a fire; and
- Assume that every person follow the experienced action profile (this is detailed better below).

The petri net should have three components:

1. The movements of people through cells, following the direction of the emergency routes;
2. The counters that allow the petri net to keep track of the dangerous conditions; and
3. The part that keeps track of the dangerous conditions.

For the movements of people, the operational semantics implemented in the previous assignments apply, except for the above assumptions. Note that the non-deterministic nature of petri-nets should make it easier to implement the movement of people, which should be non-deterministic when there are obstructions or bifurcations in the emergency route.

The counters to generate depend on the property specified by the user of the language, but each counter should be represented by a group of places and transitions that keep track of how many people there are in each room.

The dangerous condition part should use the counters part to signal when a dangerous condition occurred. Once the dangerous condition is signalled, the whole network should be deadlocked. When no property has been satisfied, the whole network should never deadlock (even when every person has reached the emergency exits). With the above requirements, the verification of properties is done by using the State Space Analysis module as follows:

- If the results state that the network is free from deadlock, it means no dangerous conditions where triggered; and
- If the results state that the network has a deadlock, the shortest path to deadlock is presented.

You will need to use inhibitor arcs.

The resulting petri-net can be exported to metaDepth using the appropriate AtomPM toolbar, which in turn is used to generate the xml representation of the petri net, to be loaded onto the tool pipe2 (see section 3).

The analysis is made with the pipe2 modules.

The resulting petri-net should always be bounded.

2.3 Task 3

Extend the `generate_pn.egl` script to generate inhibitor arcs. To know how to generate inhibitor arcs, use pipe2 to create a simple petri-net with those arcs; save the file. Then inspect the saved xml file.

2.4 Task 4

Create a floor model, with at least one bifurcation of the emergency route, model one dangerous condition, and:

1. Perform an analysis with a specific placement of people that causes that the condition happens;
2. Perform an analysis with a different placement of people that causes the condition to never happen.

2.5 Task 5

Write a report following the same requirements as Task 5 the first assignment.

Additionally:

1. Explain at least two non trivial transformation rules that you used for the generation of the petri-net;
2. Include the motif file, and explain the general approach to the transformation (e.g., informally explaining the transformation algorithm, without explaining in detail each rule);
3. Explain the floor that you decided to analyze, and show the analysis results.
4. In the case of a dangerous condition not being met, explain how to interpret the shortest path to deadlock.

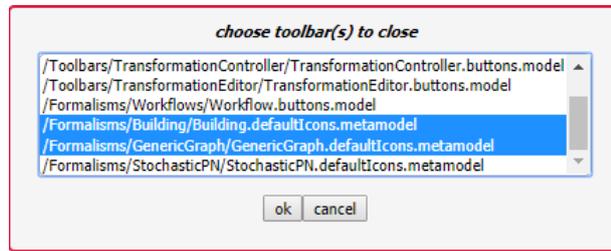
A screencast showcasing one of the analyses can optionally be included.

3 Tutorial

3.1 Exporting petri-net models from AToMPM.

After running the transformation created in Task 2, remove the traceability links and original building plant model by closing the respective toolbars:





Load the MetaDepth toolbar (inside of the */Toolbars/MetaDepth/* folder, it is called *Export.buttons.model*). This toolbar has two buttons: one for exporting models, and one for exporting metamodels.

The toolbar should look like the following:



If you open the petri-net metamodel file (called *PNMM.model*), you can export it by pressing the export metamodel button. The metamodel name should be called *PN*.

With an open petri-net model, you can export it by pressing the export model button, inserting a name of your choice. Suppose this file's name is *pnmodel*.

Both generated files (*PN.mdepth* and *pnmodel.mdepth*) will be located in *AtomPMFolder/exported_to_md/* folder.

3.2 Generating petri-net xml files

After generating the *PN.mdepth* and the *pnmodel.mdepth* files, run the provided *generate_pn.egl* script, by loading metaDepth, and running the following script:

```
load "pnmodel"
context pnmodel
load EGL "generate_pn.egl" pnmodel.xml
```

The resulting file (*pnmodel.xml*) can now be loaded with pipe2 for analysis.

4 Tips, Tricks, Pitfalls, and Issues

Bug. If you get a null pointer exception when running EGL, it may be because you forgot a “;” somewhere.