Fuzzy knowledge in automatic solving the problem of manufacturing systems control

František Čapkovič
Institute of Control Theory and Robotics, Slovak Academy of Sciences
Dúbravská cesta 9, 842 37 Bratislava, Slovak Republic
Phone: +421-7-5941 2544, Fax: +421-7-5477 6045, E-mail: utrrcapk@nic.savba.sk
http://www.savba.sk/~utrrcapk/capkhome.htm

Abstract

A new view on manufacturing systems (MS) modelling and control synthesis is presented. MS are understood here to be a kind of DEDS (discrete event dynamic system). Petri nets (PN), frequently used for DEDS modelling, are replaced here by a form of corresponding oriented graphs (OG), where the nodes of the OG are represented by the PN positions and the OG edges involve the PN transitions. The adjacency matrix of such a graph is utilized in the DEDS control synthesis procedure. It helps to generate the state reachability tree in both the straight-lined system development (from an initial state to a prescribed terminal one) and that of the backtracking one (from the terminal state to the initial one). To perform the DEDS control synthesis combinig both of kinds of the model development is used. The coincidence both of the state reachability trees yields the possible trajectories of the system development. In such a way all solutions how to reach the prescribed terminal state from the given initial one are automatically found. To choose the most suitable solution rule-based knowledge about the control task specifications is utilized. It may be fuzzy too.

Keywords: Automatic solving, control system synthesis, discrete-event dynamic systems, manufacturing systems, knowledge representation, Petri nets.

1 Introduction

Any model of a system to be controlled does not yield immediatelly the sequence of control interferences that are necessary in order to reach a prescribed terminal state of the system starting form a given initial state at fulfilling prescribed control task specifications like criteria, constraints, etc. A control synthesis procedure is necessary on this way. There are many successful methods of control synthesis suitable for the continuous-time systems (CTS) or/and discrete-time systems (DTS). However, usually they are not usable for solving the DEDS control synthesis problem. Namely, DEDS are completly different (as to the principle of their dynamic behaviour) from the CTS and DTS. Consequently, they also need different methods for their modelling and control. PN-based models of DEDS are used very frequently. To solve the control synthesis problem more effectively and to make it fully automatic the OG-based model of the system corresponding with the PN-based one (presented e.g. in [1], [2]) is used. The possibility of creating the OG-based k-variant model of DEDS corresponding to the PN-based k-invariant one was pointed out in the author's paper [9] and (in an extended form) in [10]. The main principle of a simple straight-lined control system synthesis was also presented in the latter one. Because of such an "one-way" approach the control synthesis was not fully automatic, but only automated. The idea of the backtracking approach as well as that of combining both the staight-lined approach and the backtracking one represents the main contribution of this paper. It makes the automatic solving the control synthesis problem possible.

From the system theory point of view MS are a kind of DEDS because they consist of many cooperating subsystems with many conflicts among them. Their behaviour is influenced by occurring discrete events that start or stop activities of the subsystems. In the other words, DEDS are asynchronouos systems with concurrency or/and parallelism among the activities of their subsystems. Usually, they are large-scale or/and complex. Other representants of DEDS are flexible manufacturing systems (FMS), transport systems, different kinds of communication systems, etc. Because DEDS are very important in human practice, the demand of the successful and efficient control of them is very actual. Special kinds of systems usually need special kinds of approaches to the system modelling and control synthesis. The control task specifications (like constraints, criteria, etc.) for DEDS are usualy given verbally or in another form of nonanalytical terms. The main problem of the DEDS control synthesis is:

- to express the DEDS model in the most suitable form in order to have the description of the system behaviour at disposal
- to express the control task specifications to be met in the most suitable form of a knowledge base in order to satisfy them properly
- to choose the most suitable solution of the control synthesis problem.

Just the model-based analysing the behaviour of the system to be controlled with respect to knowledge about the control task specifications connected with both the straight-lined search of the control possibilities and the backtracking one yield the automatic solving of the control synthesis problem.

2 The straight-lined development of the system behaviour

Consider the k-variant OG-based model of DEDS introduced in [9] in the form

$$\mathbf{x}_{k+1} = \mathbf{\Delta}_k.\mathbf{x}_k \quad , \quad k = 0, N \tag{1}$$

where

k is the discrete step of the DEDS dynamics development.

$\mathbf{x}_k = (x_1^{(k)}, ..., x_n^{(k)})^T$; $k = 0, N$ is the n- dimensional state vector of the DEDS in the step k; $x_i^{(k)}$, $i = 1, n$ is the state of the elementary subprocess in the step k. Its activity is expressed by 1 and its passivity by 0 (in the PN analogy it is the state of the elementary position p_i).

$\mathbf{\Delta}_k = \{\delta_{ij}^{(k)}\}$, $\delta_{ij}^{(k)} = \gamma_{t_{p_i|p_j}}^{(k)} \in \{0, 1\}$, $i = 1, n$; $j = 1, n$, because the corresponding set $\Delta_k \subseteq (X \times U) \times (U \times X)$ where X is the set of the PN positions p_i , $i = 1, n$ and U is the set of the PN transitions t_j , $j = 1, m$. The matrix expresses the causal relations between the subprocesses depending on the occurrence of the discrete events. The element $\delta_{ij}^{(k)} = \gamma_{t_{p_i|p_j}}^{(k)} \in \{0, 1\}$ expresses the actual value of the transition function of the PN transition fixed on the OG edge oriented from the node p_j to the node p_i. It can be seen that the matrix Δ_k is the transpose of a "functional" adjacency matrix. Namely, its elements are not integers but two-valued transition functions of the elementary PN transitions.

The introduced model corresponds to the PN-based model of DEDS, the simplest form of which (expressed in analytical terms) is the following

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{B}.\mathbf{u}_k \,, \quad \mathbf{B} = \mathbf{G}^T - \mathbf{F} \,, \quad k = 0, N \tag{2}$$

$$\mathbf{F}.\mathbf{u}_k \leq \mathbf{x}_k \tag{3}$$

where

k is the discrete step of the system dynamics development.

$\mathbf{x}_k = (\sigma_{p_1}^{(k)}, ..., \sigma_{p_n}^{(k)})^T$; $k = 0, N$ is the n- dimensional state vector of the DEDS in the step k; $\sigma_{p_i}^{(k)}$, $i = 1, n$ is the state of the elementary subprocess p_i in the step k. Its values express the states of the DEDS elementary subprocesses. They acquire their values from the set $\{0, 1\}$ where 0 expresses the passivity and 1 expresses the activity of the corresponding subprocess.

$\mathbf{u}_k = (\gamma_{t_1}^{(k)}, ..., \gamma_{t_m}^{(k)})^T$ is the m-dimensional control vector of the system in the step k. Its components $\gamma_{t_j}^{(k)}$, $j = 1$, m, represent the states of occurring the DEDS elementary discrete events (e.g. starting or ending the elementary subprocesses or other activities). They acquire their values from the set $\{0, 1\}$ where 1 expresses the presence and 0 expresses the absence of the corresponding discrete event.

$\mathbf{B}$, $\mathbf{F}$, $\mathbf{G}$ are, respectively, $(n \times m)$, $(n \times m)$ and $(m \times n)$- dimensional structural matrices of constant elements. The matrices $\mathbf{F}$, $\mathbf{G}$ are the incidence matrices (in the analogy with the incidence matrices of the mutual oriented interconnections among the PN positions and transitions) expressing the mutual causal relations among the DEDS subprocesses and the discrete events. The incidence matrix $\mathbf{F}$ expresses the causal relations oriented from the states of the DEDS subprocesses to the discrete events occuring during the DEDS operation. The incidence matrix $\mathbf{G}$ expresses the causal relation oriented from the discrete events to the states of the DEDS subprocesses. The elements of these matrices acquire their values from the set $\{0,1\}$ where 1 expresses the existence and 0 expresses the nonexistence of the corresponding causal relation.

$(.)^T$ symbolizes the matrix or vector transposition.

