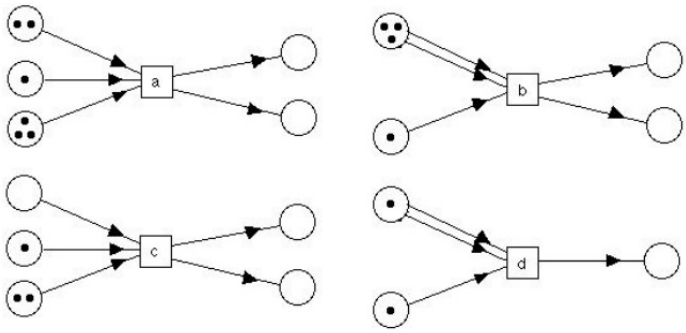
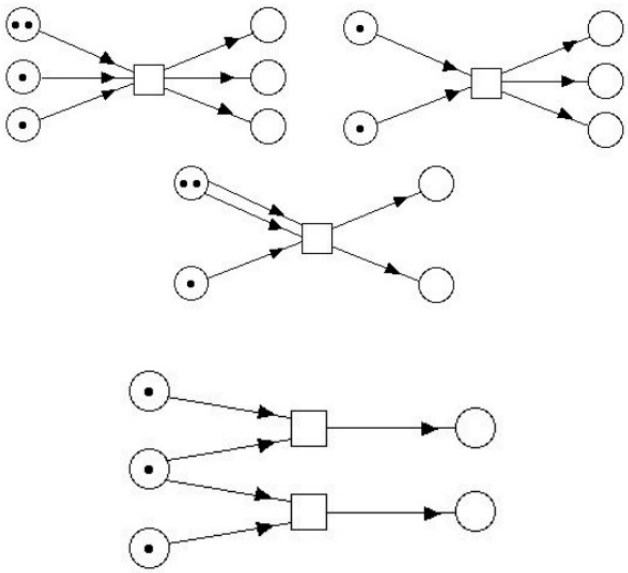
TP 6 – Petri Nets

Exercice 1 Let $T = (C, c_0, \rightarrow)$ be a transition system whose locations C are partially ordered by $\preceq$. Suppose that T is monotonic (therefore T is well-structured) : for all $c_1 \rightarrow c_2$, and all $c_1 \preceq c_3$, there exists c_4 such that $c_3 \rightarrow c_4$ and $c_2 \preceq c_4$. Let U be an upward-closed set. Show that $Pre(U)$ is upward-closed.

Exercice 2 Tell which of the four following transitions are enabled (i.e. can be fired) :



Exercice 3 Show the result of firing the following transitions :

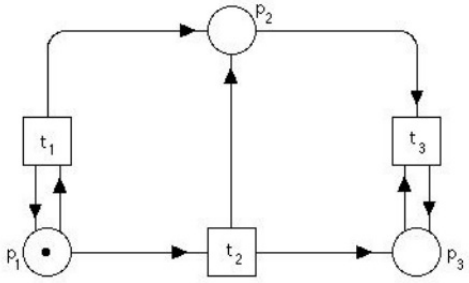


Exercice 4 Construct a Petri net model for calculating the conjunction $x \wedge y$. There must be one initial place for each valuation $x/0$, $x/1$, $y/0$, $y/1$.

Exercice 5 This exercice is about the *dining philosophers* problem [1]. Five philosophers alternatively think and eat. They are seated around a large round table on which are a variety of Chinese foods. Between each pair of philosophers is one chop stick. To eat Chinese food, two chopsticks are necessary ; hence each philosopher must pick up both the chopstick on the left and that on the right. Of course, if all the philosophers pick up the chopstick on their right and then wait for the chopstick on their left to become free, they will wait forever — a deadlock condition. Dijkstra formulated this problem to illustrate control problems that confront operating systems in which processes share resources and may compete for them with deadlock a conceivable result. To solve the problem some philosophers must eat while others are meditating.

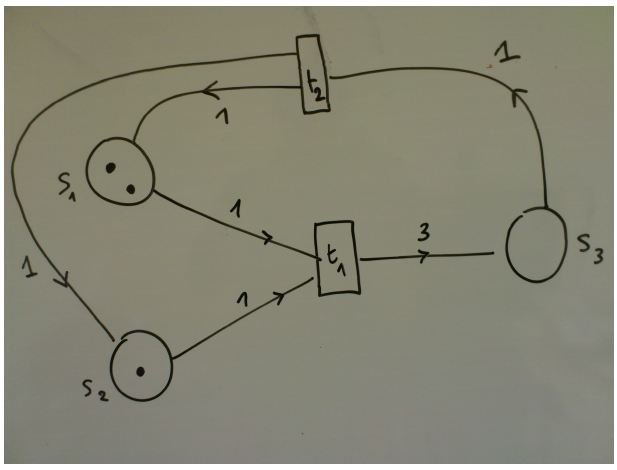
Model this problem with a Petri net. *Hint :* Use places to represent whether a philosopher is meditating or eating, and whether a chop stick is available or not.

Exercice 6 Construct the three first levels of the reachability tree for the following Petri net :



Exercice 7 Construct the finite reachability tree for the previous Petri net and deduce that there exists an infinite computation.

Exercice 8 For the following Petri net, use the backward algorithm to decide whether the marking $m = (1, 0, 4)$ is covered.



1. See `http://highered.mcgraw-hill.com/sites/dl/free/0073383090/299355/Apps_Ch24.pdf`

Exercice 9 Same question but by using the expand, enlarge and check (forward) algorithm. Which k do you need to conclude that m is covered ? Do you think that there are negative instances for some k ?