Computing and Informatics, Vol. 29, 2010, 1045–1071

# AUTOMATIC CONTROL SYNTHESIS FOR AGENTS AND THEIR COOPERATION 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

Manuscript received 24 August 2009; revised 19 April 2010
Communicated by Ngoc Thanh Nguyen

**Abstract.** Automatic synthesis of control for a kind of DES (discrete-event systems) is discussed and an approach to it is proposed and presented. The approach consists in the proposal of the control synthesis procedure based on bipartite directed graphs yielding both the feasible control trajectories and the corresponding state ones. Soundness of the approach is tested on examples. Then, the usage of the approach is combined with the supervisor synthesis in order to complement it. Applicability of such approach is demonstrated by means of several illustrative examples of both the single agents and the agent cooperation in MAS.

**Keywords:** Agents, bipartite graphs, control synthesis, cooperation, discrete-event systems, Petri nets, supervisor synthesis

## 1 INTRODUCTION

Discrete-event systems (DES) are systems driven by occurring discrete events. Thus, their behaviour is discrete in nature. The behaviour of an agent as an entity can also be understood to be a kind of DES as well as the behaviour of several agents cooperating with each other in multi agent system (MAS), because their behaviour is also discrete in nature. In [1] causality of the DES behaviour was analysed as well as the Petri nets (PN)-based model, more precisely place/transition PN (P/T PN)-based model was introduced. Moreover, the method of control synthesis for

DES modelled by P/T PN was presented there. It was based on the reachability tree (RT) and the reachability graph (RG) of the P/T PN-based model. That procedure, performed in virtue of PN-based methods, yields the space of feasible state trajectories from a given initial state to a desired terminal state. This paper immediately goes on to solve the DES control synthesis problem in order to automate this process as soon as possible, even to achieve the automatic control synthesis. The earlier results concerning the bipartite directed graph (BDG)-based model presented in [2, 3] for the control synthesis of the special kind of P/T PN – for the state machines (SM) – will also be utilized in this paper; namely, their validity will be considerably extended. The main aims of this paper are both extending the BDG-based approach to be applicable for P/T PN with general structure (not only for SM) and usage of the proposed method not only by itself but also together with the methods of supervision [5, 6, 7, 8, 11, 9, 12, 13] in order to point out that such complementary usage can be mutually advantageous.

## 2 PRELIMINARIES

First of all it is necessary to briefly introduce the mathematical expression of the PN-based model of DES to be used in this paper. To observe the DES behaviour it is necessary to have not only states of the system available, but also the causal dependency among states and discrete events occurred in the system. The causality in the DES behaviour was explained in [1]. DES are frequently modelled and analysed by means of PN, especially by the P/T PN. For DES modelled by PN the control synthesis can be performed also in virtue of PN-based methods. In [1] the method of such kind was introduced. It yields the space of feasible state trajectories from a given initial state to a desired terminal state. Not to repeat all results presented in [1], this paper will be frequently referred to as well as the earlier results presented in [2, 3]. However, what is necessary to repeat is the mathematical expression of the PN-based model of DES in order to set a starting basis for evolving novelties. The model has the form

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

$$\mathbf{B} = \mathbf{G}^T - \mathbf{F} \tag{2}$$

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

It means that it is represented by the linear system circumscribed by the unequality, where $k$ is the discrete step of the dynamics development; $\mathbf{x}_k = (\sigma^k_{p_1}, \ldots, \sigma^k_{p_n})^T$ is the $n$-dimensional state vector of DEDS in the step $k$; $\sigma^k_{p_i} \in \{0, 1, \ldots, c_{p_i}\}$, $i = 1, \ldots, n$ express the states of the DEDS elementary subprocesses or operations by 0 (passivity) or by $0 < \sigma_{p_i} \leq c_{p_i}$ (activity); $c_{p_i}$ is the capacity of the DEDS subprocess $p_i$ as to its activities; $\mathbf{u}_k = (\gamma^k_{t_1}, \ldots, \gamma^k_{t_m})^T$ is the $m$-dimensional control vector of the system in the step $k$; its components $\gamma^k_{t_j} \in \{0, 1\}$, $j = 1, \ldots, m$ represent occurrence of the DEDS elementary discrete events (e.g. starting or ending the elementary subprocesses or their activities, failures, etc.) by 1 (presence of the corresponding discrete

event) or by 0 (absence of the event); $\mathbf{B}$, $\mathbf{F}$, $\mathbf{G}$ are structural matrices of constant elements; $\mathbf{F} = \{f_{ij}\}_{n\times m}, f_{ij} \in \{0, M_{f_{ij}}\}, i = 1,\ldots,n, j = 1,\ldots,m$ expresses the causal relations among the states of the DEDS (in the role of causes) and the discrete events occurring during the DEDS operation (in the role of consequences) by 0 (nonexistence of the corresponding relation) or by $M_{f_{ij}} > 0$ (existence and multiplicity of the relation); $\mathbf{G} = \{g_{ij}\}_{m\times n}, g_{ij} \in \{0, M_{g_{ij}}\}, i = 1,\ldots,m, j = 1,\ldots,n$ expresses very analogically the causal relations among the discrete events (causes) and the DEDS states (consequences); the structural matrix $\mathbf{B}$ is given by means of the arcs incidence matrices $\mathbf{F}$ and $\mathbf{G}$ according to (2); $(.)^T$ symbolizes the matrix or vector transposition.

## 3 WHAT IS THE DES CONTROL PROBLEM?

Considering the introduced system (1)–(3) to be the model of DES, from the system theory point of view the problem of control is the problem of finding a suitable sequence of control vectors $\{\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N-1}\}$ transferring the system form of a given initial state $\mathbf{x}_0$ to a prescribed terminal state $\mathbf{x}_N$. Usually, there are several possibilities of the further course in any step $k$ of the system dynamics development (i.e. in any state $\mathbf{x}_k$ of the system). In the PN terminology, there are (theoretically) several PN transitions which are enabled in the step $k$. The problem of control consists in the choice of the most suitable one. To illustrate the situation the simple situation is displayed in Figure 1.

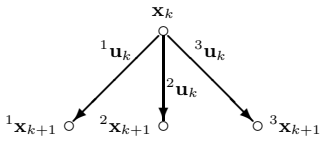



Fig. 1. Example of possible courses during the DES states development

A primitive way to control synthesis consists in finding (computing) possible routes of the dynamics development advancement in any step $k$ on the basis of simple logical consideration: if we are not able to directly compute which transitions are enabled in the step $k$, but we are able to compute only which transitions are disabled in this step, we can eliminate the disabled transitions. It means that logically

$$\mathbf{u}_k = neg\left(\mathbf{F}^T.neg(\mathbf{x}_k)\right) \tag{4}$$

where $neg(.)$ formally represents the operator of negation. Applying this operator for the vector $\mathbf{v} = (v_1, v_2, \ldots, v_n)^T$ of integers yields the vector $neg(\mathbf{v}) = \mathbf{w} =$

$(w_1, w_2, \ldots, w_n)^T$, where

$$w_i = \begin{cases} 1 & \text{if } v_i = 0 \\ 0 & \text{otherwise} \end{cases} \quad i = 1, \ldots, n. \tag{5}$$

Simultaneously, the condition (3) has to be satisfied for any ${}^i\mathbf{u}_k$.

However, because the development of the PN-based model is consecutive (successive), in case when several transitions are enabled in the step $k$ (i.e. when the vector $\mathbf{u}_k$ contains more than one non-zero entry), only one of them can be fired. It means that there exists only one route (course) how to proceed from the existing state to another one – see Figure 1. Because of the consecutiveness, any ${}^i\mathbf{u}_k$, $i = 1, 2, 3$ in Figure 1 has only one non-zero entry. It means that from global point of view in any step $k$ a branching of the system dynamics development (depending on the choice of the control interferences) can occur. Hence, if $\mathbf{u}_k$ contains $r$ non-zero entries (i.e. when $r$ transitions are enabled in the step $k$), theoretically there are mostly $r$ possibilities of the further development of the system dynamics. Thus, $\mathbf{u}_k$ has to be decomposed into $r$ control vectors ${}^i\mathbf{u}_k$, $i = 1, \ldots, r$ with single non-zero entry in such a way that $\sum_{i=1}^{r} {}^i\mathbf{u}_k = \mathbf{u}_k$. Moreover, all of the control vectors have to satisfy the condition (3). Therefore, the branching makes such a procedure too complicated for satisfying the usage. Moreover, we do not know which way is a part of trajectory leading to the prescribed terminal state (if any). Without additional backward ("backtracking") information we are not able to find the solution of the control problem by means of such a "blind" procedure. Therefore, it is better (and simpler in general) to compute the reachability tree (RT) corresponding to PN-based model. The RT yields information about the branching process in the whole. However, even the RT does not give us directly any solution of the control problem. Namely, we have to find the trajectory (or trajectories) between concrete pair of the states (namely, the initial state and the terminal one) and to "extract" the trajectory (or trajectories) from the global RT. In case of large scale RTs it is not a simple problem. Namely, in general the terminal state can be a multiple leaf of the tree. Consequently, this paper is motivated by the endeavour to find the automatic procedure for the DES control synthesis.

