





1. A workcell consists of two machines  $M_1$  and  $M_2$  and an automated guided vehicle AGV. The automata modeling the two machines are shown in Fig. 1.1.

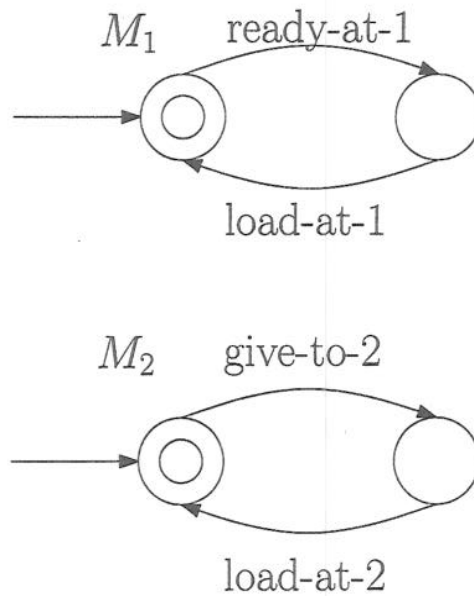


Figure 1.1 Machines  $M_1$  and  $M_2$

$M_1$  is typically working on a piece, until the piece is completed (event ready-at-1), and kept at the machine. Subsequently the AGV may come and get the piece from  $M_1$  (event load-at-1).  $M_2$  instead is waiting for the piece to come (event give-to-2), and then for the AGV to pick-up the piece (event load-at-2) once the processing is completed.

- Assuming that once a piece is taken from  $M_2$  it is transported out of the plant by the AGV (event give-out), derive an automaton  $A$  to model the AGV. Its terminal state(s) are only those in which the AGV does not carry pieces. Your answer should include an event set as well as a graphical representation. [ 8 ]
- A model for the overall workcell is obtained by parallel composition  $G = M_1 || M_2 || A$ . Compute  $G$  and represent it graphically. [ 8 ]
- Is  $G$  a blocking or a non-blocking automaton? Why? [ 4 ]

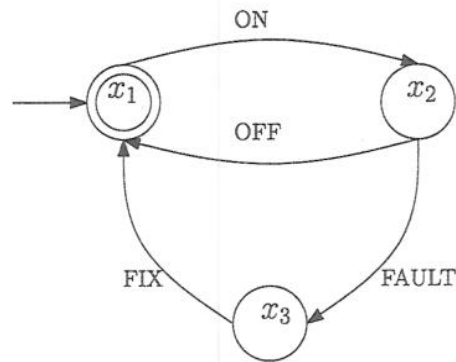


Figure 2.1 Model of a faulty machine

2. Consider the model of a faulty machine shown in Figure 2.1. Assume that 2 such machines are available, and label the corresponding events  $ON_1$  and  $ON_2$ ,  $OFF_1$  and  $OFF_2$  and so on.
  - a) Compute the Automaton associated with the concurrent composition of the two machines [ 6 ]
  - b) Assume that all events are partially observable, meaning that sensors are not able to discriminate between machine 1 and machine 2 as far as each of the events  $ON$ ,  $OFF$ ,  $FAULT$  and  $FIX$  are concerned. Construct the non-deterministic automaton associated with this scenario [ 3 ]
  - c) For the non-deterministic automaton specified above, compute and draw the observer automaton (viz. deterministic automaton which recognizes the same language). [ 6 ]
  - d) Assume next that:  $ON_1$ ,  $ON_2$ ,  $OFF_1$  and  $OFF_2$  are observable,  $FAULT$  is completely unobservable, whereas  $FIX$  is partially observable. Briefly illustrate the steps that need to be taken in order to design a diagnoser to detect whether or not  $FAULT_2$  has ever occurred. [ 5 ]

This page is intentionally left blank.

