ELECTRONICS AND ELECTRICAL ENGINEERING
ISSN 1392 – 1215 *2010. No. 3(99)*
ELEKTRONIKA IR ELEKTROTECHNIKA

AUTOMATION, ROBOTICS
T 125
AUTOMATIZAVIMAS, ROBOTECHNIKA

A Matlab/Simulink Framework for the Design of Controllers and Observers for Discrete-Event Systems

R. Campos-Rodriguez, M. Alcaraz-Mejia
Centro Universitario de la Cienega, Universidad de Guadalajara,
Av. Universidad 1115, Col. Linda Vista, C.P. 47820, Ocotlán, Jalisco, Mexico, pone: +52(392)9259400, x8369,
e-mail: rr_campos, mildrethiam@hotmail.com

Introduction

The Discrete-Event Systems (DES), such as mass production systems, communicating processes, and digital networks, has evolved into highly specialized systems. Issues such as fault tolerance, reconfiguration, adaptation and control under partial state observation are very important whithin this field. Nowadays, many DES are controlled by automated systems based on events. The design of controllers for DES has been addressed in different ways. For example, the works in [1], consider a finite automaton (FA) for modeling the system and a sublanguage of the FA for representing the specifications. The controllability and observability notions there presented induce partial order relations that partition the automata language into equivalence classes. The approach considers that a specification is feasible to control if the classes induced by the observability are finer than those induced by the controllability. In other works, as in [2], the authors translate the same approach presented in [1] to the framework of the vector-additive system (VAS). The model of the plant is a VAS and a set of linear inequalities represent the specifications. The conjunctions of the restrictions define the case of multiple specifications on the system, which is visualized as a hyper-volume on a hyper-Cartesian space. The observability notion establishes that if two different states produce the same output, then they must require the same control action. This is basically the same approach as for the FA framework. The Control by Monitor Places (CM), as in [3], uses Petri Net (PN) models for representing the system. Similarly as in the FA and VAS frameworks, a set of linear inequalities defines the specifications for the system. These restrictions correspond to the maximum number of tokens that a specified set of places must retain in any evolution of the PN. The solution of the control problem consists on inducing a conservative component on the closed-loop system. A proper set of places and control arcs attached to the transitions in the net allows for the construction of such a conservative component on the PN model. Within this framework, in [4] the authors propose a PN observer for providing the feedback information for the controller.

This work presents a novel framework for the design of controllers and observers for Discrete-Event Systems (DES). The framework implements several Matlab functions embedded on Simulink models for the simulation of a control scheme. The framework mainly considers a discrete-event system, observer and controller. These elements interact for solving the problem that arises when the system state is not completely available for feedback. This paper organizes as follows. The next section presents the modeling methodology based on Petri Nets. Then, the paper introduces the controllability and observability notions. After that, the paper presents the problem of the control under partial state observation and the simulation framework. An illustrative example and concluding remarks are at the end of the paper.

Modeling DES with Petri Nets

The framework herein proposed uses Interpreted Petri Net (IPN) models. An IPN is a pair (Q, M_0), where $Q = \{G, \Sigma, \lambda, \phi\}$, whose members are: $G = (N, M_0)$ is a Petri Net System; $\Sigma = \{\alpha_1, \ldots, \alpha_r\} \cup \{\varepsilon\}$ is an input alphabet, every α_i is an input symbol and ε the null input symbol; $\lambda: T \to \Sigma$ is a labeling function for transitions; $\varphi: R(G, M_0) \to (Z^+)^q$ is an output function relating a marking in $R(G,M_0)$ to a vector in $(Z^+)^q$, where q is the number of available outputs; M_0 is the initial marking for G. This work considers that the functions φ and λ are linear, allowing their representation as matrixes $\varphi_{q\times m}$ and $\lambda_{r\times n}$, respectively. Roughly speaking, the $i-th$ row vector $\varphi(i,:)$ represents the places associated to the $i-th$ output signal, where q and m are the number of output signals and the number of places in the net, respectively. Similarly, the $j-th$ row vector $\lambda(j,:)$ represents the transitions associated to the $j-th$ input symbol, where r and n are the number of input signals and the number of transitions in the net, respectively. By definition of φ, a