In order to express the equation (1) more exactly, let us use the following form of its description

$$\{\mathbf{x}_{k+1}\} = \mathbf{\Delta}_k.\{\mathbf{x}_k\}\ ,\ k = 0, N-1 \tag{4}$$

where $\{\mathbf{x}_{k+1}\}$ is an aggregate of all of the states that are reachable from the previous states $\{\mathbf{x}_k\}$ in one step k. There is only one exception $\{\mathbf{x}_0\} = \mathbf{x}_0$, because the initial state is only single. Let us develop the system in the straight-lined orientation. Hence,

$$\begin{aligned}
\{\mathbf{x}_1\} &= \mathbf{\Delta}_0.\mathbf{x}_0 && (5)\\
\{\mathbf{x}_2\} &= \mathbf{\Delta}_1.\{\mathbf{x}_1\} = \mathbf{\Delta}_1.\mathbf{\Delta}_0.\mathbf{x}_0 && (6)\\
&\vdots\quad\vdots\quad\vdots\\
\{\mathbf{x}_k\} &= \mathbf{\Delta}_{k-1}.\{\mathbf{x}_{k-1}\} = \mathbf{\Delta}_{k-1}.\mathbf{\Delta}_{k-2}.\mathbf{\Delta}_1.\mathbf{\Delta}_0.\mathbf{x}_0\\
\{\mathbf{x}_k\} &= \mathbf{\Phi}_{k,0}.\mathbf{x}_0 && (7)\\
\mathbf{\Phi}_{k,j} &= \prod_{i=j}^{k-1} \mathbf{\Delta}_i\ ;\quad j = 0, k-1 && (8)
\end{aligned}$$

The multiplying is made from the left because of the causality principle. It must be said that the meaning of the multiplying and additioning operators in the development of the k-variant model have symbolic interpretation. For example, an element $\phi_{i,j}^{k,0}$, $i = 1, n$; $j = 1, n$ of the transition matrix $\mathbf{\Phi}_{k,0}$ is either a product of k elements (the transition functions expressing the "trajectory" containing the sequence of elementary transitions that must be fired in order to go from the initial elementary state x_j^0 into the final state $x_i^{(k)}$) or a sum of several such product (when there are several "trajectories" from the initial state to final one). It can be said that any nonzero element $\delta_{ij}^{(k)}$ of the matrix $\mathbf{\Delta}_k$ gives us information about reachability of the state $\sigma_{p_i}^{k+1}$ from the state $\sigma_{p_j}^{(k)}$. Hence, any element $\phi_{i,j}^{k2,k1}$ of the transition matrix $\mathbf{\Phi}_{k2,k1}$ gives us information about the reachability of the state $\sigma_{p_i}^{k2}$ from the state $\sigma_{p_j}^{k1}$.

When the input vector $\mathbf{x}_k$ represents a state of the system and we do not know the actual state of the transition functions (i.e. the actual state of the functional elements $\delta_{ij}^{(k)}$ of the k-variant matrix $\mathbf{\Delta}_k$), we can use the transpose of the OG adjacency matrix - i.e. the matrix $\mathbf{\Delta} = \{\delta_{ij}\}$ having the same structure like the matrix $\mathbf{\Delta}_k = \{\delta_{ij}^{(k)}\}$, however its elements are not functional but they are defined as follows

$$\mathbf{\Delta} = \{\delta_{ij}\};\quad \delta_{ij} = \begin{cases} 1 & \text{if } \delta_{ij}^{(k)} \neq 0 \\ 0 & \text{otherwise} \end{cases}\ ;\ i = 1, n; j = 1, n \tag{9}$$

It means that $\mathbf{\Delta}$ is the constant matrix with all elements corresponding with the functional elements of the matrix $\mathbf{\Delta}_k$ equal to 1. Thus, in the below system development we can understand that $\mathbf{\Delta}_k = \mathbf{\Delta}$, $k = 0, N-1$.

3 The backtracking development of the system behaviour

The procedure very analogical to that starting from the initial state $\mathbf{x}_0$ can start from the terminal state $\mathbf{x}_t$. Let us denote the terminal state as $\mathbf{x}_N$. The aggregate state vector from which the terminal state is reachable can be expressed as follows

$$\{\mathbf{x}_{N-k-1}\} = \mathbf{\Delta}_{N-k-1}^T.\{\mathbf{x}_{N-k}\},\quad k = 0, N-1$$

where $\{\mathbf{x}_{N-k-1}\}$ is an aggregate of all of the states from which the states $\{\mathbf{x}_{N-k}\}$ are reachable in one step k. There is only one exception $\{\mathbf{x}_N\} = \mathbf{x}_N$, because the terminal state is only single. Consequently, the backtracking system development is the following

$$\begin{aligned}
\{\mathbf{x}_{N-1}\} &= \mathbf{\Delta}_{N-1}^T.\mathbf{x}_N && (10)\\
\{\mathbf{x}_{N-2}\} &= \mathbf{\Delta}_{N-2}^T.\{\mathbf{x}_{N-1}\} = \mathbf{\Delta}_{N-2}^T\mathbf{\Delta}_{N-1}^T.\mathbf{x}_N
\end{aligned}$$

$$\begin{array}{rcl} \vdots & \vdots & \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \\ \{\mathbf{x}_0\} & = & \mathbf{\Delta}_0^T.\{\mathbf{x}_1\} = \mathbf{\Delta}_0^T\mathbf{\Delta}_1^T \dots \mathbf{\Delta}_{N-2}^T\mathbf{\Delta}_{N-1}^T.\mathbf{x}_N \\ \{\mathbf{x}_0\} & = & \mathbf{\Phi}_{k,0}^T.\mathbf{x}_k \end{array} \tag{11}$$

4 The control synthesis procedure

To synthetize the DEDS control (to find the most suitable sequence of the control vectors $\{\mathbf{u}_0, \mathbf{u}_1, \dots, \mathbf{u}_{N-1}\}$ transforming the system from the prescribed initial state $\mathbf{x}_0$ into the given terminal state $\mathbf{x}_t = \mathbf{x}_N$ at the simultaneous fulfilling the prescribed control task specifications like criteria, constraints, etc.), several approaches can be used. The main problem of the control synthesis is that usually there are several possibilities how to proceed in any step of the system dynamics development. Consequently, a tree of the possibilities of the system behaviour is possible. The main idea of the approach presented in this paper is to avoid the actual tree construction. Simultaneous utilizing both the straigth-lined approach and the backtracking one reduces the amount of computations. The procedure is the following:

- to proceed from the initial state $\mathbf{x}_0$ by the straight-lined approach
- to proceed simultaneously from the desirable terminal state $\mathbf{x}_N$ by the backtracking one
- to compare the actual trajectories (the aggregated states obtained by means of the straight-lined procedure and that obtained by means of the backtracking one). In such a way all possible trajectories from the given initial state to the desirable terminal one are found.

What is important is that the state reachability trees need not be generated in the form of graphs. It is sufficient to work with the numerical adjacency matrix $\mathbf{\Delta}$ of the graph modelling the DEDS. Namely, it is the non-negative matrix defined e.g. in [12, 16, 18]. The necessary condition for the above procedure has to be fulfiled - namely, the terminal state $\mathbf{x}_N$ must be reachable from the initial state $\mathbf{x}_0$. It is not very difficult to test the reachability. For such a testing the result of the proved theorem [12, 14, 15] can be used. The test is based on computation of the k-th power (where k is unknown before) of the OG adjacency matrix. Because we use the transpose $\mathbf{\Delta}$ of the original adjacency matrix, to obtain $\mathbf{\Delta}^k$ containing the first nonzero element $\delta_{ij}^{(k)}$ we have to multiply the matrix $\mathbf{\Delta}$ from the left. The nonzero element $\delta_{ij}^{(k)}$ of the matrix $\mathbf{\Delta}^k$ gives us information not only about the reachability of the i-th element of the state vector $\mathbf{x}_k$ from the j-th element of the state vector $\mathbf{x}_0$ but also about the number of the steps k that have to be performed (more mathematical details can be found in literature [12, 14, 15]). How long is it necessary to compute the powers of the matrix? The question is answered in [15] - the exponent $k \leq n-1$.

