This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999054, IEEE Access

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

Petri net Model Based on Neural Network for Deadlock Control and Fault Detection and Treatment in Automated Manufacturing Systems

Husam Kaid[1], Abdulrahman Al-Ahmari[1], Emad Abouel Nasr[1,2], Adel Al-Shayea[1], Ali K. Kamrani[3], Mohammed A. Noman[1], and Haitham A. Mahmoud[1,2]

[1]Industrial Engineering Department, College of Engineering, King Saud University, Riyadh 11421, Saudi Arabia

[2]Mechanical Engineering Department, Faculty of Engineering, Helwan University, Cairo 11732, Egypt

[3]Industrial Engineering Department, College of Engineering, University of Houston, Houston, TX 77204 4008, USA

Corresponding authors: Husam Kaid (yemenhussam@yahoo.com) and Emad Abouel Nasr (eabdelghany@ksu.edu.sa)

ABSTRACT Previously, different deadlock control strategies for automated manufacturing systems (AMSs) based on Petri Nets with reliable resources have been proposed. However, in real-world applications, resources may be unreliable. Therefore, deadlock control strategies presented in previous research studies are not suitable for such applications. To address this issue, this paper proposes a novel three-step deadlock control strategy for fault detection and treatment of unreliable resource systems. In the first step, a controlled system (deadlock-free) is obtained using the "Maximum Number of Forbidding First met Bad Markings Problem 1" (MFFBMP1), which does not consider resource failures. Subsequently, all obtained monitors are merged into a single monitor based on a colored Petri net. The second step addresses deadlocks caused by resource failures in the Petri net model using a common recovery subnet based on colored Petri nets. The recovery subnet is applied to the system obtained in the first step to ensure that the system is reliable. The third step proposes a hybrid approach that combines neural networks with colored Petri nets obtained from the second step, for the detection and treatment of faults. The proposed approach possesses the advantages of modular integration of Petri nets and can also learn neurons and reduce knowledge, similar to neural networks. Therefore, this approach solves the deadlock problem in AMSs and also detects and treats failures. The proposed approach was tested using an example from literature.

INDEX TERMS Automated manufacturing system, colored Petri net, deadlocks, fault detection, fault treatment, neural network

I. INTRODUCTION

An automated manufacturing system is a typical example of discrete event systems. It allows different product types to enter at discrete points in time with asynchronous or concurrent operations by sharing resources such as robots, automatic guided vehicles, machines, buffers, and automated tools. In an AMS, each component can be processed according to a given process sequence using a set of system resources. However, this sharing of resources can lead to deadlocks; hence, a few operations may remain incomplete. Therefore, deadlock control is essential for AMSs. In addition, resource faults may occur in a real-world system; this can cause new deadlocks in controlled AMSs. In general, a fault is defined as an interruption of an item's ability to perform a particular function [1], and it is synonymous with errors, mistakes, disturbances, or failures leading to unwanted or unbearable equipment behavior [2]. Resource faults cannot be neglected in a real production system. However, a majority of previous studies have only considered the process definition and optimization of the process under ideal conditions. The early detection and treatment of faults is of paramount importance and great practical significance to ensure safe, efficient, and reliable operation of AMSs, and also for avoiding performance degradation, product deterioration, and damage to human health. Therefore, it is essential to develop a robust deadlock prevention policy that can perform fault detection and treatment of unreliable resource systems and also ensure deadlock-freedom in AMSs.

Petri nets are a commonly used graphical and mathematical modeling tool suitable for scheduling, deadlock analysis, and control in AMSs [3, 4]. They are convenient for describing characteristics and behaviors such as synchronization, causal dependence, conflict, concurrency, and sequencing in AMSs. Petri nets can also be used to provide behavioral features, such as boundedness and liveness [5, 6]. From a technical perspective, several policies based on Petri nets have been proposed; these policies are based on three strategies: (i) deadlock detection and recovery, (ii) deadlock avoidance, and (iii) deadlock prevention [5, 7]. Most of these policies have proposed deadlock control in Petri nets through structural analysis [3, 8] and reachability graph analysis [9-11]. Moreover, three criteria were suggested for assessing and constructing an AMS control supervisor, namely behavioral permissiveness, computational complexity, and structural complexity [5, 12].

In the literature, there have been studies dealing with the development of deadlock control strategies. Some of these studies have assumed that resources in AMSs are reliable [8, 13-19], while some others have assumed that they are unreliable [20-30]. As an exploratory study on deadlock control for reliable and unreliable resources in AMSs, Wang et al. [22] suggested two strategies for one-unit resource allocation systems with unreliable resources. In the first policy, one unreliable resource was considered, and in the second several unreliable resources were considered. Chew et al. [23] guaranteed robust system operations by developing two-controller supervisors, where the types of parts could use a central buffer to deal with different unreliable resources. Liu et al. [24] developed recovery subnets to model the failure, and recovery nets of multiple types of unreliable resources for systems of simple sequential processes with resources (S^3PRs). Deadlocks were prohibited by designing control places based on a divide-and-conquer control policy, which was accomplished by inserting normal/inhibitor arcs between recovery subnets and control places. In some cases, however, a waiting time was required for a failed resource to be repaired, which resulted in a significant reduction in the resource utilization. Liu et al. [29] developed two robust deadlock controls for GS^3PRs (generalized S^3PRs) with unreliable resources, based on a reachability graph partition technique, whereby markings were divided into forbidden and legal markings. Thereafter, controlling the deadlocks in a robust way was accomplished by preventing the prohibited markings. Li et al. [31] developed a two-step deadlock control policy and a robust legal marking. In the first step, control places were designed, based on an elementary siphon policy developed in [32], which ensured that the system model was deadlock-free if there was no failure in the resource. The second step addressed failure-induced deadlock control problems. The resource failures were modeled by recovery subnets and recoveries were added into the system derived from the first step. This resulted in an unreliable controlled system. Yue et al. [25] proposed an AMS class deadlock controller policy with multiple unreliable resources using the algorithm modified by Banker, and a set of resource capacity constraints. Liu et al. [33] proposed a deadlock control strategy for GS^3PRs with unreliable resources, based on elementary siphons theory [34, 35] and the max′-controlled siphon control approach presented in [36]. Al-Ahmari et al. [37] proposed a robust deadlock state feedback control strategy, which consisted of a two-step system with unreliable and shared resources for S^3PRs. In the first step, a controlled system (deadlock-free) was obtained using a strict minimal siphon, which did not consider resource failures. The second step was designed to deal with the deadlock caused by all the resource failures. A common recovery subnet based on colored Petri nets was proposed. This recovery subnet was applied to the obtained system from the first step to ensure that the system was reliable.

Owing to the broad scope of the problem of detection and treatment of faults and the challenges in achieving real-time solutions to this problem, several approaches for the detection and treatment of AMS faults have been proposed over the past several years. Maki and Loparo [38] proposed a multi-layered feedforward neural network method to detect and diagnose defects in industrial processes; their method required simultaneous monitoring of multiple data. Their results showed that the method based on the neural network was successful in diagnosing and detecting defects associated with prior interim periods. In addition, this method could also be appropriately generalized for industrial processes. Liu et al [39] applied a multi-layer perceptron neural network approach to develop a continuous-time method to detect and diagnose a continuous motor, using continuous magnets. Their study confirmed the feasibility of the combined use of parameter estimation and classification of a neural network for fault detection and isolation in a motor, through experiments on a real motor. Based on the governing principles of the motor, the model parameters could be easily converted into electromechanical motion parameters, and defects could be detected. In addition, the neural network had the ability to classify patterns. Riascos and Miyagi [40] utilized the distributed Petri nets method to develop a supervisor framework to detect and address failures in the operations in manufacturing systems. The proposed method was efficient, and it effectively created several graphs that represented operational units as manufacturing systems. It could exploit networks that had already been analyzed for model systems without considering anomalous situations. It also had the ability to analyze models for each diagnosis of the failure with a more rational approach.

Riascos et al. [41] proposed a methodology for detecting and treating malfunctions in automated machines. The methodology was based on the integration of Petri diagnostic networks and Bayesian networks. Miyagi and Riascos [42] proposed a methodology for modeling and analyzing manufacturing systems that were fault-tolerant. This methodology helped improve normal production processes and also in the discovery and handling of errors. This approach was based on the hierarchical integration of networks and standard Petri nets. Rajakarunakaran et al. [43] utilized the artificial neural network approach to diagnose and detect defects in technical systems, in order to realize

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999054, IEEE Access

safer factory operations, especially in rotor systems. The proposed neural network model yielded results in almost 100% of the cases, as compared to other methods. Honggui et al. [44] employed a fuzzy neural network approach to design a fault detection system using online sensors and sludge volume index and employed fault diagnosis methods for the waste water treatment process. Their proposed method was subsequently applied to a real wastewater treatment process and compared with other methods. The results demonstrated that the approach was considerably promising for several types of chemical or biological sensors in the wastewater treatment process.

Neural networks have generated significant interest as problem-solving techniques in detection and treatment of faults. The advantage of neural networks is their ability to be generalized to deal with noisy or partial inputs. Neural networks can tackle continuous input data as well. However, the learning needs to be facilitated to solve the problem of detection and diagnosis of faults. Therefore, the major objective of this work is to propose a novel three-step deadlock control strategy for fault detection and treatment of unreliable resource systems. In the first step, a controlled system (deadlock-free) is obtained using "Maximum Number of Forbidding First- Bad Markings Problem 1" [45], which does not consider resource failure. The second step addresses issues of deadlock control caused by resource failures. For all the resource failures in the Petri net model, a common recovery subnet based on colored Petri nets is proposed [6]. This recovery subnet is applied to the obtained system from the first step to ensure that the system is reliable. The third step comprises a hybrid approach that combines a neural network with colored Petri nets, obtained from the second step to perform detection and treatment of faults. The proposed approach possesses the advantages of modular integration of Petri nets, and has the ability to learn neurons and reduce the knowledge, as in neural networks. Thus, this approach provides a combination of three features: (i) deadlock-free system without considering resource failure, (ii) detection of faults, and (iii) treatment of faults.

This paper is organized as follows. Section II describes basic concepts of Petri nets and the deadlock prevention policy based on the concept of strict minimal siphon. The robust control of unreliable resources based on colored Petri nets and computational complexity of the proposed policies are presented in Section III. The "General Petri Net Simulator" (GPenSIM) code and validation of the developed method is presented in Section V. A real-world AMS case study is presented in Section VI, followed by conclusions and future research presented in Section VII.

II. PRELIMINARIES

This section presents the basics of Petri nets, concept of strict minimal siphon, deadlock prevention policy based on AMSs, and GPenSIM tool.

A. Basics of Petri Nets

Definition 1: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W)$ be a simple sequential process with resources (S^3PR) Petri net, where p^0 is a process idle place; P_A is a set of operation places, and P_R is a set of resources places. $P_A \cap P_R = \emptyset$. T is a non-empty set of transitions. $P_C = P_A \cup \{p^0\} \cup P_R$, $F \subseteq (P_C \times T) \cup (T \times P_C)$ is called flow relations and expressed using arcs, which connect places to transitions or transitions to places. W: $(P_C \times T) \cup (T \times P_C) \rightarrow \boldsymbol{IN}$ is a mapping that assigns a weight to an arc, where $\boldsymbol{IN} = \{0, 1, 2, ...\}$.

Definition 2: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W)$ be an S^3PR Petri net; an N is called an ordinary net if $p \in P_C$, $t \in T$, $\forall(p, t) \in F$, and $W(p, t) = 1$.

Definition 3: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W)$ be an S^3PR Petri net; an N is named a weighted net, if $p \in P_C$, $t \in T$, $\forall(p, t) \in F$, and $W(p, t) > 1$.

Definition 4: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W)$ be an S^3PR Petri net, where x and y are nodes in N), i.e., $x, y \in P_C \cup T$. Then, ${}^\bullet x = \{y \in P_C \cup T \mid (y, x) \in F\}$ is called the preset (input) of node x, and $x^\bullet = \{y \in P_C \cup T \mid (x, y) \in F\}$ is called the postset (output) of node x. `

Definition 5: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W)$ be an S^3PR Petri net, and M_o be an initial marking of net N. An S^3PR is called acceptably marked if it satisfies (i) $M_o(r) \geq 1$ and $\forall r \in P_R$; (ii) $M_o(p) = 0$ and $\forall p \in P_A$; (3) $M_o(p^0) \geq 1$.

Definition 6: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^3PR; N has a self-loop free if for all $x, y \in P_C \cup T$, $W(x, y) > 0$ implies $W(y, x) = 0$.

Definition 7: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^3PR; N has a self-loop if for all $x, y \in P_C \cup T$, $W(x, y) > 0$ implies $W(y, x) > 0$.

Definition 8: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^3PR. A transition $t \in T$ that is enabled at marking M can be changed from M to a new marking M', represented by $M[t\rangle M'$ and can be defined as follows:

$$M'(p) = \begin{bmatrix} M(p) + W(p, t) & \text{if } p \in {}^\bullet t \setminus t^\bullet \\ M(p) - W(t, p) & \text{if } p \in t^\bullet \setminus {}^\bullet t \\ M(p) + W(t, p) - W(p, t) & \text{if } p \in t^\bullet \cap {}^\bullet t \\ M(p) & \text{otherwise} \end{bmatrix} \quad (1)$$

Where M is a mapping $M: P \rightarrow \boldsymbol{IN}$ and $M(p)$ is the number of tokens in place p.

