Agnieszka LASOTA[1], Andrei KARATKEVICH[1]

Uniwersytet Zielonogórski, Instytut Informatyki i Elektroniki (1)

Modification of Petri nets modeling production processes when quality control is introduced

Abstract. *Petri nets can be used as a model of production processes, as far as such processes often contain the steps which can be executed in parallel. Such modeling can serve for optimization and verification of production processes, because a wide range of existing methods of Petri net analysis can be applied. Adding to a production process the steps in which quality control is performed requires changes in the modeling net to keep it well-formed. Such modifications are the topic of the article. The production processes are modeled by s-nets and α-nets.*

Streszczenie. *Sieci Petriego, jako formalny model procesów współbieżnych, doskonale nadają się do optymalizacji i weryfikacji procesów produkcyjnych. Wykorzystane w tym celu metody redukcji oraz analizy sieci pozwalają zbadać czy sieć jest dobrze zbudowana a tym samym czy proces produkcyjny będzie poprawnie realizowany również po wprowadzeniu pośrednich punktów kontroli jakości w wyniku czego może wystąpić konieczność przeprowadzenia modyfikacji. Do odwzorowania rzeczywistych procesów produkcyjnych zastosowane zostały s i α - sieci.* ***(Modyfikacje sieci Petriego modelujących procesy produkcyjne po wprowadzeniu punktów kontroli jakości)****.*

Słowa kluczowe: sieci Petriego, α-sieci, s-sieci, proces produkcyjny.
Keywords: Petri nets, α-nets, s-nets, production process.

Introduction

Petri nets [1,2,3] can be used for modeling, analysis and verification of a wide range of parallel processes, especially asynchronous ones. Among others, they can be applied to formal verification of the production processes [4,5]. A correct production process should have no deadlocks, should never try to initialize again an operation which is already active, and should be cyclic. That means that a Petri net which models such process should be well-formed. There are various methods which allow to check whether a net is well-formed and, if it is not, to diagnose its defect.

When the quality control steps are introduced to the production process (which is usually an obviously necessary element of a production) it may complicate the process structure. Among others, modeling Petri net, which usually belongs to the class of extended free-choice nets or α-nets, after introducing the quality control steps may turn to be outside this class. Besides, introducing such quality control may cause errors in the production process, related to wrong or incomplete reaction to negative results of the control.

Our proposition to provide correctness of a production process is applying the following 4 steps.

1. Perform modeling (by a Petri net), verification and, if necessary, correcting of structure of the production process before quality control is added;
2. introduce the quality control steps to the process where it is necessary;
3. modify the modeling Petri net to keep it well-formed;
4. provide the corresponding modifications to the production process to keep it formally correct.

The main topic of this paper covers steps 3 and 4 of the mentioned ones.

State of the art

Petri nets were used for modeling production systems since year 1983 [6] until today. Paper [7] presents a visual verification approach and an algorithm that employs a set of graph reduction rules to identify structural conflicts in process models for a generic workflow modeling language. There we can find information about modeling and verification of production workflows. In [8] the method is presented which maps UML specifications to high-level Petri nets; it allows checking important properties of the modeled process. A different approach presented in [9], where we can see transformation from UML activity diagrams to stochastic Petri Nets (intended for formal analysis of software systems). Paper [10] indicates application of the activity diagrams to verification of the production processes.

Production processes

A production process is a procedure of obtaining finished goods from materials and components. It consists of the operations of several kinds. Technological operations create a new product, and the quality control operations verify its quality.

Petri Nets

A Petri net [2,3] is a triple $\Sigma = (P, T, F)$, where P and T are the disjoint sets of *places* and *transitions*; $F\subseteq(P\times T)\cup(T\times P)$. For $t\in T$ ${}^{\bullet}t =\{p\in P|(p,t)\in \mathrm{F}\}$; $t^{\bullet}=\{p\in P|(t, p)\in F\}$; ${}^{\bullet}t$ and $t^{\bullet}$ are the *sets of input* and *output places*, respectively. Such notation is also used for places.