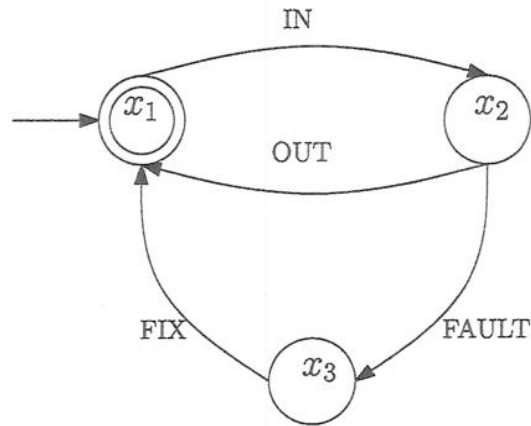


Figure 3.1 A faulty machine as in Exercise 3

3. Consider a factory composed of the following 3 systems: 2 faulty machines (see Figure 3.1) operating in series and separated by a buffer of finite capacity. In particular, event  $IN_i$  corresponds to a piece entering the  $i$ -th machine and starting being processed. Event  $OUT_i$  corresponds to the piece completing its process and exiting machine  $i$ . Operation in series means that the piece exiting Machine 1 enters the Buffer. Moreover, Machine 2, takes its pieces from the Buffer.
- a) Build a model of a buffer of capacity 1 taking into account the possibility of overflows (when more than 1 piece is deposited in the buffer) and underflows (if Machine 2 attempts to gather a piece when buffer is empty). [ 5 ]
  - b) Build a deterministic automata model  $G$  of the factory by concurrent composition of the two machines and the buffer model; assume empty buffer as an initial state. For the sake of simplicity, when drawing the automata, do not give details of the states which involve an Overflow or an Underflow condition for the buffer (replace all such states by a single OF and UF states). [ 5 ]
  - c) Construct automata  $S_1$  and  $S_2$  which recognize languages fulfilling respectively the following control specifications: [ 5 ]
    - i) Buffer should not overflow nor underflow.
    - ii) If Machine 1 and Machine 2 are both faulty, then Machine 2 should be fixed first.

Question 3 continued.

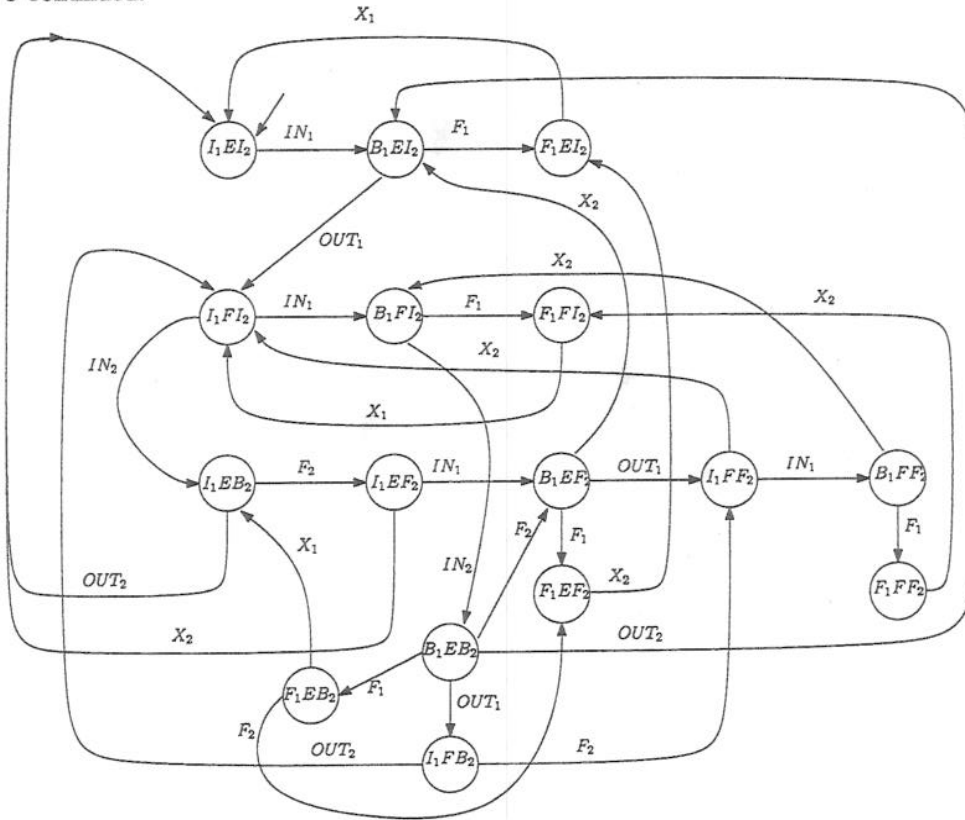


Figure 3.2 Automaton  $G_K$

- d) Let  $G_K = S_1 || S_2 || G$  where  $G$  is defined in part b). The corresponding automaton is shown in Figure 3.2. Let  $K \subset \mathcal{L}(G)$  be the language generated by  $G_K$ . Assume that:

$$E_c = \{IN_1, FIX_1, IN_2, FIX_2\}$$

$$E_{uc} = \{OUT_1, FAULT_1, OUT_2, FAULT_2\}.$$

Is  $K$  controllable with respect to  $\mathcal{L}(G)$  and  $E_{uc}$ ? Justify your response. [ 5 ]

