Abstract—This paper presents deadlock prevention is used to solve the deadlock problem of flexible manufacturing systems (FMS). Petri nets contain been successful as one of the most powerful tools for modelling on FMS. Their modelling power and a mathematical arsenal supporting the analysis of the modelled systems stimulated the increasing interest in Petri nets. With the structural object of Petri nets, siphons are important to the analysis and control of deadlocks in Petri nets (PNs) excellent properties. The deadlock prevention method is caused by the unmarked siphons during the Petri nets are an effective way to model, analyze, simulate and control deadlocks in FMS is presented in this work. The characterization of special structural elements in Petri net so-called siphons has been a major approach to the investigation into deadlock-freeness in the canter of FMS. The siphons are structures which allow for some implications on the net's can be well controlled by adding a control place (called a monitor) for each uncontrolled siphon in the net in order to become the deadlock-free situation in the system. Finally, We proposed the method of modelling, simulation, controls of FMS by using Petri nets, where deadlock analysis has a Production line of parallel processing is demonstrated by a practical example used Petri Net-tool in MATLAB, is effective, and explicitly although its off-line computation.

Keywords: Petri net, Deadlock, Siphon and Trap, Simulation, Toolbox PN-Tool 2.3

1. Introduction

Petri nets are powerful graphical and mathematical modelling tools for the modelling, control, analysis, simulation and design of asynchronous sequential and non-sequential behaviors, concurrent systems, that allows describing and analyzing complex systems exhibiting concurrency [17]. Multiple resource sharing is a common situation in parallel and complex manufacturing processes and may lead to deadlock states. PNs allow to model and visualize systems, which contain concurrency, resource sharing or synchronization. Applications of Petri nets are included computer communication networks, automated manufacturing systems (especially flexible manufacturing system), computer systems, workflow, communication protocols, web service, and software engineering verification.

The main advantage of Petri Nets are the availability of comfortable mathematical methods in order to determine features structure of Petri Nets such as the so-called liveness or the absence of deadlocks (non-resolvable blockades). The associated mathematical methods are often included in the modelling tools and therefore allow a fast verification of features like the absence of deadlocks. Petri nets have been significant as one of the most powerful formal methods of modelling FMS. The reason is that they are well suited to represent such FMS characteristics as precedence relations, concurrence, conflict, and synchronization. Their analysis methods used for deadlock prevention in FMS include structural analysis and reachability graphs.

A deadlock is a state where a set of parts are in “a circular waiting”, i.e., each part in the deadlock set waits for a resource held by another part in the same set [9]. Deadlock problems can cause unnecessary cost (e.g. long downtime and low use of some critical and expensive resources), particularly important to be solved in Flexible Manufacturing Systems. To develop efficient algorithms to improve and optimize the system performances while preventing deadlock situations becomes a basic requirement in running in FMS [1, 6, 7, 10-16, 20-27]. Therefore, it is necessary to develop an effective FMS control policy to make sure that deadlocks will never occur to such systems. Deadlock is first defined in computer scientists when the time evolution of developing resource allocation logic in operating systems. In particular, Petri nets have also the deadlock property, which is using to control systems to be deadlock-free. There are three strategies to deal with deadlock [4, 9]: deadlock prevention, deadlock avoidance, deadlock detection, and recovery. In particular, prevention methods outlaw circular waits among concurring jobs of the design stage and off-line. Detection and recovery approaches use a monitoring mechanism for detecting the deadlock occurrence and a resolution procedure for appropriately preempting some deadlock resources. Moreover, deadlock avoidance schemes for manufacturing processes prevent
circular waits from occurring by the proper operational control of part flow.

There are many research papers devoted to deadlock problems. In our approach, we have adopted Petri nets as a tool for modelling the dynamic behavior of the system. This tool has also been adopted in several papers related to the study of deadlock problems with FMS environments [1-7] [9-16] [20-27]. For a general class of Petri net models, in (Viswanadham et al. (1990) [26]) both deadlock prevention and avoidance control policies are proposed. The first one is based on the net reachability graph, while the second one is based on a look-ahead procedure that searches for deadlock situations by simulating the system evolution for a pre-established number of steps. Because the avoidance policy does not assure that deadlocks are not reachable, they propose to combine this policy of a deadlock recovery system.

In Banaszak and Krogh (1990) [2], a Deadlock Avoidance Algorithm (DAA) is proposed to a class of FMSs called Production Petri Nets (PPN), a class of Petri nets models formed into a set of sequential processes (without alternatives in its execution) that use a resource in each state. The algorithm controls the input of new tokens in a model "zone's", assuring that system evolutions always possible. For the same class of FMS (PPN) a deadlock prevention policy was proposed to Xing et. al. (1997) [27]. Based on the concept of deadlock structure, a necessary and sufficient liveness condition for a PPN was proposed. Banaszak et. al (2002) [3] developed a new approach to the distributed deadlock avoidance control for Flexible Manufacturing Systems. They constructed feasible deadlock avoidance rules for exploiting the information about resource requirements of each particular operation of a manufacturing process and/or the repetitive character of the material and data process flow.

Petri Net modelling an FMS's as Simple Sequential Process with Resource (S³PR) nets has been introduced into Ezpeleta, et. al. (1995) [7]. S³PR is an extension of PPN in Banaszak et al. [2]. A circular wait situation occurs, to S³PR just as PPN. The Petri Net model is based on the arrangement of machines, rather than on the working process of the parts. In order to avoid the deadlock, a deadlock prevention method based on the concept of siphons has been introduced. The rule of allocating and releasing resources done, but S³PR allows choices in the process of manufacturing a type of a product. That is, a type of products possibly possesses several processing routes in an S³PR. It has been shown that, for an FMS to be deadlock-free, the Petri Net structure should be free of unmarked siphons. Tricas, et. al. (2006) [22] presented methods of the computation Minimal Siphons in Petri Net model on Resource Allocation Systems (RASs). Special syntactical constraints on some classes of RASs can help in developing specific algorithms to compute to siphon in a more efficient way. The aim at this analysis was to determine which methods could take advantage of the specific syntactical structure of S³PR nets and of the fact that the siphons related to deadlock problems must contain at least one resource place.

