# Test of Discrete Event Systems - 08.10.2014

## Exercise 1

The simplified logic of a lift can be described as follows. The lift is waiting with the sliding door open (waiting state). When it receives a request to move to another floor, it starts closing the sliding door. If, during this operation, an impulse is received from the photocell located at the sliding door, for security reasons the sliding door is opened, and the lift returns in the waiting state. Otherwise, the lift moves according to the request. As soon as the destination floor is reached, the sliding door is opened, and then the lift is put in the waiting state.

1. Model the logic of the lift.

## Exercise 2

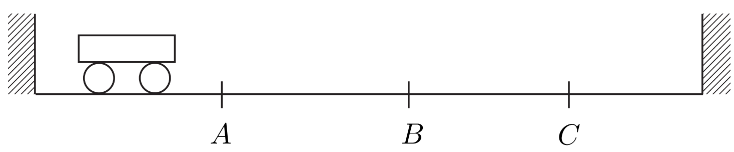
A machine performs operations of three types, denoted by $a$, $b$ and $c$. For technical reasons, an operation of type $c$ cannot be performed immediately after two consecutive operations both of type $a$, or both of type $b$.

1. Model the logic of the machine.
2. Model a system designed to support the scheduling of the operations on the machine: given a sequence of operations, the system returns whether the sequence is feasible for the machine, or not.

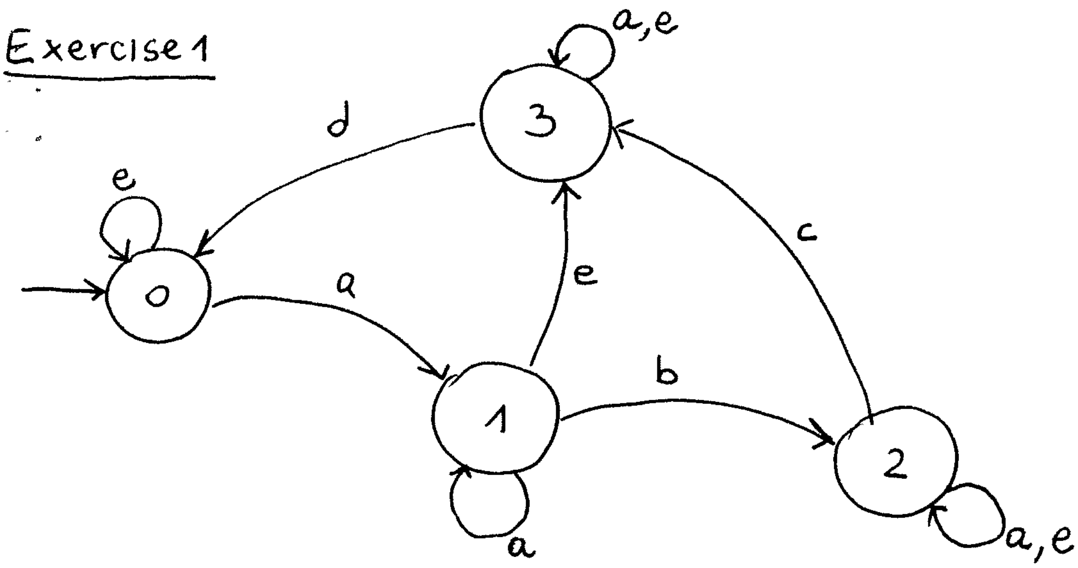
## Exercise 3

A cart moves along a track. Sensors are located at three points of the track (they are denoted by $A$, $B$ and $C$ in the figure). Each sensor sends an impulse when the cart crosses the corresponding point, in both directions. For the sake of simplicity, it is assumed that the cart never changes direction when it is across a sensor.

1. Provide a logical model of the cart position along the track.
2. Model a monitoring system which localizes the cart over the track, and detects possible failures of the sensors, by using only the signals it receives from the sensors.



# Exercise 1




events:

- a: arrival of a request
- b: door closed
- c: destination floor reached
- d: door opened
- e: impulse from photocell

states:

- 0: still with open door (waiting state) ← initial state
- 1: door closing
- 2: moving to destination floor
- 3: door opening

