REAL-TIME SYNCHRONIZATION MECHANIZM FOR COMPLEX CONCURRENTLY COMPETING PROCESSES COORDINATION

ZBIGNIEW BANASZAK*, BRUCE H. KROGH**

This paper presents a new approach to automated development of control software for concurrency control of discrete event systems as well as to the synthesis of computer-aided tools for production real-time management. A new modelling technique for complex concurrently competing processes description is introduced, then, sufficient conditions for deadlocks avoidance are presented. The processes considered consist of partially ordered elementary, i.e. sequential and pipeline-like flowing, production processes which share a set of resources in a production system. The results obtained allow one to design the real-time concurrency control programs of guaranteed correctness, i.e. the programs which guarantee absence of deadlocks, overflows and starvation during the competing processes interactions.

1. Introduction

Till now, a rigorous demonstration of the correctness of concurrent programs has proved to be extremely difficult and tedious. Conventional debugging techniques which are based on the test runs of the program with a fixed test data, are particularly inappropriate as an approach to generating satisfactory concurrent programs. That is because the nondeterministic behaviour of concurrent systems requires special techniques for the examination of the absence of deadlocks (Peterson and Silberschutz, 1983; Deitel, 1984; Raynal, 1986). A proof of the deadlock freedom for such systems is an integral part of a total correctness proof, and is often a desirable first step towards the proof. Note that in general case, the deadlock avoidance problem is

*Institute of Technical Cybernetics, Wroclaw Technical University, ul. Janiszewskiego 11/17, 50-372 Wroclaw, POLAND

**Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, USA

NP-complete (Gold, 1978). Existing automatic debugging and theorem proving techniques in the context of the deadlock phenomena, including the problem of their detection and protection methods against them, have very often been studied in the particular context of the computer operating systems (Obermarck, 1982; Hauschild, 1987).

We note, however, that the specification for resource requirements in a manufacturing system differ from those used for computer applications. The most notable difference is that the production routes, e.g. observed in a Flexible Manufacturing Systems (FMS), indicate an order in which resources of preassumed capacity must be allocated and deallocated to the pipeline- like performed jobs in the production system. For computer systems it is usually assumed that only bounds on the total number of resources required by each process consisting of a set of pipeline-like performed jobs are known. Taking into account, in the course of the competing processes execution control, a given order of shared resources utilization, makes the considered deadlock avoidance problem even more difficult. Thus, a good deadlock avoidance policy which allows to maximize the resources utilization so as to optimize the throughput of the production system under the above mentioned requirement is of primary concern in many applications.

Our approach to the deadlock avoidance problem is aimed at automated development of control software for discrete manufacturing systems, including the computer-aided tools design for modelling and simulation of FMS as well as for the design of the control logic for a class of discrete event industrial controllers, and is based on the following assumptions:

- the structure organization of the material and information processes occurring in Computer Integrated Manufacturing (CIM) systems is of a hierarchical nature. This means that each complex manufacturing process consists of simpler processes encompassing, for instance, different concurrently flowing and competing processes, linearly or partially ordered processes etc.,
- the Petri net based model encompassing all possible (including deadlocks) material flows of technological processes serves as a basis for the design of a new net model encompassing some of admissible, i.e. deadlock-free, control flows. This means that the obtained net model of control flow, reflecting some of the feasible material flows, after a dispatching rule implementation, can be applied directly as a real-time control program for an industrial controller.

This paper presents new results being a generalization of the issues dealt

with in our previous works (Banaszak, 1988a; Krogh and Banaszak, 1988; Banaszak and Roszkowska, 1988; Banaszak and Krogh, 1990). The research conducted has been motivated by the wish to enlarge a class of up to now considered processes, i.e. linearly ordered pipeline concurrent processes, to a class of the complex concurrent processes, i.e. partially ordered pipeline concurrent processes. Examples of such processes come from the foundry practice and automotive industry.

The task considered is stated as follows: Given a finite set of concurrently competing processes which must be executed asynchronously on a production system consisting of a finite number of components, find out an algorithm transforming a given processes specification into a control program whose correctness is proved. In other words, the algorithm we are looking for should realize the automatic conversion of an input problem description (presented in a standard form of processes specification) into a computer program suitable for the intended application.

To resolve the above stated problem we introduce a Petri net based modelling technique aimed at material flows net model design, following from a given processes specification. Then, we state and prove a theorem providing sufficient conditions for the deadlock avoidance in the processes considered. The conditions obtained allow us to introduce the inhibitor arcs to the net model of material flows and then to obtain a net model of admissible control flows.

This paper is organized into five parts. In part 2 we state our assumptions and introduce a model technique applicable to complex concurrent processes. Part 3 presents an overview of the deadlock handling techniques and provides the sufficient conditions for the material flows routings to be deadlock-free. These conditions are used in part 4 to develop a method of constructing deadlock-free algorithms for real-time synchronization of concurrently competing complex processes. Our conclusions are given in part 5.

2. Modelling Technique

We consider the following hierarchy of production processes. A simple process (SP) is specified by a production route indicating the sequence in which resources must be allocated to accomplish a job. On the base of the above introduced simple processes we define a class of pipeline processes. Each pipeline process (PP) consists of processes executed simultaneously along the same production route. Since each job requires only a certain resource at a given time, multiple jobs can be in progress concurrently.

As the next level in the processes hierarchy a class of pipeline concurrent processes (PCP) is considered. Each PCP contains a set of PPs executed concurrently along different production routes which are associated with manufacturing processes aimed at different products. In the case when the considered production route is described by a set of partially ordered resources required to accomplish a given production task (observed for instance in a foundry practice and automative industry where an assembly of foundry moulds and bodyworks processes is performed) we apply the concept of complex pipeline concurrent processes (CPCPs). This means that each element of CPCPs class consists of a set of processes performed concurrently along the different production routes. The precedence relation among the resources imposes a partial ordering on the PCPs executions following the order of a particular production route.

The Petri nets based techniques for PCP as well as PP and SP modelling have been presented by Banaszak (1988a), Krogh and Banaszak (1988). However, to cope with the CPCPs modelling problem some weaker requirements should be imposed on the production routes structure.

