TP1 MVFA: getting started with NuSMV

During the practical sessions, you will use a model-checker for formally verifying multi-threaded software applications: NuSMV. All the files you should download may be found at

`http://people.irisa.fr/Sophie.Pinchinat/MVFA.html`.

For the first practical session, the goal is to install NuSMV on your machines and implement some examples.

1. NuSMV can be found at `http://nusmv.fbk.eu/`.
   (a) Take some time to look around.
   (b) Download NuSMV from `http://nusmv.fbk.eu/NuSMV/download/getting_bin-v2.html` and unpack the archive.
   (c) There is a good tutorial at `http://nusmv.fbk.eu/NuSMV/tutorial/v25/tutorial.pdf` which you can look at.
2. Our first simple example is in `process.smv`. This is a simple transition system, modeling a process that receives requests which keep it busy.
   (a) Download `process.smv` to the bin directory of NuSMV and start NuSMV in a terminal with `./NuSMV -int process.smv`. This starts NuSMV in interactive mode.
   (b) In the NuSMV prompt, first type `go`, second type `pick_state -r`. This tells NuSMV to choose randomly one of the initial states[1]. Now tell NuSMV to print the current state by typing `print_current_state -v`. Do a 5-step random simulation by typing `simulate -r -k 5` and show the trace using `show_traces -v`. Then prolong your random trace by running `simulate -r -k 5` again and have another look at it with `show_traces -v`.
   (c) Experiment further more if you wish (Section 3.2 in the tutorial can give some inspiration). Use `quit` to exit the interactive NuSMV session.
3. Download the file `semaphore.smv`. There are two new keywords: `process` means that the execution is asynchronous, that is a single process is selected for progress at each step, and `FAIRNESS running` means that every process is selected infinitely often.
   (a) Look at the model `semaphore.smv` and make sure that you understand everything.
   (b) Open `semaphore.smv` in NuSMV (`./NuSMV -int semaphore.smv`) and simulate it to test if it behaves as expected (You will need at least 10 steps in your simulation to reach an interesting phenomenon).

[1] We actually have only one initial state, so this is not very random in our case.

4. We consider an extension of the terminating parallel program P of Exercise 3 in TD1: let P_1, P_2, P_3 be three instances of the same process whose codes are as follows.

for $k = 1, \ldots, 3$ **do**
  LOAD(x);
  INC(x);
  STORE(x);
end

Notice that the three processes share the common integer variable x, while the iteration variable k is local to each instance of this process.

(a) Implement processes P_1, P_2, P_3 in NuSMV; you may download the file `TP1.smv` and fill properly the `ASSIGN` part. The `pc` variable is a program counter ranging from 1 to 5.

(b) Can we have $x = 2$ at the end of the simulation? Try to answer this question by using NuSMV with the keyword `INVARSPEC` in your source file.

Send your solution with a few lines of explanations, to `victor.roussanaly@irisa.fr` with subject "[MVFA] TP1 - Names" with Names containing the names of the students involved.