<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>Supervisor simplification for AMS based on petri nets and inequality analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author(s)</strong></td>
<td>Hu, Hesuan; Liu, Yang</td>
</tr>
<tr>
<td><strong>Date</strong></td>
<td>2014</td>
</tr>
<tr>
<td><strong>URL</strong></td>
<td><a href="http://hdl.handle.net/10220/18956">http://hdl.handle.net/10220/18956</a></td>
</tr>
<tr>
<td><strong>Rights</strong></td>
<td>© 2014 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. Published version of this article is available at [DOI: <a href="http://dx.doi.org/10.1109/TASE.2013.2288645">http://dx.doi.org/10.1109/TASE.2013.2288645</a>]</td>
</tr>
</tbody>
</table>
Supervisor Simplification for AMS Based on Petri Nets and Inequality Analysis

Hesuan Hu, Senior Member, IEEE, and Yang Liu

Abstract—In the framework of automated manufacturing systems (AMS), Petri nets are widely used to model, analyze, and control them. Resolving deadlocks is of paramount significance because their emergence may zero a system’s throughput. Supervisory control technique is the most widely adopted method to resolve them. A control policy can be converted into satisfying a set of inequalities, each of which corresponds to a siphon in a Petri net structure. The number of siphons can be exponential in the worst case, so does the number of inequalities. Taking into account the independent and dependent inequalities, this paper proposes a method to remove all the dependent inequalities while preserving only the independent ones. This method can significantly reduce the size of a supervisory controller. Examples are presented to illustrate the effectiveness and efficiency of this method.

Note To Practitioners—Owing to its importance in practice, deadlock arouses interest. Various resolution methods have been proposed by researchers and practitioners. The recent advance is to associate various sensors and actuators to certain predetermined working nodes for the sake of management and coordination of limited resources among concurrent operations. This uses a supervisory controller to a plant model. Nevertheless, existing approaches suffer from structural complexity due to their failure to reduce the sophisticated supervisor structures, which impedes their implementation in practice. Without a simplified control network, the communication cost among heterogeneous nodes increases while the responsive time decreases. Through the analysis on the dependency relationship among different inequalities, this paper presents a reduction technique to significantly simplify the supervisor structures. Without any extra investment, our approach betters a controlled system’s performance.

Index Terms—Liveness enforcing supervision, Petri nets, automated manufacturing systems, supervisor simplification, linear programming.

I. INTRODUCTION

Thanks to the strong desire to reduce manufacturing cost, improve product quality, and ensure workplace safety, automated manufacturing systems (AMS) are developed to automatically schedule manufacturing tasks, assign production resources, and handle mechanical work [1], [2], [8], [12], [13], [17]–[21], [25], [26], [31]. Sophisticated software is involved to diagnose the malfunction, inspect the product quality, and monitor a production line. In a fully automated facility, any human activity is excluded to minimize the cost. Its dynamics is featured by a series of activities driven by the asynchronous occurrences of discrete events. Differential and difference equations become inapplicable despite their wide application to time-driven systems [23], [24], [28], [38]. Petri nets prove to be one of the most powerful and popular mathematical tools to tackle such event-driven systems owing to their formalism and compactness [3], [4], [6], [7], [9]–[11], [14], [16], [22], [27], [29], [30], [32]–[35], [37].

To fulfill specifications, either by intrinsic default or by external imposition, feedback control technique must be developed for such systems [28], [38]. A plant model that induces undesirable behavior must be controlled [28], [38]. In order to change its original behavior, one must design a supervisor, thereby leading to a so-called supervisory control technique (SCT). In the spirit of SCT, a supervisor must constrain the behavior of systems into the legal or admissible domain by forbidding their illegal one [38]. As an elegant supervisor, a premise is the separation between the plant model from the supervisor. Some formal methods, such as automata, are of difficulty to achieve so owing to their language-based paradigm. Petri nets prove to be promising as they can represent both of them in a compact and split way [23], [24], [28], [38].

Unlike automata, Petri nets provide abundant structure information with regard to a modeled system. Through capturing such information, one can synthesize the supervisor in a linear-algebraic way. In [15], the concept of generalized mutual exclusion constraints (GMEC) is proposed to limit the weighted sum of tokens in a subset of places. Each GMEC corresponds to a linear supervisory specification which can be implemented by a control place (monitor, in short) and its related flows to some controllable transitions. In terms of SCT, each GMEC determines a set of forbidden states while each monitor specifies the control mechanism deciding which transition to fire and which one to forbid at each state. An optimal supervisor should avoid the occurrence of all these forbidden states without intervening other good ones.

In the cases of modeling an AMS with Petri nets, it is not surprising that the feature of liveness is critical because it guarantees that all the processes can be executed to their termination [1]–[5], [12], [13], [16], [17], [19]–[21], [24]–[26], [31], [33]. Siphons are a structural object in Petri nets to characterize these deadlock states. To avoid such states, siphons cannot be undermarked [2], [3], [12], [17], [25], [26]. Alternatively, they must be always sufficiently marked during the system evolution, which leads to a typical GMEC problem.
Some novel approach is presented based upon their properties through polynomial in its computational complexity.

Our approach makes key contributions in the following respects. First, it is applicable to any Petri net models where the GMEC technique can be used. For clarification, our experiments are conducted on some special classes of Petri nets; however, there is actually no limitation upon its applicability. Second, the behavior of the controlled system can be optimal if one removes only the redundant inequalities without influencing the others. Third, it requires only a procedure to solve a list of linear homogeneous systems, which proves to be polynomial in its computational complexity.

This paper is structured as follows. Section II reviews the basic definitions and notations of Petri nets used throughout this paper. Section III is devoted to a special class of Petri nets. Some novel approach is presented based upon their properties for the supervisory control purpose. In Section IV, with the aid of inequality analysis, our method is proposed to distinguish independent and dependent inequalities and show their use in simplifying a supervisor. Section V illustrates an example to verify the effectiveness of our approach. Section VI concludes the paper.

II. PRELIMINARIES

A Petri net is $N = (P, T, F, W)$ where $P$ is a set of places, $T$ is a set of transitions, $F \subseteq (P \times T) \cup (T \times P)$ is a set of directed arcs, and $W : (P \times T) \cup (T \times P) \rightarrow \mathbb{N} = \{0, 1, 2, \ldots\}$ such that $P \cup T \neq \emptyset$, $P \cap T = \emptyset$, and $W(x, y) = 0$ if $(x, y) \notin F$. A Petri net is said to be pure if $\forall x, y \in P \cup T: W(x, y) \neq 0 \Rightarrow W(y, x) = 0$. The preset of a node $x \in P \cup T$ is defined as $*x = \{y \in P \cup T \mid (x, y) \in F\}$. Its postset $x^* = \{y \in P \cup T \mid (y, x) \in F\}$. A net system with an initial marking $M_0$ is a state machine if $W : F \rightarrow \{1\}$ and $\forall t \in T, \|t\| = |t^*| = 1$. It is a marked graph if $W : F \rightarrow \{1\}$ and $\forall p \in P, \|p\| = |p^*| = 1$. Its incidence matrix $[N^-] = [W(p, t)]$ and output one $[N^+] = [W(t, p)]$. Its incidence matrix $[N] = [N^-] - [N^+]$. $[N_p]$ (resp., $[N_p^-], [N_p^+]$) is the $i$-th row of $[N]$ (resp., $[N^-], [N^+]$).