4.1 A discussion about the exponent k for the indecomposable adjacency matrix

At the guess of the exponent k in special cases the results published in [16, 18] concerning indecomposable non-negative $n \times n$-dimensional matrices can be utilized. The indecomposable matrix is defined in [12] as the matrix which is not decomposable. The matrix $\mathbf{A}$ is decomposable if it is of the following form (12) or if there exists a permutation matrix $\mathbf{P}$ such that $\mathbf{P}^T\mathbf{A}\mathbf{P}$ is of the form (12)

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \emptyset & \mathbf{A}_{22} \end{pmatrix} \tag{12}$$

where submatrices $\mathbf{A}_{ii}$, $i = 1, 2$ are square matrices. A real matrix $\mathbf{R}$ is non-negative if all its elements $r_{ij} \geq 0$. When a power of the indecomposable non-negative $n \times n$-dimensional matrix is a positive matrix (a real matrix $\mathbf{Q}$ is positive if all its elements $q_{ij} > 0$), the matrix is primitive (as it is mentioned in [16] this term was introduced by Frobenius in 1912). For the best upper bound of the exponent k_{min} guaranting that corresponding power of the primitive matrix (for $n \geq 2$) is positive the inequality $k_{min} \leq \omega_n = (n-1)^2+1$ is valid - see [16, 18]. Hence, at least one common state can be found by both the straight-lined procedure and the backtracking one. The proposed combined approach to the control synthesis seems to be very powerful for a wide class of PN.

There are some new results for $n \geq 3$ in [13] concerning the minimal exponent. It is proved that if $k_{min} \geq \lfloor \omega_n/2 \rfloor + 2$ then the primitive *directed* graph has cycles of exactly two different lengths i, j with $n \geq j > i$. Because [11] and some others distinguishe the *oriented* and *directed* graphs (the oriented graph is understood in [11] to be a directed graph having no symmetric pair of directed edges or in other words the directed graphs without loops or multiple edges) the new results are applicable for our case only partially. In addition to this the OG adjacency matrix in general can be decomposable (and consequently it will be imprimitive). However, the considerations analogical to indecomposable matrix $\mathbf{A}$ can be done for the indecomposable submatrices $\mathbf{A}_{ii}$, $i = 1, 2$ of the decomposable matrix $\mathbf{A}$.

5 A simple example

Let us demonstrate the presented approach in details. Many particulars can be explained in such a way. Consider a simple case of DEDS the PN-based model of which is given on the left side of Fig. 1. The terminal

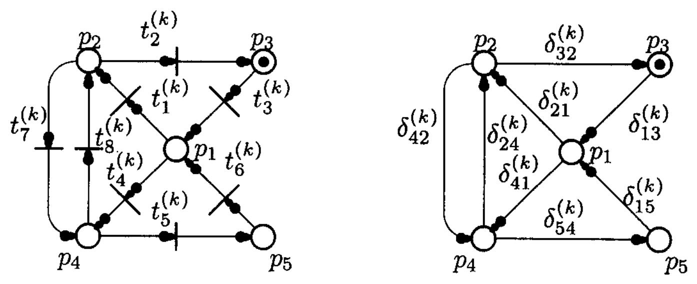

Figure 1: The PN-based and OG-based models

state vector $\mathbf{x}_N$ is equal to the initial one

$$\mathbf{x}_0 = (0\,0\,1\,0\,0)^T \tag{13}$$

The structure of the control vector is

$$\mathbf{u}_k = (t_1^{(k)}, t_2^{(k)}, t_3^{(k)}, t_4^{(k)}, t_5^{(k)}, t_6^{(k)}, t_7^{(k)}, t_8^{(k)})^T \quad ; \quad t_i^{(k)} \in \{0, 1\},\ i = 1, 8 \tag{14}$$

The parameters of the PN-based model are $\qquad n = 5 \qquad m = 8$

$$\mathbf{F} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \quad \mathbf{G} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

At the construction of the OG-based model given on the right side of Fig. 1 consider the elementary transitions as the elements $\delta_{ij}^{(k)}$ of the matrix $\mathbf{\Delta}_k$ as follows

$$\mathbf{\Delta}_k = \begin{pmatrix} 0 & 0 & t_3^{(k)} & 0 & t_6^{(k)} \\ t_1^{(k)} & 0 & 0 & t_8^{(k)} & 0 \\ 0 & t_2^{(k)} & 0 & 0 & 0 \\ t_4^{(k)} & t_7^{(k)} & 0 & 0 & 0 \\ 0 & 0 & 0 & t_5^{(k)} & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & \delta_{13}^{(k)} & 0 & \delta_{15}^{(k)} \\ \delta_{21}^{(k)} & 0 & 0 & \delta_{24}^{(k)} & 0 \\ 0 & \delta_{32}^{(k)} & 0 & 0 & 0 \\ \delta_{41}^{(k)} & \delta_{42}^{(k)} & 0 & 0 & 0 \\ 0 & 0 & 0 & \delta_{54}^{(k)} & 0 \end{pmatrix}$$

To avoid using the complicated transition matrices we can utilize the powers of the numerical adjacency matrix in order to find that after three, four, five, etc. steps from the initial state the terminal state (the same like the initial one) is reachable.

$$\mathbf{\Delta} = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} ; \mathbf{\Delta}^2 = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix} ; \mathbf{\Delta}^3 = \begin{pmatrix} 2 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 2 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

$$\mathbf{\Delta}^4 = \begin{pmatrix} 2 & 1 & 2 & 1 & 2 \\ 3 & 3 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 & 1 \\ 3 & 2 & 1 & 3 & 1 \\ 1 & 2 & 1 & 1 & 1 \end{pmatrix}; \mathbf{\Delta}^5 = \begin{pmatrix} 2 & 3 & 2 & 3 & 2 \\ 5 & 3 & 3 & 4 & 3 \\ 3 & 3 & 1 & 2 & 1 \\ 5 & 4 & 3 & 3 & 3 \\ 3 & 2 & 1 & 3 & 1 \end{pmatrix}; \mathbf{\Delta}^6 = \begin{pmatrix} 6 & 5 & 2 & 5 & 2 \\ 7 & 7 & 5 & 6 & 5 \\ 5 & 3 & 3 & 4 & 3 \\ 7 & 6 & 5 & 7 & 5 \\ 5 & 4 & 3 & 3 & 3 \end{pmatrix}$$

By the way, we can see that the 4-th power of the matrix $\mathbf{\Delta}$ is positive. However, in spite of this it cannot be said that it is primitive because it is decomposable. In Tab. 1 we have the straight-lined sequence of the aggregated states (the left part of the table) and the backtracking sequence of the aggregated states (the right part of the table).

Table 1: The straight-lined system development (on the left) and the backtracking one (on the right)

x_0	$\{\mathrm{x}_1\}$	$\{\mathrm{x}_2\}$	$\{\mathrm{x}_3\}$	$\{\mathrm{x}_4\}$	$\{\mathrm{x}_5\}$	x_0	$\{\mathrm{x}_1\}$	$\{\mathrm{x}_2\}$	$\{\mathrm{x}_3\}$	$\{\mathrm{x}_4\}$	$\{\mathrm{x}_5\}$
0	1	0	0	2	2	3	1	1	1	0	0
0	0	1	1	1	3	3	1	1	0	1	0
1	0	0	1	1	1	1	1	1	0	0	1
0	0	1	1	1	3	2	2	0	1	0	0
0	0	0	1	1	1	1	1	1	0	0	0

After comparing the corresponding columns both of the previous parts of the Tab. 1 their coincidence is given in Tab. 2. The tables are expressed graphically on Fig. 2 and Fig. 3 respectively. By means of

Table 2: The resulting trajectory

x_0	$\{\mathrm{x}_1\}$	$\{\mathrm{x}_2\}$	$\{\mathrm{x}_3\}$	$\{\mathrm{x}_4\}$	$\{\mathrm{x}_5\}$
0	1	0	0	0	0
0	0	1	0	1	0
1	0	0	0	0	1
0	0	0	1	0	0
0	0	0	0	0	0

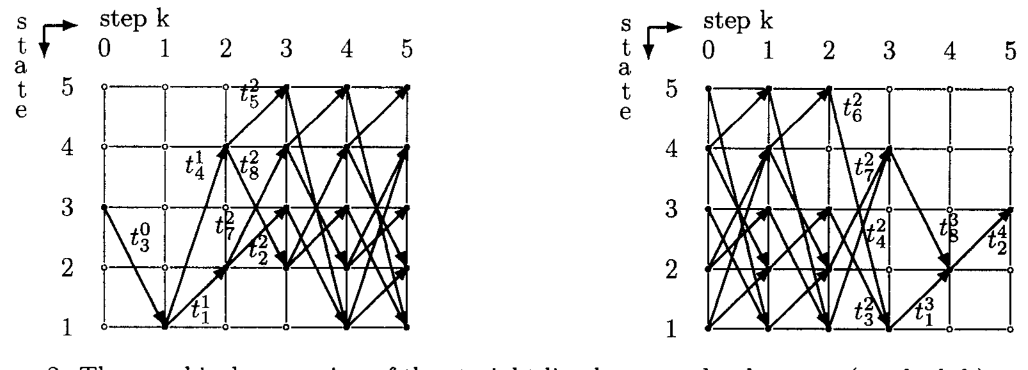

Figure 2: The graphical expression of the straight-lined system development (on the left) and the backtracking one (on the right)

comparing both the straight-lined development and the backtracking one we can find three possible solutions of the control synthesis problem. The 5-step solution (when the corresponding columns of the left and right parts of the Tab. 1 coincide), the 4-step solution (when the columns of the right part of the Tab. 1 coincide with the columns of its left part shifted to the right for one column) and the 3-step solution (when the

