Received 13 March 2023, accepted 6 April 2023, date of publication 12 April 2023, date of current version 1 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3266456

RESEARCH ARTICLE

An Optimal Transition-Based Recovery Policy for Controlling Deadlock Within Flexible Manufacturing Systems Using Graph Technique

MAHMOUD SALAHELDIN ELSAYED[1], KHALED KEFI[1,2], AND ZHIWU LI[3], (Fellow, IEEE)
[1]Department of Computer Sciences, Faculty of Computing and Information Technology, Northern Border University, Rafha 76321, Saudi Arabia
[2]Department of Mathematics, Faculty of Sciences of Tunis, University of Tunis El Manar, Tunis 1060, Tunisia
[3]Institute of Systems Engineering, Macau University of Science and Technology, Taipa, Macau SAR, China

Corresponding author: Mahmoud Salaheldin Elsayed (mahmoud.al-sayed@nbu.edu.sa)

This work was supported by the Deanship of Scientific Research, Northern Border University, Arar, Saudi Arabia, under Grant CSCR-2022-11-1781.

ABSTRACT Weighted Petri nets are common tools for modeling and validating discrete event systems involving resource allocation, such as flexible manufacturing systems (FMSs). A subclass of weighted Petri nets called a system of sequential systems with shared resources (S^4R) has the power for modeling complex FMSs where the execution of an operation may require multiple resource types and multiple units of some resource types. Deadlock resolution is a crucial issue for the operation of an FMS. A direct and efficient policy is developed in this paper for detecting deadlock markings by extracting a weighted resource flow graph (WRFG) from an S^4R and recovering them by synthesizing a recovery-transition-based controller. This study contributes to the field with five folds: 1) with S^4R, an efficient integrated policy is put forward for robust supervisor synthesis; 2) it enhances an algorithm for extracting the WRFG from an S^4R to reveal the shared resource competitions by different processes; 3) to detect partial deadlock markings, a technique for finding weighted circular wait graphs (WCWGs) in WRFG is presented; 4) with WCWGs, an algorithm is designed for the design of recovery-transition-based controller such that the resulting controlled system becomes deadlock-free; and 5) it presents a comprehensive analysis to demonstrate the proposed method by using the Integrated Net Analyzer (INA). With the proposed policy, it is not necessary to generate a reachability graph, making the method efficient. Finally, the performance of the proposed policy is illustrated by some commonly used examples.

INDEX TERMS Multi-unit resource system, flexible manufacturing system, deadlock detection and recovery, liveness, weighted Petri nets, graph theory.

I. INTRODUCTION

Multi-unit resource systems (MURSs) are made up of multiple processes and resources. Flexible manufacturing systems (FMSs) [1], [2], [3], [4], [5], automated manufacturing systems (AMSs) [6], [7], [8], [9], [10], [11], [12], various resource allocation systems (RASs) [14], [15], and multiprocessor system-on-chips (MPSoCs) [16] are examples of such systems, which fall into the category of discrete event systems (DESs) [13]. The resources in an FMS include robots, machines, fixtures, and buffers. As a kind of MURSs, an FMS is capable of processing multiple types of parts in accordance with a predefined sequence of operations by sharing resources. Circular wait [17] caused by shared-resource competition is the indication of deadlock occurrence that lowers the throughput rates and resource utilization [18], [19], [20], and even leads to catastrophic outcomes. Thus, it is crucial to resolve the deadlock problem in FMSs.

Deadlock resolution issues for MURSs have been extensively investigated by researchers and practitioners [21], [22], [23], [24], which is also important for production scheduling [89]. Depending on the number of

The associate editor coordinating the review of this manuscript and approving it for publication was Bidyadhar Subudhi.

instances (or units) of each resource type, an FMS is categorized as a single- or multi-unit resource system. If there is one instance only for every resource type in an FMS, it is a single-unit resource system; otherwise, it is an MURS. In FMSs, the resource allocation states can be described by a graph. For a single-unit resource system, a circuit in such a graph necessarily implies the occurrence of a deadlock. The deadlock issue in an MURS is more complicated due to the fact that a circuit in such a graph may not result in a deadlock since a node on the circuit may represent an instance of a multi-capacity resource. It is worth noting that, as a special kind of graphs for modeling, Petri nets (PNs) have found widespread use as a general model of many real-world systems [25], [26], [27], [28], [29], [30], [31], [32]. This paper extends the policy developed in [33] for deadlock detection and recovery in FMSs.

Different from the work in [33], to make it possible to describe the requirements that multiple units of a resource type should be used to perform an operation, a generalized class of Petri nets (S^4R) [34] is adopted in this paper. As a result, more complex real-world FMSs can be modeled well such that the well-known PN classes S^3PR [35], [36] and WS^3PR [11] are proper subclasses of S^4Rs.

For deadlock resolution in FMSs, several tools are available for system modeling, including Petri nets [42], [43], [44], [45], [46], automata [39], [40], [41], and graph theory [37], [38]. Because of their inherent characteristics, among them, PNs are popularly adopted [47], [48], [49], [50], [51], [52], [53], [54], [55], [56]. Based on PNs, three types of methods exist for deadlock control, and they are deadlock detection and recovery (DDR) [57], [58], deadlock avoidance [22], [59], and deadlock prevention [60], [61].

It is straightforward to apply the first one. By this type of method, it does not make efforts to avoid the occurrence of deadlocks; instead, it manages to detect a deadlock when it occurs. Once detected, some recovery policies are activated to bring the deadlocked state back to a normal state by reallocating the resources. In this way, it makes a net live. This paper develops a PN-based method.

Generally, with PNs, there are mainly two types of techniques for deadlock resolution in FMSs: reachability graph analysis [66], [67], [68] and structural analysis [62], [63], [64], [65]. With the latter, often it leads to deadlock prevention strategies.

Based on structural analysis techniques, typically as a structure in PNs, siphons are controlled to develop deadlock prevention policies for generalized PNs [69], [70], [71], [72], [73]. This is done by using monitors (control places) to prevent siphons from being insufficiently marked in generalized PNs. Nevertheless, when there are a large number of siphons in the PN model for a system one would face the high structural complexity issue for designing monitor-based supervisors using siphons. There are many research reports on deadlock resolution using deadlock prevention policies, while few works are done for that using deadlock detection and recovery (DDR) policies.

Usually, by reachability graph analysis, one can get a most behaviorally permissive (or optimal) supervisor, since such an analysis can discover the entire state space of a PN model. As various markings are clearly identified by a reachability graph for a PN model, such a graph is also useful for recovery policy development. However, it is always plagued by the infamous state explosion problem.

Although many methods are developed in the literature to synthesize controllers for preventing deadlocks, almost all of them design a controller that is formed by using additional control places and transitions. Nevertheless, a new method is put forward by Huang et al. [74]. Instead of adding control places, it introduces control transitions to the original PN model to make dead markings live. It is shown that the resulting controlled system is more permissive than the one obtained by using the place-adding-based ones. However, for this type of controller synthesis, no formalized algorithm is developed for designing control transitions and minimizing the number of such transitions.

To deal with this issue, a new recovery policy is proposed by Chen et al. [75] and two ways for realizing the policy are derived. One of the ways is to design the control transitions by an iterative strategy such that a control transition is obtained at each iteration. To do so, at each iteration, it formulates an integer linear programming problem (ILPP) and solves it. The other way is to develop an integrated ILPP and solve it to get all the control transitions at once. Although both methods are applicable to all classes of PN models for FMSs, the number of variables and constraints in the obtained ILPPs is very large such that the computational costs are enormous.

In [76], an iterative procedure is proposed to design controllers formed by control transitions. With iteration, it aims to avoid the state explosion problem. However, it needs to check the termination conditions at each iteration, slowing the computational speed down.

Based on the transitions that can access the entire reachable space by firing the recovery transitions, Row and Pan [77] propose a deadlock recovery policy. However, it can neither avoid generating the reachability graph for the PN model. They also propose an iterative technique to compute control transitions by following the crucial dead marking concept in [78].

An iterative vector intersection technique is developed in [79] to decide a recovery transition at each iteration. The obtained recovery transition can recover as many deadlock markings as possible. Although this approach does not need to solve ILPPs and can result in a live controlled system with all reachable markings, a reachability graph for enumerating all deadlock markings is still inevitable. Pan [80] describes an iterative policy for determining control transitions. Although his policy results in fewer control transitions than the previous methods, it is necessary to generate a reachability graph at every iteration. As a result, it cannot be used in large systems.

Latterly, Lu et al. [33] propose a novel DDR policy and develop an algorithm for the construction of a resource flow graph (RFG) of the PN model without generating a reachability graph for the model. The obtained RFG can then be used to generate a set of loop graphs. With the relationships between partial deadlocks and loop graphs, it then synthesizes recovery transitions for the loop graphs to form a controller such that the resulting controlled system is deadlock-free. Indeed, this method can make the controlled system deadlock-free without generating a reachability graph. Nevertheless, it is applicable to ordinary PNs that cannot model complex real-world systems, which is a serious limitation.

This paper focuses on the DDR policy for those FMSs presented by Lu et al. in [33]. Different from [33], this paper adopts S^{4}R, a weighted class of PNs, to model more complex real-world systems, where an operation needs to be performed by multiple units from multiple resource types. Such systems are multi-unit resource ones that contain concurrently cyclic sequential processes, and disassembly and assembly operations. First, with S^{4}R, we design an algorithm to construct the weighted RFG (WRFG), revealing the shared resource competition by various processes. Then, from the WRFG, a set of weighted circular wait graphs (WCWGs) can be constructed. With the relationship between partial deadlocks and WCWGs, we can detect partial deadlocks. An algorithm is presented for designing recovery transitions for deadlocked markings in a set of WCWGs. A complete analysis of the resulting controlled system is performed to demonstrate deadlock-freeness of the controlled S^{4}R for all reachable markings. Based on the Integrated Net Analyzer (INA), S^{4}R is validated and tested via simulation. With the proposed DDR policy, the computation of a reachability graph for an S^{4}R model is avoided, which reduces the risk of state explosion.

The remainder of this work is constructed as follows. The next section presents the transition-based DDR policy within S^{4}R and an algorithm for the construction of a WRFG. This algorithm and the concept of WCWG are demonstrated via a simple example. Then, it shows how to detect partial deadlocks by using the relationship between WCWGs and partial deadlock markings. Section III introduces an algorithm for creating recovery transitions such that when there is a partial deadlock, these transitions are enabled. The resulting net is deadlock-free for all the reachable markings when the decided recovery transitions are added into the model. The proposed policy is demonstrated by using some commonly used examples in Section IV. Section V summarizes the work with conclusions and future research directions. The necessary background, such as the definitions and properties of a system of sequential systems with shared resources (S^{4}R), the fundamentals of PNs, and the deadlock resolution issue in multi-unit resource systems are clarified in the Appendix.

II. THE PROPOSED TRANSITION-BASED RECOVERY POLICY

This section presents a transition-based recovery policy for deadlock control for systems modeled by an S^{4}R PN. This policy consists of three stages. At the first stage, it shows how to construct a WRFG from an S^{4}R. At the second stage, it gives how to find WCWGs from WRFG. The WCWGs represent a straightforward approach for detecting the competition among the processes and shared resources. By analyzing the relationship between WCWGs and partial deadlocks, one can detect all the partial deadlocks. The last stage explains how to construct the transitions that are enabled at the deadlocked markings and make the system back to live.

A. WEIGHTED RESOURCE FLOW GRAPH (WRFG)

No preemption, hold and wait, mutual exclusion, and circular wait are necessary conditions for a deadlock to occur in a resource allocation system [17]. By the features of FMSs, the first three conditions are always held, and only the last one can be broken by properly allocating the resources in a system. To avoid the occurrence of a deadlock, one needs to make sure that there will be no circular wait.

Definition 1: Given an S^{4}R PN $(N, M_0) = (P_A \cup P^0 \cup P_R, T, F, W, M_0)$, a WRFG in N is a digraph WRFG $= (V, A, W)$, where

1) $V = P_A \cup P_R$.
2) $A \subseteq V \times V$ with $A = A1 \cup A2$, satisfying the following statements:
   a) $A1 = \{(p_a, p_r) | p_a \in P_A, p_r \in P_R, p_r^{\cdot} \cap {}^{\cdot}p_a \neq \emptyset\}$.
   b) $A2 = \{(p_r, p_a) | p_r \in P_R, p_a \in P_A, p_a^{\cdot} \cap p_r^{\cdot} \neq \emptyset\}$.
   c) $W = W_{A1} \cup W_{A2}$, where $W_{A1} : (p_r \times p_a) \rightarrow \mathbb{N}_n = \{1, 2, \ldots, n\}$ gives the weight of an arc (p_r, p_a), indicating the number of tokens granted from p_r to p_a, and $W_{A2} : (p_a \times p_r) \rightarrow \mathbb{N}_n = \{1, 2, \ldots, n\}$ gives the weight of an arc (p_a, p_r), indicating the number of tokens requested from p_a to p_r

Definition 2: Given an S^{4}R PN $(N, M_0) = (P_A \cup P^0 \cup P_R, T, F, W, M_0)$, its WRFG is defined as (V, A, W). A marking of WRFG is denoted by $M_G : V \rightarrow \mathbb{N} = \{0, 1, 2, \ldots\}$, where $M_G(V) = \sum_{p \in V} M(p)$ is the number of tokens in V, its initial marking is denoted by $M_{G0} = \sum_{p \in V} M_0(p)$, and the initial number of tokens in p at marking M_{G0} is given by $M_{G0}(p)$.

It is clear from Definition 1 that the WRFG illustrates the relationship between operation and resource places, where $W(p_a, p_r)$ represents the number of tokens that p_a should need from p_r for starting or completing a process, and $W(p_r, p_a)$ represents the number of tokens that p_r should grant to p_a for starting or completing a process. By analyzing the WRFGs of an S^{4}R Petri net, the token flows can be observed directly.

The procedure for computing a WRFG in an S^{4}R is given in Algorithm 1.

To illustrate this algorithm, we use the S^{4}R given in Fig. 1 to find the WRFG. In this net, $P_R = \{p_{12}, p_{13}, p_{14}, p_{15}\}$ and $P_A = \{p_1, p_2, p_3, p_4, p_5, p_6, p_8, p_9, p_{10}\}$.

Due to Algorithm 1, the WRFG for the S^{4}R in Fig. 1 is shown in Fig. 2, where $A1 = \{(p_{12}, p_1), (p_{12}, p_{10}), (p_{13}, p_2), (p_{13}, p_5), (p_{13}, p_9), (p_{14}, p_3), (p_{14}, p_6), (p_{14}, p_8),$

Algorithm 1 Construction of the WRFG in an S^{4}R

Input: An S^{4}R $(N, M_0) = (P_A \cup P^0 \cup P_R, T, F, W, M_0)$ for an FMS.