place $p_i \in P$ is *measurable* whenever $\varphi(:, i) \neq \vec{0}$, otherwise *non-measurable*. These concepts allows for defining the observability problem. Similarly, by the definition of λ, a transition $t_i \in T$ is *manipulable* if $\lambda(t_j) \neq \varepsilon$, otherwise *non-manipulable*. This allows for defining the controllability problem. Pictorially, circles represent the places in the net and rectangles the transitions. Arcs represent the flow functions. Gray-filled bars and circles represent non-manipulable transitions and non-measurable places, respectively. Capital letters and Greek symbols, closed to transitions and places, represent the input and output signals, respectively. The IPN state equation captures the dynamic behavior of an IPN:

$$M_{k+1} = M_k + C \cdot \vec{t_j} \quad , \quad y_{k+1} = \varphi \cdot M_{k+1}. \qquad (1)$$

Some illustrative surveys in PN theory and related topics are in [5] and [2].

Controllability on IPN

The Linear Constrains defines the specification for the system under control, as defined in [2] and [3]. A Generalized Linear Constrain (GLC) is an inequality of the form $q_{lwr} \leq l \cdot X \leq q_{upp}$, where the positive integers q_{lwr} and q_{upp} represents lower and upper bounds, respectively. The conjunction of GLC's allows for the representation of multiple specifications on a system:

$$\vec{q}_{lwr} \leq LX \leq \vec{q}_{upp} \coloneqq \bigwedge_{i=1}^{n} \left(q^i_{lwr} \leq l_i X \leq q^i_{upp}\right). \qquad (2)$$

In (2), $\vec{q}_{lwr}$ and $\vec{q}_{upp}$ are vectors and L is a matrix whose rows represent single GLR's. The vector X stands for the set of places in the net. From the point of view of an IPN, a GLR represents boundaries on the number of tokens that places must retain at any evolution of the net.

A solution for (2) consists on a set of control places attached to the transitions that fill or drain tokens on the buffers of the system. The closed-loop system is:

$$D = \begin{bmatrix} C \\ -L \cdot C \\ L \cdot C \end{bmatrix} \quad , \quad \breve{M}_0^D = \begin{bmatrix} M_0 \\ \vec{q}_{upp} - L \cdot M_0 \\ L \cdot M_0 - \vec{q}_{lwr} \end{bmatrix}. \qquad (3)$$

In (3), $-LC$ and LC defines two conservative components in the controlled system that warranties the fulfillment of (3). However, the controller requires the system state as feedback, which is a problem if some sensors in the system are unavailable.

Observability on IPN

There exist several approaches dealing with the observability problem in the PN framework as in [3], [7], [8], [9] and [10]. This section follows the approach presented in [7], specially focusing on the case of State Machines (SM). Informally, an IPN (Q, M_0) is *observable* if there exists a finite integer $k < \infty$ such that for every evolution of the net, longer than k, the information of the input and output signals in the net, and the structure of (Q, M_0) suffice to uniquely determine the initial marking M_0 and the current marking M_k. The numerical value of k is known as the *observability convergence constant* of (Q, M_0).

Typically, the verification of this property is a difficult work, since it requires the verification of all the transition sequences in the net. However, the problem has efficient solutions under structural assumptions. The first assumption is the *Firing-Vector-Detectability*, a property that allows for reconstructing the Firing Vector of a net. The second is the *Marking-Detectability*, a property that allows for the reconstruction of the current marking reached by the net. In [7], a theorem shows that the *Firing-Vector-Detectability* and the *Marking-Detectability* implies the observability. Moreover, that work also shows that for the case of a strongly connected and safe SM, the former property implies the later one, and consequently, the observability reduces to *Firing-Vector-Detectability*.

