Exam sample: Discrete Event Systems
Master in Control

Lecturer: D. Angeli

March 18, 2009

1 Exercise

Consider the automata whose transition diagrams are shown in Fig. 1. They are meant to model queues with arrival events denoted by a_1 and a_2 and departure events denoted by d_1 and d_2.

- Assume that event d_1 is unobservable; build a diagnoser for the occurrence of event d_1 in Q_1
- Build the parallel composition $Q_1||Q_2$ and show its associated transition graph
- Assume that events d_1 and d_2 are partially observable, that is a sensor is available to detect occurrence of d_1 or d_2 but cannot discriminate between the two. Build a non-deterministic automaton to model such situation.
- For the automaton previously constructed build the observer automaton (that is a deterministic automaton with the same generated language)

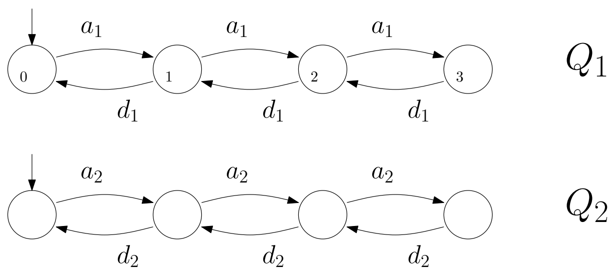

Figure 1: Queue models

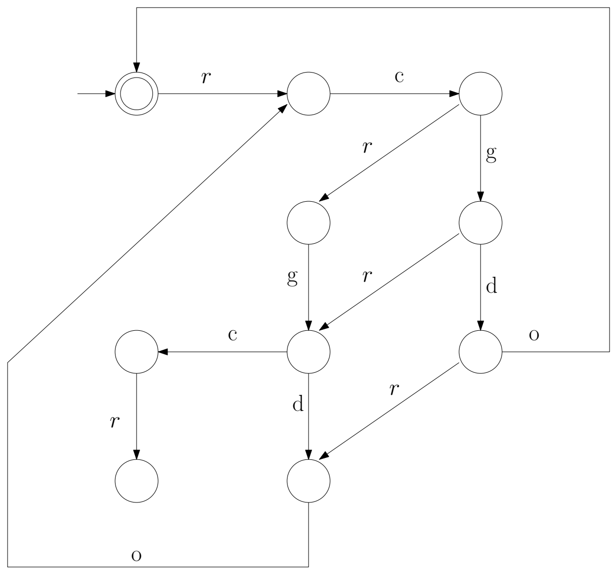

Figure 2: The automaton

2 Exercise

Consider the deterministic automaton G whose transition graph is shown in Fig. 2. The event set is $E = \{r, c, g, d, o\}$ with c and o uncontrollable events, that is $E_{uc} = \{c, o\}$. The control specification is that never the sequence g,c should occur.

- Build an automaton J which implements the control specifications
- Is the language $\mathcal{L}(G) \cap \mathcal{L}(J)$ controllable with respect to $\mathcal{L}(G)$ and E_{uc}?
- Compute the automaton associated to the supremal controllable sublanguage of $\mathcal{L}(G) \cap \mathcal{L}(J)$. Design a supervisor for the automaton and derive the automaton associated to the closed-loop behavior. Is the resulting system deadlock-free?

