Computing and Informatics, Vol. 26, 2007, 507–541

MODELLING, ANALYSING AND CONTROL OF INTERACTIONS AMONG AGENTS IN MAS

František Čapkovič

Institute of Informatics
Slovak Academy of Sciences
Dúbravská cesta 9
845 07 Bratislava, Slovakia
e-mail: {Frantisek.Capkovic, utrrcapk}@savba.sk

Revised manuscript received 20 February 2007

Abstract. An alternative approach to modelling and analysis of interactions among agents in multiagent systems (MAS) and to their control is presented in analytical terms. The attention is focused especially on the negotiation process. However, the possibility of another form of the communication is mentioned too. The reachability graph of the Petri net (PN)-based model of MAS is found as well as the space of feasible states. Trajectories representing the interaction processes among agents in MAS are computed by means of the mutual intersection of both the straight-lined reachability tree (from a given initial state towards the terminal one) and the backtracking reachability tree (from the desired terminal state towards the initial one, however oriented towards the terminal state). Control interferences are obtained on the basis of the most suitable trajectory chosen from the set of feasible ones.

Keywords: Agents, analysis, control, decision, discrete-event systems, modelling, petri nets

1 INTRODUCTION

MAS are used in intelligent control, especially for a cooperative problem-solving [27]. To analyse complicated interactions among agents modelling of them is often used. The negotiation belongs to the most important interactions. It is the process of multilateral bargaining for mutual profit. In other words [2, 25], the negotiation is a decision process where two or more participants make individual decisions and interact with each other in order to reach a compromise. In [13] Petri nets (PN) are used for e-negotiations activities. The following five principle properties of e-negotiation are defined there:

1. interactivity (it involves the agents to participate and communicate with each other);
2. informativity (it generates, transmits and stores information);
3. irregularity (it behaves differently according to the combination of agents, strategies, events, tasks, issues, alternatives, preferences and criteria);
4. integrity (it affords speed, consistency and absence of errors through efficient and effective mechanisms);
5. inexpensivity (it automates or semi-automates negotiation activities to save time and cost).

Because PN can effectively help on this way (i.e. to express the properties to be satisfied, especially the first three ones that are generic) PN were chosen to model MAS too [17, 21]. On the base of previous experience [4, 5, 8] with PN-based modelling and control synthesis of the discrete event dynamic systems (DEDS) and the agent cooperation [6, 7] a new approach to modelling, analysis and control of the negotiation process is proposed here. The negotiation process is understood to be DEDS. It seems to be natural, because the process is discrete in nature and simultaneously it is causal. The approach consists of:

1. creating the PN-based mathematical model of the negotiation process;
2. generating the space of feasible states which are reachable from the given initial state;
3. utilizing the reachability graph in order to find the feasible state trajectories to a prescribed feasible terminal state.

After a thorough analysing the set of possibilities, the most suitable strategy (the control trajectory) can be chosen.

In order to use DEDS-based approach it is necessary to mention DEDS operation and causality first. Likewise the PN-based model will be concisely mentioned because it yields both the user friendly analytical description of DEDS operation (by means of the linear discrete model consisting of the system of linear difference equations) and the basis for computing the reachability tree and/or the reachability graph making possible simple testing the DEDS properties as well as assuring the system causality.

As to the model creation the approach is sufficiently general to be utilized not only for modelling the negotiation process but also for modelling the wider spectrum of both the agent behaviour and the forms of communication among the agents.

2 THE DEDS OPERATION AND CAUSALITY

DEDS behaviour is driven by discrete events. Such a system persists in a given state as long as it is forced to change it owing to the occurrence of a discrete event. As to the structure DEDS can be very complicated. Namely, the typical representatives of DEDS are flexible manufacturing systems (FMS), communication systems of different kinds, transport systems, etc. Taking into account e.g. FMS as an example of DEDS, they consist of many devices like robots, numerically controlled machine tools, conveyers, automatically guided vehicles, pallets, buffers, etc., in order to produce the final product(s) from raw material(s) and/or semiproduct(s). The mutual cooperation among the devices is driven by discrete events (starting or ending some technological operations and/or movement of mobile devices, switching the devices on and/or off, etc.). In PN-based modelling of DEDS the level of abstraction is very important. From the global point of view the FMS devices can be understood to be atomic elements of DEDS. However, at a deeper investigation they can also be found to be DEDS, namely the subsystems of the global FMS with partial activities and/or operations being the atomic elements. While in the global understanding a robot is a device being only switched on (to be activated) and/or off (to be deactivated), going into details we can find that it performs several activities – it can be available (i.e. free and waiting for any activation), it can perform different operations like picking parts up, moving its wrist with one of the parts from one point of its working space to another one, putting a part in a machine tool, on a pallet or into a buffer, taking a part off a machine tool or a buffer, etc. Analogically, the machine tools can perform several different technological operations (e.g. drilling, milling, etc.). From the technological point of view FMSs realize a technological processes while their subsystems realize corresponding technological subprocesses. The state of FMS at a time instant consists of the states all of the atomic activities in the system. Therefore, it is a vector. Its dimensionality is equal to the global number of activities in the system.

From the system theory point of view the operation of a simple example of DEDS – e.g. one of the FMS devices – can be imagined in such a way like it is illustrated in Figure 1. The course of a DEDS state – i.e. the system dynamics development – represented there shows us how the state changes its discrete values (levels) $x_1, x_2, \dots, x_6$ owing to the occurrence of discrete events $u_1, u_2, \dots, u_5$. Here, x_i represents the state of the atomic activity a_i, $i = 1, \dots, 6$. Namely, if a_i is performed x_i is active, otherwise x_i is passive. In this very simple example only one of the six feasible atomic activities is performed in any step of the DEDS dynamic development, e.g.:

1. x_3 – the robot R moves from its waiting position to a conveyer;
2. x_1 – R picks up a part from the conveyer;
3. x_2 – R moves (with the part in its wrist) from the conveyer to a machine tool;
4. x_5 – R puts the part into the machine tool;

5. x_4 – R moves to its waiting position;

6. x_6 – the machine tool starts a tooling operation.

The activities link to each other – they pass sequentially. Namely, the system persists in a discrete state (performs an operation) until it is forced to change it owing to the occurrence of a discrete event (ending the operation and starting another one). After this change the system persists in the new discrete state (performs the operation) until another discrete event (ending/starting of an operation) occurs and causes another change, etc. It is clear from Figure 1 that there exist 6 different discrete levels of the state, namely $x_1, x_2, \dots, x_6$. At any time the system (in our simple example) can persist only in one of them, of course, because it has a character of the sequential process. Consequently, the states of the system can be expressed by the state vectors with the structure $\mathbf{x}_k = (x_1^k, x_2^k, \dots, x_6^k)^T$, $k = 0, 1, \dots, K$, where for each k only one of the components x_i^k, $i = 1, 2, \dots, n$ (in our case $n = 6$) of the vector $\mathbf{x}_k$ is different from zero. Thus, for $K = 5$ we have the following states $\mathbf{x}_0 = (0, 0, x_3^0, 0, 0, 0)^T$, $\mathbf{x}_1 = (x_1^1, 0, 0, 0, 0, 0)^T$, $\mathbf{x}_2 = (0, x_2^2, 0, 0, 0, 0)^T$, $\mathbf{x}_3 = (0, 0, 0, 0, x_5^3, 0)^T$, $\mathbf{x}_4 = (0, 0, 0, x_4^4, 0, 0)^T$, $\mathbf{x}_5 = (0, 0, 0, 0, 0, x_6^5)^T$. Name k to be the step of the system dynamics development.

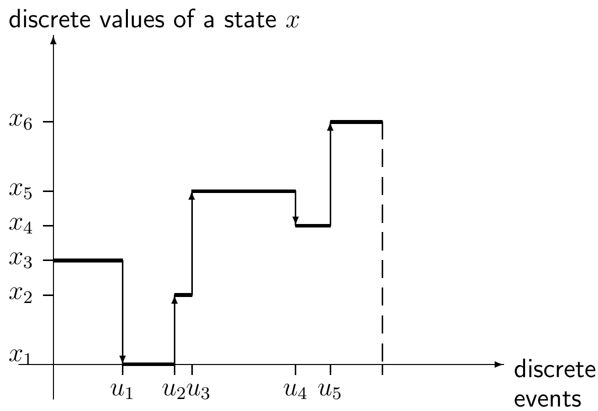


Fig. 1. The development of the state in the simple example of DEDS

It is necessary to say that in large and complicated FMS a number of devices can work simultaneously – for example, two production lines of FMS. When both lines contain a machine tool and the machine tools are served by the same robot R, the robot cannot serve both machines simultaneously. It is clear that R is able to serve the machines either alternatively or in virtue of a prescribed program. In such a case the course of the DEDS state will be more complicated than that shown in Figure 1. Therefore, it is useful to establish another graphical interpretation of the system dynamic development.

2.1 Feasible States and the State Space

To avoid any misunderstanding, it is necessary to distinguish both the feasible state vectors of the system regardless of the step k in which they occur (i.e. regardless of the causality relations) and the state vectors occurring consecutively (successively) during the system dynamics development in the steps $k = 0, 1, \ldots, K$ of the system dynamics development, i.e causally. While the former vectors (they are mutually different) represent the system state space, the latter ones express the dynamics of the system within the framework of the given state space. Let us denote in our example of FMS the feasible states as $\mathbb{X}_1, \ldots, \mathbb{X}_6$, where $\mathbb{X}_1$ corresponds to $\mathbf{x}_0$ (we will use the symbol $\triangleq$, i.e. $\mathbb{X}_1 \triangleq \mathbf{x}_0$), $\mathbb{X}_2 \triangleq \mathbf{x}_1$, $\mathbb{X}_3 \triangleq \mathbf{x}_2$, $\mathbb{X}_4 \triangleq \mathbf{x}_3$, $\mathbb{X}_5 \triangleq \mathbf{x}_4$, $\mathbb{X}_6 \triangleq \mathbf{x}_5$. The set $X_{reach} = \{\mathbb{X}_1, \mathbb{X}_2, \ldots, \mathbb{X}_6\}$ of the feasible states is created by the initial state vector $\mathbf{x}_0$ and all of the mutually different states reachable from $\mathbf{x}_0$ regardless of the number of steps which are necessary for such a reachability. The vectors of the set X_{reach} can be understood to be the columns of the matrix $\mathbf{X}_{reach}$ as follows.

$$\mathbf{X}_{reach} = (\mathbb{X}_1, \mathbb{X}_2, \ldots, \mathbb{X}_6) = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{1}$$

It is the matrix of feasible states. As already said, by the term *feasible* states the initial state together with all of the states reachable from it are meant. The dimensionality of the matrix $\mathbf{X}_{reach}$ is $(n \times N)$ with n, N being, respectively, the number of the atomic activities being in the discrete states $x_i^k, i = 1, \ldots, n$ (the components of the vectors $\mathbf{x}_k,\ k = 0, 1, \ldots, K$) of the system, and the number of all the feasible state vectors $\mathbb{X}_i,\ i = 1, 2, \ldots, N$. For the given initial state the matrix $\mathbf{X}_{reach}$ represents the system state space as a whole. However, when the initial state will be represented e.g. by the vector $\mathbf{x}_0 = (0, x_2^0, 0, 0, 0, 0)^T$, then $\mathbf{x}_1 = (0, 0, 0, 0, x_5^1, 0)^T$, $\mathbf{x}_2 = (0, 0, 0, x_4^2, 0, 0)^T$, $\mathbf{x}_3 = (0, 0, 0, 0, 0, x_6^3)^T$. Thus, $\mathbb{X}'_1 \triangleq \mathbf{x}_0$, $\mathbb{X}'_2 \triangleq \mathbf{x}_1$, $\mathbb{X}'_3 \triangleq \mathbf{x}_2$ and $\mathbb{X}'_4 \triangleq \mathbf{x}_3$. Consequently, $N = 4$ and $\mathbf{X}_{reach} = (\mathbb{X}'_1, \mathbb{X}'_2, \mathbb{X}'_3, \mathbb{X}'_4)$.

However, in general, some of the feasible states can occur repeatedly (inconsecutively) in a causal sequence of the states $\{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_K\}$ in which the system finds itself during the dynamics development.

Although no time is explicitly designated on the horizontal axis in Figure 1 its implicit presence is indisputable. Namely, the discrete events occur in concrete time instants but the time intervals between the instants of occurrence of two adjacent events have different length. However, taking no account of a time we can introduce a new simplified conception of the course of DEDS state as a formal representation of the state trajectory. In such a case we will designate the steps of the DEDS dynamics development on the horizontal axis, namely with the same geometric distance between pairs of adjacent steps. At the same time we will designate the discrete values of the DEDS state on the vertical axis, namely likewise with the same geometric distance between pairs of adjacent discrete values. Consequently, the course of the DEDS state will be represented by a broken line consisting of the elementary linear segments (the abscissae) connecting the points with the pairs of coordinates (k_i, x_{j_1}) and (k_{i+1}, x_{j_2}), where k_i is the i^{th} step of the DEDS dynamics development and x_{j_1}, x_{j_2} (it is clear that $x_{j_1} > x_{j_2}$ or $x_{j_1} < x_{j_2}$) are two adjacent discrete values of the state. At the same time, the discrete event causing the change of the state from the discrete value to another one represents the parameter of the transition of the state between these values. At such a conception the course of the state represented in Figure 1 will be transformed into the course represented in Figure 2 a).