## 4 BDG-BASED AUTOMATIC SYNTHESIS FOR CONTROL OF DES

It is well known from PN theory – see e.g. [10] – that from the structural point of view PN are bipartite directed graphs (BDG) with two kinds of nodes (i.e. places and transitions) and two kinds of edges (the arcs directed from places to transitions and the arcs directed conversely), i.e.

$$\langle P, T, F, G \rangle \tag{6}$$

where $P$ is the set of PN places, $T$ is the set of PN transitions, $F$ is the set of the edges oriented from places to transitions and $G$ is the set of edges oriented

from transitions to places. The sets $F$, $G$ can be expressed by the PN incidence matrices $\mathbf{F}$, $\mathbf{G}$. Starting from results presented in [1–3] we are able to compute the functional adjacency matrix $\mathbf{A}_k$ of the RT as well as the set of feasible state vectors reachable from the given initial state $\mathbf{x}_0$ of the PN-based model. Such a space of feasible states is represented by the matrix $\mathbf{X}_{reach}$, where the vectors create its columns. The Matlab procedure, able to find $\mathbf{A}_k$ and $\mathbf{X}_{reach}$ on the basis of $\mathbf{F}$, $\mathbf{G}$, $\mathbf{x}_0$, is introduced in [3]. The method for DES control synthesis based on the intersection of both straight-lined and backward RT was presented in [1].

### 4.1 The Procedure for State Machines

A P/T Petri net is named to be the state machine (SM), if every transition has exactly one input place and one output place. On the contrary, the P/T Petri net where each place is allowed to have only single input transition and only single output transition is named the marked graph (MG). Hence, there can not be conflicts in MGs. It is clear that these classes of P/T PN are subclasses of P/T PN with general structure, where any transition can have more input and/or more output places as well as any place can have more input and/or more output transitions.

Consider P/T PN being SM. Let $S = \{P, T\}$ be the set of BDG nodes. Let $D \subseteq S \times S$ be the set of BDG edges. Thus, the occurrence of the edges can be expressed by the $((n+m) \times (n+m))$ BDG adjacency matrix

$$\mathbf{A}_{BDG} = \begin{pmatrix} \emptyset_{n\times n} & \mathbf{F} \\ \mathbf{G} & \emptyset_{m\times m} \end{pmatrix} \tag{7}$$

where $\emptyset_{i\times j}$ is in general the $(i \times j)$ zero matrix; $\mathbf{G}$ is the $(m \times n)$ incidence matrix expressing $T \times P$; $\mathbf{F}$ is the $(n \times m)$ incidence matrix representing $P \times T$. The matrices $\mathbf{F}$ and $\mathbf{G}$ are the same as the matrices in the PN-based model (1)–(3), of course. As we will see below, the transpose of the $\mathbf{A}_{BDG}$, i.e. the following matrix $\mathbf{D}$ will be very useful.

$$\mathbf{D} = \mathbf{A}_{BDG}^T = \begin{pmatrix} \emptyset_{n\times n} & \mathbf{G}^T \\ \mathbf{F}^T & \emptyset_{m\times m} \end{pmatrix} \tag{8}$$

In general, being in a state $\mathbf{x}_k$ the system can develop its dynamic behaviour either in the straight-lined direction or (fictively) also in the backward one. The former development is performed by means of the matrix $\mathbf{D}$ given in (8) used in the state Equation (9) while the latter one by means of the transpose $\mathbf{D}^T = \mathbf{A}_{BDG}$.

To synthesize the DES control from a given initial state $\mathbf{x}_0$ to a desired terminal state $\mathbf{x}_t = \mathbf{x}_N$ the straight-lined development of the system dynamic can be computed – see [1] – by means of the following state equation

$$\{\mathbf{s}_{k+1}\} \quad = \quad \mathbf{D}.\{\mathbf{s}_k\} \ , \ k = 0, 1, \ldots, 2N-1 \tag{9}$$

with $\mathbf{s}_k$ being the augmented $(n+m)$-dimensional vector defined as follows

$$\{\mathbf{s}_k\} = \begin{cases} \left(\{\mathbf{x}_{k/2}\}^T, \not{\mathbf{0}}_m^T\right)^T & \text{if } k = 0, 2, 4, \ldots, 2N-2 \\ \left(\not{\mathbf{0}}_n^T, \{\mathbf{u}_{(k-1)/2)}\}^T\right)^T & \text{if } k = 1, 3, 5, \ldots, 2N-1 \end{cases} \tag{10}$$

where $\not{\mathbf{0}}_j$ is in general the $j$-dimensional zero vector; $\mathbf{x}_{k/2} = \mathbf{G}^T.\mathbf{u}_{(k-2)/2}, k = 2, 4, \ldots, 2N-2$; $\mathbf{u}_{(k-1)/2} = \mathbf{F}^T.\mathbf{x}_{(k-1)/2}, k = 1, 3, 5, \ldots, 2N-1$. In general, $\{\mathbf{z}\}$ symbolizes an aggregate of vectors. To be sure that during the straight-lined development the prescribed terminal state will be met, the backward development of the system dynamics can be computed by means of the following state equation

$$\{\mathbf{s}_{k-1}\} = \mathbf{D}^T.\{\mathbf{s}_k\} = \mathbf{A}_{BDG}.\{\mathbf{s}_k\}\ ,\ k = K, K-1, \ldots, 1. \tag{11}$$

However, because of the special block form of both the matrix $\mathbf{D}$ and the vector $\mathbf{s}_k$ we can alternate step-by-step two procedures with dimensionality $n$, $m$, respectively. In such a way two matrices ${}^1\mathbf{X}$ and ${}^1\mathbf{U}$ with dimensionality $(n \times (N+1))$, $(m \times N)$ can be written as follows

$${}^1\mathbf{X} = \left(\mathbf{x}_0, {}^1\{\mathbf{x}_1\}, {}^1\{\mathbf{x}_2\}, \ldots, {}^1\{\mathbf{x}_N\}\right) \tag{12}$$

$${}^1\mathbf{U} = \left({}^1\{\mathbf{u}_0\}, {}^1\{\mathbf{u}_1\}, \ldots, {}^1\{\mathbf{u}_{N-1}\}\right). \tag{13}$$

The left upper index ${}^1(.)$ points out performing the straight-lined procedure. The backtracking (backward) procedure yields

$${}^2\mathbf{X} = \left({}^2\{\mathbf{x}_0\}, {}^2\{\mathbf{x}_1\}, {}^2\{\mathbf{x}_2\}, \ldots, \mathbf{x}_N\right) \tag{14}$$

$${}^2\mathbf{U} = \left({}^2\{\mathbf{u}_0\}, {}^2\{\mathbf{u}_1\}, \ldots, {}^2\{\mathbf{u}_{N-1}\}\right) \tag{15}$$

where ${}^2\mathbf{U}$ is $(m \times N)$ matrix and ${}^2\mathbf{X}$ is $(n \times (N+1))$ matrix. The left upper index ${}^2(.)$ points out performing the backtracking procedure.

The final phase of the control problem solving consists in the special intersection described above. In such a way we have both the system state trajectories and corresponding control strategies

$$\mathbf{X} = {}^1\mathbf{X} \cap {}^2\mathbf{X} \tag{16}$$

$$\mathbf{X} = (\mathbf{x}_0, \{\mathbf{x}_1\}, \ldots, \{\mathbf{x}_{N-1}\}, \mathbf{x}_N) \tag{17}$$

$$\mathbf{U} = {}^1\mathbf{U} \cap {}^2\mathbf{U} \tag{18}$$

$$\mathbf{U} = (\{\mathbf{u}_0\}, \{\mathbf{u}_1\}, \ldots, \{\mathbf{u}_{N-1}\}). \tag{19}$$

Using the zero blocks is eliminated this way as well. Although the described approach seems to be very hopeful as to automatic synthesis of the DES control, its usage is strongly limited. Namely, in such a form it is suitable for SM only. Using such an approach also for P/T PN with a general structure, which does not satisfy

this restriction, is impossible. In order to be applicable also for P/T PN with a general structure, the approach has to be modified. Namely, we have to work with the RT adjacency matrix appertaining to such PN.

### 4.1.1 Example 1

To illustrate the approach consider the simple P/T PN in the form of SM given in Figure 2.

