MODELLING AND CONTROL OF COMPUTATIONAL PROCESSES USING MAX-PLUS ALGEBRA

Jerzy Raszka, Lech Jamroż

Abstract*: In this paper we propose a modelling technique for the control of computational processes. The Petri Net model, particularly a Timed Event Graph (TEG) can be used for analyzing. The proposed model enables the determination of state equations. The max-plus algebra represents linear algebraic form of discrete systems and supplies new tools to their modelling. We develop a linear mathematical model under constraints in the Max-plus algebra. When using max-plus algebra with TEG, the arc weights are kept equal to one in order to be able to resolve the state equations. Structure of max-plus algebra is equipped with maximization and addition operations over of the real numbers and minus infinity. It can be used appropriately to determine marking times within a given Petri net and a vector filled with marking state at the beginning. Tools of max-plus algebra are useful to investigate properties of network models. Finally, numerical examples show the use of this model.*

Keywords*: Max-Plus-Linear Systems, Petri Nets, Discrete Systems, Modelling Technique*

ACM Classification Keywords*: H. Information Systems, H.1 MODELS AND PRINCIPLES, H.1.1 Systems and Information Theory; D. Software D.4, OPERATING SYSTEMS, D.4.1 Process Management, Multiprocessing/multiprogramming/multitasking, Synchronization.*

Introduction

Recent technological achievements require advances beyond the existing computational models in order to be used effectively. For example the Internet has progressed from a simple store-and-forward network to a more complex communication infrastructure. In order to meet demands on security, flexibility and performance, network traffic not only needs to be forwarded, but also processed on routers. Pragmatic aspects of current and future computer systems will be modelled so that realistic estimates of efficiency can be given for algorithms and controlling of computations in these new settings. Proposed methods deals with the performance evaluation of a communication infrastructure system in terms of waiting times of data and starting points of computing in various connections. The behaviour of a system is studied in the framework of discrete systems

Discrete systems and specially discrete-event dynamical systems often arise in the context of parallel computing, manufacturing systems [Jamroż, Raszka 1997], for project management, railway networks [Goverde, Rob 2007], telecommunication networks, etc. In the last years there has been a growing quantity of research on discrete systems that can be modelled as max-plus linear systems. Most of the earlier literature on this class of systems set included modelling, performance and properties analysis, rather than control [Bacceli, Cohen and alt. 1992], [Jamroż, Raszka 1997], [Nait-Sidi-Moh, Manier 2009] and many others e.g. J. Bernd Heidergott, Geert Jan Olsder, Didier *Dubois*, Jean-Pierre Quadrat and Jacob van der Woude. Lately there are articles on control for max plus systems [Maia, Andrade 2011] [Nait-Sidi-Moh, Manier 2009], [Schutter, Boom 2001].

Two possible operating modes of discrete systems can be observed at each operating: periodic and no periodic mode. Two complementary tools, Petri nets and (max, +) algebra, are used to describe the network by a linear state model. This one can be solved after solving the structural conflicts associated to the graphical

representation. From the characteristic matrix of the mathematical model, we may determine eigenvalues and eigenvectors that we use to evaluate the operations.

Petri nets (PN) [Murata 1989] are very popular formalism for the analysis and representation of parallel and distributed computing in concurrent systems that has draw much attention to modelling and verification of this type of systems. P systems, also referred to as membrane systems [Gutuleac 2006], are a class of parallel and distributed computing models [6]. The interest of relating P systems with the PN model of computation lead to several important results on simulation and decidability issues. Some efforts have been made to simulate P systems with Petri nets to verifying the many useful behavioural properties such as reachability, boundedness, liveness, terminating, etc.

When considering processes from manufacturing or chemical engineering, their behaviour can often be adequately represented by a discrete event model accounting for the typically discrete sensor and actuator equipment of such processes. In addition, the behaviour of these processes is often adequately described by a sequence of transitions between discrete process states. The focus of this contribution is on a particular class of such discrete event systems where synchronization and controlling of computational processes occurs. This system class has gained significant attention in recent years due to the fact that the sequences of event times for such processes can be described by equations which are linear in a particular algebra, the so called max–plus algebra [Bacceli Cohen 1992]. The resulting equations exhibit a structural equivalence to system descriptions from conventional control engineering such as transfer functions or state space models.