To emphasize the step of the system dynamics development, the left upper index k is added to the symbol of i^{th} discrete event u_i denoting the step in which the event occurs. Thus, ${}^k u_i$ means that u_i occurs in the step k. Because the graphical expression of states is imbedded into the lattice, it is well-arranged, lucid, and easy to understand.

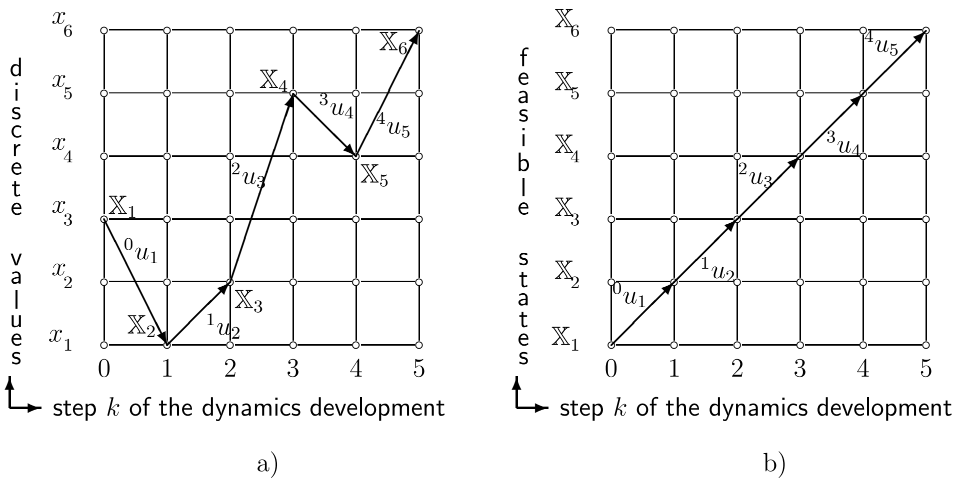


Fig. 2. The simplified representation of: a) the course of the DEDS state; b) the state trajectory in the state space

As evident from Figure 2 a), such a representation is unambiguous. Namely, the transition of the DEDS state from the discrete value in a step k to the immediately following one in the next step is realized just by means of the corresponding discrete event. Although the broken line is only fictive, it can be named as the state trajectory, because it represents (by any linear part – i.e. by any abscissa) the transition from the existing state to another one being the consequence of the occurrence of the corresponding discrete event.

When the feasible states are designated on the vertical axis, another simplified representation of the DEDS state course can be introduced. It is displayed in

Figure 2 b). As can be seen below, such a representation can be utilized also for displaying paths of the PN reachability tree and/or the reachability graph. Namely, their nodes can be designated on the vertical axis and their lengths can be designated by the number of steps on the horizontal axis.

The system can pass from a given discrete state to another state by means of the occurrence of one discrete event only. However, in general (in more complicated DEDS), several discrete events can sometimes be enabled in a given discrete state. Because only one of them can be fired, there is a conflict among them. In case of the forced events a decision has to be made which of them will be fired. Namely, each of them can transfer the system to another (however, again solely one) potential next state. Moreover, these states are mutually different. Consequently, the event transferring the system into the most suitable state is chosen, of course. The choice can be realized on the basis of a criterion or in virtue of a prescription. In case of FMS the choice is determined by the technological process. Namely, any discrete event cannot transfer the system into two or more different states simultaneously. This fact is very important for the DEDS causality and it is utilized during the control synthesis.

Chaining several successive discrete events yields the trajectory representing the transition of the system from the given initial state to a prescribed terminal one. Such a transition is causal. The DEDS causality can be (in an analogy with the causality of continuous systems described in [1]) illustrated by Figure 3 where the state in the step $k = j$ can be reached from the state in the step $k = 0$ (the initial state) after j steps. However, the same state can be reached also from the state in the step $k = i$, $0 < i < j$, after $(j - i)$ steps.

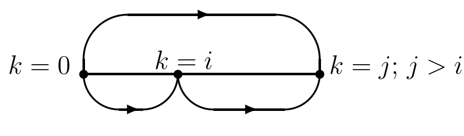


Fig. 3. The principle of causality in DEDS

Let us understand the trajectory in Figure 2 b) to be the directed graph. Its adjacency matrix is as follows:

$$\mathbf{A} = \begin{pmatrix} 0 & {}^0u_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & {}^1u_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & {}^2u_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & {}^3u_4 & 0 \\ 0 & 0 & 0 & 0 & 0 & {}^4u_5 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{2}$$

The dimensionality of $\mathbf{A}$ depends on the number of the graph nodes. Because the nodes are represented by the feasible state vectors the matrix dimensionality is

$N \times N$. In our case $N = 6$. However, the nonzero elements of $\mathbf{A}$ representing the discrete events depend on the step k of the DEDS dynamics development. Namely, the elements $a_{i,j},\ i = 1, 2, \ldots, 6,\ j = 1, 2, \ldots, 6$, of the adjacency matrix $\mathbf{A}$ represent the discrete events transfering the system from $\mathbb{X}_i$ to the adjacent state $\mathbb{X}_j$. Thus $a_{i,j} = {}^k u_{(\mathbb{X}_i \to \mathbb{X}_j)},\ {}^k u_{(\mathbb{X}_i \to \mathbb{X}_j)} \in \{0, 1\},\ i = 1, \ldots, 6,\ j = 1, \ldots, 6$ where k determines the step of the DEDS dynamics development when the event is enabled. Consequently, the element $a_{i,j}$ depends on k. In other words the element is k-variant (is a function of k). Therefore, the matrix $\mathbf{A}$ can be named as the k-variant adjacency matrix and/or the functional adjacency matrix and it will be denoted as $\mathbf{A}_k$. In order to avoid the complicated lower index $\mathbb{X}_i \to \mathbb{X}_j$ of the events we will use only the ordinal number of the event like it was done in Figures 2 a) and 2 b) as well as in (2). Thus, for $j = 0, 1, \ldots$

$$ {}^j u_r = \begin{cases} 1 & \text{if } j = k \\ 0 & \text{otherwise} \end{cases} \ ;\ r = 1, 2, \ldots m \tag{3} $$

where m is the total number of discrete events. In our case $m = 5$. Consequently,

$$ \mathbf{A}_0 = \begin{pmatrix} 0 & \boxed{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \mathbf{A}_1 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \boxed{1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \ldots\ \mathbf{A}_4 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \boxed{1} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} $$

2.2 Vicarious State Vectors

In order to work with the functional adjacency matrix $\mathbf{A}_k$ in a mathematical model of the DEDS dynamics, let us define the N-dimensional vicarious vectors $\mathbf{X}_k = ({}^k X_1, {}^k X_2, \ldots, {}^k X_N)^T,\ k = 0, 1, \ldots, K$ which will take the place of the real state vectors $\mathbf{x}_k,\ k = 0, 1, 2, \ldots, K$ representing the system dynamics development. In any step k the vector $\mathbf{X}_k$ has only one nonzero component, namely

$$ {}^k X_i = \begin{cases} 1 & \text{if } i = k + 1 \\ 0 & \text{otherwise.} \end{cases} \ ;\ i = 1, 2, \ldots, N \tag{4} $$

Thus, in our case $\mathbf{X}_0$ representing $\mathbf{x}_0$ has the form $\mathbf{X}_0 = (1, 0, 0, 0, 0, 0)^T$, $\mathbf{X}_1$ representing $\mathbf{x}_1$ has the form $\mathbf{X}_1 = (0, 1, 0, 0, 0, 0)^T$, $\ldots$, $\mathbf{X}_5$ representing $\mathbf{x}_5$ has the form $\mathbf{X}_5 = (0, 0, 0, 0, 0, 1)^T$. By means of the vicarious vectors we can develop the system dynamics as follows

$$ \mathbf{X}_{k+1} = \mathbf{A}_k^T.\mathbf{X}_k, \quad k = 0, 1, \ldots, K. \tag{5} $$

The equation (5) describes DEDS as a state machine. In any step k only one discrete event can occur (can be fired). Such a discrete event transfer the system from $\mathbf{X}_k$ to $\mathbf{X}_{k+1}$ by means of the discrete event fired in the step k.

2.3 The Example of the More Complicated DEDS

Consider the more complicated system (as to the compositional structure) with the behaviour expressed by the directed graph in Figure 4. As we can see there are three mutually different possibilities of the transition to the next state already in the initial state $\mathbf{x}_0$. The initial state vector is the first feasible vector of the system – i.e. $\mathbb{X}_1 \triangleq \mathbf{x}_0$. It can be seen from the same figure that $\mathbf{x}_1 \in \{\mathbb{X}_2, \mathbb{X}_3, \mathbb{X}_4\}$ – i.e. either $\mathbb{X}_2 \triangleq \mathbf{x}_1$ or $\mathbb{X}_3 \triangleq \mathbf{x}_1$ or $\mathbb{X}_4 \triangleq \mathbf{x}_1$. Analogically, $\mathbf{x}_2 \in \{\mathbb{X}_5, \mathbb{X}_6, \mathbb{X}_7\}$, $\mathbf{x}_3 \in \{\mathbb{X}_8, \mathbb{X}_9\}$, $\mathbf{x}_4 \in \{\mathbb{X}_{10}, \mathbb{X}_{11}\}$, $\mathbf{x}_5 \in \{\mathbb{X}_{12}, \mathbb{X}_{13}, \mathbb{X}_{14}\}$. The matrix $\mathbf{X}_{reach} = (\mathbb{X}_1, \mathbb{X}_2, \ldots, \mathbb{X}_{14})$.

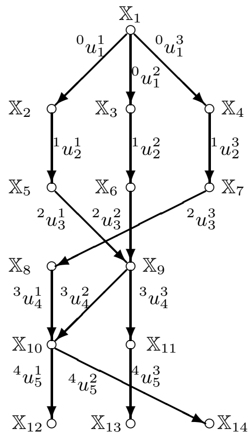


Fig. 4. The example of the more complicated DEDS

The system structure is created by the digraph with the vectors $\mathbb{X}_i$, $i = 1, 2, \ldots,$ 14 being its nodes. Consequently, the adjacency matrix of the graph is the following (14×14)-dimensional matrix:

$$\mathbf{A} = \begin{pmatrix}
0 & {}^0u_1^1 & {}^0u_1^2 & {}^ku_1^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & {}^1u_2^1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & {}^1u_2^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & {}^1u_2^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & {}^2u_3^1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & {}^2u_3^2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & {}^2u_3^3 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & {}^3u_4^1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & {}^3u_4^2 & {}^3u_4^3 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & {}^4u_5^1 & 0 & {}^4u_5^2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & {}^4u_5^3 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix} \quad (6)$$

In the k-variant adjacency matrix $\mathbf{A}_k$ only such discrete events appear which have its left upper index equal to k. It means that ${}^0u_1^1$, ${}^0u_1^2$, ${}^0u_1^3$ appear in $\mathbf{A}_0$, ${}^1u_2^1$, ${}^2u_2^2$, ${}^2u_2^3$ appear in $\mathbf{A}_1$, etc. Here it has to be repeated again that only one of the enabled discrete events can be fired in any step of the system dynamics development.

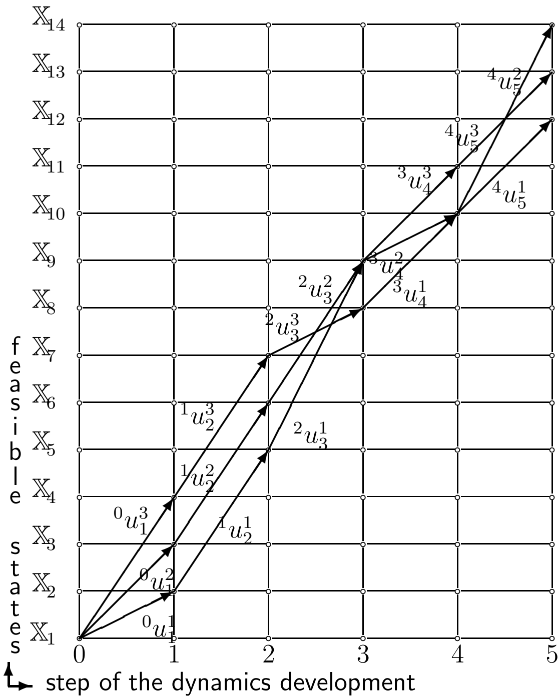


Fig. 5. The graphical expression all of the feasible trajectories

Introducing the 14-dimensional vicarious vectors $\mathbf{X}_k$, $k = 0, \ldots K$ taking the places of the vectors $\mathbf{x}_k$, $k = 0, 1, \ldots, K$ we can develop the system dynamics by means of (5). The system trajectories are given in Figure 5.

