Discrete Event Systems
Solution to Exercise Sheet 13

1 Structural Properties of Petri Nets and Token Game

Given is the following Petri net $N_1$:

![Petri Net Diagram]

a) What are the Pre and Post sets of transitions $t_5$ and $t_8$ and of place $p_3$?

b) Which transitions are enabled after $t_1$ and $t_2$ fired?

c) What is the total number of tokens in $N_1$ before and after $t_2$ fired?

d) Play the token game for $N_1$ and construct the reachability graph.

**Hint:** You may denote the states in such a way that the index indicates the places that hold a token in this state, for example $s_0 = (1, 0, 0, 0, 1, 0, 0, 0, 0, 0) \triangleq s_{1,5}.

a) The Pre and Post sets of a transition are defined as follows:

- Pre set: $\bullet t := \{ p \mid (p, t) \in F \}$
- Post set: $\bullet t := \{ p \mid (t, p) \in F \},$
where $F$ is the flow set, i.e., the set of place/transition and transition/place arcs. The Pre
and Post sets of a place are defined analogously.

For the Petri net $N_1$ we obtain the following sets:

\[ t_5 = \{p_5, p_9\}, \quad t_5\bullet = \{p_6\} \]
\[ t_8 = \{p_8\}, \quad t_8\bullet = \{p_{10}, p_5\} \]
\[ p_3 = \{t_2\}, \quad p_3\bullet = \{t_3\} \]

b) A transition is enabled if all places in its Pre set contain enough tokens. In the case of $N_1$, which has only unweighted edges, one token per place suffices. When $t_2$ fires, it consumes
one token out of each place in the Pre set of $t_2$ and produces one token on each place in
the Post set of $t_2$. Hence, the firing of $t_2$ produces one token on place $p_3$ and $p_5$ each, the
one on $p_2$ is consumed. After this, $t_5$ is enabled because both $p_9$ and $p_5$ hold one token.
However, $t_3$ is not enabled because $p_3$ contains a token but $p_{10}$ does not.

c) Before $t_2$ fires there are two tokens in $N_1$, one on $p_2$ and $p_5$ each. Directly afterwards,
there are tokens on places $p_3$, $p_9$ and $p_5$, hence 3 tokens in total.

d) A token traverses the upper cycle until $t_2$ fires. Then one token remains on $p_3$ and waits,
and another one is produced in $p_9$, which enables transition $t_5$. When $t_5$ consumes
the tokens on $p_9$ and $p_5$ and produces a token on $p_6$, this one traverses the lower cycle until $t_8$
is enabled and fired. One token now remains on $p_5$ and waits, another is in $p_{10}$ and enables
$t_3$, because there is another token on $p_3$. Then one token traverses the upper cycle again
until $t_2$ is enabled, and so on. Hence, this Petri net models two alternating processes.

This Petri net is clearly bounded, thus we can construct its reachability tree. Usually the
states of Petri nets are denoted by vectors such that the i-th position in the vector indicates
the number of tokens on place $p_i$ of the Petri net, i.e., the marking of the graph. So, for
example, the starting state $s_0$ of $N_1$, in which the places $p_1$ and $p_5$ hold one token each, is
denoted by $s_0 = (1, 0, 0, 0, 1, 0, 0, 0, 0, 0)$.

For better legibility we denote the states in such a way that the index contains the places
that hold a token in this state, for example $s_0 = (1, 0, 0, 1, 0, 0, 0, 0) \triangleq s_{1.5}$.

Then the reachability graph can also be written as,
2 Basic Properties of Petri Nets

Given is the following Petri net $N_2$:

Explain the terms boundedness and deadlock-freeness using this example, i.e. for which values of $k \in \mathbb{N}$ is the Petri net $N_2$ bounded/unbounded and not deadlock-free?

