Design of Deadlock Prevention Supervisor in Waterway with Multiple Locks and Canals

Danko Kezić[1], Stjepan Bogdan[2], Josip Kasum[3]

To avoid conflict and deadlock states in waterway with multiple locks and canals, a computer based traffic management system with proper control policy must be applied. The paper proposes a formal method for design of deadlock prevention supervisor by using discrete event theory, multiple reentrant flowlines class of Petri net and P-invariants control places calculation. By using and/or matrix algebra, authors analyze the structural characteristics of Petri net in order to find first and second level deadlocks. First level deadlocks are prevented by maintaining the number of vessels in the critical subsystems below the number of vessels in the critical circuits. A method for second level deadlock prevention, which is based on P-invariants, ensures that the key resources would not be the last available resources in the system. Functionality of the supervisor is verified by a computer simulation using Matlab software with Petri net toolbox and P-timed Petri net model of waterway.

KEY WORDS

~ Waterway traffic management system

~ Supervisory control

~ Deadlock prevention

1, 3 Faculty of Maritime Studies, Zrinsko Frankopanska 38, Split, Croatia

E-mail: danko.kezic@pfst.hr, josip.kasum@pfst.hr

2 Faculty of Electrical Engineering and Computing, Unska 3, Zagreb, Croatia

E-mail: stjepan.bogdan@fer.hr

1. INTRODUCTION

A waterway is any navigable body of water, such as river, lake, sea, ocean, and canal. Some waterways are combination of rivers, lakes and narrow canals with different levels of water. In such waterways, herein named complex waterway system (CWS), vessels must use multiple locks (devices for raising and lowering boats between stretches of water of different levels) to move through the system of locks and canals.

Safe navigation in CWS is very demanding process and needs coordination between crew members aboard vessel and traffic management staff on the ground. Some of the problems that need to be solved by the traffic management staff are: a) How to control traffic in a way that vessels moving in opposite directions make as few stops as possible during the passage through the waterway (maximally permissive control policy)? b) How to resolve possible conflicts in case that more vessels try to acquire particular lock (canal, basin) at the same time? c) How to avoid possible deadlocks in the dense traffic?

To resolve above mentioned problems in situations of dense *traffic in waterway system*, a computer based traffic management system (TMS), which observes and controls vessels in CWS, must be applied. Intelligent traffic management system is also used for real-time traffic management of the urban motorway network (Hernandez, et. al., 2002).

The exact positions of the vessels in CWS can be monitored by using DGPS and AIS on board ship with wireless communication between vessels and TMS. In our approach locks, canals and basins are treated as resources of the CWS. Resource can be *non-shared* (resource that can be occupied by the vessels moving in only one direction), and also *shared* (resource that can be occupied by the vessels moving in the opposite directions). The availability of resources is monitored by various electronic sensors like camera or proximity sensors. Main task of TMS is not only to analyze data from sensors, but also to evaluate traffic situation, and advise the traffic management staff how to resolve ongoing situation. Properly designed TMS should have the properties of an expert that is capable to cope with complex traffic situations in CWS.

The vessels moving through the resources of CWS, can generally be described as a discrete event system (DES), which consists of discrete states and events. Some of these states, such as conflicts and deadlocks, are undesirable (even dangerous). In this view, TMS should implement supervisory policy that prevents execution of some events (entering or leaving lock, canal, basin) in order to restrict the set of reachable discrete states in the system to the set of permissible (safe) states. This can be done by direct control of CWS traffic lights system or throughout a set of recommendations to CWS human operator.

The method of deadlock prevention by control places was developed in (Barkaoui & Abdallah, 1995). An algorithm for deadlock prevention for the ordinary and conservative S^3PR class of Petri nets was developed in (Ezpelta, et al.,1995). The paper from Ezpelta et al. (1995) is usually considered to be the first that uses structural analysis to design monitor-based liveness-enforcing Petri net supervisor for the flexible manufacturing systems (FMS). The algorithm for finding the minimal siphons inside the net as well as the algorithm for deadlock prevention by control places for ordinary Petri nets which do not contain source places was investigated in (Lautenbach & Ridder, 1996). Further, an efficient algorithm for deadlock prevention in the specific class of Petri nets that describes FMS was developed in (Lewis, et al., 1998). A deadlock prevention which uses iterative siphon control method is described in (Iordache & Moody, 2000) and (Kezić, et al., 2006). Similar approaches can be found in (Barkaoui, et al. 1997), (Barkaoui & Petrucci, 1988), (Tricias, et al. 2000), (Tricas, et al., 2005). Deadlock prevention policy based on elementary siphons for FMS is proposed in (Mingming, et. al., 2009). The divide-and-conquer strategy is used in (ZhiWu Li, et. al., 2009) to investigate the deadlock prevention in FMS. Efficient deadlock prevention in Petri nets through the generation of selected siphons is proposed in (Piroddi, et. al., 2009).

This paper presents enhancement of the algorithm presented in (Kezić, et al. 2009) where deadlock avoidance algorithm for river traffic system uses multiple re-entrant flowlines class of Petri net (MRF_1PN) with only one key resource (Bogdan, et al., 1997). Herein we propose a solution in case of CWS with multiple key resources. The solution represents deadlock prevention supervisor in a sense that vessels are stopped only in a case of immediate dangerous situation in dense traffic.

The first task in TMS design is modeling of the traffic system by using MRF_1PN, which consists of disjoint sets of job and resource places. The second task is structural analysis of MRF_1PN, i.e. determination of simple and cyclic circular waits, critical siphons, and finally critical subsystems. To avoid first level deadlocks, it is necessary to control number of vessels in every critical subsystem. In Petri net formalism this can be achieved by adding additional control places which block firing of particular transition and restrict the number of tokens in critical subsystems. For prevention of second level deadlocks one has to take care of so called *key resources*, i.e. the supervisor must ensure that the key resources are not the only available resources in the net.

The paper is organized as follows: section 2 reviews basics of supervisory control and Petri net theory. Section 3 describes *P*-invariant method of control places design. In section 4, a matrix description of MRF1PN is presented and modeling of CWS with MRF1PN is described. A matrix approach to deadlock prevention supervisor design, using MRF_1PN, is shown in section 5. Finally, a case study example of supervisor design for CWS, similar to Panama canal, is given in section 6.

2. BASICS OF SUPERVISORY CONTROL AND PETRI NET

A process can be defined as a set of interdependent tasks or jobs which are necessary to achieve a goal. In this paper, the main goal is to achieve uninterrupted passage of vessels through CWS. The supervisor has to ensure that the process does not get into any of forbidden states and that it performs in accordance with the given requirements (Charbonnier, et al., 2001).

The theory of DES supervisory control deals with the problem of synthesis of the supervisor, which is connected to the given process in closed loop, and which ensures the desired behavior of the whole system. The theory of supervisory control is described in (Boffey, 1982), (Overkamp & Van Schuppen, 1995), (Vaz & Wonham, 1986), (Yamalidou, et al., 1996). The theory originates from the language theory generated by the automata and Petri nets, a useful tool for analyzing DES (Hopcroft & Ullman. 1979). Petri nets formalism is a graphical and mathematical tool adapted to the modeling of the main features of discrete event systems (Gallego et. al, 1996).

The basics of supervisory control can be described using Fig 1. Suppose that the process *G* can be modeled as a DES with the finite set of discrete states and events. Every task or job in the process can be modeled as particular state. The sequence of the jobs in the process *G* causes changing of the states and generates set of events *s*. The behavior of process *G*, as a rule, does not correspond to the specified process requirements (process *G*, for example, may get into a so called deadlock state – the state in which no more events are possible) and therefore it is necessary to "modify" its behavior by introducing the supervisor *S*. The supervisor *S*, which is also DES, is connected into a closed loop with the process *G*. The task of the supervisor *S* is to monitor generated events *s* from the process *G* and, if necessary, block events in the process *G* which can cause forbidden state. In other words, the task of the supervisor *S* is to restrict the set of events generated by the process *G* to the set of allowable events $\gamma = S(s)$. This ensures absence of unallowable forbidden states in process *G*.