2.4 The Formal Expression of DEDS Causality

The functional (k-variant) matrix $\mathbf{A}_k$ is the transition matrix between two *causally adjacent* states $\mathbf{X}_k$, $\mathbf{X}_{k+1}$, $k = 0, 1, \ldots, K$. Consequently,

$$\mathbf{X}_k = \mathbf{A}_{k-1}^T. \ldots .\mathbf{A}_0^T.\mathbf{X}_0 = (\prod_{i=1}^{k} \mathbf{A}_{k-i}^T).\mathbf{X}_0 = (\prod_{i=0}^{k-1} \mathbf{A}_i)^T.\mathbf{X}_0 = \mathbf{\Phi}_{k,0}.\mathbf{X}_0 \qquad (7)$$

because $\mathbf{X}_1 = \mathbf{A}_0^T.\mathbf{X}_0$, $\mathbf{X}_2 = \mathbf{A}_1^T.\mathbf{X}_1 = \mathbf{A}_1^T.\mathbf{A}_0^T.\mathbf{X}_0$, $\ldots$, $\mathbf{X}_k = \mathbf{A}_{k-1}^T.\mathbf{X}_{k-1} = \mathbf{A}_{k-1}^T.\mathbf{A}_{k-2}^T. \ldots .\mathbf{A}_0^T.\mathbf{X}_0$. The matrix $\mathbf{\Phi}_{k,0}$ is the transition matrix of the system from the state $\mathbf{X}_0$ to the state $\mathbf{X}_k$.

2.5 The DEDS Backward Causality

Till now we considered the natural DEDS causality – the causality in straightforward direction, i.e. in the direction *cause* $\stackrel{event}{\longrightarrow}$ *consequence*. In such a case the system being in the state $\mathbf{X}_k$ passes to another state $\mathbf{X}_{k+1}$ in a natural way (i.e. owing to the occurrence of a discrete event). Thus the transformation $\mathbf{X}_k \stackrel{event}{\longrightarrow} \mathbf{X}_{k+1}$ is realized. This kind of causality determines the future on the basis of the presence. However, there are situations (the DEDS control or, better said, the need of the DEDS control synthesis is one of them), when it is important to answer the question what event (symbolically *"?event"*) caused that the system finds itself in the present state $\mathbf{X}_k$ and/or from which previous state $\mathbf{X}_{k-1}$ (symbolically *"?*$\mathbf{X}_{k-1}$*"*) the system passed (due to the *event*) into the state $\mathbf{X}_k$ – i.e. $?\mathbf{X}_{k-1} \stackrel{?event}{\longrightarrow} \mathbf{X}_k$. Thus, the question marks symbolize the questions *"from which previous state?"* and *"by means of what discrete event?"* the system passed to the state $\mathbf{X}_k$. Usually, any unambiguous answer does not exist in such a case. The system could pass into the state $\mathbf{X}_k$ from several (e.g. n_{cp}) different previous states ${}^i\mathbf{X}_{k-1}$ (causal *predecessors*), $i = 1, 2, \ldots, n_{cp}$, namely owing to different discrete events being the elements of the matrix $\mathbf{A}_{k-1}$. For example, in Figure 5 the feasible state $\mathbb{X}_9$ corresponding to the real state $\mathbf{x}_3$ (represented by the vicarious state $\mathbf{X}_3$) can be reached either from $\mathbb{X}_5$ (corresponding to ${}^1\mathbf{x}_2$ represented by ${}^1\mathbf{X}_2$), namely by means of the event ${}^2u_3^1$, or from $\mathbb{X}_6$ (corresponding to ${}^2\mathbf{x}_2$ represented by ${}^2\mathbf{X}_2$), namely by means of the event ${}^2u_3^2$.

This kind of causality finds possible causes in the past the consequence of which is just the presence (the existing state of the system). The backward causality is very important especially during the DEDS control synthesis. The mathematical description of the backward causality is (at the above introduced conception of causality) very simple. It is sufficient to transpose the transition matrix $\mathbf{A}_k^T$ (i.e. to use the matrix $\mathbf{A}_k$) in order to obtain such a procedure. For one step the description is as follows:

$$\mathbf{X}_{k-1} = \mathbf{A}_{k-1}.\mathbf{X}_k. \tag{8}$$

In general (for $i = 1, 2, \ldots$), $\mathbf{X}_{k-i} = \mathbf{A}_{k-i}.\mathbf{A}_{k-i+1}. \ldots .\mathbf{A}_{k-1}.\mathbf{X}_k$. Thus,

$$\mathbf{X}_0 = \mathbf{A}_0.\mathbf{A}_1. \ldots .\mathbf{A}_{k-2}.\mathbf{A}_{k-1}.\mathbf{X}_k = (\prod_{i=0}^{k-1} \mathbf{A}_i).\mathbf{X}_k = \mathbf{\Phi}_{0,k}.\mathbf{X}_k. \tag{9}$$

The matrix $\mathbf{\Phi}_{0,k}$ is the backward transition matrix of the system from the state $\mathbf{X}_k$ to the state $\mathbf{X}_0$. It is evident from comparing (7) and (9) that $\mathbf{\Phi}_{0,k} = \mathbf{\Phi}_{k,0}{}^T$.

3 SIMPLIFIED EXPRESSING OF THE CAUSALITY

Using the k-variant adjacency matrix $\mathbf{A}_k$ is not comfortable as to the computations. The standard (i.e. constant) adjacency matrix $\mathbf{A}$ seems to be more suitable because the graph theory results can be utilized.

Namely, the relations between powers of the digraph adjacency matrix $\mathbf{A}$ and the number of paths having a given length are defined and proved – see e.g. [20, 22] and/or [10, 11, 14, 19, 24, 26]. The digraph reachability matrix is defined there as well.

Therefore, the element $a_{i,j}^{(k)}$ of the k^{th} power $\mathbf{A}^k$ of the adjacency matrix $\mathbf{A}$ represents the number of the paths having the length k from the node i to the node j. The reachability matrix $\mathbf{R} = \sum_{i=1}^{N} \mathbf{A}^i = \mathbf{A}^1 + \mathbf{A}^2 + \ldots + \mathbf{A}^N$ yields information about the number of paths having the length N or the length less than N. Replacing the ordinary arithmetic (in general e.g. $c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj}$) by the Boolean arithmetic (accordingly, $c_{ij} = \bigvee_{k=1}^{n} a_{ik} \wedge b_{kj}$), the element $a_{i,j}^{(k)}$ of the matrix $\mathbf{A}^k$ is Boolean and expresses the nonexistence or existence of the path from the node i to the node j having the length k. The elements of the logical reachability matrix $\mathbf{R}_L = \bigvee_{i=1}^{N} \mathbf{A}^i = \mathbf{A}^1 \vee \mathbf{A}^2 \vee \ldots \vee \mathbf{A}^N$ yield information about the reachability in itself (decide the reachability).

The nonzero elements $r_{jk}, j = 1, 2, \ldots, N$, of the k^{th} column $(r_{1k}, r_{2k}, \ldots, r_{jk}, \ldots, r_{Nk})^T$ of the reachability matrix $\mathbf{R}$ contain the numbers of paths from the node j to the node k, while the elements of the same vector in the logical reachability matrix $\mathbf{R}_L$ decide whether the node k is reachable from the node j or not.

3.1 Utilizing the Constant Adjacency Matrix

Consider formally that all of the discrete events are enabled. Consequently, the k-variant adjacency matrix $\mathbf{A}_k$ is turned to the constant adjacency matrix $\mathbf{A}$ having the same structure like in (6) however, all of the nonzero elements are replaced by the integer 1. Such an approach helps us find the space of feasible trajectories from a given initial state to a prescribed terminal state. Using the matrix $\mathbf{A}$ in the straight-lined development is as follows:

$$\{\mathbf{X}_k\} = \underbrace{\mathbf{A}^T.\mathbf{A}^T.\ldots.\mathbf{A}^T.\mathbf{A}^T}_{k \text{ factors}}.\mathbf{X}_0 = (\prod_{i=0}^{k-1} \mathbf{A}^T).\mathbf{X}_0 = (\prod_{i=0}^{k-1} \mathbf{A})^T.\mathbf{X}_0 = {}^{c}\mathbf{\Phi}_{k,0}.\mathbf{X}_0 \quad (10)$$

because $\{\mathbf{X}_1\} = \mathbf{A}^T.\mathbf{X}_0$, $\{\mathbf{X}_2\} = \mathbf{A}^T.\{\mathbf{X}_1\} = \mathbf{A}^T.\mathbf{A}^T.\mathbf{X}_0$, $\ldots$, $\{\mathbf{X}_k\} = \mathbf{A}^T.\{\mathbf{X}_{k-1}\}$ $= \underbrace{\mathbf{A}^T.\mathbf{A}^T.\ \ldots\ .\mathbf{A}^T}_{k \text{ factors}}.\mathbf{X}_0$. Here, the $\{\mathbf{X}_i\}$, $i = 1, 2, \ldots, k$ express aggregated states because of the fact that all of the discrete events are (formally) fired. Denote these vectors as ${}^{sl}\{\mathbf{X}_i\}$, $i = 1, 2, \ldots, k$ (because they represent the straight-lined development) and store them as the columns of the matrix

$$\mathbf{M}_1 = ({}^{sl}\mathbf{X}_0, {}^{sl}\{\mathbf{X}_1\}, \ldots, {}^{sl}\{\mathbf{X}_k\}). \quad (11)$$

Analogically, the backward development is performed as follows

$$\{\mathbf{X}_0\}' = \underbrace{\mathbf{A}.\mathbf{A}.\ldots.\mathbf{A}.\mathbf{A}}_{k \text{ factors}}.\mathbf{X}_k = (\prod_{i=0}^{k-1}\mathbf{A}).\mathbf{X}_0 = {}^{c}\mathbf{\Phi}_{0,k}.\mathbf{X}_0 = {}^{c}\mathbf{\Phi}^{T}_{k,0}.\mathbf{X}_0 \qquad (12)$$

because $\{\mathbf{X}_{k-1}\}' = \mathbf{A}.\mathbf{X}_k$, $\{\mathbf{X}_{k-2}\}' = \mathbf{A}.\{\mathbf{X}_{k-1}\}' = \mathbf{A}^2.\mathbf{X}_k$, $\ldots$, $\{\mathbf{X}_0\}' = \mathbf{A}.\{\mathbf{X}_{k-1}\}' = \underbrace{\mathbf{A}.\mathbf{A}.\ \ldots\ .\mathbf{A}}_{k \text{ factors}}.\mathbf{X}_k$. Here, the $\{\mathbf{X}_i\}'$, $i = 0, 1, \ldots, k-1$ also means aggregated states because of the same reason. Denote these vectors as ${}^{bt}\{\mathbf{X}_i\}$, $i = 0, 1, \ldots, k-1$ (because they represent the backtracking development) and store them as the columns of the matrix

$$\mathbf{M}_2 = ({}^{bt}\{\mathbf{X}_0\}, {}^{bt}\{\mathbf{X}_1\}, \ldots, {}^{bt}\{\mathbf{X}_{k-1}\}, {}^{bt}\mathbf{X}_k). \qquad (13)$$

In both cases such an approach has the real interpretation. The nonzero components of the columns of $\mathbf{M}_1$ express the number of paths through the corresponding graph nodes in the straight-lined direction, i.e. from the initial state $\mathbf{X}_0$ to the prescribed terminal state $\mathbf{X}_k$. The nonzero components of the columns of $\mathbf{M}_2$ contain the number of paths through the corresponding graph nodes in the backward direction, i.e. from the terminal state $\mathbf{X}_k$ to the initial state $\mathbf{X}_0$, however, directed towards the terminal state.

3.2 The Space of Feasible Trajectories

After the column-to-column intersection of the matrices $\mathbf{M}_1$, $\mathbf{M}_2$ we have the final result in the form of the matrix

$$\mathbf{M} = \mathbf{M}_1 \cap \mathbf{M}_2 = ({}^{f}\mathbf{X}_0, {}^{f}\{\mathbf{X}_1\}, \ldots, {}^{f}\{\mathbf{X}_{k-1}\}, {}^{f}\mathbf{X}_k) \qquad (14)$$

$$\mathbf{M} = ({}^{sl}\mathbf{X}_0 \cap {}^{bt}\{\mathbf{X}_0\}, {}^{sl}\{\mathbf{X}_1\} \cap {}^{bt}\{\mathbf{X}_1\}, \ldots, {}^{sl}\{\mathbf{X}_{k-1}\} \cap {}^{bt}\{\mathbf{X}_{k-1}\}, {}^{sl}\{\mathbf{X}_k\} \cap {}^{bt}\mathbf{X}_k). \qquad (15)$$

The column-to-column intersection of two corresponding columns is understood to be finding minima of their corresponding elements. It is done as follows

$${}^{f}\{\mathbf{X}_i\} = {}^{sl}\{\mathbf{X}_i\} \cap {}^{bt}\{\mathbf{X}_i\} = min({}^{sl}\{\mathbf{X}_i\}, {}^{bt}\{\mathbf{X}_i\}),\ i = 0, 1, \ldots, k \qquad (16)$$

with ${}^{sl}\{\mathbf{X}_0\} = \mathbf{X}_0, {}^{bt}\{\mathbf{X}_k\} = \mathbf{X}_k$. The operation of the intersection ensures that in the matrix $\mathbf{M}$ only the paths emerging from the given initial state and entering the prescribed terminal state are stored. It can be said that in the matrix $\mathbf{M}$ the feasible trajectories are stored or, in other words, that $\mathbf{M}$ represents the space of the feasible trajectories.