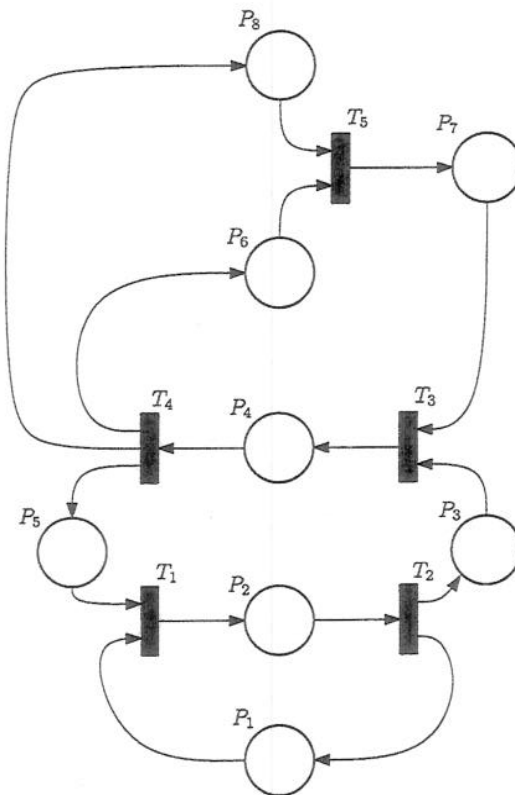


Figure 4.1 Petri Net graph

4. Consider the Petri Net in Fig. 4.1.
- Compute the incidence matrix  $C$  of the net. [ 4 ]
  - For  $M_0 = [1, 1, 0, 0, 1, 0, 1, 0]^t$ , compute the reachable set  $\mathcal{R}(M_0)$  and show the associated transition graph, with labels attached to nodes (corresponding to the marking) and edges (corresponding to the transition being fired). [ 4 ]
  - Compute the  $P$ -invariant vectors (of minimal support) of the network. [ 5 ]
  - Is the reachable set bounded for all initial markings  $M_0$ ? Why? [ 3 ]
  - Compute the  $T$ -invariant vectors of minimal support of the network and give an intuitive description of their physical meaning. [ 4 ]

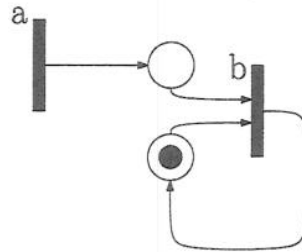


Figure 6.1 Marked Petri Net

5. At some bus stop (with only one type of buses passing by), people arrive with a probability rate of  $\lambda_a$ . Whenever a bus comes, which happens with probability rate  $\lambda_b$ , independently of the number of people waiting for it, the bus stop is emptied out. Let  $n$  be the maximum number of people which can accumulate at the bus stop (let us say that afterwards there is no more room).
- Write down the equations of a continuous-time Markov chain which keeps track (in probability) of the number of people waiting for the bus at any given instant of time. [ 3 ]
  - Give a graphical representation of the model derived. [ 2 ]
  - Is the chain ergodic? Why? [ 3 ]
  - Show that, no matter what  $n$  is, it is possible to derive a simplified Markov chain, with only two states, one representing the situation of an empty bus stop, and the other of a stop with at least one person waiting; (let  $\pi_i, i = 0 \dots n$  denote the probability of having exactly  $i$  people at the stop, define new variables  $\tilde{\pi}_0 = \pi_0$  and  $\tilde{\pi}_1 = \pi_1 + \pi_2 + \dots + \pi_n$ , and write down their evolution equation). [ 3 ]
  - Use the simplified model in order to compute the average time it takes for the bus to arrive (this is actually independent of  $n$ ). [ 3 ]
  - Compute what fraction of time the bus-stop is empty. [ 3 ]
  - The average number of people waiting at the bus-stop is instead a function of  $n$ . How can we compute it for a given value of  $n$ ? [ 3 ]
6. Consider the Petri Net  $N$  shown in Fig. 6.1. For a given initial marking  $M_0$ , we may define the language generated by  $N$  as follows:

$$\mathcal{L}(N) = \{w \in T^* : M_0[w >]\}.$$

- Compute the reachable set for  $M_0 = [0, 1]'$ ; [ 5 ]
- Compute the coverability graph. [ 5 ]
- Compute  $\mathcal{L}(N)$  for  $M_0 = [0, 1]'$ . [ 5 ]
- Show that no finite state automaton is capable of generating the same language. (Hint: call 2 states equivalent if and only if the automaton initialized at those states generates the same language; how many distinct equivalence classes do you recognize?). [ 5 ]



SOLUTIONS COURSEWORK: DISCRETE EVENT SYSTEMS  
 MASTER IN CONTROL

1. Exercise

- a) The automaton has event set  $E = \{ \text{load-at-1}, \text{load-at-2}, \text{give-to-2}, \text{give-out} \}$ . It can be graphically represented as in Fig. 1.1.