In this paper, we are using Petri net theory for modeling process *G*, and designing appropriate supervisor *S*. The advantage of Petri nets as compared to other DES modeling methods is in their rich structure, which enables the analysis of numerous characteristics of the system from the structure of the net itself, and without having to analyze the whole discrete state space. Place - transition P/T Petri net is a 6-tuple (Murata, 1989):

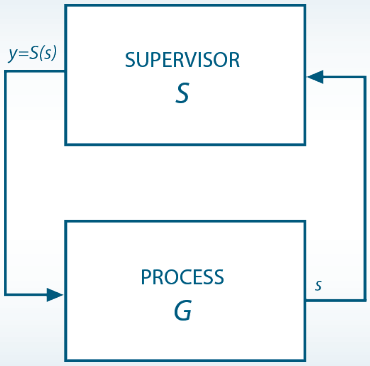

Figure 1. Process and supervisor in closed loop. Source: authors.

$$Q = (P, T, \boldsymbol{I}, \boldsymbol{O}, \boldsymbol{M}, \boldsymbol{m_0}) \qquad (1)$$

where:
$P = \{p_1, p_2, ..., p_n\}$ - set of places,
$T = \{t_1, t_2, ..., t_n\}$ - set of transitions,
$P \cap T = \varnothing$,
$\boldsymbol{I}_{(nxm)} : P \times T \rightarrow \{0,1\}$ - an input incidence matrix,
$\boldsymbol{O}_{(nxm)} : T \times P \rightarrow \{0,1\}$ - an output incidence matrix,
$\boldsymbol{M} : \boldsymbol{I}, \boldsymbol{O} \rightarrow \{1,2,3,...\}$ - is a weight function,
$\boldsymbol{m_0}$ - initial marking.

Places and transitions $v \in P \cup T$ are calling nodes and denote states and events in the DES. Given any node v, let $\bullet v$ and $v \bullet$ denote pre-set and post-set of v, i.e. the set of nodes that have arcs to and from v, respectively. An available resource or an ongoing job in DES is indicated by token in respective place. Transition $t \in T$ is enabled at marking $m(p)$ iff $\forall p \in \bullet t, m(p) \geq w(p,t)$ ($\bullet t$ is a set of input places to transition t, and $w(p,t)$ is weight of the arc between p and t). Transition t that meets enabled condition is free to fire. When transition t fires, all of its input places lose $w(p_i, t)$ tokens, and all of its output places gain $w(t, p_j)$ tokens. In Petri net Q with n transitions and m places, the incidence matrix **A** is an $n \times m$ matrix defined as:

$$\boldsymbol{A} = \boldsymbol{O} - \boldsymbol{I} \qquad (2)$$

where elements a_{ij}^+ and a_{ij}^- of $\boldsymbol{O}$ and $\boldsymbol{I}$ are:
$a_{ij}^+ = w(p_i, t_j)$ if $(p_i, t_j) \in \boldsymbol{I}$ and $a_{ij}^+ = 0$ otherwise,
$a_{ij}^- = w(t_j, p_i)$ if $(t_j, p_i) \in \boldsymbol{O}$ and $a_{ij}^- = 0$ otherwise.

The matrices $\boldsymbol{I}$ (input matrix) and $\boldsymbol{O}$ (output matrix) provide a complete description of the structure of a Petri net. If there are no self loops $\bullet p \cap p \bullet = \varnothing$, the structure can be described by incidence matrix $\boldsymbol{A}$. The incidence matrix allows an algebraic description of the evolution of Petri net:

$$\boldsymbol{m_{k+1}} = \boldsymbol{m_k} + \boldsymbol{A}^T \cdot \sigma \qquad (3)$$

where:
$\boldsymbol{A}$ - incidence matrix,
σ - firing vector.

The firing vector σ is composed of non-negative integers that correspond with the number of times a particular transition has been fired between markings $\boldsymbol{m_k}$ and $\boldsymbol{m_{k+1}}$.

A PN is said to be live if, no matter what marking has been reached from the initial marking $\boldsymbol{m_0}$, it is possible to ultimately fire any transition of the net by progressing through some further firing sequences. A transition $t \in T$ is said to be dead at $\boldsymbol{m}$ if there exists no $\boldsymbol{m}' \in \Re(\boldsymbol{m})$ that enables it, with $\Re(\boldsymbol{m})$ defined as the set of markings reachable from $\boldsymbol{m}$. A marking $\boldsymbol{m}$ is said to be dead if no $t \in T$ is enabled at $\boldsymbol{m}$. A place $p \in P$ is said to be dead or deadlocked at $\boldsymbol{m}$ if $\boldsymbol{m} = \boldsymbol{m}' = 0$ for all $\boldsymbol{m}' \in \Re(\boldsymbol{m})$. P invariant corresponds to the set of places whose weighted token count remains constant for all possible markings. P invariant $\boldsymbol{P}$ can be found by solving equation:

$$\boldsymbol{A} \cdot \boldsymbol{P} = 0 \qquad (4)$$

Siphon *S* is the set of Petri net places for which holds that each transition having an output arc from *S* also has an input arc into the *S* ($\bullet S \subseteq S \bullet$). Trap *T* is the set of places for which it holds that each transition having an input arc into *T* also has an output arc from T ($T \bullet \subseteq \bullet T$). Once the trap becomes marked, it will always be marked for all future reachable markings. Once the siphon becomes empty, it will always remain empty (Murata, 1989).

A reachability set or reachability tree shows the set of all possible markings reachable from $\boldsymbol{m_0}$ and displays every possible state that can occur in the Petri net after firing all transitions. It is possible to see some important PN properties like boundness (no capacity overflow), liveness (absence of deadlock), conservativeness (conservation of no consumable resources), and reversibility (cyclic behavior) from the reachability

doi: 10.7225/toms.v01.n01.004

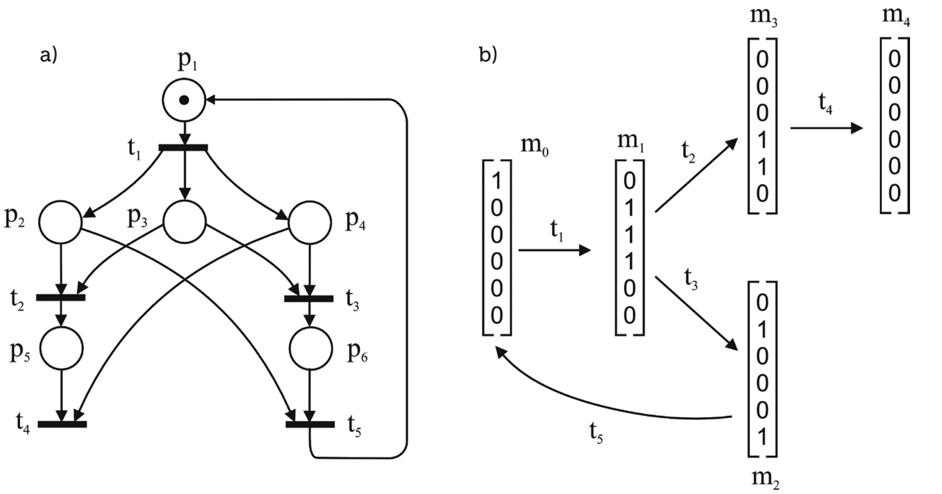

Figure 2.
P/T Petri net
a) and its reachability tree b).
Source: authors.

tree. An algorithm for calculating reachability tree is shown in (Kezić, 2004).

Example:

A simple P/T Petri net with 6 places (circles) and 5 transitions (bars) is shown in Fig.2 a. Reachability tree is shown in Fig. 2b.

Petri net in Fig 2a) is safe because the maximum number of tokens in the places is 1, and is partially reversible because it is possible to reach initial state $\boldsymbol{m_0}$ after firing transitions $\{t_1, t_3, t_5\}$. Firing the t_4 from the state $\boldsymbol{m_3}$ cause deadlock state $\boldsymbol{m_4}$. From the state $\boldsymbol{m_4}$ it is not possible to reach any other state.

The net in Fig 2 is not live. There are no place invariants in the net. The incidence matrix $\boldsymbol{A}$ of Petri net in Fig. 2 is:

$$\boldsymbol{A} = \begin{bmatrix} 1 & -1 & -1 & -1 & 0 & 0 \\ 0 & 1 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 1 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ -1 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$

3. P-INVARIANT BASED CONTROL PLACE CALCULATION

Petri net model of a process, so called Process Petri net (PPN), may contain forbidden states. These states can be avoided by using control places, which must be added and connected to the PPN. These control places form Supervisor Petri net (SPN). P-invariant method for control place calculation is one of techniques for SPN design. Short overview of this technique is shown in this paper. More can be found in (Iordache & Moody, 2000).

Suppose that process G is DES and is modeled by a PPN described by process incidence matrix $\boldsymbol{A_p}=[a_{i,j}]_{nxm_p}$.