A marking of $N$ is a mapping $M : P \rightarrow \mathbb{N}$. $(N, M_0)$ is a net system with an initial marking $M_0$, $t$ is enabled at $M$, denoted by $M \{t\}$, if $\forall p \in *t, M(p) \geq W(p, t)$. $M$ is reachable from $M_0$, denoted by $M \{\sigma\} M'$, if there exists a firing sequence $\sigma = (t_1 t_2 \ldots t_n)$ such that $M \{t_1\} M_1 \ldots \{t_n\} M'$. $\sigma$ is a $|T|$-dimensional firing count vector where $\sigma$ states the number of $t$’s appearances in $\sigma$. The set of all markings reachable from $M_0$ is denoted by $R(N, M_0)$. $(N, M_0)$ is bounded if $\exists k \in \mathbb{N}^+ = \mathbb{N} \setminus \{0\}, \forall M \in R(N, M_0), \forall p \in P, M(p) \leq k$. $t$ is live under $M_0$ if $\forall M \in R(N, M_0), \exists M' \in R(N, M), M' \{t\}$ holds. $t$ is dead at $M \in R(N, M_0)$ if $\nexists M' \in R(N, M), \exists M' \{t\}$ holds. $(N, M_0)$ is deadlock-free if $\forall M \in R(N, M_0), \exists t \in T, M \{t\}$. It is

live if it is deadlock-free and $\exists t \in T$ such that $t$ is dead in $(N, M_0)$. A $P$-vector (resp., $T$-vector) is a column vector $I : P \rightarrow \mathbb{Z}$ indexed by $P$ (resp., $T$), where $\mathbb{Z}$ is the set of integers. A $P$-vector $I \neq 0$ becomes a $P$-invariant if $[N]^{-T} \cdot I = 0$, where $0$ means a vector of zeros. A $P$-invariant is called a $P$-semiflow if $I \geq 0$, $||I|| = \{p \in P \mid I(p) \neq 0\}$ is called the support of $I$. For economy of space, $\sum_{p \in P} M(p) \cdot p$ (resp., $\sum_{t \in T} J(t) \cdot t$) is used to denote vector $M$ (resp., $I$, $J$). $(N, M_0)$ is conservative (resp., consistent) if $\exists I \geq 0$ (resp., $\exists J > 0$) so that $I^T \cdot [N] = 0$ (resp., $[N] \cdot J^T = 0$). A circuit is an ordered set $(x_1, x_2, \ldots, x_n)$ such that: 1) $(x_1, x_2, \ldots, x_n) \subseteq P \cup T$; 2) $\forall i \in \mathbb{N}_{n-1} = \{1, 2, \ldots, n - 1\}$, $x_{i+1} \in x_i^*; 3) \forall i, j \in \mathbb{N}$ except $i, j = \{1, n\}, x_i \neq x_j; 4)$ $x_1 = x_n$.

A nonempty set $S \subseteq P$ (resp., $Q \subseteq P$) is a siphon (resp., trap) if $\exists S \subseteq S^*$ (resp., $Q^* \subseteq Q^*$. A strict minimal siphon is a siphon containing neither other siphon nor trap. $M(p)$ indicates the number of tokens in $p$ at $M$. $p$ is marked by $M$ if $M(p) > 0$. The sum of tokens in $S$ is denoted by $M(S)$, where $M(S) = \sum_{p \in S} M(p)$. A subset $S \subseteq P$ is marked by $M$ if $M(S) > 0$. A siphon is undermarked if $\exists t \in S^*$ can fire.

III. PETRI NET MODELING OF AMS

For better understanding, we focus throughout this paper on a special class of Petri nets, namely, System of Sequential Systems with Shared Resources ($S^4R$). Nevertheless, this does not necessarily mean the applicability limitation of our proposed method. In fact, it can be used in more general systems without an extension. In such kind of systems, various job types are modeled by state machines. The availability of various resources is modeled by resource places. Since there is no special limitation upon the resource quantity and types at each operation stage, $S^4R$ models a general resource allocation mechanism. Although job routes in $S^4R$ are constrained by a state machine, this does not mitigate its modeling capability because any net model can be converted to a state machine with its reachability graph. Fig. 1 shows the conversion process from a marked graph to a state machine, where the net in Fig. 1(a) denotes a marked graph while the net in Fig. 1(b) denotes a state machine. Compared with the latter, the former is assumed to be able to model operations like disassembly and assembly. Specifically, places of net in Fig. 1(b), i.e., $p_1, p_2, p_3, p_4$, and $p_5$, correspond to the markings of the net in Fig. 1(a), i.e., $M_0 = p_1, M_1 = p_2 + p_3, M_2 = p_3 + p_4, M_3 = p_2 + p_5$, and $M_4 = p_4 + p_5$. Specifically, an AMS can be partitioned into a set of resource types $\mathcal{R} = \{R_i, i = 1, 2, \ldots, L\}$ and a set of process types $\mathcal{J} = \{J_j, j = 1, 2, \ldots, K\}$. Every resource type $R_i$ is further characterized by

![Fig. 1. The conversion from a marked graph to a state machine.](image-url)
its capacity $C_i \in \mathbb{N}^+$. Processing requirements of process type $\mathcal{J}_f$ are defined by a set of concurrent and/or sequential stages. Each process stage, say $k$, modeled by a place $p_{jk}$ is associated with a conjunctive resource requirement, expressed by an $L$-dimensional vector $a_{jk}$ with $a_{jk}[i]$, $i \in \mathbb{N}_L = \{1, 2, \ldots, L\}$, indicating how many units of resource $R_i$ are required to support the execution of the stage denoted by $p_{jk}$. For systems whose each process can be converted to state machines, $S^4R$ can well handle them. Nevertheless, for more complex scenarios, such as free choice multiple reentrant flowlines, they are beyond $S^4R$’s description capability and deserve further investigation. Also, our method is not appropriate to tackle with systems exhibiting fast reconfiguration and unreliable resources.

A. $S^4R$ Models

Definition 1: An $S^4R$ is a strongly-connected generalized pure Petri net $N = (P, T, F, W)$ where: 1) $P = P_0 \cup P_A \cup P_R$ is a partition such that: a) $P_0$, $P_A$, and $P_R$ are called idle, operation (or activity), and resource places, respectively; b) $P_0 = \cup_{i \in \mathbb{N}_K} \{p_{0i}\}$; c) $P_A = \cup_{i \in \mathbb{N}_K} P_{Ai}$, where for each $i \in \mathbb{N}_K$, $P_{Ai} \neq \emptyset$, and for each $i, j \in \mathbb{N}_K$, $i \neq j$, $P_{Ai} \cap P_{Aj} = \emptyset$; and d) $P_R = \{r_1, r_2, \ldots, r_n\}$, $n > 0$. 2) $T = \cup_{i \in \mathbb{N}_K} T_i$, where for each $i \in \mathbb{N}_K$, $T_i \neq \emptyset$, and for each $i, j \in \mathbb{N}_K$, $i \neq j$, $T_i \cap T_j = \emptyset$. 3) For each $i \in \mathbb{N}_K$, subnet $N_i = N \setminus (\{p_{0i}\} \cup P_{Ai}, T_i)$ is a strongly connected state machine such that every cycle contains $p_{0i}$. 4) For each $r \in P_R$, $\exists$ a unique minimal $P$-semiflow $X_r \in \mathbb{N}^{|P|}$ such that $r = \|X_r\| \cap P_R$, $P_0 \cap \|X_r\| = \emptyset$, $P_A \cap \|X_r\| = \emptyset$, and $X_r(r) = 1$ where $\mathbb{N}^{|P|}$ means $|P|$-dimensional vectors whose each component belongs to $\mathbb{N}$. 5) $P_A = \cup_{r \in P_R} (\|X_r\| \setminus \{r\})$.