Finally, the work in [7] proposes a stronger property, the *Sequence-Detectability*, as a way to verify the *Firing-Vector-Detectability*. The Sequence-Detectability allows for determining the transition sequence executed in a net. An intersection of vector spaces provides a sufficient condition for the Sequence-Detectability in a safe SM:

$$ker(\varphi C^-) \cap ker(\varphi C^+) \cap span(bT) = \vec{0}. \qquad (4)$$

Roughly, $ker\,(\varphi C^-) \cap ker\,(\varphi C^+)$ represents those transitions in the system that has the same input and output sensors. Similarly, $span(bT)$ represents the Parikh vectors of the T-Invariants of the system. Consequently, (4) represents the set of T-Invariant of the system that has the same output signals. The T-Invariants in a SM may execute forever, confusing to each other, which implies non-observability. Thus, if the intersection of the vector spaces is null, the SM is sequence-detectable, and consequently, observable. The intersection of the vector spaces in (4) is easily performed by polynomial algorithms.

Control under Partial State Observation

Any observer carries an estimation error during its convergence period that may lead to the controller to guide the system to a prohibited configuration. There exist several techniques to cope with this situation. By the one hand, there exist defensive approaches, as in [2], [3] and [4], which imply disabling all of the events in the system. However, this technique may lead to the control scheme into a deadlock condition. On the other hand, there exist offensive approaches as in [11], which imply enabling all of the events in the system. By using this technique, the observer will detect the changes in the system outputs and consequently it will produce new estimations for the controller. This avoids deadlock conditions in the control scheme, but it may cause a violation of the especifications. Thus, extra restrictions considering the convergence constant of the observer solve this new situation.

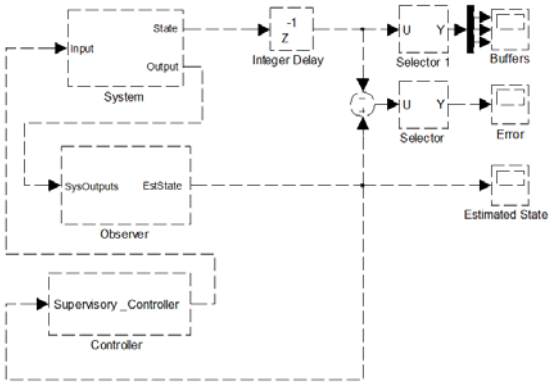

Fig. 1. Simulation framework

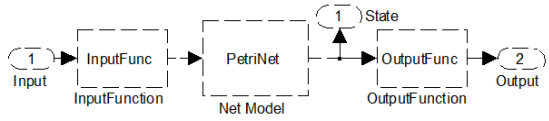

Fig. 2. Model for an IPN

As shown in [11], the problem of the control under partial state observation has a solution if the system satisfies the following conditions:

a) Every buffer p_i restricted by a GLR l_i, is filled in, and drained out, by sub-models with structures Q_{upp} and Q_{lwr}. The sub-models are observable with constants k^i_{upp} and k^i_{lwr}, respectively.

b) The initial marking $M_0(i)$ for buffer p_i fulfils the condition $q^i_{lwr} + k^i_{lwr} \leq M_0(i) \leq q^i_{upp} - k^i_{upp}$.

Roughly speaking, a) requires the verification of the observability of the subsystems that interacts with the buffers. The constant k^i_{upp} in b) defines the clearance that buffer i should have to handle the pieces that the controller can erroneously put due to false estimations of the observer. Similarly, k^i_{lwr} defines the mininum amount of pieces on the buffer i to avoid a violation in the lower bound due to lack of parts. If the system satisfies these conditions, the control scheme solves the problem. However, the initial marking of the system is unknown. Fortunately, the selection of any marking M_0 such that $\varphi \cdot M_0 = Y_0$ is suitable for the simulation process. Once M_0 was fixed, the initial marking M_{0_C} for the control places is defined accordingly to (3).