The supervisor S, in the form of SPN, prevents occurrence of forbidden states M_F, by applying constraints to the set of all reachable states of PPN. SPN can be described by supervisor incidence matrix $\boldsymbol{A_p}=[a_{i,j}]_{nxm_c)}$.
To connect process G and supervisor S in closed loop (Fig 1), SPN and PPN connect together and form a new Composite Petri net (CPN) without forbidden states.
The CPN incidence matrix $\boldsymbol{A_{pc}}=[a_{i,j}]_{nx(m_p+m_c)}$ describes a topology of composite Petri net. Each supervisor control place defines a constraint over the set of reachable states of the PPN. The constraint can be expressed in the form of linear inequality:

$$\sum_{i=1}^{m} l_i m(p_i) \leq \beta \qquad (5)$$

in which:
$m(p_i)$ - number of tokens in place p_i,
$l_i \beta$ - integer constants.

The set of inequalities (5) can be transformed into matrix equation:

$$\boldsymbol{L} \cdot \boldsymbol{m_p} + \boldsymbol{m_c} = \boldsymbol{b} \qquad (6)$$

in which:
$\boldsymbol{L}$ - constraints matrix $g_c \times m_p$,
$\boldsymbol{b}$ - vector $m_c \times 1$,
$\boldsymbol{m_p}$ - marking vector of PPN $m_p \times 1$,
$\boldsymbol{m_c}$ - marking vector of SPN $m_c \times 1$.
g_c - number of constraints.

Note that the number of control places m_c must be equal to the number of constraints g_c, so $g_c = \boldsymbol{m_c}$

P invariant $\boldsymbol{P}$, defined by relation (4), must satisfy the requirements of equation (6) so we can calculate supervisor incidence matrix $\boldsymbol{A_c}$ and supervisor initial marking $\boldsymbol{m_{c0}}$ as:

$$\boldsymbol{A_c} = -\boldsymbol{A_p} \cdot \boldsymbol{L}^T \quad (7)$$

$$\boldsymbol{m_{c0}} = \boldsymbol{b} - \boldsymbol{L} \cdot \boldsymbol{m_{p0}} \quad (8)$$

in which:
$\boldsymbol{m_{c0}}$ - SPN initial marking,
$\boldsymbol{m_{p0}}$ - PPN initial marking.

Matrix $\boldsymbol{A_c}$ and vector $\boldsymbol{m_{c0}}$ completely determine initial marking of control places, as well as the connection between each control place and other places of PPN.

The complete CPN design is partitioned in the following steps:
1. Determine a PPN. From the PPN it is possible to define process incidence matrix $\boldsymbol{A_p}=[a_{i,j}]_{nxmp)}$ and $\boldsymbol{m_{p0}}$,
2. Define the set of constraints of type (6) in order to reduce the set of reachable markings to allowed states,
3. Calculate $\boldsymbol{A_c}$ and $\boldsymbol{m_{c0}}$ from equations (7) and (8),
4. Design a CPN from the composite incidence matrix $\boldsymbol{A_{pc}}$. Check the set of reachable markings.
5. If there are forbidden states in CPN, go to step 2.

4. MODELING CWS WITH MRFPN

Deadlock prevention supervisor design begins with the traffic system modeling by using MRF1PN, which is a subclass of P/T Petri net specially designed for analysis of multiple re-entrant flowlines flexible manufacturing systems (MRF system).

4.1 MRF$_1$ Petri net

In the MRF$_1$PN, each part type $k \in \Pi$ is characterized by predetermined sequence of jobs $J^k = \left\{J_1^k, J_2^k, ..., J_{L_k}^k\right\}$, with at least one resource for each job (L_k is the number of jobs for particular part type k). Let R denote the set of system resources, with each $r \in R$ a pool of multiple copies of a given resource. Places P are divided in the MRF$_1$PN as $P = R \cup J \cup J_{in} \cup J_{out}$ with R, J_{in}, J_{out} and J as the set of places respectively representing the availability of resources, units arrivals and finished units, and J as the set of places representing the ongoing jobs. The set of transitions T can be partitioned as $T = \cup_{k\in\Pi} T^k$, where $T^k = \{t_1^k, t_2^k, ..., t_{L_k+1}^k\}$, with $t_i^k = \bullet J_i^k = J_{i-1}^k \bullet$, for $i \notin \{1, L_k\}$; while $t_1^k = \bullet J_1^k = J_{in}^k \bullet$ and $t_{L_k+1}^k = \bullet J_{out}^k = J_{L_k}^k \bullet$. Transition t is said to be job (resource) enabled if $m(\bullet t \cap J) > 0$ and $m(\bullet t \cap R) > 0$. For any $r \in R$, define the job set $J(r)$ as the set of jobs using r, and resource loop $L(r) = r \cup J(r)$. Given a set of resources $Q \subset R$, define the job set of Q as $J(Q) = \cup_{r\in Q} J(r)$. We denote $R(J_i^k)$ as the set of resources used by job J_i^k.

MRF$_1$PN satisfies following conditions: (i) $\forall p \in P$, $\bullet p \cap p \bullet = \varnothing$; (ii) $\forall k \in \Pi$, $t_1^k \bullet \cap P \setminus J = \varnothing$ and $\bullet t_{L_{k+1}}^k \cap P \setminus J = \varnothing$; (iii) $\forall J_i^k \in J$, $\left|R(J_i^k)\right| = 1$ and $R(J_i^k) \neq R(J_{i+1}^k)$; (iv) $\forall J_i^k \in J$, $\left|J_i^k \bullet\right| = 1$; (v) $\forall t_i^k \in J$, $\left|\bullet t_i^k \cap J\right| \leq 1$; (vi) $\forall r \in R$, $\left|J(r)\right| \geq 1$. This means that (i) there are no self loops, (ii) each unit-path has a well defined beginning and an end, (iii) every job requires one and only one resource with no two consequent jobs using the same resource, (iv) and (v), there are no choice jobs and no assembly jobs, (vi) there are shared resources. In MRF$_1$PN, for any $r \in R$, $J(r) = r \bullet\bullet \cap J = \bullet\bullet r \cap J$ and $R(J_j^k) = \bullet\bullet J_j^k \cap R = J_j^k \bullet\bullet \cap R$ For any two $r_i, r_j \in R$, r_i is said to wait r_j, denoted $r_i \to r_j$, if the availability or r_j is a immediate requirement for the release or r_i, i.e., $\bullet r_i \cap r_j \bullet \neq \varnothing$, or equivalently, if there exists at least one transitions $t \in \bullet r_i \cap r_j \bullet$.

Any set of resources is called *circular wait* CW, if among the set of resources $r_a, r_b, ..., r_w$ exist wait relations among them such that $r_a \to r_b \to ... \to r_w$ and $r_w \to r_a$. CW relations are characteristic among shared and nonshared resources in MRF$_1$PN and contain at least one shared resource. *Simple circular wait* (SCW) is composed of different resources while *cyclic circular wait* (CCW) is composed of unions of nondisjoint simple CWs. *Deadlock* in the MRF$_1$PN is connected with the system condition called *circular blocking* CB, which is a consequence of the existence of circular wait relations CW among resources in the system. A CW C is said to be in CB if (i) $m(C) = 0$; and (ii) for each $r \in C$, $\forall p \in J(r)$ with $m(p) \neq 0$, $p \bullet \in C \bullet$. Avoiding CB is necessary but generally not sufficient for deadlock-free dispatching policy.

To prevent deadlock in MRF$_1$PN we must first avoid CB conditions, which are closely related to the critical siphon. A *critical siphon* S is a minimal siphon that does not contain any resource loop. The next step is to find sets of job places, so called *critical subsystems* $J_0(C)$, A CW C is in CB at any $\boldsymbol{m_0} \in \Re(\boldsymbol{m})$ if and only if particular critical siphon becomes empty ($m(S_C) = 0$). The critical siphon is empty if and only if $m(J_0(C)) = m_0(C)$; or equivalently, to avoid deadlock we must ensure that the $m(S_C) \neq 0$ by applying constraint $m(J_0(C)) < m_0(C)$ to the set $\Re(\boldsymbol{m_0})$. The token sum in the critical subsystem $J_0(C)$ must be limited above value $m_0(C) - 1$. To achieve this, we must connect control places to PPN which form P-invariant with critical subsystems $J_0(C)$.

The deadlock which occurs due to improper "loading" of critical systems $J_0(C)$ is called *first level deadlock*. The system for which avoiding CB is a necessary and sufficient condition for avoiding deadlock is so called *regular system*. Regular systems contain only first level deadlocks. The above is not true for irregular systems. The *irregular system* contains a key resource. It must be noted that when system contains a key resource, the system may have so called cyclic circular wait relation CCW, that, in particular circumstances, could lead in so called second level deadlock. A *second level deadlock* is a state that is not currently a deadlock, but leads to a deadlock after the next transitions. To avoid second level deadlocks, one must find key resources and apply such control policy that key resource does not remain the last available resource in CCW.