Thus, a system theory for these max–plus–linear systems has been developed, and various concepts well known from control engineering have been adapted to this system class in control design and diagnosis.

Sometimes for modelling Timed event graphs (TEG's) are a subclass of timed Petri nets which can be used for modelling Discrete Event Dynamic Systems (DEDS) subject to synchronisation phenomena as manufacturing systems, multiprocessor systems and especially transportation.

In this paper, we propose a deterministic Petri net model for the computational system that can be considered as a discrete event system. Moreover, such DES can be easily modelled with a subclass of Petri net for evaluation purposes, we suggest then a TEG approach to model, analyse and control a computational process From this TEG model, we formulate a mathematical model based on the max-plus algebra. The behaviour of this DES can be described easily by a linear system in this algebra. In the second part, we introduce model of computational process. The third part presents an overview of max algebra theory and the specific model will be analysed. In the last part we present the simulations results.

Computational Processes

When considering computational processes that allows event-driven applications to take advantage of multiprocessors by running code for event handlers in parallel. To obtain high performance, servers must overlap computation with I/O. Programs typically achieve this overlap using threads or events. Threaded programs typically process each request in a separate thread; when one thread blocks waiting for I/O, other threads can run. Event-based programs are structured as a collection of call-back functions which a main loop calls when I/O events occur. Threads provide an intuitive programming model, but require coordination of accesses by different threads to shared state, even on a uniprocessor. Event-based programs execute call-backs serially, so the programmer need not worry about concurrency control; however, event-based programs until now have been unable to take advantage of multiprocessors. Much of the effort required to make existing event-driven programs take advantage of multiprocessors is in specifying which events may be handled in parallel.

As example in this paper is consider the simple problem of designing control of system where the cost is chosen such that it provides a trade off between minimizing delays of time of end of operations of computational process (real time of complete of all tasks in computational periodic process, times of final results of one cycle) and periodicity of desired output (wished needed time) to complete process

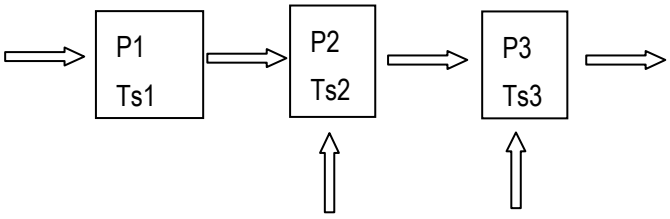

Fig. 1. Structure of the process

Simple computational process consist several tasks which linked by the waiting for I/O data (Figure 1). To illustrate our approach, we consider here a process that is constituted by three tasks: Ts1, Ts2, Ts3 which run on three processors: P1, P2 and P3. Every one of these tasks is operated on the dedicated processor. Within this process circulate information flows in several directions; these can be processing data input/output and signal data. Outer input data are processed as a first task on the P1 and its output data have to be saved in the memory waiting to be processed. The second and third processors operate in the same way but they input data are as output results from the first and second processor respectively. Moreover tasks Ts1 and Ts2 need an extra outer data.

The aim of this modelling is to evaluate command for the process according to pre-established criteria. For instance, to have a continuous computation (on processor P3, we have to prepare time for input data to other processors P1 and P2 also enough of memory to save an input, output and temporary data. A good schedule will allow maintaining a minimal costs and optimising the size of memory required.

Petri Net Model

Petri nets, a graph-oriented formalism, allow to model and analyze systems, which comprise properties such as concurrency and synchronization.

A Petri net model of a dynamic system consists of two parts: net structure and marking. A net structure is a weighted-bipartite directed graph that represents the static part of the system. A marking is representing a distributed overall state on the structure. This separation allows one to reason on net based model at two levels - structural and behavioral.

Net structure is built on two disjoint sets of objects: places and transitions, which are connected by arcs. In the graphical representation, places are drawn as circles, transitions are drawn as thin bars and arcs are drawn as arrows. Places may contain tokens, which are drawn as dots. The vector representing in every place the number of tokens is the state of the Petri net and is referred to as its marking. This marking can be changed by the firing of the transitions. A Petri nets do not include any notion of time are aimed to model only the logical behavior of systems. The introduction of a timing specification is essential if we want to use this class of model to consider performance problem.