It is very important and interesting that the principle of causality allows us to find shorter trajectories when the longer ones have already been computed. Namely, having at disposal the matrices $\mathbf{M}_1$, $\mathbf{M}_2$ (given, respectively, by (11), (13)) we can compute trajectories shorter for $1, 2, \ldots, j$ steps in such a way that before the intersection of these matrices we shift the matrix $\mathbf{M}_2$ to the left for $1, 2, \ldots, j$ columns as follows

$$
\begin{aligned}
{}^{-1}\mathbf{M} &= \mathbf{M}_1 \cap {}^{-1}\mathbf{M}_2 \ , \ \text{where} \ \ {}^{-1}\mathbf{M}_2 = ({}^{bt}\{\mathbf{X}_1\}, \dots, {}^{bt}\{\mathbf{X}_{k-1}\}, {}^{bt}\mathbf{X}_k, \boldsymbol{\emptyset}) && (17)\\
{}^{-2}\mathbf{M} &= \mathbf{M}_1 \cap {}^{-2}\mathbf{M}_2 \ , \ \text{where} \ \ {}^{-2}\mathbf{M}_2 = ({}^{bt}\{\mathbf{X}_2\}, \dots, {}^{bt}\{\mathbf{X}_{k-1}\}, {}^{bt}\mathbf{X}_k, \boldsymbol{\emptyset}, \boldsymbol{\emptyset}) && (18)\\
&\vdots\\
{}^{-j}\mathbf{M} &= \mathbf{M}_1 \cap {}^{-2}\mathbf{M}_2 \ , \ \text{where} \ \ {}^{-j}\mathbf{M}_2 = ({}^{bt}\{\mathbf{X}_j\}, \dots, {}^{bt}\{\mathbf{X}_{k-1}\}, {}^{bt}\mathbf{X}_k, \underbrace{\boldsymbol{\emptyset}, \dots, \boldsymbol{\emptyset}}_{j \text{ vectors}}). && (19)
\end{aligned}
$$

Here, $\boldsymbol{\emptyset}$ is the zero column vector of the corresponding dimensionality. When an intersection $\mathbf{M}_1 \cap {}^{-j}\mathbf{M}_2$ does not exist, the matrix ${}^{-j}\mathbf{M}$ is the zero matrix. It means that no trajectory shorter for j steps exists in such a case.

3.2.1 The Illustrative Example

To illustrate the procedure let us apply it to Figure 5. The real initial state vector is $\mathbf{x}_0$. It is the feasible state and thus $\mathbb{X}_1 \triangleq \mathbf{x}_0$. The vicarious vector $\mathbf{X}_0$ represents the initial state vector in order to work with the adjacency matrix $\mathbf{A}$. When the feasible state $\mathbb{X}_{12}$ is prescribed to be the terminal state, after 5 steps we have

$$
\mathbf{M}_1 = \begin{pmatrix}
1&0&0&0&0&0\\
0&1&0&0&0&0\\
0&1&0&0&0&0\\
0&1&0&0&0&0\\
0&0&1&0&0&0\\
0&0&1&0&0&0\\
0&0&1&0&0&0\\
0&0&0&1&0&0\\
0&0&0&2&0&0\\
0&0&0&0&3&0\\
0&0&0&0&2&0\\
0&0&0&0&0&3\\
0&0&0&0&0&2\\
0&0&0&0&0&3
\end{pmatrix}
\mathbf{M}_2 = \begin{pmatrix}
3&0&0&0&0&0\\
0&1&0&0&0&0\\
0&1&0&0&0&0\\
0&1&0&0&0&0\\
0&0&1&0&0&0\\
0&0&1&0&0&0\\
0&0&1&0&0&0\\
0&0&0&1&0&0\\
0&0&0&1&0&0\\
0&0&0&0&1&0\\
0&0&0&0&0&0\\
0&0&0&0&0&1\\
0&0&0&0&0&0\\
0&0&0&0&0&0
\end{pmatrix}
\mathbf{M} = \begin{pmatrix}
1&0&0&0&0&0\\
0&1&0&0&0&0\\
0&1&0&0&0&0\\
0&1&0&0&0&0\\
0&0&1&0&0&0\\
0&0&1&0&0&0\\
0&0&1&0&0&0\\
0&0&0&1&0&0\\
0&0&0&1&0&0\\
0&0&0&0&1&0\\
0&0&0&0&0&0\\
0&0&0&0&0&1\\
0&0&0&0&0&0\\
0&0&0&0&0&0
\end{pmatrix}
$$

However, it has to be said that the number representing the step k in which the terminal state will be reached from the initial state is not predetermined. It is bounded only by the relation $k \le (N-1)$ which results from graph theory (where $(N-1)$ is the maximal length of the paths in the graph with N nodes) as well as by the relation ${}^{sl}\{\mathbf{X}_k\} \ge \mathbf{X}_k$ (to be sure that $\mathbf{X}_k$ is comprehended in ${}^{sl}\{\mathbf{X}_k\}$). When such a k is found, the number of the columns of $\mathbf{M}_1$ is determined and the backtracking development starting from $\mathbf{X}_k$ can be performed in order to obtain the matrix $\mathbf{M}_2$.

Thus, in our case $k = 5$, $\mathbb{X}_{12} \triangleq \mathbf{x}_5$, where $\mathbf{x}_5$ is represented by $\mathbf{X}_5$. The first column of the matrix $\mathbf{M}$ is created by the initial vicarious state vector $\mathbf{X}_0$

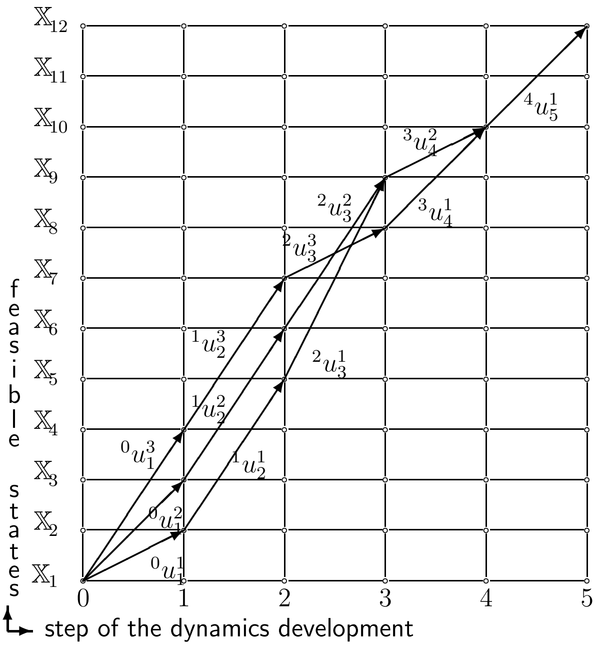


Fig. 6. The graphical expression of the feasible trajectories from $\mathbb{X}_1$ to $\mathbb{X}_{12}$

representing the real initial state vector $\mathbf{x}_0$ ($\mathbb{X}_1 \triangleq \mathbf{x}_0$). The last $(k+1)$-th column is created by the terminal vicarious state vector $\mathbf{X}_5$ representing the real terminal state vector $\mathbf{x}_5$, where $\mathbb{X}_{12} \triangleq \mathbf{x}_5$. The columns placed between them point out (by their nonzero elements) the feasible transitive states as well as the number of paths through them. The matrix $\mathbf{M}$ stores the feasible trajectories from the feasible state $\mathbb{X}_1$ to the desired feasible terminal state $\mathbb{X}_{12}$. We can distinguish the trajectories as well as make sure of the correctness of the trajectories by means of the matrix $\mathbf{A}$ displayed in (6). Namely, this matrix guarantees causality. At the same time we can assign the corresponding discrete event realizing the transition between any adjacent feasible states $(\mathbb{X}_i, \mathbb{X}_j)$ being a part of the trajectories in question. Consequently,

$$\mathbb{X}_1 \xrightarrow{^0u_1^1} \mathbb{X}_2 \xrightarrow{^1u_2^1} \mathbb{X}_5 \xrightarrow{^2u_3^1} \mathbb{X}_9 \xrightarrow{^3u_4^2} \mathbb{X}_{10} \xrightarrow{^4u_5^1} \mathbb{X}_{12} \tag{20}$$

$$\mathbb{X}_1 \xrightarrow{^0u_1^2} \mathbb{X}_3 \xrightarrow{^1u_2^2} \mathbb{X}_6 \xrightarrow{^2u_3^2} \mathbb{X}_9 \xrightarrow{^3u_4^2} \mathbb{X}_{10} \xrightarrow{^4u_5^1} \mathbb{X}_{12} \tag{21}$$

$$\mathbb{X}_1 \xrightarrow{^0u_1^3} \mathbb{X}_4 \xrightarrow{^1u_2^3} \mathbb{X}_7 \xrightarrow{^2u_3^3} \mathbb{X}_8 \xrightarrow{^3u_4^1} \mathbb{X}_{10} \xrightarrow{^4u_5^1} \mathbb{X}_{12}. \tag{22}$$

The graphical expression of the trajectories is given in Figure 6.

4 THE PN-BASED MATHEMATICAL MODEL OF DEDS

Use the analogy between the DEDS atomic activities $a_i \in \{a_1, \dots, a_n\}$ and the PN places $p_i \in \{p_1, \dots, p_n\}$ as well as between the discrete events $u_j \in \{u_1, \dots, u_m\}$ occurring in DEDS and the PN transitions $t_j \in \{t_1, \dots, t_m\}$. Consequently, DEDS can be modelled by means of PN. The analytical model of DEDS based on PN has the form of the linear discrete system as follows

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{B}.\mathbf{u}_k \quad , \quad k = 0, \dots, K \tag{23}$$

$$\mathbf{B} = \mathbf{G}^T - \mathbf{F} \tag{24}$$

$$\mathbf{F}.\mathbf{u}_k \leq \mathbf{x}_k \tag{25}$$

where k is the discrete step of the dynamics development; $\mathbf{x}_k = (\sigma^k_{p_1}, \dots, \sigma^k_{p_n})^T$ is the n-dimensional state vector in the step k; $\sigma^k_{p_i} \in \{0, 1, \dots, c_{p_i}\}$, $i = 1, \dots, n$ express the states of the DEDS atomic activities, namely the passivity is expressed by $\sigma_{p_i} = 0$ and the activity is expressed by $0 < \sigma_{p_i} \leq c_{p_i}$; c_{p_i} is the capacity as to the activities – e.g. the passivity of a buffer means the empty buffer, the activity means a number of parts stored in the buffer and the capacity is understood to be the maximal number of parts which can be put into the buffer; $\mathbf{u}_k = (\gamma^k_{t_1}, \dots, \gamma^k_{t_m})^T$ is the m-dimensional control vector of the system in the step k; its components $\gamma^k_{t_j} \in \{0, 1\}$, $j = 1, \dots, m$ represent the occurrence of the DEDS discrete events (e.g. starting or ending the atomic activities, occurrence of failures, etc.) – when the j^{th} discrete event is enabled $\gamma^k_{t_j} = 1$, when the event is disabled $\gamma^k_{t_j} = 0$; $\mathbf{B}$, $\mathbf{F}$, $\mathbf{G}$ are structural matrices of constant elements; $\mathbf{F} = \{f_{ij}\}_{n \times m}$, where $f_{ij} \in \{0, M_{f_{ij}}\}, i = 1, \dots, n, j = 1, \dots, m$ express the causal relations between the states of the DEDS (in the role of causes) and the discrete events occuring during the DEDS operation (in the role of consequences) – nonexistence of the corresponding relation is expressed by $M_{f_{ij}} = 0$, existence and multiplicity of the relation are expressed by $M_{f_{ij}} > 0$; $\mathbf{G} = \{g_{ij}\}_{m \times n}$, where $g_{ij} \in \{0, M_{g_{ij}}\}, i = 1, \dots, m, j = 1, \dots, n$ express very analogically the causal relations between the discrete events (as the causes) and the DEDS states (as the consequences); the structural matrix $\mathbf{B}$ is given by means of the arcs incidence matrices $\mathbf{F}$ and $\mathbf{G}$ according to (24); $(.)^T$ symbolizes the matrix or vector transposition.

The PN marking which in PN theory is usually denoted as μ was denoted here by the letter $\mathbf{x}$ usually denoting the state in system theory. The main reason is that we work with the term "system" rather than with the term "PN". Fuzzy PN [15] can be modelled analogically – see [3]. Using the higher-level PN is not discussed in this paper.