Output: A *WRFG* (V, A, W) from N.

```
1:  $V \leftarrow \emptyset, A2 \leftarrow \emptyset, A1 \leftarrow \emptyset, W_{A1} \leftarrow \emptyset, W_{A2} \leftarrow \emptyset$.
2:  for all $p_r \in P_R$ do
3:      for all $t \in p_r^{\cdot}$ do
4:          if there is a $p_a \in t^{\cdot} \cap P_A$ then
5:              $A1 \leftarrow \{(p_r, p_a)\} \cup A1$.
6:              $V \leftarrow \{p_r, p_a\} \cup V$.
7:              $W_{A1} \leftarrow (\{p_r\} \times T) \cup (T \times \{p_r\}) \leftarrow \mathbb{N}_n = \{1, 2, \ldots, n\}$.
8:          end if
9:      end for
10: end for
11: for all $p_a \in P_A$ do
12:     for all $t \in p_a^{\cdot}$ do
13:         if there exists a $p_r \in P_R \cap {}^{\cdot}t$ then
14:             $A2 \leftarrow \{(p_a, p_r)\} \cup A2$.
15:             $V \leftarrow \{p_r, p_a\} \cup V$.
16:             $W_{A2} \leftarrow (\{p_a\} \times T) \cup (T \times \{p_a\}) \leftarrow \mathbb{N}_n = \{1, 2, \ldots, n\}$.
17:         end if
18:     end for
19: end for
20: $A \leftarrow A1 \cup A2$.
21: $W \leftarrow W_{A1} \cup W_{A2}$.
22: output: $WRFG = (V, A, W)$.
23: End.
```

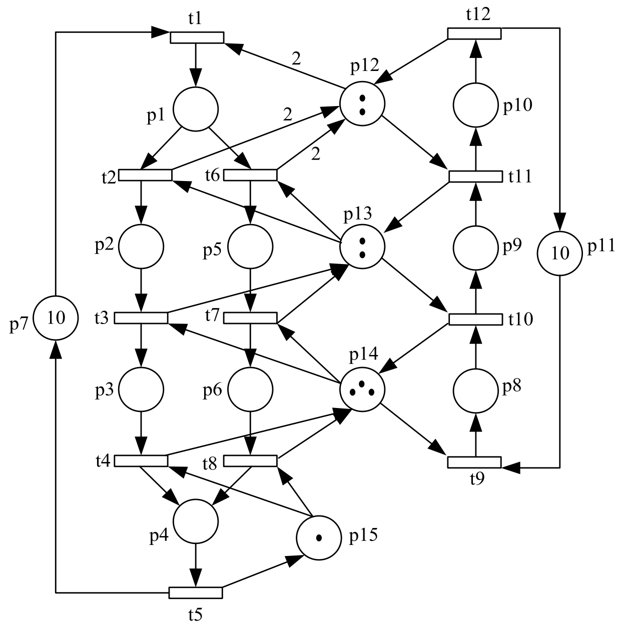

FIGURE 1. An S^{4}R (N, M_0).

$(p_{15}, p_4)\}$, and $A2 = \{(p_1, p_{13}), (p_2, p_{14}), (p_3, p_{15}), (p_5, p_{14}), (p_6, p_{15}), (p_8, p_{13}), (p_9, p_{12})\}$.

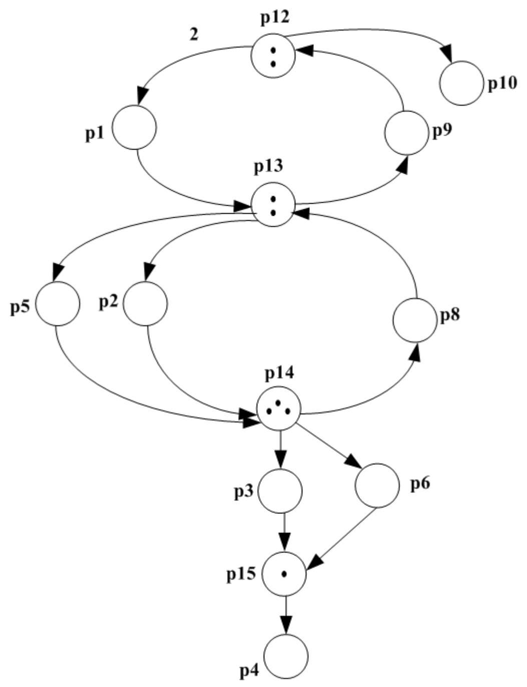

FIGURE 2. Weighted resource flow graph (WRFG) of an S^{4}R (N, M_0) in Fig. 1.

B. WEIGHTED CIRCULAR WAIT GRAPH (WCWG)

Definition 3: A WCWG is a circuit digraph $\boldsymbol{C_W} = (\boldsymbol{V'}, \boldsymbol{C}, \boldsymbol{W})$ derived from a WRFG $= (V, A, W)$ of an S^{4}R $(N, M_0) = (P_A \cup P^0 \cup P_R, T, F, W, M_0)$, where

1) $V' \subseteq V$ is a set of vertices such that $V' = \{p_{r1}, p_{a1}, p_{r2}, p_{a2}, \ldots, p_{rn}, p_{an}\}$ with $p_{ai} \in P_A$ and $p_{ri} \in P_R$
2) $C \subseteq V' \times V'$ is a set of directed arcs and the following conditions hold.
   a) $\forall i \in \{1, 2, \ldots, n\}$, there is a place pair $(p_{ri}, p_{ai}) \in C$.
   b) $\forall i \in \{1, 2, \ldots, n-1\}$, there is a place pair $(p_{ai}, p_{ri+1}) \in C$.
   c) there is a place pair $(p_{an}, p_{r1}) \in C$.
3) $W = W_{V'1} \cup W_{V'2}$, where
   a) $W_{V'1} : (p_r \times p_a) \rightarrow \mathbb{N}_n = \{1, 2, \ldots, n\}$ gives the weight of arcs $(p_r \times p_a)$, indicating the number of tokens granted from a resource place p_r to an operation place p_a.
   b) $W_{V'2} : (p_a \times p_r) \rightarrow \mathbb{N}_n = \{1, 2, \ldots, n\}$ gives the weight of arcs $(p_a \times p_r)$, indicating the number of tokens requested from an operation place p_a to a resource place p_r.
4) For $V' = \{p_{ri}, p_{ai}, p_{rj}, p_{aj}, p_{ri}\}$ with $p_{ai} \in P_A$ and $p_{ri} \in P_R$, $i \in \{1, 2, 3, \ldots, n\}$, $p_{ri} \neq p_{rj} (p_{ai} \neq p_{aj})$ if $i \neq j$, $(p_{ri}, p_{ai}) \in C$ is called an in-arc from p_{ri} to p_{ai}, and $W_{V'1}(p_{ri}, p_{ai}) = M_0(p_{ri})$ presents the number of tokens granted from p_{ri} to p_{ai}; $(p_{ai}, p_{rj}) \in C$ is called the out-arc from p_{ai} to p_{rj}, and $W_{V'2}(p_{ai}, p_{rj}) = M_0(p_{ri})$ presents the number of tokens requested from p_{ai} to p_{rj}.

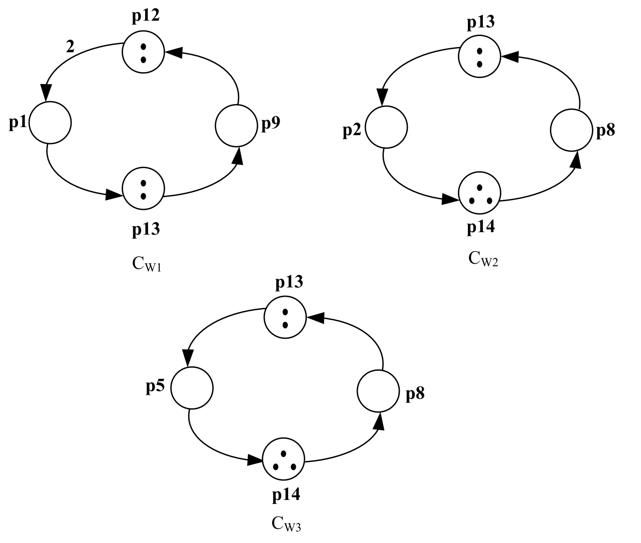

FIGURE 3. Weighted circular wait graph (WCWG) of (WRFG) in Fig. 2.

Let $C_1 = \{(p_r, p_a)\}$ and $C_2 = \{(p_a, p_r)\}$ be The sets of in-arcs and out-arcs of operation places in V', respectively. The set of WCWGs derived from a WRFG is denoted as $\mathcal{N}_{cw} = \{C_{w1}, C_{w2}, \ldots, C_{wi}\}$, where $C_{wi} = (V'_i, C_i, W_i), i \in \{1, 2, \ldots, n\}$. According to Definition 3, there are three weighted circular wait graphs in Fig. 2, $C_{w1} = (V'_1, C_1, W_1), C_{w2} = (V'_2, C_2, W_2)$, and $C_{w3} = (V'_3, C_3, W_3)$, where $C_{w1} = \{p_{12}, p_1, p_{13}, p_9\}, C_{w2} = \{p_{13}, p_2, p_{14}, p_8\}$, and $C_{w3} = \{p_{13}, p_5, p_{14}, p_8\}$. Hence, $\mathcal{N}_{cw} = \{C_{w1}, C_{w2}, C_{w3}\}$, and the three weighted circular wait graphs are shown in Fig. 3.

C. RELATIONSHIP BETWEEN WCWG AND PARTIAL DEADLOCKS

Definition 4: Given an S^{4}R $(N, M_0) = (P_A \cup P^0 \cup P_R, T, F, W, M_0)$, a marking M is defined as a partial deadlock if the following conditions hold:

1) $\nexists t \in T'$ such that $M[t\rangle$,
2) $\exists T' \neq \emptyset \wedge T' \subseteq T$, and
3) $\forall M' \in R(N, M), \nexists t \in T' \mid M'[t\rangle$.

Proposition 1: Any deadlock is also a partial deadlock.

Proof: From Definition 4, it is certain that M is a partial deadlock if any transition $T' \subseteq T$ is dead at M and those that can be reached from M. That is to say, no transition $t \in T'$ can be enabled at marking M and the markings that can be reached from M, i.e., it is at a deadlock state. Thus, with $T' = T$, a deadlock is a special partial deadlock. ■

In (N, M_0), when a partial deadlock marking M is reached, without external intervention, any transition in T' can never be enabled. To make a net live (deadlock-free), one needs to detect all partial deadlock markings and recover them.

Note that a WCWG $C_W = (V', C, W)$ contains two different types of arcs: in-arcs $\{(p_r, p_a)\}$ and out-arcs $\{(p_a, p_r)\}$. It follows from the evolution rules of S^{4}Rs, a transition $t \in {}^{\cdot}p_a$ should be enabled by tokens in p_r, i.e., there should be tokens in p_r for starting the operation modeled by p_a. Furthermore, to make such a transition t enabled, the number of tokens in p_r should be greater than or equal to the weight of the c. Thus, the relationship between p_r and p_a is presented by arc (p_r, p_a).

By an out-arc (p_a, p_r), the enabling of $t \in p_a^{\cdot}$ needs tokens in its input place p_r, i.e., there should be tokens in p_r for finishing the operation modeled by p_a. Furthermore, firing such a transition t, the number of tokens released from p_a to p_r is equal to the weight of (p_a, p_r). Thus, the relationship between p_a and p_r is described by arc (p_a, p_r).

Let $\mathcal{N}_{cw} = \{C_{w1}, C_{w2}, \ldots, C_{wn}\}$ be the set of WCWGs in a WRFG for an S^{4}R (N, M_0), where $C_{wi} = (V'_i, C_i, W_i)$, and $C_i = C_i^1 \cup C_i^2$ with C_i^1 and C_i^2 being the sets of in-arcs and out-arcs of operation places, respectively. Consider the place set in WCWGs denoted as $V'_* = \bigcup_i^n V'_i$, and the set of in-arcs (out-arcs) denoted as $C_*^1 = \bigcup_i^n C_i^1 (C_*^2 = \bigcup_i^n C_i^2)$. Given a resource place $p_r \in V'_*$, we define the output operation place set of p_r as $P_{r\to a} = \{p_a | (p_r, p_a) \in C_*^1\}$.

Definition 5: A marking $M_{cw}^d = \sum_{p \in V'} M_{cw}^d(p)p$ is said to be a deadlocked marking in a WCWG $C_w = (V', C, W)$, if for all resource places $p_r \in V'$, the following equations hold:

$$\sum_{p_a \in P_{r\to a}} M_{cw}^d(p_a) = M_{cw}^0(P_r) \tag{1}$$

$$M_{cw}^d(P_r) = 0 \tag{2}$$

$$M_{cw}^d(P) = \{1, 2, 3, \ldots\}, \quad p \in P_{r\to a} \cap V' \tag{3}$$

$$M_{cw}^d(P) = \{0, 1, 2, 3, \ldots\}, \quad p \in P_{r\to a} \setminus V' \tag{4}$$

where M_{cw}^0 is the initial marking of the WRFG.

The set of all deadlocked markings in a WCWG $C_w = (V', C, W)$ is denoted as $\mathcal{M}_{cw}^d$. Algorithm 2 presents the computation of the set of deadlocked markings of each WCWGs (C_w) in WRFG.

Consider an S^{4}R (N, M_0) and its WRFG presented in Figs. 1 and 2, respectively, where there are three WCWGs as shown in Fig. 3. According to Algorithm 2, there are three deadlocked markings: $M_{cw1}^d = p_1 + 2p_9 + 0p_{12} + 0p_{13}$ in $C_{w1} = \{p_{12}, p_1, p_{13}, p_9\}$, $M_{cw2}^d = p_2 + p_8 + 0p_{13} + 0p_{14}$ in $C_{w2} = \{p_{13}, p_2, p_{14}, p_8\}$, and $M_{cw3}^d = p_5 + p_8 + 0p_{13} + 0p_{14}$ in $C_{w3} = \{p_{13}, p_5, p_{14}, p_8\}$.

Theorem 1: Let $M_{cw}^d \in \mathcal{M}_{cw}^d$ be a deadlocked marking in a WCWGs $C_w = (V', C, W)$. Marking $M \in R(N, M_0)$ is a partial deadlock one if $M \in \mathcal{M}_{cw}^d$.