A *marking* of a net is defined as a function M: $P\rightarrow\{0, 1, 2,\ldots\}$. It can be considered as a number of *tokens* situated in the net places. Number of tokens in a place p for marking M is denoted as $M(p)$. A place p such that $M(p) > 0$ is *marked* in marking M. Initial marking is denoted as M_0.

A transition *t* is *enabled* and can *fire* if all its input places are marked. Transition firing removes one token from each input place and adds one token to each output place of the transition. A marking that can be obtained from M by a firing sequence is called *reachable* from M; the set of reachable markings is denoted as $[M\rangle$. A net is *safe*, if $\forall M\in[M\rangle$, $\forall p\in[P\rangle$: $M(p)\leq 1$. A transition is *live*, if there is a reachable marking in which it is enabled; otherwise it is *dead*. A *deadlock* is a marking in which all transitions are dead. A net is *reversible* if, for each marking M in $[M\rangle$, M_0 is reachable from M [2]. A Petri net is *well-formed*, if it is live, safe and reversible.

Concurrency relation $||\subseteq P\times P$ is a symmetric relation on the set of places P of a Petri net such that $\forall p,p'\in P$ $(p,p')\in||$ iff $\exists M\in[M_0\rangle$ such that $M(p)>0$ and $M(p')>0$.

Graphically Petri nets are presented as oriented graphs with nodes corresponding to places and transitions and with arcs joining places to transitions or transitions to places. Tokens are shown as the dots inside places (Fig. 1). Behavior of a net can be presented by its *reachability graph*. A reachability graph is a labeled directed graph. Its nodes represent reachable markings of the net, and the arcs correspond to possible single transition firings transforming a marking represented in the graph into another marking represented in it.

In graphic presentation of the reachability graphs, we will describe the markings, listing the numbers of marked places. Listing a number twice in the same node means an unsafe marking (see Fig. 5).

An *s-net* is a Petri net with single-token initial marking [11,12]. An *α-net* [13] is such s-net that for every two its transitions t_1, t_2: $({}^\bullet t_1 \cap {}^\bullet t_2 \neq \emptyset) \Leftrightarrow ({}^\bullet t_1 = {}^\bullet t_2)$.

Modeling of production processes with Petri nets

In [4] it was proposed a way to connect two different models, i.e. UML and PNs, for modeling production processes. In this method modeling UML is transformed into a Petri net (where the places correspond to technological operations), which can be formally analyzed. An example of Petri net modeling a production process is shown in figure 1.

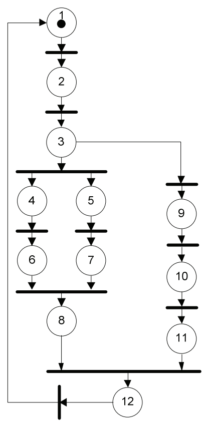

Fig. 1. A Petri net (an α-net which is not live)

Methodology of verification of production processes using Petri nets

Verification is an important part of designing the production processes. Modeling by Petri nets provides possibility of efficient formal verification of the processes, because there is a wide range of methods allowing checking properties of such nets. Among others, analysis of a Petri net can decide where it is well-formed, and a net modeling a correct production process should be well-formed.

The analysis methods which especially fit to the structure of production processes are, according to our experience, the stubborn set method constructing reduced reachability graphs (RRG) [14], the Andre's method of reduction of state machine subnets, described in [1], developed and extended in [15,16], and the classical reduction rules of Petri nets presented in [2]. The stubborn set method allows detecting all reachable deadlocks of general Petri nets [14], and checking some other important properties of α-nets and s-nets [11,12]. Net reduction in both mentioned variants preserves liveness and safeness of the nets [16,9]. A person performing modeling and verification should decide which method or combination of methods should be applied in a concrete case.