In $N$, $\forall p_{0i} \in P_0$, $M_0(p_{0i})$ indicates the upper bound of the maximum number of products that are allowed to be concurrently manufactured in a process initialized by $p_{0i}$. $\forall p \in P_A$, $M(p) > 0$ means ongoing operations modeled by $p$. $\forall r \in P_R$, $M_0(r)$ denotes the capacity of a resource $r$. From Definition 1, $S^4R$ is evidently conservative and consistent. $S^4R$ can model a set of concurrently-progressing types of parts. For products with the same type, they share the same processing route. The whole model is a composition of different processes through their sharing resources. In $S^4R$, each process exhibits routing flexibility as well as each stage allows multiple-resource acquisition. For more details, please refer to [31] and [33].

Definition 2: $M_0$ is an acceptable initial marking in $N$ if (1) $M_0(p_{0i}) \geq 1$, $\forall p_{0i} \in P_0$; (2) $M_0(p) = 0$, $\forall p \in P_A$; and (3) $M_0(r) \geq X_r(r)$, $\forall r \in P_R$, $\forall p \in P_A$.

Given an arbitrary marking $M \in R(N, M_0)$, a transition $t$ is process-enabled if $M(\cdot \cap P_A) > 0$. Note that $\|t \cap P_A\| = 1$ by definition. Correspondingly, $t$ is resource-enabled by $\forall r \in X_r \cap P_R$ if $M(r) \geq W(r, t)$. In the rest of this paper, $(N, M_0)$ is an acceptable $S^4R$.

Definition 3: Let $r \in P_R$ be a resource place in $(N, M_0)$. The set of holders of $r$ is the support of a minimal $P$-semiflow $X_r$ without $r$, i.e., $H(r) = \|X(r)\| \setminus \{r\}$. Clearly, $H(r)$ contains only operation places due to $\|X(r)\| \cap P_R = \{r\}$.

Let $S_R = S \cap P_R$ and $S_A = S \cap P_A$. Fig. 2. Two example $S^4R$ nets. (a) Net model where $x = z = 2$ and $y = 1$; (b) Net model where $x = z = 4$ and $y = 2$.

Definition 4: Let $S$ be a siphon that can be undermarked in $(N, M_0)$. Token takers, denoted by $\bar{H}(S)$, are the places that correspond to the holders of the resources in $S$ but do not belong to $S$.

Suppose $H_{SR} = \cup_{r \in S_R} H(r)$. We have $\bar{H}(S) = H_{SR} \setminus S = (H_{SR} \cap P_A) \setminus S_A$. The following condition for liveness of $S^4R$ is presented in [33]. $(N, M_0)$ is live if $\exists M \in R(N, M_0)$ and an undermarked siphon $S$ such that 1) $\forall r \in S_R$, $M(r) < W(r, t)$; 2) $\forall p \in S_A$, $M(p) = 0$; and 3) $\forall p \in H(S)$, $M(p) > 0$.

Thanks to their special structure, $S^4R$ can describe AMS in which each product is manufactured via sequential and/or concurrent manufacturing processes. It is composed of a set of subsets $N_i$, $i \in \mathbb{N}_K$, which are in one-to-one correspondence with a product and its related manufacturing processes. More specifically, each $N_i$ can be decomposed into an acyclic graph and an idle place $p_{0i}$. The operations together with their interactions required by a process are represented by the activity places and transitions involved in the respective acyclic graph of $N_i$. A set of activity places with a same ingoing transition correspond to the initialization of a number of concurrently executed processes, while the ones with a same outgoing transition correspond to an assembly operation. The initial marking of an idle place $p_{0i}$ corresponds to the number of products that are allowed in the system at a time. As a convention, $p_{0i}$ is also designated as the final destination of all finished process instances to model repetitive production. Places in $P_R$ are used to model various resource types. Their marking during the evolution of a Petri net corresponds to the number of available resources in the modeled AMS. In particular, their initial markings define the capacities of the corresponding resource types.

In the sequel, when mentioning $(N, M_0)$, we refer to an acceptably marked $S^4R$. Fig. 2 shows two typical $S^4R$ nets. Obviously, these two Petri nets share the same process nets whereas their resource allocation mechanisms are different. The former allows only single unit of resource acquisition at each operation stage whereas the latter allows multiple ones. Therefore, the latter is more general. Owing to its popularity,
the former is presented as a benchmark.

Under the assumption that \( P_0 = \{ p_1, p_7 \} \), \( P_{A_1} = \{ p_2 \} \), \( P_{A_2} = \{ p_9 - p_{10} \} \), \( P_R = \{ p_{11} - p_{13} \} \). Fig. 2 shows two \( S^R \) representing AMS consisting of five resource types \( R_1 - R_5 \) with capacities \( C_1 = C_3 = 1, C_2 = C_4 = C_5 = 1 \), and supporting two job types \( J_1 \) and \( J_2 \). Job type \( J_1 \) (resp., \( J_2 \)) is defined by the set of partially ordered job sets \( \{ p_7 - p_{10} \} \) (resp., \( \{ p_7 - p_{10} \} \)). The conjunctive resource requirements associated with various job sets in the net in Fig. 2(a) (resp., Fig. 2(b)) are as follows: \( a_{p_2} = 1 \), \( a_{p_2} = p_5 \), \( a_{p_4} = p_{11} \), \( a_{p_5} = p_{12} \), \( a_{p_6} = p_{13} \), \( a_{p_6} = p_{13} \), \( a_{p_6} = p_{12} \), and \( a_{p_{10}} = p_{11} \) (resp., \( a_{p_4} = p_{14} \), \( a_{p_5} = p_{15} \), \( a_{p_4} = 2p_{11} \), \( a_{p_5} = 2p_{11} + p_{12} \), \( a_{p_6} = p_{13} \), \( a_{p_6} = p_{13} \), \( a_{p_6} = 2p_{12} \), and \( a_{p_{10}} = 2p_{11} + 2p_{12} \). The \( S \)-semiflows corresponding to the resource areas: \( I_1 = p_4 + p_{10} + p_{11} \), \( I_2 = p_5 + p_9 + p_{12} \), \( I_3 = p_6 + p_8 + p_{13} \), \( I_4 = p_2 + p_{14} \), and \( I_5 = p_3 + p_{15} \) (resp., \( I_2 = 4p_{12} + 2p_{14} + p_{10} + p_{11} \), \( I_2 = 2p_{12} + 2p_{10} + p_{12} \), \( I_3 = p_6 + p_{13} + p_{14} \), \( I_4 = 2p_{12} + p_{14} \), and \( I_5 = p_3 + p_{15} \)).

B. Liveness-enforcing Supervisory Control Using GMEC

In Petri nets, many approaches have been proposed to forbid inadmissible states such that the remaining states satisfy a constraint \( t^P \cdot M \leq b \), where \( t^P \in \mathbb{N} \times |P| \), \( b \in \mathbb{N} \).