More formally timed Petri nets (TPN) are 5-tuples [Murata 1989] :TPN= (P,T,F,M_0,τ), where $P=(p_1,p_2,...,p_n)$, $|P|\neq 0$; $T=(t_1,t_2,...,t_m)$, $|T|\neq 0$ is a finite disjunct set of suitable places and transitions; M_0: $P\rightarrow N$ is the initial marking function which defines the initial number of tokens for every place. ($N=\{0,1,...\}$); τ: $T\rightarrow R^+$ is

the firing time function, and $F \subset (PxT) \cup (TxP)$ is the set of arcs. The problem evoked above can be modelled by a Petri Net (Figure 2).

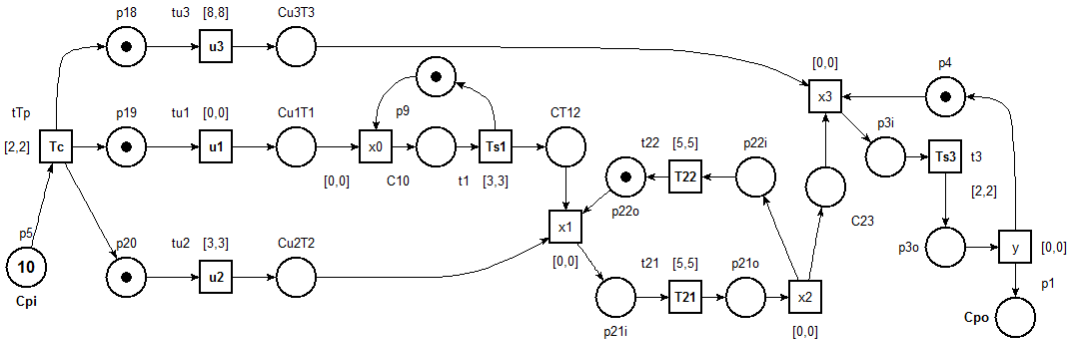

Fig. 2. Timed Petri net for simulation of the system process

The net has three inputs u_1, u_2, and u_3 and one output y. The firing time u_1, u_2, and u_3 are the start times of task Ts1, Ts2, Ts3 respectively. Finally, the ending time of the process is represented by the firing time of the transition y.

Max-Plus Algebra

Define

The basic operations of the max-plus algebra [Bacceli Cohen 1992] are maximization and addition, which will be represented by $\oplus$ and $\otimes$ respectively: $x \oplus y = \max(x, y)$ and $x \otimes y = x + y$ for $x, y \in \mathbb{R}_\varepsilon =^{def} \mathbb{R} \cup \{-\infty\}$

The reason for using these symbols is that there is a remarkable analogy between $\oplus$ and conventional addition, and between $\otimes$ and conventional multiplication: many concepts and properties from linear algebra (such as the Cayley-Hamilton theorem, eigenvectors and eigenvalues, Cramer's rule, ...) can be translated to the max-plus algebra by replacing $+$ by $\oplus$ and $\times$ by $\otimes$. Therefore, we also call $\oplus$ the max-plus-algebraic addition, and $\otimes$ the max-plus-algebraic multiplication. Note however that one of the major differences between conventional algebra and max-plus algebra is that in general there do not exist inverse elements w.r.t. $\oplus$ in $\mathbb{R}_\varepsilon$. The zero element for $\oplus$ is $\varepsilon =^{def} -\infty$ we have $a \oplus \varepsilon = a = \varepsilon + a$ for all $a \in \mathbb{R}_\varepsilon$. The structure $(\mathbb{R}_\varepsilon, \oplus, \otimes)$ is called the max-plus algebra.

Let $r \in \mathbb{R}$ The r-th max-plus-algebraic power of $x \in \mathbb{R}$ is denoted by $x^{\otimes r}$ and corresponds to rx in conventional algebra. If $r \in \mathbb{R}$ then $x^{\otimes 0} = 0$ and the inverse element of x w.r.t. $\otimes$ is $x^{\otimes -1} = -x$. There is no inverse element for ε, since ε is absorbing for $\otimes$. If $r > 0$ then $\varepsilon^{\otimes r} = \varepsilon$. If $r < 0$ then $\varepsilon^{\otimes r}$ is not defined. In this paper we have $\varepsilon^{\otimes 0} = 0$ by definition.