Abdallah and El-Maraghy (1998) [1] developed a deadlock prevention and avoidance methods based on the structure theory of Petri nets for a class called Systems of Sequential Systems with Shared Resources (S²R). They proposed S²R, which is a generalization of S³PR nets, to extend S³PR and (PPN) nets to model systems that can use not only alternative resources, as in S³PR nets, but they can also utilize more than one resource simultaneously. They adopted a deadlock prevention policy by adding a control place for each siphon to remain marked for all reachable marking. Further, they introduced a deadlock avoidance policy (DAP) to control the augmented S²R nets using a resource allocation policy based on the unsafe marking concept. With a look-ahead procedure, the controller determines one-step of the future evolution of making a resource allocation decision. However, it may be in the deadlock in some cases and a recovery procedure can initiate.

Chu and Xie (1997) [5] have developed a fast deadlock detection approach based on mixed integer programming (MIP) for structurally bounded nets whose deadlocks tied to unmarked siphons. Since no explicit enumeration of siphons is required, this formulation opens a new avenue for checking deadlock-freeness of large systems. The MIP method is able to find a maximal siphon unmarked at a reachable marking. A feasible solution corresponds to maximal unmarked siphons while there exists a siphon that can be emptied at a marking that is reachable from the initial marking. Based on this, they can formalize an algorithm that can efficiently obtain a minimal siphon from the result of the MIP method. Deadlock control is usually concerned with minimal siphons.

Huang et al. (2001) [11] proposed a minimal siphon extraction algorithm such that the complete siphon enumeration was successfully avoided. Also, Huang et al. (2006) [12] proposed some iterative deadlock prevention policies based on the MIP. In their methods, two kinds of control places called ordinary control places and weighted control places were added to the original model to prevent siphoning from being unmarked. Huang et al. (2012) [13] proposed using a marking/transition-separation instance (MTSI). They proposed policy has been implemented for FMSs based on the theory of regions and PNs, where the dead marking is identified with it is reachability graph. In the existing work, many inequalities (i.e., MTSIs), must be solved to prevent legal marking from entering the illegal zone in the original PN model.
Piroddi et al. (2008) [20] developed a selective siphon control policy that can obtain small size supervisors with highly permissive behavior. Shortly, they improve the policy of avoiding a complete, siphon enumeration [21]. Their methods can find a maximally permissive liveness enforcing the supervisor for each example presented in their studies [20, 21]. Unfortunately, no formal proof is provided to show that their policy can definitely produce a maximally permissive a supervisor in theory. On the other hand, they reduce the complexity of the supervisory structure but did not minimize it. However, many inequalities need to be solved.

Chen Y. F. et al. [6] (2013) developed a mathematical method to design a maximally permissive Petri net supervisor that is expressed by a set of control places with self-loops. A control place with a self-loop can represent by a constraint and a self-loop associated with a transition whose the firing may lead to an illegal marking. The constraint is designed to ensure that all legal markings are reachable and the self-loop is used to prevent the system from reaching illegal markings by disabling the transition to a specific marking. An MTSI in Huang et al. (2012) [13], it designs an optimal control place to separate bad markings from the reachability graph by disabling the transition to the marking/transition-separation instance (MTSI) at the corresponding marking. They proposed method are that the obtained supervisor can have self-loops.

The structural special objects called (siphons) are related to the liveness of a Petri net model, which has been widely used in the characterization and prevention/avoidance of deadlock situations. The controllability of siphons in a generalized Petri nets by Barkaoou et al. (1996) [4] proposed the concepts of max controlled-siphon property (cs-property, for short) and min cs-property. It is shown that if a marked (generalized) Petri net satisfies the max cs-property, then it is deadlock-free. Ezpeleta, et al. (1993) [8] proposed a new solution to the problem of finding generating families of siphons (structural deadlocks in classical terminology), siphon/trap-components (st-components, for short) used the addition of slack variables in the net. Li and Zhou, (2004) [15] proposed the concept of elementary siphons in S^PR. The siphon control has proposed to add a control place for each elementary siphon to make sure that it is always marked. This approach reduces the number of control places because it is not necessary to add control places in the redundant siphons.

Petri Net is using the theory of regions, Uzam (2002) [23] proposed an optimal liveness-enforcing supervisor synthesis method on the condition that such a supervisor exists. Ghaffari et al. (2003) [10] proposed conditions for the existence of an optimal supervisor that is maximally permissive by using plain and popular linear algebraic notions. For very big Petri net models, Uzam and Zhou (2006) [24] used the Petri net reduction approach to simplify the models so as to alleviate computation effort and simplify the invariant-based control method. Uzam et al. (2007) [25] proposed a redundancy test of a liveness-enforcing supervisor of an FMS in order to reduce the complexity of the controlled system. Li and Zhou et al. (2008) [16] addressed the deadlock problem by using a Petri net siphon control method and the theory of regions. The policy proposed to [16] significantly lowered the computational cost compared with the approach where the theory of regions was used alone. Using the theory of regions, one can obtain the optimal supervisor. However, there is a deadly disadvantage of these methods, that is, they need to compute a reachability graph that can cause the state explosion problem.