Simulation Framework

The Fig. 1 depicts the proposed simulation framework. The model includes the system, observer and controller. The system block provides a restricted set of outputs defined by the function φ. The observer uses the system outputs, and the structure of the system, for reconstructing the state. The controller uses the estimated estate for computing the required control actions. The overall goal of the control scheme is to satisfy the requirements even though the system state is incomplete. The System block implements the IPN state equation (1). It uses the current state, which is an internal variable, and the control pattern provided by the controller for computing the next state. The Fig. 2 depicts a simplified model for the system block, which includes sub-blocks for the input and output functions, as well as for the net structure. The block works as follows. By the one hand, the control pattern provided by the controller is a column-selector for the input function. That is, given a control pattern, the Input Function block provides to the Petri Net the transitions that were not disabled by the controller. On the other hand, the Output Function works as a filter that hides those states that are not accessible at the outside of the net. The separation of these functions into sub-blocks provides great flexibility, since the simulation of different input and output functions is easily performed by defining its matrices. Accordingly, this represents different arrangements of sensors and actuators on the system. This is a fast-prototyping capability of the framework.

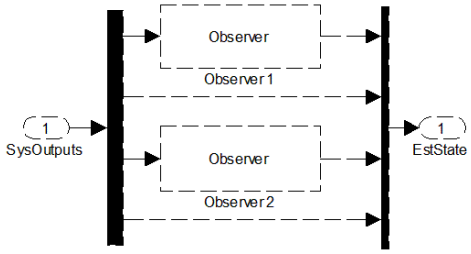

Fig. 3. Model for multiple observers

The Observer implements the algorithms for the construction of observers for IPN's following the sequence of events observed at the outputs of the system, as in [7]. The block uses the output of the net and its structure for reconstructing those states that are not directly measured by sensors. The Fig. 3 shows the model, which may include multiple instances of the Observer function. The overall block provides an integer vector that represents the estimated marking for the system model. For the particular framework herein presented, the marking estimations are maximal permissive in the sense that if the observer has a doubt about whether a place has a token or not, is assumes that indeed it has one. This marking estimation policy avoids blocking condition that otherwise may arise in the control scheme. However, such a policy is easily changed to meet the requirements in different control techniques.

The Controller implements the control policy. The framework herein presented uses supervisory controllers for guiding the system. Generalized Linear Restrictions represents the specification that the system must satisfy. These restrictions allows for defining a suitable work region for the system. The simulation framework is configured to have a discrete solver with fixed step of size one. Thus, one event may occur in every block at every second during the simulation.

Roughly speaking, the control scheme evolves as follows. The system produces a new output vector each time a transition fires. Their block computes its output as a multiplication of the current vector state and the matrix of the output function. Accordingly, the size of this vector is

the number of rows in the output matrix. The observer uses this output vector and provides an estimation of the system marking. The estimation is also given as a vector, which is generally a vector of bigger size than the system output. The block computes the estimations based on a technique presented in [7]. Accordingly to the observer policy, the marking of the system is generally overestimated until it converges to the real system state. This convergence occurs in at most the number of events in the observability convergence constant of the system. The supervisory controller uses the estimations produced by the observer and computes a control pattern that is a column vector which size is the number of rows in the input matrix. The last position of this vector is always one because it represents the null input symbol and, accordingly, the controller is unable to avoid it. This block simulates the firing of all the enabled transitions at the given estimated state. Then, it disables the firing of any manipulable transition whose firing violates the specification by avoiding its input symbol in the control pattern, closing the cycle of the control scheme.