4.1 The PN Reachability Tree and the Reachability Graph

Exact definitions of the reachability tree are introduced in PN theory basic sources – see e.g. [16, 18]. To have an idea about the reachability tree it is sufficient to introduce here only a short description of it. The PN reachabitity tree $G_{rt} = (V_{rt}, E_{rt})$ is the tree where the set of nodes $V_{rt} = \{v_0, v_1, \dots, v_{N_r}\}$ is represented by the set of PN states – i.e. the nodes v_i, $i = 0, \dots, N_r$ represent the state vectors $\mathbf{x}_i$, $i = 0, \dots, N_r$ and consequently, $V_{rt} = \{\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_{N_r}\}$, with the initial state $\mathbf{x}_0$ being the root of the reachability tree. The set of edges $E_{rt} = \{e_1, e_2, \dots, e_M\}$ consists of egdes marked by the PN transitions $t_j \in T, j = 1, \dots, m$. Namely, two nodes $v_i, v_j \in V$ are connected by the oriented arc $e = e_{v_i \to v_j} \in E$ directed from v_i towards v_j. The arc is marked by the PN transition $t = t_{v_i \to v_j} = t_{\mathbf{x}_i \to \mathbf{x}_j} \in T$ just when it is enabled in the state $\mathbf{x}_i$ and by its firing the new state $\mathbf{x}_j$ will be reached. The reachability tree has to involve a corresponding node for every PN state and a corresponding edge for any PN transition enabled in the given state. To avoid complications (especially the infiniteness of the generated reachability tree) the so called duplicity nodes are defined as the leaf nodes (the graph leaves). In such a way subtrees which have been included into the reachability tree already are eliminated. Namely, the node $v_i \in V_{rt}$ for which there is a node $v_j \in V_{rt}$ such that $v_j \prec v_i$ is named the duplicity node. Here the operator $\prec$ represents the binary antireflexive transitive antisymetric relation expressing the ordering of nodes in the set V_{rt}. Consequently, from the duplicity node no edges emerge. Connecting all duplicity nodes of the node $v_j \in V_{rt}$ together and also with the node v_j itself we obtain (after doing this for $j = 1, \dots, N_r$) the reachability graph from the reachability tree. It is important that both the reachability graph and the reachability tree have the same adjacency matrix.

Another descriptive definition of the reachability graph is presented in [12]. It is distinctive on it that it starts by defining two functions

- *enabled*(v): given a state v, this function returns the set of transitions t that are enabled in v
- $fire(v, t)$: given a state v, and a transition $t \in enabled(v)$, this function returns the state v' reached from v by firing t

and consequently, the reachability graph is defined as $G_{rt} = (V_{rt}, T, E_{rt}, v_0)$ being the graph with the smallest sets of nodes V_{rt}, transitions T, and edges E_{rt} such that

- $v_0 \in V_{rt}$, where v_0 is the initial state of the system, and
- if $v \in V_{rt}$, then for all $t \in enabled(v)$ it holds that $t \in T$, $fire(v, t) \in V_{rt}$, and $(v, t, fire(v, t)) \in E_{rt}$.

In PN theory the reachability tree and/or graph are very important, especially for testing the PN properties. In this paper it will be utilized at the DEDS control synthesis. The Matlab procedure for computing the G_{rt} was introduced in [9]. Its entries are the PN incidence matrices $\mathbf{F}$, $\mathbf{G}$ and the initial state $\mathbf{x}_0$. The procedure computes the adjacency matrix $\mathbf{A}_{rt}$ of G_{rt} in the quasi-functional form $\mathbf{A}_{rt}(k)$ (with integers representing the indices of the transitions as its nonzero elements) and the set of the reachable states given as columns of the matrix $\mathbf{X}_{reach}$. These matrices fully characterize the PN reachability graph. The functional adjacency matrix $\mathbf{A}_k$ corresponding to the quasi-functional matrix $\mathbf{A}_{rt}(k)$ can be constructed when the integers in $\mathbf{A}_{rt}$ representing the indices of PN transitions $t_j \in T, j = 1, \dots, m$ are replaced by the transition functions $\gamma_{t_j}^k, j = 1, \dots, m, k = 0, 1 \dots, K$. This matrix represents the PN causality. It can be said that when DEDS is modelled by PN, the strictness of the DEDS causality is rigorously adhered by the PN reachability tree and/or the reachability graph. Therefore, the reachability graph cannot be avoided at the DEDS control synthesis, of course.

Applying the results of the previous sections (concerning the DEDS causality) to the PN reachability graph we can utilize them at modelling, analysing and control of interactions among agents. However, it is necessary to keep in mind that the number of feasible states $N = N_r + 1$ because also the initial state is one of them.

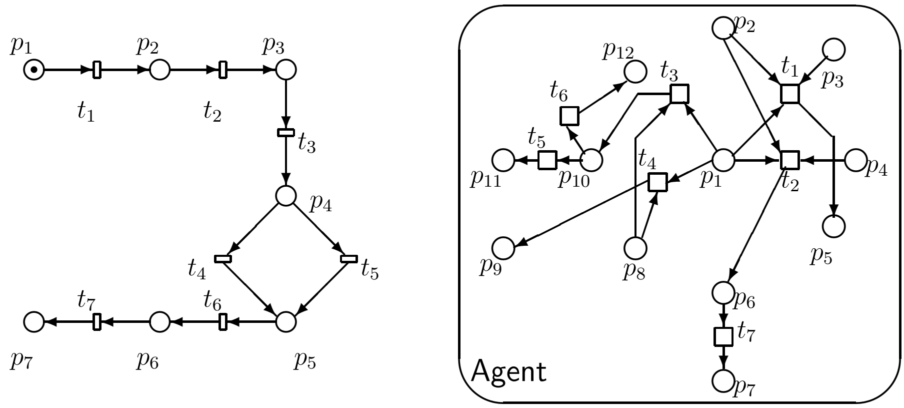
The attention will be focused especially on the negotiation process.

5 THE PRINCIPLE OF THE AGENTS NEGOTIATION PROCESS

The negotiation process itself consists of several principle activities [13]. Especially, the following are most important: defining the negotiation environment, initial contact of agents, offer(s) and counter offer(s) among them, evaluation of proposals, and outcomes of the negotiation process. The coordination plan of the negotiation process can be formally described by DEDS modelled by PN as can be seen on the left in Figure 7 a). The PN places represent the activities and the PN transitions represent the discrete events. The interpretation of the places and transitions is as follows: p_1 = start; p_2 = define negotiation environment; p_3 = initial contact; p_4 = offer(s) and counter offer(s); p_5 = evaluation; p_6 = outcomes; p_7 = end; t_1 = starting negotiation process; t_2 = negotiation plan(s); t_3 = "hand shake"; t_4 = proposal(s); t_5 = revised proposal(s); t_6 = agreement or quit; t_7 = ending negotiation process. However, the reality is more complicated. To illustrate it in details, let us introduce the PN-based model of the agent in general and the cooperation of two agents. The possible cooperation of several agents will be pointed out too.

6 THE PN-BASED MODEL OF AGENTS IN MAS

In general, the compositional structure of an agent from the collaboration/negotiation point of view can looks like that given in Figure 7 b). The PN-based model represents the atomic activities of the agent as well as their mutual interconnection. The interpretation of the PN places is as follows: p_1 = the agent (A_1) is free; p_2 = a problem has to be solved by A_1; $p_3 = A_1$ is able to solve the problem (P_{A_1}); $p_4 = A_1$ is not able to solve P_{A_1}; $p_5 = P_{A_1}$ is solved; $p_6 = P_{A_1}$ cannot be solved by A_1 and another agent(s) should be contacted; $p_7 = A_1$ asks another agent(s) to help him solve P_{A_1}; $p_8 = A_1$ is asked by another agent(s) to solve a problem P_B; $p_9 = A_1$ refuses the help; $p_{10} = A_1$ accepts the request of another agent(s) for help; $p_{11} = A_1$ is not able to solve P_B; $p_{12} = A_1$ is able to solve P_B. However, it has to



a) b)

Fig. 7. The PN-based model of: a) the coordination plan of the negotiation process in general; b) a compositional structure of the agent

be said that another concept of both the agent structure and the activities is not excluded.

In our case parameters of the PN-based model are as follows:

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

Analyse e.g. the situation when $\mathbf{x}_0 = (1,0,0,0,0,0,0,1,0,0,0,0)^T$, i.e. the situation when the agent A_1 is free and it is asked by the agent A_2 to solve the problem $P_B = P_{A_2}$. Using the algorithm introduced in [5] we have the following quasi-functional adjacency matrix $\mathbf{A}_{rt}(k)$ (its elements are the indices of the PN transitions) of the PN reachability tree and the matrix $\mathbf{X}_{reach}$ with columns being the feasible states (the initial state and all states reachable from this initial state):

$$\mathbf{A}_{rt}(k) = \begin{pmatrix} 0 & 3 & 4 & 0 & 0 \\ 0 & 0 & 0 & 5 & 6 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}; \ \mathbf{X}^T_{reach} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

The reachability graph (RG) of such a PN-based model is given in Figure 8.

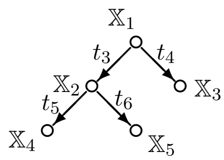


Fig. 8. The reachability graph of the agent in the situation given by the initial state

Such a PN-based model of the agent is universal and it can be used for modelling other agents of MAS too. Namely, the same interpretation of places (however with shifted numbering p_{i+12}, $i = 1, \ldots, 12$) can be used e.g. for the agent A_2.

The agent model given in Figure 7 b) can be modified by additional internal connections from t_7 to p_2 and from t_7 to p_4. Consequently, the elements g_{72}, g_{74} are added as follows:

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

Such a modification of the model does not influence the agent activities when the agent works autonomously and/or in pair. However, experiments with the models showed that the modified model has a favourable influence when the number of agents is greater than 2. Namely, when the modified model of agents is used the dimensionality of the reachability tree is less than the dimensionality of the reachability tree in case of the original model.

6.1 Analysing the Model and the Control Synthesis

Having the PN model at disposal we can use the wide spectrum of methods developed for PN, especially the methods based on PN invariants and on the PN reachability graph which are most important at testing the PN properties. When the graphical tool (suitable for drawing PN and testing the properties) is used on this way the model analysis is comfortable.

The DEDS control synthesis is based on the following simple idea:

1. developing the straight-lined reachability tree (SLRT) from the given initial state $\mathbf{x}_0$ (represented by the vicarious vector $\mathbf{X}_0$) towards the prescribed terminal state $\mathbf{x}_t$ (represented by the vicarious vector $\mathbf{X}_t$);
2. developing the backtracking reachability tree (BTRT) from $\mathbf{X}_t$ towards $\mathbf{X}_0$, however, directed to $\mathbf{X}_t$;
3. intersecting both the SLRT and BTRT.

All steps are performed numerically in Matlab. To compute SLRT and/or BTRT as well as their intersection the approach presented in Section 3 is utilized. Having the set of feasible trajectories, the most suitable one (satisfying imposed control task specifications) can be chosen. More details about the procedure can be found in [6, 9].

The GraSim graphical tool was developed to support the DEDS control synthesis. It is different from the graphical PN simulator (the tool for PN drawing and testing). The input of GraSim is the RG of the PN model. It is created by mouse clicking on appropriate icons representing the RG nodes and edges as well as the marks for designating the initial and terminal states. At its output the tool yields (in the graphical form) the feasible trajectories from a given initial state to a prescribed terminal one. The trajectories can be analysed one after another. When a trajectory is chosen for analysing, the sequence of corresponding discrete events is displayed. At present the choice of the most suitable trajectory is not self-acting yet. Namely, the choice strongly depends on the concrete kind of the system to be controlled. Therefore, a generalization in this way is problematic. Moreover, the criteria for the choice are usualy only verbal. To quantify them an appropriate knowledge representation has to be developed. However, using logical and/or fuzzylogical PN seems to be hopeful on this way.

6.2 Cooperation of Agents

Having two agents A_1, A_2, their the collaboration/negotiation is given in Figure 9. In case of more agents – e.g. N_A – the places of the agent $k = 1, \dots, N_A$ are numbered $p_{i+12.j}$, $i = 1, \dots, 12$, $j = k - 1 = 0, \dots, N_A - 1$. In case of several agents both the PN model and the RG will be more intricate. While the model size depends on modules and their interface, the RG size depends on $\mathbf{x}_0$ and on the structure of the model blocks.