Petri Net is a graphical formalism, which is gaining popularity in recent years as a tool for MATLAB. PN-Toolbox was designed to offer specific instruments for simulation, analysis, and synthesis of discrete event systems. PN-Toolbox embedding in [18] the MATLAB environment presents the considerable advantage (with respect to other PN software) of creating powerful algebraic and graphical instruments, which exploit the high-quality routines available in MATLAB.

The remainder of this paper is organized as follows. Section 2, presents briefly the basic definitions and properties of PNs are giving, to illustrating the concepts through an example of the structure and behavioral properties such as siphon and trap of PN. The main structural and behavioral properties also are present. In section 3, an illustrative example is providing to illustrate the presented Petri Net Models of Flexible manufacturing systems. In section 4, used MATLAB PN-tool to analysis properties of PN, which aimed at the computer simulation models design is introducing. Concluding to remarks gives are in Section 6.

2. Preliminaries

This section contains the basic definitions of Petri nets theory, which will be needed in the remainder of the paper. We assume the reader is familiar with basic Petri nets concepts. For more details, please can be finding in the excellent survey article in Murata [17]. A Place/Transition Petri nets (P/T-net, for short), which represents the basic class of the sample family of Petri net models.

Definitions 1.

A Petri net is a 4-tuple $\Psi = (P, T, E, W)$, where $\text{P}$ is a finite non-empty set of $n = |P|$ places; $\text{T}$ is a finite non-empty set of $m = |T|$ transitions; $P \cap T = \emptyset$, i.e., places and transitions are disjoint sets. $E \subseteq (P \times T) \cup (T \times P)$ is the flow relation (a set of directed arcs), which relates places and transitions by arcs connecting them.
dom(E) \cup \text{cod}(E) = (P \cup T)$, where \(\text{dom}(E) = \{x \mid \exists y: (x, y) \in E\}\) is the domain of \(E\), \(\text{cod}(E) = \{y \mid \exists x: (x, y) \in E\}\) is the range of \(E\). \(W(x, y) = 0\), otherwise, where \((x, y) \in (P \times T)\) and \(\Psi\) is the set of nonnegative integers. (When \(W\), of the arcs \((W) = 1\), the net \(\Psi\) is called ordinary Petri net).

**Definition 2.** The pre-set (post-set) of a transition \(t\) is the set of all input (output) places of \(t\): \(\text{Pre}(p, t) = \{p \mid W(p, t) > 0\}\) and \(\text{Post}(p, t) = \{p \mid W(p, t) > 0\}\). The preset (post-set) of a place \(p\) is the set of all input (output) transitions to \(p\): \(\text{Pre}(p, t) = \{t \mid W(p, t) > 0\}\) and \(\text{Post}(p, t) = \{t \mid W(p, t) > 0\}\). Suppose \(x \in \mathbb{X}\) is arbitrary elements of the net \(\Psi\). \(x^* = \{y \mid (x, y) \in E\}\) is called pre-set of \(x\), and \(x^* = \{y \mid (x, y) \in E\}\) is called post-set of \(x\).

The usefulness of the symbols is a pre-set and a post-set of place \(p\) in \(P\) or a transition \(t\) in \(T\). \(\text{Pre}(p, t) = \{p \mid W(p, t) > 0\}\) is the set of input places of \(t\). \(\text{Post}(p, t) = \{p \mid W(p, t) > 0\}\) is the set of output places of \(t\). \(\text{Pre}(p, t) = \{p \mid W(p, t) > 0\}\) is the set of input transitions of \(p\), and \(\text{Post}(p, t) = \{p \mid W(p, t) > 0\}\) is the set of output transitions of \(p\).

**Definition 3.** Places \(P\) and transitions \(T\) are finite and nonempty disjoint sets and let \(E \subseteq (P \times T) \cup (T \times P)\). The elements \(x\) and \(y\) is that for each \(x \in (P \cup T)\) there exists a \(y \in (P \cup T)\) satisfying \((x, y) \in E\) or \((y, x) \in E\). A net \(\Psi\) is connected if for every two elements \(x\), \(y\) of \(\Psi\). For all \(x \in (P \cup T)\), the set \(x^* = \{y \mid y \in P \cup T \cup (x, y) \in E\}\) is called pre-set of \(x\), and \(x^* = \{y \mid y \in P \cup T \cup (y, x) \in E\}\) is called post-set of \(x\). An arc \((E)\) joins a place to a transition or a transition to a place, never a transition to a transition or a place to a place.

**Definition 4.** A marking Petri net is \(\Psi = (\Psi, M_0)\) where, \(\Psi\) is a Petri net and \(M_0\) is initial marking, \(M_0: P \to \{1, 2, \ldots, N^*\}\) is the net initial marking of \(\Psi\), assigned to each place \(p \in P\), \(M_0(p)\) tokens, and where \(N^*\) is a set of non-negative integers. Every \(P/T\)-net is provided with an initial marking \(M_0\) that may change a result of the firing of a transition (or a marking Petri net is \(5 -\) tuple: \(\Psi = (P, T, E, W, M_0)\)). Moreover, \(PN\) will be described either by the pair \(\Psi = (\Psi, M_0)\), where \(\Psi\) is a Petri net and \(M_0\) is initial marking, or by \(5 -\) tuple: \(\Psi = (P, T, E, W, M_0)\).

Graphically, the Petri net \(\Psi\) is represented by a diagram having two types of nodes. The elements of the set \(P\) are called places, graphically represented by circles, while the elements of the set \(T\) are called transitions, represented by rectangles. The function \(E\) defines the set of directed arcs, weighted by non-negative integers. The elements of the flow relation \(E\), pictorially are shown as arcs connecting different types of nodes. A Petri net is called to be ordinary if all of its arcs weights \(W\), is equal to one. When the arc weight equaled to zero stands for the absence of an arc. A place has taken a number of tokens, represented as black dots or a natural number. The execution of a Petri net causes its marking to change. Execution is performed by the firing enabled transitions. A transition is enabled when each of these one its input places are marked with at least as many tokens as the weight of the arcs connecting these input places with the transition. A transition fires by removing from many tokens from each one of its input places as the weight of the arcs connecting these input places with the transition and by placing into many tokens in each one of its output places as the weight of the arcs connecting these output places with the transition.