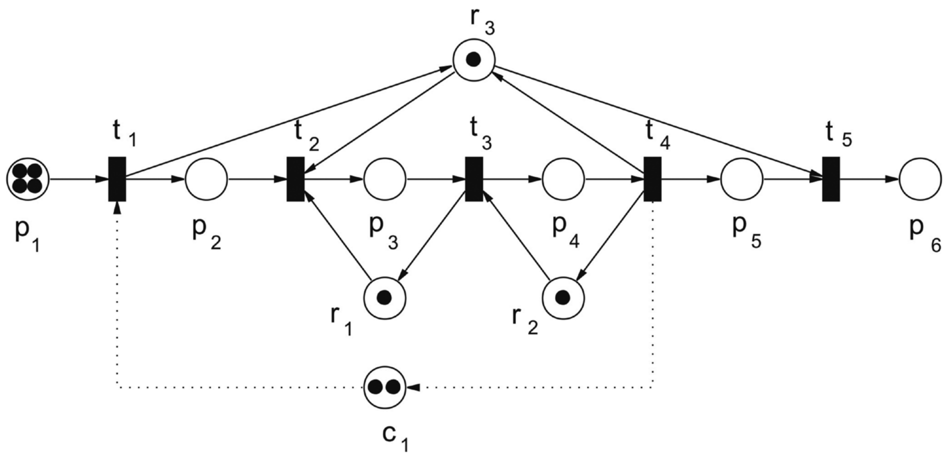

Figure 3.
MRF1PN class of Petri net.
Source: authors.

This paper focuses on the design of a deadlock prevention supervisor for CWS system which contains only first and second level deadlocks. However, there are other systems with higher level deadlocks, and the deadlock free supervisor design for such systems is presented in (Lee & Tilbury, 2007)

Example:

Fig 3. which shows MRF$_1$PN with one input and one output place $J_{in} = p_1$ and $J_{out} = p_6$. There is a set of job places $J = \{p_2, p_3, p_4, p_5\}$, a set of resource places $R = \{r_1, r_2, r_3\}$, one SCW $C = \{r_1, r_2, r_3\}$ and one critical subsystem $J_0(C) = \{p_2, p_3, p_4\}$. Initial marking of SCW is $m_0(C) = 3$.

Matrix $\boldsymbol{A_p}$ and $\boldsymbol{m_{p0}}$ of PPN (MRF$_1$PN in Fig 3 without c_1) are:

$$\boldsymbol{A_p} = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 1 & -1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & -1 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & -1 & -1 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{m_{p0}} = \begin{bmatrix} 4 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}^T$$

To avoid deadlock, one must apply constraint $m(p_2) + m(p_3) + m(p_4) \le 2$, hence:
$\boldsymbol{L} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$, $\boldsymbol{b} = \begin{bmatrix} 2 \end{bmatrix}$.
By using (7) and (8) it is possible to calculate control matrix $\boldsymbol{A_c}$ and $\boldsymbol{m_{s0}}$:
$\boldsymbol{A_c} = \begin{bmatrix} 1 & 0 & 0 & -1 & 0 \end{bmatrix}^T$
$\boldsymbol{m_{c0}} = \begin{bmatrix} 2 \end{bmatrix}$

From $\boldsymbol{A_c} = \begin{bmatrix} 1 & 0 & 0 & -1 & 0 \end{bmatrix}^T$ it is clear that SPN has control place c_1, with one input arc from t_4 and one output arc to t_1. Control place c_1 maintains the number of tokens in the critical subsystem to maximum 2. It should be noted that the system in Fig 3. is regular as there are no key resources and second level deadlocks.

One way to calculate supervisors for complex systems is to describe MRF$_1$PN by system matrices. There are two sets of system matrices: $\boldsymbol{F_u}, \boldsymbol{F_y}, \boldsymbol{F_v}, \boldsymbol{F_r}$ and $\boldsymbol{S_u}, \boldsymbol{S_v}, \boldsymbol{S_r}, \boldsymbol{S_y}$. Matrices $\boldsymbol{F}$ capture conditions that must be fulfilled before firing of transitions, while matrices $\boldsymbol{S}$ are responsible for actions after firing of transitions. A number of rows of $\boldsymbol{F_u}, \boldsymbol{F_y}, \boldsymbol{F_v}, \boldsymbol{F_r}$ define the number of transitions, while the number of columns defines the number of input places, jobs, resources and output places respectively. A number of columns of $\boldsymbol{S_u}, \boldsymbol{S_v}, \boldsymbol{S_r}, \boldsymbol{S_y}$ define the number of transitions, while the number of rows defines the number of input places, jobs, resources and output places respectively. Each entry $(f_r)_{ij}$ in the resource-requirements matrix $\boldsymbol{F_r}$ is associated with an arc connecting a place, representing resource availability, with the corresponding transition; each entry $(s_r)_{ij}$ in the resource-release matrix $\boldsymbol{S_r}$ expresses the connections between transitions and places that hold tokens when resources are idle. Correspondingly, each entry $(f_v)_{i,j}$ and $(s_v)_{i,j}$ in job-sequencing matrix $\boldsymbol{F_v}$ and job-start matrix $\boldsymbol{S_v}$ represent arcs connecting transitions and places associated operations executed by resources. The input matrix $\boldsymbol{F_u}$ portrays output arcs from input places, while output matrix $\boldsymbol{S_y}$ depicts input arcs to output places. Since we assume that input places are source places (places with no input transitions) and output places are sink places (places with no output transitions), matrices $\boldsymbol{F_y}$ and $\boldsymbol{S_u}$ are null matrices, $\boldsymbol{F_y} = \boldsymbol{S_u} = \begin{bmatrix} 0 \end{bmatrix}$. As a result, PN input and output incidence matrices $\boldsymbol{I}$ and $\boldsymbol{O}$ can be obtained from the system matrices:

$$\boldsymbol{I} = \begin{bmatrix} \boldsymbol{F_u} & \boldsymbol{F_v} & \boldsymbol{F_r} & \boldsymbol{F_y} \end{bmatrix} = \boldsymbol{F} \quad (9)$$

$$\boldsymbol{O} = \begin{bmatrix} \boldsymbol{S_u^T} & \boldsymbol{S_v^T} & \boldsymbol{S_r^T} & \boldsymbol{S_y^T} \end{bmatrix} = \boldsymbol{S^T} \quad (10)$$

System matrices $\boldsymbol{F_u}, \boldsymbol{F_y}, \boldsymbol{F_v}, \boldsymbol{F_r}$ and $\boldsymbol{S_u}, \boldsymbol{S_v}, \boldsymbol{S_r}, \boldsymbol{S_y}$ can be derived directly from the MRF$_1$PN.

5. MATRIX APPROACH OF DEADLOCK PREVENTION SUPERVISOR DESIGN USING MRF$_1$PN

The procedure for finding deadlock prevention supervisor for PPN can be divided in eight steps:

Step 1: Find all resource loops $L(r)$ via computing their covering binary P-invariants. The binary basis for P-invariants is given by the columns of the matrix $\boldsymbol{P}$:

$$\boldsymbol{P} = \begin{bmatrix} -(\hat{S}_v^T - \hat{F}_v)^{-1} \cdot (\hat{S}_r^T - \hat{F}_r)^{-1} \\ I_{r\times r} \end{bmatrix} \tag{11}$$

where:

$\boldsymbol{I}_{r\times r}$ - identity matrix with r resources in the system

Matrices $\hat{\boldsymbol{F}}_{\boldsymbol{v}}$ and $\hat{\boldsymbol{F}}_{\boldsymbol{r}}$ are formed by deleting rows that correspond to the terminal transitions. Matrices $\hat{\boldsymbol{S}}_{\boldsymbol{v}}$ and $\hat{\boldsymbol{S}}_{\boldsymbol{r}}$ are formed by deleting columns that correspond to the terminal transitions. Terminal transition have output arc to J_{out}.

Step 2: Find wait relation matrix $\boldsymbol{G}_{\boldsymbol{w}}$, all SCW and CCW. Wait relations are captured by the wait relation matrix:

$$\boldsymbol{G}_{\boldsymbol{w}} = \boldsymbol{S}_{\boldsymbol{r}} \otimes \boldsymbol{F}_{\boldsymbol{r}} \tag{12}$$

Where matrix operation $\otimes$ is defined in and/or algebra, i.e. standard addition and multiplication of matrices elements are replaced by the logical "and" and "or", respectively.