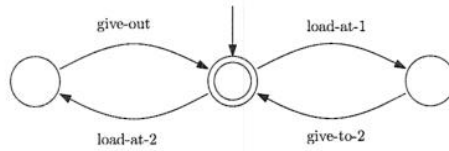


Figure 1.1 Autonomous vehicle model

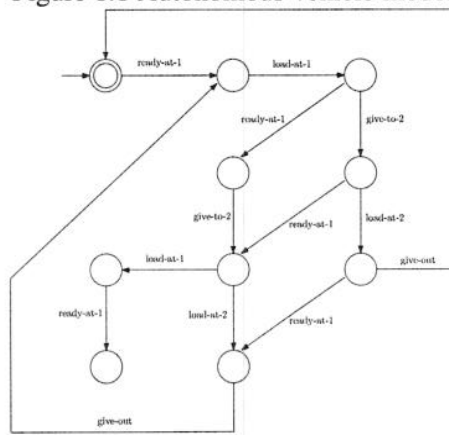


Figure 1.2 Automaton  $M_1 || M_2 || A$

- b) The concurrent composition is shown in Fig. 1.2.  
 c) It is a blocking automaton; indeed blocking occurs when both machines have a piece and the autonomous vehicle is carrying a piece from 1 to 2.

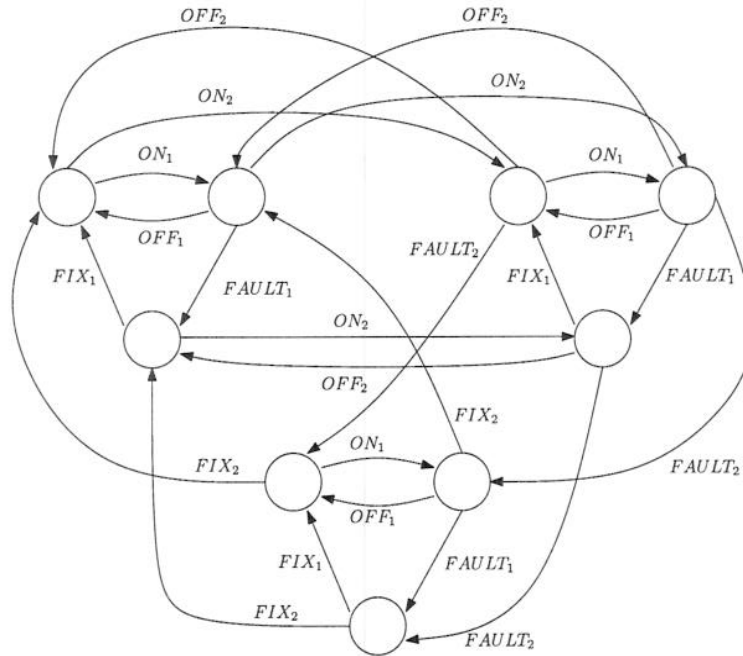


Figure 2.1 Concurrent machines automaton

2. Exercise

- a) The concurrent composition of the two machines automata is shown in Fig. 2.1.
- b) The non-deterministic automaton is shown in Fig. 2.2.
- c) The observer automaton is then derived in Fig. 2.3.
- d) In order to design a diagnoser for event  $FAULT_2$  we first need to do parallel composition with the simple automaton composed of states  $X = \{N, Y\}$ , initial state  $N$  and event set  $E = \{FAULT_2\}$ . Moreover, letting  $\delta$  denote the transition map of the automaton, we have  $\delta(N, FAULT_2) = Y$  and  $\delta(Y, FAULT_2) = Y$ . This leads to the automaton shown in Fig. 2.4. The next step is to replace unobservable events by  $\epsilon$  and partially observable events by the new label FIX. This leads to the non-deterministic automaton shown in Fig. 2.5: Finally one has to design the observer for the automaton in 2.5.

