Simulation of concurrent process with Petri-Markov nets

Alexey Nikolaevich Ivutin, Eugene Vasilevich Larkin, Yuri Ivanovich Lutskov, Aleksander Sergeevich Novikov

Tula State University, Lenin av. 92, Tula, 300012, Russia

Abstract. Mathematical apparatus of Petri-Markov nets is described. Petri-Markov simple subnets are introduced. Structures for the simulation of parallelism based on the simple subnets are proposed. Mathematical relationships for the evaluation of the time characteristics of algorithms for a wide class of concurrent computing systems are described.

[Ivutin A.N., Larkin E.V., Lutskov Y.I., Novikov A.S. **Simulation of concurrent process with Petri-Markov nets.** *Life Sci J* 2014;11(11):506-511] (ISSN:1097-8135). http://www.lifesciencesite.com. 86

Keywords: Petri net, semi-Markov process, Petri-Markov net, algorithm, parallelism, concurrency, computer system

Introduction

Currently, a parallelism is a prevailing paradigm of computational processes organisation. In the specialised computer sphere, the idea of parallelism has been used in practice for more than fifty years (CDC-6600 1964 [1] consists of ten independent functional devices operated in parallel), with up-to-date concurrent computations being used in practice in both multi-core processors and computer networks [2]. To organise computational process in such structures, programmers face the problem of non-optimal parallel hardware utilisation, which is associated with such problems as low processors load coefficients and conflicts of access to shared resources [3].

In [4], on the basis of the investigation of the instruction interpretation process using a Von-Neumann computer, it was shown that the number of machine cycles that a processor spends on the execution of a deterministic instruction is a random one. This distribution of the number of machine cycles depends on the hardware speed and the distribution of the processed data. Additionally, in [5], the characteristics of the transitions between algorithm operators for the external observer were classified as quasi-stochastic. As a result, the theory of random processes, particularly Markov (generally semi-Markov) process, should be applied for evaluation of time complexity. A major contribution to the theory was produced by U.K. Belyaev, B.V. Gnedenko, D.R. Koks, D. Lloyd, V.L. Smith, B. Harris, and A.M. Shirokov. Their work potentially may be put on a basis of the mathematical apparatus of time complexity evaluation of sequential algorithms; however, without modifications, and in particular, in the parallelism and/or concurrency area, the use of such a theory is extremely difficult.

The methodology of the modelling of concurrent processes was elaborated in the works by C. Petri, W. Reisig, J. Peterson, and V.E. Kotov [6 - 10], where the apparatus of Petri nets was applied in the research of parallelism. The situational (causal) character of switches in Petri nets [6, 11] predetermine the application of such apparatus for the modelling of algorithm structures and the logic of the events occurring. However, the asynchronous nature of Petri nets theory permits the answering of the question about the principal accessibility of states, but they cannot be used to predict the event occurrence physical time.

Broadening of the classical Petri nets theory leads to a time-extended Petri nets theory [12-14]. Counters for the control of local or global time are included in such models. The time responses of token sojourns in positions, the generation/death of tokens after certain time, etc. are also determined in the model. Models in which the time responses are connected with transitions, notably the Ramchandani-Starke discrete-time model and the Merlin time-continuous model are the most popular [15]. However, even in an improved version, the timed Petri nets do not allow the consideration of the entire variety of interactions in concurrent systems. This fact is connected particularly with the restriction of the logical conditions of net switching with elementary conjunction.

In a model focused on the evaluation of the algorithm time complexity in concurrent computer systems must provide the following features:

a certain and specific strategy of the use of resources for data processing in every parallel system;

a dynamic approach of the release/involving of the computational resources in the process of algorithm execution;

the necessity of data exchange (intermediate results) between computation units and, connected with this phenomenon, the necessity of the synchronisation of the operation of processors;

an availability of the "competition" effect between components functioning concurrently. Such demands were most completely taken into account in the apparatus of Petri-Markov nets (PMN) described below, in which both the aspects relevant to random processes in computational units and the logic aspects of the modulus interaction [16-17] are combined. The models under investigation consider the structures, described parallelism, time-stochastic parameters of semi-Markov processes, and logic of an interaction of the computational units.

