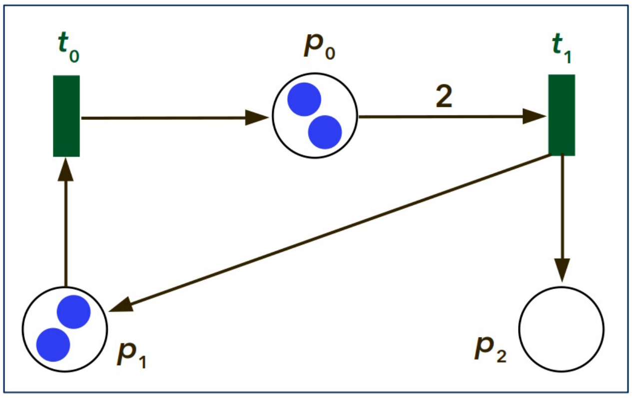
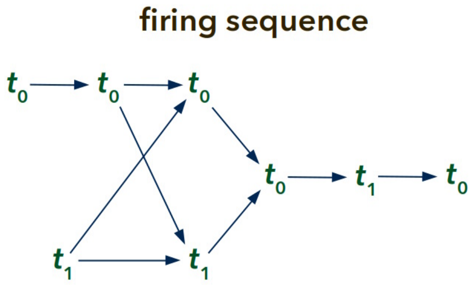
Distributed Enterprise Systems (CO3409) Lab 18: Petri Nets

You can work on this problem by pen and paper, or using PIPE, or both.

18.1 Petri net analysis

Consider the Petri net below (see also the XML input file for PIPE):