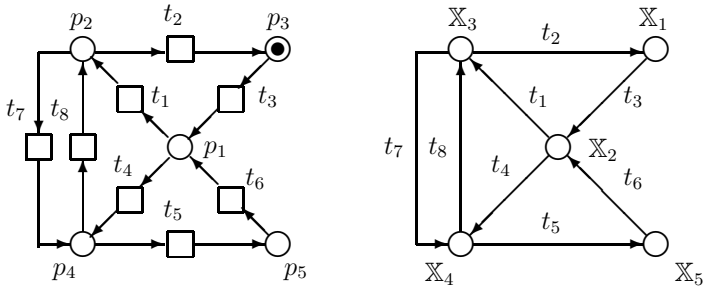



Fig. 2. The simple structure of the PN being the SM (left) and the corresponding RG (right)

The model parameters are

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

The RG is given in Figure 2 (right). It has the same adjacency matrix as RT. It is obtained by connecting RT leaves with the same names. Using the Matlab procedure presented in [3] we obtain the adjacency matrix of the RT $\mathbf{A}_{RT}$ and the set of feasible state vectors creating the RT nodes (represented by the matrix $\mathbf{X}_{reach} = (\mathbb{X}_1, \ldots, \mathbb{X}_5)^T$) in the form

$$\mathbf{A}_{RT} = \begin{pmatrix} 0&3&0&0&0 \\ 0&0&1&4&0 \\ 2&0&0&7&0 \\ 0&0&8&0&5 \\ 0&6&0&0&0 \end{pmatrix};\ \mathbf{X}_{reach} = \begin{pmatrix} 0&1&0&0&0 \\ 0&0&1&0&0 \\ 1&0&0&0&0 \\ 0&0&0&1&0 \\ 0&0&0&0&1 \end{pmatrix}$$

Understanding the SM to be the bipartite graph, its adjacency matrix has the form (7). Its transpose $\mathbf{D} = \mathbf{A}_{BDG}^T$ has the form (8) and can be utilized in the procedure of the control synthesis described above in mathematical terms. Hence, using the procedure [3] of the control synthesis (programmed in Matlab) we have

$$\mathbf{X} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}; \ \mathbf{U}^T = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Hence, it can be seen that there is only one feasible state trajectory, namely, $\mathbb{X}_1 \xrightarrow{t_3} \mathbb{X}_2 \xrightarrow{t_1} \mathbb{X}_3 \xrightarrow{t_2} \mathbb{X}_1$ which corresponds to the control trajectory $t_3 \xrightarrow{\mathbb{X}_2} t_1 \xrightarrow{\mathbb{X}_3} t_2$. In the original PN-based model (being SM) it means that the trajectories are (see the structure of $\mathbf{X}_{reach}$) as follows: $p_3 \xrightarrow{t_3} p_1 \xrightarrow{t_1} p_2 \xrightarrow{t_2} p_3$ and $t_3 \xrightarrow{p_1} t_1 \xrightarrow{p_2} t_2$.

### 4.2 Extending the Approach for P/T PN Having the General Structure

The above described BDG-based approach handles matrices $\mathbf{F}$, $\mathbf{G}$ as well as the PN transitions. However, the new approach proposed does not know either these matrices or the transitions, only the functional adjacency matrix $\mathbf{A}_k$. Thus, after enumerating the matrix $\mathbf{A}_{RT}$ we have to disassemble this matrix into the matrices $\mathbf{F}_{RT}$, and $\mathbf{G}_{RT}$. In addition, the original P/T PN transitions can occur more than once among the elements of $\mathbf{A}_{RT}$. Consequently, some confusions could occur during the computational process and decline it. To avoid these difficulties, it is necessary to rename the original P/T PN transitions in order to obtain fictive transitions that occur only once. The number of them is $T_r$ being the global number of the elements of $\mathbf{A}_{RT}$. The renaming is performed row-by-row so that the non-zero elements are replaced by integers – ordinal numbers starting from 1 and finishing at $T_r$. Thus, the auxiliary matrix $\mathbf{A}_{T_r}$ is obtained. The disassembling of the matrix $\mathbf{A}_{T_r}$ into the incidence matrices $\mathbf{F}_{RG}$ and $\mathbf{G}_{RG}$ is accomplished as follows: for $i = 1, \ldots, n_{RT}$, $j = 1, \ldots n_{RT}$ the elements of these matrices are as follows

$$\mathbf{T}_{tr}(\mathbf{A}_{RT}(i,j), \mathbf{A}_{T_r}(i,j)) = \begin{cases} 1 & \text{if } \mathbf{A}_{RT}(i,j) \neq 0 \ \& \ \mathbf{A}_{T_r}(i,j) \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

$$\mathbf{F}_{RG}(i, \mathbf{A}_{T_r}(i,j)) = \begin{cases} 1 & \text{if } \mathbf{A}_{RT}(i,j) \neq 0 \ \& \ \mathbf{A}_{T_r}(i,j) \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

$$\mathbf{G}_{RG}(\mathbf{A}_{T_r}(i,j), j) = \begin{cases} 1 & \text{if } \mathbf{A}_{RT}(i,j) \neq 0 \ \& \ \mathbf{A}_{T_r}(i,j) \neq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{22}$$

Here, $\mathbf{T}_{tr}$ is the transformation matrix between the original set of transitions and the fictive ones. Hence, $\mathbf{U} = \mathbf{T}_{tr}.\mathbf{U}^*$ where the matrix $\mathbf{U}^*$ yields the control strategies (13) computed by means of the set of the fictive transitions.

### 4.2.1 Example 2

Consider the simple PN with more general structure (i.e. different from the state machine) given in Figure 3 (left). Its RT and RG are, respectively, in Figure 3 (center and right). As we can see $n = 5$, $m = 3$, $\mathbf{x}_0 = (2, 0, 1, 0)^T$ and the structural (incidence) matrices as well as $\mathbf{A}_{RT}$ and $\mathbf{X}_{reach}$ are as follows:

$$\mathbf{F} = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 2 & 0 \end{pmatrix} \mathbf{G} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 2 \end{pmatrix} \mathbf{B} = \begin{pmatrix} -2 & 1 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & -2 & 2 \end{pmatrix}$$

$$\mathbf{A}_{RT} = \begin{pmatrix} 0 & 1 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 3 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 \end{pmatrix} \mathbf{X}_{reach} = \begin{pmatrix} 2 & 0 & 3 & 1 & 2 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 2 & 2 \\ 1 & 2 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 2 & 2 & 4 & 4 & 6 \end{pmatrix}$$

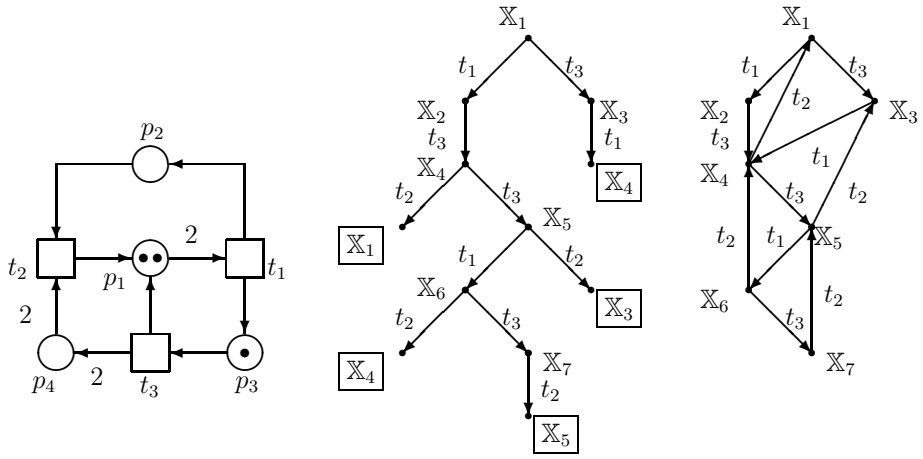



Fig. 3. The PN-based model model (left), the corresponding reachability tree (center) and the reachability graph (right) of the model

To use the BDG-based approach the renamed set of the transitions is needed (their number is $T_r = 11$ in this case). Thus, the transformation matrix $\mathbf{T}_{rTt}$ is enumerated by means of $\mathbf{A}_{RT}$ and the auxiliary matrix $\mathbf{A}_{T_r}$ constructed on the basis of $\mathbf{A}_{RT}$. Hence, the matrices $\mathbf{F}_{RG}$, $\mathbf{G}_{RG}$ are computed. All of the matrices in

question are as follows:

$$\mathbf{A} = \begin{pmatrix} 0&1&1&0&0&0&0 \\ 0&0&0&1&0&0&0 \\ 0&0&0&1&0&0&0 \\ 1&0&0&0&1&0&0 \\ 0&0&1&0&0&1&0 \\ 0&0&0&1&0&0&1 \\ 0&0&0&0&1&0&0 \end{pmatrix};\ \mathbf{A}_{T_r} = \begin{pmatrix} 0&1&2&0&0&0&0 \\ 0&0&0&3&0&0&0 \\ 0&0&0&4&0&0&0 \\ 5&0&0&0&6&0&0 \\ 0&0&7&0&0&8&0 \\ 0&0&0&9&0&0&10 \\ 0&0&0&0&11&0&0 \end{pmatrix};\ \mathbf{T}^T_{rTt} = \begin{pmatrix} 1&0&0 \\ 0&0&1 \\ 0&0&1 \\ 1&0&0 \\ 0&1&0 \\ 0&0&1 \\ 0&1&0 \\ 1&0&0 \\ 0&1&0 \\ 0&0&1 \\ 0&1&0 \end{pmatrix}$$