As we can see we have $n = 24$ places and $m = 20$ transitions in the PN model of the two agents cooperation. However, the number of transitions is higher than a simple sum being $m = 14$. Namely, some transitions have to be added as an interface in order to connect both of the agents. Consequently, for two agents A_1,

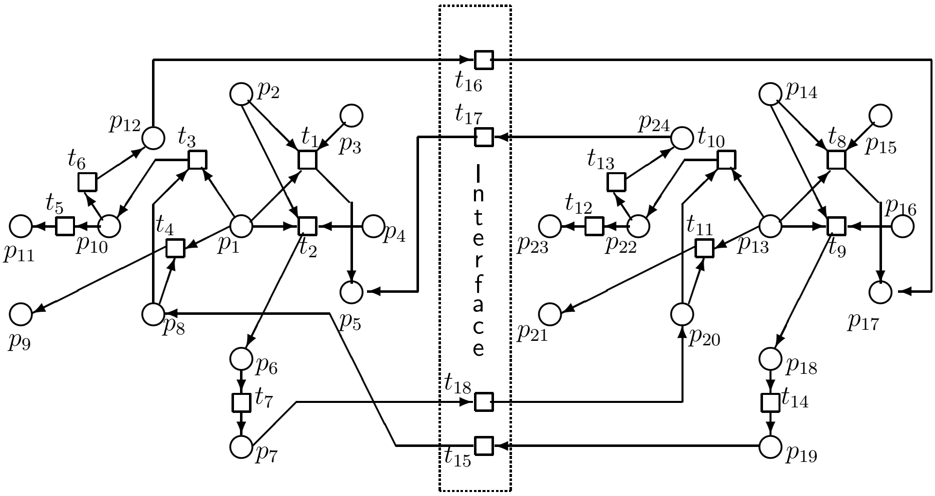


Fig. 9. The Petri net-based model of the two agents negotiation

A_2 the matrices of the MAS parameters will have the following form, where the structure of the contact interface between the agents is given by the $(n \times 6)$-dimensional matrix $\mathbf{F}_c$ and $(6 \times n)$-dimensional matrix $\mathbf{G}_c$

$$\mathbf{F} = \begin{pmatrix} \mathbf{F}_1 & \mathbf{0} & \mathbf{F}_{c_1} \\ \mathbf{0} & \mathbf{F}_2 & \mathbf{F}_{c_2} \end{pmatrix} \quad ; \quad \mathbf{G} = \begin{pmatrix} \mathbf{G}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_1 \\ \mathbf{G}_{c_1} & \mathbf{G}_{c_2} \end{pmatrix}$$

$$\mathbf{F}_{c_1} = \begin{pmatrix} 0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&1\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&1&0&0 \end{pmatrix}
\mathbf{F}_{c_2} = \begin{pmatrix} 0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\1&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&1&0 \end{pmatrix}
\mathbf{G}^T_{c_1} = \begin{pmatrix} 0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&1&0\\0&0&0&0\\0&0&0&0\\1&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0 \end{pmatrix}
\mathbf{G}^T_{c_2} = \begin{pmatrix} 0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&1&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&1\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0 \end{pmatrix}$$

In general, for agents $i = 1, 2, \ldots, N_A$ the structure of matrices is as follows:

$$\mathbf{F} = \left(\begin{array}{cccccl} \mathbf{F}_1 & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{0} & |\mathbf{F}_{c_1} \\ \mathbf{0} & \mathbf{F}_2 & \ldots & \mathbf{0} & \mathbf{0} & |\mathbf{F}_{c_2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & |\vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{F}_{N_A-1} & \mathbf{0} & |\mathbf{F}_{c_{N_A-1}} \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{F}_{N_A} & |\mathbf{F}_{c_{N_A}} \end{array}\right) \mathbf{G} = \left(\begin{array}{ccccc} \mathbf{G}_1 & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_2 & \ldots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{G}_{N_A-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{G}_{N_A} \\ \hline \mathbf{G}_{c_1} & \mathbf{G}_{c_2} & \ldots & \mathbf{G}_{c_{N_A-1}} & \mathbf{G}_{c_{N_A}} \end{array}\right)$$

$$\mathbf{B} = \left(\begin{array}{cccccl} \mathbf{B}_1 & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{0} & |\mathbf{B}_{c_1} \\ \mathbf{0} & \mathbf{B}_2 & \ldots & \mathbf{0} & \mathbf{0} & |\mathbf{B}_{c_2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & |\vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{B}_{N_A-1} & \mathbf{0} & |\mathbf{B}_{c_{N_A-1}} \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{B}_{N_A} & |\mathbf{B}_{c_{N_A}} \end{array}\right) = \left(\ \text{blockdiag}(\mathbf{B}_i)_{i=1,N_A} \quad |\mathbf{B}_c\ \right)$$

where $\mathbf{B}_i = \mathbf{G}_i^T - \mathbf{F}_i$; $\mathbf{B}_{c_i} = \mathbf{G}_{c_i}^T - \mathbf{F}_{c_i}$; $i = 1, 2 \ldots, N_A$; $\mathbf{F}_c = (\mathbf{F}_{c_1}^T, \mathbf{F}_{c_2}^T, \ldots, \mathbf{F}_{c_{N_A}}^T)^T$ $\mathbf{G}_c = (\mathbf{G}_{c_1}, \mathbf{G}_{c_2}, \ldots, \mathbf{G}_{c_{N_A}})$; $\mathbf{B}_c = (\mathbf{B}_{c_1}^T, \mathbf{B}_{c_2}^T, \ldots, \mathbf{B}_{c_{N_A}}^T)^T$ with $\mathbf{F}_i, \mathbf{G}_i, \mathbf{B}_i$, $i = 1, 2, \ldots, N_A$, representing the parameters of the PN-based model of the agent A_i, and with $\mathbf{F}_c, \mathbf{G}_c, \mathbf{B}_c$ representing the structure of the interface between the agents cooperating in MAS. The model properties can be tested by means of the graphical tool. The tool was developed in order to draw the Petri net to be tested, to simulate its dynamics development for a chosen initial marking, to compute its P-invariants and T-invariants and to compute and draw its reachability tree.

In case of two agents cooperation starting from the initial state $\mathbf{x}_0 = (\sigma_{p_1}, \sigma_{p_2}, \ldots, \sigma_{p_n})^T = ({}^{A_1}\mathbf{x}_0^T, {}^{A_2}\mathbf{x}_0^T)^T$ with ${}^{A_1}\mathbf{x}_0 = (1,1,1,0,0,0,0,0,0,0,0,0)^T$, ${}^{A_2}\mathbf{x}_0 = (1,1,0,1,0,0,0,0,0,0,0,0)^T$, i.e. from the state where only six places have nonzero marking, namely $\sigma_{p_1} = 1$, $\sigma_{p_2} = 1$, $\sigma_{p_3} = 1$, $\sigma_{p_{13}} = 1$, $\sigma_{p_{14}} = 1$, $\sigma_{p_{16}} = 1$, the reachability graph given in Figure 10 is developed.

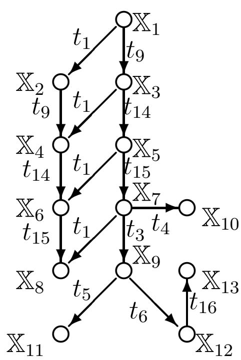


Fig. 10. The reachability graph of the two agents negotiation

It has $N = 13$ nodes representing the feasible states $\{\mathbb{X}_1, \mathbb{X}_2, \dots, \mathbb{X}_{13}\}$. The edges express feasible paths (trajectories) among the corresponding states.

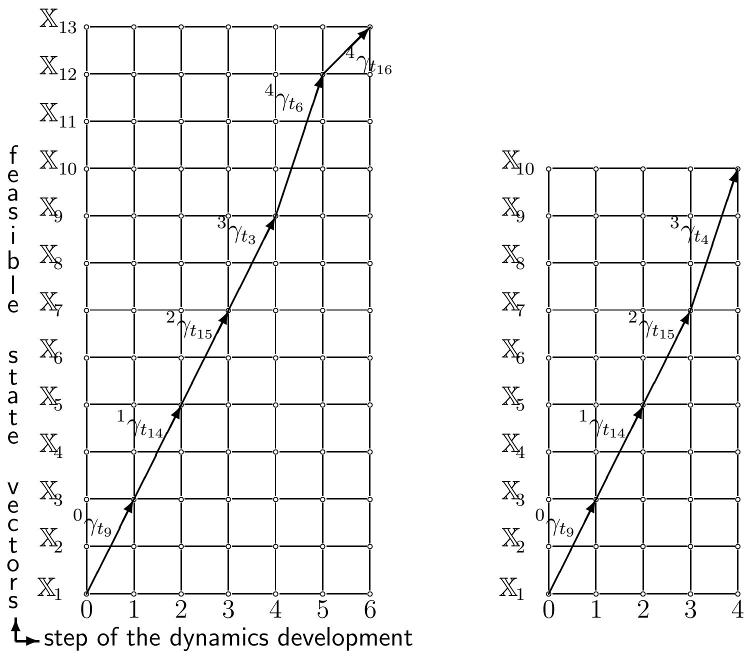
Using both graphical tools – the tool for the PN model drawing and analysing the model properties as well as the tool GraSim determined especially for finding and analysing the space of feasible trajectories and for the control synthesis – is very comfortable and user friendly. However, from the ergonomy (human factor engineering) point of view, the observed area on the monitor as well as the number of places, transitions and their interconnections in the former tool and the number of states and the intricacy of connections among them in the latter one, which the human operator is able to recognize, are limited. Hence, more formal approaches are found for DEDS analysis and control. Thus, large-scale mathematical models can be handled numerically (e.g. by means of the Matlab procedures). The matrices $\mathbf{A}_{rt}(k)$ and $\mathbf{X}_{reach}$ characterizing the RG are as follows

$$\mathbf{A}_{rt}(k) = \begin{pmatrix} 0&1&9&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&9&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&14&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&14&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&15&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&15&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&3&4&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&5&6&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&16 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0 \end{pmatrix} ; \ \mathbf{X}_{reach} = \begin{pmatrix} {}^1\mathbf{X}_{reach} \\ {}^2\mathbf{X}_{reach} \end{pmatrix}$$

$${}^1\mathbf{X}_{reach} = \begin{pmatrix} 1&0&1&0&1&0&1&0&0&0&0&0&0 \\ 1&0&1&0&1&0&1&0&1&1&1&1&1 \\ 1&0&1&0&1&0&1&0&1&1&1&1&1 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&1&0&1&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&1&0 \end{pmatrix}$$

$$
{}^{2}\mathbf{X}_{reach} = \begin{pmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
$$

The solutions of the control synthesis are illustrated in Figure 11.



a) b)

Fig. 11. The cooperation of two agents *A1*, *A2* in MAS: a) the case when *A1* solves (on the basis of the demand of *A2*) the problem to be solved by *A2*, however, *A2* is not able to do this; b) the case when *A1* refuses to help *A2*

6.2.1 The Problem of DEDS Controller and Controlled Object

The simple example introduced above describes the problem of two independent (autonomous) agents that communicate each other about the possibility of a mutual help in solving own problems. As to the authorities, the agents are equipollent. Namely, they can help each other; however, they also can refuse the help.

However, consider the controlled object (being e.g. a kind of DEDS or a system in general) to be the agent A_{syst} and its controller to be another agent A_{cont}. The role of the agent A_{syst} is to work with respect to prescribed rules. The role of the agent A_{cont} is to force A_{syst} to work with respect to the prescriptions. It can be said that a *master-slave* relation between the agents occurs, where A_{cont} is the master and A_{syst} is the slave. More precisely, the A_{syst} has the permanent problem to behave with respect to the rules, e.g. to pass from a given state towards a prescribed state when several alternatives are possible. The most suitable possibility cannot be chosen by the agent A_{syst} itself but on the contrary, it is determined in the process of the control synthesis. The control synthesis yields the sequence of control interferences which are obtruded on A_{syst}. The agent A_{cont} supervises whether A_{syst} respects the interferences or not. Such an agent A_{cont} is usually named supervisor. On the other hand, a more sophisticated agent A_{cont} can permanently be able to solve the problems concerning the A_{syst} behaviour and obtrudes the control interferences upon A_{syst}. Therefore, in case of control it cannot be spoken about a mutual help between A_{syst} and A_{cont}. The situation is displayed in Figure 12. While the process of the control synthesis is usually performed in the *off-line* way the mutual cooperation between the controller and the controlled system is performed *on-line*.

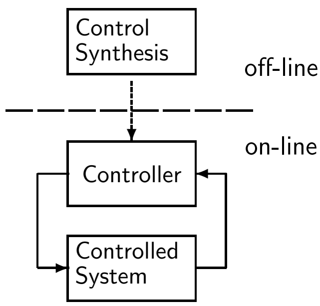


Fig. 12. Cooperation of the agents A_{syst} (the controlled system) and A_{cont} (the controller) in the process of controlling A_{syst} by A_{cont}

Nevertheless, the proposed PN-based approach to modelling, analysing and control synthesis is sufficiently general and it can be utilized in this case as well. Consider that $A_1 = A_{syst}$ and $A_2 = A_{cont}$ and utilize the above introduced mathematical model suitable for two agents with the same parameters, but with another initial state given as $\mathbf{x}_0 = ({}^{A_1}\mathbf{x}_0^T, {}^{A_2}\mathbf{x}_0^T)^T$ with ${}^{A_1}\mathbf{x}_0 = (1,1,0,1,0,0,0,0,0,0,0,0)^T$,

${}^{A_2}\mathbf{x}_0 = (1,0,0,0,0,0,0,0,0,0,0,0)^T$. Hence,

$$\mathbf{A}_{rt}(k) = \begin{pmatrix} 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 7 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 18 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 11 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 12 & 13 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 17 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

The corresponding reachability tree is on the left in Figure 13.