The rules for the order of evaluation of the max-plus algebraic operators correspond to those of conventional algebra. So max-plus-algebraic power has the highest priority, and max-plus-algebraic multiplication has a higher priority than max-plus-algebraic addition.

Max-plus-algebraic matrix operations

The basic max-plus-algebraic operations are extended to matrices as follows. If $A, B \in \mathbb{R}_\varepsilon^{m \times n}$ and $C \in \mathbb{R}_\varepsilon^{m \times p}$ then:

$$(A \oplus B)_{ij} = a_{ij} \oplus b_{ij} = \max(a_{ij}, b_{ij})$$

$$(A \otimes C)_{ij} = \bigoplus_{k=1}^{n} a_{ij} \otimes b_{ij} = \max_{k}(a_{ik}, b_{ki})$$

for all i, j. Note the analogy with the definitions of matrix sum and product in conventional linear algebra.

The matrix $\mathcal{E}_{m\times n}$ is the *m*×*n* max-plus-algebraic zero matrix: $(\mathcal{E}_{m\times n})_{ij} = \varepsilon$. for all *i*, *j*; and the matrix E_n is the *n*×*n* max-plus-algebraic identity matrix: $(E_n)_{ii} = 0$ for all *i* and $(E_n)_{ij} = \varepsilon$ *i*, *j* with $i \neq j$. If the size of the max-plus-algebraic identity matrix or the max-plus-algebraic zero matrix is not specified, it should be clear from the context. The max-plus-algebraic matrix power of $A \in \mathbb{R}_{\varepsilon}^{n\times n}$ is defined as follows: $A^{\otimes 0} = E_n$ and $A^{\otimes k} = A \otimes A^{\otimes k-1}$ for *k* = 1,2, . . .

Model of processes

Investigation

The intend of this study is to show that, and discuss how, the process satisfying the above assumptions can be modelled in max-plus algebra, to determine the input vector *u*(k) for knowing values of *y*(k), evaluate the error between real and desired output and estimate cost of the resources.

Let *u*(k) = [u_1 , u_2, u_3]^T the input vector, *x*(k) = [x_1 , x_2 , x_3]^T the state vector and *y*(k) the model output. For each transition x_i, and u_i is associated an indicator x_i(k) and u_i(k) responsibly, which corresponds to the steps of the k'th firing of transition x_j (resp. u_j), and in the same way we have y(k). The state system in the max plus algebra is follow:

$$\begin{aligned} x_1(k) &= t_1 \otimes u_1(k) \quad \oplus \quad u_2(k) \quad \oplus \quad t_{22} \otimes x_2(k-1) \quad \oplus \quad t_1 \otimes x_1(k-1) \\ x_2(k) &= t_{21} \otimes x_1(k) \\ x_3(k) &= x_2(k) \quad \oplus \quad u_3(k) \quad \oplus \quad y(k-1) \end{aligned}$$

For example the k'th firing of transition x_1 (when start Ts1), must wait of t_1 units of time until that the k'th input data u_1 for task Ts1 is ready and k'th input data u_2 is given. Then, the linear equation of evolution in $\mathbb{R}_{\varepsilon}$ of this discrete event dynamic system is as follows:

$$x(k) = A_0 \otimes x(k) \quad \oplus \quad A_1 \otimes x(k\text{-}1) \quad \oplus \quad B_0 \otimes u(k) \quad \oplus \quad D_0 \otimes y(k\text{-}1) \tag{1}$$

Where

$$A_0 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ t_{21} & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon \end{bmatrix}, \quad A_1 = \begin{bmatrix} t_1 & t_{22} & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \end{bmatrix}, \quad B_0 = \begin{bmatrix} t_1 & e & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & e \end{bmatrix}, \quad D_0 = \begin{bmatrix} \varepsilon \\ \varepsilon \\ e \end{bmatrix}.$$

A solution of (1), is

$$x(k) = A_0^* \otimes (A_1 \otimes x(k\text{-}1) \quad \oplus \quad B_0 \otimes u(k) \quad \oplus \quad D_0 \otimes y(k-1))$$