**Definition 5.** Ordinary Petri nets are those for which arcs \(E\): \((P \times T) \cup (T \times P) \to \{0, 1\}\). An ordinary Petri net is called a state machine (SM, for short) if \(\forall t \in T\), \(|t^*| = |t| = 1\). A Petri net \(\Psi = (P, T, E, W)\) is called: (1) a state machine (SM) if and only if (iff) \(|t| = |t^*| = 1\) for any \(t \in T\); (2) a mark graph (MG, for short) iff \(|p| = |p^*| \leq 1\) for any \(p \in P\); (3) free-choice net (FC, for short) iff \(|p^*| \leq 1\) or \(|p^*| \neq |p|\) for any \(p \in P\); (4) a conflict- free net (CF-net, for short) iff \(|p^*| = 1\) of \((\forall t \in p^*)\), where \(p \in p^*\) for any \(p \in P\); (5) a marked Petri net by the pair \(\Psi = (\Psi, M_0)\). \(M_0\) is a home state (SM, for short) \(M_0\) is a home state iff \(M_0 \subseteq M_{\text{init}}\) and a trap is a set of states \(\tau \in T\), such that \(\tau^* \subseteq \tau\).

**Example 1:** For the Petri net from Figure 1, is show an example of a Petri net.

![Figure 1](image-url)
Definition 6. (incidence matrix). The Pre- and Post- incidence matrix are net $\Psi$ can be represented as ($n \times m$) matrix, Pre-and Post with elements Pre($p_i$, $t_j$), and Post($p_i$, $t_j$), respectively. The linear algebraic analysis method uses the description of a Petri nets by its so-called incidence matrix $C = (C_{ij})$ which is defined by: $C(p_i, t_j) = \text{post}(p_i, t_j) - \text{pre}(p_i, t_j)$, is the change in number of tokens in $p_i$ after firing $t_j$ once, for $i = (1, 2, ..., n)$ and $j = (1, 2, ..., m)$.

Moreover, the incidence matrix $C$ of the net is defined as $C = \text{Post} - \text{Pre}$. The incidence matrix of a net is a matrix with $|P|$ rows and $|T|$ columns defined as $\text{pre}(P \times T) \rightarrow \{-1, 0, 1\}$ such that $\text{pre}(p, t) = (+1)\text{if}(t, p) \in E, (-1)\text{if}(p, t) \in E$, and 0 otherwise. The state changes of a PN occur when a transition $t$ is fire. These changes can be represented in an algebraic form using the state equation given by: $M_1 = M_0 + \text{post}(p, t) - \text{pre}(p, t)$. Any Petri Net can be the representation to incidence matrix. The incidence matrix of a net in Fig 1, is an (5 x 4) matrix. The Petri net of Fig. 1, has the following incidence matrices shown in Fig 2. The Matrix $C = C^+ - C^-$.

![Incidence Matrix](image)

**Fig.2.** The incidence matrix of Fig.1.

Definition 7. A transition $t \in T$ is enabled at M if and only if (iff) $(\forall p \in \ast t^*) (M(p) \geq 1)$ and $(\forall p \in \ast t^*) (M(p) = 0)$. A transition $t \in T$ is enabled at marking M iff $\forall p \in P$: $M(p) \geq W(p, t)$. This fact is denoted as $M[t]$. A transition $t$ enabled at a marking M can fire, it yields a new marking $M'$, which is referred to as the linear state equation of the net. The pair $(\Psi, M_0)$ is said to be reachable from $M_0$ by firing $\sigma$, and this is denoted by $M_0(\sigma)M'$.

The firing sequence is a marking $(M_1, M_2, M_3, ..., M_n = M')$ such that: $(\forall i, 1 \leq i \leq n)$, and $(M_i[t_j]M_{i-1})$, we can also write its by $[M_i]_{(\sigma^i)}M_{i+1}$. The set of all markings reachable from $M_0$ is denoted by Reachability set $R(M_0)$. The function $\sigma^i$ is the firing count vector of the firing sequence $\sigma$, i.e. $\sigma^i[t] \text{presents the algebraic sum of all the occurrence}$ of $t \in T$ in $\sigma$. If $M_0(\sigma)M'$, then we can write in vector forms $M' = M_0^+ + C.\sigma^i$ which is referred to as the linear state equation of the net. The pair $(\Psi, M_0)$ is called a

Definition 8. (firing transition ). A transition $t$ fired if this enabled. The firing of a transition $t$ enabled at marking $M$, removes $E(p, t)$ token from each input place $p$ to transition $t$, and adds tokens to each output place $p$ to transition $t$, where $E(t, p)$ is weight of arc $1$ to $p$. The marking $M'$ resulting from the firing of an enabled transition $t$ in a marking $M$, $M(t)M'$, is defined as follows:

**Example:**

Let us come back to the definition (7) (linear state equation) of an ordinary Petri net a transition $t$ of an ordinary Petri net is enabled if and only if each of its input places contains at least one token. In the example represented in Fig. 1, the initial marking is $M_0 = [1, 0, 1, 1, 0]^T$ and the following transitions can be fired starting from $M_0$.