Proof: As seen, in a WCWG $C_w = (V', C, W)$, let $V' = \{p_{r1}, p_{a1}, p_{r2}, p_{a2}, \ldots, p_{rn}, p_{an}, p_{r1}\}$, $C = \{(p_{r1}, p_{a1}), (p_{a1}, p_{r2}), (p_{r2}, p_{a2}), \ldots, (p_{rn}, p_{an}), (p_{an}, p_{r1})\}$, and $W = W_{V'1} \cup W_{V'2}$, where $W_{V'1} : (p_r \times p_a) \to \mathbb{N}_n = \{1, 2, \ldots, n\}$, $W_{V'2} : (p_a \times p_r) \to \mathbb{N}_n = \{1, 2, \ldots, n\}$, $p_{ri} \in P_R$, and $p_{ai} \in P_A$, $i \in \{1, 2, \ldots, n-1\}$. According to Definition 5, a deadlocked marking M_{cw}^d of WCWG (C_w) is a state at which the tokens in p_{r1} are granted to p_{a1}, those in p_{r2} are granted to $p_{a2}, \ldots$, and those in p_{rn} are granted to p_{an}. In this state, no token is in any resource place in V', implying that the process presented by p_{ai} has started. Then, to finish these processes, p_{a1} requests the tokens in p_{r2} that is currently

Algorithm 2 Computation of deadlocked markings of WCWGs in a WRFG

```
Input: A $WRFG = (V, A, W)$ of an $S^4R$ $(N, M_0) = (P_A \cup P^0 \cup P_R, T, F, W, M_0)$.
Output: Deadlocked markings of WCWG $C_w = (V', C, W)$ in a WRFG.
1: Find all WCWGs $(C_w)$, denoted by $\mathcal{N}_{cw} = \{C_{w1}, C_{w2}, \ldots, C_{wn}\}$, where $C_{wi} = (V_i', C_i, W_i), i \in \{1, 2, \ldots, n\}$, according to Definition 3.
2: for all $C_{wi} \leftarrow (V_i', C_i, W_i) \in \mathcal{N}_{cw}$ do
3:     for all $p_r \in V_i' \cap P_R$ do
4:         Find the deadlocked markings that satisfy Eqs. (1), (2), (3), and 4.
5:         Let $\mathcal{M}_{cw}^d$ be the set of deadlocked markings of $C_{wi}$.
6:     end for
7: end for
8: output: The deadlocked markings of each WCWG $C_w = (V', C, W)$ in weighted resource flow graph WRFG, i.e., $M_{cw1}^d, M_{cw2}^d, \ldots, M_{cwn}^d$.
9: End.
```

held by p_{a2}. Similarly, for $i \in \{1, 2, \ldots, n-1\}$, p_{ai} requests the tokens in p_{ri+1} that is currently held by p_{ai+1}. Finally, p_{an} requests the tokens in p_{r1} that is currently held by p_{a1}. which leads to a deadlocked circular wait. As a result, a subnet N' of N formed by the places in V' is dead at M_{cw}^d. Let T' be the set of transitions in N'. No transitions in T' is enabled at M_{cw}^d. For all M in M_{cw}^d, no transition t in T' is enabled at M. By Definition 4, M is a partial deadlock. ■

To break a weighted circular wait, one needs to enforce the operation places at M_{cw}^d to let the tokens move back to the corresponding resource places. One way to do so is to construct recovery transitions such that they are enabled at partial deadlock markings and firing them can recover the system to live states. In this way, one needs only to consider a WCWGs $C_w = (V', C, W)$ to detect the deadlocked markings M_{cw}^d, and then design a recovery transition for them.

III. TRANSITION-BASED RECOVERY POLICY

An optimal transition-based recovery policy for each WCWG (C_w) is proposed in this section. Notice that, at a partial deadlock marking M_{cw}^d, a designed recovery transition should be enabled such that its firing can function as resource reallocation to release the tokens from the operation places back to the corresponding resource places. When the controller formed by the designed recovery transitions is applied, the controlled system becomes deadlock-free with all reachable markings.

Let (N, M_0) be an S^4R and (N_r, M_0) be its controlled system under the control of a recovery-transition-based controller. Let t_r denote a recovery transition for a WCWG C_w with the incidence vector $[N_r](P, t_r) = [x_1, x_2, x_3, \ldots, x_n]^T$, where n is the number of places. At M_{cw}^d in a WCWG C_w, in V', the tokens originally owned by the resource places are currently occupied by the operation places and no resource place holds a token. Hence, we design recovery transitions to return the tokens to resource places. Consider the following equations with t_r being a designed recovery transition:

$$[N_r](p_a, t_r) = -W(p_r, p_a)\ \forall p_a \in P_A \cap V',\quad p_r \in P_R \wedge p_{r\to a} \tag{5}$$

$$[N_r](p_r, t_r) = W(p_r, p_a)\ \forall p_r \in P_R \cap V',\quad p_a \in P_A \wedge p_{r\to a} \tag{6}$$

$$[N_r](p, t_r) = 0\ \forall p \in P \setminus (P^0 \cup V') \tag{7}$$

Eq. (5) means that firing t_r reduces the number of tokens in the operation places in V' and the reduced number is equal to the weights of their input arcs (the arcs are from their input resource places). Eq. (6) means that firing t_r increases the number of tokens in the resource places in V' and the increased number is equal to the weights of their output arcs (the arcs are from their output operation places). Eq. (7) indicates that firing t_r does not change the number of tokens in places that do not belong to $V' \cup P^0$.

To get the fully incidence vector of t_r, $x_i's$ for each place $p^{0i} \in P^0$ must be computed. Let I denote a P-vector for an S^4R. It is known that $I^T \cdot [N] = \mathbf{0}^T$ with positive integers for I ensures the conservativeness of a PN, which presents the necessary condition for structural boundedness and liveness of an S^4R. Therefore, one needs to guarantee conservativeness of the controlled system by designing a recovery-transition-based controller.

For an S^4R, we have $\| I_{p^{0i}} \| = \{p^{0i}\} \cup P_A$. We denote a minimal P-semiflow associated with p^{0i} as $I_{p^{0i}}^T = [I_{p^{0i}}^T(p^{0i}), I_{p^{0i}}^T(P^0 \setminus \{p^{0i}\}), I_{p^{0i}}^T(P_A), I_{p^{0i}}^T(P_R)]$, where $I_{p^{0i}}^T(P^0 \setminus \{p^{0i}\}) = 0$ and $I_{p^{0i}}^T(P_R) = 0$. To ensure the conservativeness of the controlled N_r, $I_{p^{0i}}^T[N_r](P, t_r) = 0$ should hold, i.e., $I_{p^{0i}}^T[N_r](\{p^{0i}\} \cup (P^0 \setminus \{p^{0i}\} \cup P_A \cup P_R, t_r) = 0$. As known, $I_{p^{0i}}^T(P^0 \setminus \{p^{0i}\}) = 0$ and $I_{p^{0i}}^T(P_R) = 0$, i.e., $I_{p^{0i}}^T(P_A).[N_r](P_A, t_r) + I_{p^{0i}}^T(p^{0i}).[N_r](p^{0i}, t_r) = 0$. resulting in $[N_r](p^{0i}, t_r) = -I_{p^{0i}}^T(P_A).[N_r](P_A, t_r)/I_{p^{0i}}^T(p^{0i})$. By (5) and (7), we already get $[N_r](p_a, t_r)$. Then, for any $p^{0i} \in P^0$, we have

$$x_i = -\sum_{p^{ai} \in P_A} I_{p^{0i}}(p^{ai}) x_{ai} / I_{p^{0i}}(p^{0i}),\ \forall p^{0i} \in P^0 \tag{8}$$

In this way, we can fully compute the incidence vector $[N_r](p, t_r)$ for t_r. Also, in a WRFG, the number of required recovery transitions is equal to the number of WCWGs. By doing so, it is certain that a behaviorally optimal recovery-transition-based controller is designed to make the controlled net deadlock-free with all reachable markings. Algorithm 3 presents the procedure to design the recovery-transition-based policy.

Theorem 2: The controlled system (N_r, M_0) obtained from Algorithm 3 is deadlock-free.

Algorithm 3 Computation of a controlled live system (N_r, M_0) by adding recovery transitions

```
Input: An S^4R $(N, M_0) = (P_A \cup P^0 \cup P_R, T, F, W, M_0)$ for an FMS.
Output: A controlled live system $(N_r, M_0)$.
1: Construct the $WRFG = (V, A, W)$ of $N$ by applying Algorithm 1.
2: Find all the WCWGs $(C_w) = (V', C, W)$ in WRFG denoted by $\mathcal{N}_{cw} = \{C_{w1}, C_{w2}, \dots, C_{wm}\}$, where $C_{wx} = (V'_x, C_x, W_x), x \in \{1, 2, \dots, n\}$.
3: Find the deadlocked markings in WCWGs: $\mathcal{M}^d_{cw1}, \mathcal{M}^d_{cw2}, \dots, \mathcal{M}^d_{cwn}$, by applying Algorithm 2.
4: $T_r \leftarrow \emptyset$. /* the set of designed recovery transitions */
5: for all $M^d_{cwi} \in \mathcal{M}^d_{cw}$ do
6:     Compute the incidence vector $[N_r](P, t_{ri})$ by Eqs. (5)–(8).
7:     $T_r \leftarrow T_r \cup \{t_{ri}\}$. /* add a recovery transition $t_{ri}$ of $M^d_{cwi}$. */
8: end for
9: Get the controlled system $(N_r, M_0)$ by applying the controller formed by $T_r$.
10: Output: $(N_r, M_0)$
11: End.
```

Proof: Assume that there exists a deadlock marking M in (N_r, M_0). Marking M must be a partial deadlock according to Proposition 1. It follows from Theorem 1 that a deadlocked marking $M^d_{cwi} \in \mathcal{M}^d_{cw}$ exists. In this case, by applying Algorithm 3, one can design a recovery transition t_{ri} such that any partial deadlocks in $\mathcal{M}^d_{cw}$ can be recovered. Therefore, (N_r, M_0) will not be deadlocked at M. ■

Now, consider the S^{4}R given in Fig. 1. For this net, there are 1280 reachable markings with six dead states. Its WRFG = (V, A, W) is presented in Fig. 2 with three WCWGs as shown in Fig. 3, $C_{w1} = (V'_1, C_1, W_1)$, $C_{w2} = (V'_2, C_2, W_2)$, and $C_{w3} = (V'_3, C_3, W_3)$, where $C_{w1} = \{p_{12}, p_1, p_{13}, p_9\}$, $C_{w2} = \{p_{13}, p_2, p_{14}, p_8\}$, and $C_{w3} = \{p_{13}, p_5, p_{14}, p_8\}$. In C_{w1}, the set of in-arcs is $C_1 = \{(p_{12}, p_1), (p_{13}, p_9)\}$ and the set of out-arcs is $C_2 = \{(p_1, p_{13}), (p_9, p_{12})\}$; in C_{w2}, the set of in-arcs is $C_1 = \{(p_{13}, p_2), (p_{14}, p_8)\}$ and the set of out-arcs is $C_2 = \{(p_2, p_{14}), (p_8, p_{13})\}$; in C_{w3}, the set of in-arcs is $C_1 = \{(p_{13}, p_5), (p_{14}, p_8)\}$ and the set of out-arcs is $C_2 = \{(p_2, p_{14}), (p_8, p_{13})\}$. $M^d_{cw1} = p_1 + 2p_9 + 0p_{12} + 0p_{13}$ is only one deadlocked marking in $C_{w1} = \{p_{12}, p_1, p_{13}, p_9\}$, $M^d_{cw2} = p_2 + p_8 + 0p_{13} + 0p_{14}$ is only one deadlocked marking in $C_{w2} = \{p_{13}, p_2, p_{14}, p_8\}$, and $M^d_{cw3} = p_5 + p_8 + 0p_{13} + 0p_{14}$ is only one deadlocked marking in $C_{w3} = \{p_{13}, p_5, p_{14}, p_8\}$.

The recovery transition t_{r1} of M^d_{cw1} is designed and we have $[N_r](P, t_{r1}) = [x_1, x_2, x_3, \dots, x_{15}]^T$. With Algorithm 3, we find x_{ri} $\forall p_i \in V'_1$, resulting in $[x_1, x_9, x_{12}, x_{13}]^T = [-1, -1, 2, 1]^T$. Also, for $p_i \in P \setminus (P^0 \cup V'_1)$, we get $x_i = 0$ such that $[x_2, x_3, x_4, x_5, x_6, x_8, x_{10}, x_{14}, x_{15}]^T = [0, 0, 0, 0, 0, 0, 0, 0, 0]^T$. Then, we compute x_{0i}'s $\forall p^{0i} \in P^0$ to get $[N_r](P^0, t_{r1}) = [x_7, x_{11}]^T$. There are associated with idle places p_7 and p_{11}; two place invariants I_7 and I_{11} are found as given below.

$$I_7 = [1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0]^T$$
$$I_{11} = [0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0]^T$$

Furthermore, we have $I_7(P_A) = [1, 1, 1, 1, 1, 1, 0, 0, 0]^T$, $[N_r](P_A, t_{r1}) = [-1, 0, 0, 0, 0, 0, 0, -1, 0]^T$, and $I_7(p_7) = 1$. Hence, $x_7 = -I_7^T(P_A).[N_r](P_A, t_{r1})/I_7(p_7) = 1$. Similarly, $x_{11} = -I_{11}^T(P_A).[N_r](P_A, t_{r1})/I_{11}(p_{11}) = 1$. The complete incidence vector of t_{r1} is found to be $[N_r](P, t_{r1}) = [-1, 0, 0, 0, 0, 0, 1, 0, -1, 0, 1, 2, 1, 0, 0]^T$.

The recovery transition t_{r2} of M^d_{cw2} is designed and we have $[N_r](P, t_{r2}) = [x_1, x_2, x_3, \dots, x_{15}]^T$. By applying Algorithm 3, we get x_{ri}, $\forall p_i \in V'_2$, to obtain $[x_2, x_8, x_{13}, x_{14}]^T = [-1, -1, 1, 1]^T$. Then, for $p_i \in P \setminus (P^0 \cup V'_2)$, we get $x_i = 0$, resulting in $[x_1, x_3, x_4, x_5, x_6, x_9, x_{10}, x_{12}, x_{15}]^T = [0, 0, 0, 0, 0, 0, 0, 0, 0]^T$. Next, we compute x_{0i}'s, $\forall p^{0i} \in P^0$, for $[N_r](P^0, t_{r2}) = [x_7, x_{11}]^T$. Associated with idle places p_7 and p_{11}, two place invariants I_7 and I_{11} are got as given below.

$$I_7 = [1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0]^T$$
$$I_{11} = [0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0]^T$$

Furthermore, we have $I_7(P_A) = [1, 1, 1, 1, 1, 1, 0, 0, 0]^T$, $[N_r](P_A, t_{r2}) = [0, -1, 0, 0, 0, 0, -1, 0, 0]^T$, and $I_7(p_7) = 1$. Hence, $x_7 = -I_7^T(P_A).[N_r](P_A, t_{r2})/I_7(p_7) = 1$. Similarly, $x_{11} = -I_{11}^T(P_A).[N_r](P_A, t_{r2})/I_{11}(p_{11}) = 1$. We then get the complete incidence vector of t_{r2} to be $[N_r](P, t_{r2}) = [0, -1, 0, 0, 0, 0, 1, -1, 0, 0, 1, 0, 1, 1, 0]^T$.

The recovery transition t_{r3} of M^d_{cw3} is designed and we have $[N_r](P, t_{r3}) = [x_1, x_2, x_3, \dots, x_{15}]^T$. By applying Algorithm 3, we compute x_{ri} $\forall p_i \in V'_3$ and find $[x_5, x_8, x_{13}, x_{14}]^T = [-1, -1, 1, 1]^T$. Then, for $p_i \in P \setminus (P^0 \cup V'_3)$, we get $x_i = 0$, resulting in $[x_1, x_2, x_3, x_4, x_6, x_9, x_{10}, x_{12}, x_{15}]^T = [0, 0, 0, 0, 0, 0, 0, 0, 0]^T$. Next, we compute x'_{0i}s $\forall p^{0i} \in P^0$ and get $[N_r](P^0, t_{r3}) = [x_7, x_{11}]^T$. Associated with idle places p_7 and p_{11}, we get two place invariants I_7 and I_{11} as given below.

$$I_7 = [1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0]^T$$
$$I_{11} = [0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0]^T$$

Furthermore, we get $I_7(P_A) = [1, 1, 1, 1, 1, 1, 0, 0, 0]^T$, $[N_r](P_A, t_{r3}) = [0, 0, 0, 0, -1, 0, -1, 0, 0]^T$, and $I_7(p_7) = 1$. Hence, $x_7 = -I_7^T(P_A).[N_r](P_A, t_{r3})/I_7(p_7) = 1$. Similarly, $x_{11} = -I_{11}^T(P_A).[N_r](P_A, t_{r3})/I_{11}(p_{11}) = 1$. Finally, we get the complete incidence vector of t_{r3} to be $[N_r](P, t_{r3}) = [0, 0, 0, 0, -1, 0, 1, -1, 0, 0, 1, 0, 1, 1, 0]^T$.