2.1. Complex Concurrent Processes

We consider a specification of the complex concurrent pipeline process

$$PC = (PR, \varphi), \tag{1}$$

being a pair consisting of a set of production routes

$$PR = \{PR_j |\ j = \overline{1,v}\},$$

$$\begin{aligned} PR_j \;=\; & \{(R_{j_i}, B^n_{j_l} M_{j_l}), (M_{j_l} B^m_{j_l}, R_{j_k}) | j_i, j_k = \overline{1, J_j}\ \& \\ & j_l = \overline{1, J_M}\ \&\ n, m = \overline{1, J_{M_{j_l}}}\}, \end{aligned}$$

and a buffer capacity function

$$\varphi :\ \{B^n_l |\ l = \overline{1, J_M}\ \&\ n = \overline{1, J_{M_l}}\} \to N,$$

where

R_{j_l} – denotes the j_i-th robot performing a transportation operation, e.g. the workpiece displacement between machine buffers, either between conveyors and buffers or between buffers and conveyors,

$B_{j_l}^n$– denotes the n-th buffer of the j_l-th machining or assembly machine,

M_{j_l}, (M_l)– denotes the j_l-th (l-th) machine employed in the course of technological operations performed during the j-th process execution,

φ– is a function describing a maximal number of pallets which can be stored in each machine buffer,

J_j, $J_M, J_{M_{j_l}}, (J_{M_l})$– denote a number of robots, machines, and buffers of each j_l-th (l-th) machine, respectively.

Elements of the introduced routes specification describe the precedence order of system resources involved in the course of the production processes execution. It is assumed that to each robot and machine occurrence in the production route there corresponds one technological operation, e.g. transportation and assembly/disassembly operation, respectively. Of course, the precedence relation of operations performance is determined by the partial ordering of the resources occurrence in every production route.

In order to illustrate the specification introduced, let us consider the following flexible machining and assembly module shown in Figure 1.

The module considered consists of two machine tools M_2 and M_3 as well as one multifunctional machining/assembly machine M_1. The machine tools M_2 and M_3 are equipped with the buffers B_2^1, B_2^2 and B_3^1, B_3^2, respectively. The machine M_1 is equipped with four buffers B_1^1, B_1^2, B_1^3 and B_1^4. Four robots R_1, R_2, R_3 and R_4 are responsible for transportation operations. Let us assume the following buffer capacities

$$\begin{aligned} &\varphi(B_1^2) = \varphi(B_1^3) = \varphi(B_1^4) = \varphi(B_2^1) = \varphi(B_2^2) = 1, \\ &\varphi(B_3^1) = \varphi(B_3^2) = 1, \quad \varphi(B_1^1) = 2, \end{aligned} \tag{2}$$

as well as the specification of the two production routes execution in the system considered

$$\begin{aligned} PR_1 = \{&(R_1, B_1^1 M_1), (M_1 B_1^4, R_1), (R_1, B_2^1 M_2), (M_2 B_2^2, R_2), \\ &(R_2, B_1^2 M_1), (R_3, B_3^1 M_3), (M_3 B_3^2, R_2), (R_2, B_1^1 M_1), \\ &(M_1 B_1^3, R_4)\}, \end{aligned} \tag{3}$$

$$PR_2 = \{(R_2, B_2^2 M_2), (M_2 B_2^1, R_1), (R_1, B_3^2 M_3), (M_3 B_3^1, R_3)\}.$$

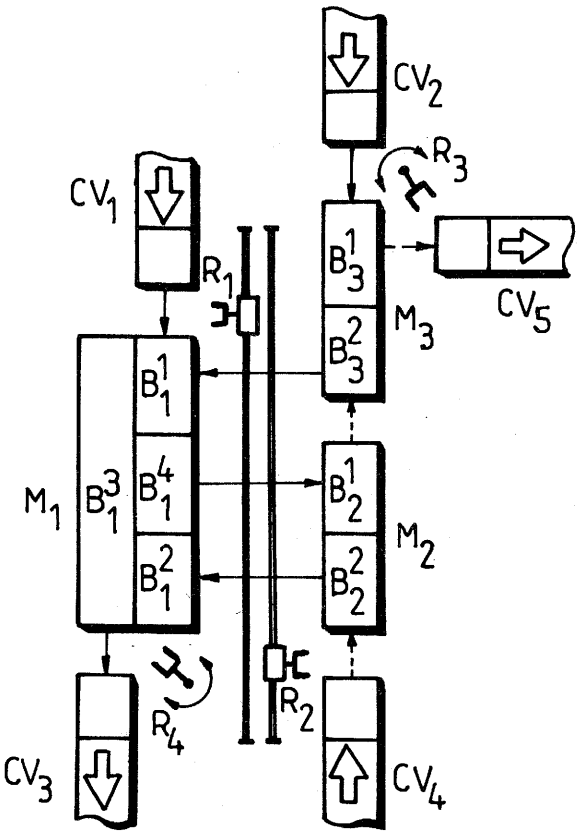

Figure 1. Flexible machining/assembly module

According to the first process described by the production route PR_1, two kinds of workpieces delivered by conveyors CV_1 and CV_2 are machined on M_1, M_2 and M_3, respectively. After machining, the workpieces are assembled on M_1 and then transported on the conveyor CV_3. The second process accomplishes machining of workpieces delivered by conveyor CV_4. After machining operations, performed on the machine tools M_2 and M_3 the workpieces are transported to the conveyor CV_5.

Note that in the case when $\varphi(B_1^1) = 1$, the first process cannot be continued. To illustrate such a situation consider the case when the buffer B_1^1 is busy with a workpiece which has come from CV_2, while no workpieces are observed in B_1^4, B_2^1, B_2^2 and B_1^2. In order to take into account the requirements mentioned above we assume that the admissible buffer capaci-

ties are determined by the maximal number of the buffer occurrences in the process specifications. Elimination of such situations do not exclude, however, the deadlock occurrences which can take place during the asynchronous processes execution. For illustration, let us consider the situation where a workpiece performed along the first process is stored in B_2^1, while B_2^2 is busy with a workpiece performed along the second process.

2.2. Net Model Representation

To model CPCPs we apply the Petri net formalism (Peterson, 1981; Reisig, 1982). Petri nets have been used extensively in the representation and analysis of distributed concurrent systems including distributed computing systems, operating systems of large computers as well as industrial processes control systems.

The problem we are facing now concerns the designing of an algorithm which allows us to transform a given processes specification into a related Petri net model encompassing all the possible overflow-free realizations of processes considered. Because a well known feature of Petri nets is that they rapidly become very large, along with the increase of the number of events modelled, i.e. net models obtained become very intricate and have quite unreadable graphs as well as their state exploration is limited by the combinatorial explosion, we focus our attention on the algebraic representation of the models constructed. In other words, our aim is to find out such an algorithm which performs the following transformation

$$PS = (PR, \varphi) \rightarrow PN = (C, K, M_0), \tag{4}$$

where C, K and M_0 denote the incidence matrix, the place function capacity, and the initial marking, respectively.

Note that the introduced algebraical representation of a Petri net is equivalent to the following one $PN = (P, T, E, K, M_0)$, where (P, T, E) such that $P \cup T \neq \emptyset$, $P \cap T = \emptyset$, $E \subset (P \times T) \cup (T \times P)$ hold, determines the incidence matrix C.