Having applied definition (7), state equation $M' = M_0 + C.\sigma$, and taking the column vector of firing transition from the incidence matrix on Fig. 2, the marking evolve from: $M_1 = [0, 1, 0, 1, 0]^T$ after the firing of $t_1$. Such that $M_1 = M_0 + C.\sigma_1 = M_0 [1, 0, 1, 1, 0]^T + t_1 [-1, -1, -1, 1, 0]^T = [0, 1, 0, 1, 0]^T = M_1$. Each of these new marking represents a node of the reachability tree and used next-state at firing transition. In the sequel, the next marking can be found by adding last marking to the column of transition shown in Fig. 2. The marking evolve from: $M_1 = [0, 1, 0, 1, 0]^T$ after the firing of $t_1$. Such that $M_2 = M_1 + C.\sigma_1 = M_0 [1, 0, 1, 1, 0]^T + t_1 [-1, -1, -1, 1, 0]^T = [0, 1, 0, 1, 0]^T = M_2$. The rest, of our mathematical computation of the net of Fig. 1, the reachability tree of Petri net is shown in Fig. 3. The reachability tree can show deadlock in Petri net.

Definition 9. (Invariant). P-invariant (resp., T-invariant) of a net $\Psi = (\Psi, M_0)$ is a non-negative row integer $|P|$-vectors $x$ (resp., $|T|$-vector $y$) satisfying the equation $x^T.C = 0$ (respectively, $C. y^T = 0$), where $C$ is the incidence matrix of $\Psi$. A non-zero integer vector $y \neq 0$, (resp. $x \neq 0$).

Definition 10. (Properties Boundedness, and Safeness). A marked PN $(\Psi, M_0)$ is said to be (marking) k-bounded if each of its places is k-bounded. A 1-bounded net is called safe. A marked PN $(\Psi, M_0)$ is bounded if there
exists $k \in \mathbb{N}$ such that $(\Psi^*, M_0)$ is $k$-bounded. A net $\Psi$ is structurally bounded if \( \forall M_0 \) the marked PN $(\Psi, M_0)$ are $k$-bounded for some $k \in \mathbb{N}$.

**Definition 11.** (Siphon). A siphon of Petri net $\Psi$ is a subset $S \subseteq P$ of places that satisfies $^*S \subseteq S^*$. A siphon is empty under $M \in \Psi^0$ if $M(p) = 0$ for all $s \in P$. A siphon is initially empty if all transitions that produce tokens on it. For a P/T-net $\Psi$, a siphon is said to be minimal if and only if there does not exist another siphon $S'$ in $\Psi$ such that $S' \subseteq S$. Moreover, a siphon in a Petri Net is that a structural deadlock defined as a non-empty subset of places $S$. A P/T-net $\Psi$ is said to satisfy the siphon-trap property if and only if every siphon contains a marked trap (or every minimal siphon contains a marked trap). A siphon is said to be minimal if it contains no other siphons as its proper subset. A minimal siphon is strict if it contains no trap. A siphon is said to be controlled if it cannot be represented as a union of other siphons (traps). The union of some basic siphons (traps) can generate so all the siphons (traps) in a Petri net. A siphon (trap) is said to be minimal if it does not contain any other siphon (trap). Minimal siphons (traps) are basic siphons (traps), but not all basic siphons (traps) are minimal.

**Definition 12.** (trap). A trap of Petri net $\Psi$ is a subset $\tau \subseteq P$ of places that satisfies $^*\tau \subseteq \tau^*$. A trap is empty under $M \in \Psi^0$ if $M(p) = 0$ for all $s \in P$. A trap is initially empty blocks all transitions that produce tokens on it. For a P/T-net $\Psi$, a trap is said to be minimal if and only if there does not exist another trap $\tau'$ in $\Psi$ that $\tau' \subseteq \tau$. Moreover, a trap in a Petri Net is that a minimal trap is called a strict minimal siphon (SMS). A marked trap cannot never be emptied. A trap is said to be minimal if it does not contain another trap as a proper subset. $S$ is maximal if no other siphon includes it. It is called a strict minimal siphon (SMS), denoted $S$, if it does not contain a trap. A trap is said to be the controller if it is marked under all reachable markings. A trap is said to be minimal if it does not contain another trap as a proper subset. $\tau$ is maximal if no other trap includes it.

The union of two siphons (resp., traps) is again a siphon (resp., trap). A siphon (trap) is called a basic siphon (basic trap) if it cannot be represented as a union of other siphons (traps). The union of some basic siphons (traps) can generate so all the siphons (traps) in a Petri net. A siphon (trap) is said to be minimal if it does not contain any other siphon (trap). Minimal siphons (traps) are basic siphons (traps), but not all basic siphons (traps) are minimal.

**Lemma 1.** (Siphon property). Consider Petri net $\Psi = (\Psi, M_0)$ with siphon $S \subseteq P$ that is empty under the initial marking $M_0 \in \Psi^0$. Then $\Sigma \in S, M(p) = 0$, holds for all elements $M \in R(\Psi)$.

A siphon is called minimal if it does not contain another siphon as a proper subset. $S$ is maximal if no other siphon includes it. It is called a strict minimal siphon (SMS), denoted $S$, if it does not contain a trap. A siphon is said to be the controller if it is marked under all reachable markings. A trap is said to be minimal if it does not contain another trap as a proper subset. $\tau$ is maximal if no other trap includes it.

The union of two siphons (resp., traps) is again a siphon (resp., trap). A siphon (trap) is called a basic siphon (basic trap) if it cannot be represented as a union of other siphons (traps). The union of some basic siphons (traps) can generate so all the siphons (traps) in a Petri net. A siphon (trap) is said to be minimal if it does not contain any other siphon (trap). Minimal siphons (traps) are basic siphons (traps), but not all basic siphons (traps) are minimal.