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

In case when $\mathbf{x}_t = \mathbf{x}_6 = (1,2,0,6)^T$ is chosen to be the terminal state, the ODG-based methods of the DEDS control synthesis yields the solution (23) of the state trajectories. Its graphical expression is in Figure 4 (left). When the BDG-based method is used the solution of the state trajectories is the same like that in the ODG-based method presented in [1]. The solution of the control trajectories is given by (25) and graphically expressed in Figure 4 (right).

$$^1\mathbf{X} = \begin{pmatrix} 1&0&0&2&0&0 \\ 0&1&0&0&2&0 \\ 0&1&0&0&4&0 \\ 0&0&2&0&0&8 \\ 0&0&0&2&0&0 \\ 0&0&0&0&2&0 \\ 0&0&0&0&0&2 \end{pmatrix}\ ^2\mathbf{X} = \begin{pmatrix} 2&0&0&0&0&0 \\ 0&1&0&0&0&0 \\ 0&1&0&0&0&0 \\ 0&0&1&0&0&0 \\ 3&0&0&1&0&0 \\ 0&2&0&0&1&0 \\ 0&0&1&0&0&1 \end{pmatrix}\ \mathbf{X} = \begin{pmatrix} 1&0&0&0&0&0 \\ 0&1&0&0&0&0 \\ 0&1&0&0&0&0 \\ 0&0&1&0&0&0 \\ 0&0&0&1&0&0 \\ 0&0&0&0&1&0 \\ 0&0&0&0&0&1 \end{pmatrix} \quad (23)$$

$$
{}^1\mathbf{U}^* = \begin{pmatrix} 1&0&0&2&0\\ 1&0&0&2&0\\ 0&1&0&0&2\\ 0&1&0&0&4\\ 0&0&2&0&0\\ 0&0&2&0&0\\ 0&0&0&2&0\\ 0&0&0&2&0\\ 0&0&0&0&2\\ 0&0&0&0&2\\ 0&0&0&0&0 \end{pmatrix}
{}^2\mathbf{U}^* = \begin{pmatrix} 1&0&0&0&0\\ 1&0&0&0&0\\ 0&1&0&0&0\\ 0&1&0&0&0\\ 0&0&0&0&0\\ 0&0&1&0&0\\ 1&0&0&0&0\\ 2&0&0&1&0\\ 0&1&0&0&0\\ 0&1&0&0&1\\ 0&0&1&0&0 \end{pmatrix}
\mathbf{U}^* = \begin{pmatrix} 1&0&0&0&0\\ 1&0&0&0&0\\ 0&1&0&0&0\\ 0&1&0&0&0\\ 0&0&0&0&0\\ 0&0&1&0&0\\ 0&0&0&0&0\\ 0&0&0&1&0\\ 0&0&0&0&0\\ 0&0&0&0&1\\ 0&0&0&0&0 \end{pmatrix} \quad (24)
$$

$$
\mathbf{U} = \mathbf{T}_{rTt}.\mathbf{U}^* = \begin{pmatrix} 1&1&0&1&0\\ 0&0&0&0&0\\ 1&1&1&0&1 \end{pmatrix} \quad (25)
$$

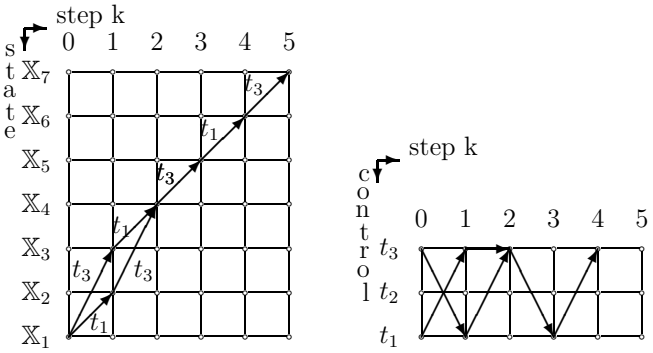



Fig. 4. The solution of the state trajectories (left) and the control trajectories (right)

It means that there are two feasible state trajectories, namely, $\mathbb{X}_1 \xrightarrow{t_1} \mathbb{X}_2 \xrightarrow{t_3} \mathbb{X}_4 \xrightarrow{t_3} \mathbb{X}_5 \xrightarrow{t_1} \mathbb{X}_6 \xrightarrow{t_3} \mathbb{X}_7$ and $\mathbb{X}_1 \xrightarrow{t_3} \mathbb{X}_3 \xrightarrow{t_1} \mathbb{X}_4 \xrightarrow{t_3} \mathbb{X}_5 \xrightarrow{t_1} \mathbb{X}_6 \xrightarrow{t_3} \mathbb{X}_7$ as well as two corresponding control trajectories $t_1 \xrightarrow{\mathbb{X}_2} t_3 \xrightarrow{\mathbb{X}_4} t_3 \xrightarrow{\mathbb{X}_5} t_1 \xrightarrow{\mathbb{X}_6} t_3$ and $t_3 \xrightarrow{\mathbb{X}_3} t_1 \xrightarrow{\mathbb{X}_4} t_3 \xrightarrow{\mathbb{X}_5} t_1 \xrightarrow{\mathbb{X}_6} t_3$.

### 4.3 Automatic Procedure of the DES Control Synthesis

Let us outline the principle of the automatic approach to DES control synthesis.

1. Given are the P/T PN-based model of DES to be controlled, the initial state $\mathbf{x}_0$ and the desired terminal state $\mathbf{x}_t$. The model is specified by the incidence matrices $\mathbf{F}, \mathbf{G}$.

2. If all of the **G** rows and all of the **F** columns contain only single non-zero element, the P/T PN is of a special kind – it is the state machine. In such a case we can proceed in two different ways:

   (a) To use $\mathbf{F}, \mathbf{G}, \mathbf{x}_0$ in order to compute the RT parameters. Afterwards, the RG-based model can be utilized to compute the straight-lined RT (SLRT) from $\mathbf{x}_0$ towards $\mathbf{x}_t$ as well as to compute the backward RT (BTRT) from $\mathbf{x}_t$ towards $\mathbf{x}_0$ (however, its edges are oriented towards $\mathbf{x}_t$). The mutual intersection (of a special kind) both of the trees yields the space of feasible state trajectories. This approach was sufficiently described in [1].
   (b) To understand SM to be the bipartite directed graph. Then, the matrix $\mathbf{D} = \mathbf{A}_{BDG}^T$ can be utilized in the procedure computing not only the feasible state trajectories but (simultaneously) also the control trajectories. Namely, the state trajectories are the DES responses to the corresponding control trajectories. In such a way both the feasible state trajectories and the control trajectories (strategies) are found automatically. This approach was comprehensively described in [2].

3. If any **G** row and/or any **F** column contain more than one non-zero elements, P/T PN is not an SM but it has a general structure. Here, two different ways are also possible:

   (a) To compute the RT parameters on the basis of $\mathbf{F}, \mathbf{G}, \mathbf{x}_0$. Consequently, the RG-based model [1] can be automatically computed and utilized in order to find the space of feasible trajectories, analogically to the procedure used for SM.
   (b) To use the BDG-based approach. However, the BDG-based approach suitable for SM cannot be directly used here in automatic finding of the feasible state trajectories and of the control ones. In this case, it is necessary to unfold the set of the PN-transitions into the set of fictive (mutually different) transitions in order to obtain unambiguous decomposition of the RG adjacency matrix into the matrices $\mathbf{F}_{RG}, \mathbf{G}_{RG}$. This activity is performed automatically too. Afterwards (when the fictive artificial BDG is obtained in such a way), the BDG based approach can be utilized in order to automatically find the feasible state trajectories and the control ones. Because the control trajectories were computed by means of the fictive transitions, return into the original space of the real transitions is necessary. It is very simple and can be performed automatically too.