Where

$$A_0^* = E \oplus A_0 \oplus A_0^2 \oplus A_0^3 \ldots$$

$$A_0^* = \begin{bmatrix} e & \varepsilon & \varepsilon \\ t_{21} & e & \varepsilon \\ t_{21} & e & e \end{bmatrix}, \qquad A = A_0^* \otimes A_1 = \begin{bmatrix} \varepsilon & t_{22} & \varepsilon \\ \varepsilon & t_{22} \otimes t_{21} & \varepsilon \\ \varepsilon & t_{22} \otimes t_{21} & \varepsilon \end{bmatrix}.$$

$$B = A_0^* \otimes B_0 = \begin{bmatrix} t_1 & e & \varepsilon \\ t_1 \otimes t_{21} & t_{21} & \varepsilon \\ t_1 \otimes t_{21} & t_{21} & e \end{bmatrix}, \qquad D = A_0^* \otimes D_0 = \begin{bmatrix} \varepsilon \\ \varepsilon \\ e \end{bmatrix}.$$

Then, the model becomes:

$$x(k) = A \otimes x(k\text{-}1) \quad \oplus \quad B \otimes u(k) \quad \oplus \quad D \otimes y(k-1)) \tag{2}$$

By recurrence we obtain the expression:

$$x(k) = A_0^{k-1} \otimes x(1) \quad \oplus \quad \sum_{i=2}^{k} A_0^{k-1} \otimes B \otimes u(i) \quad \oplus \quad \sum_{i=1}^{k-1} A_0^{k-i-1} \otimes D \otimes y(i) \tag{3}$$

To determine the command for the process, we have to define at first the whole order-l model, which describes its global behaviour:

$$\begin{cases} x(k) = A \otimes x(k\text{-}1) \oplus \ B \otimes u(k) \oplus D \otimes y(k-1)) \\ y(k) = C \otimes x(k) \\ u(1) = \left[u_1(1) \quad u_2(1) \quad u_3(1)\right]^{\mathrm{T}} \\ x(1) = \left[x_1(1) \quad x_2(1) \quad x_3(1)\right]^{\mathrm{T}} \end{cases} \tag{4}$$

Where $C = \left[\varepsilon \quad \varepsilon \quad t_3\right]$, $u(1)$ and $x(1)$ are the initial conditions that we are going to determine below.

Initials conditions

To respect the previous assumptions, we determine the initial values of $u(1)$ and $x(1)$ according to the end process $y(1)$. We suppose that initially there is at least one freight vehicle ready for departure:

$$y(1) = t_1 \otimes t_{21} \otimes t_3 \otimes u_1(1) = t_{21} \otimes t_3 \otimes u_2(1) = t_3 \otimes u_3(1)$$

Then the initial control is:

$$\begin{bmatrix} u_1(1) \\ u_2(1) \\ u_3(1) \end{bmatrix} = \begin{bmatrix} y(1) \phi (t_1 \otimes t_{22} \otimes t_3) \\ y(1) \phi (t_{22} \otimes t_3) \\ y(1) \phi (t_1 \otimes t_{22} \otimes t_3) \end{bmatrix}. \tag{5}$$

Where "ϕ" means "–", in $\square_{\varepsilon}$.

Consequently, the initial value of the state vector is

$$\begin{bmatrix} x_1(1) \\ x_2(1) \\ x_3(1) \end{bmatrix} = \begin{bmatrix} t_1 \otimes u_1(1) \\ t_{21} \otimes u_2(1) \\ u_3(1) \end{bmatrix}. \tag{6}$$

These two initial vectors mean that, if the task Ts1 begins for instance at $t = 0$, t_1 units time later, the data for task Ts2 (u_2) is need be prepared and this task begins. This ensures a good starting up without delay for the evolution of the model.

Procedure

As indicated first, the network operates under a schedule defined for final result; according to this schedule we find the suitable inputs of the model. We formulate the model output more explicitly as:

$$y(k) = C \otimes A_0^{k-1} \otimes x(1) \quad \oplus \quad \sum_{i=2}^{k} C \otimes A_0^{k-1} \otimes B \otimes u(i) \quad \oplus \quad \sum_{i=1}^{k-1} C \otimes A_0^{k-i-1} \otimes D \otimes y(i) \tag{7}$$

or

$$y(k) \geq \max\{C \otimes A_0^{k-1} \otimes x(1), \quad \sum_{i=2}^{k} C \otimes A_0^{k-1} \otimes B \otimes u(i), \quad \sum_{i=3}^{k-1} C \otimes A_0^{k-i-1} \otimes D \otimes y(i)$$

which is equivalent to:

$$y(k) \geq C \otimes A_0^{k-1} \otimes x(1) \tag{8}$$

$$y(k) \geq \sum_{i=2}^{k} C \otimes A_0^{k-1} \otimes B \otimes u(i) \tag{9}$$

$$y(k) \geq \sum_{i=3}^{k-1} C \otimes A^{k-i-1} \otimes D \otimes y(i) \tag{10}$$