Example of net analysis (reduction approach)

Applying reduction methods to the α-net shown in figure 1 allows easy checking of its properties. Figure 2 demonstrates such checking. In figure 2a the net from figure 1 after reduction of the SM-subnets [16] is shown. Subnet specified by places p_1, p_2, p_3 is replaced by macroplace A; p_4, p_6 by B; p_5, p_7 by C; and p_9, p_{10}, p_{11} by D. Figure 2b shows the net after applying fusion of parallel places – one of classical reduction operations [2]; macroplaces B and C are replaced by macroplace X. Then we can apply reduction of an SM-subnet again: in figure 2c the net in demonstrated in which places A, X, p_8 are replaced by macroplace Z. Reachability graph of the reduced net is trivial (Fig. 2d), and it shows, that the net is not live and not reversible.

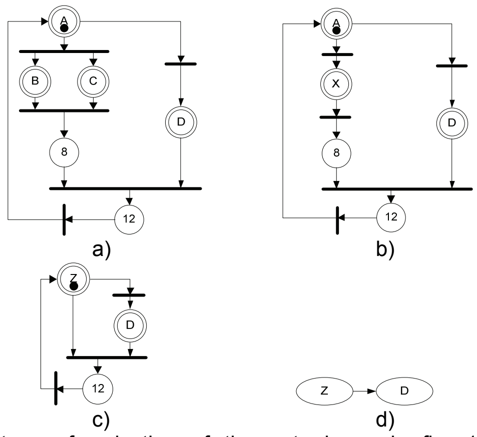

Fig. 2. Steps of reduction of the net shown in fig. 1 (a-c), reachability graph of the reduced net (d)

Application of the reduction methods for analysis of the production processes is limited by the fact that they usually do not allow to localize a fault easily, and such localization is very important for correction of the processes. For example, the reduced net from figure 2c evidently has a deadlock, but it is not so evident which deadlock corresponds to it in the original net (Fig. 1). When reduction approach causes such problems, applying only the stubborn set method seems to be more reasonable.

Steps of verification and transformation

The following algorithm presents a possible way of verification and modification of a production process using Petri net modeling.

Algorithm 1

1. Verify the α-net modeling the production process:
   a. Reduce the automaton subnets.
   b. Apply classical reduction rules, if necessary, otherwise skip this step.
   c. Construct a reduced reachability graph for the reduced net (by means of the stubborn set method).
   d. If the RRG is strongly connected[1], go to 3.
2. Perform an appropriate correction of the net and go to 1.
3. Introduce the intermediate quality control steps where it is necessary.
4. Verify the obtained s-net:
   a. Reduce the SM-subnets.
   b. Apply classical reduction rules, if necessary, otherwise skip this step.
   c. Construct a reduced reachability graph for the reduced net.
   d. Analyze the RRG. If the net is well-formed[2], go to 8.

[1] As it is shown in [12], RRG of an α-net is strongly connected, if and only if the net is well-formed.

[2] Analysis of s-nets is described in [11,12].

5. Perform modification of the net (Algorithm 2).
6. Verify the obtained s-net (as in item 4). If the net is well-formed, go to 8.
7. Perform correction of the net and go to 6.
8. Perform the modifications of the production process corresponding to the modifications of the net.
9. The end.

Methodology of Correction of Production Processes

There are several causes why modifications of a production processes may be necessary. First of all, it is necessary for optimization and acceleration of the process. Such modifications can be performed when the process is formally correct (and the modeling Petri net is well-formed). Another cause is correction of errors in the process structure. Third essential cause is introducing the intermediate steps of quality control into the process.

If the quality control is performed when the process is finished, it does not influence its structure. If it is performed *before* end of the process, the situation is different, especially if it is a local control operation in one of the parallel branches of the process. Then the structure of the modeling net should be modified and verified. Correspondingly, structure of the process may also require modifications.