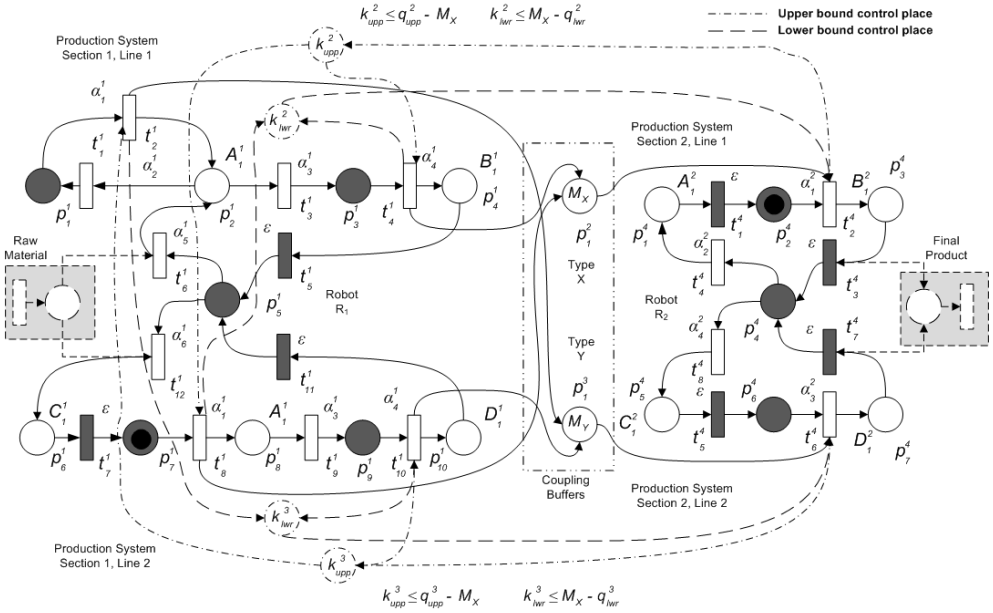

Fig. 4. Application example

As expected, the overall goal of the control scheme is to keep the specifications even when some fails in sensors measuring the system state occurred. As mentioned, this is accomplished by introducing new restrictions to the system buffers. The new restrictions allow for the observer to completely reconstruct the system state while avoiding a violation to the restrictions. Due to the overestimating policy used by the Observer, the system usually evolves freely during the convergence period of the observer. However, the additional restrictions warranty the fulfillment of the requirements. In this way, the blocks in the scheme evolve together for solving the problem of the control under partial state observation.

Application Example

The following example illustrates the IPN model for a production system. Similarly, it is possible to obtain models for communicating processes, queue systems, and other systems based on events.

The Fig. 4 depicts a system with two sections. The places from p_1^1 to p_{10}^1 and transitions from t_1^1 to t_{12}^1 represents the Section 1. Similarly, the places from p_1^4 to p_7^4 and transitions from t_1^4 to t_8^4 represents the Section 2. The buffers p_1^2 and p_1^3 couple together both sections. The robots R_1 and R_2, at the center of each section, move the raw material and semi-finished pieces among the lines. In Section 1, the production system processes the raw material from non-depleting inventory. The Line 1 may produce one semi-finished part Type Y by every loop at p_1^1 and p_2^1. Finally, the line produces one part Type X. The Line 2 produces one semi-finished piece of each type. In Section 2, the system further post-processes the semi-finished pieces Type X and Type Y for producing final product. The sensors for Section 1 and Section 2 are $\{A_1^1, B_1^1, C_1^1, D_1^1\}$ and $\{A_1^2, B_1^2, C_1^2, D_1^2\}$, respectively. These provide the output information for the observer. The input commands for Section 1 and Section 2 are $\{\alpha_1^1, \alpha_2^1, \alpha_3^1, \alpha_4^1, \varepsilon\}$ and $\{\alpha_1^2, \alpha_2^2, \alpha_3^2, \alpha_4^2, \varepsilon\}$, respectively. These allow the controller to guide the system evolution.