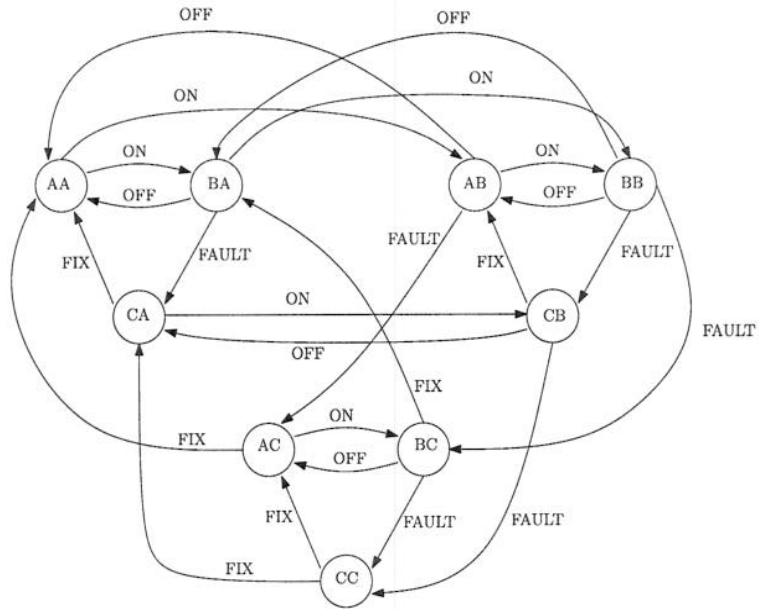


Figure 2.2 Non-deterministic Concurrent machines automaton

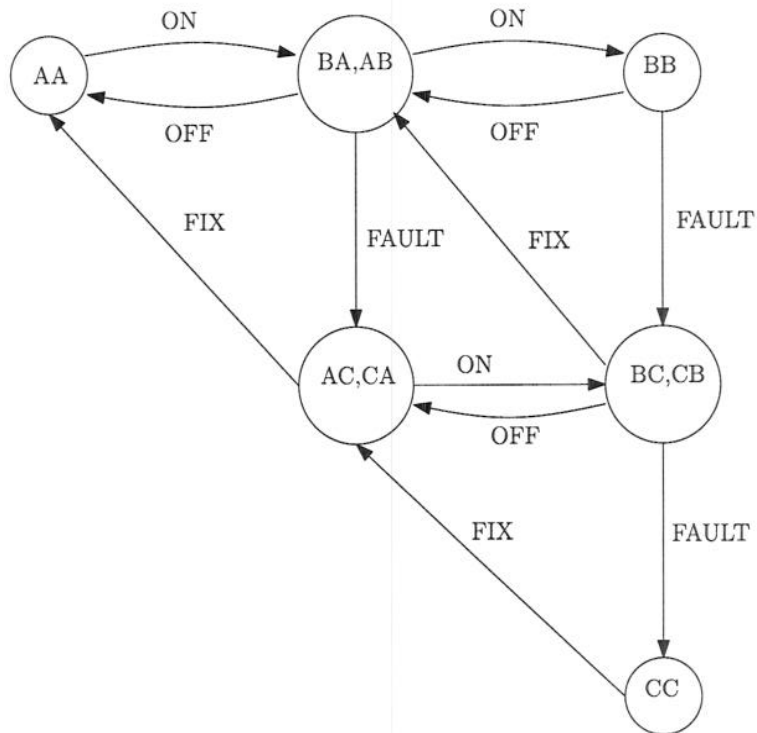


Figure 2.3 Observer automaton

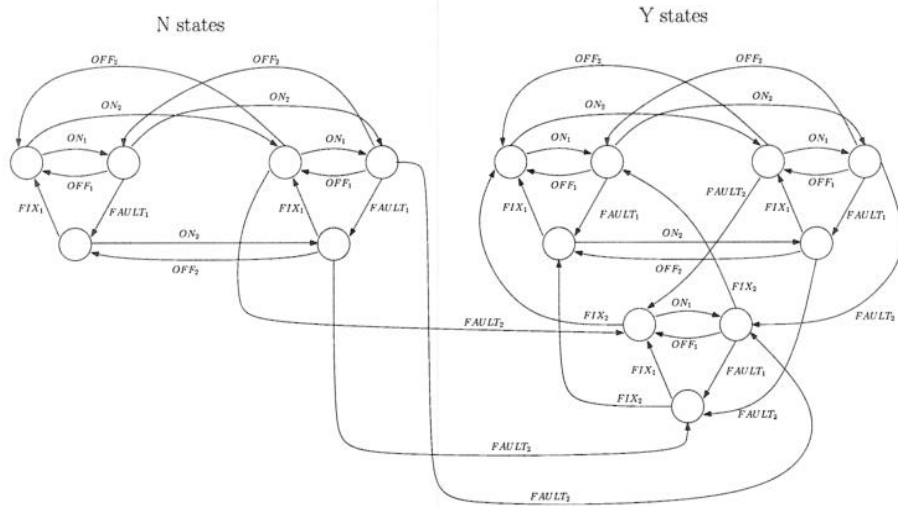


Figure 2.4 Concurrent composition for diagnosis

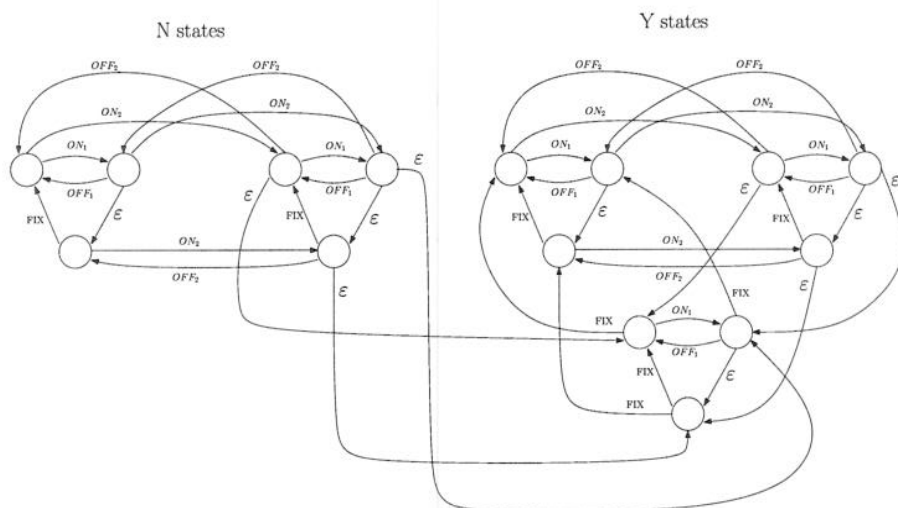


Figure 2.5 Non-deterministic automaton for  $FAULT_2$  diagnosis

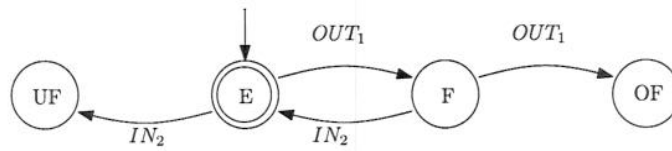


Figure 3.1 Buffer model

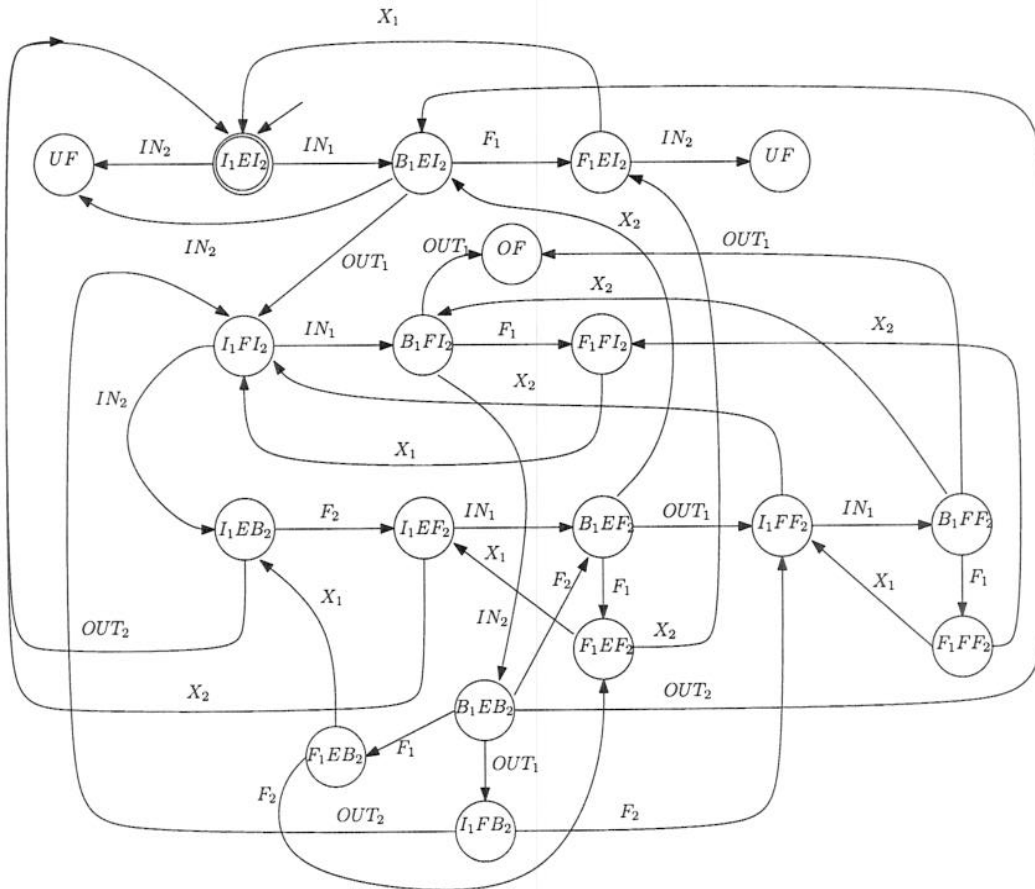


Figure 3.2 Factory model

3. Exercise

- a) The buffer has  $E = \{OUT_1, IN_2\}$ ; in particular, it takes a piece in whenever  $OUT_1$  occurs, and it sends a piece out whenever  $IN_2$  occurs. A buffer of capacity one has the model shown in Fig. 3.1. The letters E,F,OF,UF stand for Empty, Full, Overflow and Underflow.
- b) Let us denote the states  $x_1, x_2$  and  $x_3$  by I,B and F, standing for Idle, Busy and Faulty. Then, we may denote with a combination of 3 letters every state of the concurrent composition of Machine1, Machine2 and the buffer. For instance state  $I_1EI_2$  means, Machine 1 is Idle, Buffer is empty and Machine 2 is idle. The resulting automaton is shown in Fig. 3.2 (for the sake of simplicity we replaced events  $FAULT_i$  by  $F_i$  and  $FIX_i$  by  $X_i$ )