Correction before quality control is introduced

Usual errors at the first steps of design of a production process are related to wrong way of beginning (Fig. 1) or ending (Fig 5) of the parallel processes. At the level of modeling net it may cause non-liveness or non-safeness of the net. Then a net can be corrected in two ways. The first one is introducing a common transition at the beginning (Fig. 3) or at the end (Fig. 6) of a concurrent section of the process. Common transition for the beginning of a set of parallel processes can be applied when all these processes can start at the same time. Another way of correcting is introducing, together with the common transition, the synchronization places (in figure 4 such places are S1 and S2). Adding the places for synchronization makes possible initialization of the parallel processes at different moments of time.

Number of the synchronization places depends on number of parallel processes. Decision whether to introduce such places should belong to the designer, because every case should be considered individually.

Modifications after introducing quality control

Petri net modeling a production process usually has a structure which situates it within the class of α-nets [17]. Introducing the quality control often causes the changes after which the modeling net belongs to the class of s-nets, but not to the class of α-nets any more.

When intermediate quality control steps are introduced, the feedbacks in the process structure should also be introduced. It is necessary for taking into account situations when the results of the quality control are negative and correction of the produced item or its part should be performed (see figures 7, 8).

There are three possible ways of dealing with negative result of quality control. First one is cancelling the whole process and transporting materials and parts to the warehouses (with possible restarting of the process). It is illustrated by place KJ in figure 7 and the transition from this place to the initial one (however such modification of the net makes it unsafe). The second way is repeating the previous operation, adding if necessary an "undo" operation, such as disassembling (places KJ2 and DE in Fig. 8).

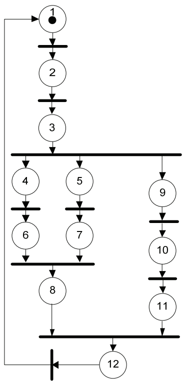

Fig. 3. Well-formed α-net being a modification of the net from figure 1 – a common transition for all parallel processes is used at their beginning

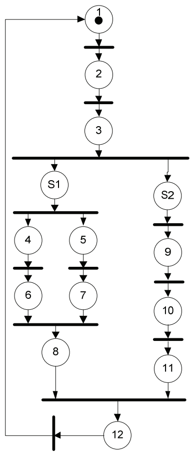

Fig. 4. Well-formed α-net being a modification of the net from figure 1 – a common transition for all parallel processes and the synchronization places

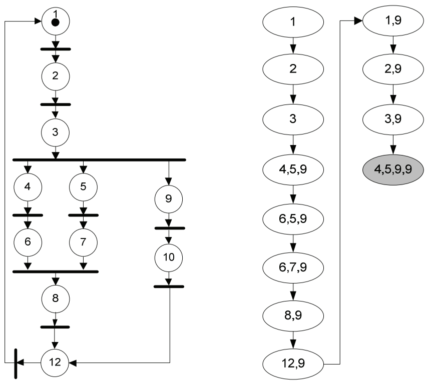

Fig. 5. Wrong structure of joining of the parallel processes and RRG of the net. The net is not safe (unsafe marking is grayed)

The third way is introducing a special operation for problem correction (it is not considered in this paper). Those ways can be combined; for example, figure 8 demonstrates a case in which results of quality control may cause continuation of the process, repeating an operation or restarting the whole process. As it can be seen in figures 7a and 8a, when the first way of dealing with negative results of quality control is applied without taking into account parallel processes, wrong structure of modeling net and of a production process can be obtained. But the structure can be corrected, as it is shown in figures 7b and 8b. Methodology of correction of the nets after introducing the intermediate quality control steps presents Algorithm 2.