Definition 9: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^3PR. $R(N, M)$ is a set of reachable markings of the Petri net model that are reachable from M in N. A transition $t \in T$ is live, if for all $M \in R(N, M)$, there exists a reachable marking $M' \in R(N, M)$ such that $M'[t\rangle$ holds. A net system (N, M_0) is dead at M_o if there does not exist $t \in T$ such that $M_o[t\rangle$ holds.

Definition 10: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^3PR. A marking M_o is a reversible, if for each marking $M' \in R(N, M_o)$, M_o is reachable from M'.

Definition 11: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^3PR. The marking M' is a coverable if there exists a marking $M'' \in R(N, M_o)$ such that $M''(p) \geq M'(p)$ for each p in the N.

Definition 12: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^3PR. $[N]$ is said to be the incidence matrix of net N, where

$[N]$ is an integer matrix that consists of $|T|$ columns and $|P|$ rows with $[N](p, t) = W(t, p) - W(p, t)$.

Definition 13: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^{3}PR. A marking M' is said to be reachable from M, if there exists a finite transition sequence $\delta = t_1\ t_2\ t_3\ \ldots\ t_n$ that can be fired, and markings $M_1, M_2, M_3, \ldots,$ *and* M_{n-1} are such that $M[t_1\rangle\ M_1[t_2\rangle M_2[t_3\rangle M_2\ \ldots\ M_{n-1}[t_n\rangle\ M'$, denoted as $M[\delta\rangle M'$, satisfies the state equation $M' = M + [N]\ \vec{\delta}$, where $\vec{\delta}$: $T \rightarrow \boldsymbol{IN}$ maps t in T to the number of appearances of t in δ, and is called a Parikh vector or a firing count vector.

Definition 14: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^{3}PR. N with initial marking M_o is called q-bounded, if it satisfies the relation: if for all $p \in P_C$, for all $M \in R(N,M_0)$, $M(p) \leq q\ (q \in \{1, 2, 3, \ldots\})$.

Definition 15: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^{3}PR. N is called safe if all of its places are safe, i.e., in each place p, the number of tokens does not exceed one. A net N is q -safe if it is q -bounded.

Definition 16: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^{3}PR. A place vector of N is defined as a column vector I: $P_C \rightarrow \boldsymbol{Z}$ indexed by P_C, and a transition vector of N is defined as a column vector J: $T \rightarrow \boldsymbol{Z}$ indexed by T, where $\boldsymbol{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$.

Definition 17: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^{3}PR. A place vector I of N is defined as a place invariant if I^T. $[N] = \mathbf{0}^T$ and $I \neq \mathbf{0}$, and a transition vector of N is defined as a transition invariant if $[N]$. $J = \mathbf{0}$ and $J \neq \mathbf{0}$.

Definition 18: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^{3}PR. A place invariant I of N is defined as a place semi-flow, if each element of I is non-negative. $||I|| = \{p\ |I(p) \neq 0\}$ is called the support of place invariant of I. $||I||^+ = \{p|I(p) > 0\}$ is called the positive support of place invariant I. $||I||^- = \{p\ |I(p) < 0\}$ is called the negative support of place invariant I. I is a minimal place invariant if $||I||$ is not a superset of the support of any other one and its components are mutually prime.

Definition 19: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^{3}PR. A transition invariant J of N is defined as a transition semi-flow, if each element of J is non-negative. $||J|| = \{t\ |J(t) \neq 0\}$ is called the support of transition invariant of J. $||J||^+ = \{t|J(t) > 0\}$ is called the positive support of transition invariant J. $||J||^- = \{t\ |J(t) < 0\}$ is called the negative support of transition invariant J. J is a minimal transition invariant, if $||J||$ is not a superset of the support of any other one, and its components are mutually prime.

Definition 20: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ be an S^{3}PR. l_i is said to be the coefficients of place invariant I, if for all $p_i \in P_C$, $l_i = I(p_i)$.

Definition 21: Let $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, SC, C_f, N_f, A_f, G_f, I_f\ M_o)$ be a colored S^{3}PR, where p^0, P_A, P_R, T, F, and W are defined as mentioned above. SC is a set of colors that comprises colors c_i and the operations on c_i. C_f is the color function that traces p into colors c_i, $p \in P_C$, and $c_i \in SC$. N_f is the node function that traces F into $(P_C \times T) \cup (T \times P_C)$. A_f is the arc function that traces each flow (arc) $f \in F$ into the term e. G_f is the guard function that traces each transition $t \in T$ to a guard expression g that has a Boolean value. I_f is the initialization function that traces each place $p \in P_C$ into an initialization expression.

B. DEADLOCK PREVENTION POLICY BASED ON AN ITERATIVE METHOD AND COLORED PETRI NETS

In this section, an MFFBMP1 that was developed in [45] is presented to construct a place invariant (PI), which can forbid as many first met bad markings (FBMs) as feasible. Subsequently, the approach of vector covering method is used to determine the minimal covering sets of legal markings and FBMs [5]. The solution of the integer linear programming problem (ILPP) can then derive invariant place coefficients and monitors. The design process of the iterative monitor is performed as follows. At each iteration, the PI is intended to forbid as many FBMs as feasible. All the FBMs forbidden by the PI are removed from the minimal covered set of FBMs. This process is stopped when all the minimal covered set of FBMs are prohibited.

Definition 22* [46]*: Let (N, M_o) be an S^{3}PR with $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$. M and M' are two markings in $R(N,M_0)$. M C-covers M', if for all $p \in P_A$, $M(p) \geq M'(p)$, which is expressed as $M \geq_C M'$.

Definition 23* [46]*: Let (N, M_o) be an S^{3}PR with $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ and M_{FBM} be the set of FBMs in N. $\forall M \in M_{FBM}$, a subset of M_{FBM}, which C-covers M is expressed as $F_M = \{M' \in M_{FBM} | M' \geq_C M\}$.

Definition 24* [46]*: Let (N, M_o) be an S^{3}PR with $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ and M_L be the set of legal markings in N. $\forall M \in M_L$, a subset of M_L, which C-covers M is expressed as $R_M = \{M' \in M_L | M' \geq_C M\}$.

Definition 25* [46]*: Let (N, M_o) be an S^{3}PR with $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ and M^*_{FBM} be a subset of FBMs in N and called the minimal covered set of FBMs, satisfying:

1. $\forall M \in M_{FBM}, \exists M' \in M^*_{FBM}$, subject to $M \geq_C M'$; and
2. $\forall M \in M_{FBM}, \nexists M'' \in M^*_{FBM}$, subject to $M \geq_C M''$ and $M \neq M''$.

Definition 26* [46]*: Let (N, M_o) be an S^{3}PR with $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ and M^*_L be a subset of legal markings in N and called the minimal covered set of legal markings, satisfying:

1. $\forall M \in M_L, \exists M' \in M^*_L$, subject to $M' \geq_C M$; and
2. $\forall M \in M_L, \nexists M'' \in M^*_L$, subject to $M'' \geq_C M$ and $M \neq M''$.

The method of compute the set of legal markings M_L, the set of FBMs M_{FBM}, minimal set of FBMs M^*_{FBM}, and minimal set of legal markings M^*_L is introduce in Chen et al. [46].

The following mathematical model shows the model developed by [45].

MFFBMP1:

$$max\ f = \sum_{k \in N^*_{FBM}} f_k$$

Subject to

$$\sum_{i\in N_A} l_i . M_l(p_i) \leq \beta, \forall M_l \in M_L^* \tag{2}$$

$$\sum_{i\in N_A} l_i . M_k(p_i) \geq \beta + 1 - Q.(1 - f_k), \forall M_k \in M_{FBM}^* \tag{3}$$

$l_i \in \{0,1,2,\dots.\}, \forall i \in \mathbb{N}$
$\beta \in \{1,2,\dots.\}$
$f_k \in \{0,1\}, \forall k \in N_{FBM}^*$.

The objective function f is used to maximize the number of FBMs that are prohibited by place invariant PI. Denote its optimal value by $f*$. M^*_{FBM} and M^*_L are defined in Definitions 25 and 26, respectively. N^*_{FBM} is used to represent $\{i|M_i \in M^*_{\text{FBM}}\}$. If $f* = 0$, we have $f_k = 0$, $\forall k \in N^*_{\text{FBM}}$; that is to say no FBMs in M^*_{FBM} can be prohibited by the place invariant. In (2), let I be a place invariant, where l_i's ($\{i|p_i \in P_A\}$) are the coefficients of I, β is a positive integer variable, and $M_l(p_i)$ is the number of tokens in minimal covering set of legal markings, $\forall M_l \in M^*_L$.

All legal markings should be kept after the addition of a monitor, meaning that any marking $M_l \in M^*_L$ cannot be prevented from being reached; *li's* coefficients should satisfy (2), which is called the condition of reachability. In (3), Q is a constant positive integer that must be sufficiently large; $M_k(p_i)$ denotes the number of tokens in minimal covering set of FBMs; $\forall M_k \in M^*_{\text{FBM}}$ and fk a set of M^*_{FBM} variables, where $f_k \in \{0, 1\}$, $k = 1,2,\dots$). In addition, in (3), $f_k = 1$ indicates that M_k is forbidden by I, $f_k = 0$ indicates that this constraint is redundant, and M_k cannot be forbidden by I.

Definition 27: Let (N_x, M_x) and (N_y, M_y) be two S³PR Petri nets with $N_x = (P_{Cx}, T_x, F_x, W_x)$ and $N_y = (P_{Cy}, T_y, F_y, W_y)$, respectively. (N_z, M_z) is called a synchronous Petri net resulting from the integration of (N_x, M_x) and (N_y, M_y), expressed as $(N_x, M_x) \| (N_y, M_y)$, and satisfying: (i) $P_z = P_x \cup P_y$ and $P_x \cap P_y = \emptyset$. (ii) $T_z = T_x \cup T_y$, (iii) $F_z = F_x \cup F_y$, (iv) $W(b) = W_i(b)$, where $b \in F_i$, $i = x, y$, and (5) $M(p) = M_i(p)$, $p \in P_i$, $i = x, y$.

Definition 28: Let (N, M_o) be an S³PR with $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$. (DC, M_{DCo}) denotes a proposed deadlock controller for N by [45], expressed as $(DC, M_{DCo}) = (P_{DC}, T_{DC}, F_{DC}, M_{DCo})$, where (i) $P_{DC} = \{V_S \backslash V_S \in V_p\}$ is set of control places, and $V_p = \{V_{S1}, V_{S2}, \dots, V_{Si}\}$ is the set of control places to be detected. (ii) $T_{DC} = \{t \backslash t \in {}^{\bullet}V_S \cup V_S^{\bullet}\}$. (iii) $F_{DC} \subseteq (P_{DC} \times T_{DC}) \cup (T_{DC} \times P_{DC})$ is called a flow relation of (DC, M_{DCo}). (iv) $M_{DCo}(V_S)$ is an initial marking of a control place V_S, expressed as $M_{DCo}(V_S) = \beta$, $V_S \in P_{DC}$. (N_{DC}, M_{DCo}) is called a controlled Petri net model resulting from the integration of (N, M_o) and (DC, M_{DCo}), expressed as $(N, M_o) \| (DC, M_{DCo})$.

***Definition 29* [3]:** Let (N, M_o) be an S³PR with $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$. (DC, M_{DCo}) denotes a proposed deadlock controller for N by [45], expressed as $(DC, M_{DCo}) = (P_{DC}, T_{DC}, F_{DC}, M_{DCo})$. (DC, M_{DCo}) can be simplified and replaced by a colored common deadlock control subnet, which is a Petri net $N_{CC} = (\{p_{combined1}\}, \{T_{CCi} \cup T_{CCo}\}, F_{CC}, C_{vsi}, M_{CCo})$, where $p_{combined1}$ is called the combined monitor of P_{DC}. $T_{CCo} = \{t \backslash t \in V_S^{\bullet}\}$. $T_{CCi} = \{t \backslash t \in {}^{\bullet}V_S\}$. $F_{CC} \subseteq (\{p_{combined1}\} \times \{T_{CCi} \cup T_{CCo}\}) \cup (\{T_{CCi} \cup T_{CCo}\} \times \{p_{combined1}\})$ is called a flow relation of N_{CC}. C_{cri} is the color that maps $p_{combined1}$ into colors $C_{vsi} \in C_R$, where $C_R = \cup_{i \in V_S} \{C_{vsi}\}$. (N_{CC}, M_{CCo}) is called a colored common deadlock control subnet. $M_{CCo}(p_{combined1})$ is an initial token with the color markings of the combined monitor, expressed as $M_{CCo}(p_{combined1}) = \sum M_{DCo}(V_S)$, $V_S \in P_{DC}$.

***Definition 30* [3]:** Let (N, M_o) be an S³PR with $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ and (N_{CC}, M_{CCo}) be a colored common deadlock control subnet. We call (N_{CN}, M_{CNo}) a controlled colored S³PR Petri net. Furthermore, $(N_{CN}, M_{CNo}) = (N, M_o) \| (N_{CC}, M_{CCo})$, which is the integration of (N, M_o) and (N_{CC}, M_{CCo}), where $N_{CN} = (\{p^0\} \cup \{p_{combined1}\} \cup P_A \cup P_R, T \cup T_{CCi} \cup T_{CCo}, F \cup F_{CC}, C_R, M_{Co})$, and $R(N_{CN}, M_{CNo})$ be its reachable graph.