We are interested rather in the second in equation (9) that we transform on its equation form:

$$y(k) = \sum_{i=2}^{k} C \otimes A^{k-1} \otimes B \otimes u(i) \tag{11}$$

It is straightforward now that from (11) we can formulate the command $u(k)$ for process if the values of the output $y(k)$ are known. For all the rest y(k) will be the desired of the final result.

For k = 2, 3, 4,...:

$$\begin{aligned} y(2) &= C \otimes B \otimes u(2) \\ y(3) &= C \otimes A^1 \otimes B \otimes u(2) \quad \oplus \quad C \otimes B \otimes u(3) \\ y(4) &= C \otimes A^2 \otimes B \otimes u(2) \quad \oplus \quad C \otimes A^1 \otimes B \otimes u(3) \quad \oplus \quad C \otimes B \otimes u(4) \\ &\ldots \end{aligned} \tag{12}$$

Since our aim is to compute $u(k)$ for specified $y(k)$, we have to solve the following equation:

$$y(k) = C \otimes B \otimes u(k) \tag{13}$$

For example, to calculate $u(2)$, a solution of $y(2) \geq C \otimes B \otimes u(2)$, we resolve its equation form (12) and we keep its smallest solution to be sure that it is verifies also the in equation. Note here that we proceed by a simplification of the terms such "$C \otimes A^2 \otimes B \otimes u(2)$ .... $C \otimes A^1 \otimes B \otimes u(k-1)$" in the expression of y(k). Indeed, these terms constitute the first condition of the desired output y(k). More explicitly, if we consider that all expression $y(k)$ are equal to $C \otimes B \otimes u(k)$, we must then assure that:

$$C \otimes A^2 \otimes B \otimes u(2) \oplus \ldots \oplus C \otimes A^1 \otimes B \otimes u(k-1) \leq C \otimes B \otimes u(k)$$

Which means that we must have?

$$y(k) \geq t_1 \otimes t_{21} \otimes t_3 \otimes u_1(k-1) \quad \oplus \quad t_{21} \otimes t_3 \otimes u_2(k-1) \quad \oplus \quad t_3 \otimes u_3(k-1)$$

Periodic processing

We assumed that we wish the output of final results of the process be once every time interval T_c. We shall see later how the network reacts according of various value of T_c.

Let $y(k) \geq T_C^k \otimes y(0)$ where T_C is the periodicity of desired output. We use in our computations this condition in the following form:

$$T_C \geq \max(\, t_1 + t_{21} + t_3 + u_1(k-1),\ \ t_{21} + t_3 + u_2(k-1),\ t_3 + u_3(k-1)) / k$$

Solve the equations (11) and determine the control vector as follows:

$$u_j(k) = y(k)\phi(C \otimes B)_{i,j}; \qquad k = 2,3,\ldots; \quad i = 1 \ \ and \ \ j = 1,2,3.$$

Where:

$$C \otimes B = \begin{bmatrix} \varepsilon & \varepsilon & t_3 \end{bmatrix} \otimes \begin{bmatrix} t_1 & e & \varepsilon \\ t_1 \otimes t_{21} & t_{21} & \varepsilon \\ t_1 \otimes t_{21} & t_{21} & e \end{bmatrix} = \begin{bmatrix} t_1 \otimes t_{21} \otimes t_3 & t_{21} \otimes t_3 & t_3 \end{bmatrix}$$

More explicitly, for every $k \in N / \{1\}$, the general solutions are:

$$\begin{aligned} u_1(k) &= y(k)\phi(C \otimes B)_{1,1} \\ u_2(k) &= y(k)\phi(C \otimes B)_{1,2} \\ u_3(k) &= y(k)\phi(C \otimes B)_{1,3} \end{aligned} \tag{14}$$

These equations determine the appropriate control of the modelled process. On the other hand, (8) and (10) contain two constraints for the desired outputs of the system. While T_C is the periodicity of these outputs, $y(k) \geq T_C^k \otimes y(1) = T_C^k$ the first constraint which ensues from (8) is:

$$y(k) = T_C^k \geq \ C \otimes A^{k-1} \otimes x(1) \quad \text{or} \quad T_C^k \geq \ t_3 \otimes (t_1 \otimes t_{21})^{k-1} \otimes x_2(1)$$

In practice, $T_C \geq \ (t_2 + (t_1 + t_{21})(k-1) + x_2(1)) / k$ means that periodicity must be superior at certain value in order to have a good control.