Having applied definition (11), and (12). The Petri net of Fig. 1, has contains 7-sets of siphons and 5-sets of traps (excluding the entire set of places, which is both a siphon and a trap). In the net shown in Figure 1, $S_1 = \{p_2, p_3\}$, $S_2 = \{p_1, p_2, p_3\}$, $S_3 = \{p_2, p_3, p_4, p_5\}$, $S_4 = \{p_1, p_2, p_3\}$, $S_5 = \{p_2, p_3, p_4\}$, $S_6 = \{p_3, p_4\}$, and $S_7 = \{p_2, p_3, p_4\}$ are siphons, among which $S_1$ and $S_2$ are minimal siphon. There are five traps: $\tau_1 = \{p_2, p_3\}$, $\tau_2 = \{p_1, p_2, p_3\}$, $\tau_3 = \{p_1, p_2, p_4, p_5\}$, $\tau_4 = \{p_1, p_2, p_4\}$, and $\tau_5 = \{p_1, p_2, p_3, p_4, p_5\}$, among which $\tau_1$, and $\tau_5$ are minimal trap. For example, the set of $\{p_2, p_3\}$ is both a siphon and trap.

Analyse the structures of first siphon are $S_1 = \{p_2, p_3\}$. The pre-transitions are $^*S_1 = \{t_1, t_3\}$, and the post-transitions are $S_1^* = \{t_1, t_3\}$. That is a fact the set of siphon $S_1$ is a minimal siphon. Another siphon analyzes $S_2 = \{p_1, p_2, p_3\}$, and the pre-transitions are $^*S_2 = \{t_1, t_3\}$, and the post-transitions are $S_2^* = \{t_1, t_3\}$, therefore, $^*S_2 \subseteq S_1$ is true. In addition, $S_1$, and $S_2$ are minimal siphon. The union of two siphons is again a siphon, $S_1 \cup S_2 = S_4 = \{p_1, p_2, p_3\}$.

The computation of siphons can be complete the remaining sets of siphons in this manner. Similarly, the calculation of a minimal Trap is: $\tau_1 = \{p_2, p_3\}$, for example $\tau_1^* \subseteq \tau_1$, where $\{\tau_1 = \{p_2, p_3\}\} \subseteq \{\tau_1 = \{p_2, p_3\}\}$, because $\{t_1, t_3\} \subseteq \{t_1, t_3\}$. So that, the computation of traps can be complete the rest sets of traps in this manner.

**Definition 13.** (Reversibility). A Petri net $(\Psi, M_0)$, is reversible if, in any marking $M_k$ reachable from initial marking $M_0$, $M_k$ is reachable from $M_0$, that is, it is always possible to go back to the initial marking.
Among many properties of Petri nets being searched in literature. For the purpose of modeling on Flexible Manufacturing Systems, the most important properties are liveness, boundness, safeness, deadlock, conservative, persistence, reachability tree, reversibility, Siphon, and Trap. Let us recall the definition of the Petri nets liveness playing the essential role in further considerations.

**Definition 14.** (Liveness). A transition \( t \in T \) is live in net \( (\Psi, M_0) \) iff \( \forall M \in R(\Psi, M_0) \exists M' \in R(\Psi, M_0) \) such that \( M' \) enables \( t \). The marked net \( (\Psi, M_0) \) is live iff all its transitions are live (i.e. liveness of the net guarantees the possibility of an infinite activity of all transitions). A net \( \Psi \) is structurally live iff \( \exists M_0 \) such that the marked net \( (\Psi, M_0) \) is live. Transition \( t_1 \) has a shared place \( p \) with transition \( t_2 \) iff \( p \in \bullet t_1 \cap \bullet t_2 \). It should be noted that the property of Definition 14, implies the absence of deadlocks in the system modeled. Liveness in a Petri net means that for all reachable markings from an initial marking any transition can eventually be fired. This implies an absence of deadlock. In general, Petri nets may not be live.

**Definition 15.** (dead marking). A PN is live under \( M_0 \) iff \( \forall t \in T, t \) is live under \( M_0 \). A transition \( t \in T \) is live under \( M_0 \) iff \( \forall M \in R(\Psi, M_0), \exists M' \in R(\Psi, M_0), t \) is firable under \( M' \). A transition \( t \in T \) is dead under \( M_0 \) if \( \exists M' \in R(\Psi, M_0) \), where \( t \) is firable. A marking \( M \in R(\Psi, M_0) \) is a (total) deadlock iff \( \forall t \in T, t \) is dead.

**Definition 16.** (Conservative). A Petri net \( \Psi \) with the initial marking \( M_0 \) is conservative if \( \forall M' \in R(\Psi, M_0), \sum_{p_i \in P} M_0(p_i) = \sum_{p_i \in P} M(p_i) \).Conservation means that the total number of tokens are the same for all markings reachable from \( M \).

### 3. Petri net models of FMSs.

Petri nets modeling on the flexible manufacturing systems provided us the possibility to consider integrally the effects of different elements of producing a system, elements like machinery, robots, parts, etc., and determine the improved exploiting program in the planning phase can be successful.

**Example 2.** A production line is shown in Fig. 4. The system consists of two different machines (a lathe (M1) and drill machine (M2)), a robot (R) and buffer (B) with two storing intermediate parts between the two machines. The machines are automatically loaded and are unloading by the robot and a shared robot for loading and unloading between the two machines. A piece of raw material is first processed by Machine M1 and then by Machine M2 to produce a final product. In addition, types Part-A and Part-B are processing by M1 and M2 with two operations [operation1 (OP1), and operation2 (OP2)]. The manufacturing system is planning with two Production Line-1(PL-1) require two operations, op1 on the machine M1, and op2 on the machine M2, and vice versa in Production Line-2 (PL-2). As well as, the production line sequences are given as follows: Production Line-1(PL-1): Part-A: (M1) \( \rightarrow R\rightarrow (M2) \), and Production Line-2(PL-2): Part-B: (M2) \( \rightarrow R \rightarrow (M1) \).