The main idea of the algorithm proposed lies in the assumption that to each resource occurrence, i.e. the robots and machines in the processes specification graph PR, is associated one transition of PN. To each element of PR_i, $i = \overline{1, v}$, there corresponds a place connecting the transitions which map resources distinguished in this element. Such places are treated as the models of the relevant machine buffers. Their capacity K is determined by function φ. Moreover, with each element of PR_i, $i = \overline{1, v}$, a place

modelling the actual state of the workpieces flow is associated in PN. The capacity of places considered is equal to 1 in the case when the capacity of the corresponding buffer is minimal, and may be determined in the range from 1 to R if this capacity is bigger, where R is a difference between the preassumed buffer capacity and its minimal value (see conditions (5)).

A minimal value of the buffer capacity is selected from a set of admissible buffer capacities. Note, that the admissible value of each buffer capacity is determined by the structure of production routes and guarantees that if in each separately considered production route at each of its input only one workpiece is supplied, then the relevant assembly process can be completed.

The initial marking M_0 is a zero-value row-vector and encompasses the initial state of the process performance corresponding to the situation where no workpieces are stored in the buffers.

To illustrate an application of the above described procedure, let us consider the Petri net model shown in Figure 2.

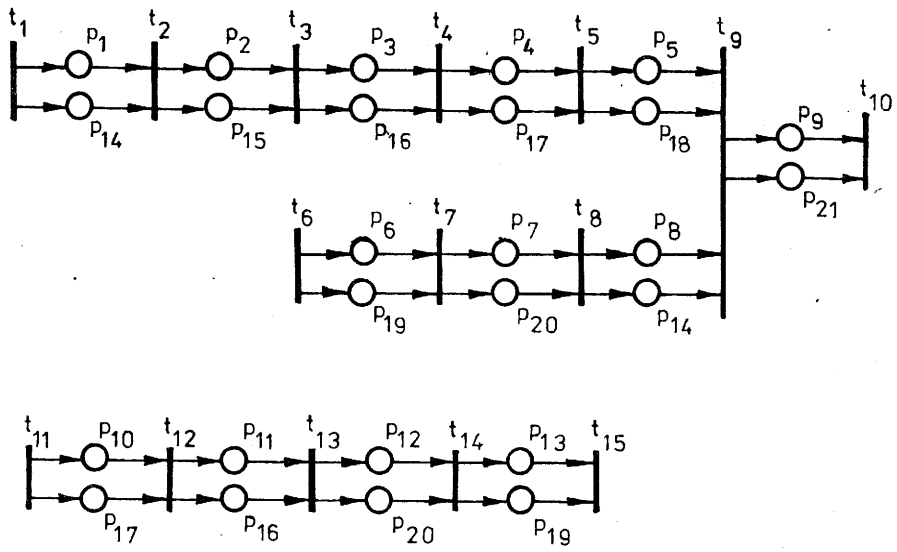

Figure 2. Petri net model of a material flow

The model above describes the net encompassing all the material flows which can be observed in processes specified by production routes (3) and performed in the flexible module from Figure 1. Places $p_1 - p_{13}$ reflect the actual workpiece flows. The place p_{14} corresponds to the buffer B_1^1, p_{15} to B_1^4, p_{16} to B_2^1, p_{17} to B_2^2, p_{18} to B_1^2, p_{19} to B_3^1, p_{20} to B_3^2, and p_{21} to B_1^3. Transitions t_1, t_3 and t_13 correspond to the robot R_1, t_6, t_{15} to R_3, t_8, t_{11}, t_5 to R_2 and t_{10} to R_4 as well as the transitions t_2 and t_9 correspond to the machine M_1, t_4 and t_{12} to M_2, and t_7 and t_{14} to M_3.

As it was assumed, in each $PN = (P, T, E, K, M_0)$ two types of places are distinguished, modelling the actual workpieces flow W and modelling the machine buffers B. The following conditions hold: $P = W \cup B$, $W \cap B = \emptyset$.

In our further considerations we shall assume that for each PN the following conditions hold

$$\begin{aligned} &\text{(i)} && (\forall p \in B)(\ K(p) \geq \min(p)\ ), \\ &\text{(ii)} && (\forall p' \in W)(1 \leq K(p') \leq K(p) - \min(p) + 1\ \ \& \\ & && \ \& \ p^{\cdot} \in B\ \& \ \ {}^{\cdot}p \cap {}^{\cdot}p' \neq \emptyset\ \& \ \ p^{\cdot} \cap p'^{\cdot} \neq \emptyset), \end{aligned} \tag{5}$$

where

$$\min(p) = \sum_{i=1}^{g_j} U_i(p), \qquad j = \overline{1, v},$$

$$U_i(p) = \begin{cases} 1 & \text{if in the } i\text{-th branch of } PR_j \text{ digraph there} \\ & \text{exists a buffer modelled by } p \\ 0 & \text{otherwise,} \end{cases}$$

g_j–denotes a number of branches in PR_j.

The algebraic representation of the Petri net model considered has the following form

$$PN = (C, K, M_0), \tag{6}$$

where the incidence matrix C has the structure shown in Fig. 3, and for the capacity function K as well as for the initial marking M_0 the following conditions hold

(i) $(\forall i = \overline{1,21} \setminus \{14\})(K(p_i) = 1), \quad K(p_{14}) = 2,$

(ii) $(\forall i = \overline{1,21})(M(p_i) = 0).$

$C =$

$t_j \setminus p_i$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
2	−1	1	0	0	0	0	0	0	0	0	0	0	0	−1	1	0	0	0	0	0	0
3	0	−1	1	0	0	0	0	0	0	0	0	0	0	0	−1	1	0	0	0	0	0
4	0	0	−1	1	0	0	0	0	0	0	0	0	0	0	0	−1	1	0	0	0	0
5	0	0	0	−1	1	0	0	0	0	0	0	0	0	0	0	0	−1	1	0	0	0
6	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
7	0	0	0	0	0	−1	1	0	0	0	0	0	0	0	0	0	0	0	−1	1	0
8	0	0	0	0	0	0	−1	1	0	0	0	0	0	1	0	0	0	0	0	−1	0
9	0	0	0	0	−1	0	0	−1	1	0	0	0	0	−1	0	0	0	−1	0	0	1
10	0	0	0	0	0	0	0	0	−1	0	0	0	0	0	0	0	0	0	0	0	−1
11	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0
12	0	0	0	0	0	0	0	0	0	−1	1	0	0	0	0	1	−1	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	−1	1	0	0	0	−1	0	0	0	1	0
14	0	0	0	0	0	0	0	0	0	0	0	−1	1	0	0	0	0	0	1	−1	0
15	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	0	0	0	−1	0	0

Figure 3. The incidence matrix of PN from Figure 2

Note that the algebraic representation of the material flow models allows to apply the simple form of the following next-state function

$$M' = M + e[t]C, \tag{7}$$

where
M' is a marking obtained from the transition t firing after being enabled at M,
$e[t]$ is the unit vector which is zero everywhere except at the component corresponding to t.