Mathematical apparatus of Petri-Markov nets (PMN)

The most general description of the PMN approach is based on the construction of a system of sets that comprises the PMN. The Petri-Markov net is a structural parametric model determined by the set:

$$\Psi = \{\Pi, M\}, \quad (1)$$

where $\Pi = \{A, Z, \widetilde{\mathbf{R}}, \hat{\boldsymbol{R}}\}$ - is a set that describes a structure of a directed bichromatic graph, which describes a Petri net; $A = \{a_{1(a)}, ..., a_{j(a)}, ..., a_{J(a)}\}$ - is a finite set of positions; $Z = \{z_{1(z)}, ..., z_{j(z)}, ..., z_{J(z)}\}$ - is a finite set of transitions; $J(a)$ - is the size of a set of positions; $J(z)$ - is the size of a set of transactions; $\widetilde{\boldsymbol{R}} = \left(\tilde{r}_{j(a)j(z)}\right)$ - is an adjacency matrix of size $J(a) \times J(z)$, which represents the positions of set A to the transactions of set Z; $\hat{\boldsymbol{R}} = \left(\hat{r}_{j(z)j(a)}\right)$ - is an adjacency matrix of size $J(z) \times J(a)$, which represents the transactions of set Z to the positions of set A; $M = \{\boldsymbol{q}, \boldsymbol{h}(t), \boldsymbol{\Lambda}\}$ - is a set of parameters, being applied to structure Π; $\boldsymbol{q} = (q_{1(z)}, ..., q_{j(z)}, ..., q_{J(z)})$ - is a vector that determines the probabilities of beginning a process in one of the transitions of set Z; $\boldsymbol{h}(t) = [h_{j(a)j(z)}(t)]$ - is a semi-Markov matrix of size $J(a) \times J(z)$; t - is time; $\boldsymbol{\Lambda} = [\lambda_{i(z)i(a)}]$ - is a matrix of the logical conditions of size $J(a) \times J(z)$;

$$\hat{r}_{j(z)j(a)} = \begin{cases} 1, \text{ if } a_{j(a)} \in O_A\left(z_{j(z)}\right); \\ 0, \text{ if } a_{j(a)} \notin O_A\left(z_{j(z)}\right); \end{cases}$$

$O_A(Z) = \{O_A(z_{1(z)}), ..., O_A(z_{j(z)}), ..., O_A(z_{J(z)})\}$ - is an output of the functions of the transitions;

$$\boldsymbol{h}(t) = \boldsymbol{p} \otimes \boldsymbol{f}(t) = \left[p_{j(a)j(z)} \cdot f_{j(a)j(z)}(t)\right] = \left[h_{j(a)j(z)}(t)\right] \quad (2)$$

$$\lambda_{j(z)j(a)} = \begin{cases} \lambda[I_A(z_{j(z)})], \text{ if } a_{j(a)} \in O_A(z_{j(z)}); \\ 0, \text{ if } a_{j(a)} \notin O_A(z_{j(z)}); \end{cases} \quad (3)$$

$I_A(Z) = \{I_A(z_{1(z)}), ..., I_A(z_{j(z)}), ..., I_A(z_{J(z)})\}$ - are the input functions of the transitions; $\boldsymbol{p} = \left[p_{j(a)j(z)}\right]$ - is the stochastic matrix of a semi-Markov process; $\boldsymbol{f}(t) = \left[f_{j(a)j(z)}(t)\right]$ - is the matrix of the time densities of a semi-Markov process; $\otimes$ - is the symbol of a matrix direct multiplication.

The positions of the PMN simulate the operators of the concurrent algorithm. The transitions simulate the interactions of the operators.