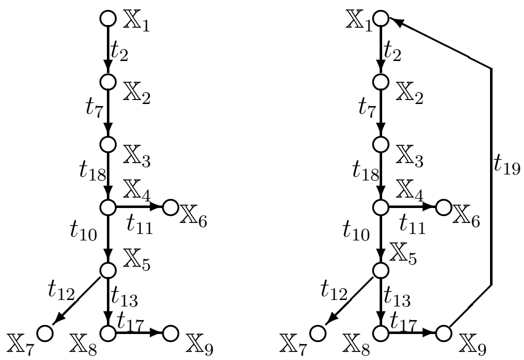


Fig. 13. The reachability graph of the cooperation A_{syst} and A_{cont}. The cooperation without the feedback – on the left side; the cooperation with the feedback – on the right side

Because the control problem needs not be solved in one cooperative cycle the feedback connection can be introduced. It can be simply realized by adding the transition t_{19} into the system interface – see Figure 14. It can be seen that its input place is p_5 (namely, its firing realizes the process of the control problem solving) and its output places are the places determining the initial state of the agents, i.e. the places p_1, p_2, p_4 and p_{13}. Of course, the PN incidence matrices have to respect the new transition t_{19} too. Consequently, the additional 19^{th} column of $\mathbf{F}$ is $(0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)^T$ and the 19th row of the matrix $\mathbf{G}$ is $(1,1,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0)$. In the reachability graph on the right in Figure 13 such a feedback manifests itself by connecting $\mathbb{X}_9$ with $\mathbb{X}_1$ by the arc marked by t_{19}. Consequently, the first column of $\mathbf{A}_{rt}(k)$ is $(0,0,0,0,0,0,0,0,19)^T$. The space of the feasible states is the same, i.e. the matrix $\mathbf{X}_{reach}$ does not change. It means that when the problem is not solved during one cooperative cycle of the agents their cooperation can continue in another cycle. The state trajectories are given in Figure 15.

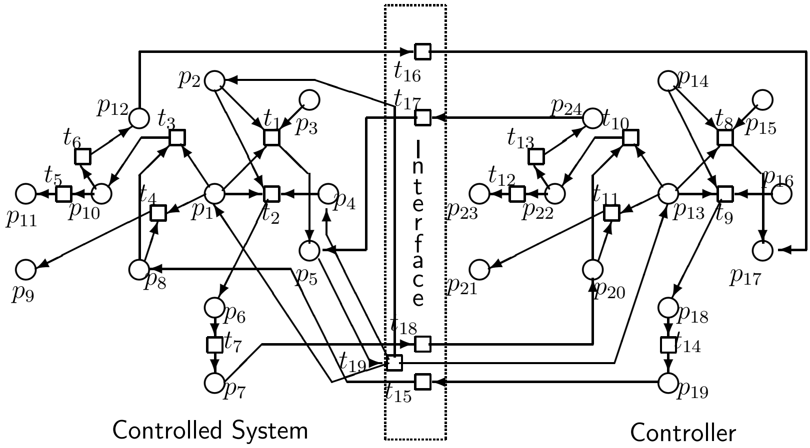


Fig. 14. The PN-based model of the feedback cooperation of the A_{syst} and A_{cont}

6.2.2 Cooperation of Three Agents

Analyse the cooperation of three agents A_1, A_2, A_3 mutually connected as given in Figure 16. The structural matrices describing the agents interactions can be written on the basis of this figure as follows

$$\mathbf{F}_{c_1} = \begin{pmatrix} 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0&0&1&0&0 \end{pmatrix}; \quad \mathbf{G}^T_{c_1} = \begin{pmatrix} 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \end{pmatrix}$$

$$\mathbf{F}_{c_2} = \begin{pmatrix} 0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&1&0&0&0&0&0&0&0 \end{pmatrix} ; \quad \mathbf{G}^T_{c_2} = \begin{pmatrix} 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \end{pmatrix}$$

$$\mathbf{F}_{c_3} = \begin{pmatrix} 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0&1&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&1&0&0&0 \end{pmatrix} ; \quad \mathbf{G}^T_{c_3} = \begin{pmatrix} 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0&0&0&0&1 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \end{pmatrix}$$

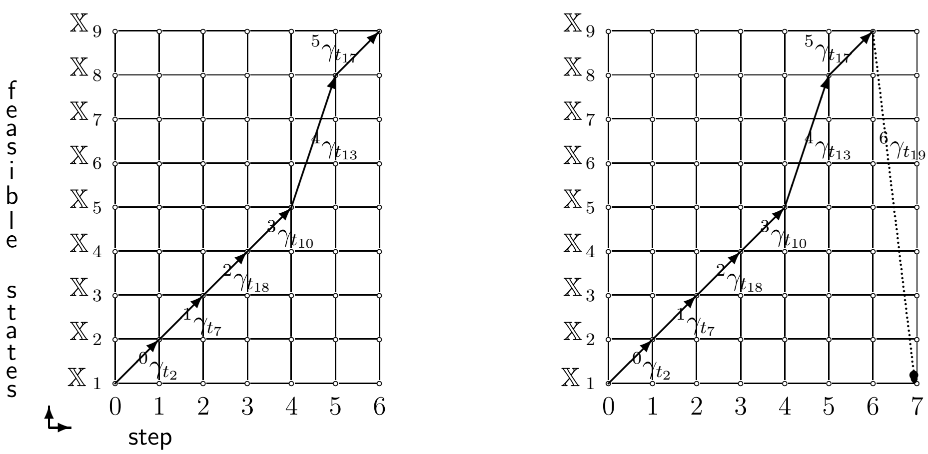


Fig. 15. The state trajectories at the cooperation of A_{syst} and A_{cont}: on the left – at the cooperation without feedback; on the right – at the feedback cooperation

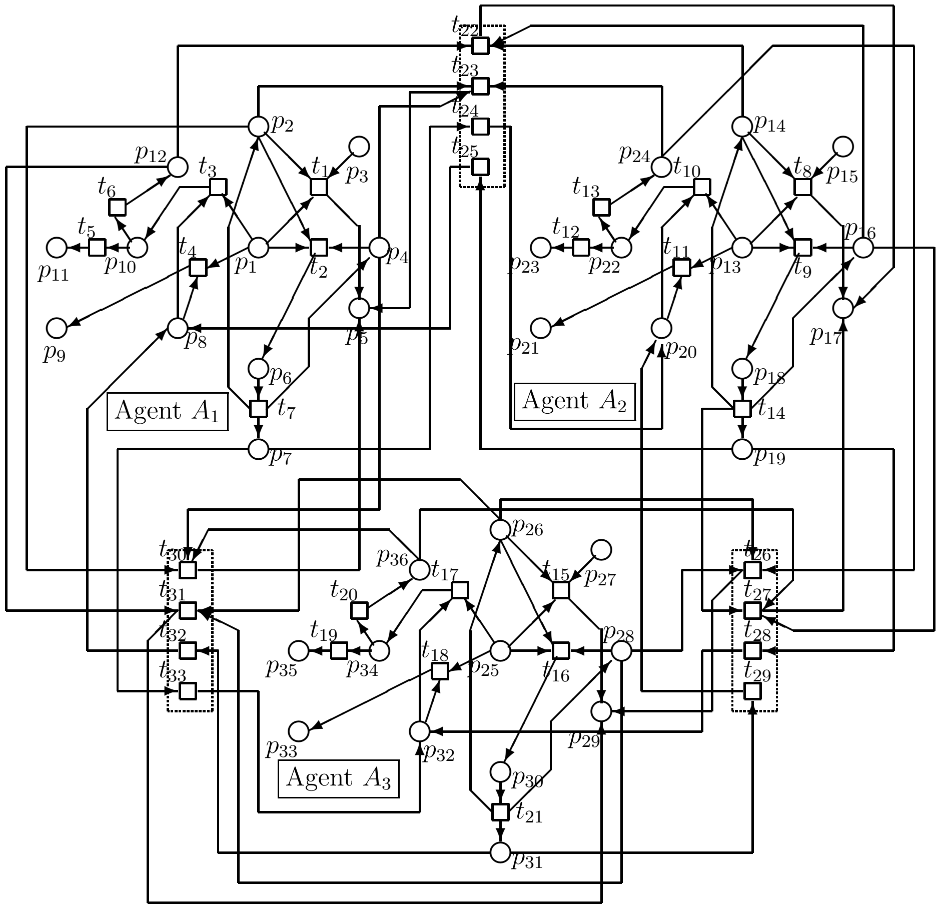


Fig. 16. The Petri net-based model of the three agents cooperation

Consider

$$\mathbf{x}_0 = ({}^{A_1}\mathbf{x}_0^T, {}^{A_2}\mathbf{x}_0^T, {}^{A_3}\mathbf{x}_0^T)^T \tag{26}$$

where ${}^{A_1}\mathbf{x}_0 = (1,1,1,0,0,0,0,0,0,0,0,0)^T$, ${}^{A_2}\mathbf{x}_0 = (1,1,1,0,0,0,0,0,0,0,0,0)^T$, ${}^{A_3}\mathbf{x}_0 = (1,1,0,1,0,0,0,0,0,0,0,0)^T$. It is the situation where A_3 is not able to solve its problem P_{A_3} and it has to ask either A_1 or A_2 for help. A_1, A_2 are able to solve their problems. There are $N = 40$ different feasible states in this system. The reachability graph is in Figure 17.

The segment of the reachability graph demarcated by the dotted line indicates the part of the graph which is active during the system dynamics development in case when the initial state is given by (26). Of course, the interaction with other

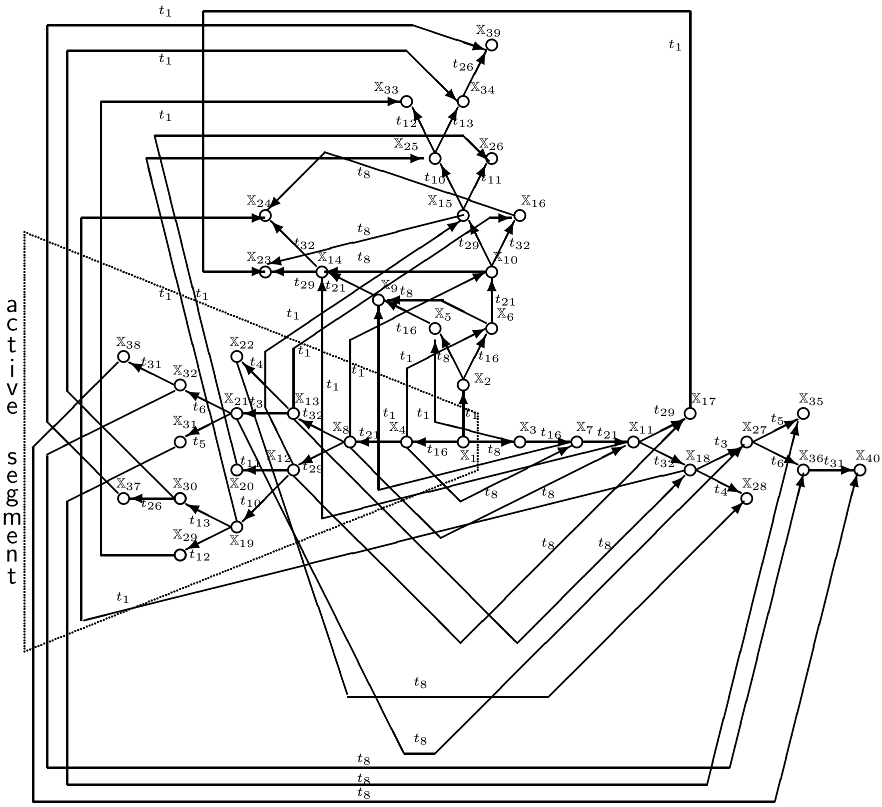


Fig. 17. The reachability graph of the three agents cooperation developed from the given initial state. The *active segment* indicates the part when A_1 or A_2 solves the problem P_{A_3}

parts of the graph should not be broken in order to preserve the model universality. Two solutions can be found for the given initial state. Namely, the solution depends on the fact which of the agents A_1, A_2 will be asked by the agent A_3 for help. The solutions are as follows:

$$\mathbb{X}_1 \xrightarrow{t_{16}} \mathbb{X}_4 \xrightarrow{t_{21}} \mathbb{X}_8 \xrightarrow{t_{32}} \mathbb{X}_{13} \xrightarrow{t_3} \mathbb{X}_{21} \xrightarrow{t_6} \mathbb{X}_{32} \xrightarrow{t_{31}} \mathbb{X}_{38}$$

$$\mathbb{X}_1 \xrightarrow{t_{16}} \mathbb{X}_4 \xrightarrow{t_{21}} \mathbb{X}_8 \xrightarrow{t_{29}} \mathbb{X}_{12} \xrightarrow{t_{10}} \mathbb{X}_{19} \xrightarrow{t_{13}} \mathbb{X}_{30} \xrightarrow{t_{26}} \mathbb{X}_{37}$$

where $\mathbb{X}_i = ({}^{A_1}\mathbb{X}_i^T, {}^{A_2}\mathbb{X}_i^T, {}^{A_3}\mathbb{X}_i^T)^T$, $i = 37, 38$, ${}^{A_1}\mathbb{X}_{37} = (1,1,1,0,0,0,0,0,0,0,0,0)^T$, ${}^{A_2}\mathbb{X}_{37} = (0,1,1,0,0,0,0,0,0,0,0,0)^T$, ${}^{A_3}\mathbb{X}_{37} = (0,0,0,0,1,0,0,0,0,0,0,0)^T$ and ${}^{A_1}\mathbb{X}_{38} = (0,1,1,0,0,0,0,0,0,0,0,0)^T$, ${}^{A_2}\mathbb{X}_{38} = (1,1,1,0,0,0,0,0,0,0,0,0)^T$, ${}^{A_3}\mathbb{X}_{38} = (0,0,0,0,1,0,0,0,0,0,0,0)^T$.