### 4.4 The Concluding Remark about Applicability

The proposed method can be used autonomously either for a single agent or for a group of cooperating agents in MAS. However, it is gratifying that it can be successfully used also in combination with other control synthesis methods. In the

following sections we will use it in conjunction with supervisory control, where we will point out how both approaches complement each other. Namely, the supervisor synthesis methods produce the results (the structure of the supervisor) that ensure only such properties and behaviour of the supervised object that correspond to the conditions for which the supervisor was synthesized. Although on the one hand the supervision indeed represents the method able to ensure a prescribed behaviour within the bounds of possibility, on the other hand it does not yield either the state trajectories or the control trajectories (which yield the sequences of all the control interferences, not only those cohering with the supervision but in general) of the whole complex – i.e. the supervised system together with the supervisor. Thus, the control synthesis method proposed and presented in this paper complements the results and makes possible complex analyzing the behaviour of the supervised system and its control within full range. Namely, for our approach the supervised system represents the system in general, not any specific one, even if it is supervised. On the other hand, the supervision helps our approach in such a way that it yields (sometimes strongly) limited extent of possibilities how the system to be controlled by our method can behave. In an ideal case, only one sequence of control interferences could occur, i.e. there could exist only one control trajectory and consequently only one state trajectory. In such a case we could say that the supervision comprehensively covers the control synthesis and it remains the automatic control instead of supervision. However, in general, the ideal cases are usually only wishful thinking.

## 5 SUPERVISORY CONTROL OF DES

The agents and group of cooperating agents are typical entities of distributed systems. The agents need not have only software fundamentals. They can also have a material substance – see e.g. [4] – of different kinds (e.g. robots and other intelligent technical devices like automatically guided vehicles, etc.). Even, there exist also social agents of human origin. Varied forms of the agents cooperation call for coordination activities or at least for a form of supervising. Namely, the elementary agents working in real environment are not able to control either themselves or each other. Many times a need of the supervising agent is very important. The possibility of utilizing the results of DES control theory (see e.g. [11]) and PN applications [5, 12] on this way is pointed out in this paper. Especially, the knowledge about the supervision based on the P-invariants [6, 13, 7, 8] will be applied.

### 5.1 P/T PN Invariants and their Use

Two kinds of invariants in PN theory are defined, namely the P-invariants concerning the PN places and the T-invariants having the relationship with PN transitions. Their formal definitions are very simple – see e.g. [10]. The P-invarians are the $n$-dimensional vectors $\mathbf{v}$ of integers satisfying the homogenous system of linear equations

in the form $\mathbf{B}^T.\mathbf{v} = \mathbf{0}$. The T-invariants are the $m$-dimensional vectors $\mathbf{w}$ of integers satisfying the homogenous system of linear equations in the form $\mathbf{B}.\mathbf{w} = \mathbf{0}$. However, both kinds of invariants are very important in PN analysis and testing their properties [10]. Even, they are very important also in the PN control synthesis. The P-invariant $\mathbf{v} \geq \mathbf{0}$ is called nonnegative P-invariant. Analogically, the T-invariant $\mathbf{w} \geq \mathbf{0}$ is called nonnegative T-invariant. In [10] the set of all possible minimal support P-invariants and the set of all possible minimal support T-invariants are also defined. These sets of invariants are called generators of invariants, because any invariant can be written as the linear combination of minimal support invariants.

From the definition of T-invariants a very interesting PN property results. Namely, T-invariant is a sequence of transitions which (however only in case when the sequence is firable) transfers the system into its initial state.

P-invariants are vectors, $\mathbf{v}$, with the property that multiplication of these vectors with any state vector $\mathbf{x}$ reachable from a given initial state vector $\mathbf{x}_0$ yields the same result (the relation of the state conservation)

$$\mathbf{v}^T.\mathbf{x} = \mathbf{v}^T.\mathbf{x}_0.$$

Taking into account the consecutive states (obtained by firing of only one transition), it results that

$$\mathbf{v}^T.col_t(\mathbf{B}) = 0$$

for each transition $t$. Here, $col_t(\mathbf{B})$ is the column of $\mathbf{B}$ corresponding to the transition $t$. That means that, algebraically, these vectors are solutions of the equation

$$\mathbf{B}^T.\mathbf{v} = \emptyset \quad \text{or} \quad \mathbf{v}^T.\mathbf{B} = \emptyset^T$$

where $\mathbf{v}$ is $n$-dimensional vector ($n$ expresses the number of PN places) and $\emptyset$ is $m$-dimensional zero vector. This equation is (as already mentioned) usually presented as the definition of the P-invariant $\mathbf{v}$ of PN.

On the other hand, the P-invariants can be utilized in supervisor synthesis [9, 12]. Starting from the definition of the P-invariant $\mathbf{v}$, in case of several invariants (e.g. $n_x$) the set of the P-invariants is created by the columns of the $(n \times n_x)$-dimensional matrix $\mathbf{V}$ ($n_x$ expresses the number of invariants) being the solution of the equation

$$\mathbf{V}^T.\mathbf{B} = \emptyset. \tag{26}$$

It is this equation which represents the basis for the supervisor synthesis method. Namely, some additional PN places (so called slacks) can be added to the PN in question. These slacks will create the places of the PN-based model of the supervisor. In PN with the slacks (i.e. the original PN together with the extension by means of the slacks) we have to use the following structure of the previous equation – the structure with the augmented matrices

$$[\mathbf{L}, \mathbf{I}_s].\begin{bmatrix} \mathbf{B} \\ \mathbf{B}_s \end{bmatrix} = \emptyset$$

where $\mathbf{I}_s$ is $(n_s \times n_s)$-dimensional identity matrix with $n_s$ being the number of slacks, $(n_s \times n)$-dimensional matrix $\mathbf{L}$ represents (in a suitable form) the conditions imposed on marking of the original PN and the $(n_s \times m)$-dimensional matrix $\mathbf{B}_s$ yields (after its finding by computing) the structure of the PN-based model of the supervisor. Consequently, in general

$$\mathbf{L}.\mathbf{B} + \mathbf{B}_s = \emptyset;\ \mathbf{B}_s = -\mathbf{L}.\mathbf{B} \text{ and } \mathbf{B}_s = \mathbf{G}_s^T - \mathbf{F}_s$$

where the actual structure of the matrix $\mathbf{L}$ has to be respected. The augmented state vector (the state vector of the supervised system – i.e. the state vector of the original PN connected together with the supervisor) and the augmented matrices are as follows

$$\mathbf{x}_a = \begin{bmatrix} \mathbf{x} \\ \mathbf{x}_s \end{bmatrix};\ \mathbf{F}_a = \begin{pmatrix} \mathbf{F} \\ \mathbf{F}_s \end{pmatrix};\ \mathbf{G}_a^T = \begin{pmatrix} \mathbf{G}^T \\ \mathbf{G}_s^T \end{pmatrix}$$

where the submatrices $\mathbf{F}_s$ and $\mathbf{G}_s^T$ correspond to the interconnections of the incorporated slacks with the actual PN structure. Because of the prescribed conditions we have

$$[\mathbf{L}|\mathbf{I}_s].\begin{bmatrix} \mathbf{x}_0 \\ {}^s\mathbf{x}_0 \end{bmatrix} = \mathbf{b}$$

where $\mathbf{b}$ is the vector of the corresponding dimensionality (i.e. $n_s$) with integer entries representing the limits for number of tokens. Many times $\mathbf{b} = \mathbf{1}$. Here $\mathbf{1}$ is the vector with all its entries equal to 1. Because

$$\mathbf{L}.\mathbf{x}_0 + {}^s\mathbf{x}_0 = \mathbf{b}$$

the initial state vector of the supervisor is given as

$${}^s\mathbf{x}_0 = \mathbf{b} - \mathbf{L}.\mathbf{x}_0.$$

### 5.1.1 Example 3

Consider the PN given in Figure 5 which adapts a PN model from [9] of an unreliable machine [5].