A control specification on a net system \((N, M_0)\) can be expressed in the form of \( t^P \cdot M \leq b \), where \( M \) means a reachable marking, while \( l \) and \( b \) are an integer vector and a scalar, respectively. To implement it, a monitor \( p_c \) should be superimposed on the net structure according to the following incidence matrix \([N_{P_c}] = -t^P \cdot [N] \). Correspondingly, the initial marking of \( p_c \) must be set as \( M_0(p_c) = b - t^P \cdot M_0 \). Obviously, the problem is feasible only if \( b - t^P \cdot M_0 > 0 \).

According to [2], the liveness of an \( S^R \) can be ensured when all of its siphons are properly controlled during the system evolution. Therefore, the basic idea to enforce its liveness is to prevent siphons from becoming undermarked by the addition of monitors. Hereby, we adopt max-control, which is a special SCT initially proposed in [2]. For the self-completeness of this paper, we cite the following definitions in [2]. In the sequel, when talking about a siphon, we mean a strict minimal one.

**Definition 5:** [2] A siphon \( S \) is said to be max-marked at \( M \in R(N, M_0) \) if \( \forall p \in S \) such that \( M(p) \geq \max_{p_c} \cdot (W(p, t)) \).

**Definition 6:** [2] A siphon \( S \) is said to be max-controlled if \( S \) is max-marked at any reachable marking, i.e., \( \forall M \in R(N, M_0), \forall S \in S \) such that \( M(p) \geq \max_{p_c} \cdot (W(p, t)) \).

**Definition 7:** [2] \((N, M_0)\) is said to be satisfying the max-controlled-siphon property (max cs-property, for short) if each siphon of \((N, M_0)\) is max-controlled.

**Proposition 1:** [2] A siphon \( S \) is a siphon. If there exists a \( P \)-invariant \( I \) such that \( \forall p \in [I]^+ \cap S, \max_{p_c} = 1, [I]^+ \subseteq S \), and \( \forall p \in P \), \( I(p) \cdot M_0(p) > \sum p \in S I(p) \cdot (\max_{p_c} - 1) \), then \( S \) is max-controlled.

**Proof:** Two siphons \( S \) and \( S' \) are on \( R(N, M_0), (N', M_0') \), \( N' = (P_0 \cup P_A \cup P_{R} \cup \{ p_c \}, T, F', W') \) is a max-extended net, and \( p \in P_0 \cup P_A \cup P_R, M_0'(p) = M_0(p) \). If \( S \) is not stable, then there exists a \( P \)-invariant \( I \) such that \( \forall p \in [I]^+ \cap S, \max_{p_c} = 1, [I]^+ \subseteq S \), and \( \forall p \in P \), \( I(p) \cdot M_0(p) > \sum p \in S I(p) \cdot (\max_{p_c} - 1) \), then \( S \) is max-controlled.
Evidently, a siphon can be prevented from being under-marked by simply applying a specification \( I^T \cdot M \leq M_0(S_R) - \sum_{p \in S} I(p) \cdot (\text{max} p, \cdot - 1) - 1 \) to the net. For the sake of brevity, we assume a scalar \( b = M_0(S_R) - \sum_{p \in S} I(p) \cdot (\text{max} p, \cdot - 1) - 1 \) in the sequel.

Based on Corollary 1, each siphon can be \( \text{max} \)-controlled through a GMEC denoted by a pair \((l, b)\). The latter can be easily identified through the structural analysis of an \( S^4R \) system becomes live when all siphons are \( \text{max} \)-controlled with \((l, b)\). For more detail, please refer to [2], [31], [33]. In the sequel, we assume all siphons are produced with an enumeration method.

Take the net shown in Fig. 2(a) as an example. \( S = \{p_{5}, p_{10} - p_{12}\} \) is a siphon. Evidently, we have \( g_S = p_4 + p_5 + p_9 + p_{10} + p_{11} + p_{12} \), \( h_S = p_4 + p_9 + p_e \), and \( I_S = g_S - h_S = p_5 + p_{10} + p_{11} + p_{12} - p_e \). As a result, we have \( \vartheta_S = p_5 + p_{10} + p_{11} + p_{12} \) and \( l_S = p_4 + p_9 + p_8 \). \( M_0(S_R) = M_0(p_{11}) + M_0(p_{12}) = 2 + 1 = 3 \).

\( \sum_{p \in S} I(p) \cdot (\text{max} p, \cdot - 1) = (\text{max} p_5, \cdot - 1) + (\text{max} p_9, \cdot - 1) + (\text{max} p_{10}, \cdot - 1) + (\text{max} p_{11}, \cdot - 1) + (\text{max} p_{12}, \cdot - 1) = (1 - 1) + (1 - 1) + (1 - 1) + (1 - 1) = 0 \).

Therefore, \( b = M_0(S_R) - \sum_{p \in S} I(p) \cdot (\text{max} p, \cdot - 1) = 3 - 0 - 1 = 2 \).

A pair \((l, b)\) is obtained, a monitor \( p_e \) along with its outgoing and ingoing arcs can be calculated with the technique in [28]. Tables I and II show the corresponding \( S \), \( g_S \), \( h_S \), \( I_S \), \( \vartheta \), \( l \), and \( b \) in the Petri nets shown in Figs. 2(a) and (b), respectively.

### IV. Supervisor Simplification via AMS Analysis

The above analysis shows that the liveness supervision in the framework of Petri nets can be converted to the satisfaction of a set of inequalities. These inequalities constitute a linear system that restricts the behavior of the plant model. However, one must notice that such a set of inequalities is not minimal. In other words, some inequalities may be redundant, implying that some inequalities are dependent on the others. As one inequality corresponds to one monitor, the removal of those dependent ones can reduce the size of the final supervisor. Therefore, it is attractive to identify a way to remove them while preserving the independent ones. The separation between them might not be unique. Some selection techniques should be developed to properly make their identification and distinction.

#### A. Identification of Independent and Dependent Inequalities

Suppose \( L = [l_1 \ l_2 \ldots \ l_n] \) and \( B = [b_1 \ b_2 \ldots b_n]^T \). \( L^T \cdot M \leq B \) means \( n \) GMECs. Among them, some are dependent on others.

**Definition 10:** Let \( L^T \cdot M \leq B \) be a set of inequalities, \( M = \{M[i]^T \cdot M \leq b_i, \forall i \in \mathbb{N}_n\} \), and \( M_{\mathbb{N}_n \setminus \{k\}} = \{M[i]^T \cdot M \leq b_i, \forall i \in \mathbb{N}_n - \{k\}\} \). \( l_k^T \cdot M \leq b_k \) is said to be dependent on other inequalities iff \( M = M_{\mathbb{N}_n \setminus \{k\}} \).

**Proposition 3:** An inequality \( l_k^T \cdot M \leq b_k \) is dependent on the others iff \( \min \{b_k - l_k^T \cdot M, M \in M_{\mathbb{N}_n \setminus \{k\}}\} \geq 0 \).