Theorem 1. The controlled colored S³PR Petri net (N_{CN}, M_{CNo}) is live.

Proof. We must prove that all transitions T, T_{CCi}, T_{CCo} in (N_{CN}, M_{CNo}) are live. There is no M^*_{FBM}, since all $t_x \in T$ are live. For all $t_y \in T_{CCi}$, if $\forall p_i \in {}^{\bullet}t_y$, $M_{CN}(p_i) > 0$, then t_y can fire in any case because it is uncontrollable. Thus, $M_{CN}(p_{combined1}) > 0$; for all $t_z \in T_{CCo}$, if $M_{CN}(p_{combined1}) > 0$, then t_z can fire. Therefore, the controlled colored S³PR Petri net (N_{CN}, M_{CNo}) is live.□

Based on the concept of MFFP1[45] and colored Petri net [3], the deadlock-prevention algorithm is developed as follows:

Algorithm 1: *(Deadlock-prevention algorithm based on an iterative method and colored Petri nets)*

Input: *An S³PR (N, M_o).*
Output: *A controlled colored S³PR Petri net (N_{CN}, M_{CNo})*
Initialization: $V_p = \emptyset$.
Step 1: *Compute M_{FBM}, M_L, M^*_{FBM}, M^*_L.*
Step 2: while** $M^*_{\text{FBM}} \neq \emptyset$ **do

1. *Build MFFBMP1.*
2. *Solve MFFBMP1. Let li's and β be the solution*
   if** $f* \neq 0$ **then
   Let li's and β be the solution
   else
   Exit, as the solution does not exist.
   end if
3. *Design PI and a monitor Vs.*
4. *$V_p := V_p \cup \{Vs\}$ and $M^*_{\text{FBM}} = M^*_{\text{FBM}}$ - FI. /*FI is covered M^*_{FBM}*

end while

Step 3: *Merge all monitors V_p into common monitor ($p_{combined1}$) and consider the following steps:*

1. *Insert output arcs of $p_{combined1}$ T_{CCo}. /* By Definition 29.*/*

2. *Insert input arcs of $p_{combined1}$ T_{CCi}. /* By Definition 11.*/*
3. *Assign colors C_{vsi} for all monitors P_{DC} /* By Definition 29.*/*
4. *Compute an initial token with the colors marking of a merged monitor $M_{CCo}(p_{combined1}) = \sum M_{DCo}(V_S)$/* By Definition 29.*/*

Step 4: *Add the merged monitor into the net (N, M_o).*
Step 5*: Output a controlled colored S^3PR Petri net (N_{CN}, M_{CNo})*
Step 6: End

C. ROBUST CONTROL FOR UNRELIABLE RESOURCES BASED ON COLORED PETRI NETS

In Algorithm 1, the resources of the system are usually assumed to be reliable. After applying Algorithm 1 to such a system, one common monitor is inserted. In practice, the resources of the system may fail. Therefore, a common recovery subnet is designed and inserted to model all the resource failures in the system. This section concentrates on the relation between the controlled system in Algorithm 1 and resource failures.

Definition 31 [6]: Let (N_{CN}, M_{CNo}) be a controlled colored S^3PR Petri net and $r_u \in P_R$ be an unreliable resource in N_{CN}. A colored common recovery subnet of r_u is a Petri net $N_{ccr} = (\{p_i, p_{combined2}\}, \{t_{fi}, t_{ri}\}, F_{ccr}, C_F)$, where $p_{combined2}$ is called the recovery place of all p_i, t_{fi} is a failure transition, and t_{ri} is the recovery transition. $F_{ccr} = \{(p_i, t_{fi}), (t_{fi}, p_{combined2}), (p_{combined2}, t_{ri}), (t_{ri}, p_i)\}$, and an unreliable resource may fail when it is busy (its holders) or in an idle state r_u. Thus, we describe $P_H = \{r_u\} \cup H(r_u)$ as a set of places, where $H(r_u)$ is a set of holders of r_u, expressed by $H(r_u) = \{p \backslash p \in P_A, p \in {}^{\bullet\bullet}r_u \cap P_A \neq \emptyset\}$; $p_i \in P_H$. C_F is a set of colors that maps $p_i \in P_H$ into colors $C_{ccri} \in C_F$, where $C_F = \cup_{i\in pi} \{C_{ccri}\}$. (N_{ccr}, M_{ccro}) is called a colored common recovery subnet, where $M_{ccrio}(p_{combined2}) = 0$ and $M_{ccro}(p_i) \geq 0$.

Definition 32 [6]: Let (N_{CN}, M_{CNo}) be a controlled colored S^3PR Petri net with $N_{CN} = (\{p^0\} \cup \{p_{combined1}\} \cup P_A \cup P_R, T \cup T_{CCi} \cup T_{CCo}, F \cup F_{CC}, C_R, M_{Co})$, and let P_{UR} be an unreliable resources set. For all $r_u \in P_{UR}$, insert one common recovery subnet for all $p_i \in P_H$ resulting in a colored controlled unreliable S^3PR Petri net expressed as $(N_{CU}, M_{CUo}) = (N_{CN}, M_{CNo}) \parallel (N_{ccr}, M_{ccro})$, which is the integration of (N_{CN}, M_{CNo}) and (N_{ccr}, M_{ccro}).

Definition 33 [6]: Let $N_{CU} = ((\{p^0\} \cup \{p_{combined1}, p_{combined2}\} \cup P_A \cup P_R, T \cup T_{CCi} \cup T_{CCo} \cup T_F \cup T_R, F \cup F_{CC} \cup F_{ccr}, \{C_R, C_F\}, M_{CUo})$ be a colored controlled unreliable S^3PR Petri net and T_F and T_R be sets of failure transitions and recovery transitions, respectively. Here, $T_F = \cup_{i\in \mathrm{NA}}\{t_{fi}\}$, $T_R = \cup_{i\in \mathrm{NA}}\{t_{ri}\}$, $\mathrm{NA} = \{i|p_i \in P_H\}$ and M_{CUo} is an initial marking of N_{CU}.

Theorem 2: The net (N_{CU}, M_{CUo}) is live.

Proof: We must prove that all transitions T, T_{CCi}, T_{CCo}, T_F, and T_R in (N_{CU}, M_{CUo}) are live. If there is no failure in $r_u \in P_{UR}$ and for all $t_x \in T$, if $\forall p_i \in {}^{\bullet}t_x$, $M_{CU}(p_i) > 0$, then t_x can fire. For all $t_y \in T_{CCi}$, if $\forall p_i \in {}^{\bullet}t_y$, $M_{CU}(p_i) > 0$, then t_y can fire. For all $t_z \in T_{CCo}$, if $M_{CU}(p_{combined1}) > 0$, then t_z can fire. For all $t_e \in T_F$, if for all $p_i \in {}^{\bullet}t_e$, $M_{CU}(p_{combined2}) > 0$, then t_e can fire, leading to $M_{CU}(p_{combined2}) > 0$. For all $t_s \in T_R$, if $M_{CU}(p_{combined2}) > 0$, then t_s can fire. Therefore, we can say that the net (N_{CN}, M_{CNo}) is live.□

Based on above Definitions 31–33 and Theorem 2, the developed unreliable resources based on colored Petri net algorithm is stated as follows:

Algorithm 2: *(Unreliable resources based on colored Petri net algorithm)*

Input: *A net (N_{CN}, M_{CNo}) by Algorithm 1*
Output: *A net (N_{CU}, M_{CUo})*
Initialization: *Design a $p_{combined2}$.*
Step 1: for *each $r_u \in P_{UR}$* ***do***

1. *Add a failure transition t_{fi}. /* By using Definition 31.*/*
2. *Assign color C_{ccri} for failure transition t_{fi}. /* By using Definition 31.*/*
3. *Add a recovery transition t_{ri}. /* By using Definition 31.*/*
4. *Add an arc from p_i to t_{fi}. /*By using Definition 31.*/*
5. *Add an arc from t_{fi} to $p_{combined2}$. /* By using Definition 31.*/*
6. *Add an arc from $p_{combined2}$ to t_{ri}. /* By using Definition 31.*/*
7. *Add an arc from t_{ri} to p_i. /By using Definition 31.*/*

end for
Step 2*: Output a net (N_{CU}, M_{CUo}).*
Step 3: End

D. NEURAL NETWORK PETRI NET MODEL STRUCTURE FOR FAULT DETECTION AND TREATMENT

Neural networks have attracted considerable interest as problem-solving techniques in the detection and treatment of faults. The main element here is a neuron with multiple inputs and a single output. Each input is multiplied by a weight, the inputs are summed, and this quantity is operated by the neuron's transfer function to create an output [47]. Occasionally, the output is referred to as the level of activity. A multi-layer feedforward neural network is utilized in this study, which has one hidden layer. To adjust the weighted sum input of each neuron, the bias unit, whose activity level is corrected at one, is linked to all the neurons in the hidden and output layers. Each application determines the number of neurons in the input and output layers, and the number of neurons in the hidden layer will be adapted throughout the learning phase to enable successful learning of the network. The learning algorithm for neural network simulates the system of neuronal biology, which relies on a dynamic structure that depends on simulation, inhibition and cooperation, and competition for information processing, to guide the network's learning and working. However, unlike many other neural networks it does not take network errors or energy function as the algorithm's rules. A neural network can build several types of nets that can organize themselves. Neural networks have many forms and algorithms. However, only the basic structure is mostly used.

Definition 34: Let (N_N, M_{No}) be a neural Petri net model with $N_N = (P_N, T_N, F_N, X_k, Y_k, W_N, M_{No})$, where

1. $P_N = \cup_{i\in n}\{p_{ni}\}$ is a set of places;
2. $T_N = \cup_{i\in m}\{t_{di}\}$ is a set of transitions;
3. $F_N \subseteq (P_N \times T_N) \cup (T_N \times P_N)$ is the flow relation, and the elements of F_N are called arcs;
4. $X_k = \cup_{i\in n}\{x_i^k\}$ is a set of input neurons of input pattern k, with each x_i mapped to the corresponding p_i, and representing the input variable of neural Petri net model, for $i = 1,\ldots, n$ and $k = 1, 2, \ldots, \varphi$;
5. $Y_k = \cup_{i\in m}\{y_i^k\}$ is a set of outputs of pattern k, with each y_i mapped to the corresponding p_i for $I = 1,\ldots, m$ and $k = 1,\ldots, \varphi$;
6. $W_N \to [0,1]$ is a set of a place to transition the connectivity matrix;
7. M_{No} is an initial marking of N_N.

Definition 35: Let (N_N, M_{No}) be a neural Petri net model with $N_N = (P_N, T_N, F_N, X_k, Y_k, W_N, M_{No})$; a place $p_i \in P_N$ is said to be an axiom if ${}^\bullet p_i = \emptyset$.

Definition 36: Let (N_N, M_{No}) be a neural Petri net model with $N_N = (P_N, T_N, F_N, X_k, Y_k, W_N, M_{No})$; a place $p_i \in P_N$ is said to be a concluding place if $p_i^\bullet = \emptyset$.

Definition 37: Let (N_N, M_{No}) be a neural Petri net model with $N_N = (P_N, T_N, F_N, X_k, Y_k, W_N, M_{No})$; $p_i, p_j \in P_N$, p_j is promptly reachable from p_i, if $p_i \in {}^\bullet t_{dm}$ and $p_j \in t_{dm}{}^\bullet$, and expressed by $p_j \in PRM(p_i)$.

Definition 38: Let (N_N, M_{No}) be a neural Petri net model with $N_N = (P_N, T_N, F_N, X_k, Y_k, W_N, M_{No})$; $p_i, p_j, p_k \in P_N$, p_k is reachable from p_i, if $p_j \in PRM(p_i)$ and $p_k \in PRM(p_j)$, and expressed by $p_k \in RM(p_i)$.

Definition 39: Let (N_N, M_{No}) be a neural Petri net model with $N_N = (P_N, T_N, F_N, X_k, Y_k, W_N, M_{No})$; $p_i, p_j, p_k \in P_N$, p_k is promptly reachable from p_i with a degree of reachability f, if $p_k \in PRM^f(p_i)$, where f is flows times and $PRM^f(p_i) = PRM(PRM\ (\ldots f\ times)\ldots\ PRM\ (p_j))$.