7 GENERALIZATION

The above described approach to modelling the agents in MAS is not rigid. It is suitable not only for modelling and analysing the negotiation process but also for the agent cooperation in general. Namely, the PN subnets modelling the structure of agents can be built arbitrarily (according to demands of the model creator). Even, the agents can have the mutually different structure. Also the interface among the agents in MAS needs not be modelled only by the additional PN transitions. On the contrary, in general the interface can be modelled by a PN subnet consisting of both the additional PN transitions and/or the additional PN places. The structure of such an interface can be created without restrictions at pleasure of the model creator. Let us introduce e.g. the simple model of the agent communication in order to illustrate the possibilities of the PN modelling the agents cooperation in MAS.

7.1 The Example of the Agent Communication

Consider the simple structure of the agents defined in [23]. The interpretation of the places in the PN model in Figure 18 is as follows: p_1 – A_1 does not want to communicate; p_2 – A_1 is available; p_3 – A_1 wants to communicate; p_4 – A_2 does not want to communicate; p_5 – A_2 is available; p_6 – A_2 wants to communicate; p_7 – communication; p_8 – availability of the communication channel(s) Ch (representing the interface). The PN transition t_9 fires the communication when A_1 is available and A_2 wants to communicate with A_1, t_{10} fires the communication when A_2 is available and A_1 wants to communicate with A_2, and t_{12} fires the communication when both A_1 and A_2 wants to communicate each other.

It is clear that the interface (the communication channel) has a form of the PN module (subnet) consisting of both the places and transitions. In spite of this fact, the above introduced approach to modelling, analysing and control synthesis can be utilized also in such a case, of course. It is sufficient to use the following model parameters and to choose an initial state. After doing this, there are no restrictions as to using the approach.

$$\mathbf{F} = \left(\begin{array}{cccc|cccc|cccc} 0&1&0&0&0&0&0&0&0&0&0&0 \\ 1&0&1&0&0&0&0&0&1&0&0&0 \\ 0&0&0&1&0&0&0&0&0&1&0&1 \\ \hline 0&0&0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&1&0&1&0&0&1&0&0 \\ 0&0&0&0&0&0&0&1&1&0&0&1 \\ \hline 0&0&0&0&0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0&1&1&0&1 \end{array} \right) = \begin{pmatrix} \mathbf{F}_{A_1} & \emptyset_{(3\times4)} & \mathbf{F}_{A_1\to A_2} \\ \emptyset_{(3\times4)} & \mathbf{F}_{A_2} & \mathbf{F}_{A_2\to A_1} \\ \emptyset_{(2\times4)} & \emptyset_{(2\times4)} & \mathbf{F}_{Ch_{1,2}} \end{pmatrix}$$

$$
\mathbf{G}^T = \left(\begin{array}{cccc|cccc|cccc}
1&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&0&0&0&0&0&1&0\\
0&0&1&0&0&0&0&0&0&0&0&0\\
\hline
0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&1&0&0&1&0\\
0&0&0&0&0&0&1&0&0&0&0&0\\
\hline
0&0&0&0&0&0&0&0&1&1&0&1\\
0&0&0&0&0&0&0&0&0&0&1&0
\end{array}\right) = \begin{pmatrix}
\mathbf{G}^T_{A_1} & \emptyset_{(3\times 4)} & \mathbf{G}^T_{A_1\to A_2}\\
\emptyset_{(3\times 4)} & \mathbf{G}^T_{A_2} & \mathbf{G}^T_{A_2\to A_1}\\
\emptyset_{(2\times 4)} & \emptyset_{(2\times 4)} & \mathbf{G}^T_{Ch_{1,2}}
\end{pmatrix}
$$

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

The number of agents cooperating in such a kind of MAS can be greater than two, of course.

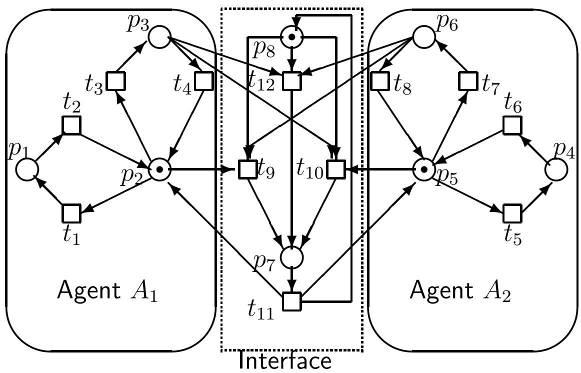


Fig. 18. The example of the PN-based model of the two agents communication in general

8 CONCLUSIONS

First of all, both the straightforward and backward causality of DEDS in general were described in this paper. The DEDS control synthesis method based on them was introduced. Then, the alternative PN-based approach to modelling, analysis and control of the agent interactions in MAS was presented. The approach yields the results in analytical terms (computed by means of Matlab) as well as graphically (obtained by means of the graphical tool – the PN simulator). Next, the PN-based model of the interaction process was created in analytical terms. The reachability tree and/or the reachability graph as well as the space of reachable states were generated on the basis of both the model parameters and the given initial state of the system. The coherence between the causality and the reachability graph was pointed out. Finally, the feasible state trajectories were found by means of the mutual intersection of both the SLRT representing the straightforward causal development of the system and BTRT representing the backward causal development of the system. The graphical tool GraSim developed in order to automate the process of finding the space of feasible trajectories at the control synthesis was mentioned. Finally, the possibility of the wider utilizing the proposed approach was pointed out.

Acknowledgements

The presented results were achieved in the project # 2/6102/26 partially supported by the Slovak grant agency VEGA. The author thanks VEGA for the partial support.

REFERENCES

[1] Bellman, R.—Kalaba, R.: Dynamic Programming and Modern Control Theory. Academic Press, New York-London, 1966.

[2] Brams, S. J.—Kilgour, D. M.: Fallback Bargaining. Group Decision and Negotiation, Vol. 10, 2001, pp. 287–316.

[3] Čapkovič, F.: Modelling and Control of Discrete Event Dynamic Systems. BRICS Report Series, RS-00-26, University of Aarhus, Denmark, 2000, 58p.

[4] Čapkovič, F.: Control Synthesis of a Class of DEDS. Kybernetes, Vol. 31, 2002, No. 9/10, pp. 1274–1281.

[5] Čapkovič, F.: The Generalised Method for Solving Problems of DEDS Control Synthesis. In: P.W.H. Chung, C. Hinde, M. Ali (Eds.): Developments in Applied Artificial Intelligence, Lecture Notes in Artificial Intelligence, Vol. 2718, 2003, pp. 702–711.

[6] Čapkovič, F.: An Application of the DEDS Control Synthesis Method. In: M. Bubak, G.D. van Albada, P.M.A. Sloot, J.J. Dongarra (Eds): Computational Science – ICCS 2004. Part III, Lecture Notes in Computer Science, Vol. 3038, 2004, pp. 528–536.

[7] Čapkovič, F.: DEDS Control Synthesis Problem Solving. Proc. 2nd IEEE Conf. on Intelligent Systems, Varna, Bulgaria. IEEE Press, Piscataway, NJ, USA, 2004, pp. 299–304.

[8] Čapkovič, F.—Čapkovič, P.: Petri Net-Based Automated Control Synthesis for a Class of DEDS. Proc. 2003 IEEE Conf. on Emerging Technologies and Factory Automation, Lisbon, Portugal, IEEE Press, Piscataway, NJ, USA, 2003, pp. 297–304.

[9] Čapkovič, F.: An Application of the DEDS Control Synthesis Method. Journal of Universal Computer Science, Vol. 11, 2005, No. 2, pp. 303–326.

[10] Diestel, R.: Graph Theory. Springer-Verlag, New York, 1997.

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

[12] Heljanko, K.: T79.4301. Parallel and Distributed Systems (4 ECTS). Lecture 5. Department of Computer Science, University of Helsinki, Finland, pp. 1–24. http://www.tcs.hut.fi/Studies/T-79.4301/2006SPR/lecture5.pdf.

[13] Hung, P. C. K.—Mao, J. Y.: Modeling E-Negotiation Activities with Petri Nets. Proc. 35th Hawaii Int. Conf. on System Sciences, Big Island, Hawaii, Vol. 1. IEEE Computer Society Press (CD/ROM), 2002, 10 p.

[14] Kirkland, S.—Olesky, D. D.—van den Driessche, P.: Digraphs with Large Exponent. The Electronics Journal of Linear Algebra, Vol. 7, 2000, pp. 30–40.

[15] Looney, C. G.: Fuzzy Petri Nets for Rule-Based Decision-Making. IEEE Trans. Syst. Man Cybern., Vol. 18, 1998, No. 1, pp. 178–183.

[16] Murata, T.: Petri Nets: Properties, Analysis and Applications. Proceedings of the IEEE, Vol. 77, April 1989, No. 4, pp. 541–588.

[17] Nowostawski, M.—Purvis, M.—Cranefield, S.: A Layered Approach for Modelling Agent Conversations. Proc. 2nd Int. Workshop on Infrastructure for Agents, MAS, and Scalable MAS, Montreal, Canada, 2001, pp. 163–170.

[18] Peterson, J. L.: Petri Nets Theory and the Modelling of Systems. Prentice-Hall Inc., Englewood Cliffs, New York, 1981.

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

[20] Preparata, F. P.—Yeh, R. T.: Introduction to Discrete Structures. Addison-Wesley Publ. Comp., Reading, USA, 1974.

[21] Purvis, M.—Cranefield, S.—Nowostawski, M.—Ward, R.—Carter, D.—Oliveira, M. A.: Agentcities Interaction Using the Opal Platform. Proc. Workshop on Agentcities: Research in Large-Scale Open Agents Environments, 1st Int. Joint Conf. on Autonomous Agents and Multi-Agent, Bologna, Italy, 2002, CD/ROM, 5 p.

[22] Rosen, K.: Discrete Mathematics and its Applications. McGraw-Hill Inc., 2002.

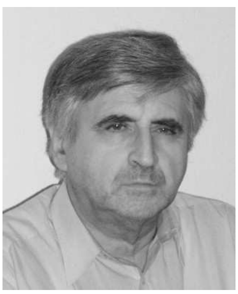
[23] Saint-Voirin, D.—Lang, C.—Zerhouni, N.: Distributed Cooperation Modeling for Maintenance Using Petri Nets and Multi-Agents Systems. In: Int. Symp. on Computational Intelligence in Robotics and Automation, CIRA '03, Kobe, Japan, July 16–20, 2003, pp. 366–371.

[24] Sedláček, J.: Introduction to Graph Theory. Academia, Prague, 1977 (in Czech).

[25] Thompson, L.: The Mind and Heart of the Negotiator. Prentice-Hall Inc., 1998.

[26] Wielandt, H.: Indecomposable Non-Negative Matrices. Math. Zeitschrift, Vol. 52, 1950, pp. 642–648 (in German).

[27] Yen, J.—Yin, J.—Ioerger, T. R.—Miller, M. S.—Xu, E.—Volz, R. A.: Cast: Collaborative Agents for Simulating Teamwork. Proc. 17th International Joint Conference on Artificial Intelligence – IJCAI '2001, Seatle, USA, 2001, pp. 1135–1142.



František Čapkovič received his master degree in 1972 from the Slovak Technical University, Bratislava, Slovakia. Since 1972 he has been working with the Slovak Academy of Sciences (SAS), Bratislava, namely in 1972–1991 at the Institute of Technical Cybernetics, in 1991–2001 at the Institute of Control Theory and Robotics and in 2001 till now at the Institute of Informatics. In 1980 he received the Ph. D. degree from SAS. Since 1998 he has been the Associate Professor. He works in the area of modelling, analysing and control of Discrete-Event Systems (DES).