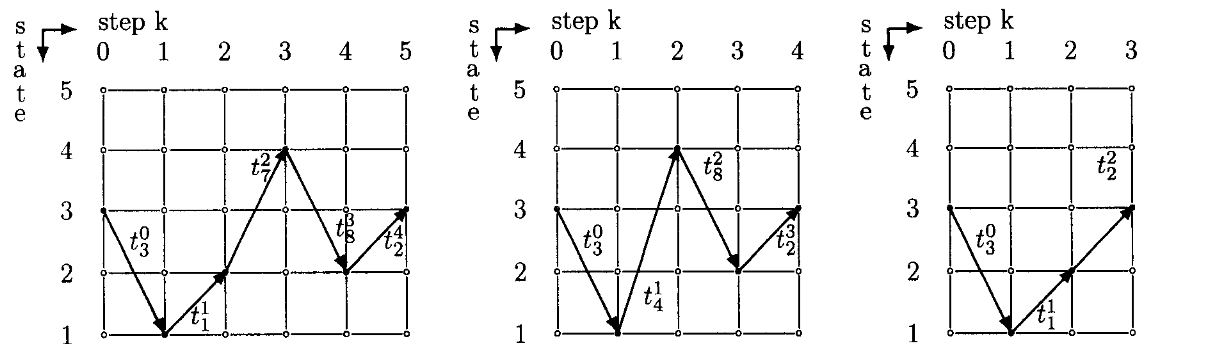

Figure 3: The 5-step solution (on the left), the 4-step solution (in the center) and the 3-step one (on the right) of the control synthesis problem

columns of the right part of the Tab. 1 coincide with the columns of its left part shifted to the right for two columns). These solutions are given on Fig. 3. In order to denote the elementary sections of trajectories by the transitions that have to be fired, the adjacency matrix $\mathbf{\Delta}_k$ can be utilized. It stores information about the placement of the transitions. Consequently, the sequence of the control vectors for the 5-step solution is the following

$$\begin{aligned}\mathbf{u}_0 &= (0,0,1,0,0,0,0,0)^T \; ; \; \mathbf{u}_1 = (1,0,0,0,0,0,0,0)^T \; ; \; \mathbf{u}_2 = (0,0,0,0,0,0,1,0)^T \\ \mathbf{u}_3 &= (0,0,0,0,0,0,0,1)^T \; ; \; \mathbf{u}_4 = (0,1,0,0,0,0,0,0)^T\end{aligned}$$

6 An illustration of solving the complex control synthesis problem

Consider the maze problem introduced by Ramadge and Wonham in [19] and completly solved by author's PN-based approaches [1, 2, 3, 4, 17]. Two "participants" - in [19] a cat and a mouse - can be as well e.g. two mobile robots or two automatically guided vehicles (AGVs) of the FMS, two cars on a complicated crossroad, two trains in a railway network, etc. They are placed in the maze (however, it can also be e.g. the complicated crossroad, etc.) given on Fig. 4 consisting of five rooms denoted by numbers 1, 2,..., 5

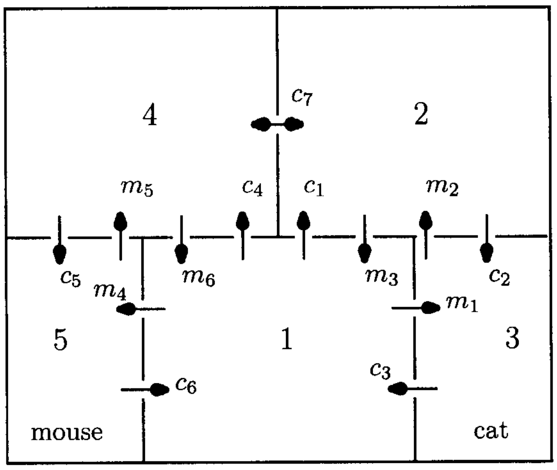

Figure 4: The maze structure.

connecting by the doorways exclusively for the cat denoted by $c_i, i = 1,7$ and the doorways exclusively for the mouse denoted by $m_j, j = 1,6$. The cat is initially in the room 3 and the mouse in the room 5. Each doorway can be traversed only in the direction indicated. Each door (with the exception of the door c_7) can be opened or closed by means of control actions. The door c_7 is uncontrollable (or better, it is continuously open in both directions). The controller to be synthetized observes only discrete events generated by sensors in the doors. They indicate that a participant ist just running through. The control problem is to find a feedback controller (e.g. an automatic pointsman or switchman in railways) such that the following control task specifications - three criteria or/and constraints will be satisfied:

1. The participants never occupy the same room simultaneously.
2. It is always possible for both of them to return to their initial positions (the first one to the room 3 and the second one to the room 5).
3. The controller should enable the participants to behave as freely as possible with respect to the constraints imposed.

At the construction of the PN-based model of the system the rooms 1 - 5 of the maze will be represented by the PN positions p_1 - p_5 and the doorways will be represented by the PN transitions. The permanently open door c_7 is replaced by means of two PN transitions t_7 and t_8 symbolically denoted as $c_7^{(k)}$ and $c_8^{(k)}$. Both the PN-based representation of the maze and the OG-based one are given on Fig. 5 and Fig. 6. Let

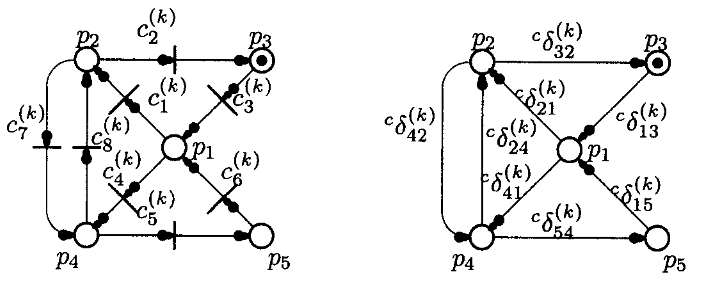

Figure 5: The PN-based and OG-based model of the cat

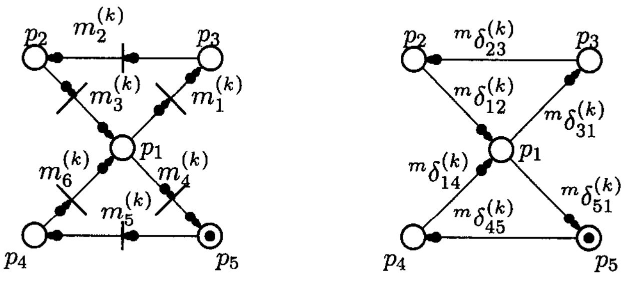

Figure 6: The PN-based and OG-based model of the mouse

us demonstrate the presented approach in details on this case of DEDS. Many particulars can be explained in such a way. The terminal state vector of the cat ${}^c\mathbf{x}_N$ is equal to the initial one

$$ {}^c\mathbf{x}_N = {}^c\mathbf{x}_0 = (0\,0\,1\,0\,0)^T \tag{15}$$

The terminal state vector of the mouse ${}^m\mathbf{x}_N$ is equal to the initial one

$$ {}^m\mathbf{x}_N = {}^m\mathbf{x}_0 = (0\,0\,0\,0\,1)^T \tag{16}$$

The structure of the cat control vector is

$$^{c}\mathbf{u}_k = (c_1^{(k)},\, c_2^{(k)},\, c_3^{(k)},\, c_4^{(k)},\, c_5^{(k)},\, c_6^{(k)},\, c_7^{(k)},\, c_8^{(k)})^T \; ; \; c_i^{(k)} \in \{0,1\}, \;\; i = 1,8 \qquad (17)$$