Definition 40: Let (N_N, M_{No}) be a neural Petri net model with $N_N = (P_N, T_N, F_N, X_k, Y_k, W_N, M_{No})$, where W_N is said to be a place to transition the connectivity matrix, and its element can be expressed as a synaptic weight w_{ij}:

$$W_N = \begin{bmatrix} \sum_{i=1}^{n} w_{ij} = 1 & \forall i,\ p_j \in {}^\bullet t_{dm} \\ & i = 1,2,\ldots,n \\ & j = 1,2,\ldots,m \\ w_{ij} = 0 & \text{otherwise} \end{bmatrix} \tag{4}$$

Definition 41: Let (N_N, M_{No}) be a neural Petri net model with $N_N = (P_N, T_N, F_N, X_k, Y_k, W_N, M_{No})$; if a transition t_{dm} is enabled and fired, the input O_j of all the neurons in the competitive fault detection layers can be generated and expressed as:

$$O_j = \begin{bmatrix} \sum_{i=1}^{n} w_{ij}\, x_i^k & \forall i,\ x_i \in X \\ & i = 1,2,\ldots,n \\ & j = 1,2,\ldots,m \\ & k = 1,2,\ldots,\varphi \\ 0 & \text{otherwise} \end{bmatrix} \tag{5}$$

Definition 42: Let (N_N, M_{No}) be a neural Petri net model with $N_N = (P_N, T_N, F_N, X_k, Y_k, W_N, M_{No})$; the j^{th} neuron whose corresponding input $O_j \in Y_N$ is maximum (i.e., the winner). Therefore, its output value y_j is taken as 1; otherwise, the others' output values are taken as 0 and expressed as

$$Y_k = \begin{bmatrix} \max(O_j) = & O_j > O_i \\ y_j = 1 & i = 1, 2,\ldots,m \\ & i \neq j \\ y_i = 0 & i \neq j \end{bmatrix} \tag{6}$$

Definition 43 **[47]*****:*** Let (N_N, M_{No}) be a neural Petri net model with $N_N = (P_N, T_N, F_N, X_k, Y_k, W_N, M_{No})$; $w_{ij} \in W_N$ is the winner neuron's synaptic weight of the winner j^{th} neuron, whose corresponding input $v_j \in V_N$ and can be updated as

$$w_{ij} = \begin{bmatrix} w_{ij} + \Delta w_{ij} & \forall i = 1,2,\ldots,n \\ w_{ij} & \text{Otherwise} \end{bmatrix} \tag{7}$$

$$\Delta w_{ij} = \gamma\left(\frac{x_i^k}{\delta} w_{ij}\right) \qquad \begin{matrix} i = 1,2,\ldots,n \\ j = 1,2,\ldots,m \\ k = 1,2,\ldots,\varphi \end{matrix} \tag{8}$$

where, $\gamma \to [0,1]$ is a learning rate, which usually takes values [0.01, 0.03]. δ is the number of elements that are equal to 1 in the learning pattern vector $X_k = \cup_{i\in n}\{x_i^k\}$.

Definition 44: Let (N_{CU}, M_{CUo}) be a colored controlled unreliable S^3PR Petri net and (N_N, M_{No}) be a neural Petri net model. We call (N_{NN}, M_{NNo}) a neural colored controlled unreliable S^3PR Petri net model, expressed as $(N_{NN}, M_{NNo}) = (N_{CU}, M_{CUo}) \parallel (N_N, M_{No})$, which is the integration of (N_{CU}, M_{CUo}) and (N_N, M_{No}), where $N_{NN} = (P_{NN}, T_{NN}, F_{NN}, SC, X_k, Y_k, W_N, M_{NNo})$, and

1. $P_{NN} = \{p^0\} \cup P_A \cup P_R \cup P_N \cup \{p_{combined1}\} \cup \{p_{combined2}\}$.
2. $T_{NN} = T \cup T_{DCi} \cup T_{DCo} \cup T_F \cup T_R \cup T_N$.
3. $F_R \subseteq (T_{NN} \times P_{NN}) \cup (P_{NN} \times T_{NN})$.
4. $F_{NN} = F \cup F_{CC} \cup F_{ccr} \cup F_N$
5. $SC = C_R \cup C_F$
6. M_{NNo} is an initial marking of N_{NN}.

Theorem 3: The net (N_{NN}, M_{NNo}) is live.

Proof: We must prove that all transitions T, T_{CCi}, T_{CCo}, T_F, T_R, and T_N in (N_{NN}, M_{NNo}) are live. If there is no failure in $r_u \in P_{UR}$ and for all $t_x \in T$, if $\forall p_i \in {}^\bullet t_x$, $M_{NN}(p_i) > 0$, then t_x can fire. For all $t_y \in T_{CCi}$, if $\forall p_i \in {}^\bullet t_y$, $M_{NN}(p_i) > 0$, then t_y can fire.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999054, IEEE Access

For all $t_z \in T_{CCo}$, if $M_{NN}(p_{combined1}) > 0$, then t_z can fire. For all $t_e \in T_F$, if for all $p_i \in {}^{\bullet}t_e$, $M_{NN}(p_{combined2}) > 0$, then t_e can fire, leading to $M_{NN}(p_{combined2}) > 0$. For all $t_s \in T_R$, if $M_{NN}(p_{combined2}) > 0$, then t_s can fire. For all $t_h \in T_N$, if for all $p_i \in P_N$, $p_i \in {}^{\bullet}t_h$, $M_{NN}(p_i) > 0$, then t_h can fire. Therefore, we can say that the net (N_{NN}, M_{NNo}) is live. □

Based on Definitions 34–44 and an algorithm proposed by [47], the developed learning process and solution neural network Petri net model of the fault detection and treatment algorithm is stated as follows:

Algorithm 3: *(Training process and solution of the proposed Model)*

Input: *An* (N_{NN}, M_{NNo}).
Output: *A set of fault outputs* Y_k
Initialization: *Generate randomly all values of* $w_{ij} \to [0,1]$
Step 1: **if** *a transition* t_{fi} *fires,* **then**
  while $w_{ij} < \theta$, **do** /* θ *is a target weight**/
    for $(1, \varphi, i{+}{+})$, ***do***
      for $(1, n, j{+}{+})$, ***do***
        for $(1, m, k{+}{+})$, ***do***
1. *Choose a pattern* X_k *from* φ *input patterns* φ *to be the neural network input layer;*
2. *Calculate the input* O_j *of all the neurons respectively in the competitive fault detection layers; /* By (5) in Definition 41.*/*
3. *Calculate* y_j; */* By (6) in Definition 42.*/ Transition* t_{dj} *is enabled;*
4. *Update the winner neuron's synaptic weight* w_{ij}; */*By (7) and (8) in Definition 43.*/*

        end for
      end for
    end for
  end **while**
elseif
  break
end **if**
Step 2*: Output a set of faults* Y_k
Step 3: End

E. Computational Complexity

Algorithm 1 is used to design common monitor $p_{combined1}$ based on P-invariants PI and monitors V_p to prevent all M^*_{FBM} in S^{3}PR (N, M_o) with $N = (\{p^0\} \cup P_A \cup P_R, T, F, W, M_o)$ to satisfy the liveness of a Petri net model. Let n is the number of M^*_{FBM}, i.e., $|M^*_{\mathrm{FBM}}| = n$. The "While" loop is executed n times to design PI and V_p in S^{3}PR (N, M_o). Then, all monitors V_p will be merged into common monitor $p_{combined1}$. Therefore, in the worst case, the computational complexity of algorithm 1 is O(n).

Algorithm 2 is used to design a common recovery subnet $p_{combined2}$ for all unreliable resources in controlled S^{3}PR (N_{CN}, M_{CNo}) with $N_{CN} = (\{p^0\} \cup \{p_{combined1}\} \cup P_A \cup P_R, T \cup T_{CCi} \cup T_{CCo}, F \cup F_{CC}, C_R, M_{CNo})$. Each unreliable resource $r_u \in P_{UR}$ may fail when it is idle r_u or in a busy state $H(r_u)$. Thus, P_H as a set of places and defined in Definition 31. Let the number of set of places that require recovery subnets P_{UR} be x, i.e., $|P_{UR}| = x$. In addition, each recovery subnet requires only one common recovery place for all places in P_{UR} and failure transition and recovery transition for each $p_i \in P_{UR}$. Thus, the computational complexity of Algorithm 2 is O($2x$). The "FOR loop" is executed $2x$ times to design the common recovery subnet for all unreliable resource in colored controlled S^{3}PR (N_{CN}, M_{CNo}). Thus, the computational complexity is O($2x$), which is O(x).

Algorithm 3 is used to design training process and solution of the proposed model for a neural colored controlled unreliable S^{3}PR Petri net model (N_{NN}, M_{NNo}) with $(P_{NN}, T_{NN}, F_{NN}, SC, X_k, Y_k, W_N, M_{NNo})$. Algorithm 3 has nested loops, let h is set of iterations to achieve a target weight θ, where $h = \{1, 2, 3, \ldots\}$, y is set of input pattern φ, i.e., $|\varphi| = y$, z is the input neurons n, i.e., $|n| = z$, and e is the output neurons m, i.e., $|m| = e$. The "While" loop is executed h times to achieve a target weight θ in (N_{NN}, M_{NNo}). The "NESTED FOR loops" are executed yze times design training process and solution for the model (N_{NN}, M_{NNo}). Thus, the computational complexity is O($hyze$).

III. NUMERICAL EXAMPLE

To illustrate the proposed methodology, consider the AMS example shown in Fig. 1(a). The S^{3}PR Petri net model was given in Chen et al. [45] and Kaid et al. [3]. The system consists of two machines M1 and M2. Each machine processes one part at a time and one robot R1 holds one part at a time. There are two buffers for loading/unloading. In addition, in the system, two types of part A and B are considered to be processed. The operation sequences for the two-part types are illustrated in Fig. 1(b). The Petri net model of this AMS example is shown in Fig. 2. It comprises 11 places and eight transitions. The places can be defined as the following set partitions: $P_A = \{p_2, p_3, \ldots, p_7\}$, $P_R = \{p_9, p_{10}, p_{11}\}$, and $P^0 = \{p_1, p_8\}$. The Petri net model contains 20 reachable markings. We have $M^*_{\mathrm{FBM}} = \{p_3 + p_5, p_2 + p_5, p_2 + p_6\}$ and $M^*_L = \{p_5 + p_6 + p_7, p_2 + p_3 + p_4\}$.

By considering the application of steps 1 and 2 in Algorithm 1, at the first iteration, assume that I_1 is the place invariant to be computed, and that it satisfies equation (2) for the two legal markings in M^*_L, i.e., $l_5.1 + l_6.1 + l_7.1 \le \beta$ and $l_2.1 + l_3.1 + l_4.1 \le \beta$. Thus, we have two constraints:

$l_5 + l_6 + l_7 \le \beta$ and

$l_2 + l_3 + l_4 \le \beta$.

We have three variables f_1, f_2, and f_3 to reflect whether I_1 prohibits FBM1s $\mathrm{FBM}_1 = p_3 + p_5$, $\mathrm{FBM}_2 = p_2 + p_5$, and $\mathrm{FBM}_3 = p_2 + p_6$, respectively. Thus, we have three constraints:

$l_3 + l_5 \ge \beta + 1 - Q\cdot(1 - f_1)$

$l_2 + l_5 \ge \beta + 1 - Q\cdot(1 - f_2)$, and

$l_2 + l_6 \ge \beta + 1 - Q\cdot(1 - f_3)$.

Subsequently, MFFP1 is described as

MFFBMP1:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999054, IEEE Access

$$max\ f = f_1 + f_2 + f_3$$

subject to

$$l_5 + l_6 + l_7 \le \beta$$
$$l_2 + l_3 + l_4 \le \beta$$
$$l_3 + l_5 \ge \beta + 1 - Q\cdot(1-f_1)$$
$$l_2 + l_5 \ge \beta + 1 - Q\cdot(1-f_2)$$
$$l_2 + l_6 \ge \beta + 1 - Q\cdot(1-f_3)$$
$$\beta \in \{1,2,...\}$$
$$l_i \in \{0,1,2,...\}, \forall i \in \{2,3,4,5,6,7\}$$
$$f_1, f_2, f_3 \in \{0,1\}$$

We solve the MFFBMP1 and obtain the optimal solution as $l_2 = 2$, $l_5 = 1$, $l_6 = 1$, $\beta = 2$, $f_2 = 1$, $f_3 = 1$, and all the other variables are zero. Then, a control place V_{S1} is developed for I_1: $2\mu_2 + \mu_5 + \mu_6 + \mu_{Vs1} = 2$. As a result, I_1 prohibits FBM2 and FBM3, and we have ${}^\bullet V_{S1} = \{2t_2, t_7\}$, $V_{S1}{}^\bullet = \{2t_1, t_5\}$, and $M_{CNo}(V_{S1}) = \beta = 2$. Thus, we have F_{I1} = {FBM2, FBM3}, $M^*_{\text{FBM}} := M^*_{\text{FBM}} - \text{F}_{\text{I1}}$; therefore, M^*_{FBM} = FBM1 = $\{p_3 + p_5\}$.

At the second iteration, let I_2 place invariant be computed and satisfy (2) for the two legal markings in M^*_L, i.e., $l_5.1 + l_6.1 + l_7.1 \le \beta$ and $l_2.1 + l_3.1 + l_4.1 \le \beta$. Thus, we have two constraints:

$$l_5 + l_6 + l_7 \le \beta \text{ and}$$
$$l_2 + l_3 + l_4 \le \beta.$$

We have one variable f_1 to reflect whether I_2 prohibits FBM1 $\text{FBM}_1 = p_3 + p_5$. Thus, we have one constraint:

$$l_3 + l_5 \ge \beta + 1 - Q\cdot(1-f_1)$$

Subsequently, a new MFFP1 is described as below.