Due to restrictions on the system performance, the controller must hold the levels of buffers p_1^2 and p_1^3 between the bounds q_{upp}^2 and q_{lwr}^2, and q_{upp}^3 and q_{lwr}^3, respectively. Then the GLR of the required behavior is:

$$q_{lwr}^2 \leq p_1^2 \leq q_{upp}^2 \quad \wedge \quad q_{lwr}^3 \leq p_1^3 \leq q_{upp}^3. \quad (5)$$

The Fig. 4 shows the control structure as dot-dashed places and control arcs accordingly to (3). The control places stop

the firing of the transitions that otherwise may violate the buffer bounds. The initial load and the required bounds on the buffers define the initial marking for the control places, as remarked by inequalities shown at the top and bottom in the Fig. 5. The intersection of the restrictions in (5) defines a closed region depicted as a big rectangle in Fig. 5. The objective of the controller is to maintain the system state within that region.

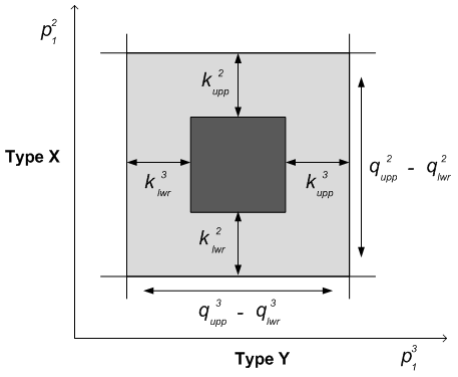

Fig. 5. Specification for system buffers

The simulated scenario considers that multiple sensors in the system failed. The gray-filled places in Fig. 4 represent the faulty sensors. In the Section 1, five sensors failed, while in Section 2, three sensors failed. At the time that the sensors failed, the workflow of the Section 1 was at the place p_7^1, while the workflow at the Section 2 was at place p_2^4. However, the controller is unable to know that. In order to avoid the violation of the restrictions imposed to the buffer, a set of additional conditions must meet. The new conditions has to consider the observability convergence constant of the observers of the different sections on the system. In the Fig. 5, the constant k^2_{upp} inside the big rectangle defines the clearance that buffer p_1^2 should have to handle the pieces Type X that the controller erroneously can place due to false estimations. Similarly, the constant k^2_{lwr} defines the mininum amount of pieces of Type X that the buffer must have in order to avoid a violation in the lower bound due to lack of parts. Simmetricaly, the constants k^3_{upp} and k^3_{lwr} represent the additional restriction for buffer of pieces Type Y. If the initial state of the system is inside the inner rectangle of Fig. 5, then the scheme solves the problem of the control under partial state observation. The Fig. 6 shows the buffer levels during a simulation in the framework for the case of the restrictions $3 \leq p_1^2 \leq 10$ and $3 \leq p_1^3 \leq 10$. The initial load of pieces Type X and Y on the buffers is $M_X = 6$ and $M_Y = 6$. The observability convergence constants for the IPN representing Section 1 and Section 2 are three and one, respectively. Consequently there exist initial markings that solve the control problem within the control scheme.