3. Synchronization Mechanizm

In an FMS, several production processes may compete for a finite number of resources, e.g. machine tools, workpiece and tool stores, robots and conveyors. To increase the system throughput an asynchronous control strategy of processes cooperation should be employed. This strategy should maximize the system resources utilization, however, it may lead to deadlocks occurrence. Thus, resource sharing techniques guaranteeing the deadlock

avoidance for real-time processes control are the primary goal of programmers and the FMS dispatchers.

3.1. Deadlock Handling Techniques

A problem of mutual exclusion which arises during the concurrent processes management, i.e. the resource allocation, is the fundamantal one for desigining synchronization machanisms of competing processes. This problem, which directly corresponds to the development of deadlock-free resources scheduling policies, has been analyzed extensively by Raynal (1986) and Deitel (1984).

The major areas of deadlock research consist of prevention methods in which deadlocks may be prevented by denying one or more of the necessary conditions for the deadlock. The following four conditions are necessary for a deadlock to occur among the concurrent processes:

- **mutual exclusion**; processes claim exclusive control of their resources,
- **hold while waiting**; processes may hold resources while waiting for additional resources to be allocated,
- **no preemption**; resources may not be removed forcibly from processes,
- **circular wait**; there is a closed chain of processes in which each process is waiting for a resource held by the next process in the chain.

Our approach to the deadlock prevention is to assume that the circular wait condition never holds.

3.2. Synchronization Conditions

In our (Krogh and Banaszak, 1988; Banaszak and Krogh, 1990) earlier works the processes considered have been determined by orders in which resources are requested or released. In other words, the production routes determining PCP are the ordered lists of resources indicating the sequences in which resources must be allocated to accomplish their execution. This allows us to decompose each production route into synchronization zones, where each zone is a sequence of shared resources followed by a sequence of unshared resources. Shared resources are resources which are used by more than one process, while unshared resources are used exclusively by one process.

Thus, the circular wait condition may be avoided by requiring that some resources of synchronization zones in each production route are released. An idea supporting this assumption employs the observation according to which keeping some of unshared resources released, while some of shared

resources in the synchronization zone are requested, always allows the system to complete the processes execution. This guarantees that from a safe state only the safe states can be reached in a system governed by such a policy.

To cope with the synchronization problem of CPCPs we employ the same approach. Note that each complex process can be treated as a set of PCPs. That is because each production route, determined by the digraph reflecting the partially ordered resources, can be considered as a set of linearly ordered digraphs covering the former one. This observation allows us to apply directly the solution of PCPs synchronization problem to the CPCPs one.

The problem considered now is how to transform a Petri net model obtained from (4) into a deadlock-free one being a real-time resource allocation scheme. Our objective is to show how the concept of synchronization zones can be applied in the case considered.

Let $PN = (P, T, E, K, M_0)$ be the net model obtained from the transformation (4). In general each PN can be considered as the following set

$$PN = \bigcup_{i=\overline{1,v}} PN_i = (P_i, T_i, E_i, K_i, M_0^i), \tag{8}$$

with the following conditions held

(i) $(\forall\, i = \overline{1,v})(\exists\, W_i \subset P_i)(\exists\, B_i \subset P_i)(P_i = W_i \cup B_i \;\&\; W_i \cap B_i = \emptyset)$,

where

$W = \bigcup_{i=\overline{1,v}} W_i$ is a subset of places modelling the workpieces flow,

$B = \bigcup_{i=\overline{1,v}} B_i$ is a subset of places modelling the machine buffers,

(ii) $(\forall\, i,j = \overline{1,v})(W_i \cap W_j = \emptyset \;\&\; T_i \cap T_j = \emptyset \;\&\; \|W_i\| = \|T_i\| - 1 \;\&\; i \neq j)$,

(iii) $P = \bigcup_{i=\overline{1,v}} P_i, \quad T = \bigcup_{i=\overline{1,v}} T_i,$

(iv) there exists a function

$$f : \bigcup\{PR_i |\; i = \overline{1,v}\} \xrightarrow{1-1} \{({}^wP,\, {}^wT,\, {}^wE) |\; w = \overline{1,z}\},$$

where $z = \|\bigcup\{PR_i |\; i = \overline{1,v}\}\|$,

such that for each

$(R_{j_i}, B^n_{j_l} M_{j_l})$ and $(M_{j_k} B^m_{j_k}, R_{j_h})$ there exists
$({}^zP, {}^zT, {}^zE)$ and $({}^uP, {}^uT, {}^uE)$ respectively, where
${}^zP = \{p^I, p^{II}\}$, ${}^zT = \{t^I, t^{II}\}$,
${}^zE = \{(t^I, p^I), (t^I, p^{II}), (p^I, t^{II}), (p^{II}, t^{II})\}$,
${}^zP = \{p^{III}, p^{IV}\}$, ${}^uT = \{t^{III}, t^{IV}\}$,
${}^uE = \{(t^{III}, p^{III}), (t^{III}, p^{IV}), (p^{III}, t^{IV}), (p^{IV}, t^{IV})\}$,
and
$p^I, p^{III} \in W$, $p^{II}, p^{IV} \in B$, $t^I, t^{II}, t^{III}, t^{IV} \in T$, $t^I \neq t^{IV}$
the following implications hold
if $M_{j_l} = M_{j_k}$, then $t^{II} = t^{III}$,
if $M_{j_l} = M_{j_k}$ and $m = n$, then $t^{II} = t^{III}$ and $p^{II} = p^{IV}$,

(v) $(\forall p \in P)(\forall i = \overline{1,v})(M_0(p) = M_0^i(p) = 0 \;\&\; K(p) = \max\{K_i(p) \mid i = \overline{1,v}\})$.