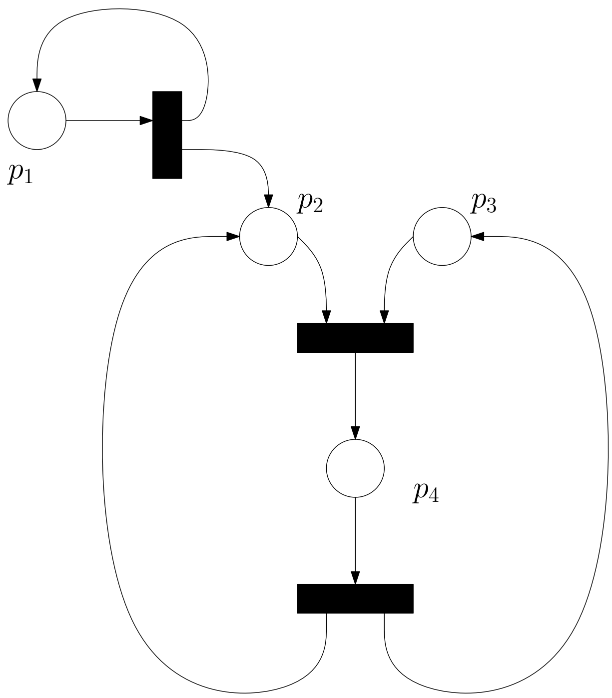

Figure 3: A Petri Net

3 Exercise

- For the Petri Net in the picture, derive the incidence matrix.
- Take the initial marking $M_0 = [0, 1, 1, 1]'$ and compute the associated coverability graph
- Is the network bounded ? Is the network structurally bounded ?
- What are the T-invariants of the network ?

4 Solution of Exercise 1

- Consider the automaton shown in Fig. 4. The automaton sits in the N state until some d_1 event occurs; it then switches to the Y state and stays there everafter. In order to design a diagnoser we build the concurrent composition between this automaton and the queue. This results in the automaton shown in Fig. 5.

  The next step is to replace unobservable events by ε transitions. This leads to the non-deterministic automaton shown in Fig. 6. A diagnoser is