MFFBMP1:

$$max\ f = f_1$$

subject to

$$l_5 + l_6 + l_7 \le \beta$$
$$l_2 + l_3 + l_4 \le \beta$$
$$l_3 + l_5 \ge \beta + 1 - Q\cdot(1-f_1)$$
$$\beta \in \{1,2,...\}$$
$$l_i \in \{0,1,2,...\}, \forall i \in \{2,3,4,5,6,7\}$$
$$f_1 \in \{0,1\}$$

We now, solve the new MFFBMP1, with the optimal solution being $l_3 = 1$, $l_5 = 1$, $\beta = 1$, $f_1 = 1$, and all the other variables being zero. Then, a control place V_{S2} is developed for I_2: $\mu_3 + \mu_5 + \mu_{Vs2} = 1$. As a result, I_2 prohibits FBM1; we have ${}^\bullet V_{S2} = \{t_3, t_6\}$, $V_{S2}{}^\bullet = \{t_2, t_5\}$, and $M_{CNo}(V_{S2}) = \beta = 1$. Thus, we have F_{I2} = {FBM1}, $M^*_{\text{FBM}} := M^*_{\text{FBM}} - \text{F}_{\text{I2}}$; therefore, $M^*_{\text{FBM}} = \emptyset$, and steps 1 and 2 are terminated. Thus, in total, two control places are computed for this net.

Next, consider the application of step 3 in Algorithm 1. The two obtained control places are merged into $p_{combined1}$. The output arcs T_{CCo} of $p_{combined1}$ are expressed as $T_{CCo} = \{2t_1, t_2, 2t_5\}$. The input arcs T_{CCi} of $p_{combined1}$ are expressed as $T_{CCi} = \{2t_2, t_3, t_6, t_7\}$. In addition, $M_{CNo}(p_{combined1}) = \sum M_{DCo}(V_S) = M_{DCo}(V_{S1}) + M_{DCo}(V_{S2}) = 2 + 1 = 3$. Thus, we have two color types: $C_R = \{C_{vs1}, C_{vs2}\}$. Therefore, the total number of colored tokens is three, two tokens of C_{vs1} color and one token of C_{vs2} color. Fig. 3 displays the proposed single controller for the controlled colored S^3PR Petri net model shown in Fig. 2, obtained by using Algorithm 1. Fig. 3 shows that if transition t_1 fires, it chooses two tokens with color C_{vs1} from the common place $p_{combined1}$, one token from p_9, one token from the input place p_1, and deposits them into p_2. If transition t_2 fires, it chooses one token with color C_{vs2} from *the* common place $p_{combined1}$, one token from p_2, one token from p_{10}, and deposits them into p_3. In addition, if transition t_5 fires, it chooses one token with color C_{vs1} from the common place $p_{combined1}$, one token with color C_{vs2} from the common place $p_{combined1}$, one token from p_{11}, one token from input place p_8, and deposits them into p_5. If transition t_2 fires, it generates two colors C_{vs1} on the tokens from p_2 and p_{10}, and deposits them into common place $p_{combined1}$. Moreover, if transition t_3 fires, it generates one color C_{vs2} on the tokens from p_3 and p_{11}, and deposits them into the common place $p_{combined1}$. If transition t_6 fires, it generates one color C_{vs2} on the tokens from p_5 and p_{10}, and deposits them into the common place $p_{combined1}$. Finally, if transition t_7 fires, it generates one color C_{vs2} on the tokens from p_6 and p_9, and deposits them into the common place $p_{combined1}$.

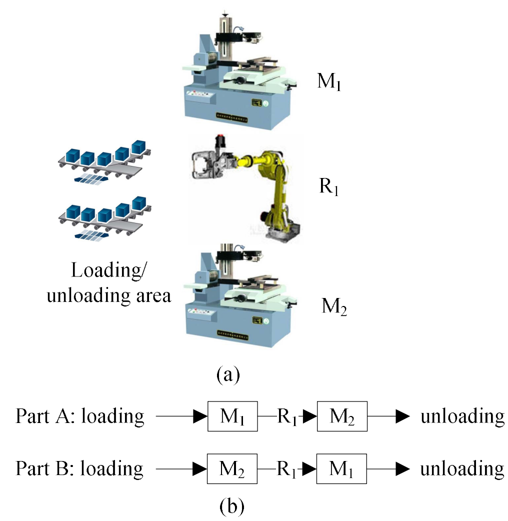


FIGURE 1. (a) AMS example and (b) Operation sequence.

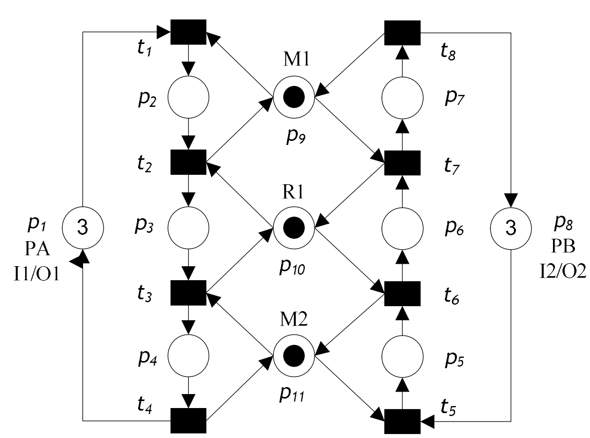


FIGURE 2. S^3PR Petri net model of the AMS.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999054, IEEE Access

To validate Algorithm 1 developed earlier, we have coded Algorithm 1 using the GPenSIM tool [3, 48], and the results from the code are compared with Ezpeleta et al. [49], Chen et al. [45], and Kaid et al. [3]. Table I and Fig. 4 show the results for the number monitors and arcs, initial marking, reachable markings, and liveness. We observe that the obtained controlled colored S^{3}PR Petri net model by Algorithm 1 has minimal structural complexity compared to the methods used by Ezpeleta et al. [49], Chen et al. [45], and Kaid et al. [3]. For a comparison of the performance time criteria, the simulation was performed for 480 min in MATLAB. The results of the simulation are illustrated in Table II, which shows the results in context of the performance time criteria. Overall, the proposed model based on Algorithm 1 is better in terms of resource utilization than the other techniques from the literature while also achieving greater overall throughput but less throughput time per part.

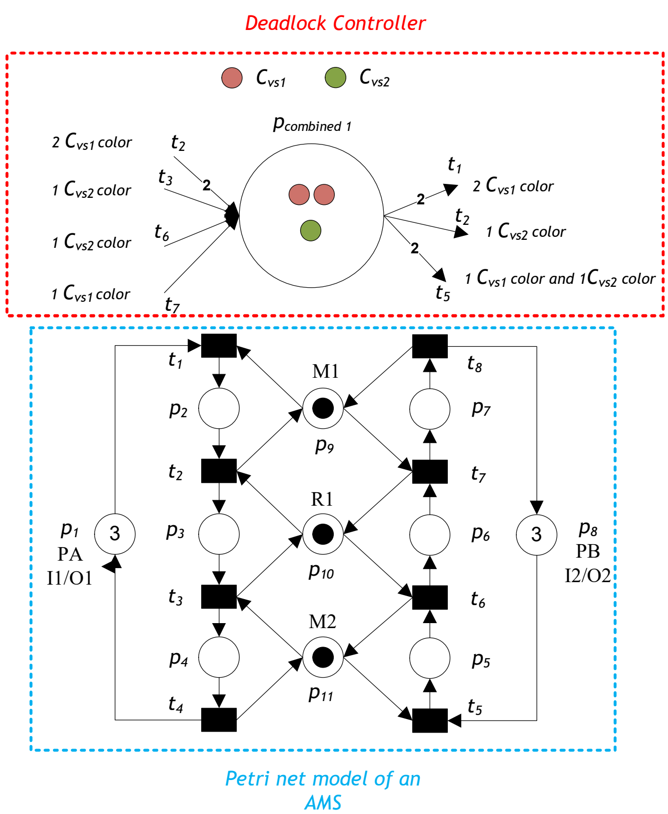


FIGURE 3. **Controlled colored S3PR Petri net model by Algorithm 1.**

TABLE I
STRUCTURAL COMPLEXITY COMPARISON WITH THE EXISTING METHODS.

Parameters	Ezpeleta et al. [49]	Chen et al. [45]	Kaid et al. [3]	Algorithm 1
Monitor	3	2	1	1
Arcs	12	8	7	7
Initial markings	4	3	4	3
Reachable markings	15	15	15	15
Liveness	Live	Live	Live	Live

TABLE II
TIME PERFORMANCE COMPARISON WITH THE EXISTING METHODS.

Parameter	Ezpeleta et al. [49]	Chen et al. [45]	Kaid et al. [3]	Algorithm 1
M1 utilization (%)	49.375	48.75	48.75	49.375
M2 utilization (%)	65	65.8333	65.8334	65
R1 utilization (%)	32.9167	32.9167	32.9166	32.9167
Throughput (parts)	78	78	78	79
Throughput time (min/ part)	6.1538	6.1538	6.1538	6.0759

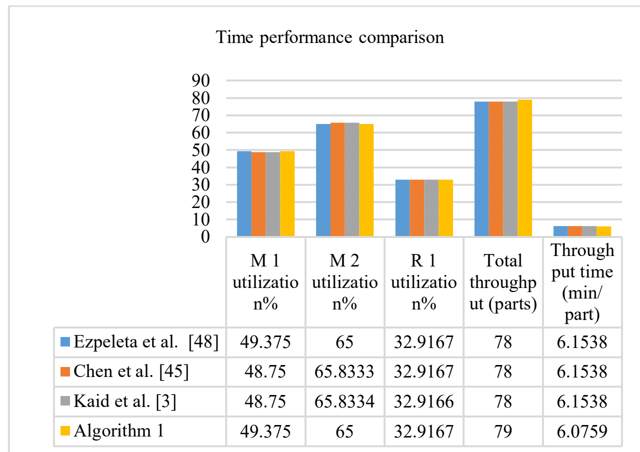


FIGURE 4. **Comparison of Algorithm 1 with the existing methods.**

To illustrate the effectiveness of Algorithm 2, consider an S^{3}PR net displayed in Fig. 3. We have $P_{UR} = \{p_9, p_{10}, p_{11}\}$, $H(p_9) = \{p_2, p_7\}$, $H(p_{10}) = \{p_3, p_6\}$, and $H(p_{11}) = \{p_4, p_5\}$. A common recovery subnet for p_9, p_{10}, and p_{11} is added to an unreliable S^{3}PR net model, as shown in Fig. 5. Thus, the index set NA = {2, 3, 4, 5, 6, 7, 9, 10, 11}, $T_F = \{t_{f2}, t_{f3}, t_{f4}, t_{f5}, t_{f6}, t_{f7}, t_{f9}, t_{f10}, t_{f11}\}$, $T_R = \{t_{r2}, t_{r3}, t_{r4}, t_{r5}, t_{r6}, t_{r7}, t_{r9}, t_{r10}, t_{r11}\}$, and $C_R = \{C_{ccr2}, C_{ccr3}, C_{ccr4}, C_{ccr5}, C_{ccr6}, C_{ccr7}, C_{ccr9}, C_{ccr10}, C_{ccr11}\}$.

If machine 1 breaks down in busy state p_2 or p_7, i.e., t_{f2} or t_{f7} fires, it generates a token of color C_{ccr2} from p_2 or a token of color C_{ccr7} from p_7, and deposits the same into $p_{combined2}$. If machine 1 breaks down in idle state p_9, i.e., t_{f9} fires, it generates a token of color C_{ccr9} from p_9, and deposits it into $p_{combined2}$.Then, machine p_9 is repaired, and the token with color in $p_{combined2}$ moves into p_2, p_7, or p_9 by firing t_{r2}, t_{r7}, or t_{r9}. If the transition t_{r2}, t_{r7}, or t_{r9} fires, it chooses the token with color C_{ccr2}, C_{ccr7}, or C_{ccr9} from $p_{combined2}$ and deposits it into p_2, p_7, or p_9, implying that the recovery of machine 1 p_9 is completed. If robot 1 breaks down in busy state p_3 or p_6, i.e., t_{f3} or t_{f6} fires, it generates a token of color C_{ccr3} from p_3 or a token of color C_{ccr6} from p_6, and deposits the same into the $p_{combined2}$. If robot 1 breaks down in idle state p_{10}, i.e., t_{f10} fires, it generates a token of color C_{ccr10} from p_{10}, and deposits it into $p_{combined2}$. Then, robot 1 p_{10} is repaired, and the token with color in $p_{combined2}$ moves into p_3, p_6, or p_{10} by firing t_{r3}, t_{r6}, or t_{r10}. If the transition t_{r3}, t_{r6}, or t_{r10} fires, it

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999054, IEEE Access

chooses the token with color C_{ccr3}, C_{ccr6}, or C_{ccr10} from $p_{combined2}$ and deposits the same into p_3, p_6, or p_{10}, implying that the recovery of robot 1 p_{10} is completed. Finally, if machine 2 breaks down in busy state p_4 or p_5, i.e., t_{f4} or t_{f5} fires, it generates a token of color C_{ccr4} or a token of color C_{ccr5}, from p_4 or p_5, respectively, and deposits the same into the $p_{combined2}$. If machine 2 breaks down in idle state p_{11}, i.e., t_{f11} fires, it generates a token of color C_{ccr11} from p_{11}, and deposits it into $p_{combined2}$. Then, machine 2 is repaired, and the token with color in $p_{combined2}$ moves into p_4, p_5, or p_{11} by firing t_{r4}, t_{r5}, or t_{r11}. If the transition t_{r4}, t_{r5}, or t_{r11} fires, it chooses the token with color C_{ccr4}, C_{ccr5}, or C_{ccr11} from $p_{combined2}$ and deposits it into p_4, p_5, or p_{11}, implying that the recovery of machine 2 p_{11} is completed.