Note that each PN_i, $i = \overline{1,v}$, can be described by its elementary nets PN_i^k, $k = \overline{1,g_i}$, i.e. by the set

$$PN_i = \bigcup_{k=\overline{1,g_i}} PN_i^k = (P_i^k, T_i^k, E_i^k, K_i^k, M_0^{i,k}), \tag{9}$$

for which the following conditions are fulfilled

(i) $(\forall k = \overline{1,g_i})(\exists! t^I \in T_i^k)(\exists! t^{II} \in T_i^k)(\cdot t^I = \emptyset \;\&\; t^{II} \cdot = \emptyset)$,

(ii) $(\forall k = \overline{1,g_i})(\forall t \in T_i^k \setminus \{t^{II}\})(\exists! t^* \in T_i^k)(t \cdot = \cdot t^*)$,

(iii) $(\forall k = \overline{1,g_i})(\exists l = \overline{1,g_i})(W_i^k \cap W_i^l = \emptyset \;\&\; l \neq k \;\&\; P_i^k = W_i^k \cup B_i^k \;\&\; P_i^l = W_i^l \cup B_i^l)$,

(iv) $P_i = \bigcup_{k=\overline{1,g_i}} P_i^k$, $T_i = \bigcup_{k=\overline{1,g_i}} T_i^k$, $E_i = \bigcup_{k=\overline{1,g_i}} E_i^k$,

(v) $(\forall p \in P_i)(\forall k = \overline{1,g_i})(M_0^{i,k}(p) = M_0^i(p) = 0 \;\&\; K_i(p) = \max_{k=\overline{1,g_i}}\{K_i^k(p) \mid k = \overline{1,g_i}\})$.

It means that each PN_i^k contains only the one source, i.e. $t \in T_i^k$ such that $\cdot t = 0$, and only the one sink, i.e. $t^I \in T_i^k$ such that $t^I \cdot = \emptyset$. Moreover, from condition (ii) it follows that $\|W_i^k\| = \|T_i^k\| - 1$, and that for each $t, t^I \in T_i^k$ such that $t \cdot \cap \cdot t^I \neq \emptyset$ there exists only the one $p \in t \cdot \cap \cdot t^I$ such that

$p \in B_i^k$. From condition (iv) it follows that if $(\exists k = \overline{1,g_i})(\exists l = \overline{1,g_i})(k \neq l \ \& \ T_i^k \cap T_i^l \neq \emptyset)$ holds, then for $t, t^I \in T_i^k$ such that $t^{\cdot} = {}^{\cdot}t^I = \emptyset$, and $t^{II}, t^{III} \in T_i^l$ such that $t^{II\cdot} = {}^{\cdot}t^{III} = \emptyset$, whether $t = t^{II}$ and $t^I = t^{III}$ or $t = t^{II}$ and $t^I \neq t^{III}$ or $t \neq t^{II}$ and $t^I = t^{III}$ and $t^I = t^{II}$ and $t \neq t^{III}$ or $t = t^{III}$ and $t^I \neq t^{II}$ hold.

To illustrate the above mentioned decomposition let us consider the Petri net model shown in Figure 2.

Note that $PN = (P, T, E, K, M_0)$ consists of two following subnets:

$$PN_1 = (P_1, T_1, E_1, K_1, M_0^1),$$

$$PN_2 = (P_2, T_2, E_2, K_2, M_0^2),$$

where

$$P_1 = \{p_i |\ i = \overline{1,21} \setminus A\}, \quad A = \{10, 11, 12, 13\},$$

$$T_1 = \{t_i |\ i = \overline{1,10}\}, \quad T_2 = \{t_i |\ i = \overline{11,15}\},$$

$$P_2 = \{p_i |\ i \in A \cup D\}, \quad D = \{16, 17, 19, 20\}.$$

PN_1 is covered by the following nets

$$PN_1^1 = (P_1^1, T_1^1, E_1^1, K_1^1, M_0^{1,1}),$$

$$PN_1^2 = (P_1^2, T_1^2, E_1^2, K_1^2, M_0^{1,2}),$$

where

$P_1^1 = \{p_i |\ i = \overline{1,5} \cup \overline{14,18} \cup \{9,21\}\}$,
$T_1^1 = \{t_i |\ i = \overline{1,5} \cup \overline{9,10}\}$,
$T_1^2 = \{t_i |\ i = \overline{6,10}\}$,
$P_1^2 = \{p_i |\ i = \overline{6,9} \cup \{14,19,20,21\}\}$,

while PN_2 consists only of the covering net $PN_2^1 = PN_2$.

Note that in each elementary net PN_i^k there exists P_i^k consisting of the two subsets of places, modelling the workpieces flow W_i^k, and modelling the machine buffers B_i^k, respectively. Of course, the following conditions are satisfied

$$\text{(i)} \qquad P_i^k = W_i^k \cup B_i^k, \tag{10}$$

$$\text{(ii)} \qquad W_i^k \cap B_i^k = \emptyset.$$

The linear ordering of the W_i^k elements, observed in PN_i^k, implies the linear order of transitions from the set T_i^k. Thus, there exists the following sequence

$$BS_i^k = (p|\ p \in B_i^k\ \&\ p \in t_1 \cap^{\cdot} t_{l+1}\ \&\ l = \overline{1, \|T_i^k\|}), \tag{11}$$

which reflects the machine buffers occurrence along the relevant path in the production route.

In a general case, a set B consists of two subsets SP and UP. Distinguished subsets, i.e. a set of shared places SP and a set of unshared places UP, are such that $B = SP \cup UP$ and $SP \cap UP = \emptyset$ hold. The subsets considered are defined as follows

$$\begin{aligned} UP = \{p|\ p \in B_i^k\ \&\ p = \operatorname{crd}^l BS_i^k\ \&\ (\forall j = \overline{1, |BS_i^k|})(l \neq j \rightarrow p \neq \\ \operatorname{crd}^j BS_i^k)\ \&\ (\forall n = \overline{1,v} \setminus \{i\})(\forall k = \overline{1, g_n})(\forall j = \overline{1, |BS_n^k|}) \\ (p \neq \operatorname{crd}^j BS_n^k)\ \&\ i = \overline{1,v}\ \&\ k = \overline{1, g_i}), \end{aligned} \tag{12}$$

$$SP = W \setminus UP,$$

where $\operatorname{crd}^i S = s_i$ for $S = (s_1, ..., s_i, ..., s_r)$, and $|D| = z$ stands for the length of $D = (d_i|\ i = \overline{1,z})$.

In other words the set of shared resources contains the places which correspond to the machine buffers uniquely occurring in the specification of production routes PR, i.e. the places where each one uniquely occurrs in a unique BS_i^k. The places occurring at least twice either in one BS_i^k or two different sequences BS_i^k, BS_j^l, $i,j = \overline{1,v}$, $k = \overline{1, g_i}$, $l = \overline{1, g_j}$ belong to the set of shared places.