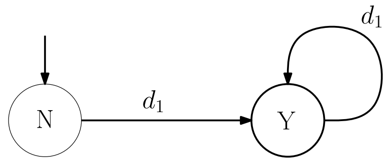

Figure 4: An automaton keeping track of d_1 occurrence

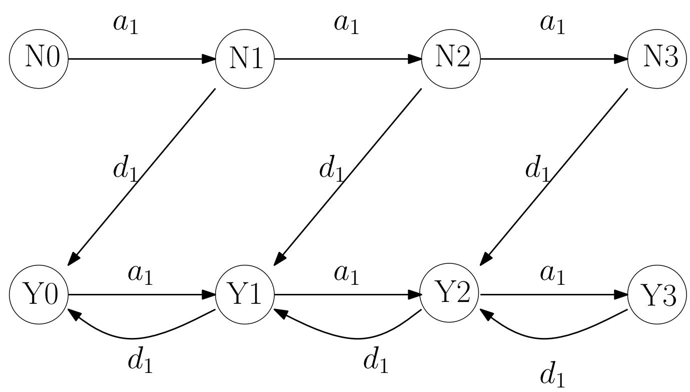

Figure 5: Concurrent composition: deterministic automaton

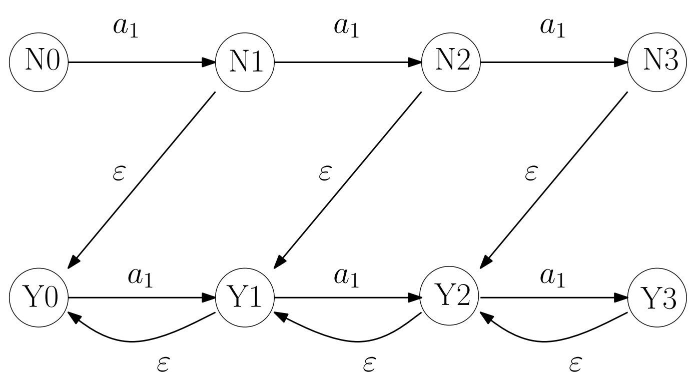

Figure 6: Concurrent composition: non deterministic automaton

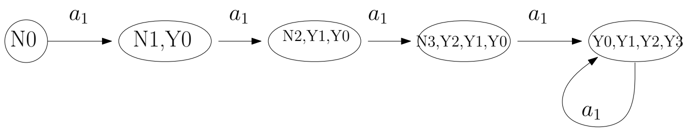

Figure 7: The diagnoser

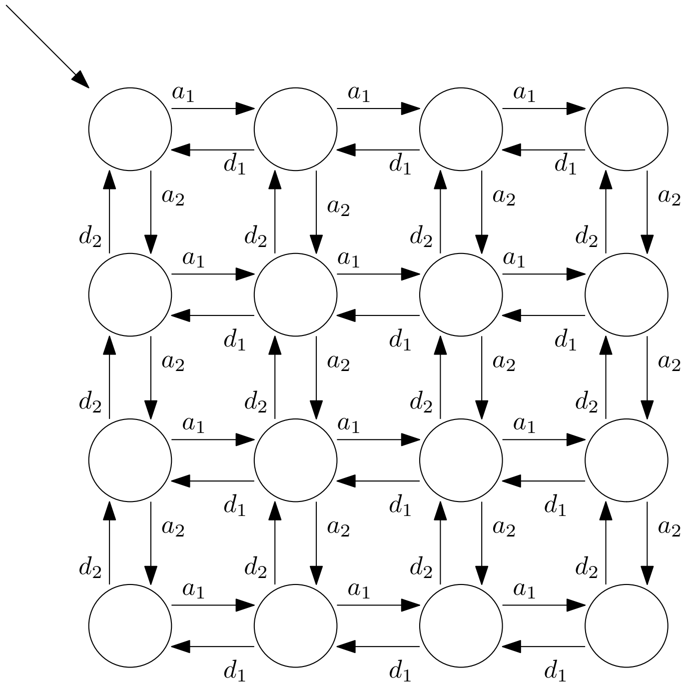

Figure 8: Composition of Q_1 and Q_2

then designed by building the observer of the previous non-deterministic automaton. See Fig. 7.

- The concurrent composition of Q_1 and Q_2 is shown in Fig. 8.
- Taking into accounts partial observability of d_1 and d_2 we end up considering the non-deterministic automaton in Fig. 9. Let us label the states of $Q1$ and $Q2$ with numbers from 0 to 3 according to the queue length. Then, the states of the nondeterministic automaton in Fig. 9 can be labeled with pairs of numbers between 0 and 3, such as $(0, 0)$, $(0, 1)$, $(1, 0) \ldots$ and so on. Let X denote the state-space of our non-deterministic automaton. We partition it as:

$$X = \bigcup_{k=0}^{6} X_i$$

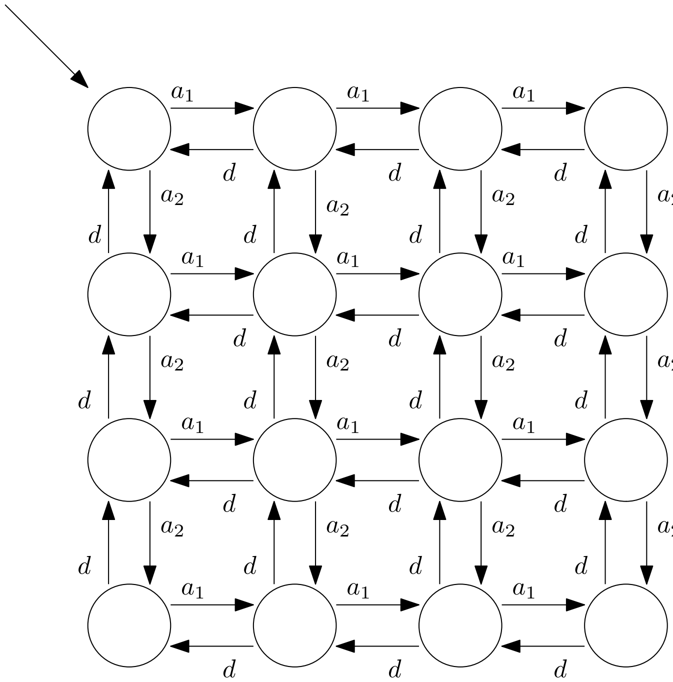

Figure 9: Partial observable events d_1 and d_2 are replaced by d

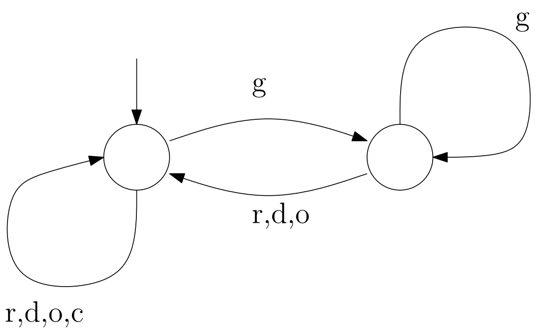

Figure 10: Automaton J: implementing the specification gc should not occur

where $X_i = \{(m,n) \in X : m+n = i\}$. It turns out that the observer automaton has a state space which can be taken to be:

$$X_d = \bigcup_{k=0}^{6} 2^{X_i}$$

The initial state is obviously $\{(0,0)\}$ and the transitions are organized according to the following rule:

$$\delta(x, a_1) = \bigcup_{(m,n)\in x} \{(m+1, n)\}$$

$$\delta(x, a_2) = \bigcup_{(m,n)\in x} \{(m, n+1)\}$$

$$\delta(x, d) = \bigcup_{(m,n)\in x} \{(\max\{m-1, 0\}, n), (m, \max\{n-1, 0\})\}$$

5 Solution of Exercise 2

- The automaton J can be realized with two states: one corresponding to event g just happened and one corresponding to some other event just happened. A scheme of the automaton is shown in Fig. 10.
- The language $\mathcal{L}(G) \cap \mathcal{L}(J)$ is not controllable with respect to $\mathcal{L}(G)$ and E_{uc}. Indeed, the word: $rcrg$ belongs to $\mathcal{L}(G) \cap \mathcal{L}(J)$, and concatenated with the uncontrollable event c yields $rcrgc$ which belongs to $\mathcal{L}(G)$ but not to $\mathcal{L}(J)$, thus violating controllability.
- The supremal controllable sublanguage of $\mathcal{L}(G) \cap \mathcal{L}(J)$ can be obtained by removing a path from the original automaton, as in Fig. 11. The supervisor, hence, disables event g in the state reached after word rcr. It is a blocking supervisor.

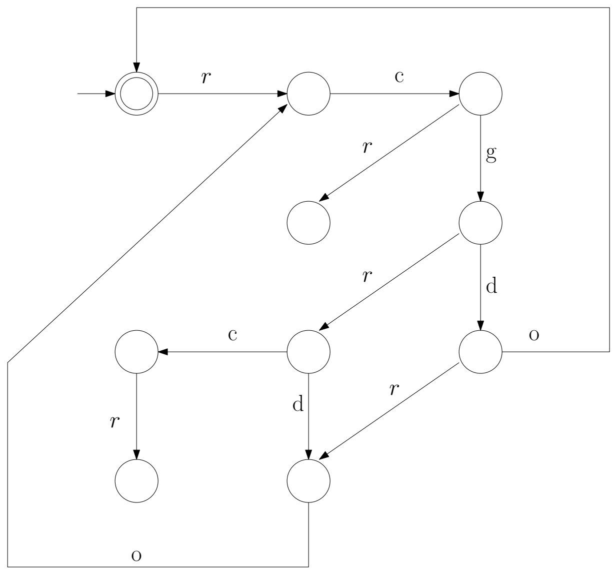

Figure 11: Supremal controllable sublanguage of $\mathcal{L}(G) \cap \mathcal{L}(J)$

6 Solutions of Exercise 3

- The incidence matrix is given by:

$$C = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -1 & 1 \\ 0 & -1 & 1 \\ 0 & 1 & -1 \end{bmatrix}.$$

- The coverability and reachability graph are the following:

$$[0,0,0,2]' \leftrightarrow [0,1,1,1]' \leftrightarrow [0,2,2,0]'$$

- The network is bounded, but not structurally bounded. Indeed the minimal P-semiflows of the network are given by $[1,0,0,0]$ and $[0,0,1,1]$, and the union of their supports does not include all of the places.

- There is only one T-semiflow, namely the vector $[0,1,1]'$. There is one T-increasing vector, namely $[1,0,0]'$.