**Proof:** For the necessary part, we have \( M = M_{\mathbb{N}_n \setminus \{k\}} \) according to Definition 10. Since \( l_k^T \cdot M \leq b_k \) holds in the space determined by \( M \) according to the hypothesis of this proposition, it is apparent that \( l_k^T \cdot M \leq b_k \) holds in \( M_{\mathbb{N}_n \setminus \{k\}} \). This means that \( \min \{b_k - l_k^T \cdot M, M \in M_{\mathbb{N}_n \setminus \{k\}}\} \geq 0 \).

For the sufficient part, we have \( l_k^T \cdot M \leq b_k \) because of \( \min \{b_k - l_k^T \cdot M, M \in M_{\mathbb{N}_n \setminus \{k\}}\} \geq 0 \). This means \( M_{\mathbb{N}_n \setminus \{k\}} \) implicitly ensures \( l_k^T \cdot M \leq b_k \). Moreover, the combination of \( M_{\mathbb{N}_n \setminus \{k\}} \) and \( l_k^T \cdot M \leq b_k \) is equivalent to \( M \). As a result, we have \( M = M_{\mathbb{N}_n \setminus \{k\}} \).

Proposition 3 implies that the dependance of one inequality upon others can be determined by solving a mathematical programming problem:

\[
\min b_k - l_k^T \cdot M
\]

subject to

\[
l_k^T \cdot M \leq b_i, \quad i \in \mathbb{N}_n - \{k\}
\]

**Corollary 2:** An inequality \( l_k^T \cdot M \leq b_k \) is independent iff \( \min \{b_k - l_k^T \cdot M, M \in S_{\mathbb{N}_n \setminus \{k\}}\} < 0 \).

**Proof:** Owing to its duality to Proposition 3, this statement holds obviously.

Consider the three inequalities in Fig. 2(a). To verify one’s dependency upon the other two, we can have the following three mathematical programming formulations.

For \( M(p_4) + M(p_9) \leq 2 \), we have

\[
\min 2 - M(p_4) - M(p_9)
\]

subject to

\[
M(p_5) + M(p_8) \leq 2
\]
\[
M(p_4) + M(p_5) + M(p_8) + M(p_9) \leq 4
\]
For $M(p_2) + M(p_8) \leq 2$, we have

$$\min \ 2 - M(p_5) - M(p_8)$$

subject to

$$M(p_4) + M(p_9) \leq 2$$
$$M(p_4) + M(p_5) + M(p_8) + M(p_9) \leq 4$$

For $M(p_4) + M(p_8) + M(p_9) \leq 4$, we have

$$\min \ 4 - M(p_4) - M(p_5) - M(p_8) - M(p_9)$$

subject to

$$M(p_4) + M(p_8) \leq 2$$
$$M(p_5) + M(p_8) \leq 2$$

There exists an optimal solution

$$\max \ M \leq \sum_{i \in \mathbb{N}_n \setminus \{k\}} \alpha_i \cdot b_i$$

According to Definition 10, we know that $l_k^T \cdot M \leq b_k$ is dependent on other inequalities, i.e., $l_k^T \cdot M \leq b_k$, where $i \in \mathbb{N}_n \setminus \{k\}$.

In Fig. 2(a), $M(p_4) + M(p_5) + M(p_8) + M(p_9) \leq 4$ is dependent on the other two, i.e., $(M(p_4) + M(p_5)) + (M(p_8) + M(p_9)) \leq 4$. Thus, because $(M(p_4) + M(p_5)) + (M(p_8) + M(p_9)) \leq 4$, we have $2 \leq 4$. An important and interesting issue is that we can replace the scalar 4 with 2 so that $2 \leq 2$ holds, which makes $M(p_4) + M(p_9) \leq 2$ dependent on $M(p_5) + M(p_8) \leq 2$ and $M(p_4) + M(p_5) + M(p_8) + M(p_9) \leq 4$. The same analysis applies to the dependency of $M(p_5) + M(p_8) \leq 2$ on the other.

In Fig. 2(b), $2M(p_4) + 3M(p_5) + M(p_8) + 2M(p_9) \leq 7$ is independent of the other two, i.e., $2M(p_4) + 2M(p_9) \leq 3$ and $M(p_5) + M(p_8) \leq 4$. This is because, although $2M(p_4) + 2M(p_9) + 3(M(p_5) + M(p_8)) \geq 2M(p_4) + 3M(p_5) + M(p_8) + 2M(p_9)$, we have $7 \geq 3 \times 4 + 3$. To make $2M(p_4) + 3M(p_5) + M(p_8) + 2M(p_9) \leq 7$ dependant, we can decrease $4$ to $1$ such that $7 \geq 3 \times 1 + 3$. $2M(p_4) + 2M(p_9) \leq 3$ is not dependent on $M(p_5) + M(p_8) \leq 4$ and $2M(p_4) + 3M(p_5) + M(p_8) + 2M(p_9) \leq 7$. This is because, despite the fact that $2p_4 + 2p_5 \leq 2p_4 + 3p_5 + p_8 + 2p_9$, we have $3 \geq 7$. Similarly, we can decrease the right-hand scalar $7$ to $3$ such that $3 \leq 3$ holds, which makes $2M(p_4) + 2M(p_9) \leq 3$ dependent on $M(p_5) + M(p_8) \leq 4$ and $2M(p_4) + 3M(p_5) + M(p_8) + 2M(p_9) \leq 3$. The same analysis applies to the dependency of $M(p_5) + M(p_8) \leq 4$ on the other.

**B. Analysis of Independent and Dependent Inequalities**

It is noticed that the above analysis presents a simple procedure to derive the dependency relationship between one inequality and the others. An interesting issue is that the independent inequalities are actually not unique. As shown by the examples, two independent inequalities may result when different sets of inequalities are determined as the dependent ones. With the aid of a multisite technique, we herein present some structural analysis of these inequalities and their dependency relationships.

**Proposition 4:** Let $L^T \cdot M \leq B$ be a set of inequalities and $k \in \mathbb{N}_n$. If $l_k^T \cdot M \leq b_k$ is dependent on other inequalities, $l_k \in L \setminus \{k\}$.

Proof: We can prove it by contradiction. Suppose that $l_k$ is dependent on $l_i$ where $i \in \mathbb{N}_n \setminus \{k\}$ and $\exists p \in \|l_k\|$ so that $t \notin \mathbb{N}_n \setminus \{k\}$. Then, we have $l_k(p) \geq 1$ and $\sum_{t \in \mathbb{N}_n \setminus \{k\}} l_t(p) = 0$. As a result, we have $l_k(p) > \sum_{t \in \mathbb{N}_n \setminus \{k\}} l_t(p)$, leading to the fact that $l_k \notin \mathbb{N}_n \setminus \{k\}$.

In Fig. 2(a), $M(p_4) + M(p_5) \leq 2$ and $M(p_5) + M(p_8) \leq 2$ cannot be dependent on each other because neither $\{p_4, p_5\} \subseteq \{p_5, p_8\}$ nor $\{p_5, p_8\} \subseteq \{p_4, p_5\}$ is true. Both $M(p_3) + M(p_9) \leq 2$ and $M(p_5) + M(p_8) \leq 2$ are dependent on $M(p_4)$.
+ M(p_5) + M(p_8) + M(p_9) \leq 4 because \{p_4, p_9\} \subseteq \{p_4, p_5, p_8, p_9\} and \{p_5, p_8\} \subseteq \{p_4, p_5, p_8, p_9\}. M(p_4) + M(p_5) + M(p_8) + M(p_9) \leq 4 is not dependent on either M(p_4) + M(p_9) \leq 2 or M(p_5) + M(p_8) \leq 2 because \{p_4, p_8, p_9\} \not\subseteq \{p_4, p_9\} and \{p_3, p_5, p_8, p_9\} \not\subseteq \{p_5, p_8\}. Nevertheless, M(p_4) + M(p_5) + M(p_8) + M(p_9) \leq 4 is dependent on the union of M(p_4) + M(p_9) \leq 2 and M(p_5) + M(p_8) \leq 2 because \{p_4, p_5, p_8, p_9\} \subseteq \{p_4, p_9\} \cup \{p_4, p_5, p_8, p_9\}. To be concise, the analysis upon the Petri net model in Fig. 2(b) is omitted owing to its similarity to the case in Fig. 2(a).