Let us consider a set $PN_i^K = \{PN_i^l, PN_i^k, PN_i^j, \ ...\ , PN_i^m, PN_i^n\}$, $PN_i^K \subset PN_i$ such that $t, t^I \in T_i^l$, $^{\cdot}t = t^{I\cdot} = \emptyset$, $t^{II}, t^{III} \in T_i^k$, $t^I = t^{II}$, $^{\cdot}t^{II} = t^{III\cdot} = \emptyset$, and $t^{IV}, t^V \in T_i^j$, $t^{III} = t^{IV}$, $^{\cdot}t^{IV} = t^{V\cdot} = \emptyset$, and so on.

It means that PN_i^K can be considered as a model of a PP distinguished in PN_i modelling a CPCP.

According to (11) there exists

$$BS_i^K = BS_i^l \hat{} BS_i^k \hat{} BS_i^j \hat{} \ ... \ \hat{} BS_i^m \hat{} BS_i^n, \tag{13}$$

being a concatenation of sequences encompassing the machine buffers occurrence along a PP defined by PN_i^K.

Let H_i be a set of all the possible PN_i^K which can be obtained from PN_i. Note that the set H_i corresponds to the following set

$$BS_i = \{BS_i^K \mid K = \overline{1, \|H_i\|}\}. \tag{14}$$

Consider a set of so called synchronization zones SZ_i^K defined for BS_i^K as follows

$$SZ_i^K = \{(SS_i^K(l,m), SU_i^K(m+1,n)) \mid l,m,n = \overline{1, \|BS_i^K\|}\}, \tag{15}$$

where

$$\begin{aligned} SS_i^K(l,m) &= (p \mid p = \mathrm{crd}^r BS_i^K \ \&\ p \in SP;\ r = \overline{l,m}\ \& \\ &\quad \mathrm{crd}^{r-1} BS_i^K \notin SP\ \&\ \mathrm{crd}^{m+1} BS_i^K \notin SP), \\ SU_i^K(m+1,n) &= (p \mid p = \mathrm{crd}^r BS_i^K \ \&\ p \in UP\ \&\ r = \overline{m+1,n}\ \& \\ &\quad \mathrm{crd}^m BS_i^K \notin UP\ \&\ \mathrm{crd}^{n+1} BS_i^K \notin UP). \end{aligned}$$

Thus, the set of synchronization zones for a given PN is determined as follows

$$SZ = \bigcup_{i=\overline{1,v}} \{SZ_i^K \mid K = \overline{1, \|H_i\|}\}. \tag{16}$$

On the basis of (14) the following definition can be introduced.

Definition 1. Let $PN = (P, T, E, K, M_0)$ be a Petri net model of CPCP processes which satisfy conditions (5). A $PN_I = (P, T, E, F, K, M_0)$ is said to be a net model of CPCP control flow. The set of the inhibitor arcs $F = F^I \cup F^{II}$ is determined by the following expressions

$$\begin{aligned} F^I &= \{[p,t] \mid t \in \widetilde{T}\widetilde{S}_i^K(l,m)\ \&\ p \in \widetilde{S}\widetilde{S}_i^K(l,m)\ \&\ p \notin t^{\cdot} \cup {}^{\cdot}t\ \& \\ &\quad l,m = \overline{1, |BS_i^K|}\ \&\ K = \overline{1, \|H_i\|}\ \&\ i = \overline{1,v}\}, \\ F^{II} &= \{[u,t] \mid t \in TS_i^K(l,m)\ \&\ u \subset \widetilde{S}\widetilde{U}_i^K(m+1,n)\ \&\ l,m,n = \\ &\quad 1, \overline{|BS_i^K|}\ \&\ K = \overline{1, \|H_i\|}\ \&\ i = \overline{1,v}\}, \end{aligned} \tag{17}$$

where
$[p,t] \in P \times T$ is a so called disjunctive inhibitor arc,

$[u,t] \overset{\text{def}}{\Longleftrightarrow} \{[p,t] |\ p \in u\ \&\ u \subset P\} \subset P \times T$ is a set of so called conjunctive inhibitor arcs,

$TS_i^K(l,m) = (t_r | \mathrm{crd}^r SS_i^K(l,m) \in t_r^{\cdot}\ \&\ l = \overline{1,a}\ \&\ a = |SS_i^K(l,m)|)\,(t_{a+1} |$ $\mathrm{crd}^a SS_i^K(l,m) \in {}^{\cdot}t_{a+1})$ is a concatenation of transition sequences determined by sequence $SS_i^K(l,m)$ consisting of the shared places,

$\widetilde{T}\widetilde{S}_i^K(l,m) = \{t_r |\ t_r = \mathrm{crd}^r TS_i^K(l,m)\ \&\ r = \overline{1,a+1}\}$ is an ordered set of transitions determined by $TS_i^K(l,m)$,

$\overline{TS}_i^K(l,m) = \widetilde{T}\widetilde{S}_i^K(l,m) \in \{t_{a+1}\}$,

$\widetilde{S}\widetilde{S}_i^K(l,m) = \{\mathrm{crd}^r SS_i^K(l,m) |\ r = \overline{1,m}\}$ is an ordered set of shared places included in the sequence $SS_i^K(l,m)$,

$\widetilde{S}\widetilde{U}_i^K(m+1,n) = \{\mathrm{crd}^r SU_i^K(m+1,n) |\ r = \overline{m+1,n}\}$ is an ordered set of unshared places included in the sequence $SU_i^K(m+1,n)$.

The above definition, showing the way of the inhibitor arcs introduction to the net model of material flows, allows one to find conditions preventing the deadlock occurrence in PN_I. The following theorem presents enableness conditions guaranteeing the deadlock-freeness of PN for M_0 being a zero value row-vector.

Theorem 1. Let PN_I be a net model fulfilling conditions of Definition 1. PN_i is a live Petri net, if the following conditions determining its transitions firing hold

$$\begin{aligned}
&\text{(i)} && (\forall p \in {}^{\cdot}t)(\ M(p) \geq 1\ ),\\
&\text{(ii)} && (\forall p \in t^{\cdot})(\ M(p) < K(p)\ ),\\
&\text{(iii)} && (\forall [u,t] \in F^{II})(\exists p \in u)(\ M(p) < K(p)\ ),\\
&\text{(iv)} && (\forall p \in \{p |\ [p,t] \in F^I\})(\ M(p) < K(p)\ ).
\end{aligned} \tag{18}$$

Note that in a general case the liveness of PN_I is not guaranteed for other firing rules. The proof of the theorem is a direct consequence of the observation that PCP which may consist of several PPs performed along the same production route is equivalent to the one of its PP processes. Thus, each PN_i, i.e. a Petri net model of CPCP which can be covered by a set of Petri net models of PPs such that the source and sink transitions of each "covering" net correspond to some transitions being the source and sink transitions of the PN_i Petri net model. Since the synchronization problem of PCP has been formulated and solved by Banaszak (1988a), Krogh and

Banaszak (1988) by employing the same concept of synchronization zones, hence the proof technique used in the case considered here is the same.