Having obtained matrix $\boldsymbol{G}_{\boldsymbol{w}}$, there are standard efficient techniques of polynomial complexity, such as string algebra, for identifying matrices $\boldsymbol{C}$ and $\boldsymbol{\emptyset}$. From $\boldsymbol{C}_{r\times(SCW+CCW)}$ it is possible to determine all C_i, and from $\boldsymbol{\emptyset}_{SCW\times(SCW+CCW)}$ it is possible to detect which SCW-s are involved in particular CCW. Columns of matrix $\boldsymbol{C}$ which contain non shared resources are denoted by vector $\boldsymbol{C}_{\boldsymbol{i}}$, and those containing shared resources are denoted by vector $\boldsymbol{C}_{\boldsymbol{si}}$ (Bogdan, et al., 2006).

Step 3: Find critical siphons matrix $\boldsymbol{S}_{\boldsymbol{C}_i}$ and critical subsystem matrix $\boldsymbol{J}_{\boldsymbol{0}}(\boldsymbol{C})$ by using equations:

$$\boldsymbol{S}_{\boldsymbol{C}_i} = \begin{bmatrix} \boldsymbol{c}_{si}^T \otimes \boldsymbol{S}_r \otimes \boldsymbol{F}_v \wedge \overline{\boldsymbol{c}_i^T \otimes \boldsymbol{F}_r^T \otimes \boldsymbol{F}_v} \\ \boldsymbol{c}_i \end{bmatrix} \tag{13}$$

$$\boldsymbol{J}_{\boldsymbol{0}}(\boldsymbol{C}) = \boldsymbol{P} \otimes \boldsymbol{C} \wedge \overline{\boldsymbol{S}_c} \tag{14}$$

Where matrix operation $\wedge$ denotes element-by element logical "and" operation.

Columns of matrix $\boldsymbol{S}_{\boldsymbol{C}_i}$ are critical siphons, and column of matrix $\boldsymbol{J}_{\boldsymbol{0}}(\boldsymbol{C})$ are critical subsystem.

The DES modeled by MRF$_1$PN can be regular or irregular. For regular system, only condition for deadlock free policy is that the token count in the critical subsystem must be controlled to ensure the system stability in sense of deadlock. If the system is irregular, than the second level deadlock can arise, and one must find key resources, which can be done by using next step.

Step 4: Key resources can be identified by analyzing interconnections of SCWs and their siphons. To confirm the existence of key resources, one must determine presence of CCW loops. These structures specify a particular sharing among circular waits, and are a requisite for the existence of key resources. To find CCWs among all CWs in the system, one must calculate $\boldsymbol{C}_{\boldsymbol{cw}}$:

$$\boldsymbol{C}_{\boldsymbol{cw}} = (\boldsymbol{T}_{\boldsymbol{S_c}}^{-T} \otimes \boldsymbol{T}_{\boldsymbol{S_c}}^{+})^T \wedge (\boldsymbol{T}_{\boldsymbol{S_c}}^{-T} \otimes \boldsymbol{T}_{\boldsymbol{S_c}}^{+}) \tag{15}$$

where:

$\boldsymbol{T}_{\boldsymbol{S_c}}^{-T} = \left(\boldsymbol{v}_{oc}^T \otimes \boldsymbol{S}_v - \boldsymbol{v}_{oc}^T \otimes \boldsymbol{S}_v \wedge \boldsymbol{v}_{oc}^T \otimes \boldsymbol{F}_v^T\right)$ - is matrix containing transitions which decrease token counts in every critical siphon and

$\boldsymbol{T}_{\boldsymbol{S_c}}^{+} = \left(\boldsymbol{v}_{oc}^T \otimes \boldsymbol{S}_v - \boldsymbol{v}_{oc}^T \otimes \boldsymbol{S}_v \wedge \boldsymbol{v}_{oc}^T \otimes \boldsymbol{F}_v^T\right)$ - is matrix containing transitions which increase token counts in every critical siphon,

$\boldsymbol{v}_{\boldsymbol{oc}}$ - critical subsystems matrix.

When $\boldsymbol{C}_{\boldsymbol{cw}} = [0]$ the system is *regular*, otherwise an element $C_{CW}(i,j)=1$ indicates that C_i and C_j form a CCW. Obviously $\boldsymbol{C}_{\boldsymbol{cw}}$ is symmetric matrix.

To identify the key resources we must apply the following straightforward matrix formula:

$$\boldsymbol{R}_{\boldsymbol{CCW}} = \left(\boldsymbol{F}_r^T \otimes \boldsymbol{T}_{\boldsymbol{CCW}}^{+}\right) \wedge \left(\boldsymbol{F}_r^T \otimes \boldsymbol{T}_{\boldsymbol{CCW}}^{-}\right) \tag{16}$$

where:

$\boldsymbol{T}_{\boldsymbol{CCW}}^{-} = \left(\boldsymbol{T}_{\boldsymbol{S_c}}^{+} \otimes \boldsymbol{C}_{\boldsymbol{CW}}\right) \wedge \boldsymbol{T}_{\boldsymbol{S_c}}^{+}$ - matrix containing transitions which decrease token counts in CCWs

$\boldsymbol{T}_{\boldsymbol{CCW}}^{+} = \left(\boldsymbol{T}_{\boldsymbol{S_c}}^{-} \otimes \boldsymbol{C}_{\boldsymbol{CW}}\right) \wedge \boldsymbol{T}_{\boldsymbol{S_c}}^{+}$ - matrix containing transitions which increase token counts in CCWs

Matrix $\boldsymbol{R}_{\boldsymbol{CCW}}$ provides key resources which are shared with other CWs in one or more CCW. If this matrix is zero, there are no key resources in the system.

Step 5: To avoid first level deadlock, one must define a set of constraints of type (5) to ensure that the token count in each critical subsystem $J_0(C)$ remains below $m_0(C)-1$. Using *P* invariant method (section 3), calculate a control place for each $J_0(C)$ and add to PPN to derive CPN.

Step 6: After identifying all key resources in step 4, one must find all second level deadlocks in the CPN derived in step 5. These second level deadlocks arise when one or more key resources become last available resources in the net. To find second level deadlocks, one must calculate reachability tree (Kezić, 2004).

Step 7: To avoid second level deadlocks in step 6, the constraints of type (5) must be applied to the CPN. The new control places can be calculated using P-invariant method

doi: 10.7225/toms.v01.n01.004

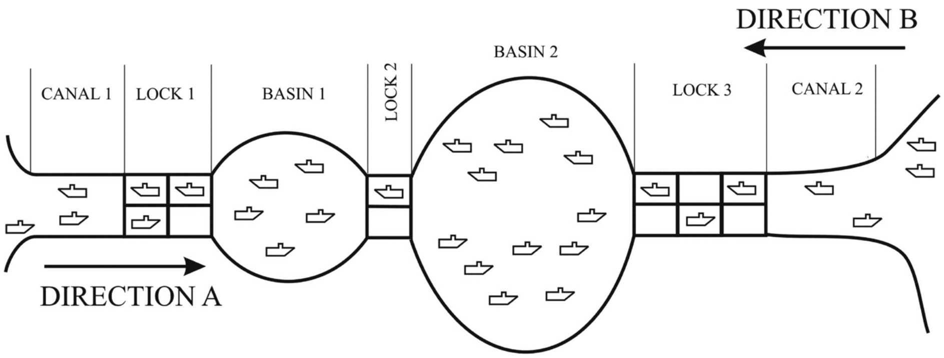

Figure 4.
Complex waterway system.
Source: authors.

described in section 3 and added to Petri net to derive final CPN for irregular system.

Step 8: Find reachability tree of the CPN derived in step 7. If there are new deadlocks go to step 5, otherwise the algorithm ends and final deadlock free CPN is found.

6. DEADLOCK AVOIDANCE IN WATERWAY WITH MULTIPLE LOCKS AND CANALS – CASE STUDY

This chapter deals with a supervisor design of the CWS (Fig. 4). This example will clarify the theory presented in previous sections. The presented case study example is very similar to the Panama canal. However, the above theory is applicable for more complex systems.

The CWS in Fig.4 connects two oceans, and consists of 2 canals (Cl_1 , Cl_2), 3 double locks (L_1 , L_2 , L_3) for lifting or lowering the vessels, 2 lakes or basins (B_1 , B_2). Lake B_1 is above sea level, and lake B_2 is above water level of B_1 . The vessels in direction A are moving thought the Cl_1 , lift in lock L_1 , move and wait for the availability of the lock L_2 in the lake B_1 . Then lift in the lock L_2 to the level B_2 , move toward the lock L_3 and lower to the sea level. The procedure for direction of B is inversed. The vessels can move through the CWS using their own propulsion, tugboats or towing vehicles. All canals, locks and basins represent resources of a CWS. The vessels in both directions share canals Cl and basins B . All locks L are one, two or three stages double locks with one side for direction A, and the other for direction B.