The second constraint of T_C becomes from (10):

$y(k) = T_C^k \geq \sum_{i=3}^{k-1} C \otimes A^{k-i-1} \otimes D \otimes y(i)$, this constraint is verified all time since the product $C \otimes A^{k-i-1} \otimes D$ is null. To conclude this section we describe the preceding steps in the following algorithm:

1- Determine the state equations in max plus algebra as (4).

2- Calculate the global recurrence equation of the linear evolution of the system.

3- Determine initials conditions.

4- Calculate constraints of desired model output.

5- Calculate control vector using max plus algebra operations.

Simulation and Results

For the simulation of the model of process we use a fixed interval T_C for desired outputs, we have then: $y(k) = y(k-1) \otimes T_C; \quad k = 2,3,\ldots$ .Using (14), we can compute the vectors $u(k)$ and we consider the initial values. Interested to the state of the places C12, C23 (Figure 2) which represents intermediate buffered data and

cost of its maximal length and calculate the error between wished outputs and real outputs. Figures 3 - 6 show time evolution interpolated values of count of processes - particularly finished last tasks of following processes and generated input tasks appointed by signs blue ▼ and red ● respectively.

All the following examples results are obtained for various values of T_C and t_1 (time of operations of task Ts1). In first example, computations have periodicity equal to 10. This value is enough large to contain all tasks of process and no error between wished and real outputs occur. Some differences in the following examples, can be reduced only by increasing resources - processors and / or memory units. Figures 8 and 9 show the time dependence of markers in places that represent changes in the allocation of memory (buffers).

