Petri Net assignment

Fall Term 2003

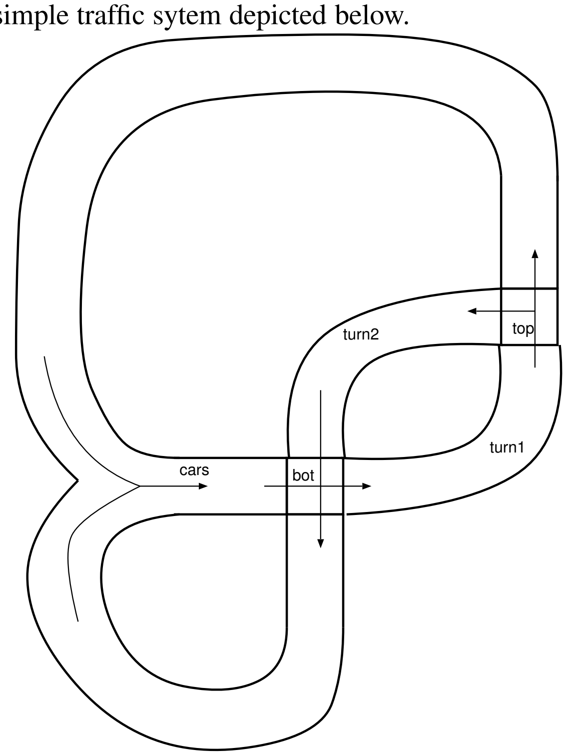
General Information

- The due date is **Wednesday 5 November 2003**, before 23:55.
- Submissions must be done via WebCT. Beware that WebCT's clock may differ slightly from yours. As described on the Assignments page, *all* results must be uploaded to WebCT and accessible from links in the index.html file. There is no need to upload AToM3.
- The assignment must be made in groups of 2 people. It is understood that all partners will understand the complete assignment (and will be able to answer questions about it).
- Grading will be done based on correctness and completeness of the solution. Do not forget to document your requirements, assumptions, design, implementation *and* modelling and simulation results in detail !
- Extensions, if given, will involve extending not only the alotted time, but also the assignment !

The assignment

You will use the meta-modelling environment AToM^3 (which you also used for the first assignment), but now load the Petri Net formalism to interactively construct a Petri Net model.

The system to model consists of a simple traffic sytem depicted below.



Initially, there are *INIT* cars in the `cars` part of the system. All other parts of the system are empty. The `cars` part actually spans the top loop (above the `top` intersection) as well as the bottom loop (below the `bot` intersection) and can hold an arbitrary number of cars. The merging of cars coming from `top` and from `bot` should not be explictly modelled.

Cars enter the centre loop of the system from the `cars` road segment and proceed through the system as indicated by the arrows. Intersections `bot` and `top` can only hold one car at a time (more than one car would mean a collision). Road segments `turn1` and `turn2` also have capacity 1. In intersection `top`, there is a choice between going straight (joining the `cars` section) or going back to intersection `bot` via `turn2`. The car/driver actually makes the decision whether to go straight or turn left while in `turn1`.

The assignment consists of three parts.

1. Using AToM^3, build and document a Petri Net model of the behaviour of this system. Documentation must describe how you've addressed the different constraints given above. Include an image of this model in your solution.

2. Perform a few simulation steps of this process if the number of cars in the system $INIT = 2$, and comment.

3. For $INIT = 2$,

   - Is the system bounded ?
   - Does deadlock occur ?
   - Is the system live ?
   - Which quantities are conserved ?

   Answer these questions by means of the reachability graph for this Petri Net. Include an image of this reachability graph in your solution.

4. From which number of cars $INIT$ onwards do you expect deadlock to occur ? For that $INIT$, using AToM^3, build the reachability graph. Include an image of this reachability graph in your solution. Show occurrence of deadlock on this graph.

Constructing Petri Net models in AToM^3

Download AToM3-2.2.tgz archive. Note how this is a new archive (not the same as the one for the first assignment). In particular, it contains the necessary directories for dealing with Petri Nets as well as Reachability Graphs.

Expand the archive locally (for example with the command `tar --ungzip -xvof AToM3-2.2.tgz` if you're working on a UNIX machine. This will create a directory `AToM3-2.2`.

To start the AToM^3 environment, `python ATOM3.py`. `python` should be at least version 2.2 of Python (this is installed in the SOCS labs). On UNIX, the script `atom3` is a shortcut for the above command. On windows, you can just click on the ATOM3 icon to launch it.

AToM^3 will be started with the PetriNets formalism loaded. You can either build your own model (and `File/Save` it) or `File/open` an existing one.

In the `PNModels/models` directory, you will find some examples.