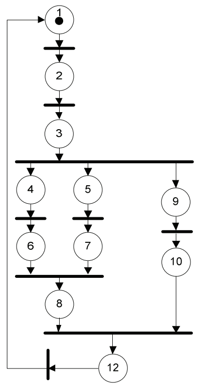

Fig. 6. Well-formed α-net being a modification of the net from figure 4 – a common transition for all parallel processes is used at their end

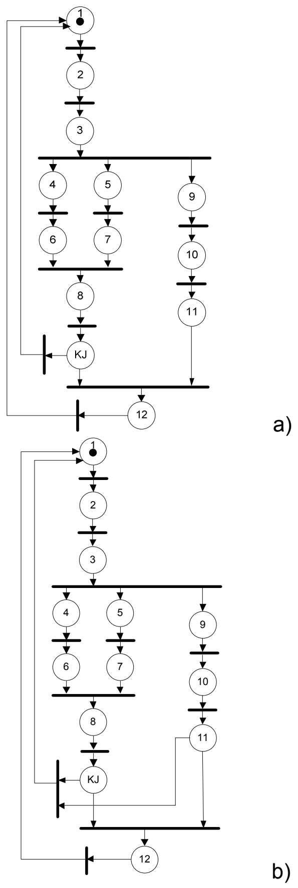

Fig. 7. The net from figure 2 after introducing an intermediate quality control step. a) unsafe net, b) well-formed net

It uses the fact that the modeling net before quality control is introduced is usually an α-net, and for the safe α-nets an efficient algorithm exists allowing to obtain concurrency relation on the set of places [17]. A modeling α-net before introducing the quality control (when item 3 of Algorithm 1 is achieved) is guaranteed to be safe.

Algorithm 2

1. Calculate the concurrency relation for the modeling α-net (before the quality control steps are introduced) using the algorithm from [17].
2. For every place p such that after introducing the quality control steps $(p^{\bullet})^{\bullet}=\{\mathrm{KJ}\}$, where KJ is a place corresponding to quality control step and there is transition t such that $\mathrm{KJ}\in{}^{\bullet}t$ and $p_1\in t^{\bullet}$, where p_1 is the initially marked place:
   - a. Calculate $S=\{p'\in P \mid (p,p')\in\|\}$.
   - b. If $S\neq\varnothing$:
     - i. Reduce S by repetitive applying the following operation: if $\exists p'\in S$: $(p'^{\bullet})^{\bullet}\in S$, then remove p' from S. When further reduction is impossible, go to ii.
     - ii. If a quality control step KJ' is introduced immediately after $p'\in S$ ($(p'^{\bullet})^{\bullet}=\{\mathrm{KJ'}\}$), then replace in S p' by KJ'.
     - iii. For every maximal set $S'\subseteq S$ such that $\forall p',p''\in S'$: $(p',p'')\in\|$, introduce transition t' such that ${}^{\bullet}t'=S'\cup\{\mathrm{KJ}\}$, $t'^{\bullet}=\{p_1\}$.
     - iv. Remove transition t.
3. The end.

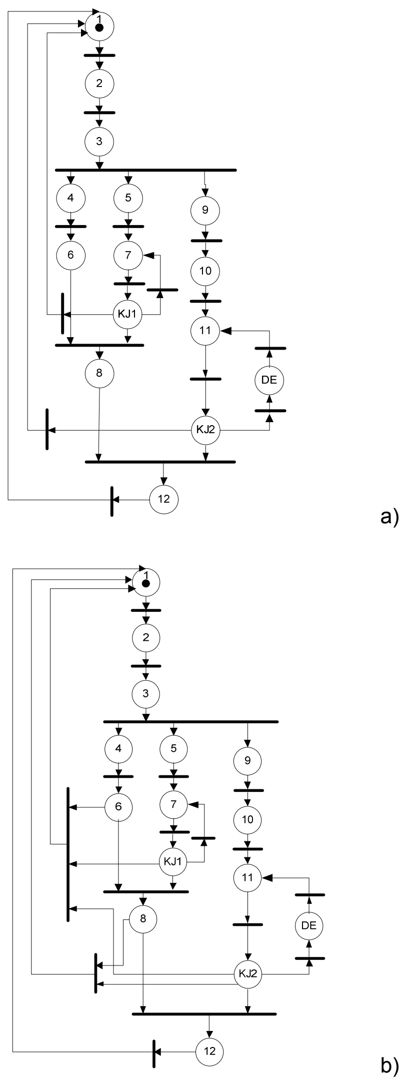

Fig. 8. The net from figure 2 after introducing the intermediate quality control steps with possibility of correction. a) unsafe net, b) well-formed net