With the designed controller formed by the recovery transitions t_{r1}, t_{r2} and t_{r3}, the controlled system (N_r, M_0) is obtained as shown in Fig. 4. Its deadlock-freeness is checked with all 1280 reachable markings.

For computational complexity, in the worst case, it takes time $\mathcal{O}(n^2)$ and space $\mathcal{O}(n)$ for Algorithm 1 to find the WRFG of an S^{4}R, where n is the maximum number of

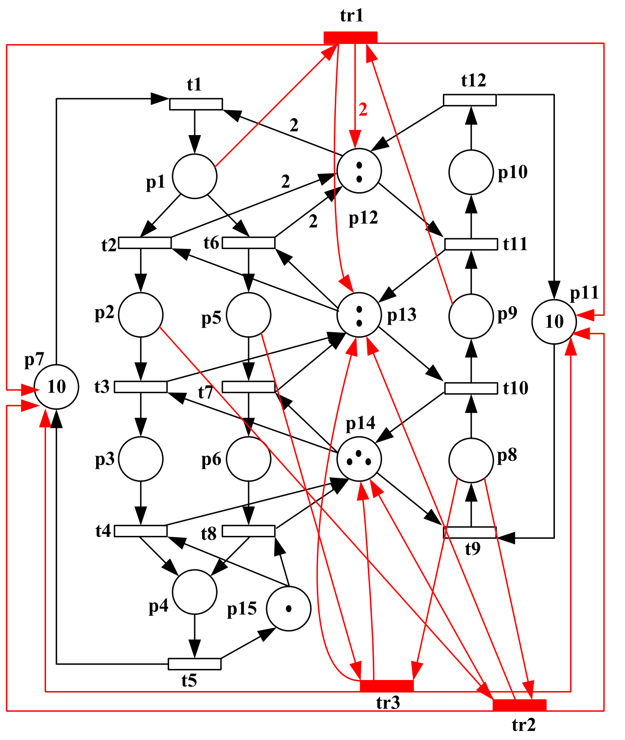

FIGURE 4. The Controlled System (N_r, M_0) With Three recovery transitions.

places and transitions in the model. The time to find the WCWGs (C_W) from the WRFG by Algorithm 2 is with $O((k+l)(m+1))$, where k, l, and m are the numbers of places in the WRFG, arcs, and WCWGs (C_W), respectively. Then, recovery transitions are decided via applying Algorithm 3. Consequently, the developed method is of polynomial complexity. Especially, with the proposed policy, one does not have to compute all the siphons or full reachability graphs, which is the real and main cause of high computational complexity.

IV. ILLUSTRATION WITH WIDELY APPLIED EXAMPLES

To demonstrate the proposed DDR policy, a few widely used FMS examples are presented in this section. An S^{4}R is given in Fig. 5 with $P_A = \{p_2 - p_6, p_8 - p_{11}\}$, $P_R = \{p_{12} - p_{16}\}$, and $P^0 = \{p_1, p_7\}$. For this net, there are 165 reachable markings with four dead states.

By Algorithm 1, the WRFG is obtained as shown in Fig. 6. It is clear that, for this WRFG, we have $A1 = \{(p_{12}, p_2), (p_{12}, p_{11}), (p_{13}, p_3), (p_{13}, p_{10}), (p_{14}, p_5), (p_{14}, p_9), (p_{15}, p_6), (p_{15}, p_8), (p_{16}, p_4)\}$, and $A2 = \{(p_2, p_{13}), (p_3, p_{14}), (p_3, p_{16}), (p_4, p_{15}), (p_5, p_{15}), (p_8, p_{14}), (p_9, p_{13}), (p_{10}, p_{12})\}$.

There are four WCWGs in the WRFG shown in Fig. 6, i.e., $\mathcal{N}_{cw} = \{C_{w1}, C_{w2}, C_{w3}, C_{w4}\}$. These WCWGs and the deadlocked markings are detailed in Tables 1 and 2, respectively. Associated with the two idle places p_1 and p_7, we get two place invariants I_1 and I_7, given as follows:

$$I_1 = [1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]^T$$
$$I_7 = [0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0]^T$$

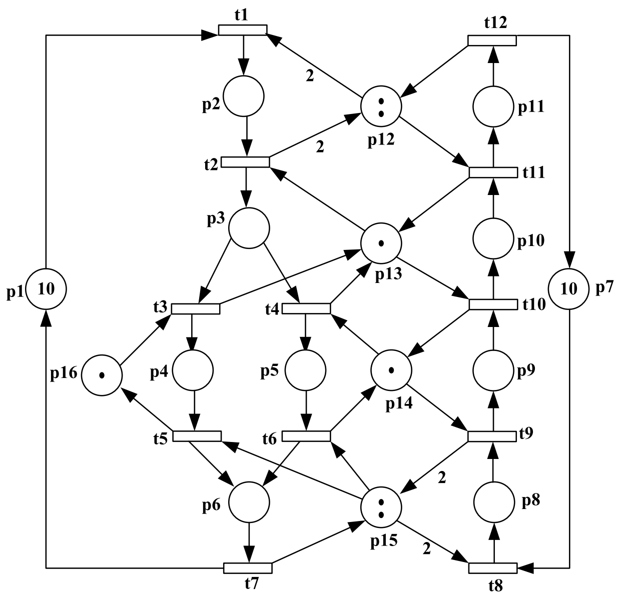

FIGURE 5. An S^{4}R (N_r, M_0).

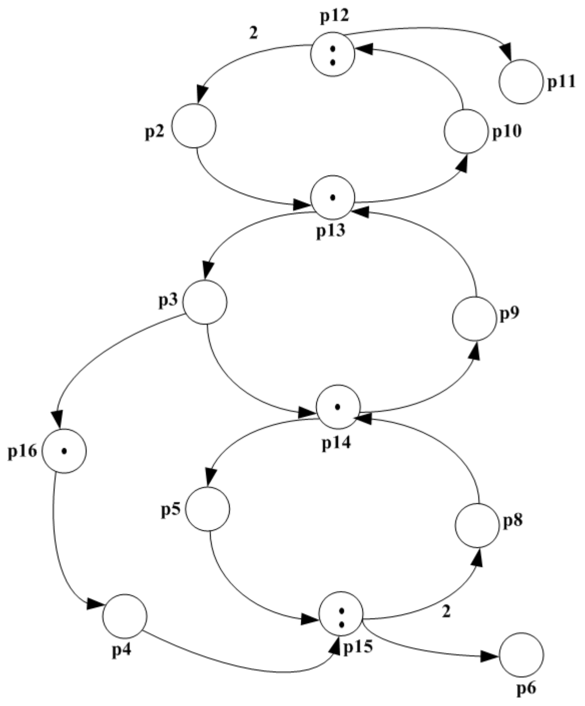

FIGURE 6. Weighted Resource Flow Graph (WRFG) of an S^{4}R (N, M_0) in Fig. 5.

By Algorithm 3, as given in Table 3, the incidence vectors of the recovery transitions are found. With the designed transitions t_{r1}, t_{r2}, t_{r3} and t_{r4}, the controlled system (N_r, M_0) is shown in Fig. 7. The deadlock-freeness is checked with all 165 reachable markings.

Consider an FMS with four machines M$_1$–M$_4$ and three robots R$_1$–R$_3$. Its layout is given in Fig. 8, while the part processing routes are given in Fig. 9. While both R$_1$ and

TABLE 1. Weighted circular wait graphs in the WRFG shown in Fig. 6.

$(C_i^W) = (V'_i, C_i, W_i)$	V'_i	C_i	W_i
$(C_1^W) = (V'_1, C_1, W_1)$	$V'_1 = \{p_{12}, p_2, p_{13}, p_{10}\}$	$C_1 = \{(p_{12}, p_2), (p_2, p_{13}), (p_{13}, p_{10}), (p_{10}, p_{12})\}$	$W_1(p_{12}, p_2) = 2$
$(C_2^W) = (V'_2, C_2, W_2)$	$V'_2 = \{p_{13}, p_3, p_{14}, p_9\}$	$C_2 = \{(p_{13}, p_3), (p_3, p_{14}), (p_{14}, p_9), (p_9, p_{13})\}$	
$(C_3^W) = (V'_3, C_3, W_3)$	$V'_3 = \{p_{14}, p_5, p_{15}, p_8\}$	$C_3 = \{(p_{14}, p_5), (p_5, p_{15}), (p_{15}, p_8), (p_8, p_{14})\}$	$W_2(p_{15}, p_8) = 2$
$(C_4^W) = (V'_4, C_4, W_4)$	$V'_4 = \{p_{13}, p_3, p_{16}, p_4, p_{15}, p_8, p_{14}, p_9\}$	$C_4 = \{(p_{13}, p_3), (p_3, p_{16}), (p_{16}, p_4), (p_4, p_{15}), (p_{15}, p_8), (p_8, p_{14}), (p_{14}, p_9), (p_9, p_{13})\}$	$W_3(p_{15}, p_8) = 2$

TABLE 2. Sets of deadlocked markings of weighted circular wait graphs listed in Table 1.

$(C_i^W) = (V'_i, C_i, W_i)$	$\mathcal{M}_{Cw}^d$
$(C_1^W) = (V'_1, C_1, W_1)$	$M_{Cw1}^d = \{p_2 + 2p_{10} + 0p_{12} + 0p_{13}\}$
$(C_2^W) = (V'_2, C_2, W_2)$	$M_{Cw2}^d = \{p_3 + p_9 + 0p_{13} + 0p_{14}\}$
$(C_3^W) = (V'_3, C_3, W_3)$	$M_{Cw3}^d = \{p_8 + 2p_5 + 0p_{14} + 0p_{15}\}$
$(C_4^W) = (V'_4, C_4, W_4)$	$M_{Cw4}^d = \{p_3 + p_4 + p_8 + p_9 + 0p_{13} + 0p_{14} + 0p_{15} + 0p_{16}\}$

TABLE 3. Incidence vectors of recovery transitions for deadlocked markings listed in Table 2.

C_i^W	$[N_r](P_A, t_{ri})$
C_1^W	$[N_r](P, t_{r1}) = [1, -1, 0 , 0, 0, 0, 1, 0, 0, -1, 0, 2, 1, 0, 0, 0]^T$
C_2^W	$[N_r](P, t_{r2}) = [1, 0, -1 , 0, 0, 0, 1,0 - 1, 0, 0, 0, 1, 1, 0, 0]^T$
C_3^W	$[N_r](P, t_{r3}) = [1, 0, 0 , 0, -1, 0, 1, -1, 0, 0, 0, 0, 0, 1, 2, 0]^T$
C_4^W	$[N_r](P, t_{r4}) = [2, 0, -1 , -1, 0, 0, 2, -1, -1, 0, 0, 0, 1, 1, 2, 1]^T$

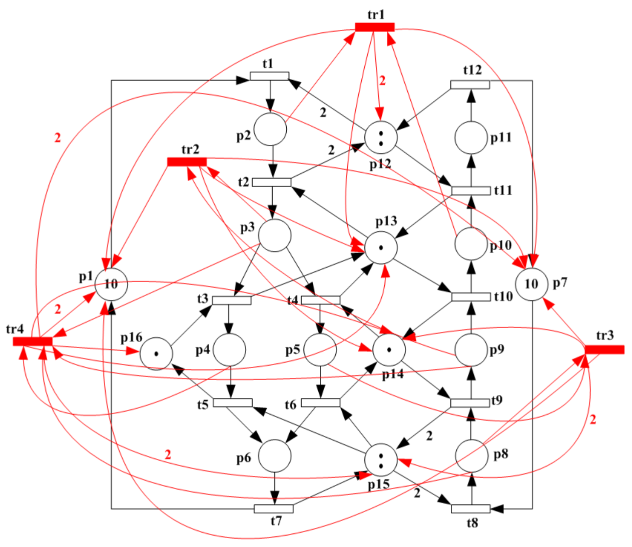

FIGURE 7. Petri net model (N_r, M_0) with four recovery transitions.

R_2 can hold two parts at a time, the capacity of R_3 is one. For the machines, while M_1 can process one product only at a time, two parts can be processed concurrently for any of M_2, M_3, and M_4. Three unloading buffers O_1–O_3 and three loading buffers I_1–I_3 are configured to hold completed and

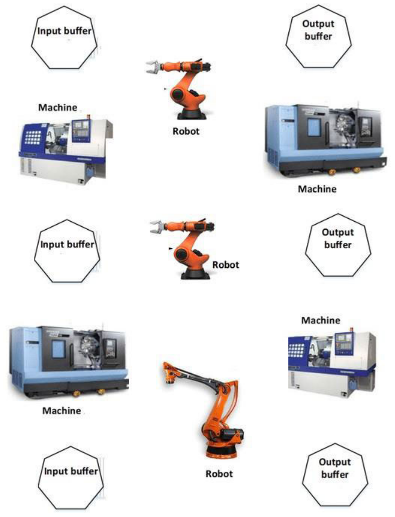

FIGURE 8. An FMS layout.

$$\begin{tikzcd}
\text{Part A: } I_1 \arrow[r] & R_2 \arrow[r] & M_2 \arrow[r] & R_2 \arrow[r] & O_1 & & \\
\text{Part B: } I_2 \arrow[r] & R_3 \arrow[r] & M_4 \arrow[r] & R_2 \arrow[r] & M_3 \arrow[r] & R_1 \arrow[r] & O_2 \\
& & M_1 \arrow[r] & R_2 \arrow[r] & M_2 \arrow[dr] & & \\
\text{Part C: } I_3 \arrow[r] & R_1 \arrow[ur] \arrow[dr] & & & & R_3 \arrow[r] & O_3 \\
& & M_3 \arrow[r] & R_2 \arrow[r] & M_4 \arrow[ur] & &
\end{tikzcd}$$

FIGURE 9. The production routings of an FMS layout.

raw parts. Three part types A, B, and C are to be processed with the production cycles given in Fig. 9.

Now, we consider each working process individually. With the cycles given in Fig. 9, R_2 picks a raw part A up from I_1 and loads to M_2. Then, after its completion by M_2, R_2 then unloads it to O_1.

A raw part B is picked up from I_2 by R_3 and loaded to M_4 for processing. After its completion, R_2 delivers it to M_3. After its completion at M_3, R_1 delivers it to O_2.

A raw part C is picked up from I_3 by R_1. Then, it can be loaded into M_1 and M_3 for processing. If it is processed by M_1, then after its completion, it is delivered to M_2 by R_2; otherwise, if it is processed by M_3, then after its completion, it is delivered to M_4 by R_2. Finally, after being processed by M_2 or M_4, R_3 delivers it to O_3.