$\Rightarrow$ state automaton $(\mathcal{E}, \mathcal{X}, \Gamma, f, x_0)$

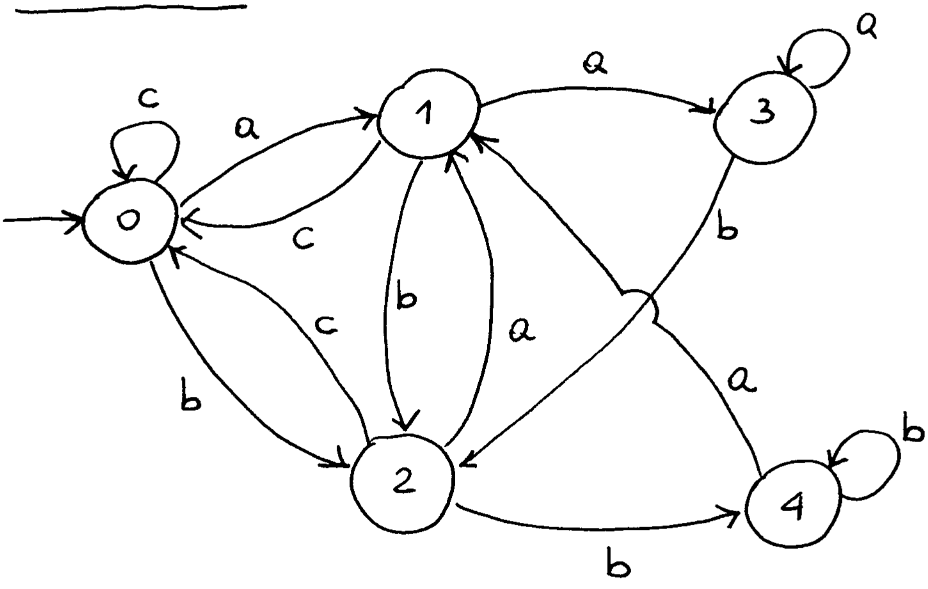
$\mathcal{E} = \{a, b, c, d, e\}$

$\mathcal{X} = \{0, 1, 2, 3\}$

$\Gamma, f, x_0$: implicitly defined by the transition graph

# Exercise 2

1.




states:

1: last symbol "a", second last symbol <u>not</u> "a"

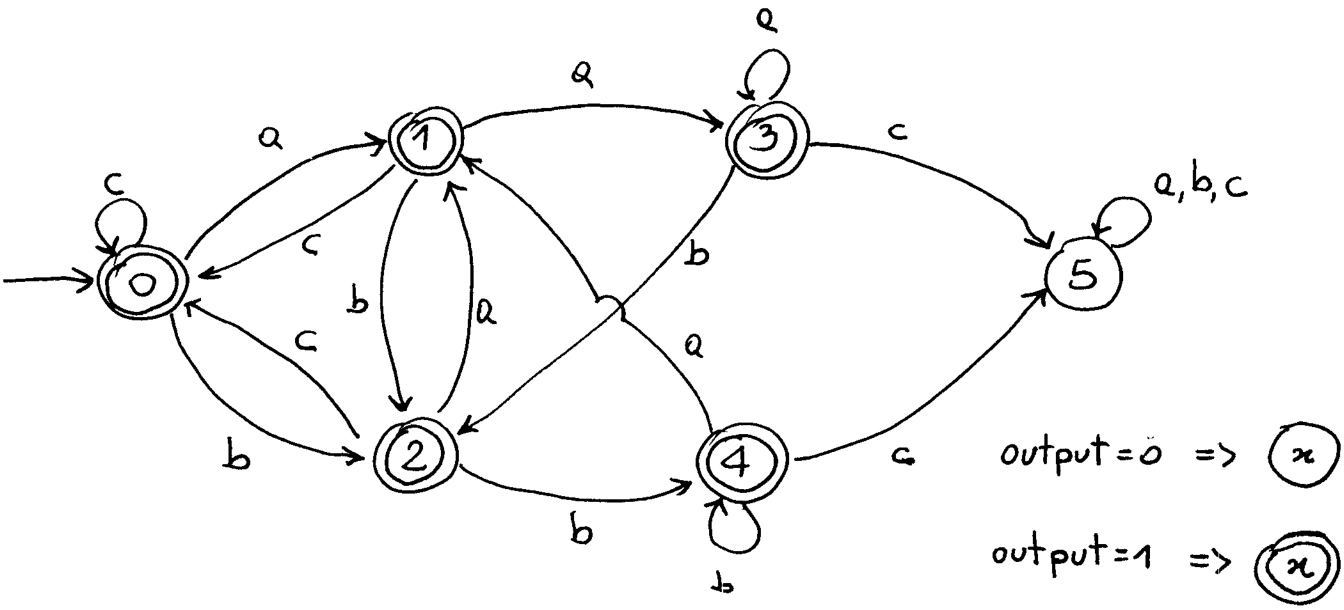
2: " " "b", " " " " "b"

3: last two symbols "a"

4: " " " "b"

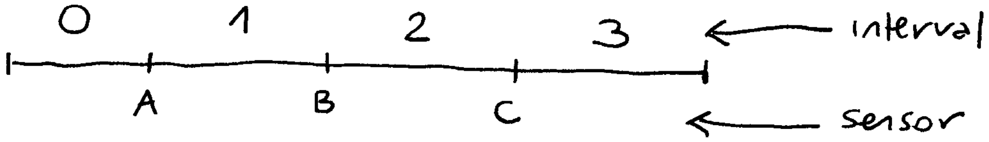
0: otherwise

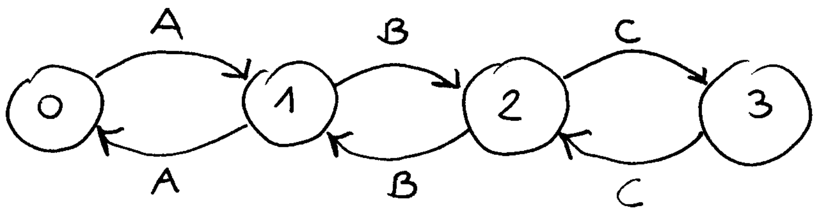
2. we add an absorbing state with output 0 (sequence is infeasible); the output of the other states is 1 (sequence is feasible)



# Exercise 3

1.




2. we add an absorbing state "collecting" warning situations