- c) The control specifications automata are given in Fig. 3.3

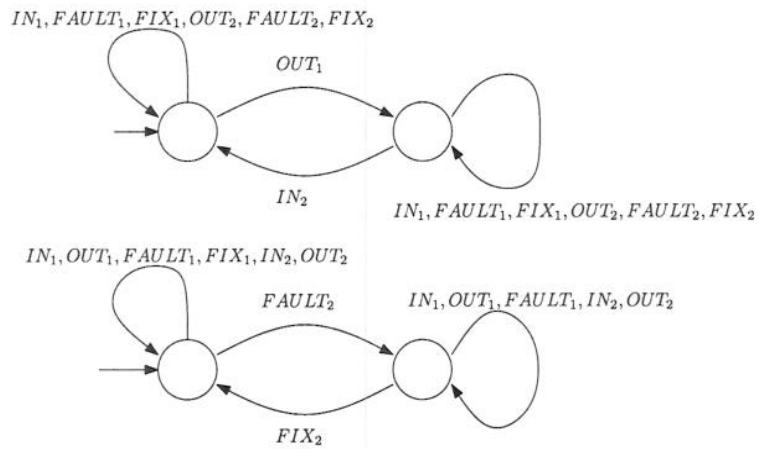


Figure 3.3 Control specifications

- d)  $K$  is uncontrollable: for instance  $IN_1OUT_1IN_1OUT_1$  leads to an overflow state, hence it belongs to  $\mathcal{L}(G)$ , it does not belong to  $K$ ; moreover  $IN_1OUT_1IN_1$  belongs to  $K$ , but the last event  $OUT_1 \in E_{uc}$ .

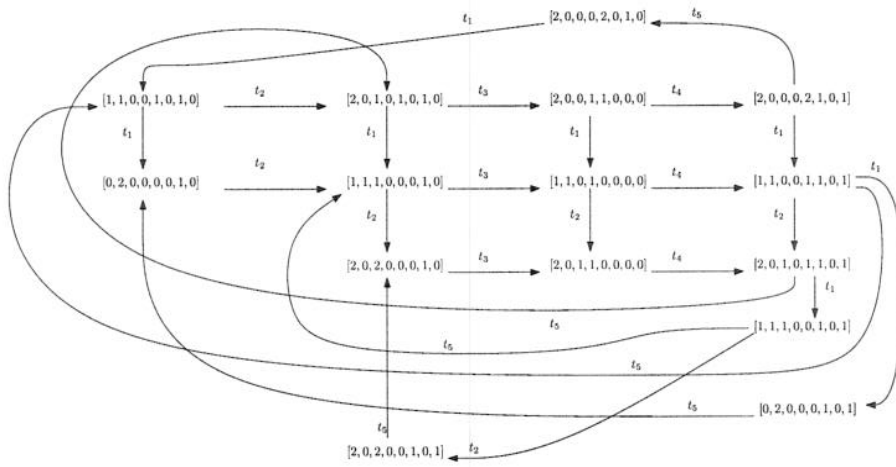


Figure 4.1 Transition graph

4. Exercise

- a) With the order of places and transitions used in Figure 4.1 in the question sheet, the incidence matrix is given by:

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

- b) The reachable set  $\mathcal{R}(M_0)$  and the associated transition graph are shown in Fig. 4.1.
- c) The  $P$ -invariant vectors of minimal support are given by:  
 $[1, 1, 0, 0, 0, 0, 0, 0], [0, 1, 1, 1, 1, 0, 0, 0], [0, 0, 0, 1, 0, 1, 1, 0], [0, 0, 0, 1, 0, 0, 1, 1]$
- d) Notice that  $[1, 2, 1, 2, 1, 1, 2, 1]$  is a  $P$ -invariant vector of support coinciding with  $P$ ; hence the network is structurally bounded and the reachable set is finite for all initial markings.
- e) There is only one  $T$ -invariant vector  $[1, 1, 1, 1, 1]'$ . Whenever each transition fires once (in any order) the state goes back to its initial value.