![Figure 4. Petri net model of a resource-sharing manufacturing system.](image)

Legend: IS1 (OS1) - the i-th input (resp. output) store, R- Robot, M1- Machine Lather, M2 – Machine Dill, and (B)-buffer- workpiece buffer.

![Figure 5. An FMS with two production line [23].](image)

This example is taken from [23] and is used in [6, 12]. Here, the PN model of the system is shown in Fig. 5, we used different approaches when can be used experimental approach and testing Petri net tool in MATLAB 2008A.
The Petri net model corresponding to the given processes of flexible manufacturing systems cells is shown in Figure 5. The elements (or the set of places and transitions) of the PN are interpreted as follows: Transitions $t_1$, $t_3$, $t_4$, and $t_7$ correspond to the operations constituting Production Line 1 (PL-1), while transitions $t_2$, $t_5$, $t_6$, and $t_8$ correspond to the operations constituting Production Line 2 (PL-2), and Places $P_0$, and $P_1$ are interpreted as Machines M1, (resp. M2). Place $P_{10}$ corresponds to the robot. A robot R can pick up and deliver products between M1 and M2. Remaining places $P_7$, $P_8$, and $P_9$ correspond to performed op1 at M1 (i.e. $P_0$), after the finishing of op1, the robot unloaded the machine M1 and stores the intermediate part of the buffer at (p3), the robot loaded the machine M2 with an intermediate part of the buffer, and operation op2 at M2 respectively for (PL-2). The number of tokens in $t_4$, $t_6$, and $t_8$ correspond to the operations constituting to perform op1 at M2 ($P_2$), then a robot gets work-piece from the M2 and put into the buffer (p4), and start op2 at M2 (p6). For more detail of a sub-net model of fig. 5, are using to represent two long operations, i.e., M1 and M2 unloading, M2 loading, and R unloading. Two places are used to represent four short operations, i.e., M1 loading, R using to represent two long operations, i.e., M1 and M2

![Fig. 6, the activities P/T-net of sub-net of Fig. 5](image)

The structure of $S^3$ PR Nets

**Definition 17.** A simple sequential process ($S^2P$) is a Petri net $\Psi = (P_A \cup \{p^0\}, T, E)$, where the following statements are true:

1. $P_A \neq \emptyset$ is called a set of operation places;
2. $p^0 \notin P_A$ is called the process idle place;
3. A net $\Psi$ is a strongly connected state machine; and
4. every circuit of $\Psi$ contains place $p^0$.

**Definition 18 [14]:** Let $\Psi = (P, T, E)$ be an $S^2P$ with idle process place $p^0$. Let $C$ be a circuit of $\Psi$, and $x$ and $y$ be two nodes of a circuit (C). Node $x$ is said to be previous to $y$ if there exists a path in $C$ from $x$ to $y$, the length (given a circuit $C$, we denote $||C||$ the set of nodes in it, and the length $(C = ||C||)$ of which is greater than one and does not pass over the idle place $p^0$. This fact is denoted by $x < \Psi y$. Let $x$ and $y$ be two nodes in $\Psi$. Node $x$ is said to be previous to $y$ in $\Psi$ if there exists a circuit $C$ such that is $x < \Psi y$. This fact is denoted by $x < \Psi y$.

**Definition 19 ($S^3PR$) [taken from[14]].**

A system of simple sequential processes with resources ($S^3PR$): $\Psi = \bigcup_{i=1}^{k} \Psi_i = (P_A \cup p_0 \cup P_R, T, E)$, is defined as the union of a set of nets:

$\Psi_i = (P_A \cup \{p_0^i\} \cup P_R^i, T_i, E_i)$, sharing common places, where the following statements are true:

1. $p_0^i$ is called the process idle places of net $\Psi_i$.
2. Elements in $P_A$ and $P_R^i$ are called operation places and resource places respectively;
3. $p_0^i \neq P_A \neq \emptyset$; $p_0^i \neq P_R^i$; and $(p_A^i \cup \{p_0^i\} \cap P_R^i = \emptyset$;
4. $\forall p \in P_A^i, \forall t \in t^*, \forall t^i \in t^*$,
   $\exists r \in P_R^i, t^i \cap P_R^i = t^i \cap P_R^i = \{r^i\}$;
5. $\forall r \in P_R^i, t^i \cap P_R^i = t^i \cap P_R^i = \emptyset$;
6. $\Psi_i^t$ is a strongly connected state machine, where $\Psi_i^t = (P_A^i \cup \{p_0^i\}, T_i, E_i)$, is the resulting net after the places in $P_R^i$ and related arcs are removed from $\Psi_i$.
7. Every circuit of $\Psi_i^t$ contains place $p_0^i$.
8. Any two $\Psi_i^t$ are composable when they share a set of common places. Every shared place must be a resource one.
9. For $p \in P_A^i$, $(p^t) \cap P_R^i = \{r^t\}$, where resource place $r^t_p$ is called the resource used by p.
10. Transitions in $(p_0^i)^t$ and $t^t(p_0^i)$ are called source and sink transitions of an $S^3PR$ respectively. In an $S^3PR$,
$p^0$ is called the set of process idle places, $p_A$ is called the set of operation places and $p_R$ is called the set of resource places.

4. Simulation and Petri net Editor

In order to be planning an FMSs must possess the Petri Nets has been used successfully for modeling, simulation and analysis structure of the MATLAB Tool Petri net is shown in Figure 5, are needed to be solved by MATLAB.