In order to illustrate the application of Definition 1 to the design of a set of synchronization zones, let us consider the Petri net model shown in Figure 2. The introduced sets are as follows

$$BS_1^1 = (p_{14}, p_{15}, p_{16}, p_{17}, p_{18}, p_{21}), \quad BS_1^2 = (p_{19}, p_{20}, p_{14}, p_{22}),$$

$$BS_2^1 = (p_{17}, p_{16}, p_{20}, p_{19}), \quad UP = (p_{15}, p_{18}, p_{21}),$$

$$SP = (p_{14}, p_{16}, p_{17}, p_{19}, p_{20}), \quad SZ_1^1(1,2) = (p_{14}, p_{15}),$$

$$SZ_1^1(3,6) = (p_{16}, p_{17}, p_{18}, p_{21}), \quad SZ_1^2(1,4) = (p_{19}, p_{20}, p_{14}, p_{21}),$$

$$SZ_2^1(1,4) = (p_{17}, p_{16}, p_{20}, p_{19}), \quad \tilde{T}\tilde{S}_1^1(1,1) = \{t_1, t_2\},$$

$$\tilde{T}\tilde{S}_1^1(3,4) = \{t_3, t_4, t_5\}, \quad \tilde{T}\tilde{S}_1^2(1,3) = \{t_{13}, t_7, t_8, t_9, \},$$

$$\tilde{T}\tilde{S}_1^2(1,4) = \{t_{11}, t_{12}, t_{13}, t_{14}, t_{15}\}, \quad \tilde{S}\tilde{S}_1^1(1,1) = \{p_{14}, \},$$

$$\tilde{S}\tilde{S}_1^1(2,3) = \{p_{16}, p_{17}\}, \quad \tilde{S}\tilde{S}_1^2(1,3) = \{p_{17}, p_{20}, p_{14}\},$$

$$\tilde{S}\tilde{S}_2^1(1,4) = \{p_{17}, p_{18}, p_{20}, p_{19}\}, \quad SU_1^1(2,2) = \{p_{15}\},$$

$$SU_1^1(5,6) = \{p_{18}, p_{21}\}, \quad SU_2^1(4,4) = \{p_{21}\},$$

$$\begin{aligned} F^I = \{&[p_{17}, t_3], [p_{16}, t_5], [p_{20}, t_6], [p_{14}, t_6], [p_{14}, t_7], [p_{19}, t_8], \\ &[p_{19}, t_9], [p_{20}, t_9], [p_{16}, t_{11}], [p_{20}, t_{11}], [p_{19}, t_{11}], [p_{20}, t_{12}], \\ &[p_{19}, t_{12}], [p_{19}, t_{13}], [p_{20}, t_{15}], [p_{16}, t_{15}], [p_{17}, t_{15}], [p_{16}, t_{14}], \\ &[p_{17}, t_{14}], [p_{17}, t_{13}]\}, \end{aligned}$$

$$\begin{aligned} F^{II} = \{&[p_{15}, t_1], [p_{18}, t_3], [p_{18}, t_4], [p_{21}, t_3], [p_{21}, t_4], [p_{21}, t_6], \\ &[p_{21}, t_7], [p_{21}, t_8]\}. \end{aligned}$$

Theorem 1 provides sufficient conditions for a live Petri net models design. An algorithm aimed at the design of these models, being a direct implementation of the conditions mentioned as well as of the synchronization zones concept, allows one to transform Petri nets models (4) into a live net models of CPCPs control flow. The transformation considered has the following form

$$PN = (C, K, M_0) \rightarrow PN_I = (\widehat{C}, K, M_0), \tag{19}$$

where $\widehat{C}$ is the incidence matrix designed on the base of matrix C according to the following rule

$$c_{i,j} = \begin{cases} 2 & \text{if } [p_j, t_i] \in F^I, \\ 3 & \text{if } p_j \in u \text{ and } [u, t_i] \in F^{II}, \\ c_{ij} & \text{otherwise.} \end{cases}$$

Note that compounding of transformations (4) and (19) allow us to consider some methods aimed at the automatic program synthesis.

4. Real-Time concurrency Control

Treating the future production systems as systems which will operate with much more dynamic resources allocation and where more processes will run concurrently, it should be pointed out that deadlock prevention and avoidance problems will become fundamental issues. Such systems, being either operating or distributed real-time control systems, will be oriented more than the actual ones toward the asynchronous parallel operation as well as toward the concurrent control allowing to acquire and release resources as freely as needed.

Thus, in order to increase the system resources utilization, the reliable, i.e. deadlock-free proved, real-time concurrency control policies will play an important role. So, the problem of automatic programming of concurrency control programs guaranteeing their correctness become very actual.

4.1. Automatic Program Synthesis

Our approach to the automatic programming assumes the existence of the following three elements:

- input data language for the problem specification,
- computer-aided tool for the control program evaluation,
- object oriented interface for the program implementation.

Note that the rules determining the introduced CPCPs specification can be treated as an input data language which allows one to describe the structure of the processes controlled as well as the requirements preassumed on the resources utilization, e.g. the capacity of machine buffers. Also, according to transformations (4), (19) an algorithm aimed at the design of Petri net models of control flows can be designed easily. Net models obtained from

that algorithm will reflect some of feasible, i.e. deadlock-free, realizations of the processes modelled. However , because of the nondeterminancy of the algorithms modelled, caused by the occurrence of conflict, their direct implementation as control programs is limited in practice.

To select the best one from among the available processes realizations, e.g. allowing to maximize a given index of the system performance, a proper dispatching rule should be applied. The application of a dispatching rule allows to supervise the conflicts occurrence, however, depending on the rule applied, different values of the performance index may be obtained. This leads us to the design of a computer-aided tool aimed at modelling and the performance evaluation of control algorithms obtained by compounding a net model with different dispatching rules. The net model of control flow, enriched by a selected dispatching rule, can be treated then as a model of a control algorithm oriented toward a given application.

Depending on the application, the net models can be transformed either into the RAM stored programs of logic controllers or implemented as a software package supporting the task oriented computer-aided tool, e.g. aimed at scheduling, dispatching, processes and/or production planning and so on.

The above methodology of automatic program synthesis also may be applied during the designing of adaptive control systems for discrete manufacturing processes. Such a possibility directly comes from the observation that both the algorithm for design of control flow net models and the algorithm for the best dispatching rule selection may be implemented easily as standard software procedures. During the adaptive control stage, started by the system resources breakdowns or changes of the production goals, the package is utilized each time when a new control flow scheme or dispatching rule is required.