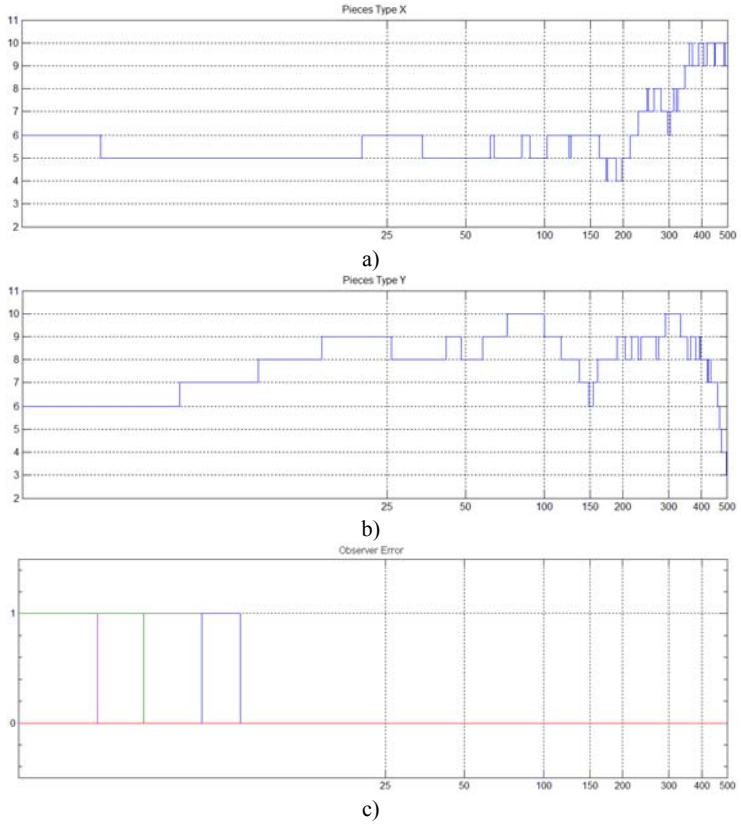

Fig. 6. Buffer levels and observer error

The **Fig. 6** shows that the observer estimations carry an error during a short period at the beginning of the

simulation. However, buffer levels fulfill the restrictions imposed by the GLR as shown. Notice that the level of the buffers for pieces Type X and Type Y remain between the required levels even during the converges period of the observer. This confirms that the current observability convergence constants work for the required GLR and for the initial amount of pieces on the buffer. The framework allows for several simulations of faulty sensors providing useful information to cope with the most common scenarios in the real world.

By using the proposed framework, the designer in charge of the construction of controllers and observers for a given system, can obtain useful information. Additionally, the proposed framework allows for visualizing the effect of the tuned parameters on the buffers.

Conclusions

This paper presented a novel framework for the design of controllers and observers for discrete-event systems based on simulations. The framework considers three entities, the system, the controller and the observer interacting among them. This allows simulating the different parts of the solution for the problem of the control under partial state observation. The framework implements Matlab functions embedded into Simulink models for representing these important entities. The framework uses a fixed-step discrete-event solver for the simulations.

Acknowledgements

This work was partially supported by Subsecretariat for Higher Education and Scientific Research under the grant PROMEP/103.5/08/2919.