Regarding the probabilities and densities, the following constraints apply:

$$\sum_{j(z)=1}^{J(z)} q_{j(z)} = 1;$$

$$\sum_{j(z)=1}^{J(z)} p_{j(a)j(z)} = 1;$$

$$f_{j(a),j(z)}(t) = 0, \text{ if } t < 0;$$

$$\int_0^\infty f_{j(a),j(z)}(t)dt = 1.$$

For a numerical analysis of the processes in parallel computing systems, the following parameters may be determined:

the matrix of expectations

$$\boldsymbol{T} = \left[T_{j(a)j(z)}\right] = \int_0^\infty t\boldsymbol{f}(t)dt\,\boldsymbol{T} \quad (4)$$

the matrix of dispersions $\boldsymbol{D} = (D_{j(a)j(z)})$ in the form

$$\boldsymbol{D} = \int_0^\infty t^2 \boldsymbol{f}(t)dt - \boldsymbol{T} \otimes \boldsymbol{T}. \quad (5)$$

We will differentiate between the connected and the disconnected PMN. Let us change the arcs $\left[a_{j(a)}, z_{j(z)}\right]$, $\left[z_{j(z)}, a_{j(a)},\right]$, $a_{j(a)} \in A$, $z_{j(z)} \in Z$ onto the edges $\left\{a_{j(a)}, z_{j(z)}\right\}$. The modified net may be called feebly connected if one of the following paths can be built between its different objects:

$$\begin{aligned} &a_{j(a)} \leftrightarrow a_{k(a)},\ 1(a) \le j(a),\ k(a) \le J(a); \\ &z_{j(z)} \leftrightarrow z_{k(z)},\ 1(z) \le j(z),\ k(z) \le J(z); \\ &a_{j(a)} \leftrightarrow z_{j(z)},\ 1(a) \le j(a) \le J(a);\ 1(z) \le j(z) \le J(z). \end{aligned} \quad (6)$$

The initial net may be called strongly connected if the modified net is a feebly connected one, and the paths

$$\begin{aligned} &a_{j(a)} \to a_{k(a)},\ 1(a) \le j(a),\ k(a) \le J(a); \\ &z_{j(z)} \to z_{k(z)},\ 1(z) \le j(z),\ k(z) \le J(z); \\ &a_{j(a)} \to z_{j(z)},\ 1(a) \le j(a),\ k(a) \le J(a);\ 1(z) \le j(z),\ k(z) \le J(z); \end{aligned} \quad (7)$$

$z_{j(z)} \to a_{j(a)}$, $1(a) \le j(a)$, $k(a) \le J(a)$; $1(z) \le j(z)$, $k(z) \le J(z)$

are retained after the restoration of the arcs $\left\lfloor a_{j(a)}, z_{j(z)} \right\rfloor$, $\left\lfloor z_{j(z)}, a_{j(a)}, \right\rfloor$ instead of the edges $\left\{ a_{j(a)}, z_{j(z)} \right\}$.

Set $Z = \{z_{1(z)}, ..., z_{j(z)}, ..., z_{J(z)}\}$ is divided into four disjoint subsets: the subsets of the beginning transitions Z_B, the subsets of the ending transitions Z_E, the subsets of the primitive transitions Z_P and the subset of the synchronisation transitions Z_S:

$$Z = Z_B \cup Z_E \cup Z_P \cup Z_S, \quad (8)$$

$$Z_B \cap Z_E = \varnothing;\ Z_B \cap Z_P = \varnothing;\ Z_B \cap Z_S = \varnothing;\ Z_E \cap Z_P = \varnothing;\ Z_E \cap Z_S = \varnothing;\ Z_P \cap Z_S = \varnothing;$$

For the mentioned types of transitions, the following conditions are true:

if $z_{j(z)} \in Z_B$, then $|I_A(z_{j(z)})| = 0$ and $|O_A(z_{j(z)})| \ge 1$;
if $z_{j(z)} \in Z_E$, then $|I_A(z_{j(z)})| \ge 1$ and $|O_A(z_{j(z)})| = 0$;
if $z_{j(z)} \in Z_P$, then $|I_A(z_{j(z)})| = 1$ and $|O_A(z_{j(z)})| = 1$;
if $z_{j(z)} \in Z_S$, then $|I_A(z_{j(z)})| \ge 1$ and $|O_A(z_{j(z)})| \ge 1$,

where $|...|$ denotes the size of a set.