**Theorem 3:** Let \( L^T \cdot M \leq B \) be a set of inequalities and \( k \in \mathbb{N}_n \). \( I_k^T \cdot M \leq b_k \) is dependent on other inequalities iff there exist \( n - 1 \) nonnegative coefficients \( \alpha_i, i \in \mathbb{N}_n \setminus \{k\} \) such that \( \|l_k\| \leq \sum_{i \in \mathbb{N}_n \setminus \{k\}} \alpha_i \cdot \|l_i\| \) and \( b_k \geq \sum_{i \in \mathbb{N}_n \setminus \{k\}} \alpha_i \cdot b_i \).

**Proof:** This proof is a multisets version of the proof of Theorem 2. Owing to their similarity, it is omitted.

**C. Supervisor Simplification**

Suppose \( \mathcal{L} = \{ ||l_i|| \} \), where \( i \in \mathbb{N}_n \). From the above analysis, the key issue is to identify \( \Omega = \{ \omega_1, \omega_2, \ldots, \omega_m \} \) \( \subseteq \mathbb{N}_n \) leading to two sets, i.e., \( \mathcal{L}_L = \{ ||l_i|| : i \in \Omega \} \subseteq L \) and \( \mathcal{L}_D = L \setminus \mathcal{L}_L \) such that: 1) \( \forall \omega_k \in \Omega, \exists \alpha_k \geq 0 \) such that \( l_{w_k} = \sum_{i \in \omega_k} \alpha_i \cdot l_i \omega \); and 2) \( \forall \mu \in \mathbb{N}_n \setminus \Omega, \exists \mu \geq 0 \), \( l_{\mu} \leq \sum_{i \in \mu} \alpha_i \cdot l_i \omega \). Obviously, \( \mathcal{L}_L \) is not unique.

**Definition 11:** Let \( i \in \Omega, j = \mathbb{N}_n \setminus \Omega \). \( \{ ||l_i\|, ||l_\omega\|, \ldots, ||l_\omega\| \} \) is called a max-\( L \)-set (resp., min-\( L \)-set) if \( \forall i, j \), \( ||l_i|| \geq ||l_j|| \) (resp., \( ||l_i|| \leq ||l_j|| \)).

As opposed to \( \mathcal{L} \), we have \( \mathcal{B} = \{ b_i \} \), where \( i \in \mathbb{N}_n \), \( \mathcal{B}_L = \{ b_i \} \), where \( i \in \Omega \), and \( \mathcal{B}_D = \{ b_i \} \), where \( i \in \mathbb{N}_n \setminus \Omega \). \( L_D, B_D, \) and \( B_L \) denote vectors while their corresponding multisets are represented by \( \mathcal{L}_D, B_D, \) and \( B_L \).

Take the Petri net model in Fig. 2(a) as an example. \( M(p_4) + M(p_5) + M(p_8) + M(p_9) \leq 4 \) is a max-\( L \)-set while \( M(p_4) + M(p_9) \leq 2 \) and \( M(p_5) + M(p_8) \leq 2 \) are a min-\( L \)-set.

Therefore, to reduce the supervisor size, our first step is to identify these dependent inequalities, which can be realized by the following algorithm.

**Algorithm 1:** Identification of Independent and Dependent Inequalities

**Input:** \( L^T \cdot M \leq B \)

**Output:** Independent Inequalities \( L^T \cdot M \leq B^L \) and Dependent Inequalities \( L^T \cdot M \leq B^D \) with \( L^T = [l_{w_1}, l_{w_2}, \ldots, l_{w_m}], B^L = [b_{w_1}, b_{w_2}, \ldots, b_{w_m}], L^D = [l_{\mu_1}, l_{\mu_2}, \ldots, l_{\mu_{m-n}}], \) and \( B^D = [b_{\mu_1}, b_{\mu_2}, \ldots, b_{\mu_{m-n}}] \) where \( \{\mu_1, \mu_2, \ldots, \mu_{m-n}\} \subseteq \mathbb{N}_n \setminus \{\omega_1, \omega_2, \ldots, \omega_m\} \).

1) \( i = 1, m = 1, \mathcal{L}_L := \emptyset, \mathcal{L}_D := \mathcal{L}, \mathcal{B}_L := \emptyset, \) and \( \mathcal{B}_D := \mathcal{B} \);
2) Arrange all the elements in \( \mathcal{L} \) according to the ascending (resp., descending) order of |||l||| for the min-\( L \)-set (resp., max-\( L \)-set), respectively;
3) while \( i \leq |\mathcal{L}| \) do begin
4) Check whether \( \exists \alpha_j \geq 0 \) so that \( l_i \leq \sum_{j=1}^{m} \alpha_j \cdot l_j \). If so, go to Step 6; otherwise, go to Step 5;
5) \( m := m + 1, \mathcal{L}_L := \mathcal{L}_L \cup \{l_i\}, \mathcal{L}_D := \mathcal{L}_D \setminus \{l_i\}, \mathcal{B}_L := \mathcal{B}_L \cup \{b_i\}, \) and \( \mathcal{B}_D := \mathcal{B}_D \setminus \{b_i\} \);
6) \( i := i + 1 \);
end

Based on Algorithm 1, the inequality set is divided into two disjoint sets, i.e., \( L_L \) and \( L_D \). The computational complexity is polynomial with regard to the number of inequalities. Using Theorem 3, we can decrease the right-hand scalars such that the ones in \( L_L \) can be ignored during the supervisor synthesis process. Algorithm 1 distinguishes the independent and dependant inequalities without resorting to integer programming techniques whose computational complexity proves to be exponential. Its execution involves only a limited number of comparison between some coefficient vectors and scalars.

**Algorithm 2:** Supervisor Simplification

**Input:** \( L^T \cdot M \leq B \)

**Output:** \( L_L \cdot M \leq B_L \)

1) Identify \( L_L \) and \( L_D \) using Algorithm 1;
2) \( i := 0 \);
3) while \( (i < n - m) \) do begin
4) \( i := i + 1 \);
5) Find \( \alpha_j \geq 0, \) \( j \in \mathbb{N}_m \), such that \( l_{\mu_j} \leq \sum_{j=1}^{m} \alpha_j \cdot b_j \);
6) Decrease \( b_j \), \( j \in \mathbb{N}_m \), such that \( b_j \geq \sum_{j=1}^{m} \alpha_j \cdot b_j \).