The structure of the mouse control vector is

$$^{m}\mathbf{u}_k = (m_1^{(k)},\, m_2^{(k)},\, m_3^{(k)},\, m_4^{(k)},\, m_5^{(k)},\, m_6^{(k)})^T \; ; \; m_i^{(k)} \in \{0,1\}, \;\; i = 1,6 \qquad (18)$$

The parameters of the PN-based model of the cat movement possibilities are

$$n = 5 \qquad m = 8$$

$$^{c}\mathbf{F} = \begin{pmatrix} 1&0&0&1&0&0&0&0 \\ 0&1&0&0&0&0&1&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&0&1&0&0&1 \\ 0&0&0&0&0&1&0&0 \end{pmatrix} \; ^{c}\mathbf{G} = \begin{pmatrix} 0&1&0&0&0 \\ 0&0&1&0&0 \\ 1&0&0&0&0 \\ 0&0&0&1&0 \\ 0&0&0&0&1 \\ 1&0&0&0&0 \\ 0&0&0&1&0 \\ 0&1&0&0&0 \end{pmatrix}$$

The parameters of the PN-based model of the mouse movement possibilities are

$$n = 5 \qquad m = 6$$

$$^{m}\mathbf{F} = \begin{pmatrix} 1&0&0&1&0&0 \\ 0&0&1&0&0&0 \\ 0&1&0&0&0&0 \\ 0&0&0&0&0&1 \\ 0&0&0&0&1&0 \end{pmatrix} \; ^{m}\mathbf{G} = \begin{pmatrix} 0&0&1&0&0 \\ 0&1&0&0&0 \\ 1&0&0&0&0 \\ 0&0&0&0&1 \\ 0&0&0&1&0 \\ 1&0&0&0&0 \end{pmatrix}$$

At the construction of the OG-based models consider the elementary transitions as the elements $\delta_{ij}^{(k)}$ of the matrix $\mathbf{\Delta}_k$ as follows. The matrix for the cat is

$$^{c}\mathbf{\Delta}_k = \begin{pmatrix} 0 & 0 & c_3^{(k)} & 0 & c_6^{(k)} \\ c_1^{(k)} & 0 & 0 & c_8^{(k)} & 0 \\ 0 & c_2^{(k)} & 0 & 0 & 0 \\ c_4^{(k)} & c_7^{(k)} & 0 & 0 & 0 \\ 0 & 0 & 0 & c_5^{(k)} & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & {}^{c}\delta_{13}^{(k)} & 0 & {}^{c}\delta_{15}^{(k)} \\ {}^{c}\delta_{21}^{(k)} & 0 & 0 & {}^{c}\delta_{24}^{(k)} & 0 \\ 0 & {}^{c}\delta_{32}^{(k)} & 0 & 0 & 0 \\ {}^{c}\delta_{41}^{(k)} & {}^{c}\delta_{42}^{(k)} & 0 & 0 & 0 \\ 0 & 0 & 0 & {}^{c}\delta_{54}^{(k)} & 0 \end{pmatrix}$$

and the matrix for the mouse is

$$^{m}\mathbf{\Delta}_k = \begin{pmatrix} 0 & m_3^{(k)} & 0 & m_6^{(k)} & 0 \\ 0 & 0 & m_2^{(k)} & 0 & 0 \\ m_1^{(k)} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & m_5^{(k)} \\ m_4^{(k)} & 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & {}^{m}\delta_{12}^{(k)} & 0 & {}^{m}\delta_{14}^{(k)} & 0 \\ 0 & 0 & {}^{m}\delta_{23}^{(k)} & 0 & 0 \\ {}^{m}\delta_{31}^{(k)} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & {}^{m}\delta_{45}^{(k)} \\ {}^{m}\delta_{51}^{(k)} & 0 & 0 & 0 & 0 \end{pmatrix}$$

To avoid using the complicated transition matrices we can utilize the powers of the numerical adjacency matrices in order to find that after three, four, five, etc. steps from the initial state the terminal state (the same like the initial one) is reachable. In the case of the cat

$$^{c}\mathbf{\Delta} = \begin{pmatrix} 0&0&1&0&1 \\ 1&0&0&1&0 \\ 0&1&0&0&0 \\ 1&1&0&0&0 \\ 0&0&0&1&0 \end{pmatrix} \; ^{c}\mathbf{\Delta}^2 = \begin{pmatrix} 0&1&0&1&0 \\ 1&1&1&0&1 \\ 1&0&0&1&0 \\ 1&0&1&1&1 \\ 1&1&0&0&0 \end{pmatrix} \; ^{c}\mathbf{\Delta}^3 = \begin{pmatrix} 2&1&0&1&0 \\ 1&1&1&2&1 \\ 1&1&\mathbf{1}&0&1 \\ 1&2&1&1&1 \\ 1&0&1&1&1 \end{pmatrix}$$

$$^{c}\mathbf{\Delta}^4 = \begin{pmatrix} 2&1&2&1&2 \\ 3&3&1&2&1 \\ 1&1&\mathbf{1}&2&1 \\ 3&2&1&3&1 \\ 1&2&1&1&1 \end{pmatrix} \; ^{c}\mathbf{\Delta}^5 = \begin{pmatrix} 2&3&2&3&2 \\ 5&3&3&4&3 \\ 3&3&\mathbf{1}&2&1 \\ 5&4&3&3&3 \\ 3&2&1&3&1 \end{pmatrix} \; ^{c}\mathbf{\Delta}^6 = \begin{pmatrix} 6&5&2&5&2 \\ 7&7&5&6&5 \\ 5&3&\mathbf{3}&4&3 \\ 7&6&5&7&5 \\ 5&4&3&3&3 \end{pmatrix}$$

In the case of the mouse

$$
{}^m\mathbf{\Delta}=\begin{pmatrix}0&1&0&1&0\\0&0&1&0&0\\1&0&0&0&0\\0&0&0&0&1\\1&0&0&0&0\end{pmatrix}
{}^m\mathbf{\Delta}^2=\begin{pmatrix}0&0&1&0&1\\1&0&0&0&0\\0&1&0&1&0\\1&0&0&0&0\\0&1&0&1&0\end{pmatrix}
{}^m\mathbf{\Delta}^3=\begin{pmatrix}2&0&0&0&0\\0&1&0&1&0\\0&0&1&0&1\\0&1&0&1&0\\0&0&1&0&\mathbf{1}\end{pmatrix}
$$

$$
{}^m\mathbf{\Delta}^4=\begin{pmatrix}0&2&0&2&0\\0&0&1&0&1\\2&0&0&0&0\\0&0&1&0&1\\2&0&0&0&0\end{pmatrix}
{}^m\mathbf{\Delta}^5=\begin{pmatrix}0&0&2&0&2\\2&0&0&0&0\\0&2&0&2&0\\2&0&0&0&0\\0&2&0&2&0\end{pmatrix}
{}^m\mathbf{\Delta}^6=\begin{pmatrix}4&0&0&0&0\\0&2&0&2&0\\0&0&2&0&2\\0&2&0&2&0\\0&0&2&0&\mathbf{2}\end{pmatrix}
$$

The transition matrices (i.e. the corresponding powers of the matrices $\mathbf{\Delta}$) yields information (see the bold elements ${}^c\delta_{3,3}$ of the corresponding powers of the matrix ${}^c\mathbf{\Delta}$) that the cat has single solutions with the length 3, 4, and 5 steps and three 6-step solutions. The mouse has (see the bold elements ${}^m\delta_{5,5}$ of the corresponding powers of the matrix ${}^m\mathbf{\Delta}$) the single solution of the length 3 and two 6-step solutions. This matrix is not primitive. It is a special matrix, where ${}^m\mathbf{\Delta}^{k+3} = 2.{}^m\mathbf{\Delta}^k$. Let us illustrate the proposed approach to the control synthesis. In Tab. 3 we have the straight-lined sequence of the aggregated states of the cat and mouse behaviour. Very analogically (using the powers of the transpose of the adjacency matrix) the backtracking sequence of the aggregated states of the cat and mouse are found (Tab. 4). Consider the situation when the the left parts of the Tab. 3 and the Tab. 4 are overlapped. Let us compare their corresponding columns and create their coincidence. In such a way the resulting Tab. 5 can be obtained.