The beginning, ending, and synchronisation transitions form the subset of the so-called non-primitive ones:

$$Z_{NP} = Z_B \cup Z_E \cup Z_S. \quad (9)$$

One of the elements of PMN as a mathematical apparatus for the simulation of concurrent algorithms is the notion of a token as a certain pointer. If a token is placed at position $a_{j(a)}$, this indicates that the named position at the present time t is in the active state. This, in turn, means that at the present time t, the operator that is simulated by position $a_{j(a)}$ is executed by a pre-defined computation unit.

The functional similarity of the interpretation of algorithm operators in a parallel computing system is a sequence of state exchanges that is realised as a sequence of half-steps $\sigma_{j(a),j(z)} = (a_{j(a)}, z, j_{(z)})$, i.e., executed from positions into transitions, or $\sigma_{j(z),j(a)} = (a_{j(a)}, z, j_{(z)})$, i.e., executed from transitions into positions. Two consecutive half-steps form a step. While executing a half-step $\sigma_{j(a),j(z)}$, the token is withdrawn from the position $a_{j(a)} \in I_A(z, j_{(z)})$ and placed into the transition z, $j_{(z)} \in O_Z(a, j_{(a)})$. While executing a half-step $\sigma_{j(z),j(a)}$, the token is withdrawn from the transition z, $j_{(z)}$ and placed into the position $a_{j(a)} \in O_A(z, j_{(z)})$.

The half-step $\sigma_{j(a),j(z)}$ is executed if a decision about the execution of a certain half-step is made with probability $p_{j(a)j(z)}$ and the time of half-step execution, which is defined with density $f_{j(a)j(z)}(t)$, has elapsed. The half-step $\sigma_{j(z)j(a)}$ is executed if logical condition (3) is fulfilled, i.e., if $\lambda_{j(z)j(a)}$ is equal to $\lambda_{j(z)j(a)} = 1$.

Petri-Markov simple subnets

A Petri-Markov net (1) may be divided into subnets, whose structure is described as follows:

$$\Pi^k = \{A^k, Z^k, \widetilde{\boldsymbol{R}}^k, \hat{\boldsymbol{R}}^k\}, \quad (10)$$

where $A^k \subseteq A$; $Z^k \subseteq Z$. $\widetilde{\boldsymbol{R}}^k \subseteq \widetilde{\boldsymbol{R}}$; $\hat{\boldsymbol{R}}^k \subseteq \hat{\boldsymbol{R}}$.

The matrix $\widetilde{\boldsymbol{R}}^k$ can be determined from matrix $\widetilde{\boldsymbol{R}}$ by means of the deletion of the rows corresponding to positions $A \backslash A^k$ and the columns corresponding to the transitions $Z \backslash Z^k$, where \ is a symbol designating the set difference operation. The matrix $\hat{\boldsymbol{R}}^k$ is determined from matrix $\hat{\boldsymbol{R}}$ by means of the deletion of the rows corresponding to transitions $Z \backslash Z^k$ and the columns corresponding to the positions $A \backslash A^k$. The subnet Π^k is also a directed bichromatic graph.

Let us assume that a PMN is broken up into subnets in the following manner:

$$\Psi = \bigcup_{k=1}^{K} \Psi^k, \quad (11)$$

where $\Psi^k = \{\Pi^k, M^k\}$ - is a Petri-Markov simple subnet (PMSS); $\Pi^k = \{A^k, Z^k, \widetilde{\boldsymbol{R}}^k, \hat{\boldsymbol{R}}^k\}$ - is the structure of a PMSS; $A^k = \{a^k_{1(a,\,k)}, ..., a^k_{j(a,\,k)}, ..., a^k_{J(a,\,k)}\} \subseteq A$; $Z^k = \{z^k_{1(z,\,k)}, ..., z^k_{j(z,\,k)}, ..., z^k_{J(z,\,k)}\} \subseteq A$; $\widetilde{\boldsymbol{R}}^k = \left(\tilde{r}^k_{j(a,k)j(z,k)}\right)$; $\hat{\boldsymbol{R}}^k = \left(\hat{r}^k_{j(z,k)j(a,k)}\right)$; $M^k = \{\boldsymbol{q}^k, \boldsymbol{h}^k(t), \boldsymbol{\Lambda}^k\}$ - are the characteristics of a PMSS; $\boldsymbol{q}^k = (q^k_{1(z,\,k)}, ..., q^k_{j(z,\,k)}, ..., q^k_{J(z,\,k)})$; $\boldsymbol{h}^k(t) = (h^k_{j(a,\,k)j(z,\,k)}(t))$; $\boldsymbol{\Lambda}^k = (\lambda^k_{i(z,\,k)i(a,\,k)})$; $\boldsymbol{h}^k(t) = \boldsymbol{p}^k \otimes \boldsymbol{f}^k(t)$; $\boldsymbol{p}^k = (p^k_{j(a,\,k)j(z,\,k)})$; $\boldsymbol{f}^k(t) = (f^k_{j(a,\,k)j(z,\,k)}(t))$.

The Petri-Markov simple subnets Ψ^k intersect on non-primitive transitions, i.e.,

$$\left[\Pi^k \cap \left(\bigcup_{\substack{l=1,\\ l\ne k}}^{K} \Pi^l\right)\right] \subseteq Z_{NP},$$

$$1 \le k \le K. \quad (12)$$

All other transitions of a PMSS are the primitive ones, i.e., if $z_{j(z,k)} \in Z^k$ and

$z_{j(z,k)} \notin \left(\bigcup_{\substack{l=1,\\ l\neq k}}^{K} \Pi^{l} \right)$, then $I_A\left(z_{j,k}\right) \subset A^k$, $O_A\left(z_{j,k}\right) \subset A^k$ and $\left|I_A\left(z_{j,k}\right)\right| = \left|O_A\left(z_{j,k}\right)\right| = 1$.

Every PMSS Ψ^k includes at least one transition of a subset $Z_B \cup Z_S$ and one transition of a subset $Z_E \cup Z_S$. If $O_A\left(Z_B^k\right) \subseteq A^k$, then the transitions of subset $Z^k \cap \left(Z_B \cup Z_S\right) = Z_B^k$ form a subset Z_B^k of the initial transitions of PMSS Ψ^k. If $I_A\left(Z_E^k\right) \subseteq A^k$, then the transitions of subset $Z^k \cap \left(Z_E \cup Z_S\right) = Z_E^k$ form a subset Z_E^k of the ending transitions of PMSS Ψ^k.

The positions and primitive transitions inside of PMSS Π^k form typical structures, which are shown in fig. 1.

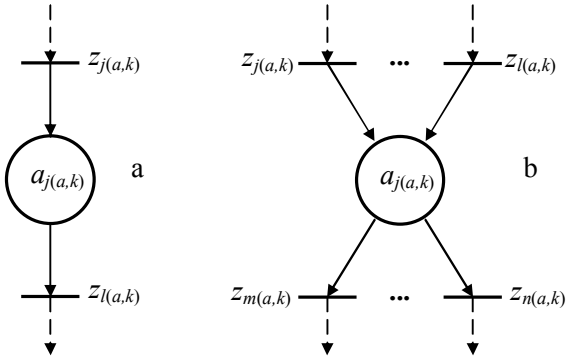

Fig. 1. Typical structures inside a PMSS

Structure

$$\left\{\left\{a_{j(a,k)},\ldots\right\},\left\{z_{j(z,k)},z_{l(z,k)},\ldots\right\},\begin{bmatrix}\ldots & \ldots & \ldots & \ldots\\ \ldots & 0 & 1 & \ldots\\ \ldots & \ldots & \ldots & \ldots\end{bmatrix},\begin{bmatrix}\ldots & \ldots & \ldots\\ \ldots & 1 & \ldots\\ \ldots & 0 & \ldots\\ \ldots & \ldots & \ldots\end{bmatrix}\right\}, \quad (13)$$

which is shown in fig. 1 a, simulates the linear part of a sequential algorithm, which is executed by one computation unit (i.e., of Von-Neumann type).