Number of vessels in resources (capacity of resources) is, as a rule, limited due to the various reasons (numbers of available tugboats, weather conditions, water and sea conditions etc). If a particular resource is occupied in a moment of time, and if there are vessels waiting to use them, then these vessels wait for the availability of the occupied resource at the exit of the resource where they are in the moment of time. When the resource becomes available, it is occupied by awaiting vessels. The capacities $Cap(r)$ of canals and basins are $Cap(Cl_1, Cl_2) = 4$, $Cap(B_1) = 5$, $Cap(B_2) = 10$. The capacities of locks in direction A are: $Cap(L_1) = 2$, $Cap(L_2) = 1$, $Cap(L_3) = 3$ (same capacity in direction B).

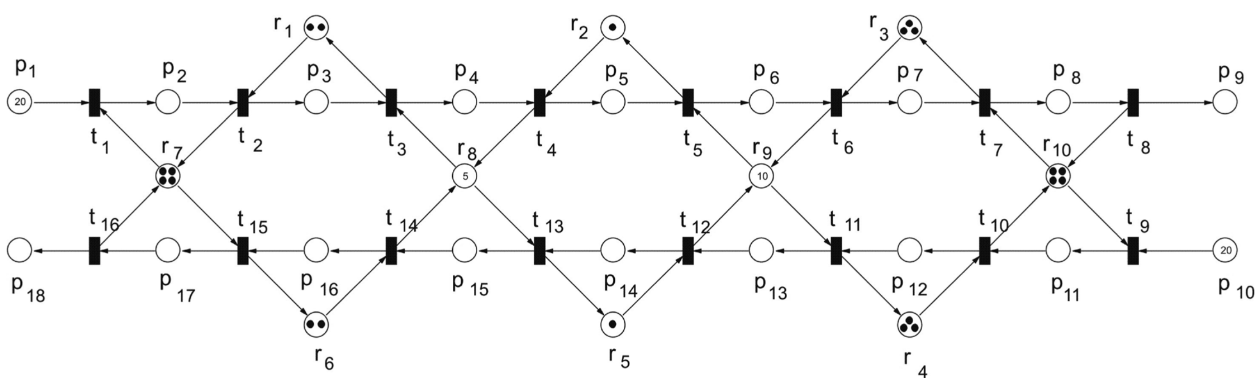

Figure 5.
Process Petri net of CWS.
Source: authors.

The first step which must be taken is to make MRF$_1$PN model of CWS. Figure 5 shows PPN of CWS, and Table 1 describes places belonging to PPN. Tokens in input places $\{p_1, p_{10}\}$ represent the vessels waiting for entering the system, and the tokens in the set of output places $\{p_9, p_{18}\}$ represent the vessels leaving the system. The set of all places that represent jobs in the system (the number of tokens in a job place represent the number of vessels in particular resource) are $\{p_2, ..., p_{17}\}$, and the set of places that represent availability of resources is $\{r_1, ..., r_{10}\}$ (the number of tokens in a resource place represents the capacity of particular resource).

Table 1 shows description of the places p_i in the PPN (Fig. 5), their initial marking $m_o(p_i)$, and time T_i in hours associated to the places (simulation in Fig 7, 8). Places $\{p_1, ..., p_9, r_1, r_2, r_3\}$ describe moving in direction A, places $\{p_{11}, ..., p_{18}, r_4, r_5, r\}$ describe moving in direction B. Places $\{r_7, ..., r_{10}\}$ are shared resources.

There are two problems that must be solved. The first problem is a conflict, and the second problem is deadlock. The conflict arises when vessels from both directions try to occupy the same shared resources $\{r_7, ..., r_{10}\}$ with limited capacity. In this situation the 4 pairs of transitions (t_1, t_{15} and/or t_3, t_{13} and/or t_5, t_{11} and/or t_7, t_9) can be in conflict (both transitions are enabled at the same time). A conflict free supervisor disables one of the transitions in conflict. Firing both of the transitions in conflict cannot occur simultaneously.

The second problem is how to design the deadlock free supervisor. The supervisor is required to be the maximally permissible i.e. not hindering the passage of the vessels. To achieve this we must apply matrix approach described in section 6. Here are the results:

Step 1: The *P* invariants can be calculated applying (6). There are 10 *P*-invariants in the net: $P_1 = \{p_3, r_1\}$, $P_2 = \{p_5, r_2\}$, $P_3 = \{p_7, r_3\}$ $P_4 = \{p_{12}, r_4\}$, $P_5 = \{p_{14}, r_5\}$, $P_6 = \{p_{16}, r_6\}$, $P_7 = \{p_2, p_{17}, r_7\}$, $P_8 = \{p_4, p_{15}, r_8\}$, $P_9 = \{p_6, p_{13}, r_9\}$, $P_{10} = \{p_8, p_{11}, r_{10}\}$

Step 2: Applying (12) and string algebra we can find 3 SSW, $C_1 = \{r_1, r_6, r_7, r_8\}$, $C_2 = \{r_2, r_5, r_8, r_9\}$, $C_3 = \{r_3, r_4, r_9, r_{10}\}$, and 3 CCW: $C_4 = C_1 + C_2 = \{r_1, r_2, r_5, r_6, r_7, r_8, r_9\}$, $C_5 = C_2 + C_3 = \{r_2, r_3, r_4, r_5, r_8, r_9, r_{10}\}$, $C_5 = C_1 + C_2 + C_3 = \{r_1, r_2, r_3, r_4, r_5, r_6, r_7, r_8, r_9, r_{10}\}$

Step 3: By applying (13) and (14) it is possible to find all critical siphons and critical subsystems.

There are 6 critical siphons:

$S_{C_1} = \{r_1, r_6, r_7, r_8, p_4, p_{17}\}$,
$S_{C_2} = \{r_2, r_5, r_8, r_9, p_6, p_{15}\}$,
$S_{C_3} = \{r_3, r_4, r_9, r_{10}, p_8, p_{13}\}$,
$S_{C_4} = \{r_1, r_2, r_5, r_6, r_7, r_8, r_9, p_6, p_{17}\}$,
$S_{C_5} = \{r_2, r_3, r_4, r_5, r_8, r_9, r_{10}, p_8, p_{15}\}$,
$S_{C_6} = \{r_1, r_2, r_3, r_4, r_5, r_6, r_7, r_8, r_9, r_{10}, p_8, p_{17}\}$
and 6 critical subsystems

Table 1.
Description of the PPN in Figure 5.
Source: authors.

p_i	Description	mo (p_i)	T_i [h]
$\{p_1\}$	Waiting for Cl_1	20	0
$\{p_2\}$	Vessel is in Cl_1	0	0,9
$\{p_3\}$	Vessel is L_1	0	0,784
$\{p_4\}$	Vessel is in B_1	0	0,78
$\{p_5\}$	Vessel is in L_2	0	0,39
$\{p_6\}$	Vessel is in B_2	0	3,77
$\{p_7\}$	Vessel is in L_3	0	1,69
$\{p_8\}$	Vessel is in Cl_2	0	0,9
$\{p_9\}$	Passed CWS	0	0
$\{p_{10}\}$	Waiting for Cl_2	20	0
$\{p_{11}\}$	Vessel is in Cl_2	0	0,9
$\{p_{12}\}$	Vessel is in L_3	0	1,69
$\{p_{13}\}$	Vessel is in B_2	0	3,77
$\{p_{14}\}$	Vessel is in L_2	0	0,39
$\{p_{15}\}$	Vessel is in B_1	0	0,78
$\{p_{16}\}$	Vessel is in L_1	0	0,784
$\{p_{17}\}$	Vessel is in Cl_1	0	0,9
$\{p_{18}\}$	Passed CWS	0	0
$\{r_1\}$	L_1 is available	2	0
$\{r_2\}$	L_2 is available	1	0
$\{r_3\}$	L_3 is available	3	0
$\{r_4\}$	L_3 is available	3	0
$\{r_5\}$	L_2 is available	1	0
$\{r_6\}$	L_1 is available	2	0
$\{r_7\}$	Cl_1 is available	4	0
$\{r_8\}$	B_1 is available	5	0
$\{r_9\}$	B_2 is available	10	0
$\{r_{10}\}$	Cl_2 is available	4	0