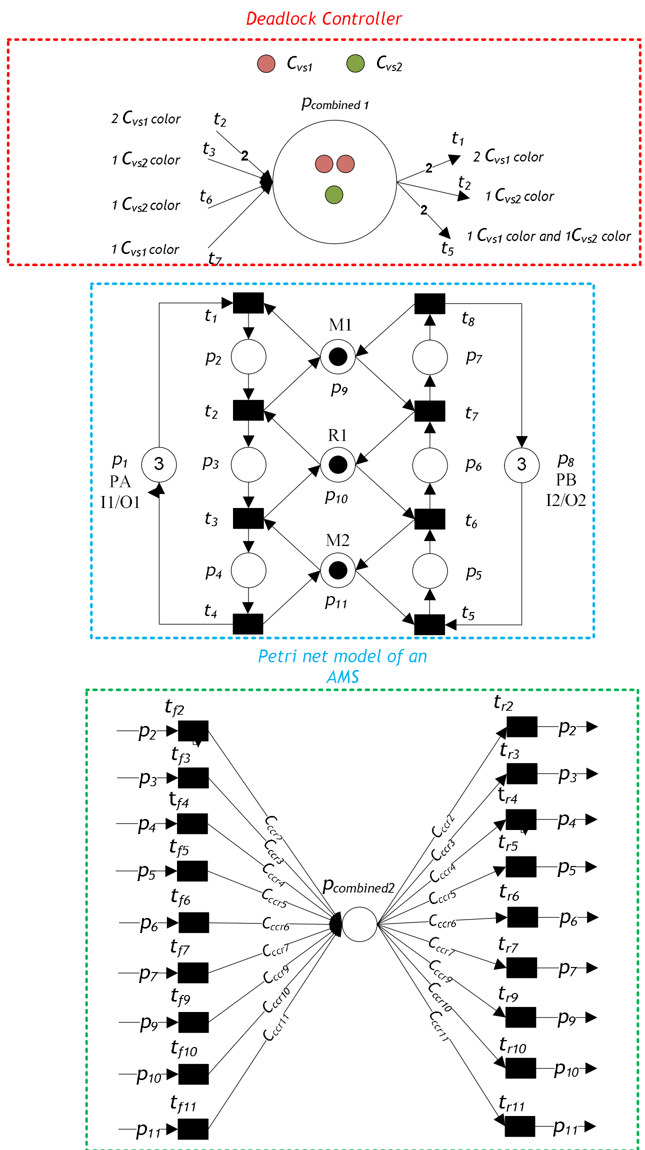


FIGURE 5. **Colored controlled unreliable Petri net model using Algorithm 2.**

To validate the developed Algorithm 2, we have coded the algorithm using the GPenSIM tool [3, 48], and the results from the code are compared with those of Al-Ahmari et al. [6]. Table III and Fig. 6 show the results in the context of the performance time criteria. Overall, the proposed model based on Algorithm 2 is better in terms of resource utilization than that of Al-Ahmari et al. [6]. In addition, the proposed model can obtain greater overall throughput, but less throughput time per part than the latter.

TABLE III
TIME PERFORMANCE COMPARISON WITH THE EXISTING METHODS

Parameter	Al-Ahmari et al. [6]	Algorithm 2
M1 utilization (%)	46.875	48.125
M2 utilization (%)	61.6666	64.1666
R1 utilization (%)	30.8334	32.0834
Throughput (parts)	74	77
Throughput time (min/ part)	6.4865	6.2338

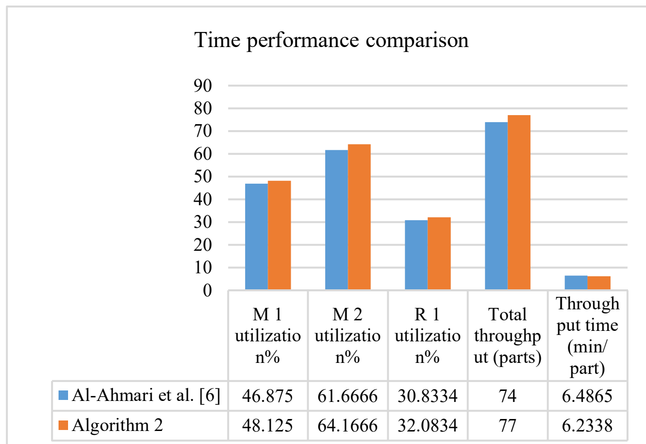


FIGURE 6. **Comparison of Algorithm 2 with the existing methods.**

Finally, to illustrate the performance of the proposed Algorithm 3, consider an unreliable S^3PR net model displayed in Fig. 5. The proposed Algorithm 3 is used to detect the type of failure of the parameters given, which results in various faults. For this purpose, the neural network approach has two phases, training and testing. The neural network is trained during the training phase to detect the underlying significant relation between the selected inputs and outputs. The networks were tested after training with a test data set that was not used for training. When the networks have been trained and tested, they are ready to detect the faults under various operating conditions. The following concerns are to be tackled while developing the model for system failure detection: (a) input and output variable selection, (b) data generation, (c) data normalization, and (d) network structure selection and network training.

Using an unreliable S^3PR net model displayed in Fig. 5, the required data were obtained. The data consisted of four input variables (continuous variables) [50-52], which are listed in Table IV and one output that is labeled as a fault (exactly one fault), is listed in Table V. The output was expressed as [1 0 0 0 0] for fault 1, [0 1 0 0 0] for fault 2, [0 0 1 0 0] for fault 3, [0 0 0 1 0] for fault 4, and [0 0 0 0 1] for fault 5. The proposed neural colored controlled unreliable S^3PR Petri net model was implemented on MATLAB R2015a. A PC with Intel(R) Core (TM) i7-4702MQ CPU @ 2.20 GHz, 16 GB RAM, and running on Windows 10, 64-bit operating system was employed.

In this study, the dataset selected for evaluating the proposed model is segmented into three subsets, which include training, validation, and test sets. In the learning phase, Algorithm 3 uses the training dataset to determine the feature vector; the test phase uses the dataset to calculate the accuracy of fault detection; and the validation phase uses the dataset to decide to stop the iterations to achieve the maximum generalization capacity. The complete dataset comprised 2250 patterns belonging to all of the five faults types. In Furthermore, the dataset contained 75% training, 5% validation, and 20% testing patterns. The generated data were utilized as 1687 patterns for training, 113 patterns for validation and 450 patterns for testing. The training data were completely different from the testing data. Fig. 7 shows the neural colored controlled unreliable S^3PR Petri net model for fault detection in the unreliable model displayed in Fig. 7. Note that only one unreliable resource can fail at a time.

In the input layer displayed in Fig. 7, the data extraction system was made up of an accelerometer x_1, electrical current sensor x_2, strain gage x_3, coolant sensor x_4, sensor in the robot's grip x_5, and acoustic emission sensor x_6. An acquisition signal system collects the signals of these sensors that identify the machine tool states (normal or abnormal). The signal peaks produced by the machine tool may be "random" or "uniform". Signals with uniform peaks imply erroneous programming of the machining parameters or slight tool-wear. On the other hand, signals with random peaks imply very high tool-wear or tool-break. The load or unload process can be monitored by sensors in the robot's grip or wrist. The crashing or collision effects can be detected from the differences between the sensor signals and the kinematics/dynamics effects [53].

In the output layer displayed in Fig. 7, the effects of failures are related to tool wear, tool break, coolant lack, programming mistakes, and the robot's grip or wrist. Failures resulting from programming errors, such as an erroneous machine parameter or non-appropriate tools produce uniform peaks. Failures caused by tool-break produce random peaks. Failures from tool-wear are recognized by uniform peaks and motor oscillations. Failures owing to the robot's grip or wrist are recognized by the differences between sensor signals and kinematics/dynamics effects. Finally, failures resulting from a lack of coolant produce coolant failures. The suggested treatments used to recover a failed machine or robot are shown in Table VI. The treatments include parameter change t_{t1}, tool change t_{t2}, intervention of human operator t_{t3}, or coolant change t_{t4}. If the neural subnet model output is tool wear failure [1 0 0 0 0], the suggested treatment will require a parameter change. If the neural subnet model output is tool break failure [0 1 0 0 0], the suggested treatment is a tool change. When the neural subnet model output is coolant failure [0 0 1 0 0], the suggested treatment is a coolant change. In addition, if the neural subnet model output is a programming error or machining parameter failure [0 0 0 1 0], the suggested treatment is a parameter change. Finally, when the neural subnet model output is a robot's grip or wrist failure [0 0 0 0 1], the suggested treatment will involve intervention by human operator. Many training experiments were conducted to determine the optimal network structure and the best training parameters of the neural networks, which can yield minimal errors during the training phase. Similarly, several training experiments with various numbers of the hidden neurons with 1, 4, 7, 9, and 12 neurons were trained to analyze the effect of the number of neurons in the hidden layer on the performance of fault detection. We used a neural network with 12 hidden layers for this purpose.

Fig. 8 shows the results from the proposed model. The proposed algorithm can detect faults as a function of time. The optimal value of mean square error (MSE) corresponding to 26 iterations is 0.14567 with a learning rate of 0.0001, and the accuracy of the model is 96%, as summarized in Table VII. To illustrate how the value of learning rate γ influences the convergence of the proposed algorithm, Fig. 9 shows how much we change the weights of W at each step. If γ is too small, then the algorithm converges slowly towards the optimal solution. On the other hand, if γ is too large, the algorithm deviates and produces a significant loss of performance. Fig. 10 demonstrates that we achieve the best solution corresponding to a correlation coefficient (R = 0.92074), which indicates the effectiveness of Algorithm 3. In addition, it is observed that the correlation coefficient value properly represents the quality of fault detection. It follows from these results that the proposed algorithm, integrated with regression is better suited and more practical when dealing with problems of detection and treatment of faults.

TABLE IV
NAMES OF THE INPUT VARIABLES IN THE SYSTEM DISPLAYED IN FIG. 7

Input	Description
x_1	The accelerometer detects mechanical vibrations in the structure of the machine that are generated by cutting the force oscillations.
x_2	The current sensor relates and detects variations in the current consumed by the electrical motor and relates it with the condition of the tool and workpiece.
x_3	The tool flexion or tool torsion is detected by the strain gages
x_4	Coolant Sensor to monitor the coolant level
x_5	Sensor in the robot's grip or wrist can detect crashing or collision effects from the differences between the sensor signals and the kinematics/dynamics effects
x_6	Acoustic emission sensor detects acoustic effects of stress waves for tool break detection

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999054, IEEE Access

TABLE V
NAMES OF THE OUTPUTS IN THE SYSTEM DISPLAYED IN FIG. 7

Output	Description
y_1	Tool wear failure
y_2	Tool break failure
y_3	Coolant failure
y_4	Programming errors failure
y_5	Robot's grip or wrist failure

TABLE VI
NAMES OF THE FAULT TREATMENT IN FIG. 7

Fault treatment	Description
t_{t1}	Parameter change
t_{t2}	Tool change
t_{t3}	Intervention of human operator
t_{t4}	Coolant change

TABLE VII
NETWORK PERFORMANCE OF FAULT DETECTION

Description of parameters	**Value**
Training patterns (%)	75
Testing patterns (%)	20
Testing patterns (%)	5
Time (s)	10
MSE	0.14567
Learning rate	0.0001
Accuracy of fault detection (%)	96

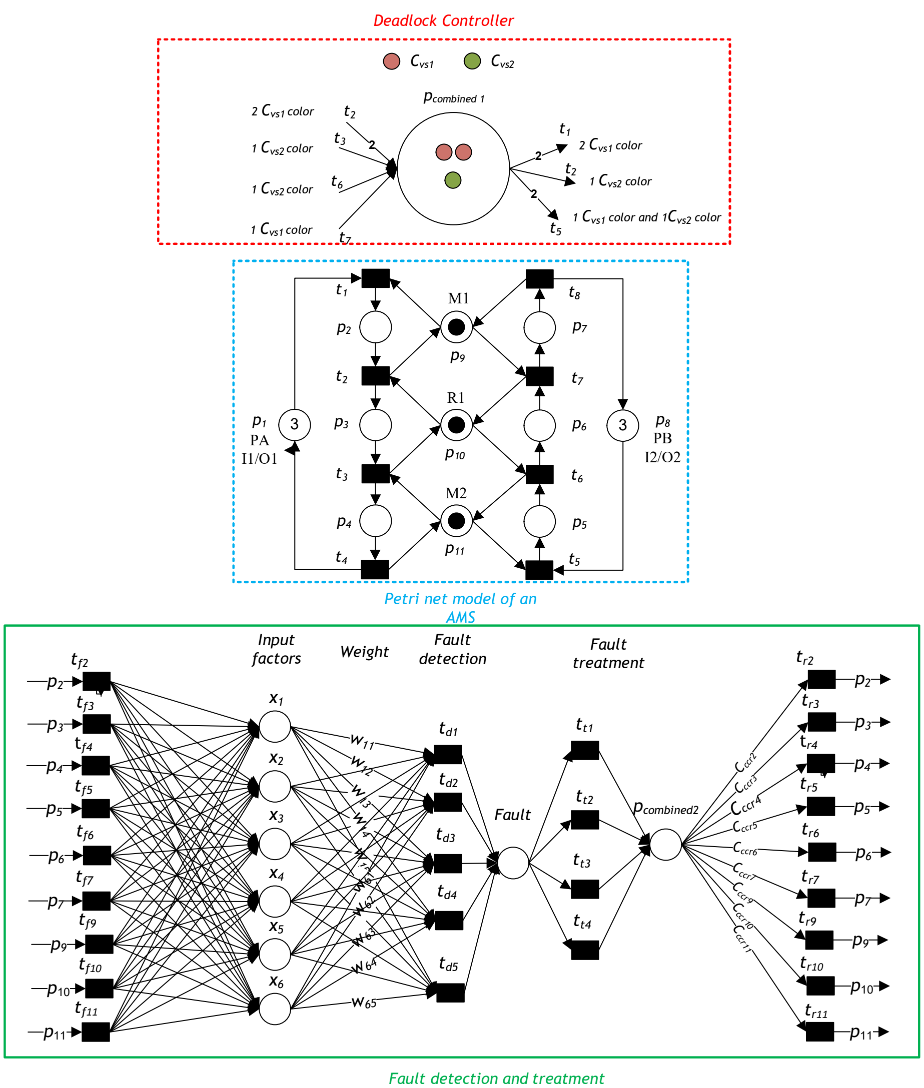


FIGURE 7. **Neural colored controlled unreliable S^{3}PR Petri net model using Algorithm 3.**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999054, IEEE Access