Structure

$$\left\{\left\{a_{j(a,k)},\ldots\right\},\left\{z_{j(z,k)},z_{l(z,k)},z_{j(z,k)},z_{l(z,k)},\ldots\right\},\begin{bmatrix}\ldots & \ldots & \ldots & \ldots & \ldots & \ldots\\ \ldots & 0 & 0 & 1 & 1 & \ldots\\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots\end{bmatrix},\begin{bmatrix}\ldots & \ldots & \ldots\\ \ldots & 1 & \ldots\\ \ldots & 1 & \ldots\\ \ldots & 0 & \ldots\\ \ldots & 0 & \ldots\\ \ldots & \ldots & \ldots\end{bmatrix}\right\}, \quad (14)$$

which is shown in fig. 1 b, simulates the ramified part of an algorithm, which is interpreted using one computation unit.

The typical structures, which are shown in fig. 2, are intended for the simulation of parallelism as follows:

"fork" (fig. 2 a) -

$$\left\{\left\{a_{j(a,k)},a_{j(a,l)},a_{j(a,m)},\ldots\right\},\left\{z_{j(z)},\ldots\right\},\begin{bmatrix}\ldots & \ldots & \ldots\\ \ldots & 1 & \ldots\\ \ldots & 0 & \ldots\\ \ldots & 0 & \ldots\\ \ldots & \ldots & \ldots\end{bmatrix},\begin{bmatrix}\ldots & \ldots & \ldots & \ldots & \ldots\\ \ldots & 0 & 1 & 1 & \ldots\\ \ldots & \ldots & \ldots & \ldots & \ldots\end{bmatrix}\right\}; \quad (15)$$

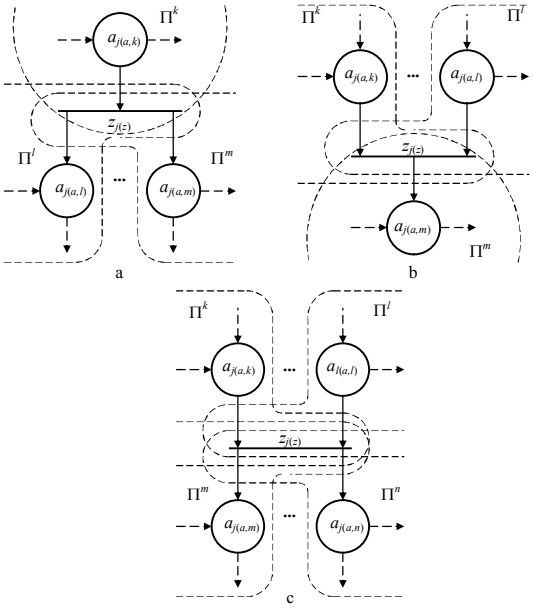

Fig. 2. Structures for the simulation of parallelism

"joint" (fig. 2 b) -

$$\left\{\left\{a_{j(a,k)},a_{j(a,l)},a_{j(a,m)},\ldots\right\},\left\{z_{j(z)},\ldots\right\},\begin{bmatrix}\ldots & \ldots & \ldots\\ \ldots & 1 & \ldots\\ \ldots & 1 & \ldots\\ \ldots & 0 & \ldots\\ \ldots & \ldots & \ldots\end{bmatrix},\begin{bmatrix}\ldots & \ldots & \ldots & \ldots & \ldots\\ \ldots & 0 & 0 & 1 & \ldots\\ \ldots & \ldots & \ldots & \ldots & \ldots\end{bmatrix}\right\}; \quad (16)$$

"sync" (fig. 3 c) -

$$\left\{\left\{a_{j(a,k)},a_{j(a,l)},a_{j(a,m)},a_{j(a,n)},\ldots\right\},\left\{z_{j(z)},\ldots\right\},$$

$$\left\{\begin{bmatrix} \dots & \dots & \dots \\ \dots & 1 & \dots \\ \dots & 1 & \dots \\ \dots & 0 & \dots \\ \dots & 0 & \dots \\ \dots & \dots & \dots \end{bmatrix}, \begin{bmatrix} \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & 0 & 0 & 1 & 1 & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix}\right\}. \quad (17)$$

In (15) and (16) transition $z_{j(z)} = Z^k \cap Z^l \cap Z^m$. In (17), transition $z_{j(z)} = Z^k \cap Z^l \cap Z^m \cap Z^n$.

