Exercise 1  Mutual Exclusion algorithm

a. Consider the simplification of the Peterson’s mutual exclusion algorithm as given at the lecture (with \( n_2 \lor t_2 \) and \( n_1 \lor t_1 \) as the only guards on the edges). Draw a graphical representation of the model discussed at the lecture.

b. Give the model for the algorithm given in Figure 1.1 in the lecture notes. Does this algorithm ensure mutual exclusion and eventual access? Why? Draw the graphical representation of this model and compare it with the one from a.

c. Is it possible to hide/abstract away the two (explicit) different propositions \( s_1 \) and \( s_2 \) and to model the information carried by them only with one (implicit) proposition? Construct the graphical representation of this (new) model and compare it with b.

d. Is it always possible to ”merge” two states that satisfy the same propositions into one state? Why?

Exercise 2  a. The following ferryman problem is a simplification of the well-known ferryman+goat+cabbage+wolf problem.

A ferryman has to transfer two big stones from one side of the river to the other side. He can carry only one stone on his boat. Make a model of this system with its graphical representation. Formalize the goal of the system: ”both stones are on the other side of the river” using propositional logic.

b. Give a model of the ferryman+goat+cabbage+wolf problem. How would you define and formalize a safety property (nothing bad will happen) in this system? Is it satisfied?

Exercise 3  A machine \( M \) can be in three different states: ready - the machine is ready to execute a task, busy - the machine is executing a task or fail - the machine has failed. The environment \( E \) (or other process) sends requests to the machine to execute a task. The machine can accept a request if it is ready, after which it starts executing the task, that is, it becomes busy. After finishing the task, if there are no more requests the machine returns to ready state. From both states, ready and busy the machine can fail. Once the machine is repaired it returns to ready state. The environment that provides and stops requests is given as:

a. Construct the model of the machine alone (Note: Do not forget to include guards on transitions)

b. Construct the model of \( M || E \) with the initial state \( \text{ready} \land \text{request} \).