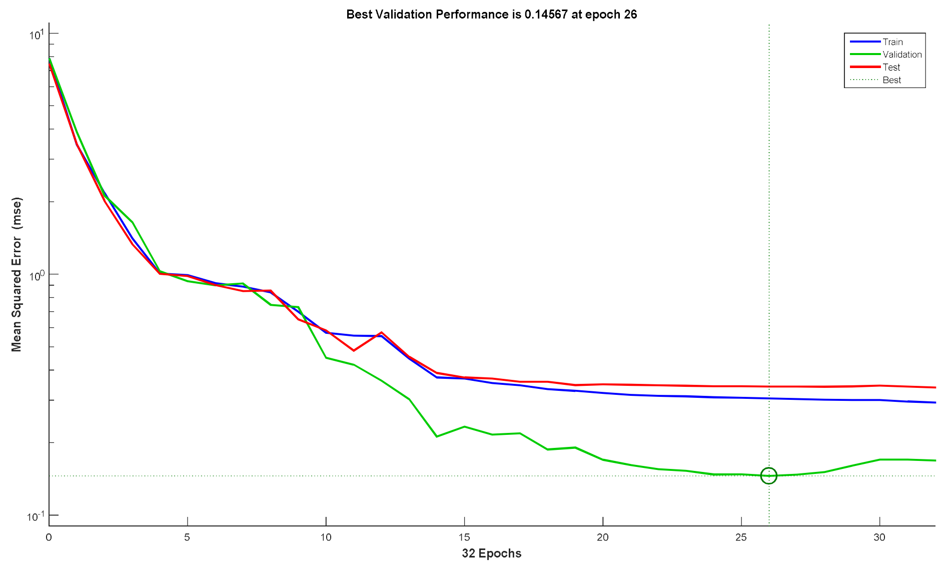


FIGURE 8. **Learning performance of neural colored controlled unreliable S³PR Petri net model using Algorithm 3.**

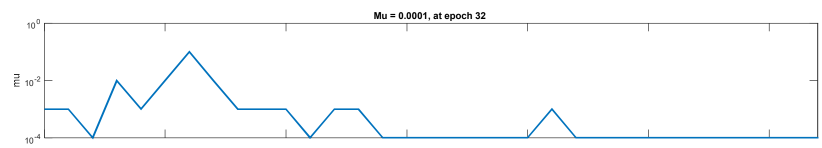


FIGURE 9. **Learning rate and validation check of neural colored controlled unreliable S³PR Petri net model using Algorithm 3.**

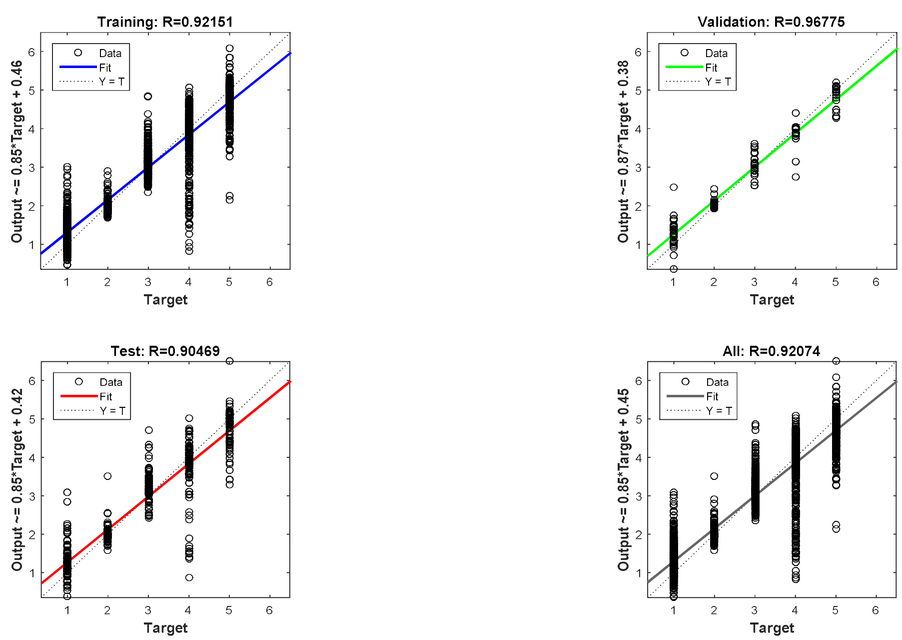


FIGURE 10. Regression results of neural colored controlled unreliable S³PR Petri net model using Algorithm 3.

Finally, Algorithm 3 was compared with that of Al-Ahmari et al. [6] and Algorithm 2. Table VIII and Fig. 11 show the results with respect to the performance time criteria. Overall, the proposed model based on Algorithm 3 is better in terms of resource utilization than that of Al-Ahmari et al. [6] and Algorithm 2. In addition, the proposed model can obtain greater throughput than the latter two, and also achieve less throughput time per part than the latter two.

TABLE VIII
TIME PERFORMANCE COMPARISON WITH THE [6] AND ALGORITHM 2.

Parameter	Al-Ahmari et al. [6]	Algorithm 2	Algorithm 3
M1 utilization (%)	46.875	48.125	49.0909
M2 utilization (%)	61.6666	64.1666	64.6465
R1 utilization (%)	30.8334	32.0834	32.7273
Throughput (parts)	74	77	80
Throughput time (min/ part)	6.4865	6.2338	6.0000

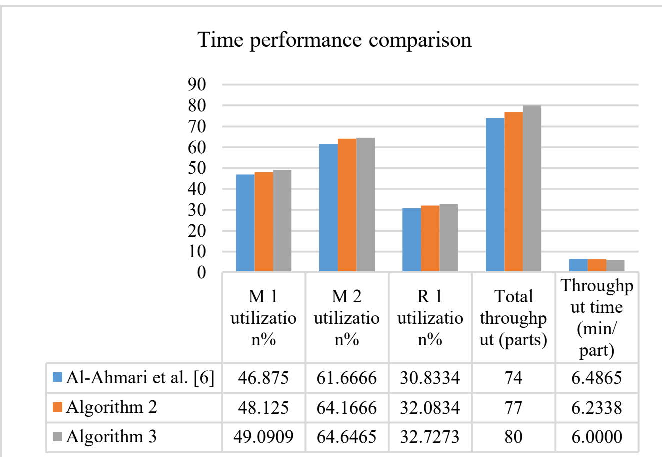


FIGURE 11. **Comparison of Algorithm 3 with [6] and Algorithm 2.**

V. CONCLUSIONS

This paper presents a three-step robust deadlock control strategy for fault detection and treatment of unreliable resource systems in an AMS. In the first step, a controlled system was obtained using MFFBMP1, which did not consider resource failure. The second step addressed the problems of deadlock control caused by resource failures. For all the resource failures in the Petri net model, a common recovery subnet based on colored Petri nets was proposed. The recovery subnet was added to the obtained system at the first step to ensure that the system was reliable. The third step proposed a novel hybrid approach that combined neural networks with colored Petri nets, which were obtained at the second step to detect and treat faults. The proposed strategy was validated using the GPenSIM tool and compared with existing methods in the literature.

The major advantages of the developed strategy are as follows: (i) It has a simpler structure and low-overhead computation, and is more powerful compared with Ezpeleta et al. [49], Chen et al. [45], Kaid et al. [3], and Al-Ahmari et al. [6]. (ii) It takes the descriptive advantages of modular integration of Petri nets and has the ability to learn neurons and reduce the knowledge, as in neural networks. (iii) It can be applicable to other types of complex AMSs (e.g., liquid-crystal display manufacturing and semiconductor fabrication) by considering suitable variables and faults. It can include automated guided vehicles, automated storage and retrieval systems, and machines. (iv) It can consider systems that require sequential and complex resources. (v) It provides a combination of three types of procedures: deadlock-free system without considering resource failure, detection of faults, and treatment of faults. (vi) It considers not only solving deadlock problem in AMSs but also detection and treatment of failures.

The main drawback of the proposed approach is that it may undergo changes in control specifications and requirements, such as changing the system's processing routes or adding new machines, and new products. In the event of the system facing these issues, the system needs to be reconfigured. Then, new deadlock problems can occur in the proposed model. Therefore, our future research will further investigate the proposed methodology to improve its efficiency for valid and rapid reconfiguration of manufacturing systems.

ACKNOWLEDGMENT

This Project was funded by the National Plan for Science, Technology and Innovation (MAARIFAH), King Abdulaziz City for Science and Technology, Kingdom of Saudi Arabia, Award Number (14-ELE69-02-R)

REFERENCES

[1] P. Jalote, *Fault tolerance in distributed systems*: Prentice-Hall, Inc., 1994.

[2] L. Riascos, L. A. Moscato, and P. E. Miyagi, "Detection and treatment of faults in manufacturing systems based on Petri Nets," *Journal of the Brazilian society of mechanical Sciences and Engineering,* vol. 26, no. 3, pp. 280-289, 2004.

[3] H. Kaid, A. Al-Ahmari, Z. Li, and R. Davidrajuh, "Single Controller-Based Colored Petri Nets for Deadlock Control in Automated Manufacturing Systems," *Processes,* vol. 8, no. 1, p. 21, 2020.

[4] H. Kaid, A. Al-Ahmari, Z. Li, and R. Davidrajuh, "Intelligent Colored Token Petri Nets for Modeling, Control, and Validation of Dynamic Changes in Reconfigurable Manufacturing Systems," *Processes,* vol. 8, no. 3, p. 358, 2020.

[5] Y. Chen, Z. Li, M. Khalgui, and O. Mosbahi, "Design of a maximally permissive liveness-enforcing Petri net supervisor for flexible manufacturing systems," *Automation Science and Engineering, IEEE Transactions on,* vol. 8, no. 2, pp. 374-393, 2011.

[6] A. Al-Ahmari, H. Kaid, Z. Li, and R. Davidrajuh, "Strict Minimal Siphon-Based Colored Petri Net Supervisor Synthesis for Automated Manufacturing Systems with Unreliable Resources," *IEEE Access,* 2020.

[7] H. Kaid, A. Al-Ahmari, A. M. El-Tamimi, E. Abouel Nasr, and Z. Li, "Design and implementation of deadlock control for automated manufacturing systems," *South African Journal of Industrial Engineering,* vol. 30, no. 1, pp. 1-23, 2019.

[8] D. Y. Chao, "Improvement of suboptimal siphon- and FBM-based control model of a well-known S^3PR," *Automation Science and Engineering, IEEE Transactions on,* vol. 8, no. 2, pp. 404-411, 2011.

[9] A. Ghaffari, N. Rezg, and X. Xie, "Design of a live and maximally permissive Petri net controller using the theory of regions," *Robotics and Automation, IEEE Transactions on,* vol. 19, no. 1, pp. 137-141, 2003.

[10] M. Uzam, "The use of the Petri net reduction approach for an optimal deadlock prevention policy for flexible manufacturing systems," *The International Journal of Advanced Manufacturing Technology,* vol. 23, no. 3-4, pp. 204-219, 2004.

[11] D. Sun, Y. Chen, M. A. El-Meligy, M. A. F. Sharaf, N. Wu, and Z. Li, "On algebraic identification of critical states for deadlock control in automated manufacturing systems modeled with Petri nets," *IEEE Access,* vol. 7, pp. 121332-121349, 2019.

[12] E. A. Nasr, A. M. El-Tamimi, A. Al-Ahmari, and H. Kaid, "Comparison and Evaluation of Deadlock Prevention Methods for Different Size Automated Manufacturing Systems," *Mathematical Problems in Engineering,* vol. 501, pp. 1-19, 2015.

[13] Z. Li and M. Zhou, "Elementary siphons of Petri nets and their application to deadlock prevention in flexible manufacturing systems," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on,* vol. 34, no. 1, pp. 38-51, 2004.

[14] M. Uzam, "An optimal deadlock prevention policy for flexible manufacturing systems using Petri net models with resources and the theory of regions," *The International Journal of Advanced Manufacturing Technology,* vol. 19, no. 3, pp. 192-208, 2002.