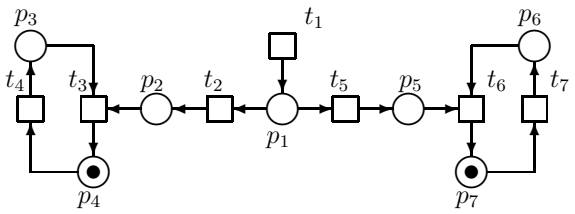



Fig. 5. The original structure of the PN model of the system

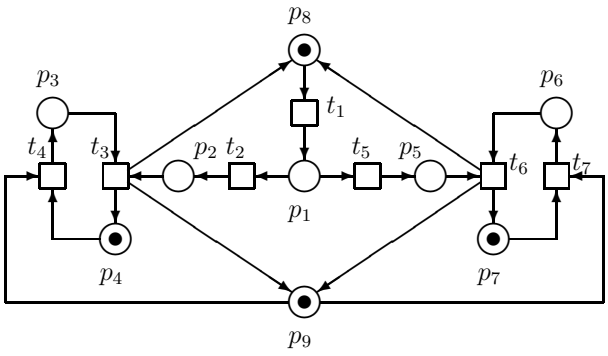



Fig. 6. The structure of the system with the supervisor

Although the PN seems to be very simple, the RT is very complicated (although it contains only 19 nodes, there are many edges and cycles) because of the transition $t_1$. Namely, $t_1$ is permanently enabled. Consequently, it can consecutively generate (by its firing) further and further input tokens for the place $p_1$. Even this process is infinite. There exist many "auto-cycles" (self-loops) in the RT. A cycle is a path with the same node at the beginning and the end. By the "auto-cycle" we mean the cycle where an edge emerges from the node and simultaneously enters the same node. There are even several "auto-cycles" in some nodes of the RG, each created by the edge labelled by a different PN transition. From the control theory point of view, such a system behaviour is unacceptable (such a system can be understood to be e.g. unstable). In order to forbid such a situation, a supervisor (see e.g. [11, 6, 9, 12]) can be synthesized. The parameters of the P/T PN based model as well as the initial state vector are as follows:

$$\mathbf{F} = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} ; \mathbf{G} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} ; \mathbf{x}_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Consider the conditions for the supervisor synthesis as follows:

$$\sigma_{p_1} + \sigma_{p_2} + \sigma_{p_5} \leq 1 \quad (27)$$

$$\sigma_{p_3} + \sigma_{p_6} \leq 1 \quad (28)$$

i.e. only one of the places $p_1$, $p_2$, $p_5$ can possess the token at the same time. The same prescription is valid for the possession of the token by the places $p_3$ and $p_5$. In the controlled system the RT is a lot less than that of the uncontrolled one and the

stability of the system is better. The supervisor is synthesized as follows:

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

$$\mathbf{G}_s^T = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix};\ {}^s\mathbf{x}_0 = \mathbf{1} - \mathbf{L}.\mathbf{x}_0 = (1,1)^T.$$

The system controlled by the supervisor has the following parameters:

$$\mathbf{F}_a = \left(\begin{array}{ccccccc} 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{array}\right);\ \mathbf{G}_a = \left(\begin{array}{ccccccc|cc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array}\right);\ \mathbf{x}_0 a = \left(\begin{array}{c} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ \hline 1 \\ 1 \end{array}\right)$$

The structure of the supervised system is given in Figure 6. Although the supervisor assures the prescribed conditions, it does not assure anything more, even it gives no information about the supervised system behaviour.

### 5.2 Behaviour of the System Controlled by the Supervisor

Let us illustrate that also in such a case our approach to the automatic synthesis (introduced above) can be utilized. Namely, not even in the supervised system the control problem of the system is fully solved. Although the supervisor keeps the system within the prescribed conditions, it is not able to choose either the most suitable transition (from enabled ones in the step $k$ of the dynamic development) to be fired in an existing state $\mathbf{x}_a$ or how to reach a prescribed terminal state $\mathbf{x}_t a$ of the supervised system from the given initial state $\mathbf{x}_0 a$. Our approach yields such a possibility and offers both the set of feasible state trajectories and the set of control ones.

#### 5.2.1 Example 4

Let us continue in the previous example. The parameters of the corresponding RT (the adjacency matrix and the set of nodes being the feasible states) are as follows:

$$\mathbf{A}_{RT}a = \begin{pmatrix} 0 & 1 & 4 & 7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 4 & 5 & 7 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 5 \\ 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

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

The corresponding RG is shown in Figure 7.

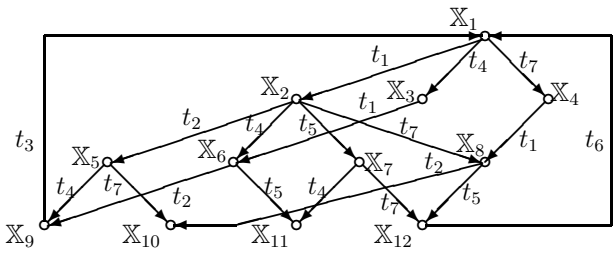



Fig. 7. The reachability graph of the system controlled by the supervisor

Set $\mathbf{D}_a = \mathbf{A}_{RT}^T a$. Now, we can use the method for the automatic synthesis in this case as well. It is useful at least when analysing the system behaviour. In case of a working cycle – when the $\mathbf{x}_t a = \mathbf{x}_0 a$, using the Matlab procedure we can obtain

$$\mathbf{X} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}; \ \mathbf{U}^* = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{29}$$

Because of the transformation matrix

$$\mathbf{T}_{rTt} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

we finally have

$$\mathbf{U} = \mathbf{T}_{rTt}.\mathbf{U}^* = \begin{pmatrix} 1 & 2 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

Here, one can be startled that there is the integer 2 in the matrix $\mathbf{U}$. However, it does not mean that a control vector $\mathbf{u}_k$ in a step $k$ contains such an entry greater than 1. It only means (because $\mathbf{U}$ represents control trajectories) that in this point

two different control trajectories "touch" each other. The first of them is trajectory $t_4, t_1, t_2, t_3$ and the second one is the trajectory $t_7, t_1, t_5, t_6$ – it can be checked in Figure 7. Just this fact is expressed by means of the integer 2. The control trajectories are given in Figure 8 (left), where the enabled transitions from the set $T = \{t_1, \dots t_7\}$ are displayed on the vertical axis.

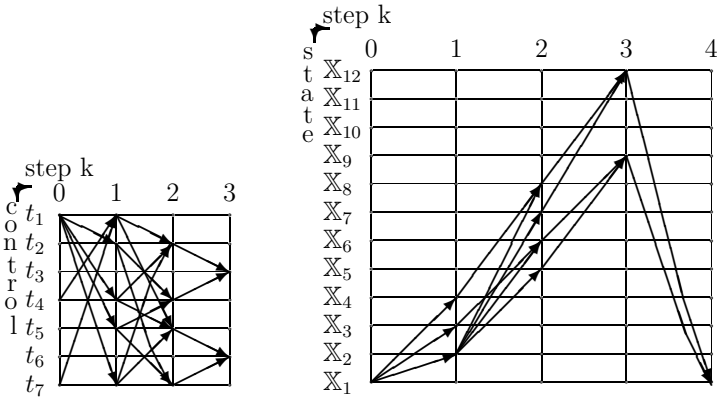



Fig. 8. The solution of the control trajectories (left) and the solution of the corresponding state trajectories (right)

The responses of the system to these control trajectories are the state trajectories given in Figure 8 (right).

## 6 SUPERVISING THE COOPERATION OF DES SUBSYSTEMS AND AGENTS IN MAS

Now, let us try to utilize the combination of both the supervisor synthesis and our approach to automatic control synthesis, which was illustrated in the previous two examples, for the cooperation of DES subsystems and agents in MAS. In order to bring DES subsystems to a better cooperation or to forbid agents in MAS (e.g. autonomous robots, automatically guided vehicles (AGV), etc.) to be selfish at usage of common sources (like energy, raw materials, yards, roads, etc.) the supervisor can be designed by the above-mentioned method. There are many possible strategies as to the supervisor synthesis. Note two of them.

### 6.1 Mutex Form of Supervising

The cooperation introduced here has the Mutex (mutual exclusion) form. It means that the aims of a whole are preferred over the aims of its elementary agents. A rule for behaviour of agents is forced in order to avoid any selfish behaviour of agents which weakens the whole.

### 6.1.1 Example 5

In Figure 9 (left) the PN model of cooperation of two general subsystems/agents is introduced.

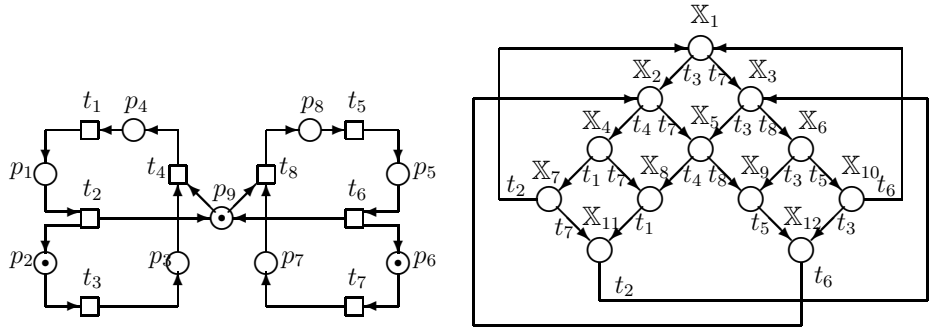



Fig. 9. The PN model of the system (left) and the corresponding causal model represented by means of the full RG (right)

The model parameters and the initial state are given as follows:

$$\mathbf{F} = \begin{pmatrix} 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 1&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&1&0&0&0&1 \end{pmatrix}; \ \mathbf{G}^T = \begin{pmatrix} 1&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \\ 0&1&0&0&0&1&0&0 \end{pmatrix}; \ \mathbf{x}_0 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

The RG of the system is given in Figure 9 (right). It can be computed by Matlab and its parameters are as follows:

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

$$\mathbf{A} = \begin{pmatrix} 0 & 3 & 7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 8 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 7 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 5 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 \\ 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

The single agent looks like that in Figure 10. It can execute its autonomouos activities without any restrictions from without. Its parameters are

$${}^1\mathbf{F} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}; \; {}^1\mathbf{G}_1^T = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}; \; {}^1\mathbf{x}_0 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

The same is valid for another agent with same structure (but its elementary activities can be different from the former agent). The structural parameters of both the non-cooperating agents are as follows:

$$\mathbf{F} = \begin{pmatrix} {}^1\mathbf{F} & \emptyset \\ \emptyset & {}^2\mathbf{F} \end{pmatrix}; \; \mathbf{G}^T = \begin{pmatrix} {}^1\mathbf{G}^T & \emptyset \\ \emptyset & {}^2\mathbf{G}^T \end{pmatrix}; \; \mathbf{B} = \begin{pmatrix} {}^1\mathbf{B} & \emptyset \\ \emptyset & {}^2\mathbf{B} \end{pmatrix}$$

$$\mathbf{x}_0 = \begin{pmatrix} {}^1\mathbf{x}_0 \\ {}^2\mathbf{x}_0 \end{pmatrix} = (0,1,0,0,0,1,0,0)^T.$$

Let us synthesize the supervisor for these agents based on the requirement that only one of the activities represented by the PN place $p_1$, $p_4$, $p_5$, $p_8$ can run at the same time. Hence,

$$\begin{aligned} \mathbf{L} &= (1,0,0,1,1,0,0,1); \; \mathbf{B}_s = -\mathbf{L}.\mathbf{B} = (0,1,0,-1,0,1,0,-1) \\ \mathbf{F}_s &= (0,0,0,1,0,0,0,1); \; \mathbf{G}_s^T = (0,1,0,0,0,1,0,0); \; {}^s\mathbf{x}_0 = 1 - \mathbf{L}.\mathbf{x}_0 = 1 \\ \mathbf{F}_a &= \begin{pmatrix} \mathbf{F} \\ \mathbf{F}_s \end{pmatrix}; \; \mathbf{G}_a^T = \begin{pmatrix} \mathbf{G}^T \\ \mathbf{G}_s^T \end{pmatrix}; \; \mathbf{B}_a = \begin{pmatrix} \mathbf{B} \\ \mathbf{B}_s \end{pmatrix} \\ \mathbf{x}_0 a &= \begin{pmatrix} \mathbf{x}_0 \\ {}^s\mathbf{x}_0 \end{pmatrix} = (0,1,0,0,0,1,0,0,1)^T \end{aligned}$$

Comparing these results with Figure 9 we can see that the structure was synthesized correctly.

Of course, the approach can also be used for more than two agents. When we consider e.g. four agents of the same structure and the same initial state, i.e.

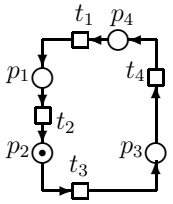



Fig. 10. The simple agent

$$\mathbf{F} = \begin{pmatrix} {}^1\mathbf{F} & \emptyset & \emptyset & \emptyset \\ \emptyset & {}^1\mathbf{F} & \emptyset & \emptyset \\ \emptyset & \emptyset & {}^1\mathbf{F} & \emptyset \\ \emptyset & \emptyset & \emptyset & {}^1\mathbf{F} \end{pmatrix};\ \mathbf{G}^T = \begin{pmatrix} {}^1\mathbf{G}^T & \emptyset & \emptyset & \emptyset \\ \emptyset & {}^1\mathbf{G}^T & \emptyset & \emptyset \\ \emptyset & \emptyset & {}^1\mathbf{G}^T & \emptyset \\ \emptyset & \emptyset & \emptyset & {}^1\mathbf{G}^T \end{pmatrix};\ \mathbf{x}_0 = \begin{pmatrix} {}^1\mathbf{x}_0 \\ {}^1\mathbf{x}_0 \\ {}^1\mathbf{x}_0 \\ {}^1\mathbf{x}_0 \end{pmatrix}$$

we can model different kinds of their mutual cooperation. Let us analyse at least two of them:

1. Pure Mutex – when the simultaneous work of agents is absolutely excluded. Then only one of these agents can work and no cooperation exists. Here,

$$\mathbf{L} = \begin{pmatrix} 1001100110011001 \end{pmatrix};\ \mathbf{F}_s = \begin{pmatrix} 0001000100010001 \end{pmatrix}$$

$$\mathbf{G}_s^T = \begin{pmatrix} 0100010001000100 \end{pmatrix};\ {}^s\mathbf{x}_0 = 1.$$

   In this case the supervisor consists of one PN-place with one token (see ${}^s\mathbf{x}_0$) and its connections with the elementary agents are given by means of matrices $\mathbf{F}_s$ (its non-zero entries represent connections oriented to corresponding agent transitions) and $\mathbf{G}_s$ (its non-zero entries represent connections oriented from corresponding agent transitions to the supervisor place). The matrix $\mathbf{L}$ represents the conditions for the supervisor synthesis.

2. Partial Mutex – when there are e.g. two groups of agents: $\{A_1, A_2, A_3\}$ and $\{A_3, A_4\}$. Only one agent from each group (a representative of the group) can cooperate with the agent from other group. Here,

$$\mathbf{L} = \begin{pmatrix} 1001100110010000 \\ 0000000010011001 \end{pmatrix};\ \mathbf{F}_s = \begin{pmatrix} 0001000100010000 \\ 0000000000010001 \end{pmatrix}$$

$$\mathbf{G}_s^T = \begin{pmatrix} 0100010001000000 \\ 0000000001000100 \end{pmatrix};\ {}^s\mathbf{x}_0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

   Here, the supervisor consists of two PN-places. Each of them has one token. The sense of the matrices is the same as in the previous case.

### 6.2 Mutual Influence of Agents Due to Cooperation

The approach is also suitable for more complicated (practically arbitrary) structures of agents as well as for their interconnections. It is very important as to mutual cooperation of agents in a group. Let us introduce a simple illustrative example.

#### 6.2.1 Example 6

Consider the agent system with structure given in Figure 11 (left).

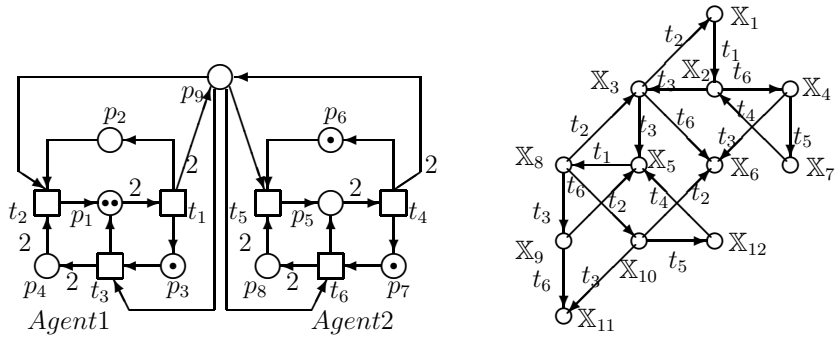



Fig. 11. The PN-based model with a general structure (left) and the corresponding RG of the supervised system

The parameters of the autonomous agents are as follows:

$$ {}^1\mathbf{F} = {}^2\mathbf{F} = \begin{pmatrix} 2&0&0 \\ 0&1&0 \\ 0&0&1 \\ 0&2&0 \end{pmatrix}; {}^1\mathbf{G} = {}^2\mathbf{G} = \begin{pmatrix} 0&1&1&0 \\ 1&0&0&0 \\ 1&0&0&2 \end{pmatrix}; {}^1\mathbf{x}_0 = \begin{pmatrix} 2 \\ 0 \\ 1 \\ 0 \end{pmatrix}; {}^2\mathbf{x}_0 = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} $$

$$ \mathbf{F} = \begin{pmatrix} {}^1\mathbf{F} & \emptyset \\ \emptyset & {}^2\mathbf{F} \end{pmatrix}; \ \mathbf{G} = \begin{pmatrix} {}^1\mathbf{G} & \emptyset \\ \emptyset & {}^2\mathbf{G} \end{pmatrix}; \ \mathbf{B} = \mathbf{G}^T - \mathbf{F}; \ \mathbf{x}_0 = \begin{pmatrix} {}^1\mathbf{x}_0 \\ {}^2\mathbf{x}_0 \end{pmatrix} $$

Let us request that $p_1$ and $p_5$ may keep maximally two tokens together. It may correspond to a real situation from the manufacturing area, where two different machine tools depend on a common source of a material or half-finished products.

$$ p_1 + p_5 \leq 2 \quad \text{or after introducing the slack } p_9: \quad p_1 + p_5 + p_9 = 2. $$