A Petri net is $k$-bounded, if there is no fire sequence that makes the number of tokens in one place grow larger than $k$. It is obvious that Petri net $N_2$ is 1-bounded if $k \leq 1$. This holds because in the initial state there is only one token in the net, and in the case $k \leq 1$ no transition increases the number of tokens in $N_2$. If $k \geq 2$, the number of tokens in $p_1$ can grow infinitely large by repeatedly firing $t_1$, $t_3$ and $t_4$. So, the Petri net $N_2$ is unbounded for $k \geq 2$.

A Petri net is deadlock free if no fire sequence leads to a state in which no transition is enabled. If $k = 0$, $N_2$ is not deadlock-free. The fire sequence $t_1, t_3, t_4$ causes the only existing token to be consumed and hence, there is no enabled transition any more. For $k \geq 1$, however, no deadlock can occur.
3 Identifying a deadlock

The following Petri net $N_3$ describes two linear processes ($P_{A0/A1/A2}$ and $P_{B0/B1/B2}$) sharing resources $R_1$ and $R_2$.

In the following, use $M = (P_{A0}, P_{A1}, P_{A2}, P_{R1}, P_{R2}, P_{B0}, P_{B1}, P_{B2})$ as marking vector and $T = (t_{A0}, t_{A1}, t_{A2}, t_{B0}, t_{B1}, t_{B2})$ as firing vector.

a) Determine the reachability graph of this net for the given initial marking. Explicit one or several firing sequences leading to a blocking marking (i.e., to a deadlock). What is this blocking marking?

b) Write down the upstream ($W^-$) and downstream ($W^+$) incidence matrices and deduce the incidence matrix $A$. Use it to compute the marking obtained in the deadlock state (i.e., by firing the blocking sequence) from the previous question.

c) Using the upstream incidence matrix $W^-$, how can you prove that this previous state is a deadlock?

d) Suggest a modification to this Petri net which allows the two linear processes $P_A$ and $P_B$ to run as intended in the first place.
a) There are an infinite number of blocking sequence: any number of cycles \( t_{A0}t_{A1}t_{A2} \) and/or \( t_{B0}t_{B1}t_{B2} \) terminated by either \( t_{A0}t_{B0} \) or \( t_{B0}t_{A0} \). It can be read directly from the marking graph below:

\[
M_0 = (1,0,1,1,1,0,0) (0,1,0,0,0,1,0)
\]

\[
W^+ = \begin{bmatrix}
0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0
\end{bmatrix}
\quad\text{and}\quad
W^- = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
A = W^+ - W^- = \begin{bmatrix}
-1 & 0 & 1 & 0 & 0 & 0 \\
1 & -1 & 0 & 0 & 0 & 0 \\
0 & 1 & -1 & 0 & 0 & 0 \\
-1 & 0 & 1 & 0 & -1 & 1 \\
0 & -1 & 1 & -1 & 0 & 1 \\
0 & 0 & 0 & -1 & 0 & 1 \\
0 & 0 & 0 & 1 & -1 & 0 \\
0 & 0 & 0 & 0 & 1 & -1
\end{bmatrix}
\]

Consider the basic blocking sequence \( t_{A0}t_{B0} \). It entails:

\[
M_{\text{deadlock}} = M_0 + A \cdot \begin{bmatrix}
1 \\
0 \\
1 \\
0 \\
0
\end{bmatrix} = \begin{bmatrix}
1 \\
0 \\
1 \\
0 \\
0
\end{bmatrix} + \begin{bmatrix}
-1 \\
1 \\
-1 \\
1 \\
0
\end{bmatrix} = \begin{bmatrix}
0 \\
1 \\
0 \\
1 \\
0
\end{bmatrix}
\]

As expected, we find again the blocking marking from the reachability graph of the previous question.