An S^4R for this FMS can be established and shown in Fig. 10, where $P_A = \{p_2 - p_4, p_6 - p_{13}, p_{15} - p_{19}\}$,

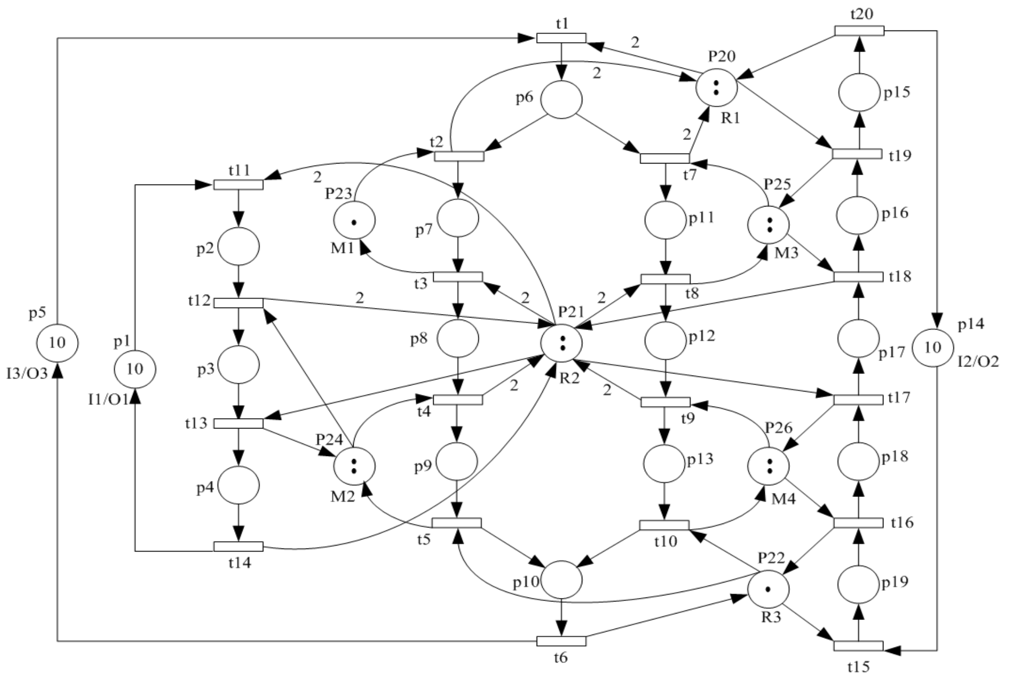

FIGURE 10. The S^4R that represents an FMS layout shown in Fig. 8.

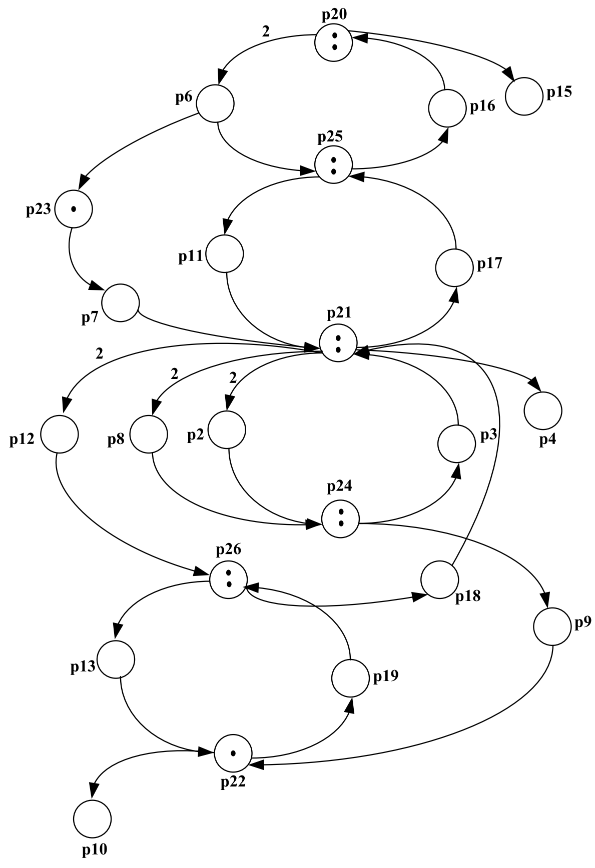

FIGURE 11. Weighted resource flow graph (WRFG) of an S^4R in Fig. 10.

$P_R = \{p_{20} - p_{26}\}$, and $P^0 = \{p_1, p_5, p_{14}\}$; the places $p_{20}, p_{21}, p_{22}, p_{23}, p_{24}, p_{25}$ and p_{26} model R$_1$, R$_2$, R$_3$, M$_1$, M$_2$, M$_3$, and M$_4$, respectively. There are 28051 reachable markings, among them, 120 are deadlock ones.

With Algorithm 1 being applied to this example, the WRFG is obtained, as shown in Fig. 11. In the WRFG, we have $A1 = \{(p_{20}, p_6), (p_{20}, p_{15}), (p_{21}, p_8), (p_{21}, p_4), (p_{21}, p_2), (p_{21}, p_{12}), (p_{21}, p_{17}), (p_{22}, p_{10}), (p_{22}, p_{19}), (p_{23}, p_7), (p_{24}, p_3), (p_{24}, p_9), (p_{25}, p_{11}), (p_{25}, p_{16}), (p_{25}, p_{13}), (p_{26}, p_{18})\}$, and $A2 = \{(p_2, p_{24}), (p_3, p_{21}), (p_6, p_{25}), (p_6, p_{23}), (p_7, p_{21}), (p_8, p_{24}), (p_9, p_{22}), (p_{11}, p_{21}), (p_{12}, p_{26}), (p_{13}, p_{22}), (p_{16}, p_{20}), (p_{17}, p_{25}), (p_{18}, p_{21}), (p_{19}, p_{26})\}$.

As shown in Fig. 11, in the WRFG, there are nine WCWGs, i.e., $\mathcal{N}_{cw} = \{C_{w1}, C_{w2}, C_{w3}, \ldots\ldots, C_{w9}\}$. Their details are presented in Table 4 and the deadlocked markings in $\mathcal{N}_{cw}$ are given in Table 5. There are associated with the three idle places p_1, p_5 and p_{14}; we have three place invariants I_1, I_5 and I_{14} as given below.

$$I_1 = [1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]^T$$
$$I_5 = [0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]^T$$
$$I_{14} = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0]^T$$

With Algorithm 3, the recovery transitions and their incidence vectors are obtained, as presented in Table 6. In this way, we get the recovery-transition-based controller and the resulting controlled system (N_r, M_0) by adding $\{t_{r1}, t_{r2}, \ldots\ldots t_{r9}\}$ as shown in Fig. 12. This controlled system is checked to be deadlock-free with all 28051 reachable markings.

TABLE 4. Weighted circular wait graphs in the WRFG shown in Fig. 11.

$(C_i^W) = (V'_i, C_i, W_i)$	V'_i	C_i	W_i
$(C_1^W) = (V'_1, C_1, W_1)$	$V'_1 = \{p_{20}, p_6, p_{25}, p_{16}\}$	$C_1 = \{(p_{20}, p_6), (p_6, p_{25}), (p_{25}, p_{16}), (p_{16}, p_{20})\}$	$W_1(p_{20}, p_6) = 2$
$(C_2^W) = (V'_2, C_2, W_2)$	$V'_2 = \{p_{25}, p_{11}, p_{21}, p_{17}\}$	$C_2 = \{(p_{25}, p_{11}), (p_{11}, p_{21}), (p_{21}, p_{17}), (p_{17}, p_{25})\}$	
$(C_3^W) = (V'_3, C_3, W_3)$	$V'_3 = \{p_{21}, p_2, p_{24}, p_3\}$	$C_3 = \{(p_{21}, p_2), (p_2, p_{24}), (p_{24}, p_3), (p_3, p_{21})\}$	$W_2(p_{21}, p_2) = 2$
$(C_4^W) = (V'_4, C_4, W_4)$	$V'_4 = \{p_{21}, p_8, p_{24}, p_3\}$	$C_4 = \{(p_{21}, p_8), (p_8, p_{24}), (p_{24}, p_3), (p_3, p_{21})\}$	$W_3(p_{21}, p_8) = 2$
$(C_5^W) = (V'_5, C_5, W_5)$	$V'_5 = \{p_{21}, p_{12}, p_{26}, p_{18}\}$	$C_5 = \{(p_{21}, p_{12}), (p_{12}, p_{26}), (p_{26}, p_{18}), (p_{18}, p_{21})\}$	$W_4(p_{21}, p_{12}) = 2$
$(C_6^W) = (V'_6, C_6, W_6)$	$V'_6 = \{p_{26}, p_{13}, p_{22}, p_{19}\}$	$C_6 = \{(p_{26}, p_{13}), (p_{13}, p_{22}), (p_{22}, p_{19}), (p_{19}, p_{26})\}$	
$(C_7^W) = (V'_7, C_7, W_7)$	$V'_7 = \{p_{24}, p_9, p_{22}, p_{19}, p_{26}, p_{18}, p_{21}, p_2\}$	$C_7 = \{(p_{24}, p_9), (p_9, p_{22}), (p_{22}, p_{19}), (p_{19}, p_{26}), (p_{26}, p_{18}), (p_{18}, p_{21}), (p_{21}, p_2)\}$	$W_5(p_{21}, p_2) = 2$
$(C_8^W) = (V'_8, C_8, W_8)$	$V'_8 = \{p_{24}, p_9, p_{22}, p_{19}, p_{26}, p_{18}, p_{21}, p_8\}$	$C_8 = \{(p_{24}, p_9), (p_9, p_{22}), (p_{22}, p_{19}), (p_{19}, p_{26}), (p_{26}, p_{18}), (p_{18}, p_{21}), (p_{21}, p_8)\}$	$W_6(p_{21}, p_8) = 2$
$(C_9^W) = (V'_9, C_9, W_9)$	$V'_9 = \{p_{20}, p_6, p_{23}, p_7, p_{21}, p_{17}, p_{25}, p_{16}\}$	$C_9 = \{(p_{20}, p_6), (p_6, p_{23}), (p_{23}, p_7), (p_7, p_{21}), (p_{21}, p_{17}), (p_{17}, p_{25}), (p_{25}, p_{16}), (p_{16}, p_{20})\}$	$W_7(p_{20}, p_6) = 2$

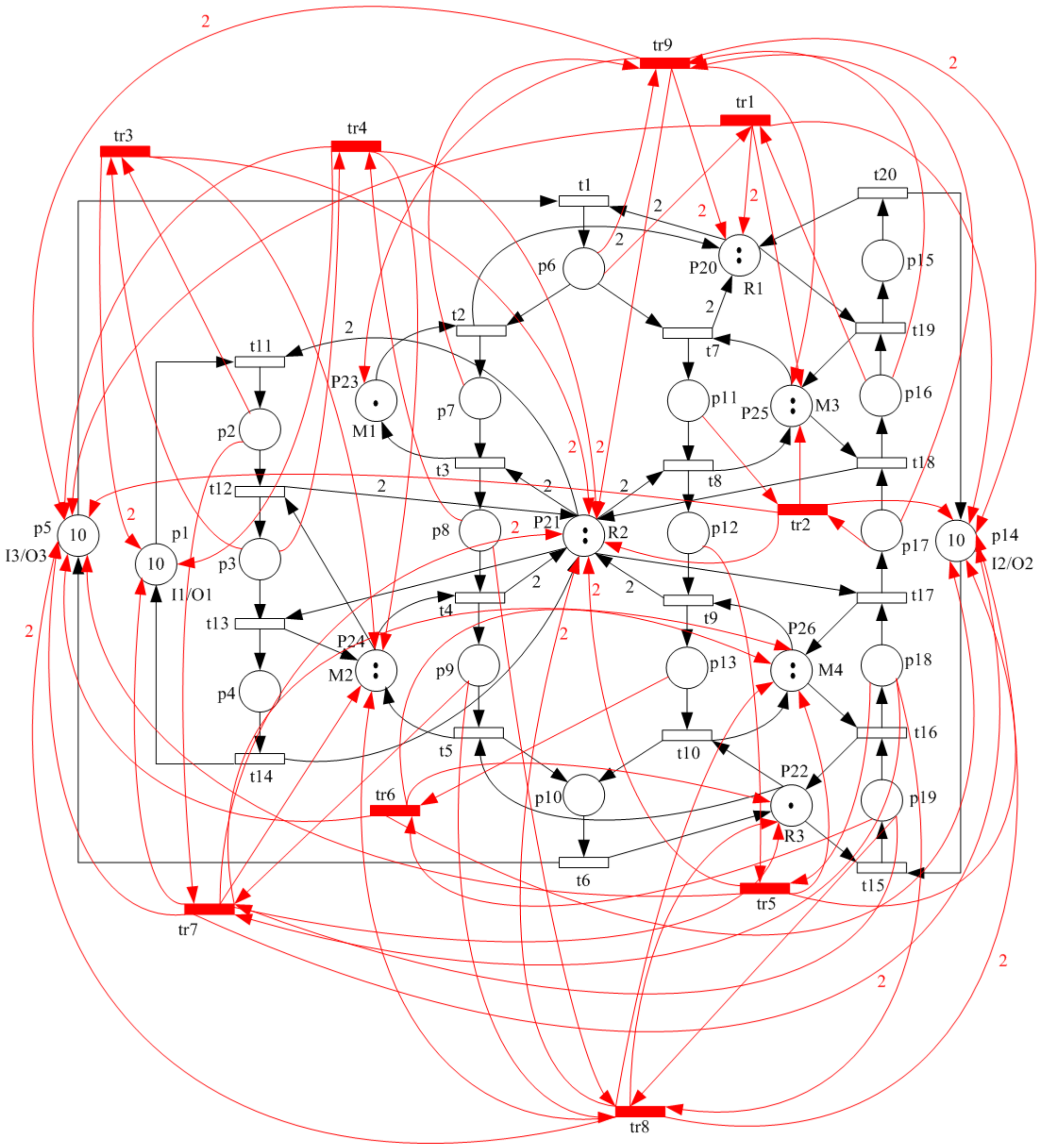

FIGURE 12. Petri net model (N_r, M_0) with nine recovery transitions.

TABLE 5. Sets of deadlocked markings of weighted circular wait graphs listed in Table 4.

$(C_i^W) = (V'_i, C_i, W_i)$	$\mathcal{M}_{Cw}^d$
$(C_1^W) = (V'_1, C_1, W_1)$	$M_{Cw1}^d = \{p_6 + 2p_{16} + 0p_{20} + 0p_{25}\}$
$(C_2^W) = (V'_2, C_2, W_2)$	$M_{Cw2}^d = \{p_{11} + p_{17} + 0p_{25} + 0p_{21}\}$
$(C_3^W) = (V'_3, C_3, W_3)$	$M_{Cw3}^d = \{p_2 + 2p_3 + 0p_{21} + 0p_{24}\}$
$(C_4^W) = (V'_4, C_4, W_4)$	$M_{Cw4}^d = \{p_8 + 2p_3 + 0p_{21} + 0p_{24}\}$
$(C_5^W) = (V'_5, C_5, W_5)$	$M_{Cw5}^d = \{p_{12} + 2p_{18} + 0p_{21} + 0p_{26}\}$
$(C_6^W) = (V'_6, C_6, W_6)$	$M_{Cw6}^d = \{p_{13} + p_{19} + 0p_{26} + 0p_{22}\}$
$(C_7^W) = (V'_7, C_7, W_7)$	$M_{Cw7}^d = \{p_9 + p_{19} + p_{18} + p_2 + 0p_{21} + 0p_{22} + 0p_{24} + 0p_{26}\}$
$(C_8^W) = (V'_8, C_8, W_8)$	$M_{Cw8}^d = \{p_9 + p_{19} + p_{18} + p_8 + 0p_{21} + 0p_{22} + 0p_{24} + 0p_{26}\}$
$(C_9^W) = (V'_9, C_9, W_9)$	$M_{Cw9}^d = \{p_6 + p_7 + p_{17} + 2p_{16} + 0p_{20} + 0p_{21} + 0p_{23} + 0p_{25}\}$

TABLE 6. Incidence vectors of recovery transitions for deadlocked markings listed in Table 5.