[15] M. Uzam and M. Zhou, "Iterative synthesis of Petri net based deadlock prevention policy for flexible manufacturing systems," in *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, 2004, pp. 4260-4265.

[16] Y.-L. Pan, C.-Y. Tseng, and T.-C. Row, "Design of improved optimal and suboptimal deadlock prevention for flexible manufacturing systems based on place invariant and reachability graph analysis methods," *Journal of Algorithms & Computational Technology,* vol. 11, no. 3, pp. 261-270, 2017.

[17] M. Zhao and M. Uzam, "A suboptimal deadlock control policy for designing non-blocking supervisors in flexible manufacturing systems," *Information Sciences,* vol. 388, pp. 135-153, 2017.

[18] M. Abdulaziz, E. A. Nasr, A. Al-Ahmari, H. Kaid, and Z. Li, "Evaluation of Deadlock Control Designs in Automated Manufacturing Systems," in *International Conference on Industrial Engineering and Operations Management*, USA, 2015.

[19] S. Wang, D. You, and M. Zhou, "A necessary and sufficient condition for a resource subset to generate a strict minimal siphon in S 4PR," *IEEE Transactions on Automatic Control,* vol. 62, no. 8, pp. 4173-4179, 2017.

[20] M. A. Lawley and W. Sulistyono, "Robust supervisory control policies for manufacturing systems with unreliable resources," *IEEE Transactions on Robotics and Automation,* vol. 18, no. 3, pp. 346-359, 2002.

[21] F.-S. Hsieh, "Robustness analysis of Petri nets for assembly/disassembly processes with unreliable resources," *Automatica,* vol. 42, no. 7, pp. 1159-1166, 2006.

[22] S. Wang, S. F. Chew, and M. A. Lawley, "Using shared-resource capacity for robust control of failure-prone manufacturing systems," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans,* vol. 38, no. 3, pp. 605-627, 2008.

[23] S. F. Chew, S. Wang, and M. A. Lawley, "Robust supervisory control for product routings with multiple unreliable resources," *IEEE Transactions on Automation Science and Engineering,* vol. 6, no. 1, pp. 195-200, 2009.

[24] G. Liu, Z. Li, K. Barkaoui, and A. M. Al-Ahmari, "Robustness of deadlock control for a class of Petri nets with unreliable resources," *Information Sciences,* vol. 235, pp. 259-279, 2013.

[25] H. Yue, K. Xing, and Z. Hu, "Robust supervisory control policy for avoiding deadlock in automated manufacturing systems with unreliable resources," *International journal of production research,* vol. 52, no. 6, pp. 1573-1591, 2014.

[26] H. Yue, K. Xing, H. Hu, W. Wu, and H. Su, "Robust supervision using shared-buffers in automated manufacturing systems with unreliable resources," *Computers & Industrial Engineering,* vol. 83, pp. 139-150, 2015.

[27] F. Wang, K.-Y. Xing, M.-C. Zhou, X.-P. Xu, and L.-B. Han, "A robust deadlock prevention control for automated manufacturing systems with unreliable resources," *Information Sciences,* vol. 345, pp. 243-256, 2016.

[28] Y. Feng, K. Xing, Z. Gao, and Y. Wu, "Transition cover-based robust Petri net controllers for automated manufacturing systems with a type of unreliable resources," *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* vol. 47, no. 11, pp. 3019-3029, 2016.

[29] G. Liu, P. Li, Z. Li, and N. Wu, "Robust Deadlock Control for Automated Manufacturing Systems With Unreliable Resources Based on Petri Net Reachability Graphs," *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* vol. 49, no. 7, pp. 1371-1385, 2018.

[30] N. Ran, J. Hao, S. Wang, Z. Dong, Z. He, Z. Liu, and Y. Ruan, "K-codiagnosability verification of labeled Petri nets," *IEEE Access,* pp. 1-8, 2019.

[31] X. Li, G. Liu, Z. Li, N. Wu, and K. Barkaoui, "Elementary Siphon-Based Robust Control for Automated Manufacturing Systems With Multiple Unreliable Resources," *IEEE Access,* vol. 7, pp. 21006-21019, 2019.

[32] Z. Li and M. Zhou, *Deadlock resolution in automated manufacturing systems: a novel Petri net approach*: Springer Science & Business Media, 2009.

[33] G. Liu, L. Zhang, L. Chang, A. Al-Ahmari, and N. Wu, "Robust deadlock control for automated manufacturing systems based on elementary siphon theory," *Information Sciences,* vol. 510, pp. 165-182, 2020.

[34] Z. Li and M. Zhou, "Control of elementary and dependent siphons in Petri nets and their application," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on,* vol. 38, no. 1, pp. 133-148, 2008.

[35] Z. Li and M. Zhao, "On controllability of dependent siphons for deadlock prevention in generalized Petri nets," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans,* vol. 38, no. 2, pp. 369-384, 2008.

[36] C. Zhong and Z. Li, "Self-liveness of a class of Petri net models for flexible manufacturing systems," *IET control theory & applications,* vol. 4, no. 3, pp. 403-410, 2010.

[37] A. Al-Ahmari, H. Kaid, Z. Li, and R. J. I. A. Davidrajuh, "Strict Minimal Siphon-Based Colored Petri Net Supervisor Synthesis for Automated Manufacturing Systems With Unreliable Resources," vol. 8, pp. 22411-22424, 2020.

[38] Y. Maki and K. A. Loparo, "A neural-network approach to fault detection and diagnosis in industrial processes," *IEEE Transactions on Control Systems Technology,* vol. 5, no. 6, pp. 529-541, 1997.

[39] X.-Q. Liu, H.-Y. Zhang, J. Liu, and J. Yang, "Fault detection and diagnosis of permanent-magnet DC motor based on parameter estimation and neural network," *IEEE transactions on industrial electronics,* vol. 47, no. 5, pp. 1021-1030, 2000.

[40] L. A. M. Riascos and P. E. Miyagi, "Supervisor System for Detection and Treatment of Failures in Manufacturing Systems using Distributed Petri Nets," in *Manufacturing, Modelling, Management and Control:(MIM 2001): a Proceedings Volume from the IFAC Workshop, Prague, Czech Republic, 2-4 August 2001*, 2001, p. 83.

[41] L. A. Riascos, F. G. Cozman, and P. E. Miyagi, "Detection and treatment of faults in automated machines based on Petri nets and Bayesian networks," in *2003 IEEE International Symposium on Industrial Electronics (Cat. No. 03TH8692)*, 2003, pp. 729-734.

[42] P. E. Miyagi and L. A. M. J. C. E. P. Riascos, "Modeling and analysis of fault-tolerant systems for machining operations based on Petri nets," vol. 14, no. 4, pp. 397-408, 2006.

[43] S. Rajakarunakaran, P. Venkumar, D. Devaraj, and K. S. P. Rao, "Artificial neural network approach for fault detection in rotary system," *Applied Soft Computing,* vol. 8, no. 1, pp. 740-748, 2008.

[44] H. Honggui, L. Ying, and Q. Junfei, "A fuzzy neural network approach for online fault detection in waste water treatment process," *Computers & Electrical Engineering,* vol. 40, no. 7, pp. 2216-2226, 2014.

[45] Y. Chen, Z. Li, and M. Zhou, "Behaviorally optimal and structurally simple liveness-enforcing supervisors of flexible manufacturing systems," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on,* vol. 42, no. 3, pp. 615-629, 2012.

[46] Y. Chen and Z. Li, *Optimal supervisory control of automated manufacturing systems*: CRC Press, 2013.

[47] X.-Q. Wu, "An intelligent Petri nets model based on competitive neural network," in *International Conference on Computer Supported Cooperative Work in Design*, 2004, pp. 388-397.

[48] R. Davidrajuh, *Modeling Discrete-Event Systems with GPenSIM: An Introduction*: Springer, 2018.

[49] J. Ezpeleta, J. M. Colom, and J. Martinez, "A Petri net based deadlock prevention policy for flexible manufacturing systems," *Robotics and Automation, IEEE Transactions on,* vol. 11, no. 2, pp. 173-184, 1995.

[50] M. Santos, "Analysis of Monitoring of Tool-wear in Frontal Milling based on Application of Sensors," *Dr. Eng. Thesis, Escola Politécnica da USP, São Paulo, SP, 184p,* 1998.

[51] V. Cunha, M. Santos, and C. Tu, "End Mill Sensor System for Tool Condition Monitoring," *M. Eng. Dissertation, Escola Politécnica da USP. São Paulo, SP, 136p,* 2000.

[52] G. Byrne, D. Dornfeld, I. Inasaki, G. Ketteler, W. König, and R. Teti, "Tool condition monitoring (TCM)—the status of research and industrial application," *CIRP annals,* vol. 44, no. 2, pp. 541-567, 1995.

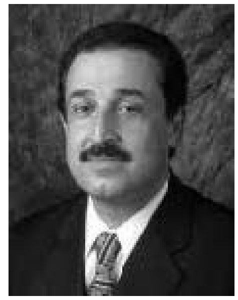
[53] H. Asada and J.-J. Slotine, *Robot analysis and control*: John Wiley & Sons, 1986.

HUSAM KAID is a researcher and PhD student in the Industrial Engineering Department, College of Engineering, King Saud University, Saudi Arabia. He received his BS in Industrial Engineering from the University of Taiz, Taiz, Yemen, in 2010. He received his MS in Industrial Engineering from the King Saud University, Saudi Arabia, in 2015. His research areas and specialties are design and analysis of manufacturing systems, deadlock control in manufacturing systems, supply chain, simulation, operations research, optimization techniques, and bibliometric network analysis.

ABDULRAHMAN AL-AHMARI a Professor of industrial engineering with King Saud University, Riyadh, Saudi Arabia. He received the Ph.D. degree in manufacturing systems engineering from the University of Sheffield, Sheffield, U.K., in 1998. He worked as Dean of the Advanced Manufacturing Institute, Chairman of Industrial Engineering Department and He led a number of funded projects from different organizations in Saudi Arabia. He has published papers in leading Journal of Industrial and Manufacturing Engineering. His current research interests include advanced manufacturing technologies, Petri nets, analysis and design of manufacturing systems, computer integrated manufacturing, optimization of manufacturing operations, flexible manufacturing systems and cellular manufacturing systems, and applications of decision support systems in manufacturing.

EMAD ABOUEL NASR received the Ph.D. degree in industrial engineering from the University of Houston, TX, USA, in 2005. He is currently a Professor with the Industrial Engineering Department, College of Engineering, King Saud University, Saudi Arabia, and an Associate Professor with Mechanical Engineering Department, Faculty of Engineering, Helwan University, Egypt. His current research interests include CAD, CAM, rapid prototyping, advanced manufacturing systems, supply chain management, and collaborative engineering.

ADEL AL-SHAYEA was a consultant at King Saud University Rector's Of_ce and worked for SABIC Marketing Ltd., Riyadh, and at the Institute of Public Administration (IPA). In addition, he is a consultant in the King Abdullah Institute of Research and Consulting Studies. He is a consultant industrial engineer (CE-SCE). He is a member of the Saudi Council of Engineers (SCE), Saudi Arabia, as well as a member of several committees such as the national committee for the codification and standardization of operation and maintenance works. He is currently the Assistant Vice President for academic and educational affairs at King Saud University. He participated and conducted several consultative works for governmental and private organizations. He also refereed several engineering works in Saudi Arabia.



ALI K. KAMRANI received the B.S. degree in electrical engineering, the M.Eng. degree in electrical engineering, the M.Eng. degree in computer science and engineering mathematics, and the Ph.D. degree in industrial engineering from the University of Louisville, Louisville, Kentucky. He is currently an Associate Professor of industrial engineering with the University of Houston and the departments Ph.D. Program Coordinator and Advisor. He is the Founder and the Director of the Free Form Fabrication and Rapid Prototyping (FFF&RP) Laboratory. He is also developing the Design and Manufacturing Automation (D&MA) Laboratory, College of Engineering. Prior to joining the University of Houston, he was an Associate Professor of industrial and manufacturing systems engineering with the University of Michigan-Dearborn, an Adjunct Professor Faculty with Wayne State University, and a Faculty Member for Program in Manufacturing (PIM), the University of Michigan at Ann Arbor Interdisciplinary Program. He is the Founder and previous Coordinator of the Rapid Prototyping Laboratory, College of Engineering and Computer Science, University of Michigan-Dearborn.

MOHAMMED A. NOMAN is a Researcher and PhD. Student in Industrial Engineering Department, College of Engineering, King Saud University, Saudi Arabia. He received his M. Sc. in Industrial Engineering Department, College of Engineering, King Saud University, Saudi Arabia, in 2019. He also received his BS in Industrial Engineering from University of Taiz, Yemen, in 2013. His research areas and specialties are maintenance, safety, human factors, operation researches, optimization techniques, statistical quality control, process monitoring and performance analysis, and bibliometric network analysis.

HAITHAM A. MAHMOUD received the Ph.D. degree in industrial engineering from the University of Helwan, Egypt, in 2012. He is currently an Assistant Professor in the Department of Industrial Engineering, College of Engineering, King Saud University, Riyadh, Saudi Arabia, and the Mechanical Engineering Department, Faculty of Engineering, Helwan University, Egypt. He worked as an engineering consultant for several industrial organizations in Egypt. His current research interests include optimization modeling, theory, and algorithm design with applications in waste management and energy management, financial engineering, and big data.