b) From the Petri net structure, we get:

\[
M_0 = (1,0,1,1,1,0,0) (0,1,0,0,0,1,0)
\]

\[
A = W^+ - W^- = \begin{bmatrix}
-1 & 0 & 1 & 0 & 0 & 0 \\
1 & -1 & 0 & 0 & 0 & 0 \\
0 & 1 & -1 & 0 & 0 & 0 \\
-1 & 0 & 1 & 0 & -1 & 1 \\
0 & -1 & 1 & -1 & 0 & 1 \\
0 & 0 & 0 & -1 & 0 & 1 \\
0 & 0 & 0 & 1 & -1 & 0 \\
0 & 0 & 0 & 0 & 1 & -1
\end{bmatrix}
\]

Consider the basic blocking sequence \( t_{A0}t_{B0} \). It entails:

\[
M_{\text{deadlock}} = M_0 + A \cdot \begin{bmatrix}
1 \\
0 \\
1 \\
0 \\
0
\end{bmatrix} + \begin{bmatrix}
-1 \\
1 \\
-1 \\
1 \\
0
\end{bmatrix} = \begin{bmatrix}
0 \\
1 \\
0 \\
1 \\
0
\end{bmatrix}
\]

As expected, we find again the blocking marking from the reachability graph of the previous question.

c) A deadlock state is a state at which no transition in enabled. Hence, one can use the upstream transition matrix \( W^- \) to assess whether or not a marking is blocking. It is the case if and only if the marking vector does not **cover** (i.e., is strictly bigger than...) any of \( W^- \) column. Otherwise, it implies the transition associated to this column is enabled, hence this marking is not blocking.

d) In order to avoid that deadlock, we need to forbid both process to run concurrently. This can be solved easily using a semaphore, as illustrated thereafter:
4 From mutual exclusion to starvation

Your task is to model a system as a Petri net in which two processes want to access a common exclusive resource is a similar fashion as in Exercise 3. This means that the two processes have to exclude each other mutually from the concurrent access to the resource (e.g. a critical program section). More precisely:

1. A process executes its program.
2. In order to enter the critical section, a given mutex variable must be 0.
3. If this is the case, the process sets the mutex to 1 and executes its critical section.
4. When done, it resets the mutex to 0 and enters an uncritical section.
5. Then the procedure starts all over again.

a) Propose a Petri net representing the desired behavior.

\textit{Hint:} use 5 places and 4 transitions.

b) In this setting, it may happen that a process starves the other. That means one process always uses the resource and the other never enters the critical section. Correct this in such a way that each process cannot get the resource more than twice in a row. This may yield that only one process can start running from the initial marking.

a) For each process we introduce two places \((p_1, p_2, p_3, p_4)\) representing the process within the normal program execution \((p_1, p_2)\) as well as in the critical section \((p_3, p_4)\). For each process, we have a token indicating which section of the program is currently executed. Additionally, we introduce a place \(p_0\) representing the mutex variable. If the mutex variable is 0, then we have a token at \(p_0\). We have to make sure that a process can only enter its critical section if there is a token at the mutex place. The resulting Petri net looks as follows.
Assume that initially, both processes are in a non-critical section (in the Petri net, this is denoted by a token in place $p_1$ and $p_2$ respectively). A process can only enter its critical section ($p_3/p_4$) if there is a token at $p_0$. In this case, the token is consumed when entering the critical section. A new mutex token at $p_0$ is not created until the process leaves its critical section. Hence, both processes exclude each other mutually from the concurrent access to the critical section.

This is a classical benefit of Petri nets over other DES models. It models very efficiently the sharing of resources, the concurrency of processes, and so on...

b) In order to avoid starvation of either of the processes, one option is to count the number of execution the each of them, or more precisely the difference between them. Assume that at initial state, none has been previously run. According to the specification, we can allow one to the process (say $A$) to run twice by creating a "counter-resource" with 2 tokens at initial state. Running the process $A$ consume one of these tokens and a new token will be generated in this place on completion of process $B$. Doing that symmetrically (on each process) yields the number of executions of each process together...