C_i^W	$[N_r](P_A, t_{ri})$
C_1^W	$[N_r](P, t_{r1}) = [0,0,0,0,1,-1,0,0,0,0,0,0,0,1,0,-1,0,0,0,2,0,0,0,0,1,0]^T$
C_2^W	$[N_r](P, t_{r2}) = [0,0,0,0,1,0,0,0,0,0,-1,0,0,1,0,0,-1,0,0,0,1,0,0,0,1,0]^T$
C_3^W	$[N_r](P, t_{r3}) = [2,-1,-1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,0,1,0,0]^T$
C_4^W	$[N_r](P, t_{r4}) = [1,0,-1,0,1,0,0,-1,0,0,0,0,0,0,0,0,0,0,0,0,2,0,0,1,0,0]^T$
C_5^W	$[N_r](P, t_{r5}) = [0,0,0,0,1,0,0,0,0,0,0,-1,0,1,0,0,0,-1,0,0,2,0,0,0,0,1]^T$
C_6^W	$[N_r](P, t_{r6}) = [0,0,0,0,1,0,0,0,0,0,0,0,-1,1,0,0,0,0,-1,0,0,1,0,0,0,1]^T$
C_7^W	$[N_r](P, t_{r7}) = [1,-1,0,0,1,0,0,0,-1,0,0,0,0,2,0,0,0,-1,-1,0,2,1,0,1,0,1]^T$
C_8^W	$[N_r](P, t_{r8}) = [0,0,0,0,2,0,0,-1,-1,0,0,0,0,2,0,0,0,-1,-1,0,2,1,0,1,0,1]^T$
C_9^W	$[N_r](P, t_{r9}) = [0,0,0,0,2,-1,-1,0,0,0,0,0,0,2,0,-1,-1,0,0,2,1,0,1,0,1,0]^T$

V. CONCLUSION

This paper extends the policy of deadlock detection and recovery for flexible manufacturing systems (FMSs) presented in [33]. Different from the study in [33], this work is done on a weighted class of Petri nets called S^{4}R that can model FMSs where a process stage may require multiple resource types and multiple units for a type to complete a task. Thus, the proposed method can be applied to more complex real-world FMSs and MURSs containing concurrently cyclic sequential processes, disassembly and assembly operations. We first design an algorithm to construct the WRFG of an S^{4}R to describe the competition for shared resources by various processes. Then, from the WRFG, we find the weighted circular wait graphs (WCWGs). In this way, the relationship between the WCWGs and partial deadlocks is revealed such that it can detect partial deadlocks. Then, for the detected deadlocked markings, an algorithm is derived to design recovery transitions. By analysis, it demonstrates that the resulting controlled S^{4}R net system is deadlock-free. The S^{4}R is validated and tested using the Integrated Net Analyzer (INA) simulator.

Overall, we conclude that the proposed policy has the following advantages: (1) it is applicable to any generalized classes of Petri nets such as ES^{3}PR, S*PR, S^{2}LSPR, S^{3}PGR^2, and S^{3}PMR; (2) it is computational more efficient than the existing ones since it does not require calculating reachability graphs, which suffers from the state explosion problem; (3) we design a recovery-transition-based controller to make the controlled S^{4}R deadlock-free, which provides an efficient way for the operation of FMSs; (4) the method can be applied to systems with sequential and complex resource requirements.

The follow-up study is, with affordable efforts, to optimize the supervisors' behavioral permissiveness, reduce structural complexity, and extend the reported method in this research to the systems with unreliable resources.

APPENDIX A BACKGROUND

A. DEADLOCK IN MULTI-UNIT RESOURCE SYSTEMS (MRS)

A resource allocation graph (RAG) [16] is a directed graph (V, E) with V and E being the sets of nodes and directed edges, respectively. Let $V = P \cup Q$ with $P \cap Q = \emptyset$, where P is a set of processes and Q is the set of resources. In this sense, an RAG is a bipartite graph. An edge $d_{ij} = (x_i, y_j)$ is a request edge if $x_i \in P$ and $y_j \in Q$. An edge $d_{ji} = (y_j, x_i)$ is a grant edge if $y_j \in Q$ and $x_i \in P$. Each edge in a path of a resource allocation graph $(x_{i1}, y_{j1}), (y_{j1}, x_{i2}), \ldots, (x_{ik}, y_{jk}), \ldots, (y_{js}, x_{is+1})$ is distinct.

The reachable set of a node x is defined as a set of nodes to which the node can reach. A knot is a non-empty set K of nodes satisfying the fact that the reachable set of each node in K is exactly K [49].

Resource requests and allocation in a production process can be represented by a weighted resource allocation graph (WRAG). Fig. 13(a) depicts System-on-a-Chip

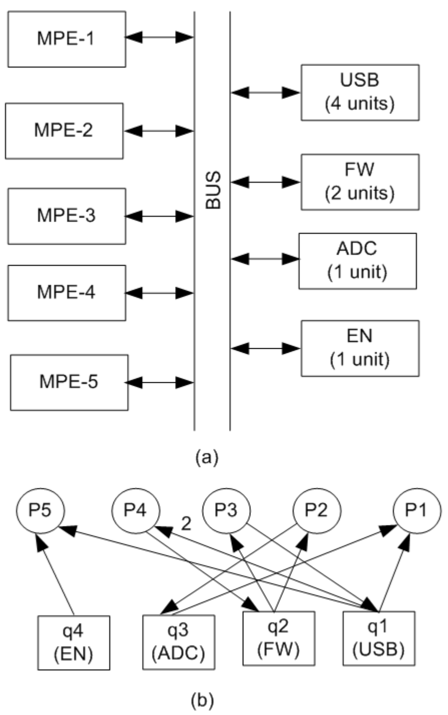

FIGURE 13. SoC example with its corresponding (WRAG).

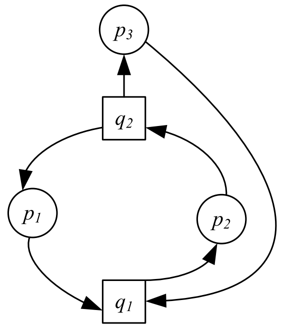

FIGURE 14. Resource allocation graph (RAG) with a knot.

(SoC) with five processors and four multi-unit resources. Fig. 13(b) visualizes a current resource allocation situation by WRAG. Fig. 14 illustrates an RAG with a knot $K = \{p_1, p_2, p_3, q_1, q_2\}$ that represents a circuit wait, where the following statements are assumed to be true: (1) the capacity of a resource is fixed, i.e., it has a fixed number of processing units, and (2) a resource unit is granted without any time delay if it is available.

B. BASICS OF PETRI NETS

The basics of Petri nets used in this paper can be found in [81], [82], [83], [84], [85], and [88].

Definition 6: A Petri net is a four-tuple $N = (P, T, F, W)$, where P and T are finite, non-empty, and disjoint sets; P is a set of places and T is a set of transitions. $F \subseteq (P \times T) \cup (T \times P)$ is its flow relation, represented by arcs with arrows from places to transitions or from transitions to places. $W : (P \times T) \cup (T \times P) \rightarrow \mathbb{N}$ is a mapping that assigns a weight to an arc: $W(x, y) > 0$ if $(x, y) \in F$, and $W(x, y) = 0$, otherwise, where $(x, y) \in (P \times T) \cup (T \times P)$ and $\mathbb{N}$ is a set of non-negative integers. N is ordinary, denoted as $N = (P, T, F)$, if for all $(x, y) \in F$, $W(x, y) = 1$.

Definition 7: A marking is a mapping $M : P \rightarrow \mathbb{N}$. Notation $M(p)$ denotes the number of tokens in p at M. Usually, a multi-set $\sum_{p \in P} M(p)p$ is used to describe M. A net N with an initial marking M_0 is called a net system, denoted as (N, M_0).

Definition 8: ${}^{\cdot}x = \{y \in P \cup T | (y, x) \in F\}$ is called the preset of x and $x^{\cdot} = \{y \in P \cup T | (x, y) \in F\}$ is called the postset of x. A net is pure (self-loop-free) if there do not exist $x, y \in P \cup T$ such that $(x, y) \in F$ and $(y, x) \in F$ hold.

Definition 9: A pure net $N = (P, T, F, W)$ can be represented by its incidence matrix $[N]$, a $|P| \times |T|$ integer matrix with $[N](p, t) = W(t, p) - W(p, t)$. A transition $t \in T$ is enabled at marking M if for all $p \in {}^{\cdot}t$, $M(p) \geq W(p, t)$. This fact is denoted by $M[t\rangle$. Once t fires, it yields a new marking M', denoted as $M[t\rangle M'$, where $M'(p) = M(p) - W(p, t) + W(t, p)$, for all $p \in P$. A marking M' is said to be reachable from M if there exist a sequence of transactions $\sigma = t_0 t_1 \ldots\ldots t_n$ and a sequence of markings $M_1, M_2, \ldots, M_n$ such that $M[t_0\rangle M_1[t_1\rangle M_2 \ldots . M_n[t_n\rangle M'$ holds. The set of markings reachable from M in N is denoted as $R(N, M)$.

Definition 10: A transition $t \in T$ is live at M_0 if for all $M \in R(N, M_0)$, there exists $M' \in R(N, M), M'[t\rangle$. (N, M_0) is live if for all $t \in T$, t is live at M_0. (N, M_0) is dead at M_0 if there does not exist $t \in T, M_0[t\rangle$.

Definition 11: A P-vector is a column vector $I : P \rightarrow \mathbb{Z}$ indexed by P and a T-vector is a column vector $J : T \rightarrow \mathbb{Z}$ indexed by T, where $\mathbb{Z}$ is the set of integers. A P-vector I is called a P-invariant (place invariant) if $I \neq \mathbf{0}$ and $I^T \bullet [N] = \mathbf{0}^T$. A T-vector J is called a T-invariant (transition invariant) if $J \neq \mathbf{0}$ and $[N] \bullet J = \mathbf{0}$. P-invariant I is a P-semiflow if every element of I is non-negative. $\parallel I \parallel = \{p | I(p) \neq 0\}$ is called the support of I. I is called a minimal P-invariant if $\parallel I \parallel$ is not a superset of the support of any other one and its components are mutually prime..

Definition 12: Let $N = (P, T, F)$ be a net. $X \subseteq P \cup T$ generates a subnet $N_X = (P_X, T_X, F_X)$, where $P_X = P \cap X$, $T_X = T \cap X$, and $F_X = F \cap (X \times X)$. A string $x_1 \ldots . x_n$ is called a path of N if for all $i \in \{1, 2, \ldots .., n-1\}$, $x_{i+1} \in x_i^{\cdot}$ for all $x \in \{x_1, \ldots\ldots, x_n\}$, $x \in P \cup T$. A simple path is a path whose nodes are all different (except, perhaps, x_1 and x_n). A path $x_1, \ldots x_n$ is called a circuit if it is a simple path and $x_1 = x_n$.

A Petri net N is called a state machine if for all $t \in T$, $|{}^{\cdot}t| = |t^{\cdot}| = 1$. For more basic knowledge and applications of Petri nets, readers are referred to [86], [50], [51], and [52].

C. SYSTEM OF SEQUENTIAL SYSTEMS WITH SHARED RESOURCES (S^4R)

This work considers a class of systems, namely a system of sequential systems with shared resources (S^4R) [34], [87]. Given a place p, we denote $max_{t \in p^{\cdot}}\{W(p, t)\}$ by $max_{p^{\cdot}}$.

Definition 13 [88]: A Simple Sequential Process (S^{2}P) is a Petri net $N = (P \cup \{p^0\}, T, F)$, where

1) $P \neq \emptyset$ ($p \in P$ is called an operation place), $p^0 \notin P$ (p^0 is called a process idle place);
2) N is a strongly connected state machine; and
3) Every circuit in N contains place p^0.

Definition 14 [88]: A Simple Sequential Process with Resources (S^{2}PR) is a net $N = (P \cup \{p^0\} \cup P_R, T, F)$ such that

1) The subnet generated by $(P \cup \{p^0\} \cup T)$ is an S^{2}P;
2) $P_R \neq \emptyset$ ($r \in P_R$ is called a resource place) and $(P \cup \{p^0\}) \cap P_R = \emptyset$;
3) $\forall p \in P, \forall t \in {}^{\cdot}p, \forall t' \in p^{\cdot}, \exists r_p \in P_R, {}^{\cdot}t \cap P_R = t'^{\cdot} \cap P_R = \{r_p\}$;
4) The two following statements are verified:
   a) $\forall r \in P_R, {}^{\cdot\cdot}r \cap P = r^{\cdot\cdot} \cap P \neq \emptyset$
   b) $\forall r \in P_R, {}^{\cdot}r \cap r^{\cdot} = \emptyset$; and
5) ${}^{\cdot\cdot}(p^0) \cap P_R = (p^0)^{\cdot\cdot} \cap P_R = \emptyset$; for $r \in P_R, H(r) = {}^{\cdot\cdot}r \cap P$, the operation places that use r are called the set of holders of r.

Definition 15 [88]: Let $N = (P \cup \{p^0\} \cup P_R, T, F)$ be an S^{2}PR. An initial marking M_0 is acceptable for N if $M_0(p^0) \geq 1, M_0(p) = 0$ for all $p \in P$, and $M_0(r) \geq 1$ for all $r \in P_R$. Recursively, a system S^{2}PR is called an S^{3}PR.

Definition 16 [88]: A System of S^{2}PR, S^{3}PR, is defined recursively as follows:

1) An S^{2}PR is an S^{3}PR;
2) Let $N_i = (P_i \cup P_i^0 \cup P_i^R, T_i, F_i), i \in \{1, 2\}$, be two S^{3}PRs such that $(P_1 \cup P_1^0) \cap (P_2 \cup P_2^0) = \emptyset, P_1^R \cap P_2^R = P_c \neq \emptyset$, and $T_1 \cap T_2 = \emptyset$ (in this case we say that N_1 and N_2 are composable); then, $N = (P \cup \{p^0\} \cup P_R, T, F)$ resulted from the composition of N_1 and N_2 via P_c (denoted as $N_1 \circ N_2$) is defined as: 1) $P = P_1 \cup P_2$, 2) $p^0 = p_1^0 \cup p_2^0$, 3) $P_R = P_1^R \cup P_2^R$, 4) $T = T_1 \cup T_2$, and $F = F_1 \cup F_2$, is also an S^{3}PR.

Definition 17 [88]: Let $N_i = (P_i \cup P_i^0 \cup P_i^R, T_i, F_i), i \in \{1, 2\}$, be an S^{3}PR. An initial marking M_0 is called an acceptable initial marking for N if one of the two following statements is true:

1) (N, M_0) is an acceptably marked S^{3}PR;
2) $N = N_1 \circ N_2$ such that (N_i, M_0^i) is an acceptably marked S^{3}PR and a) for all $i \in \{1, 2\}$, for all $p \in P_i \cup P_i^0, M_0(p) = M_0^i(p)$; b) for all $i \in \{1, 2\}$, for all $r \in P_i^R \setminus P_c, M_0(r) = M_0^i(r)$; c) for all $r \in P_c$, $M_0(r) = max\{M_0^1(r), M_0^2(r)\}$.