However, it simultaneously yields a "space" for an agreement among the agents. Namely, the agents can coalesce when the ratio $p_1$:$p_2$ will be 2:0, 1:1 or 0:2. Thus,

$$ \begin{aligned} \mathbf{L} &= \begin{pmatrix} 1&0&0&0&1&0&0&0 \end{pmatrix}; \ b = 2; \ \mathbf{B}_s = -\mathbf{L}.\mathbf{B} = \begin{pmatrix} 2&-1&-1&2&-1&-1 \end{pmatrix} \\ \mathbf{F}_s &= \begin{pmatrix} 0&1&1&0&1&1 \end{pmatrix}; \ \mathbf{G}_s^T = \begin{pmatrix} 2&0&0&2&0&0 \end{pmatrix}; \ {}^s\mathbf{x}_0 = b - \mathbf{L}.\mathbf{x}_0 = 2 - 2 = 0. \end{aligned} $$

The RG of the closed loop structure of the agents together with their supervisor is given in Figure 11 (right). Here, the nodes of the RG are given by columns of the following matrix:

$$\mathbf{X}_{reach} = \left(\begin{array}{cccccccccccc}
2 & 0 & 1 & 0 & 2 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 \\
1 & 2 & 1 & 2 & 0 & 1 & 2 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 2 & 0 & 4 & 2 & 0 & 4 & 6 & 4 & 6 & 4 \\
- & - & - & - & - & - & - & - & - & - & - & - \\
0 & 0 & 0 & 1 & 0 & 1 & 2 & 0 & 0 & 1 & 1 & 2 \\
1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\
1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 2 & 0 & 2 & 0 & 0 & 0 & 2 & 2 & 0 \\
- & - & - & - & - & - & - & - & - & - & - & - \\
0 & 2 & 1 & 1 & 0 & 0 & 0 & 2 & 1 & 1 & 0 & 0
\end{array}\right) \begin{array}{l} \left.\begin{array}{l}\\\\\\\\\end{array}\right\} \text{Agent 1} \\ \\ \left.\begin{array}{l}\\\\\\\\\end{array}\right\} \text{Agent 2} \\ \\ \left.\begin{array}{l}\end{array}\right\} \text{Supervisor} \end{array}$$

Any state vector consists of three subvectors – the state of the Agent 1, the state of the Agent2 and the state of the supervisor. The supervisor is the PN subnet consisting of the place $p_9$ and the corresponding connections with subnets representing the agents. These connections are expressed by means of the submatrix $\mathbf{B}_s$ in the matrix $\mathbf{B}_a$ of the closed loop system. This is displayed in the following relation as well as in the initial state vector of the closed loop system:

$$\mathbf{B}_a = \begin{pmatrix} \mathbf{B} \\ \mathbf{B}_s \end{pmatrix} = \begin{pmatrix}
-2 & 1 & 1 & 0 & 0 & 0 \\
1 & -1 & 0 & 0 & 0 & 0 \\
1 & 0 & -1 & 0 & 0 & 0 \\
0 & -2 & 2 & 0 & 0 & 0 \\
- & - & - & - & - & - \\
0 & 0 & 0 & -2 & 1 & 1 \\
0 & 0 & 0 & 1 & -1 & 0 \\
0 & 0 & 0 & 1 & 0 & -1 \\
0 & 0 & 0 & 0 & -2 & 2 \\
- & - & - & - & - & - \\
2 & -1 & -1 & 2 & -1 & -1
\end{pmatrix}; \mathbf{x}_0 a = \begin{pmatrix} \mathbf{x}_0 \\ {}^s\mathbf{x}_0 \end{pmatrix} = \begin{pmatrix}
2 \\ 0 \\ 1 \\ 0 \\ - \\ 0 \\ 1 \\ 1 \\ 0 \\ - \\ 0
\end{pmatrix}$$

Using our approach, we can find e.g. the feasible trajectories from the initial state $\mathbf{x}_0 a = \mathbb{X}_1$ to the terminal state $\mathbf{x}_t a = \mathbb{X}_6$. The feasible state trajectories and the control ones are shown in Figure 12.

## 7 CONCLUSION

The procedure of the automatic synthesis of control for DES and agents and/or MAS which can be described and modelled by P/T PN was proposed and presented here. In the first step the BDG-based method was extended in order to be used for P/T PN with general structure. Namely, till now it could be used only for the

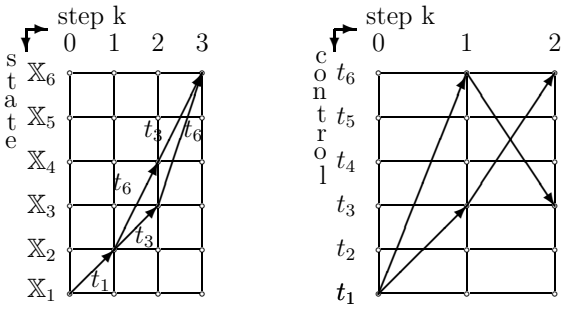



Fig. 12. The feasible state trajectories (left) and the control trajectories (right)

special class of P/T PN named SM, where each PN transition is allowed to have only single input and single output PN place. After extending, each P/T PN transition is allowed to have more than one input PN places and/or more than one output ones. The approach was illustrated on examples. In the second step common utilization of both the proposed approach and of the supervisor synthesis was pointed out and illustrated on several examples. Namely, the supervisor synthesis methods yield the structure of the supervisor. However, the supervisor guarantees only such properties and behaviour of the supervised object that correspond to the conditions for which the supervisor was synthesized, nothing more. It does not yield either the state trajectories or the control trajectories. On the other hand, the approach proposed in this paper yields the sequences of all the control interferences (not only of those cohering with the supervision but in general) of the whole complex (the supervised system together with the supervisor) as well as the feasible state trajectories. Thus, it complements the results achieved by the supervision and allows complex analysis of the behaviour of the supervised system and its control within a full range. Namely, the supervised system is understood to be a system in general, not any specific one, in spite of the fact that it has already been supervised. The supervision helps our approach by means of yielding the (sometimes strongly) limited extent of possibilities how the system to be controlled by our method can behave.

**Acknowledgements**

The research was partially supported by the Slovak Grant Agency for Science VEGA under grant # 2/0075/09. The author thanks VEGA for the support.

## REFERENCES

[1] Čapkovič, F.: Modelling, Analysing and Control of Interactions Among Agents in MAS. Computing and Informatics, Vol. 26, 2007, No. 5, pp. 507–541.

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

[3] 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.

[4] Čapkovič, F.: Supervisory Control of Agents Cooperation. In: Yager, R. R., Sgurev, V. S. and Jotsov, V. S. (Eds.): Proceedings of the 4th IEEE International Conference on Intelligent Systems, Varna, Bulgaria, September 6–8, 2008, Vol. 1, IEEE Press, Piscataway, NJ, USA, 2008, pp. 6_8–6_13.

[5] Desrochers, A.—Al Jaar, R.: Applications of Petri Nets in Manufacturing Systems: Modelling, Control and Performance Analysis. IEEE Press, Piscataway, NJ, 1995.

[6] Iordache, M. V.—Antsaklis, P. J.: Supervision Based on Place Invariants: A Survey. Discrete Event Dynamic Systems, Vol. 16, 2006, pp. 451–492.

[7] Iordache M. V.—Antsaklis, P. J.: Supervisory Control of Concurrent Systems: A Petri Net Structural Approach. Birkhauser, Boston, USA, 2006.

[8] Iordache, M. V.: Methods for the Supervisory Control of Concurrent Systems Based on Petri Net Abstractions. Ph. D. Thesis, University of Notre Dame, USA, Dec. 2003.

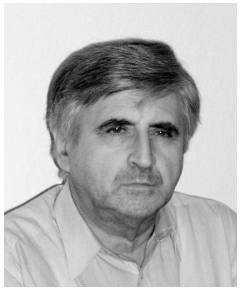
[9] Moody, J. O.—Antsaklis, P. J.: Supervisory Control of Discrete Event Systems Using Petri Nets. Kluwer Academic Publishers, Norwell, Massachusetts, USA, 1998.

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

[11] Ramadge, P.—Wonham, W.: Supervisory Control of a Class of Discrete Event Processes. SIAM Journal on Control and Optimization, Vol. 26, Jan. 1987, pp. 206–230.

[12] Stremersch, G.: Supervision of Petri Nets. Kluwer Academic Publishers, Dordrecht, The Netherland, 2001.

[13] Yamalidou, E.—Moody, J. O.—Antsaklis, P. J.—Lemmon, M. D.: Feedback Control of Petri Nets Based on Place Invariants. Automatica, Vol. 32, No. 1, 1996, pp. 1–28.



**František Čapkovič** received his master degree in 1972 from the Electrotechnical Faculty of 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 Slovak Academy of Sciences. Since 1998 he has been the Associate Professor. He works in the area of modelling, analysing and control of Discrete-Event Systems (DES).