Let us consider the transitions $z_{j(z,k)} \in Z^k \cap Z_S$, which have on one position belonging to subset A^k both input and output functions (fig. 3), i.e., $\left|I_A\left(z_{j(z,k)}\right) \cap A^k\right| = 1$, $\left|O_A\left(z_{j(z,k)}\right) \cap A^k\right| = 1$.

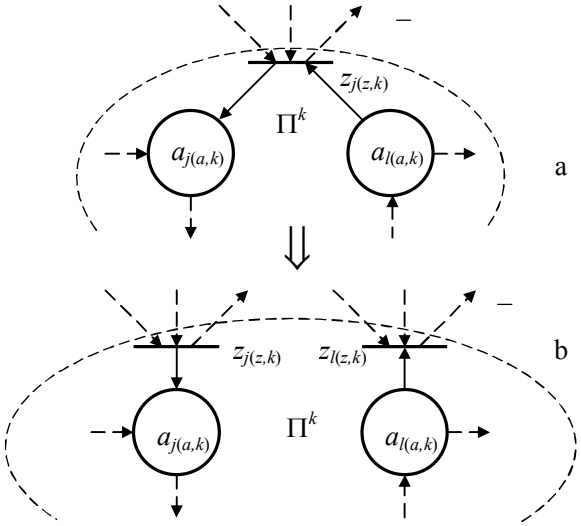

Fig. 3. Disintegration of the synchronising transition

The structure

$$\left\{\{a_{j(a,k)}, a_{l(a,k)}, \dots\}, \{z_{j(z,k)}, \dots\}, \begin{bmatrix} \dots & \dots & \dots \\ \dots & 0 & \dots \\ \dots & 1 & \dots \\ \dots & \dots & \dots \end{bmatrix}, \begin{bmatrix} \dots & \dots & \dots & \dots \\ \dots & 1 & 0 & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix}\right\} \quad (18)$$

may be transformed to the structure

$$\left\{\{a_{j(a,k)}, a_{l(a,k)}, \dots\}, \{z_{j(z,k)}, z_{j(z,k)}, \dots\}, \begin{bmatrix} \dots & \dots & \dots & \dots \\ \dots & 0 & 0 & \dots \\ \dots & 0 & 1 & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix}, \begin{bmatrix} \dots & \dots & \dots & \dots \\ \dots & 1 & 0 & \dots \\ \dots & 0 & 0 & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix}\right\} \quad (19)$$

by means of splitting the transition $z_{j(z,k)}$ into the transitions $z_{j(z,k)}$ and $z_{l(z,k)}$. The transition $z_{j(z,k)}$ has no positions in the input function belonging to subset A^k, i.e., $\left|I_A\left(z_{j(z,k)}\right) \cap A^k\right| = 0$ and $\left|O_A\left(z_{j(z,k)}\right) \cap A^k\right| = 1$, and it can be classified as the initial transition of PNSS Π^k. The common output function of this transition does not change, and from the input function, a position belonging to subset A^k is excluded.

Transition $z_{l(z,k)}$ has no positions in the output function belonging to subset A^k, i.e., $\left|I_A\left(z_{l(z,k)}\right) \cap A^k\right| = 1$ and $\left|O_A\left(z_{l(z,k)}\right) \cap A^k\right| = 0$. Thus, the transition can be classed as the final transition of PNSS Π^k. The common input function of this transition is equal to the input function of transition $z_{j(z,k)}$ before splitting. The output function of $z_{l(z,k)}$ may be formed by excluding from the output function of $z_{j(z,k)}$ a position belonging to the subset A^k that does not change and excluding from the input function a position belonging to subset A^k.