Definition 18 [88]: A Simple Sequential Process with Weighted Resources Allocation (WS^{2}PR), is a generalized Petri net $N = (P \cup \{p^0\} \cup P_R, T, F, W)$ such that

1) The subnet generated by $X = P \cup \{p^0\} \cup T$ is an S^{2}P;
2) $P_R \neq \emptyset$ and $(P \cup \{p^0\}) \cap P_R = \emptyset$;
3) $\forall p \in P, \forall t \in {}^{\cdot}p, \forall t' \in p^{\cdot}, \exists r_p \in P_R, {}^{\cdot}t \cap P_R = t'^{\cdot} \cap P_R = \{r_p\}$;
4) The two following statements are verified:
   a) $\forall r \in P_R, {}^{\cdot\cdot}r \cap P = r^{\cdot\cdot} \cap P \neq \emptyset$ and
   b) $\forall r \in P_R, {}^{\cdot}r \cap r^{\cdot} = \emptyset$;
5) ${}^{\cdot\cdot}(p^0) \cap P_R = (p^0)^{\cdot\cdot} \cap P_R = \emptyset$; and
6) The three following statements are verified:
   a) $W(p, t) = 1$ and $W(t, p) = 1, \forall p \in P \cap \{p^0\}, \forall t \in T$;
   b) $W(r, t) \geq 1$ and $W(t, r) \geq 1, \forall r \in P_R, \forall t \in T$; and
   c) Two arcs of any arc pair have the same weight.

Definition 19 [88]: Let $N = (P \cup \{p^0\} \cup P_R, T, F, W)$ be a WS^{2}PR. An initial marking M_0 is acceptable for N if $M_0(p^0) \geq 1, M_0(p) = 0$ for all $p \in P$, and $M_0(r) \geq W(r, t)$ for all $r \in P_R, t \in r^{\cdot}$.

Definition 20 [88]: A System of WS^{2}PR called WS^{3}PR for short, is defined recursively as follows:

1) An WS^{2}PR is a WS^{3}PR; and
2) Let $N_i = (P_i \cup P_i^0 \cup P_i^R, T_i, F_i), i \in \{1, 2\}$, be two WS^{3}PRs such that $(P_1 \cup P_1^0) \cap (P_2 \cup P_2^0) = \emptyset, P_1^R \cap P_2^R = P_c \neq \emptyset$, and $T_1 \cap T_2 = \emptyset$ (in which case we say that N_1 and N_2 are composable); then, $N = (P \cup \{p^0\} \cup P_R, T, F)$ due to the composition of N_1 and N_2 via P_c (denoted as $N_1 \circ N_2$) is defined as: 1) $P = P_1 \cup P_2$, 2) $p^0 = p_1^0 \cup p_2^0$, 3) $P_R = P_1^R \cup P_2^R$, 4) $T = T_1 \cup T_2$, and $F = F_1 \cup F_2$, is also an WS^{3}PR.

Definition 21 [34]: An S^{4}R is a marked Petri net $(N, M_0) = (P, T, F, W, M_0)$, such that

1) $P = P_A \cup P^0 \cup P_R$, where $P_A = \bigcup_{j=1}^{n} P_A^j$ is called the set of operation places such that $P_A^i \cap P_A^j = \emptyset$, for all $i \neq j$, $P^0 = \bigcup_{i=1}^{n} \{p_i^0\}$ is called the set of idle places with $P^0 \cap P_A = \emptyset$, and $P_R = \{r_1, r_2, \ldots, r_m\}$ is called the set of resource places such that $(P^0 \cup P_A) \cap P_R = \emptyset$;
2) $T = \bigcup_{j=1}^{n} T_j$, and for all $i \neq j, T_i \cap T_j = \emptyset$;
3) $W = W_A \cup W_R$, where $W_A : ((P_A \cup P^0) \times T) \cup (T \times (P_A \cup P^0)) \to \{0, 1\}$ such that for all $j \neq i, ((P_A^j \cup \{p_j^0\}) \times T_i) \cup (T_i \times (P_A^j \cup \{p_j^0\})) \to \{0\}$, and $W_R : (P_R \times T) \cup (T \times P_R) \to \mathbb{N}$;
4) $\forall j \in \mathbb{N}_n = \{1, 2, 3, \ldots, n\}$, the subnet N_j derived from $P_A^j \cup \{p_j^0\} \cup T_j$ is a strongly connected state machine such that every circuit contains p_j^0;
5) $\forall r \in P_R$, there exists a unique P-semiflow I_r such that $\| I_r \| \cap P_R = \{r\}, \| I_r \| \cap P^0 = \emptyset, \| I_r \| \cap P_A \neq \emptyset$, and $I_r(r) = 1$. Furthermore, it holds that $P_A = (\bigcup_{r \in P_R} \| I_r \|) \setminus P_R$;
6) N is pure and strongly connected;
7) $\forall p \in P_A, M_0(p) = 0; \forall r \in P_R, M_0(r) \geq max_{p \in \| I_r \|} I_r(p)$; and $\forall p_j^0 \in P^0, M_0(p_j^0) \geq 1$.

ACKNOWLEDGMENT

The authors gratefully acknowledge the approval and the support of this research study by the grant no. CSCR-2022-11-1781 from the Deanship of Scientific Research at Northern Border University, Arar, K.S.A.

REFERENCES

[1] J. Ezpeleta, J. M. Colom, and J. Martinez, "A Petri net based deadlock prevention policy for flexible manufacturing systems," *IEEE Trans. Robot. Autom.*, vol. 11, no. 2, pp. 173–184, Apr. 1995, doi: 10.1109/70.370500.

[2] G. Liu and K. Barkaoui, "Necessary and sufficient liveness condition of GS^{3}PR Petri nets," *Int. J. Syst. Sci.*, vol. 46, no. 7, pp. 1147–1160, 2015.

[3] D. Liu, Z. Li, and M. Zhou, "Hybrid liveness-enforcing policy for generalized Petri net models of flexible manufacturing systems," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 43, no. 1, pp. 85–97, Jan. 2013, doi: 10.1109/TSMCA.2012.2192266.

[4] L. Piroddi, R. Cordone, and I. Fumagalli, "Combined siphon and marking generation for deadlock prevention in Petri nets," *IEEE Trans. Syst. Man, Cybern. A, Syst. Humans*, vol. 39, no. 3, pp. 650–661, May 2009, doi: 10.1109/TSMCA.2009.2013189.

[5] Y. Chen, Z. Li, and M. Zhou, "Optimal supervisory control of flexible manufacturing systems by Petri nets: A set classification approach," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 2, pp. 549–563, Apr. 2014, doi: 10.1109/TASE.2013.2241762.

[6] K. Xing, M. C. Zhou, H. Liu, and F. Tian, "Optimal Petri-Net-based polynomial-complexity deadlock-avoidance policies for automated manufacturing systems," *IEEE Trans. Syst. Man, Cybern. A, Syst. Humans*, vol. 39, no. 1, pp. 188–199, Jan. 2009, doi: 10.1109/TSMCA.2008.2007947.

[7] H. Kaid, A. Al-Ahmari, A. M. El-Tamimi, E. A. Nasr, and Z. Li, "Design and implementation of deadlock control for automated manufacturing systems," *South Afr. J. Ind. Eng.*, vol. 30, no. 1, pp. 1–23, May 2019.

[8] 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, doi: 10.1109/ACCESS.2019.2936052.

[9] 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*, vol. 8, pp. 22411–22424, 2020, doi: 10.1109/ACCESS.2020.2968469.

[10] X. Zan, "A Pareto-based genetic algorithm for multi-objective scheduling of automated manufacturing systems," *Adv. Mech. Eng.*, vol. 12, no. 1, pp. 1–15, 2020, doi: 10.1177/1687814019885294.

[11] Y. Feng, K. Xing, M. Zhou, and H. Liu, "Liveness analysis and deadlock control for automated manufacturing systems with multiple resource requirements," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 50, no. 2, pp. 525–538, Feb. 2020, doi: 10.1109/TSMC.2017.2767902.

[12] Y. Wu, K. Xing, J. Luo, and Y. Feng, "Robust deadlock control for automated manufacturing systems with an unreliable resource," *Inf. Sci.*, vols. 346–347, pp. 17–28, Jun. 2016, doi: 10.1016/j.ins.2016.01.049.

[13] Y. Chen, Z. Li, K. Barkaoui, N. Wu, and M. Zhou, "Compact supervisory control of discrete event systems by Petri nets with data inhibitor arcs," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 47, no. 2, pp. 364–379, Feb. 2017, doi: 10.1109/TSMC.2016.2521833.

[14] S. Lee and D. M. Tilbury, "Deadlock-free resource allocation control for a reconfigurable manufacturing system with serial and parallel configuration," *IEEE Trans. Syst. Man Cybern., C Appl. Rev.*, vol. 37, no. 6, pp. 1373–1381, Nov. 2007, doi: 10.1109/TSMCC.2007.905843.

[15] S. Reveliotis and Z. Fei, "Robust deadlock avoidance for sequential resource allocation systems with resource outages," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 4, pp. 1695–1711, Oct. 2017.

[16] X. Xiao and J. J. Lee, "A parallel multi-unit resource deadlock detection algorithm with O(log$_2$(min(m,n))) overall run-time complexity," *J. Parallel Distrib. Comput.*, vol. 71, no. 7, pp. 938–954, Jul. 2011.

[17] E. G. Coffman, M. J. Elphick, and A. Shoshani, "System deadlocks," *ACM Comput. Surv.*, vol. 3, no. 2, pp. 67–78, Jun. 1971.

[18] Z. Li and M. Zhao, "On controllability of dependent siphons for deadlock prevention in generalized Petri nets," *IEEE Trans. Syst. Man, Cybern. A, Syst. Humans*, vol. 38, no. 2, pp. 369–384, Mar. 2008.

[19] Z. Li, G. Liu, H. Hanisch, and M. Zhou, "Deadlock prevention based on structure reuse of Petri net supervisors for flexible manufacturing systems," *IEEE Trans. Syst. Man, Cybern. A, Syst. Humans*, vol. 42, no. 1, pp. 178–191, Jan. 2012.

[20] G. J. Liu, C. J. Jiang, Z. H. Wu, and L. J. Chen, "A live subclass of Petri nets and their application in modeling flexible manufacturing systems," *Int. J. Adv. Manuf. Technol.*, vol. 41, nos. 1–2, pp. 66–74, Mar. 2009.

[21] L. Bai, N. Wu, Z. Li, and M. Zhou, "Optimal one-wafer cyclic scheduling and buffer space configuration for single-arm multicluster tools with linear topology," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 46, no. 10, pp. 1456–1467, Oct. 2016.

[22] X. Wang, I. Khemaissia, M. Khalgui, Z. Li, O. Mosbahi, and M. Zhou, "Dynamic low-power reconfiguration of real-time systems with periodic and probabilistic tasks," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 1, pp. 258–271, Jan. 2015.

[23] Wang, Xi, Z. Li, and W. M. Wonham, "Dynamic multiple-period reconfiguration of real-time scheduling based on timed DES supervisory control," *IEEE Trans. Ind. Informat.*, vol. 12, no. 1, pp. 101–111, Feb. 2016.

[24] J. Zhang, M. Khalgui, Z. Li, G. Frey, O. Mosbahi, and H. Ben Salah, "Reconfigurable coordination of distributed discrete event control systems," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 1, pp. 323–330, Jan. 2015.

[25] N. Q. Wu, F. Chu, C. B. Chu, and M. C. Zhou, "Petri net modeling and cycle time analysis of dual-arm cluster tools with wafer revisiting," *IEEE Trans. Syst. Man, Cybern., Syst.*, vol. 43, no. 1, pp. 196–207, Jan. 2013.

[26] N. Q. Wu, M. C. Zhou, and Z. W. Li, "Short-term scheduling of crude-oil operations: Petri net-based control-theoretic approach," *IEEE Robot. Autom. Mag.*, vol. 22, no. 2, 64–76, Jun. 2015.

[27] N. Wu, M. Zhu, L. Bai, and Z. Li, "Short-term scheduling of crude oil operations in refinery with high-fusion-point oil and two transportation pipelines," *Enterprise Inf. Syst.*, vol. 10, no. 6, pp. 581–610, Jul. 2016.

[28] S. Zhang, N. Wu, Z. Li, T. Qu, and C. Li, "Petri net-based approach to short-term scheduling of crude oil operations with less tank requirement," *Inf. Sci.*, vol. 417, pp. 247–261, Nov. 2017.

[29] Q. Zhu, Y. Qiao, and N. Wu, "Optimal integrated schedule of entire process of dual-blade multi-cluster tools from start-up to close-down," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 2, pp. 553–565, Mar. 2019.

[30] Q. Zhu, Y. Qiao, N. Wu, and Y. Hou, "Post-processing time-aware optimal scheduling of single robotic cluster tools," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 2, pp. 597–605, Mar. 2020.

[31] M. S. Elsayed, P. El Kafrawy, and N. Wu, "Modeling and deadlock control of reconfigurable multi-unit resource systems," *IEEE Access*, vol. 8, pp. 133605–133621, 2020, doi: 10.1109/ACCESS.2020.3010237.

[32] M. S. Elsayed, G. Liu, A. M. Mostafa, A. A. Alnuaim, and P. E. Kafrawy, "Fault-recovery and robust deadlock control of reconfigurable multi-unit resource allocation systems using siphons," *IEEE Access*, vol. 9, pp. 67942–67956, 2021, doi: 10.1109/ACCESS.2021.3073639.

[33] Y. Lu, Y. Chen, Z. Li, and N. Wu, "An efficient method of deadlock detection and recovery for flexible manufacturing systems by resource flow graphs," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 3, pp. 1707–1718, Jul. 2022, doi: 10.1109/TASE.2021.3114498.

[34] G. Liu, Z. Li, and C. Zhong, "New controllability condition for siphons in a class of generalised Petri nets," *IET Control Theory Appl.*, vol. 4, no. 5, pp. 854–864, May 2010, doi: 10.1049/iet-cta.2009.0264.

[35] X. Guan, W. Wu, and S. Wang, "New controllability condition for siphons in WS^{3}PR nets," *Asian J. Control*, vol. 17, no. 3, pp. 908–916, May 2015, doi: 10.1002/asjc.939.

[36] X. Guo, S. Wang, D. You, Z. Li, and X. Jiang, "A siphon-based deadlock prevention strategy for S^{3}PR," *IEEE Access*, vol. 7, pp. 86863–86873, 2019, doi: 10.1109/ACCESS.2019.2920677.

[37] M. P. Fanti and M. Zhou, "Deadlock control methods in automated manufacturing systems," *IEEE Trans. Syst. Man, Cybern. A, Syst. Humans*, vol. 34, no. 1, pp. 5–22, Jan. 2004, doi: 10.1109/TSMCA.2003.820590.

[38] M. P. Fanti and M. C. Zhou, "Deadlock control methods in automated manufacturing systems," in *Deadlock Resolution Computer-Integrated Systems*. New York, NY, USA: Marcel Dekker, 2005, pp. 1–22.

[39] P. J. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes," *SIAM J. Control Optim.*, vol. 25, no. 1, pp. 206–230, Jan. 1987, doi: 10.1137/0325013.

[40] P. J. G. Ramadge and W. M. Wonham, "The control of discrete event systems," *Proc. IEEE*, vol. 77, no. 1, pp. 81–98, Jan. 1989, doi: 10.1109/5.21072.

[41] M. A. Lawley, S. A. Reveliotis, and P. M. Ferreira, "A correct and scalable deadlock avoidance policy for flexible manufacturing systems," *IEEE Trans. Robot. Autom.*, vol. 14, no. 5, pp. 796–809, Oct. 1998, doi: 10.1109/70.720355.