**end**

**end**

n, m, and \( n - m \) denote the numbers of all, independent, and dependant inequalities, respectively. \( l_{\mu_j} \) and \( b_j \) (resp., \( l_{\mu_j} \) and \( b_j \)) are the coefficient vectors and scalars of the dependent (resp., independent) inequalities. In Fig. 2(a), \( M(p_4) + M(p_9) \leq 2 \) and \( M(p_5) + M(p_8) \leq 2 \) are identified as min-\( L \)-set while \( M(p_4) + M(p_5) + M(p_8) + M(p_9) \leq 4 \) is identified as max-\( L \)-set. For Fig. 2(b), \( 2M(p_4) + 2M(p_9) \leq 3 \) and \( M(p_5) + M(p_8) \leq 4 \) are identified as min-\( L \)-set while \( 2M(p_4) + 3M(p_5) + M(p_8) + 2M(p_9) \leq 7 \) is identified as max-\( L \)-set. Fig. 3 represents all these monitors where \( p_{c_1} \) and \( p_{c_2} \) realizes \( M(p_4) + M(p_9) \leq 2 \) and \( M(p_5) + M(p_8) \leq 2 \), \( p_{c_3} \) realizes \( M(p_4) + M(p_5) + M(p_8) + M(p_9) \leq 4 \); \( p_{c_4} \) and \( p_{c_5} \) realizes \( 2M(p_4) + 2M(p_9) \leq 3 \), \( M(p_5) + M(p_8) \leq 4 \), and \( 2M(p_4) + 3M(p_5) + M(p_8) + 2M(p_9) \leq 7 \).

min-\( L \)-set may lead to more, if not the most, permissive control control whereas max-\( L \)-set can result in a concise, if not the simplest, supervisor. For the Petri net model in
compact supervisor. can achieve the maximal permissiveness with a structurally way, thus avoiding astronautical number of monitors. Most of enforcing supervisor can be synthesized in a quite compact three ones. After the imposition of the first two inequalities, system is controlled by the first two inequalities or all the one is the first research to cope with siphon-induced deadlock problems in general Petri nets while the latter aims to derive the simplest supervisors. A comparison is thus conducted only between our method and the ones in [2] and [26].

TABLE III
SUPERVISOR FOR PETRI NET IN FIG. 2(a)

<table>
<thead>
<tr>
<th>Method</th>
<th>(M_0(p_{c1}))</th>
<th>(p_{c1})</th>
<th>(p_{s})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Approach</td>
<td>1</td>
<td>({t_6, t_{10}})</td>
<td>({t_3, t_5})</td>
</tr>
<tr>
<td>Approach in [26]</td>
<td>1</td>
<td>({t_6, t_{10}})</td>
<td>({t_3, t_5})</td>
</tr>
<tr>
<td>Approach in [2]</td>
<td>1</td>
<td>({t_6, t_{10}})</td>
<td>({t_3, t_5})</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>({t_5, t_{10}})</td>
<td>({t_3, t_5})</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>({t_5, t_{10}})</td>
<td>({t_3, t_5})</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>({t_6, t_{10}})</td>
<td>({t_3, t_5})</td>
</tr>
</tbody>
</table>

TABLE IV
SUPERVISOR FOR PETRI NET IN FIG. 2(b)

<table>
<thead>
<tr>
<th>Method</th>
<th>(M_0(p_{c2}))</th>
<th>(p_{c2})</th>
<th>(p_{s})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Approach</td>
<td>1</td>
<td>({3t_6, 2t_{10}})</td>
<td>({2t_5, t_5, t_8, t_9})</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>({2t_6, 2t_{10}})</td>
<td>({2t_5, t_5})</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>({t_6, 6t_9})</td>
<td>({t_5, t_8})</td>
</tr>
<tr>
<td>Other Approach</td>
<td>3</td>
<td>({2t_6, 2t_{10}})</td>
<td>({2t_5, t_5, t_8, t_9})</td>
</tr>
</tbody>
</table>

Fig. 2(a), the same behavior can be obtained whether the system is controlled by the first two inequalities or all the three ones. After the imposition of the first two inequalities, the third one’s implicitness is obvious and its removal has no influence upon the system, thus leading to a maximal permissiveness with a simplified supervisor structure. If one only removes the inequalities whose removal has no influence upon the system behavior, the controlled system definitely can achieve the maximal permissiveness with a structurally compact supervisor.

For an \(S^4R\), the above results indicate that the liveness enforcing supervisor can be synthesized in a quite compact way, thus avoiding astronomical number of monitors. Most of the existing approaches, such as [12], [17], are not applicable. Approaches in [2], [26] are feasible since they consider general nets. Moreover, they are representative because the former one is the first research to cope with siphon-induced deadlock problems in general Petri nets while the latter aims to derive the simplest supervisors. A comparison is thus conducted only between our method and the ones in [2] and [26].

For both nets in Fig. 2, the method in [2] leads to 3 monitors, namely, \(p_{c1} - p_{c2}\), since there are 3 siphons. These monitors constitute supervisors as shown in Tables III and IV, respectively.

In Fig. 2(a), among 3 siphons, 2 of them, are found to be elementary siphons according to [26]. Thus, 2 monitors are necessary to constitute a supervisor, as shown in Table III. In Fig. 2(b), no siphons are found to be elementary siphons. Therefore, the supervisor remains the same as in [2] and is shown in Table IV.

By contrast, our approach shows that only one monitor is necessary for each Petri net because inequalities \(M(p_4) + M(p_5) + M(p_8) + M(p_9) \leq 2\) and \(2M(p_4) + 3M(p_5) + M(p_8) + 2M(p_9) \leq 3\) prove to be independent in Figs. 2(a) and (b), respectively. It shows that our approach can reduce the supervisor size more compared to approaches in [2] and [26]. For all supervisors, the net liveness can be achieved after their outgoing transitions move to \(p_{s}\) as done in [12].

V. ILLUSTRATIVE EXAMPLE

Fig. 4 shows the block diagram of an AMS where three product types, i.e., \(J_1 - J_3\), are manufactured. It is composed of three robots \(R_1 - R_3\) and four machines \(M_1 - M_4\). Each robot can hold two products. Each machine can deal with four products at a time. There are three loading buffers \(I_1 - I_3\) and three unloading buffers \(O_1 - O_3\) to load and unload the AMS. The action area for \(R_1\) is \(I_1, O_1, M_1, M_2, M_3, \) and \(M_4\); for robot \(R_2\) is \(I_2, O_3, M_1,\) and \(M_3\); for robot \(R_3\) is \(I_3, O_2, M_2,\) and \(M_4\). By these resources, products \(J_1 - J_3\) can be concurrently manufactured. Every arriving raw product belongs to one of these three products. According to the predefined routes, a raw product \(J_1\) is taken from \(I_1\) by \(R_1\). After being processed by \(M_2\), it is moved to \(O_1\) by \(R_1\). A raw product \(J_2\) is taken from \(I_2\) by \(R_2\). Two flexible routes are available for its further treatment. First, it is manufactured in \(M_1\) and then moved to \(M_2\) by \(R_1\). Second, it is manufactured...
in M₂ and then moved to M₃ by R₁. After its process in either M₂ or M₅, it is moved to O₂ by R₃ so that a final product of J₂ is obtained. A raw product of J₃ is taken from I₅ by R₃. After being processed by M₄, it is moved to M₃ by R₁. After that, it is moved to O₃ by R₂. Noticeably, two copies of resources are required when J₂ is processed by M₁ or J₃ is processed by M₁. In Fig. 4, resource allocation is indicated by a directed arc as labeled by t_j, j ∈ N_{20}. A number across the arc indicates the resource quantity while the default value is one.