Table 3: The straight-lined system development of the cat (on the left) and that of the mouse (on the right)

${}^c\mathrm{x}_0$	$\{{}^c\mathrm{x}_1\}$	$\{{}^c\mathrm{x}_2\}$	$\{{}^c\mathrm{x}_3\}$	$\{{}^c\mathrm{x}_4\}$	$\{{}^c\mathrm{x}_5\}$	$\{{}^c\mathrm{x}_6\}$	${}^m\mathrm{x}_0$	$\{{}^m\mathrm{x}_1\}$	$\{{}^m\mathrm{x}_2\}$	$\{{}^m\mathrm{x}_3\}$	$\{{}^m\mathrm{x}_4\}$	$\{{}^m\mathrm{x}_5\}$	$\{{}^m\mathrm{x}_6\}$
0	1	0	0	2	2	2	0	0	1	0	0	2	0
0	0	1	1	1	3	5	0	0	0	0	1	0	0
1	0	0	1	1	1	3	0	0	0	1	0	0	2
0	0	1	1	1	3	5	0	1	0	0	1	0	0
0	0	0	1	1	1	3	1	0	0	1	0	0	2

Table 4: The backtracking system development of the cat (on the left) and that of the mouse (on the right)

$\{{}^c\mathrm{x}_0\}$	$\{{}^c\mathrm{x}_1\}$	$\{{}^c\mathrm{x}_2\}$	$\{{}^c\mathrm{x}_3\}$	$\{{}^c\mathrm{x}_4\}$	$\{{}^c\mathrm{x}_5\}$	${}^c\mathrm{x}_6$	$\{{}^m\mathrm{x}_0\}$	$\{{}^m\mathrm{x}_1\}$	$\{{}^m\mathrm{x}_2\}$	$\{{}^m\mathrm{x}_3\}$	$\{{}^m\mathrm{x}_4\}$	$\{{}^m\mathrm{x}_5\}$	${}^m\mathrm{x}_6$
5	3	1	1	1	0	0	0	0	2	0	0	1	0
3	3	1	1	0	1	0	0	2	0	0	1	0	0
3	1	1	1	0	0	1	2	0	0	1	0	0	0
4	2	2	0	1	0	0	0	2	0	0	1	0	0
3	1	1	1	0	0	0	2	0	0	1	0	0	1

Table 5: The resulting 6-step trajectory of the cat behaviour (on the left) and that of the mouse (on the right)

${}^c\mathrm{x}_0$	$\{{}^c\mathrm{x}_1\}$	$\{{}^c\mathrm{x}_2\}$	$\{{}^c\mathrm{x}_3\}$	$\{{}^c\mathrm{x}_4\}$	$\{{}^c\mathrm{x}_5\}$	${}^c\mathrm{x}_6$	${}^m\mathrm{x}_0$	$\{{}^m\mathrm{x}_1\}$	$\{{}^m\mathrm{x}_2\}$	$\{{}^m\mathrm{x}_3\}$	$\{{}^m\mathrm{x}_4\}$	$\{{}^m\mathrm{x}_5\}$	${}^m\mathrm{x}_6$
0	1	0	0	1	0	0	0	0	1	0	0	1	0
0	0	1	1	0	1	0	0	0	0	0	1	0	0
1	0	0	1	0	0	1	0	0	0	1	0	0	0
0	0	1	0	1	0	0	0	1	0	0	1	0	0
0	0	0	1	0	0	0	1	0	0	1	0	0	1

The straight-lined and backtracking development of the cat is given on Fig. 7 and those of the mouse on the Fig. 8.

Using the 6-step tables the independent solutions for the cat and the mouse expressed on Fig. 9 are obtained. The overlaping of both the 6-step independent solution of the cat and that of the mouse the final

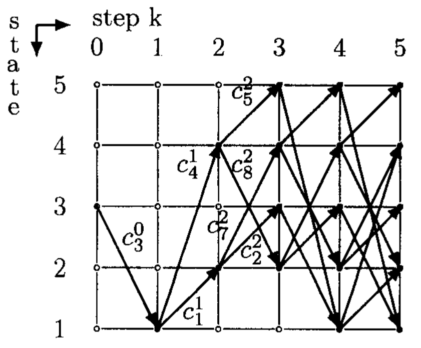
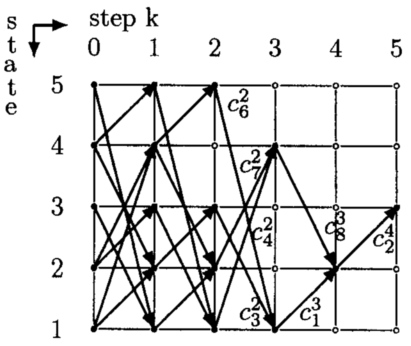

Figure 7: The graphical expression of the straight-lined (on the left) and the backtracking (on the right) development of the cat behaviour

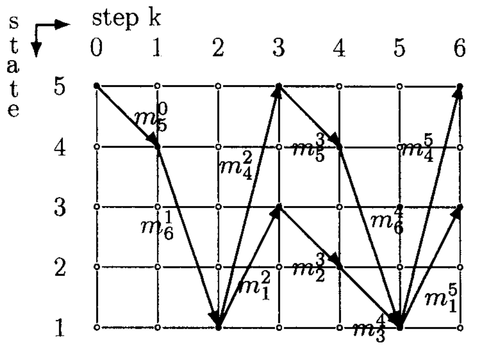
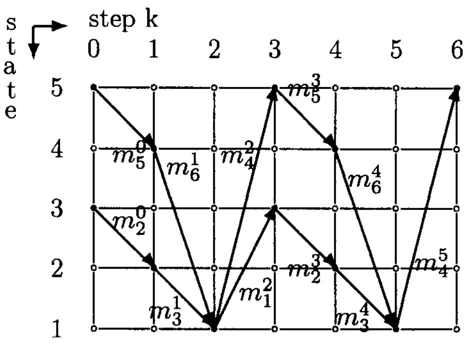

Figure 8: The graphical expression of the straight-lined (on the left) and the backtracking (on the right) development of the mouse behaviour

solution of the problem inquestion can be found. It is necessary to apply the rules expressing the above introduced three control task specifications. Thus, c_5^2 is eliminated and m_4^2 is prefered because the room 5 is the place of the mouse comeback. Analogically, m_1^2 is eliminated and c_2^2 is prefered because the room 3 is the place of the cat comeback. The final solution of the cat and mouse control synthesis is given on Fig. 10.

Hence, the sequence of the final control vectors for the cat is the following

$$
{}^c\mathbf{u}_0 = (0, 0, 1, 0, 0, 0, 0, 0)^T \; ; \; {}^c\mathbf{u}_1 = (1, 0, 0, 0, 0, 0, 0, 0)^T \; ; \; {}^c\mathbf{u}_2 = (0, 1, 0, 0, 0, 0, 0, 0)^T
$$
$$
{}^c\mathbf{u}_3 = (0, 0, 1, 0, 0, 0, 0, 0)^T \; ; \; {}^c\mathbf{u}_4 = (1, 0, 0, 0, 0, 0, 0, 0)^T \; ; \; {}^c\mathbf{u}_5 = (0, 1, 0, 0, 0, 0, 0, 0)^T
$$

In order to fulfil the demand of the free movement also the following sequence is possible (however, it contains movement through the uncontrollable door)

$$
{}^c\mathbf{u}_0 = (0, 0, 1, 0, 0, 0, 0, 0)^T \; ; \; {}^c\mathbf{u}_1 = (0, 0, 0, 1, 0, 0, 0, 0)^T \; ; \; {}^c\mathbf{u}_2 = (0, 0, 0, 0, 0, 0, 0, 1)^T
$$

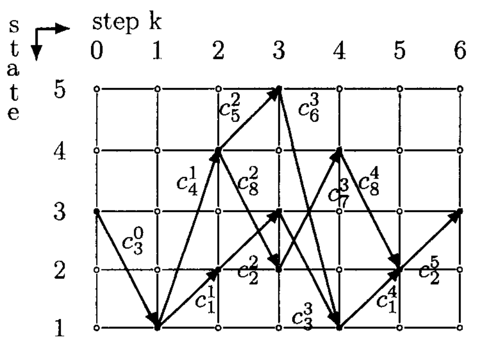
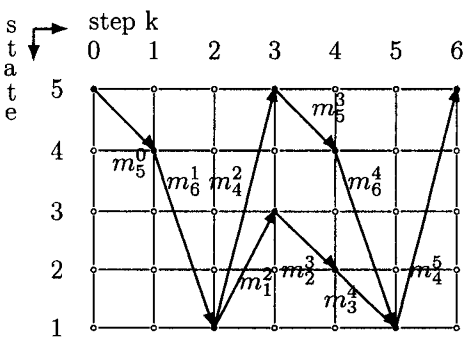