doi: 10.7225/toms.v01.n01.004

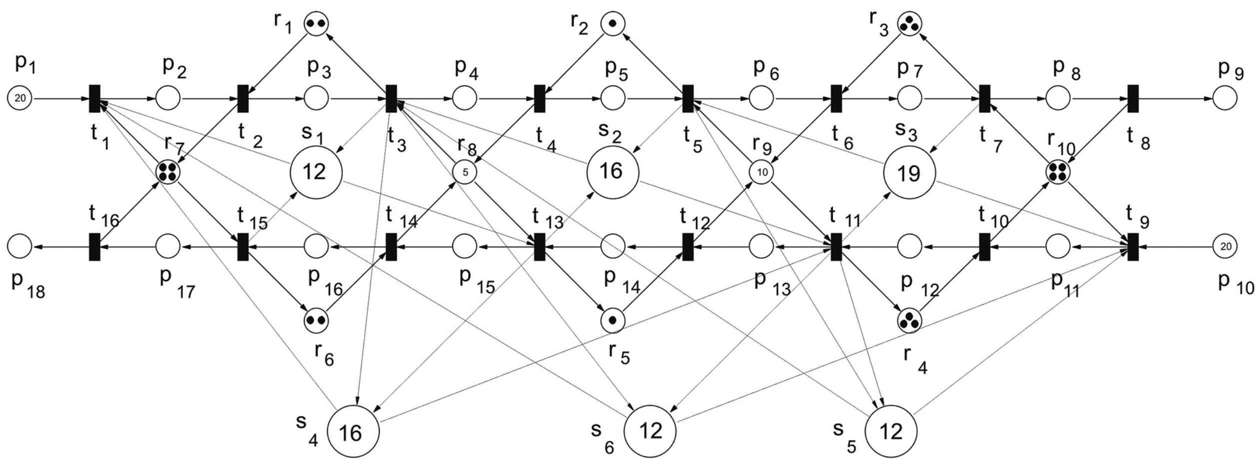

Figure 6.

Final Composite Petri net of CWS

Source: authors.

$J_{0_1} = \{p_2, p_3, p_{15}, p_{16}\}$,
$J_{0_2} = \{p_4, p_5, p_{13}, p_{14}\}$,
$J_{0_3} = \{p_6, p_7, p_{11}, p_{12}\}$,
$J_{0_4} = \{p_2, ..., p_5, p_{13}, ..., p_{16}\}$
$J_{0_5} = \{p_4, ..., p_7, p_{11}, ..., p_{14}\}$;
$J_{0_6} = \{p_2, ..., p_7, p_{11}, ..., p_{16}\}$.

Step 4: To check if the system is regular, matrix $\boldsymbol{C_{cw}}$ must be calculated by applying (15). The system is irregular $\boldsymbol{C_{cw}} \neq [0]$ From $\boldsymbol{R_{ccw}}$, by applying (16), it is possible to see that the key resources in the system are r_7 and r_8 (B_1 and B_2).

Step 5: The initial markings of CWs are $m_0(C_1) = 13$ $m_0(C_2) = 17$, $m_0(C_3) = 20$, $m_0(C_4) = 25$, $m_0(C_5) = 27$ and $m_0(C_6) = 35$. To avoid first level deadlock we can calculate 3 control places (S_1, S_2 and S_3) for controlling $J_{0_1}, J_{0_2}, J_{0_3}$. By applying constraints $m(J_0) \leq m_0(C) - 1$ it is possible to calculate $m_0(s_1) = 12$, $m_0(s_2) = 16$, $m_0(s_3) = 19$. We are adding control places S_1, S_2 and S_3 to the PPN (large circuits in Fig 5). There is no need for controlling J_{0_4}, J_{0_5}, J_{0_6}.

Step 6: To ensure the absence of second level deadlock, the supervisor has to take care of the availability of key resources r_7

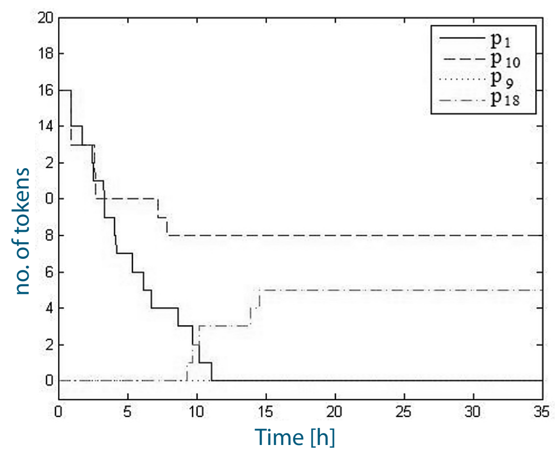

Figure 7.

CWS without supervisor.

Source: authors.

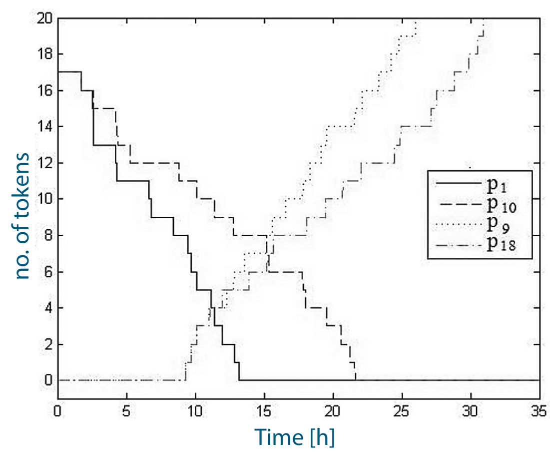

Figure 8.

CWS with supervisor.

Source: authors.

and r_8 in the way that each of the resources and both of them must not remain last available resources in the system. From the reachability tree one can find three deadlocks which occur in case of:

$m(p_2+p_3+p_{13}+p_{14})=m(r_1+r_5+r_7+r_9)$,

$m(p_4+p_5+p_{11}+p_{12})=m(r_2+r_4+r_8+r_{10})$

$m(p_2+p_3+p_{11}+p_{12})=m(r_1+r_4+r_7+r_{10})$

Step 7: To avoid second level deadlocks in step 6 one must calculate additional control places s_4, s_5, s_6 using the constraints:

Control place s_4:

$m(p_2+p_3+p_{13}+p_{14})\le m(r_1+r_5+r_7+r_9)-1$

Control place s_5:

$m(p_4+p_5+p_{11}+p_{12})\le m(r_2+r_4+r_8+r_{10})-1$

Control place s_6:

$m(p_2+p_3+p_{11}+p_{12})\le m(r_1+r_4+r_7+r_{10})-1$

The initial markings are $m_0(s_4)=16$, $m_0(s_5)=12$, $m_0(s_6)=12$ can be derived from (8). New control places s_4, s_5 and s_6 are added to the PPN.

Step 8: There are no new deadlocks in reachability tree of the CPN derived in step 7 and the algorithm ends. In total, six control places are added to the PPN to derive deadlock free CPN in Fig 6.

The deadlock prevention supervisor for CWS is verified using computer simulation and P-timed Petri net, in which time is associated to the particular places (see Table 1). Fig 7 and Fig 8 show number of vessels in input and output places $\{p_1, p_9, p_{10}, p_{11}\}$. First buffer first served control policy is applied, and maximum number of vessels in both directions passing through the CWS. Graph in Fig. 7 shows number of vessels in CWS without supervisor (deadlock occurs in critical system J_{0_3}, 15 h after beginning of simulation). Fig 8 shows a deadlock free CWS with supervisor. All vessels passed CWS in 31 h.

7. CONCLUSION

This paper shows a straightforward matrix based method for calculating conflict and deadlock prevention supervisor using MRF_1PN class of Petri net and P-invariant method for control places design. To achieve this, the first step is to make a suitable Petri net model of complex waterway system. Then, the structural properties of the Petri net, like P-invariants, circular waits, critical siphons and critical subsystems are investigated. The authors propose the addition of control places to the Petri net, which forms a supervisor. Conflicts and the first level deadlock can be avoided by adding control places, which disable firing of particular transitions and limit the number of vessels in critical subsystems. The second level deadlock can still exist if the system is irregular and if it contains so called key resource, and the existence of key resources must be checked. To avoid the second level deadlock, the supervisor must take care that one or more key resources would not be the last available resource in the net. The authors propose the novel method for second level deadlock prevention in case of more key resources. The calculated controller is verified using a P-timed Petri net model and computer simulation by using Matlab environment. The proposed matrix based method of supervisor design is not time consuming, and is suitable for design of complex traffic management system and can be easily implemented by men or by traffic lights. Future research will be focused on deadlock analysis and avoidance of systems with nondeterministic job routing.