Fig. 5 shows the net model of this AMS, which allows multiple resource acquisitions and flexible routes. The system is an S₄R where P₀ = {p₁, p₅, p₁₄}, P₁ = {p₂₋p₄}, P₂ = {p₆₋p₁₃}, P₃ = {p₁₅₋p₁₉}, P₄ = {p₂₀₋p₂₆}, l₀ = t₁, l₂ = t₅, and l₀ = t₁₅. Places p₂₀₋p₂₆ denote M₁, M₂, R₁, R₂, M₃, R₄, and R₅, respectively. Initially, it is assumed that there are no parts in process. M(p₁) = M(p₅) = M(p₁₅) = 8 represents that the maximum job instances that are allowed for part types J₁ = J₅ at a time, respectively.

### TABLE VI

**GENERATED MONITORS FOR THE NET IN Fig. 5 DUE TO min-L₇**

<table>
<thead>
<tr>
<th>i</th>
<th>M(pₙ)</th>
<th>pₙ</th>
<th>M(pₙ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>{t₅, t₁₅}</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>{t₅, t₁₅}</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>{t₅, t₁₅}</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>{t₅, t₁₅}</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>{t₅, t₁₅}</td>
<td>13</td>
</tr>
</tbody>
</table>


For the net in Fig. 5, the method in [2] leads to 21 monitors, namely, M(p₄) = M(p₁₅), respectively corresponding to S₁₋S₂₁, as shown in Table VII. When these 21 monitors are added, a liveness-enforcing supervisor is obtained.

Among the above-derived 21 siphons S₁₋S₂₁, 8 of them, i.e., S₁₋S₈, S₁₀, and S₁₃, are found to be elementary ones according to [26]. Thus, only 8 monitors are necessary to constitute a supervisor, as shown in Table VIII.

Further analysis shows that the permissible states produced by the approaches in [2] and [26] are 642743 and 986, respectively. The work in [2] is the first approach on the liveness supervisory control in terms of general Petri nets. Its solution requires that all siphons satisfy the max-CS property. Therefore, the size of a supervisor is proportional to the number of siphons, which proves to be in an exponential relationship with the size of the plant model. Compared with [2], our method can implement a supervisor with much fewer monitors. The work in [26] distinguishes siphons as elementary and dependent ones. Only the elementary ones are required to be controlled. However, this method does not really provide the lower bound of the size of the liveness-supervising supervisor, particularly in cases where some weakly dependent siphons exist. This defect can be exemplified by the net shown in Fig. 2(a). According to the approach in [26], S₁ can be a weakly dependent siphon with regard to the corresponding elementary siphons, i.e., S₂ and S₃, since η₁ = η₃ - η₂. The theory in [26] requires two monitors to be imposed on S₂ and S₃, respectively. Comparatively, our approach shows that only one monitor is necessary because both M(p₄) + M(p₁₅) ≤ 2 and M(p₅) + M(p₆) ≤ 2 are dependent on M(p₄) + M(p₅) + M(p₆) ≤ 4 after we decrease the right-hand scalar 4 to 2. Therefore, compared with [26], our method can produce more concise supervisor structure and syntactically explain the principle behind such a reduction. Since their applicability is validated through P-invariants, elementary siphons must involve some characteristic vectors so as to establish supervisor simplification techniques. Their computation is unnecessarily complicated compared with our work which distinguishes the independent and dependent inequalities in a quite straightforward way. Thus, our method outperforms existing strategies based on elementary siphons.

### TABLE VIII

**GENERATED MONITORS FOR THE NET IN Fig. 5 DUE TO min-L₇**

<table>
<thead>
<tr>
<th>i</th>
<th>M(pₙ)</th>
<th>pₙ</th>
<th>M(pₙ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>{t₅, t₁₅}</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>{t₅, t₁₅}</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>{t₅, t₁₅}</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>{t₅, t₁₅}</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>{t₅, t₁₅}</td>
<td>13</td>
</tr>
</tbody>
</table>

### VI. CONCLUSION

This work focuses on the synthesis of liveness enforcing supervisors of automated manufacturing systems allowing both
flexible routes and multiple resource acquisition operations. Deadlocks are related to the emergence of a class of Petri net objects, namely undermarked siphons. To avoid their undermarkedness, the number of tokens in these siphons’ complementary places must be strictly limited, thus leading to a set of general mutual exclusive constraints. They can be distinguished as the independent and dependent ones. The dependent ones can be implicitly controlled after these independent ones are properly controlled. This technique can reduce the number of needed monitors. Numerical results show that our resulting supervisor is simple in its structure and can ensure more permissive behavior. The generalization of the research results in this paper will be conducted for more complex systems.

**REFERENCES**


Hesuan Hu (M’11–SM’12) received the B.S. degree in computer engineering and the M.S. and Ph.D. degrees in electromechanical engineering from Xidian University, Xi’an, China, in 2003, 2005, and 2010, respectively.

From 1997 to 2000, he was an Assistant Engineer with the Metrology Department, Xi’an Aircraft Engine Control Engineering Company Ltd., where he served as the Director of the Electrical and Electronic Laboratory from 2001 to 2002 and the Leader for several innovation projects. From 2002 to 2011, he worked as a Research Scholar in the Department of Electrical and Computer Engineering at New Jersey Institute of Technology, Newark, NJ, USA. At the same institute, he also served as an Adjunct Instructor in charge of several courses in the field of electrical engineering at the graduate level. He is currently a Ph. D advisor at Xidian University. His research interests include Petri nets, discrete event systems, automated manufacturing systems, multimedia streaming systems, and artificial intelligence. In the aforementioned areas, he has over 60 publications in journals, book chapters, and conference proceedings.

Dr. Hu was a Program Committee member for the numerous conferences including the 2008 International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems, the 2008 IEEE International Conference on Automation Science and Engineering, the 2008-2013 IEEE International Conference on Networking, Sensing and Control, the 2010 IEEE International Conference on Systems, Man, and Cybernics (SMC), and the 2010 IEEE International Conference on Mechatronics and Automation. He serves as a frequent Reviewer for more than 30 international journals, including the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS-PART A: SYSTEMS AND HUMANS, the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING; the IEEE COMMUNICATIONS LETTERS; the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY; the IEEE TRANSACTIONS ON COMPUTERS; the ACM TRANSACTIONS ON EMBEDDED COMPUTING SYSTEMS; and DISCRETE EVENT DYNAMIC SYSTEMS. He is on the Editorial Board of 10+ international journals He is listed in Who’s Who in North American Education, Who’s Who in America, Who’s Who in the World. He was the recipient of many national and international awards including the Franklin V. Taylor Outstanding Paper Award from the IEEE SMC Society in 2010 and the finalist of the Best Automation Paper from the IEEE ICRA Society in 2013.

Yang Liu graduated in 2005 with a Bachelor of Computing in the National University of Singapore (NUS). In 2010, he obtained his PhD and continued with his post doctoral work in NUS. Since 2012, he joined Nanyang Technological University as a Nanyang Assistant Professor. His research focuses on software engineering, formal methods and security. Particularly, he specializes in software verification using model checking techniques. This work led to the development of a state-of-the-art model checker, Process Analysis Toolkit.