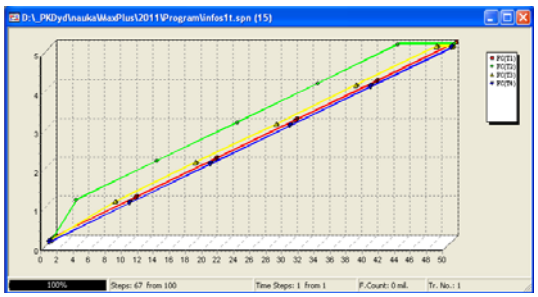

Fig. 3. TC=10 , t1 = 3

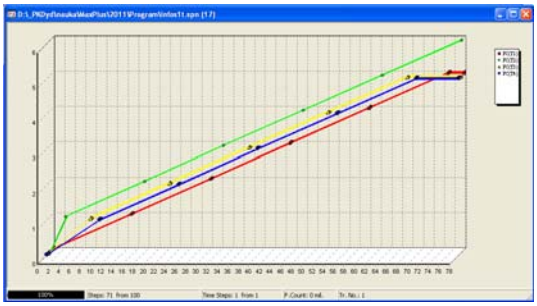

Fig. 4. TC=15 and t1=3

Fig. 5. TC=5 and t1=3

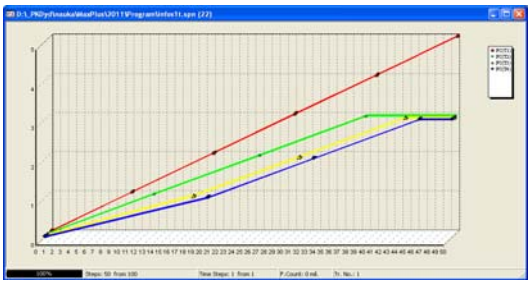

Fig. 6. TC=10 and t1=13

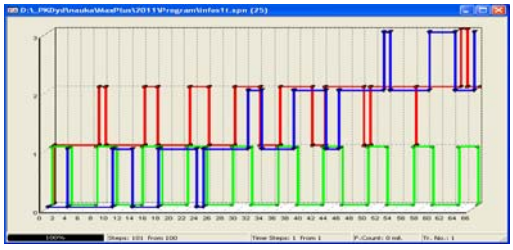

Fig. 7. T_C=7, Places P7, ▼ P8,●

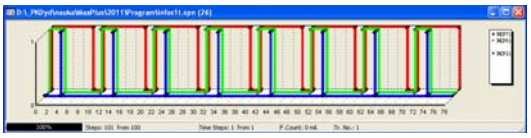

Fig. 8. T_C=10 and t_1=3, Places P7, ▼ P8,●

Conclusion

Engineers who build discrete event systems have to confront dynamical problems as a matter of course. For the most part, they have had little mathematical support to do this, despite the considerable understanding of dynamical systems arising from classical methods. This article proposed and introduced max plus algebra - a new methodology which used to modelling and simulating discrete event processes. A control of tasks of process on simple multi-processors computational system is using as an example. This is only a part of the carried out studies, which require additional testing and even wider range of experience, especially practical applications. Further more, have to develop a way to control this processes and studied conditions for periodicity of the required results. An other research will include the application to larger models and improvements of the

optimisation procedure with respect to its efficiency. Specially topic for future research include all over methods of designing, synthesis controls of processes with output and/or state feedback and using models of predictive and adaptive control.

Bibliography

[Bacceli Cohen 1992] Bacceli F.L.,Cohen G., Olsder G. J., Quadrat J.P.: Synchronization and Linearity. An Algebra for Discrete Event Systems, London, John Wiley & Sons Ltd, 1992.

[Balduzzi 2000] F. Balduzzi, Has. Giua and G. Menga. "First-order hybrid Petri net: In model for optimisation and control" IEEE trans. On Rob. And Aut. 1 6 (4):382-399,2000.

[Cassandras 2007] Cassandras Ch.G., Lafortune St.: Introduction to Discrete Event Systems Springer 2008, Kluwer Academic Publishers 2007

[Elmahi 2004] Elmahi I., Grunder O., Elmoudni A.: A max plus algebra approach for modeling and control of lots delivery. Industrial Technology, 2004. IEEE ICIT '04. 8-10 Dec. 2004 p. 926- 931 Vol. 2

[Goverde, Rob 2007] Goverde Rob M.P.: Railway timetable stability analysis using max-plus system theory, Elsevier, Transportation Research Part B 41, 2007, p. 179–201

[Jamroż, Raszka 1997] Jamroż L., Raszka J.: Simulation method for the performance evaluation of system of discrete cyclic processes. Proceedings of the 16-th IASTED International Conference on Modelling, Identification and Control, Innsbruck, Austria, 17-19.02.97, p. 190-193

[Maia, Andrade 2011] C.A. Maia, C.R. Andrade and L. Hardouin On the control of max-plus linear system subject to state restriction Automatica, Volume 47, Issue 5, May 2011, Pages 988-992 C.A. Maia, C.R. Andrade and L. Hardouin

[Murata 1989] Murata T.: Petri nets: properties, analysis and applications. Proceedings of the IEEE, vol. 77, no. 4, p.541-580, 1989.

[Gutuleac 2006] Emilian Gu¸tuleac Descriptive Timed Membrane Petri Nets for Modelling of Parallel Computing International Journal of Computers, Communications & Control, Vol. I (2006), No. 3, pp. 33-39

[Nait-Sidi-Moh, Manier 2009] A Nait-Sidi-Moh, M -A Manier, A El Moudni; Spectral analysis for performance evaluation in a bus network European Journal of Operational Research Volume 193: Issue 1. Pages 289-302 16 February 2009

[Schutter, Boom 2001] De Bart De Schutter, Ton van den Boom Model predictive control for max-plus-linear discrete event systems. Automatica 37(7):1049-1056, July. 2001

Authors' Information

Jerzy Raszka *PhD, Institute of Computer Science, Faculty of Physics, Mathematics and Computer Science, Tadeusz Kościuszko Cracow University of Technology Warszawska 24 St., 31-155 Kraków*

tel.:+48 12 628 27 08, *e-mail: jraszka@mail.pk.edu.pl*

Lech Jamroż – *PhD, Institute of Computer Science, Faculty of Physics, Mathematics and Computer Science, Tadeusz Kościuszko Cracow University of Technology Warszawska 24 St., 31-155 Kraków*

tel.:+48 12 628 27 08, e-mail: ljamroz@mail.pk.edu.pl