References

1. **Lin F., Wonham W. M.**. Observability of place/transition nets // Decentralized supervisory control of discrete-event systems. – Inf. Sci., 1988. – No. 44(3) – P.199–224.
2. **Li Y., Wonham W. M.** Control of vector discrete-event systems I. The base model // Automatic Control, IEEE Transactions on. – No. 38(8). – P. 1214–1227.
3. **Moody J. O., Antsaklis P. J.** Petri net supervisors for des with uncontrollable and unobservable transitions // Automatic Control, IEEE Transactions on. – No. 45(3). –P. 462–476.
4. **Giua A., Seatzu C.** Observability of place/transition nets // Automatic Control IEEE Transactions on. – No. 47(9). – P. 1424–1437.
5. **Desel J., Esparza J**. Free Choice Petri Nets, Vol. 1. – Cambridge University Press, 1st edition, 1995.
6. **Meda M. E., Ramirez A., Malo A.** Identification in discrete event systems // Systems, Man, and Cybernetics, 1998 IEEE International Conference on. – Vol.1. – P. 740–745.
7. **Campos-Rodriguez R., Dang P., Mireles J., Lewis Frank L.** Implementing a multiple specification regulation controller: a case of study // IEEE Int. Conf. On Systems, Man, and Cybernetics 2004, (SMC'04). – P.1849-1856.
8. **Rivera-Rangel I., Ramírez-Treviño A., Aguirre-Salas L. I. and Ruiz-León J. J.** Geometrical Characterization of Observability in Interpreted Petri Nets // Kybernetika,2005. – No. 41(5). – P.553-574.
9. **Ramírez-Treviño A., Rivera-Rangel I., Lopez-Mellado E.** Observability of discrete event systems modeled by interpreted petri nets // IEEE Transactions on Robotics and Automation, 2003. – No.19(4). – P.557—565.
10. **Aguirre-Salas L., Begovich O., Ramirez-Treviño A.** Observability in Interpreted Petri Nets using Sequence Invariants // 41st IEEE Conf. on Decision and Control, 2002. – P.3602-3607.
11. **Campos-Rodriguez R., Alcaraz-Mejia M., Mireles-Garcia J.** Supervisory control of discrete event systems by using observers // Proc. of 15th Mediterranean Conference on Control & Automation, 2007, Athens, Greece.
12. **Campos-Rodriguez R., Ramírez-Treviño A., López-Mellado L.** Observability Analysis of Free-Choice Petri Net Models // IEEE SMC International Conference on System of Systems Engineering 2006, (SoSE'06). – P. 77–82.
13. **Campos-Rodriguez R., Ramírez-Treviño A., López-Mellado L.** Regulation control of partially observed discrete event systems // IEEE Int. Conf. On Systems, Man, and Cybernetics, 2004, pages 1837-1842.

Received 2010 01 15

R. Campos-Rodriguez, M. Alcaraz-Mejia. A Matlab/Simulink Framework for the Design of Controllers and Observers for Discrete-Event Systems // Electronics and Electrical Engineering. – Kaunas: Technologija, 2010. – No. 3(99). – P. 63–68.

This paper presents a novel framework for the design of controllers and observer, based on simulations, for discrete-event systems. The framework mainly considers a discrete-event system, a discrete-event controller and a discrete-event observer. The structure of the control scheme implements proper Matlab functions capturing the behavior of these entities. A Simulink model uses the functions for simulating a closed-loop system where the controller uses the estimated state produced by an observer as feedback. The graphs produced by the simulations allows to the engineer for tuning important parameters of the system under study. Ill. 6, bibl. 14, tabl. 0 (in English; abstracts in English, Russian and Lithuanian).

Р. Кампос-Родригез, М. Алказар-Мейя. Анализ дискретных процессов контроля на основе функции „Матлаб“ // Электроника и электротехника. – Каунас: Технология, 2010. – № 3(99). – С. 63–68.

Описывается новая система наблюдается дискретных процессов. Система контроля построена на основе функции „Матлаб“ Полученные графики моделирования позволяют корригировать параметры системы, и позволяют определить обратные связи исследуемых процессов. Ил. 6, библ. 14 (на английском языке; рефераты на английском, русском и литовском яз.).

R. Campos-Rodriguez, M. Alcaraz-Mejia. Projektavimo priežiūros ir kontrolės diskrečiųjų įvykių analizė „Matlab/Simulink“ sistemos aplinkoje // Elektronika ir elektrotechnika. – Kaunas: Technologija, 2010. – Nr. 3(99). – P. 63–68.

Apžvelgta nauja sistema, skirta projektavimo priežiūros ir kontrolės diskretiesiems įvykiams stebėti. Sistema nusakoma kaip diskretinių įvykių stebėjimo sistema, diskretinių įvykių valdiklis ir diskretinių įvykių stebėsena. Kontrolės schema įgyvendinama taikant „Matlab“ funkcijas. „Simulink“ modeliai naudoja modeliavimo funkcijas grįžtamajam ryšiui kaip galima labiau kontroliuojant ir stebėti. Modeliavimo metu gauti grafikai leidžia inžinieriams koreguoti svarbius sistemos parametrus. Il. 6, bibl. 14 (anglų kalba; santraukos anglų, rusų ir lietuvių k.).