APPENDIX: LIST OF MATRICES

Step 1: **P** matrix

$$\begin{array}{cc}
 & \begin{matrix} P_1 & P_2 & P_3 & P_4 & P_5 & P_6 & P_7 & P_8 & P_9 & P_{10} \end{matrix} \\
\begin{matrix} p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \\ p_8 \\ p_{10} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{15} \\ p_{16} \\ p_{17} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \\ r_9 \\ r_{10} \end{matrix} &
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\end{array}$$

Step 2: **C** matrix

$$\begin{array}{cc}
 & \begin{matrix} C_1 & C_2 & C_3 & C_{12} & C_{23} & C_{123} \end{matrix} \\
\begin{matrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \\ r_9 \\ r_{10} \end{matrix} &
\begin{bmatrix}
1 & 0 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1
\end{bmatrix}
\end{array}$$

Step 3: $\boldsymbol{S_{C_i}}$ and $\boldsymbol{J_0}(\boldsymbol{C})$ matrices

$\boldsymbol{S_{C_i}}$

$$\begin{array}{cc}
 & \begin{matrix} S_{C_1} & S_{C_2} & S_{C_3} & S_{C_4} & S_{C_5} & S_{C_6} \end{matrix} \\
\begin{matrix} p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \\ p_8 \\ p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{15} \\ p_{16} \\ p_{17} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \\ r_9 \\ r_{10} \end{matrix} &
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1
\end{bmatrix}
\end{array}$$

$\boldsymbol{J_0}(\boldsymbol{C})$

$$\begin{array}{cc}
 & \begin{matrix} J_{0_1} & J_{0_2} & J_{0_3} & J_{0_4} & J_{0_5} & J_{0_6} \end{matrix} \\
\begin{matrix} p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \\ p_8 \\ p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{15} \\ p_{16} \\ p_{17} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \\ r_9 \\ r_{10} \end{matrix} &
\begin{bmatrix}
1 & 0 & 0 & 1 & 0 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\end{array}$$

Step 4,5: $\boldsymbol{C_{cw}}$ and $\boldsymbol{R_{ccw}}$ matrices

$$\boldsymbol{C_{cw}} \neq [0]$$

$$\boldsymbol{R_{ccw}} = \begin{array}{cc}
 & \begin{matrix} r_1 & r_2 & r_3 & r_4 & r_5 & r_6 & r_7 & r_8 & r_9 & r_{10} \end{matrix} \\
\begin{matrix} C_1 \\ C_2 \\ C_3 \\ C_{12} \\ C_{23} \\ C_{123} \end{matrix} &
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0
\end{bmatrix}
\end{array}$$

REFERENCES

Barkaoui, K. and Abdallah, J., (1995), Deadlock avoidance in FMS based on structural theory of Petri Nets, IEEE Symposium on Emerging Technologies and Factory Automation, Paris, France, 10 – 13 October 1995, pp. 499-510., http://dx.doi.org/10.1109/ISIE.2005.1528915

Barkaoui, K., Chaoui, A. and Zuoari, B., (1997), Supervisory control of discrete event systems based on structure theory of Petri nets, In IEEE Proceeding of International Conference on System, Man, Cybernetic, Orlando, FL, pp. 3750-3755., http://dx.doi.org/10.1109/ICSMC.1997.633253

Barkaoui, K. and Petrucci, L., (1988), Structural analysis of workflow nets with shared resources, In Proceedings Workshop Workflow Management: Net – Based Concepts, Models, Tech, Tools, Lisbon, Portugal , 22 June 1988, pp. 82-85.

Boffey, T.B., (1982), Graph theory in Operations research, London: Macmillan.

Bogdan, S. and Lewis, F.L., (1997), Matrix approach to deadlock avoidance of dispatching in multi-class finite buffer reentrant flow lines, IEEE Proceedings of the International Symposium of Intelligent control, Istambul, Turkey, 16-18 July 1997, pp. 397 – 402.
http://dx.doi.org/10.1109/ISIC.1997.626525

Bogdan, S., Lewis, F.L., Kovačić, Z. and Mireles, J., (2006), Manufacturing systems control design – A matrix-based approach, London: Springer.

Charbonnier, F., Alla, H. and David, R., (2001), The Supervised Control of Discrete-Event Dynamic Systems, IEEE Transactions on Control Systems Technology, 7(2), pp. 175–187.

Ezpelta, J., Colom, J. and Martinez, J., (1995), A Petri net based deadlock prevention policy for flexible manufacturing systems, IEEE Transactions on Robotics and Automation, 11(2), pp. 173-184.
http://dx.doi.org/10.1109/ISIC.1997.626525

Gallego, J.L., Farges, J.L. and Henry, J.J., (1996), Design by Petri nets of an intersection signal controller, Transportation Research Part C: Emerging Technologies, 4(4), pp. 231-248.

Hernández, J., Ossowski, S. and Garcia-Serrano, A., (2002), Multiagent architectures for intelligent traffic management systems, Transportation Research Part C: Emerging Technologies, 10(5-6), pp. 473-506.

Hopcroft, J. E. and Ullman, J. D., (1979), Introduction to Automata Theory, Languages, and Computation, Reading: Addison-Wesley.

Iordache, M.V., Moody, J.O. and Antsaklis, P.J., (2000), Automated synthesis of deadlock prevention supervisor using Petri nets, Technical report of the ISIS Group at the University of Notre Dame, ISIS-2000-003.

Kezić, D., (2004), Sprječavanje potpunog zastoja u sustavima s diskretnim događajima primjenom Petrijevih mreža, PhD Thesis, Faculty of Electrical Engineering and Computing, University of Zagreb.

Kezić, D., Perić, N. and Petrović, I., (2006), An algorithm for deadlock prevention based on iterative siphon control of Petri net, Automatica, 47(1-2), pp. 19–30.

Kezić, D., Gudelj, A. and Vujović, I., (2009), Matrix Based Supervisor Design of Marine Traffic System. In Proceedings of the International conference ports and waterways, Zagreb, Croatia, 10-11 September 2009.

Lautenbach, K. and Ridder, H., (1996), The linear algebra of deadlock avoidance – a Petri net approach, Research Report of Institute for Computer Science, University of Koblenz.

Lee, S. and Tilbury, D. M., (2007), Deadlock-Free Resource Allocation Control for a Reconfigurable Manufacturing System With Serial and Parallel Configuration, IEEE Transactions on System, Man and Cybernetics – Part C, 37(6), pp. 1373-1381., http://dx.doi.org/10.1109/TSMCC.2007.905843

Lewis, F., Huang, H., Tacconi, D., Gurel, A. and Pastravanu, O., (1998), Analysis of deadlocks and circular waits using a matrix model for discrete event systems, Automatica, 34(9), pp. 1083-1100.

Mingming, Y., Hesuan, H. and Zwiwu, L., (2008), Deadlock Prevention Policy based on Elementary Siphons for Flexible Manufacturing Systems, Proceedings of the International conference on advanced intelligent mechatronics, Xi'an, China, 2-5 July 2008, pp. 229-234.
http://dx.doi.org/10.1109/AIM.2008.4601664

Murata, T., (1989), Petri Nets: „Properties, Analysis and Applications", In IEEE Proceedings, 77(4), pp. 541-580.

Overkamp, A.A.F. and van Schuppen, J.H., (1995), Control of discrete event systems – Research at the interface of control theory and computer science, CWI Quarterly, 8 (1995), pp. 31-45.

Piroddi, L., Cordone, R. and Fumagalli, I., (2009), Efficient Deadlock Prevention in Petri Nets through the Generation of Selected Siphons, American Control Conference, St. Louis, MO, USA, 10-12 June 2009, pp. 5006-5011.
http://dx.doi.org/10.1109/ACC.2009.5159861

Tricas, F., Garcia-Valles, F., Colom, J. M. and Ezpelta J., (2000), An iterative method for deadlock prevention in FMS, In Proceedings of the 5th Workshop Discrete Event System Ghent, Belgium, 21-23 August 2000, pp. 139 –148.

Tricas, F., Garcia-Valles, F., Colom, J. M. and Ezpelta J., (2005), A Petri net structure based deadlock prevention solution for sequential resource allocation systems, In Proceedings of the IEEE International Conference on Robotic and Automation, Barcelona, Spain, 18-22 April 2005, pp. 271-277.
http://dx.doi.org/10.1109/ROBOT.2005.1570131

Vaz, A.F. and Wonham, W.M., (1986), On supervisor reduction in discrete-event systems, International Journal of Control, 44(2), pp. 475-491.
http://dx.doi.org/10.1080/00207178608933613

Yamalidou, K., Moody, J., Lemmon, M. and Antsaklis, P., (1996), Feedback Control of Petri Nets Based on Place Invarijants,. Automatica, 32(1), pp. 15-28.

ZhiWu, L., Sen, Z. and MengChu, Z., (2009), A Divide-and-Conquer Strategy to Deadlock Prevention in flexible Manufacturing Systems, IEEE Transactions on System, Man and Cybernetics, 39(2), pp. 156-169.
http://dx.doi.org/10.1109/TSMCC.2008.2007246