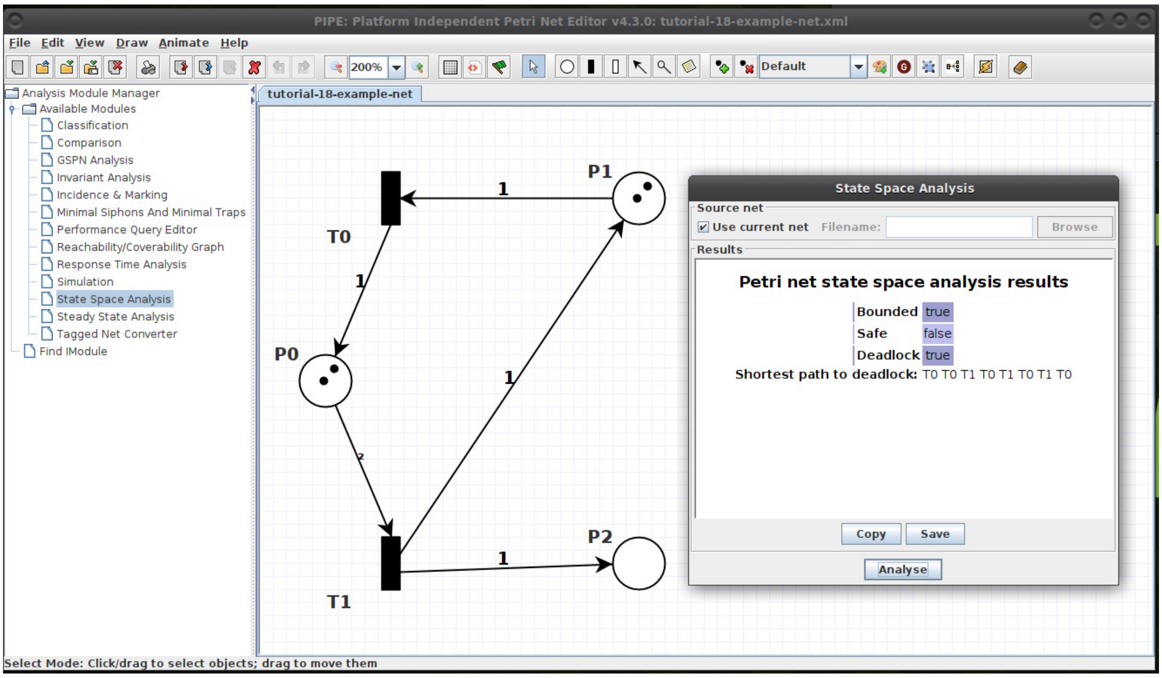


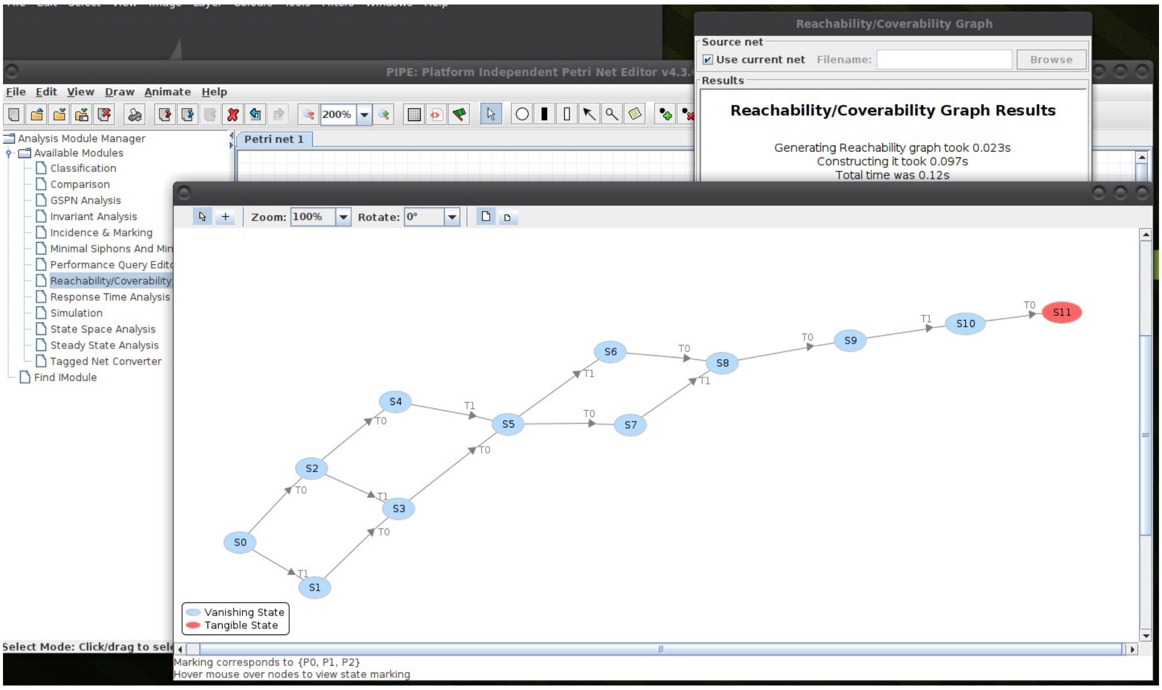


Make sure that you understand why the net exhibits the following behaviour:

a. Transitions can be fired in the following order: $t_0t_0t_1t_0t_1t_0t_1t_0$, $t_0t_0t_1t_1t_0t_0t_1t_0$, $t_0t_1t_0t_0t_1t_0t_1t_0$, $t_0t_1t_0t_1t_0t_0t_1t_0$, $t_1t_0t_0t_0t_1t_0t_1t_0$, and $t_1t_0t_0t_1t_0t_0t_1t_0$. At that point, respectively, a deadlock is reached.
b. The net is **not safe**: Places can contain more than one token.
c. Firing any of the transitions preserves the overall number of tokens (hence, the total number of tokens constitutes an **invariant** - it does not change).
d. The net is **bounded**: There is a limit (greater than one, but finite) to the number of tokens in every place.

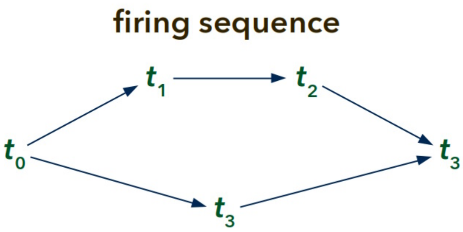
The screenshots below illustrate the analysis functionality from PIPE. However, pen and paper should be enough to establish the observations made above.





18.2 Petri net design

a. Design a Petri net with four transitions that can (only) be fired as follows until a deadlock is reached: $t_0t_1t_2t_3t_3$, $t_0t_1t_3t_2t_3$, and $t_0t_3t_1t_2t_3$.
b. Is your net safe? Is it bounded?



If you are interested in feedback on your work, send an email to [Aaron Bryant and Martin Horsch].