Figures 3, 7, 8 illustrate the method. Consider figure 7a. Here one quality control step is introduced, but the net is modified in such a way that it became unsafe. To correct the net, Algorithm 2 can be applied. Then $p'=p_8$, S calculated in item 2a of Algorithm 2 is $\{p_9, p_{10}, p_{11}\}$. After reducing of S in item 2.b.i $S=\{p_{11}\}$. "Wrong" transition is replaced by the transition t' such that ${}^{\bullet}t'=\{\mathrm{KJ}, p_{11}\}$, $t'^{\bullet}=\{p_1\}$.

Figure 8 demonstrates introducing to the process of the same structure (Fig. 3) two quality control steps in other locations. The net in fig. 8a is not safe. Let us apply Algorithm 2. For KJ1 $p = p_7$, $S = \{p_4, p_6, p_9, p_{10}, p_{11}\}$. After reduction (item 2.b.i of Algorithm 2) $S = \{p_6, p_{11}\}$. As far as quality control step KJ2 is introduced immediately after p_{11}, in item 2.b.ii of Algorithm 2 S is modified: $S = \{p_6, \mathrm{KJ2}\}$. $S' = S$. This set is used as the set of input places of newly introduced transition.

But there is one more quality control step in the example presented in Fig. 8. For this second control step (at the second iteration of item 2) S before reduction is $\{p_4, p_5, p_6, p_7, p_8\}$, after reduction $S = \{p_8\}$, and new transition has places p_8 and KJ2 as the input ones.

Affirmation: if the modeling net before introducing quality control step was well-formed, and after such introducing it became unsafe, then applying Algorithm 2 to it changes it into a well-formed net.

Idea of the proof: introducing the quality control steps can result in an unsafe net only if a transition placing a token to the initially marked place is added (the whole process is restarted if the quality control result is negative). Then the net becomes unsafe if such transition does not gather all tokens from the net. In such case its firing leads to the marking in which initial place is marked together with some other places of the net; such net is evidently not safe. Introducing of quality control steps does not affect the concurrency relation between the places which were in the net before quality control is introduced. Then, taking into account such relation (calculated in item 1 of the Algorithm), we can construct a set of transitions which remove all the tokens from the places which is concurrent in respect to the quality control place (item 2.b.iii), avoiding possibility of unsafe makings.

If the modeling Petri net after introducing quality control is not well-formed, then production process is not correct. Correction of the Petri net should be followed by correction of structure of the production process.

Conclusions

Modeling of the production processes by Petri nets allows verifying and optimizing of the processes. A Petri net modeling a correct production process is supposed to be well-formed (live, safe and reversible). If the net does not satisfy the mentioned conditions, we may conclude that structure of the modeled process has some faults, which should be detected (it also can be done with the help of the Petri net model) and corrected. Introducing the intermediate quality control steps in a process complicates its structure and also may lead to incorrect behavior. The method proposed in this paper allows correction of the mistakes which may occur at different stages of process projecting, when such mistakes can be detected using the modeling Petri nets.

The author Agnieszka Lasota is a scholar within Sub-measure 8.2.2 Regional Innovation Strategies, Measure 8.2 Transfer of knowledge, Priority VIII Regional human resources for the economy Human Capital Operational Programme co-financed by European Social Fund and state budget.