4.2. Applications

The presented approach, being our contribution into solving the automatic programming problem, may be applied to the design of computer-aided tools aimed at the modelling and the performance evaluation of FMSs as well as to the design of logic control programs for real-time industrial controllers.

Early results concerning PCP processes have been implemented by Banaszak and Lech (1988) as well as by Krogh and Ekberg (1987) in the self-programmable controllers of Flexible Machining Modules. The programming of the real-time controllers should take into account actual changes of

operation processing times and controls ensuring the desirable order of the process performance. The approach having been introduced here allows us to avoid the laborious and time-consuming programming of the admissible synchronization of module components activities as well as to select the best dispatching rule.

Other implementation of such an approach has been described by Banaszak (1988b), Kus and Banaszak (1988). Two computer-aided modelling and performance evaluation systems for AGVs and FMMs have been aimed at processes and production planning as well as competing processes scheduling and dispatching. Both of them allow us to consider components breakdowns as well as the production goal changes.

The procedure of adaptive control applied to the components occurrence consists of the following three stages.

Identification. Determine which system component has broken down, and which other components can replace its functions. If there is no component capable to replace the one broken down then some of the processes performed in the system cannot be continued, else go to the next stage.

Decision Making. From a set of system components capable of replacing the broken down component, choose the best one and then determine the corresponding constraint vector. According to the algorithm of a net model modification, set up a new control procedure and go to the next stage. Note that the selection of the best component as well as of a new dispatching rule can be made either according to a priori determined priority rules or due to the results obtained from the computer simulation.

Modification. Suspend the realization of the to be replaced control procedure, and determine from its reachability tree the marking M associated with the actual state of the stopped processes. In the reachability tree associated with the control procedure provided in the previous stage, find out the marking M' reflecting the admissible displacement of the workpieces in the system. Then after the actualization of the workpieces displacement, start the newly obtained procedure taking into account M' as the initial marking.

In order to avoid execution of the above mentioned searching-like procedure an alternative approach can be applied.
It assumes that at the stage of Modification each buffer capacity can be enlarged by 1. The alternative procedure assumes that:

- the control procedure employed before the breakdown occurrence is taking into account all buffer capacities as equal to $K(p)-1,\ p \in B$,
- the new control procedure obtained at the stage of Decision Making takes into account all buffer capacities as equal to $K(p),\ p \in B$,
- after the period during which the processes that requested their access to the system are stopped and some processes performed currently in the system become completed which is leading to the state M such that $(\forall p \in B)(M(p) \leq K(p)-2))$, the newly employed procedure starts to take into account all buffers capacities as equal to $K(p)-1,\ p \in B$.

A similar procedure has been applied to production goal changes, i.e. to situations when new processes are introduced to the system while other ones are actually executed.

We believe that similar applications to the above mentioned will be found for systems including CPCP-like processes.

5. Conclusions

In the paper we have described the design and implementation of a new deadlock prevention technique for CPCPs. Our considerations refer to a class of systems in which production routes, being partially ordered digraphs of resources precedence, are known a priori, but the mixture of active processes performed along them is not specified.

The results obtained concern the CPCPs modelling in the class of Place/Transition nets as well as the conditions sufficient for a live net model design. The models considered encompass real-time control schemes and serve as a basis for searching for the best resource scheduling policy.

Directions for future research include an investigation of more efficient conditions for the deadlock prevention, i.e. conditions allowing us to consider a larger state space then it is possible for the actually developed conditions. Also, the investigation of requirements imposed for buffers capacity in order to guarantee CPCPs accomplishment will be of our primary concern.

The application perspective of the results obtained concerns the automated programming of the reliable real-time industrial controllers as well as the designing of computer-aided tools aimed at modelling and performance evaluation of FMSs. We believe that the approach presented may be extended to other areas where deadlock avoidance is critical, such as message routing in data communication networks and multiprogramming operating systems of multiprogrammed computing systems.

References

Banaszak Z. (1988): Synchronization mechanism for competing processes synchronization. - In: R. Trappl (Ed.), *Cybernetics and Systems'88.*–Kluwer Academic Publishers, pp.811–816.

Banaszak Z. and Roszkowska E. (1988): Deadlock avoidance in pipeline concurrent processes. - *Podstawy Sterowania*, 18, Warsaw, pp.3–17.

Banaszak Z. and Lech J. (1988): Laboratory stand of a flexible manufacturing Module. - *Scientific Papers of the Institute of Technical Cybernetics.*– Wroclaw Technical University, 76, pp.13–20, (in Polish).

Banaszak Z. (1988b): Computer-aided scheduling of the automated guided vehicle system. - *Scientific Papers of the Silesian Technical University*, 96, pp.11–20 (in Polish).

Banaszak Z. and Krogh B.H. (1990): Deadlock avoidance in flexible manufacturing systems with concurrently competing process flows. - *IEEE Journal on Robotics and Automation*, v.6, No.6, pp.724–734.

Deitel H.M. (1984). *An Introduction to Operating Systems.*–Massachusetts: Addison-Wesley Publ. Co.

Gold E.M. (1978): Deadlock prediction: easy and difficult cases. - *SIAM J. Comput.*, v.7, No.3, pp.320–356.

Hauschild D. and Valk R. (1987): Safe states in banker-like resource allocation problems. - *Information and Computation.*, No.75, pp.232–263.

Krogh B.H. and Banaszak Z. (1988): Deadlock avoidance in pipeline concurrent processes. - *Preprints of the 22-nd Annual Conference on Information Sciences and Systems.*–Princeton University, Princeton, New Jersey, March 16-18, 1988, Princeton, pp.45–49.

Krogh B.H. and Ekberg G. (1987): Automatic programming of controllers for discrete manufacturing processes. - In: R. Isermann (Ed.) *Prep. 10-th IFAC World Congress.*–Munich. v.4, pp.160–164.

Kus J. and Banaszak Z. (1988): BATRACE wersion 2.2. *Tech. report SPR 18/88.*–Technical University of Wroclaw, Institute of Technical Cybernetics, Wroclaw, (in Polish).

Obermarck R. (1982): Distributed deadlock detection algorithm. - *ACM Trans. on Database Systems*, v.7, No.2, pp.187–208.

Peterson J.I. (1981): *Petri Net Theory and the Modelling of Systems.*–New York: Prentice-Hall, Englewood Cliffs.

Peterson J.L. and Silberschatz A. (1983): *Operating System Concepts.*– Massachusetts: Addison-Wesley Publ. Co.

Raynal M. (1986): *Algorithms for Mutual Exclusion.*– Massachusetts: The MIT Press: Cambridge.

Reisig W. (1982): Petrinetze.–Berlin: Springer-Verlag.