Figure 9: The independent 6-step solution of the cat control synthesis problem and the that of mouse

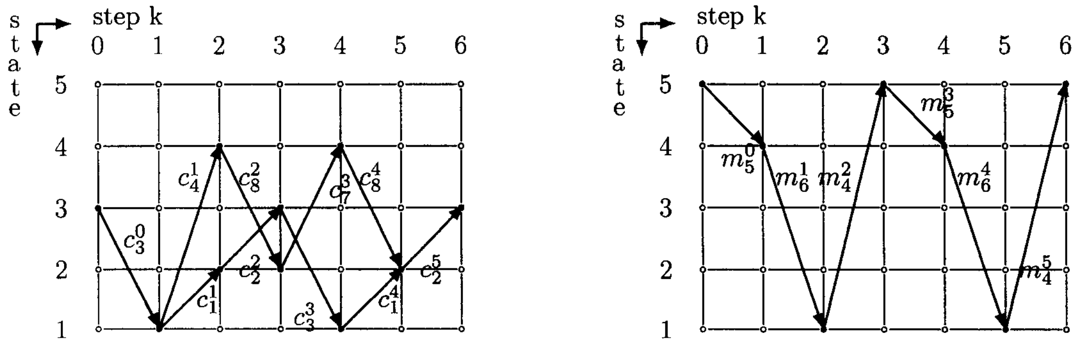
state
step k
0 1 2 3 4 5 6
5 4 3 2 1
c_3^0 c_1^1 c_4^1 c_2^2 c_8^2 c_3^3 c_7^3 c_1^4 c_8^4 c_2^5
state
step k
0 1 2 3 4 5 6
5 4 3 2 1
m_5^0 m_6^1 m_4^2 m_5^3 m_6^4 m_4^5

Figure 10: The final (mutually dependent) solution of the cat control synthesis problem and that of mouse

$${}^c\mathbf{u}_3 = (0,0,1,0,0,0,1,0)^T \;;\; {}^c\mathbf{u}_4 = (1,0,0,0,0,0,0,1)^T \;;\; {}^c\mathbf{u}_5 = (0,1,0,0,0,0,0,0)^T$$

The sequence of the final control vectors for the mouse is the following

$$\begin{aligned}
{}^m\mathbf{u}_0 &= (0,0,0,0,1,0)^T \;;\; {}^m\mathbf{u}_1 = (0,0,0,0,0,1)^T \;;\; {}^m\mathbf{u}_2 = (0,0,0,1,0,0)^T \\
{}^m\mathbf{u}_3 &= (0,0,0,0,1,0)^T \;;\; {}^m\mathbf{u}_4 = (0,0,0,0,0,1)^T \;;\; {}^m\mathbf{u}_5 = (0,0,0,1,0,0)^T
\end{aligned}$$

To explain better the principle of the proposed approach to the control synthesis, it is better to use the 5-step coincidence of the straight-lined and backtracking developments of the system. Namely, in such a case Table 6 yields single trajectory. This table can be obtained by means of shifting the left part of the Table 4

Table 6: The resulting 5-step trajectory of the cat behaviour

${}^c\mathbf{x}_0$	$\{{}^c\mathbf{x}_1\}$	$\{{}^c\mathbf{x}_2\}$	$\{{}^c\mathbf{x}_3\}$	$\{{}^c\mathbf{x}_4\}$	${}^c\mathbf{x}_5$
0	1	0	0	0	0
0	0	1	0	1	0
1	0	0	0	0	1
0	0	0	1	0	0
0	0	0	0	0	0

to the left for one column (in comparison with the left part of the Table 3) and coinciding the overlapped columns. After shifting to the left once more and coinciding the corresponding columns the 4-step solution can be found. After further shifting to the left we have 3-step solution of the control synthesis problem. Fig. 11 expresses graphically the 5-, 4- and 5-step solutions of the cat, while the Fig. 12 expresses graphically the 3-step solution of the mouse obtained analogically. The 5- and 4-step solution of the mouse do not exist.

6.1 Summary

By means of comparing 6-step both the straight-lined development of the system and the backtracking one we are able to find three 6-step independent solutions of the cat control synthesis problem. In addition to these in any step of the shifting we are able to find single independent solution. In such a way we have three possible single independent solutions (5-, 4-, and 3-step ones). Analogically, two 6-step independent solutions and the single 3-step independent solution of the mouse control synthesis problem can be found.

In order to denote the elementary sections of the trajectories by the corresponding transitions that have to be fired, the corresponding adjacency matrix (${}^c\mathbf{\Delta}_k$ or ${}^m\mathbf{\Delta}_k$) can be utilized. It stores information about the placement of the transitions.

To satisfy the prescribed control task specifications some of the independent solutions had to be eliminated and consequently, the final solutions (dependent on the control task specifications) were found.

The main aim of involving the details of solving the example was to demonstrate the applicability of the approach proposed in this paper. Although the control task specifications were given only verbally, the

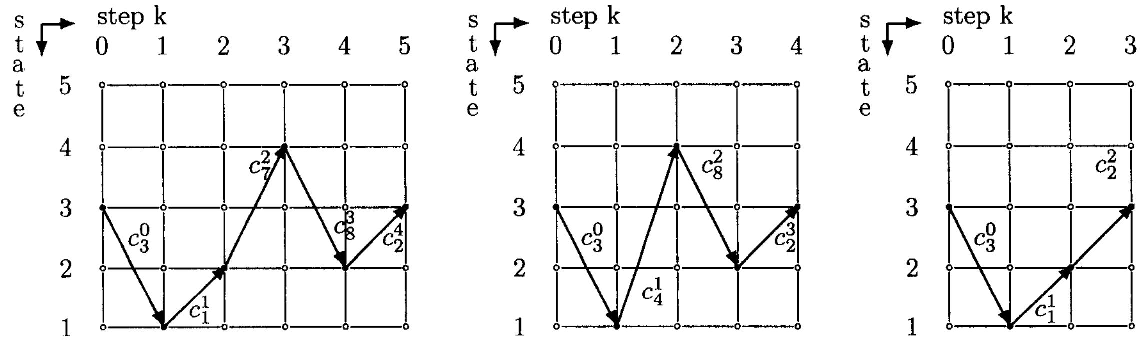

Figure 11: The 5-step solution (on the left), the 4-step solution (in the center) and the 3-step one (on the right) of the cat control synthesis problem

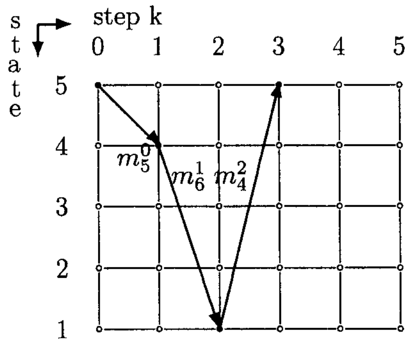

Figure 12: The 3-step solution of the mouse control synthesis problem

approach makes possible to solve the control synthesis problem in analytical terms. However, knowledge about the control task specifications has to be represent in appropriate form. It is very important at the choice of the most suitable solution.

7 The choice of the most suitable solution

It can be seen in the above example that a suitable knowledge base can be used in order to choose the most suitable possibility when there are several solutions of the control synthesis problem. In order to create such a knowledge base special kinds of PN (logical or/and fuzzy PN) can be utilized. Knowledge in general can be understood to be a complex causal discrete system consisting of some pieces of knowledge (e.g. some statements S_i, $i = \overline{1, n}$) that are mutually interconnected by means of causal relations (e.g. into the rules R_j, $j = \overline{1, m}$). In order to represent rule-based knowledge - consisting of a system of the IF-THEN rules e.g. like the following one

$$R_j : IF\ (S_a\, and\, S_b\, and\, S_c)\ THEN\ (S_d\, and\, S_e) \tag{19}$$

represented by the PN fragment given on Fig. 13, where S_a, S_b, S_c are the input statements (causes) and S_d, S_e are the output ones (consequences) - several different approaches can be used. The PN-based approach [6] can be utilized on this way. The truth propagation in case of logical PN is given on Fig. 13. In case of fuzzy rules the fuzzy PN are utilized. On Fig. 14 the situation before and after firing a fuzzy rule is illustrated. More details about the mathematical tool concerning the knowledge representation as well as about the knowledge base (KB) construction can be found in the author's works - see e.g. [6, 8, 9, 17]. In [9] even the actual KB for solving the same example, however described by means of the PN-based approach, was introduced too. The same mathematical tool can be utilized here, but the actual elementary statements and rules of the KB can be a little different like those in [9].