KAPITAŁ LUDZKI NARODOWA STRATEGIA SPÓJNOŚCI · Lubuskie Warte zachodu · UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

LITERATURA

[1] Banaszak Z., Kuś J., Adamski M., Sieci Petriego. Modelowanie, sterowanie i synteza systemów dyskretnych, Wydawnictwo Wyższej Szkoły Inżynierskiej, Zielona Góra, 1993

[2] Murata T., Petri Nets: Properties, Analysis and Applications, *Proceedings of the IEEE*, nr 77, 1989, 541–580

[3] Szpyrka, M., Sieci Petriego w modelowaniu i analizie systemów współbieżnych. *Warszawa: WNT*, 2008

[4] Lasota A., Modelowanie produkcji za pomocą diagramów aktywności UML i α-sieci na przykładzie obudowy separatora olejowego", Metody Informatyki Stosowanej, nr 2, 2010, 85—96

[5] Lasota A., Karatkevich A., Modeling of production processes using UML and Petri nets, *Preprints of the 4th IFAC Workshop on Discrete-Event System Design - DESDes '09*, Gandia Beach, Hiszpania, 2009, 247-252

[6] Dubois D., Stecke K. E., Using Petri nets to represent production processes, Decision and Control, *The 22nd IEEE Conference on*, 1983, 1062 – 1067

[7] Sadiq W., Orlowska M. E., Applying graph reduction techniques for identifying structural conflicts in process models, *LNCS*, Volume 1626/1999, 1999, 195-209

[8] Baresi L., Pezz M., On Formalizing UML with High-Level Petri Nets, *Lecture Notes in Computer Science*, Volume 2001/2001, 2001, 276-304

[9] López-Grao J. P., Merseguer J., Campos J., From UML Activity Diagrams To Stochastic Petri Nets: Application To Software Performance Engineering, *Proceedings of the 4th international workshop on Software and performance - WOSP'04*, New York, USA, 2004

[10] Debasish K., Debasis S., A novel approach to generate test cases from UML activity diagrams, *Journal of Object Technology*, Vol. 8, nr 3, May-June 2009, 65-83

[11] Karatkevich A., To behavior analysis of a class of Petri nets, *Proceedings of 27th IFAC/IFIP/IEEE Workshop on Real-Time Programming - WRTP'03*, Łagów, Polska, 2003,- Elsevier Ltd, Oxford, UK, 2003, 33-38

[12] Karatkevich A., Dynamic Analysis of Petri Net-Based Discrete Systems. *LNCIS* (356), Berlin Heidelberg: Springer, 2007

[13] Zakrevskij A., High-level design of logical control devices, *Third International Conference Computer-Aided Design of Discrete Devices – CAD DD'99*, Minsk, Belarus, 1999, 13-18

[14] Valmari A., State of the Art Report: STUBBORN SETS, *Petri Net Newsletter*, nr 46, 1994, 6–14

[15] Karatkevich A., Zagadnienia redukcji podsieci automatowych w sieciach Petriego, *Przegląd Telekomunikacyjny i Wiadomości Telekomunikacyjne*, nr 6, 2008, 806-808

[16] Karatkevich A., Minimized Representation of State Machine Subnets of Petri Nets, *Proceedings of the Seventh International Conference on Computer-Aided Design of Discrete Devices CAD DD'2010*, Minsk, Belarus, 2010, 65-72

[17] Kovalyov A.V., Concurrency Relations and the Safety Problem for Petri Nets, *Proceedings of 13th International Conference on Application and Theory of Petri Nets – ATPN'92*, Sheffield, UK, 1992,- LNCS (161), Berlin Heidelberg: Springer-Verlag, 1992, 299-309

***Autorzy**:*
mgr inż. Agnieszka Lasota, Uniwersytet Zielonogórski, Wydział Elektrotechniki Informatyki i Telekomunikacji, Instytut Informatyki i Elektroniki, ul. prof. Z. Szafrana nr 2, 65-516 Zielona Góra, E-mail: a.lasota@weit.uz.zgora.pl;
dr hab. inż. Andrei Karatkevich, Wydział Elektrotechniki Informatyki i Telekomunikacji, Instytut Informatyki i Elektroniki, ul. prof. Z. Szafrana nr 2, 65-516 Zielona Góra, E-mail: a.karatkevich@iie.uz.zgora.pl;