[42] T. Murata, "Petri nets: Properties, analysis and application," *Proc. IEEE*, vol. 77, no. 4, pp. 541–580, Apr. 1989.

[43] R. Robidoux, H. Xu, L. Xing, and M. Zhou, "Automated modeling of dynamic reliability block diagrams using colored Petri nets," *IEEE Trans. Syst. Man, Cybern. A, Syst. Humans*, vol. 40, no. 2, pp. 337–351, Mar. 2010, doi: 10.1109/TSMCA.2009.2034837.

[44] L. C. Tsironis, D. S. Sfiris, and B. K. Papadopoulos, "Fuzzy performance evaluation of workflow stochastic Petri nets by means of block reduction," *IEEE Trans. Syst. Man, Cybern. A, Syst. Humans*, vol. 40, no. 2, pp. 352–362, Mar. 2010, doi: 10.1109/TSMCA.2009.2035303.

[45] N. Wu and M. Zhou, "Modeling and deadlock avoidance of automated manufacturing systems with multiple automated guided vehicles," *IEEE Trans. Syst. Man, Cybern., B Cybern.*, vol. 35, no. 6, pp. 1193–1202, Dec. 2005, doi: 10.1109/TSMCB.2005.850141.

[46] Z. Li, M. Zhou, and M. Jeng, "A maximally permissive deadlock prevention policy for FMS based on Petri net siphon control and the theory of regions," *IEEE Trans. Autom. Sci. Eng.*, vol. 5, no. 1, pp. 182–188, Jan. 2008, doi: 10.1109/TASE.2006.884674.

[47] G. Tian, H. Zhang, M. C. Zhou, and Z. W. Li, "AHP, gray correlation, and TOPSIS combined approach to green performance evaluation of design alternatives," *IEEE Trans. Syst. Man, Cybern., Syst.*, vol. 48, no. 7, pp. 1093–1105, Jul. 2018.

[48] M. B. Yildirim, T. Cakar, U. Doguc, and J. C. Meza, "Machine number, priority rule, and due date determination in flexible manufacturing systems using artificial neural networks," *Comput. Ind. Eng.*, vol. 50, nos. 1–2, pp. 185–194, May 2006.

[49] R. C. Holt, "Some deadlock properties of computer systems," *ACM Comput. Surv.*, vol. 4, no. 3, pp. 179–196, Sep. 1972.

[50] Y. Chen, Z. Li, K. Barkaoui, and A. Giua, "On the enforcement of a class of nonlinear constraints on Petri nets," *Automatica*, vol. 55, pp. 116–124, May 2015.

[51] N. Wu and M. Zhou, "Schedulability analysis and optimal scheduling of dual-arm cluster tools with residency time constraint and activity time variation," *IEEE Trans. Autom. Sci. Eng.*, vol. 9, no. 1, pp. 203–209, Jan. 2012, doi: 10.1109/TASE.2011.2160452.

[52] N. Qi Wu and M. Zhou, "Modeling, analysis and control of dual-arm cluster tools with residency time constraint and activity time variation based on Petri nets," *IEEE Trans. Autom. Sci. Eng.*, vol. 9, no. 2, pp. 446–454, Apr. 2012, doi: 10.1109/TASE.2011.2178023.

[53] P. H. Starke. (2003). *INA: Integrated Net Analyzer*. [Online]. Available: http://www2.informatik.hu-berlin.de/~starke/ina.html

[54] M. G. Mehrabi, A. G. Ulsoy, and Y. Koren, "Reconfigurable manufacturing systems: Key to future manufacturing," *J. Intell. Manuf.*, vol. 11, pp. 403–419, Aug. 2000, doi: 10.1023/A:1008930403506.

[55] R. Katz, "Design principles of reconfigurable machine," *Int. J. Adv. Manuf. Technol.*, vol. 34, pp. 430–439, Sep. 2007.

[56] R. Patel, A. Gojiya, and D. Deb, *Failure Reconfiguration of Pumps in Two Reservoirs Connected to Overhead Tank. Innovations in Infrastructure*. Berlin, Germany: Springer, 2019, pp. 81–92.

[57] Z. Ding, M. Zhou, and S. Wang, "Ordinary differential equation-based deadlock detection," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 44, no. 10, pp. 1435–1454, Oct. 2014, doi: 10.1109/TSMC.2014.2311757.

[58] J. Li, X. Dai, and Z. Meng, "Automatic reconfiguration of Petri net controllers for reconfigurable manufacturing systems with an improved net rewriting system-based approach," *IEEE Trans. Autom. Sci. Eng.*, vol. 6, no. 1, pp. 156–167, Jan. 2009, doi: 10.1109/TASE.2008.2006857.

[59] B. Tarnauca, D. Puiu, V. Comnac, and C. Suciu, "Modelling a flexible manufacturing system using reconfigurable finite capacity Petri nets," in *Proc. 13th Int. Conf. Optim. Electr. Electron. Equip. (OPTIM)*, May 2012, pp. 1079–1084, doi: 10.1109/OPTIM.2012.6231954.

[60] R. M. da Silva, I. F. Benítez-Pina, M. F. Blos, D. J. S. Filho, and P. E. Miyagi, "Modeling of reconfigurable distributed manufacturing control systems," *IFAC-PapersOnLine*, vol. 48, no. 3, pp. 1284–1289, 2015.

[61] X. Cong, C. Gu, M. Uzam, Y. Chen, A. M. Al-Ahmari, N. Wu, M. Zhou, and Z. Li, "Design of optimal Petri net supervisors for flexible manufacturing systems via weighted inhibitor arcs," *Asian J. Control*, vol. 20, no. 1, pp. 511–530, Jan. 2018.

[62] Z. Li and M. Zhou, "Clarifications on the definitions of elementary siphons in Petri nets," *IEEE Trans. Syst. Man, Cybern. A, Syst. Humans*, vol. 36, no. 6, pp. 1227–1229, Nov. 2006, doi: 10.1109/TSMCA.2006.878966.

[63] Z. Li and M. Zhou, "Two-stage method for synthesizing liveness-enforcing supervisors for flexible manufacturing systems using Petri nets," *IEEE Trans. Ind. Informat.*, vol. 2, no. 4, pp. 313–325, Nov. 2006, doi: 10.1109/TII.2006.885185.

[64] D. Y. Chao, "Direct minimal empty siphon computation using MIP," *Int. J. Adv. Manuf. Technol.*, vol. 45, nos. 3–4, pp. 397–405, Nov. 2009, doi: 10.1007/s00170-009-1967-1.

[65] D. Y. Chao, "Improved controllability test for dependent siphons in S^3PR based on elementary siphons," *Asian J. Control*, vol. 12, no. 3, pp. 377–391, May 2010.

[66] A. Ghaffari, N. Rezg, and X. Xie, "Design of a live and maximally permissive Petri net controller using the theory of regions," *IEEE Trans. Robot. Autom.*, vol. 19, no. 1, pp. 137–141, Feb. 2003, doi: 10.1109/TRA.2002.807555.

[67] M. Uzam and M. Zhou, "Iterative synthesis of Petri net based deadlock prevention policy for flexible manufacturing systems," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, Oct. 2004, pp. 4260–4265.

[68] M. Uzam and M. Zhou, "An iterative synthesis approach to Petri net-based deadlock prevention policy for flexible manufacturing systems," *IEEE Trans. Syst. Man, Cybern. A, Syst. Humans*, vol. 37, no. 3, pp. 362–371, May 2007, doi: 10.1109/TSMCA.2007.893484.

[69] Y.-S. Huang, T.-H. Chung, and P.-J. Su, "Synthesis of deadlock prevention policy using Petri nets reachability graph technique," *Asian J. Control*, vol. 12, no. 3, pp. 336–346, Feb. 2010, doi: 10.1002/asjc.188.

[70] Y. F. Hou, A. M. Al-Ahmari, and Z. W. Li, "Optimal controllability of 2-composed siphons in a class of Petri nets," *Electron. Lett.*, vol. 48, no. 24, pp. 1535–1537, Nov. 2012, doi: 10.1049/el.2012.2336.

[71] Y. F. Hou, Z. W. Li, L. Hong, and A. M. Al-Ahmari, "Optimal controllability of 3-composed siphons in a class of Petri nets," *Electron. Lett.*, vol. 49, no. 11, pp. 697–699, May 2013, doi: 10.1049/el.2012.4512.

[72] Y. Hou, Z. Li, and A. Al-Ahmari, "Extended elementary siphons and their application to liveness-enforcement of generalized Petri nets," *Assian J. Control*, vol. 16, no. 6, pp. 1–22, 2014, doi: 10.1002/ASJC.801.

[73] Y. Hou, Z. Li, M. Zhao, and D. Liu, "Extended elementary siphon-based deadlock prevention policy for a class of generalised Petri nets," *Int. J. Comput. Integr. Manuf.*, vol. 27, no. 1, pp. 85–102, Jan. 2014.

[74] Y.-S. Huang, Y.-L. Pan, and P.-J. Su, "Transition-based deadlock detection and recovery policy for FMSs using graph technique," *ACM Trans. Embedded Comput. Syst.*, vol. 12, no. 1, pp. 1–13, Jan. 2013, doi: 10.1145/2406336.2406347.

[75] Y. Chen, Z. Li, A. Al-Ahmari, N. Wu, and T. Qu, "Deadlock recovery for flexible manufacturing systems modeled with Petri nets," *Inf. Sci.*, vol. 381, pp. 290–303, Mar. 2017.

[76] M. Bashir, D. Liu, M. Uzam, N. Wu, A. Al-Ahmari, and Z. Li, "Optimal enforcement of liveness to flexible manufacturing systems modeled with Petri nets via transition-based controllers," *Adv. Mech. Eng.*, vol. 10, no. 1, Jan. 2018, Art. no. 168781401775070.

[77] T.-C. Row and Y.-L. Pan, "Maximally permissive deadlock prevention policies for flexible manufacturing systems using control transition," *Adv. Mech. Eng.*, vol. 10, no. 7, Jul. 2018, Art. no. 168781401878740.

[78] T.-C. Row, W.-M. Syu, Y.-L. Pan, and C.-C. Wang, "One novel and optimal deadlock recovery policy for flexible manufacturing systems using iterative control transitions strategy," *Math. Problems Eng.*, vol. 2019, pp. 1–12, Mar. 2019.

[79] Y. Dong, Y. Chen, S. Li, M. A. El-Meligy, and M. Sharaf, "An efficient deadlock recovery policy for flexible manufacturing systems modeled with Petri nets," *IEEE Access*, vol. 7, pp. 11785–11795, 2019, doi: 10.1109/ACCESS.2018.2889305.

[80] Y.-L. Pan, "One computational innovation transition-based recovery policy for flexible manufacturing systems using Petri nets," *Appl. Sci.*, vol. 10, no. 7, p. 2332, Mar. 2020.

[81] J. Luo, Z. Liu, and M. Zhou, "A Petri net based deadlock avoidance policy for flexible manufacturing systems with assembly operations and multiple resource acquisition," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3379–3387, Jun. 2019.

[82] J. Sha, Y. Du, and L. Qi, "A user requirement oriented web service discovery approach based on logic and threshold Petri net," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 6, pp. 1528–1542, Nov. 2019.

[83] J. Zhou, J. Wang, and J. Wang, "A simulation engine for stochastic timed Petri nets and application to emergency healthcare systems," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 4, pp. 969–980, Jul. 2019.

[84] F. Yang, N. Wu, Y. Qiao, and R. Su, "Polynomial approach to optimal one-wafer cyclic scheduling of treelike hybrid multi-cluster tools via Petri nets," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 1, pp. 270–280, Jan. 2018.

[85] M. C. Zhou and F. DiCesare, *Petri Net Synthesis for Discrete Event Control of Manufacturing Systems*. Norwell, MA, USA: Kluwer, 1993.

[86] M. C. Zhou and K. Venkatesh, *Modeling, Simulation and Control of Flexible Manufacturing Systems: A Petri Net Approach*, Singapore: World Scientific, 1998.

[87] K. Barkaoui and J.-F. Pradat-Peyre, "On liveness and controlled siphons in Petri nets," in *Proc. Int. Conf. Appl. Theory Petri Nets*, vol. 1091, 1996, pp. 57–72, doi: 10.1007/3-540-61363-3_4.

[88] Z. W. Li and M. C. Zhou, *Deadlock Resolution in Automated Manufacturing Syatems: A Novel Petri Net Approach*. London, U.K.: Springer, 2009.

[89] Z. Tian, X. Jiang, W. Liu, and Z. Li, "Dynamic energy-efficient scheduling of multi-variety and small batch flexible job-shop: A case study for the aerospace industry," *Comput. Ind. Eng.*, vol. 178, Apr. 2023, Art. no. 109111.

MAHMOUD SALAHELDIN ELSAYED received the B.S. degree in mathematics and computer sciences and the Ph.D. degree in computer sciences from the Faculty of Science, Menoufia University, Menoufia, Egypt.

From 2006 to 2008, he was with the Teachers College, Arar, Saudi Arabia. Since 2008, he has been a Lecturer of computer sciences with the Department of Computer Sciences, Faculty of Computing and Information Technology, Northern Border University, Rafha, Saudi Arabia. He has published many papers, including international journals and conferences ones, which all focus on deadlock control for single and multi-unit resource systems. He has published a book entitled *Analysis and Resolution in Distributed Systems* (Germany: LAP LAMBERT Academic Publishing, 2017). His research interests include Petri net theory and applications, supervisory control of discrete event systems, and robust supervisory control of flexible manufacturing systems.

Dr. Elsayed was a recipient of the Excellent Paper Award from the International Institute of Engineers and Researchers (IIER), in 2016.

KHALED KEFI was born in Tunisia, Tunis, in 1977. He received the B.S. and M.S. degrees from the University of Tunis El-Manar, in 2002, and the Ph.D. degree in pure mathematics, in 2007. From 2007 to 2010, he was a Research Assistant with the University of Sousse and the University of Tunis. From 2011 to 2019, he was an Associate Professor with the University of Tunis and Northern Border University. Since 2019, he was an Associate Professor with Northern Border University. He has nearly 40 research articles in prestigious journals. His research interests include the interplay between nonlinear functional analysis, mathematical physics, and the calculus of variations.

ZHIWU LI (Fellow, IEEE) received the B.S., M.S., and Ph.D. degrees from Xidian University, Xi'an, China, in 1989, 1992, and 1995, respectively. He was a Visiting Professor with the University of Toronto, Martin-Luther University at Halle (supported by Alexander von Humboldt Foundation), the University of Cagliari, Politecnico di Bari, Conservatoire National des Arts et Métiers (Cnam, supported by the program of Research in Paris), King Saud University, and Meliksah University. He is currently with the Institute of Systems Engineering, Macau University of Science and Technology. He has published two monographs in Springer (2009) and CRC Press (2013). His research interests include supervisory control of discrete event systems, Petri net theory and application, industrial automation, and production scheduling. His research was cited by leading business giants, such as IBM, HP, ABB, Volvo, GE, GM, Mitsubishi, and Huawei. He serves/served as an Associate Editor for IEEE Transactions on Automation Science and Engineering, IEEE Transactions on Systems, Man, and Cybernetics: Systems and Human Beings, IEEE Transactions on Systems, Man, and Cybernetics: Systems, and *Information Sciences* (Elsevier), and a Senior Editor for IEEE Access and *Scientific Reports*. He was selected as a Thomson Reuters Highly Cited Researchers in the category of engineering, from 2014 to 2018.