The PN Toolbox has an easy to exploit Graphical User Interface (GUI) (Mahulea et al., 2003) \[18\] that gives the user the possibility to draw PNs in a natural fashion, to store, retrieve and resize such drawings. This GUI also permits the simulation, analysis, and design of PN models. The simulation mechanism is based on the rule for enabling and firing of transitions specific to the type of the current PN model. The PN Toolbox is divided into five types of classic PN models are accepted, namely: untimed, transition-timed, place-timed, stochastic and generalized stochastic. In our application, we used untimed PN models, which are the behavioral properties (e.g. boundedness, liveness, reversibility, etc.) may be studied based on the coverability tree of the net. The coverability tree is built with or without the $\omega$-convention (Murata, \[17\]). The structural properties are approached as the integer programming problems; the minimal-support P- and T-invariants are displayed, on request, in separate windows 7.

An FMS is shown in Fig. 5. This PN is denoted by $(\Psi, M_0)$. To show our deadlock prevention policy, the reachability tree $R(\Psi, M_0)$ of the PN system can be constructed as shown in Figure 7.

![Coverability Tree - Graphic Mode](image)

Fig. 7, Coverability tree of Fig. 5

The Menu bar (placed horizontally, on top of the window in MATLAB, displays a set of seven drop-down menus at the top of the window, where the user can select all the facilities available in the PN Toolbox. These menus are File, Options, Draw, Nets, Simulation, Analysis, and Help, …etc. The simulation experiment is to consider a uniformly distributed duration, the untimed PN model of this is FMS presented in Fig. 5. The initial marking $M_0 = [0,0,0,0,3,3,1,1,1]$ corresponds to the situation when all the resources are idle. The analysis of the behavioral properties of the PN model starts with consulting its coverability tree (Fig. 7). The dead marking $M_{13} = [1,1,0,0,1,2,0,0,0]$ and $M_{14} = [1,1,0,1,0,2,1,0,0,0]$ matches up the situation when robot $R$ is used to place part-A on $M1$ the same time when robot $R$ is used to place part-B on Machine (M2).

As one can see the asynchronous execution of processes can result in their mutual blocking. A possible deadlock situation corresponds to the marking shown in fig. 7. From the reachability tree PN, we can see two dead marking (i.e. $M_{13}$ and $M_{14}$) in the red color can be the identifier. Note that the deadlock occurred when marking $M_k$ fire $t_k(M_k, t_k) = M_{13}$, and $(M_k, t_k) = M_{14}$ are obtained. By considering the possible controlled-siphon property, that is added control place can be connected arcs to the necessary places in order to become deadlock-free. Three optimal control places $v_1$, $v_2$, and $v_3$ can be control of Petri net of Fig.5, shown in Fig. 8.

![Additional place v1, v2, and v3, called control place in order to controlled siphon.](image)

Fig. 8. Additional place $v_1$, $v_2$, and $v_3$, called control place in order to controlled siphon.

There are three local control places as seen in the augmented net in Fig. 8, that is produced method eliminate the states associated with primary deadlock. We
can prove by identifying the three states these control places eliminate. Control places V1, and V3 eliminates the state of the net when tokens are present in places (p1 and p4). In the other side, when part-A is occupying the lathe and part-B is occupying the drill, and a result of this is a circular wait condition or primary deadlock. The second control place V2 eliminates the state when tokens occupy places (p1 and p2) that state corresponds to part-A occupying the robot R and part-B occupying the drill. This situation also corresponds to a circular wait pattern or primary deadlock. We are connected control place V1 to the transition, where \( V1 = \{t_2, t_4\} \) and \( V1^* = \{t_1, t_4\} \). The second is the added control place \( V2 = \{t_6, t_8\} \) and \( V2^* = \{t_3, t_6\} \), while the third controls placed \( V3 = \{t_1, t_3, t_8\} \) and \( V3^* = \{t_1, t_8\} \).

After adding three control places \( V_i \) (where \( i = 1, 2, 3 \)) to show in Fig. 8, the initial marking of V is one. The simulation of Petri net is shown in Fig. 9, the additional places of \( (V_i) \) leads to an extra minimal siphon that is a marked trap. As a result, no siphon can be to empties in the net and it is deadlock-free (currently live). This example motivates one to discover the mechanism to make a siphon controlled by adding a monitor. The result final in the analysis of Petri nets is shown in Fig. 9, because the MATLAB tool can show the PN shape of the simulation is accurate and correctness results.

![Coverability Tree - Graphic Mode](image)

**Fig. 9** Coverability tree of Petri Net is deadlock-free

5. Conclusion

In this paper, we have a characteristic deadlock prevention problem of FMS is solving used the concept of siphons. This paper’s presents a deadlock prevention method of FMS, which are insufficiently marked in siphons cause deadlocks. This method is a base of the structural theory of Petri nets that can be led to unmarked siphons in their Petri net models. Petri nets provided a method of the analysis of concurrent systems. Deadlocks can be avoided by adding a control place, associated with the arcs to each emptiable siphon S to prevent it from being unmarked. However, generally too many control places and arcs are required. This is controlled on the net correspond to a deadlock-free.

This paper presents an efficient siphon for control structure analysis of PN in MATLAB Toolbox. This tool has much grateful help us to finding from a siphon and trap easy reachability or coverability tree, deadlock detection, liveness of an ordinary Petri net can also be investigated the control. The properties menu on MATLAB provides computational tools for the analysis of the behavioral and structural properties of the current PN model. Petri net model as a tool helps us to behavioral investigation and structural properties. These structures that PNs represent a well-known, powerful and widely used analytical formalism.

**REFERENCES**


Mowafak Hassan Abdul-Hussin received the B.S. (1977) from the University Al-Mustansiriyyah, Baghdad and M.S., and Ph. D. degrees from the Technical University of Wroclaw, Poland, 1984, and 1989 respectively. Since 1991, he has been with University of Technology, Baghdad. His research interests include Petri net theory and applications of the Flexible Manufacturing Systems.