Lots of rather obscure explanations. Want the net?
5 Reachability Analysis for Petri Nets

In the lecture we presented an algorithm to perform a reachability analysis on Petri nets.

a) Why is it not possible with a reachability algorithm to determine in general, whether a given state in a Petri net is reachable or not?

b) Consider the Petri net $N_2$ from exercise 2. Is the state $s = (p_1 = 101, p_2 = 99, p_3 = 4)$ reachable from the initial state $s_0 = (1, 0, 0)$ if $k = 2$? Prove your answer.

Hint: Start with the necessary condition presented in the lecture for the reachability of a state in a weighted Petri net, then eventually explain whether or not the marking is reachable.

a) Petri nets may possess infinite reachability graphs, e.g. $N_2$ with $k \geq 2$. If the state in question is actually reachable in such a Petri net, the reachability algorithm will eventually terminate. If it is not reachable, the algorithm will never be able to determine this with absolute certainty (cf. halting problem).

b) We determine the incidence matrix of the Petri net as explained in the lecture.

$$A = \begin{pmatrix} -1 & 1 & 0 & 2 \\ 1 & -1 & -1 & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix}$$

We are interested in whether the state $\vec{s} = (101, 99, 4)$ is reachable from the initial state $\vec{s}_0 = (1, 0, 0)$. If the equation system $A \cdot \vec{f} = \vec{s} - \vec{s}_0$ has no solution, we know that the state $\vec{s}$ is not reachable from $\vec{s}_0$. "Unfortunately",

$$\begin{pmatrix} -1 & 1 & 0 & 2 \\ 1 & -1 & -1 & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix} = \begin{pmatrix} 100 \\ 99 \\ 4 \end{pmatrix}$$

is satisfiable. To show that $\vec{s}$ is reachable from $\vec{s}_0$, we have to give a firing sequence through which we get from $\vec{s}_0$ to $\vec{s}$. From the last equation of the above equation system, we know that $f_3 = f_4 + 4$. Hence, in the desired firing sequence, $f_3$ is fired four times more than $f_4$. However, $\vec{f}$ does not tell us about the firing order. Considering the Petri net, we can see that – starting from $\vec{s}_0$ – the number of tokens in $p_1$ increases by one after firing $t_1$, $t_3$, and $t_4$ in this order. Repeating this for 203 times yields the state $(204, 0, 0)$. Firing $t_1$ for 103 times followed by firing $t_3$ for four times finally yields state $\vec{s}$. 

6 Coverability tree and graph

Given is the following Petri net \( N_6 \), compute its coverability tree and coverability graph. Deduce which are the unbounded places of this net given the initial marking.

Following the procedure from the lecture note, we can construct the following coverability tree:

\[
M_0 = \begin{array}{c}
(1,0,0) \\
\text{t}_1 \\
(1,\omega,0) \\
\text{t}_2 \\
(1,\omega,0) \\
\text{t}_3 \\
(1,\omega,0) \\
\text{t}_1 \\
(1,\omega,0,\omega) \\
\text{t}_2 \\
(1,\omega,0,\omega) \\
\text{t}_3 \\
(1,\omega,0,\omega) \\
\text{t}_1 \\
(1,\omega,0,\omega) \\
\text{t}_2, t_3 \\
(1,\omega,0,\omega) \\
\text{t}_1, t_2, t_3
\end{array}
\]

One can merge the equivalent node and obtain the coverability graph:

\[
M_0 = \begin{array}{c}
(1,0,0) \\
\text{t}_1 \\
(1,\omega,0) \\
\text{t}_2 \\
(1,\omega,0) \\
\text{t}_3 \\
(1,\omega,0) \\
\text{t}_1 \\
(1,\omega,0) \\
\text{t}_2, t_3 \\
(1,\omega,0) \\
\text{t}_1, t_2, t_3
\end{array}
\]

It follows that for this net with initial marking \((1,0,0)\), places \( p_2 \) and \( p_3 \) are unbounded.