Thus, instead of set Z^k, due to the splitting during every synchronisation transition of set Z^k on the beginning and ending transitions, subset $\hat{Z}^k$ may be formed, in which the elements may be clustered onto three groups:

the subset of beginning transitions $\hat{Z}^k_B$, which includes transitions of subset Z^k_B and the corresponding parts of the disintegrated synchronisation transitions, with numbers $1 \le j(z, k) \le M(z, k)$;

the subset of primitive transitions Z_P^k with numbers $M(z, k) + 1 \le j(z, k) \le N(z, k)$;

the subset of ending transitions $\hat{Z}_E^k$, which includes transitions of subset Z_E^k and the corresponding parts of the disintegrated synchronisation transitions, with numbers $N(z, k) + 1 \le j(z, k) \le \hat{J}(z,k)$.

Conclusions

An effective and rather straightforward mathematical apparatus was developed for modelling concurrent computing systems. The apparatus is oriented towards the evaluation of the time complexity algorithm implemented in such systems. In models along with the structural and time aspects of the functioning computer units the logics if interaction of units is taken into account.

Acknowledgements

The authors thank the anonymous reviewers for their helpful suggestions. The Ministry of Education and Science of the Russian Federation supported this research.

Corresponding Author:
Dr.Ivutin Alexey Nikolaevich
Tula State University
Lenin av. 92, Tula, 300012, Russia

References

1. Bell, C., 1985. A new class of multiprocessor computers. Science, 228(April): 462-467.
2. Culler, B., O. Singh and A. Gupta, 1999. Parallel computer architecture: a hardware/software approach. San Francisco: Morgan Kaufmann Publishers, Inc.
3. Lewis, T., 1997. Foundations of Parallel Programming: A Machine-Indepedent Approach. Los Alamitos, CA: IEEE Computer Society Press.
4. Beizer, B., 1978. Micro-Analysis of Computer System Performance. New York, NY: John Wiley & Sons, Inc.
5. Ferrari, B., 1978. Computer Systems Performance Evaluation. Englewood Cliffs, NJ: Prentice-Hall.
6. Petri, C.A., 1996. Nets, time and space. Theoretical Computer Science, 153(1-2): 3–48.
7. Reisig, W., 1998. Elements Of Distributed Algorithms: Modeling and Analysis with Petri Nets. Springer-Verlag.
8. Reisig, W., 1994. Correctness proofs of distributed algorithms. Theory and Practice in Distributed Systems. International Workshop, Dagstuhl Castle, Germany, pp: 164–177.
9. Peterson, J.L., 1981. Petri Net Theory and the Modeling of Systems. Prentice Hall.
10. Kotov, V.E., 1984. Petri Nets. Moscow: Nauka.
11. Murata, T., 1989. Petri nets: Properties, analysis and applications. Proceedings of the IEEE, 77(4): 541–580.
12. Balsamo, S., P.G. Harrison and A. Marin, 2012. Methodological construction of product-form stochastic Petri nets for performance evaluation. Journal of Systems and Software, 85(7): 1520–1539.
13. Choi, H., V. Kulkarni and K. Trivedi, 1994. Markov regenerative stochastic Petri nets. Performance Evaluation, 5316(94): 337–357.
14. Kristensen, L.M., J.B. Jorgensen and K. Jensen, 2004. Application of coloured petri nets in system development. Lectures on Concurrency and Petri Nets, 3098: 19–27.
15. Jensen, K., 1992. Coloured Petri Nets - Basic Concepts, Analysis Methods and Practical Use. - Volume 1: Basic Concepts. Springer-Verlag.
16. Ramaswamy, S. and K.P. Valavanis, 1996. Hierarchical Time-Extended Petri Nets (H-EPN) Based Error Identification and Recovery for Hierarchical System. IEEE Trans. on Systems, Man, and Cybernetics- Part B: Cybernetics, 26(1): 164 -175.
17. Ignatiev, V.M. and У.M. Larkin, 1997. Patri-Markov Nets (In Russian). Tula: Tula State University.

7/4/2014