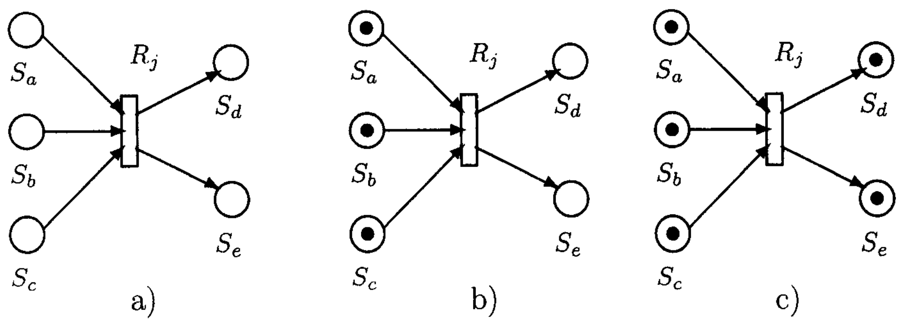

Figure 13: The rule R_j with the input and output statements - a). The example of the rule R_j in bivalued logic and its state: b) before its evaluation; c) after its evaluation

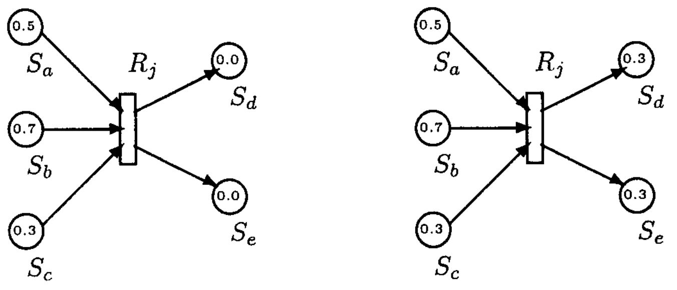

Figure 14: An example of a fuzzylogical rule R_j before firing (on the left) and after firing (on the right).

8 Conclusions

In this paper the new approach to MS modelling and control synthesis was presented. MS were understood here to be a kind of DEDS. The PN-based approach to modelling and control synthesis was replaced here by means of the corresponding OG-based one with the OG nodes being the original PN positions and the OG edges involving the PN transitions. The adjacency matrix of such a graph was utilized here in the DEDS control synthesis procedure. The matrix and its transpose helps to generate the state reachability tree in both the straight-lined system development (from an initial state to a prescribed terminal one) and that of the backtracking one (from the terminal state to the initial one). By means of combinig both of the kinds of the model development automatic solving the control synthesis problems was performed. The coincidence of both the straight-lined state reachability tree and that of backtracking one yields the possible trajectories of the system development. In such a way all solutions how to reach the prescribed terminal state from the given initial one were automatically found. To choose the most suitable solution with respect to the prescribed control task specifications, rule-based knowledge about them has to be used. On this way the logical or/and fuzzy PN can be used, because the knowledge can be fuzzy too. More details about the KB construction are given in [9].

References

[1] F. Čapkovič, A Petri nets-based approach to the maze problem solving, In: *Discrete Event Systems: Modelling and Control* (S. Balemi, P. Kozák and R. Smedinga, Eds.), Series: *Progress in Systems and Control Theory*, Vol. 13, Birkhäuser Verlag, Basel - Boston - Berlin. (1993) 173–179.

[2] F. Čapkovič, Knowledge-based control of DEDS, In: *Proc. of the 13th IFAC World Congress 1996, San Francisco, USA, June 30-July 5, 1996* (J.J. Gertler, J.B. Cruz and M. Peshkin, Eds.). Vol. J. also Compact Disc, Elsevier Science Ltd., Pergamon, 1996, paper J-3c-02.6. (1996) 347–352.

[3] F. Čapkovič, Petri nets and oriented graphs in fuzzy knowledge representation for DEDS control purposes, *BUSEFAL* **69** (1997) 21–30.

[4] F. Čapkovič, Fuzzy knowledge in DEDS control synthesis, In: *Proc. of the 7th International Fuzzy Systems Association World Congress - IFSA'97, Prague, Czech Republic, June 25-29, 1997* (M. Mareš, R. Mesiar, V. Novák, J. Ramík, A. Stupňanová, Eds.), Academia, Prague, Czech Republic. (1997) 550–554.

[5] F. Čapkovič, An approach to knowledge-representation at control of DEDS, In: *Advances in Intelligent Systems* (F.C. Morabito, Ed.). IOS Press, Ohmsha, Amsterdam-Berlin-Oxford-Tokyo-Washington DC, ISBN 90 5199 355 2 (IOS Press), 4 274 90180 7 C3000 (Ohmsha). (1997) 458–463.

[6] F. Čapkovič, Representation of fuzzy knowledge about control task specifications, In: *IT & KNOWS Information Technologies and Knowledge Systems. Proceedings of XV. IFIP World Computer Congress, August 31-September 4, 1998, Vienna, Austria, and Budapest Hungary* (J. Cuena, Ed.), Riegelnik, Vienna, (also Compact Disc.). (1998) 215–228.

[7] F. Čapkovič, Knowledge-based control synthesis of discrete event dynamic systems, In: *Advances in Manufacturing Systems. Decision, Control and Information Technology* (S.G. Tzafestas, Ed.). Chapter 19. Springer Verlag, London, ISBN 1-85233-126-7. (1998) 195–206.

[8] F. Čapkovič, Knowledge-based control of production systems, In: *Proc. of the 6th European Congress on Intelligent Techniques & Computing - EUFIT'98, September 7-10, 1998, Aachen, Germany* (H.J. Zimmermann, Ed.), ELITE Foundation, Aachen, **Vol. 3** (1998) 1565-1569.

[9] F. Čapkovič, Fuzzy knowledge in control of manufacturing systems, *BUSEFAL* **75** (1998) 4–17.

[10] F. Čapkovič, Automated solving of the DEDS control problems, In: *Multiple Approaches to Intelligent Systems* (I. Imam, Y. Kodratoff, A. El-Dessouki and M. Ali, Eds.). Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Sciences). Springer, Berlin-Heidelberg-New York-Barcelona-Hong Kong-London-Milan-Paris-Singapore-Tokyo, **Vol. 1611** (1999) 735–746.

[11] R. Diestel, *Graph Theory*, Springer-Verlag, New York (1997).

[12] M. Fiedler, *Special Matrices and their Using in Numerical Mathematics (in Czech)*, SNTL Publishing House, Prague, Czechoslovakia (1981).

[13] S. Kirkland, D.D. Olesky, and P. van den Driessche, Digraphs with large exponent, *The Electronics Journal of Linear Algebra* **7** (2000) 30–40.

[14] J. Plesník, *Graph Algorithms (in Slovak)*, VEDA. Bratislava, Czechoslovakia (1983).

[15] F.P. Preparata, R.T. Yeh, *Introduction to Discrete Structures*, Addison-Wesley Publ. Comp. Reading, USA (1974).

[16] J. Sedláček, *Introduction to Graph Theory (in Czech)*, Academia, Prague, Czechoslovakia (1977).

[17] S.G. Tzafestas, F. Čapkovič, Petri net-based approach to synthesis of intelligent control for DEDS, In: *Computer Assisted Management and Control of Manufacturing Systems* (S.G. Tzafestas, Ed.). Chapter 12. Springer Verlag, Berlin-Heidelberg-New York, ISBN 3-540-76110-1. (1996) 325–351.

[18] H. Wielandt, Indecomposable non-negative matrices (in German), *Math. Zeitschrift* **52** (1950) 642–648.

[19] W.M. Wonham, P.J. Ramadge, On the supremal controllable sublanguage of a given language, *SIAM J. Control and Optimization* **25** (1987) 637–659.