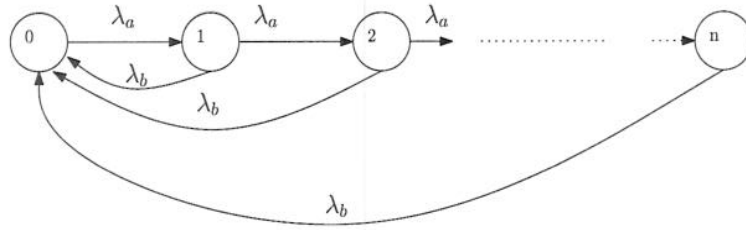


Figure 5.1 Bus stop queue Markov chain

5. Exercise

- a) Its equations (for the case of  $n$  generic) are given by:

$$\begin{aligned}
 \dot{\pi}_0 &= -\lambda_a \pi_0 + \lambda_b \pi_1 + \lambda_b \pi_2 + \dots + \lambda_b \pi_n \\
 \dot{\pi}_1 &= -(\lambda_a + \lambda_b) \pi_1 + \lambda_a \pi_0 \\
 &\vdots \\
 \dot{\pi}_i &= -(\lambda_a + \lambda_b) \pi_i + \lambda_a \pi_{i-1} \\
 &\vdots \\
 \dot{\pi}_n &= -\lambda_b \pi_n + \lambda_a \pi_{n-1}
 \end{aligned}$$

- b) The requested continuous time Markov chain is shown in Fig. 5.1.  
 c) The chain is ergodic, because its graph is strongly connected (hence associated matrix is irreducible).  
 d) Let us define

$$\tilde{\pi}_0 = \pi_0 \quad \tilde{\pi}_1 = \pi_1 + \pi_2 + \dots + \pi_n$$

The equations governing the evolution of this two variables are given as:

$$\begin{aligned}
 \dot{\tilde{\pi}}_0 &= -\lambda_a \tilde{\pi}_0 + \lambda_b \tilde{\pi}_1 \\
 \dot{\tilde{\pi}}_1 &= \lambda_a \tilde{\pi}_0 - \lambda_b \tilde{\pi}_1
 \end{aligned}$$

Notice that these are again equations of a Markov chain.

- e) Assume that we are in state 1, (and we have been there for it doesn't matter how much time). The next bus will come when we transition to state 0. Let us make 0 into an absorbing state as we are only interested in the first transition time from state 1 to state 0. Hence, our equation reads:

$$\dot{\tilde{\pi}}_1 = -\lambda_b \tilde{\pi}_1 \quad \tilde{\pi}_1(0) = 1$$

with  $\lambda_b \tilde{\pi}_1(t)$  denoting the probability that the transition occurs at time  $t$ . Hence,  $\tilde{\pi}_1(t) = e^{-\lambda_b t}$ . The average time is computed as:

$$\int_0^{+\infty} t \lambda_b e^{-\lambda_b t} dt = \frac{1}{\lambda_b}$$

Indeed, average time is independent of  $n$  as the reduced model is also independent of  $n$ .

- f) Let us compute steady-state probability distributions for the reduced Markov chain. Also, adding the additional normalization constraint:  $\tilde{\pi}_0 + \tilde{\pi}_1 = 1$  we can easily derive

$$\tilde{\pi}_0 = \frac{\lambda_b}{\lambda_a + \lambda_b}$$

- g) On the original Markov chain we can apply the ergodic theorem to compute the average number of people waiting. We can weight the steady state probability distribution  $\pi_i(\infty)$  by the factor  $i$ . Hence, the desired quantity is computed as:

$$\sum_{i=0}^n i\pi_i(\infty)$$

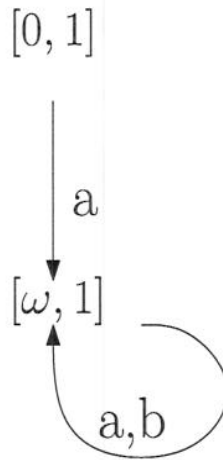


Figure 6.1 Coverability graph

6. Exercise

- a) Notice that occurrence of  $b$  does not change the amount of tokens in the second place. Transition  $a$ , on the other hand, can occur an arbitrary number of times, as it does not require any tokens to be fired. The reachable set is:

$$\mathcal{R}(M_0) = \{[n, 1] : n \in \mathbb{N}\}$$

- b) The coverability graph is given in Fig. 6.1:
- c)  $\mathcal{L}(N) = \{w \in T^* : \forall \text{ prefix } v \text{ of } w, |v|_a \geq |v|_b\}$  where  $|v|_a$  denotes the number of  $a$  events within the word  $v$  and similarly  $|v|_b$  denotes the number of  $b$  events.
- d) Clearly,  $a^n$  belongs to  $\mathcal{L}(N)$  for all  $n \in \mathbb{N}$ . Let  $S_n$  be the state of an automaton which generates  $\mathcal{L}_N$  which is reached after word  $a^n$ . Clearly,  $b^m$  belongs to the language generated by the automaton initialized at  $S_n$  iff  $m \leq n$ . Hence,  $n_1 \neq n_2$  implies that the corresponding states  $S_{n_1}$  and  $S_{n_2}$  are not equivalent. By arbitrariness of  $n$ , infinitely many states are needed.