7 Structural Deadlock Prevention Policies for Flexible Manufacturing Systems

A Petri Net Outlook

Juan-Pablo López-Grao, José-Manuel Colom and Fernando Tricas

CONTENTS

7.1 INTRODUCTION

Petri nets (PNs) [29] have a success story as models aimed at the study of *resource allocation systems (RASs)* from a systemic perspective [3]. An RAS is a discrete event system in which a set of concurrent processes coexist, and these must compete in order to be granted the allocation of some shared resources. Deadlocks arise when a set of processes is indefinitely waiting for resources that are already held by other processes of the same set. From a qualitative standpoint, the RAP consists

in meeting the demand for resources of the set of processes while dealing with the set of potential system deadlocks.

PNs constitute a fertile ground to deal with such deadlocks. Many real-world RASs can be abstracted into a conceptualization constructed around two entities: processes and resources. PNs are constructively simple models which feature an appealing graphical representation for modelling these abstractions [3]. Besides, there exist powerful structural results for certain subclasses of PNs for RASs which enable powerful analysis and synthesis techniques for identifying and fixing potential or factual deadlocks [5,31,38]. In the end, the corrections computed for the model are deployed over the real-world system.

This methodology has been successfully applied to FMSs, although some other application areas have been approached from this perspective [9,25,35]. The adopted viewpoint is at the global coordination level in the typical hierarchical architecture of the control of an FMS. From this perspective, the system is composed of processes (parts, jobs, etc.,) following predefined production plans and resources which can be artefacts such as robots, machines or conveyor belts or passive elements such as storage area. Diverse classes of PN models, such as S^3PR [5], S^4PR [31,38] and many others (e.g., augmented marked graphs [2]), were defined for this aim, with specific attributes for modelling different plant configurations of FMSs.

The particular physical constraints of FMSs at this level have usually led to syntactic restrictions of these classes that enable a structural characterization of liveness by means of the concept of insufficiently marked siphon. This has proven to be a fertile ground for many works that try to exploit this fact. In effect, considering the blooming of publications around PN-based RAS approaches in the context of FMSs, it can be observed that the most significant proliferation of works emerges in the context of synthesis. Most of these papers are related to siphon computation [1,22] as well as to applying integer linear programming (ILP) to liveness enforcing [17,38]. Another family of works focuses on synthesis based on reachability state analysis and on the theory of regions [14,39].

This chapter aims at presenting an overview of some cutting-edge techniques in exploiting the structure of a RAS model of an FMS for deadlock prevention, from the perspective of using PN as modelling paradigm. As we shall see, some of these approaches take advantage of the concept of insufficiently marked siphon in rather different ways. Section 7.2 presents the basic fundamentals regarding the abstraction and modelling of FMSs perceived as RASs through PN along with a review of some reference models and analysis results relevant to them. Section 7.3 provides an overview of the structural synthesis techniques based on the introduction of virtual resources (monitors) for the restriction of anomalous behaviour. The nature of such techniques is illustrated through the detailed review of an iterative correction methodology belonging to this family of methods. In Section 7.4, there is another family of liveness enforcing techniques based on the privatization of the usage of some system resources by the production plans in order to prevent deadlocks. Finally, Section 7.5 summarizes the main conclusions of the chapter. Basic PN concepts and notations are briefly presented in Annex A.

7.2 PETRI NET MODELLING OF FMSs PERCEIVED AS RASs

7.2.1 RAS Abstraction and Modelling Methodologies

The use of PN in RASs is an active research field devoted to define and exploit different subclasses of PN allowing to model the widest set of RASs. The definition of these subclasses is based on the net structure, allowing to obtain structure-based characterizations of the partial/total deadlocks. The goal of these characterizations is to synthesize controllers preventing/avoiding deadlocks arising from the resource allocation.

Most of the approaches to construct the PN model of an RAS assume a modular methodology in three steps:

- *Characterization of the production plans*. Each part or part component that enters the system is a process. Processes are modelled as tokens that move through a PN, $\mathcal{N}$, representing the production plan for a type of product. The places (partial states) of $\mathcal{N}$ are related to the different operations (either transformations, handling or assembly/disassembly operations) to be carried out over the parts contained in these places. The transitions of $\mathcal{N}$ allow to progress a part towards its final state. A production plan has distinguished input points of raw material and output points of final products. The execution of a production plan is achieved by the execution of a production path, and several of them can exist in the same production plan. A production path is a sequence of transitions firable in $\mathcal{N}$, whose occurrence represents the production of a finished product.
- *Incorporation of resources to each production plan*. A physical element composing the FMS (a machine, a buffer, a robot, a tool, etc.) is a resource with a given capacity (the number of parts that, at a given time, the resource is able *to store/to be used by*). Each state of a production plan has associated the (multi-)set of resources needed for the corresponding processing step (including the buffering capacity to hold the part itself). A resource type is represented by means of a place whose initial marking represents either the number of available copies of the resource or its capacity. A resource place has input (output) arcs to (from) those transitions of a production plan that moves a process to (from) a state that requires (was using) a number of copies of this resource type. In all cases, the considered resources can neither be created nor destroyed.
- *Construction of the global model by composition of the production plans with resources*. In an FMS, there is a set of production plans: one type per each type of product. In order to obtain the global model of the RAS, we must compose the production plans with the needed resources. This composition is based on the fusion of the resource places representing the same resource type in the different production plans. The initial marking of the resources, after the fusion, normally is computed as the maximum of the initial markings of the instances that have been merged.

This common methodology has been applied to the modelling of restricted classes of RASs by means of PNs that are defined imposing restrictions either on the class of production plans to be considered or on the way that system resources can be used by a production plan at a given state.

A first group of RASs are those where a workpiece only undergoes successive transformations until its final state. In this group, a production plan is represented (or modelled) by means of a state machine. Additional restrictions on the production plans refer to the availability of different routings in the system; another important question is whether a part can choose different paths once it is in the system. The first feature is offered in some models, but many of them do not allow online decisions, and the path is fixed once the part selects one of the available routes. Several studies allow online decisions for part routing [5,8].

Restrictions related to resources, in this first group, refer to the number and type of resources that can be used by a process at a given state. In most previous work, only one resource of a unique type was allowed at each state of each process. This restriction was finally relaxed for the most general case in which conservative resources are considered [8]: alternative routings are allowed in processes and no restriction is imposed on the resources that can be used at a given state of a process.

A second group of RASs presents a concurrent processing nature. This is usually due to some assembly/disassembly operations that introduce the possibility of independent processing steps of different part components. In this case, more complicated models are needed to represent the production plans of the involved components [7,20,21,33]. As a matter of fact, while a lot of work related to the first group of RASs can be found in the literature, it is much more difficult to find solutions for this second group from a structural point of view [2]. For this reason, we will focus on RASs with sequential processes in the following.

7.2.2 Taxonomy of RAS Oriented Nets

Let us now define the class of S^{4}PR nets: a class of PN to model RASs, following the methodology just outlined. In these nets, the production plans are modelled by means of strongly connected state machines with no internal cycles and they share a set of non-consumable and reusable resources.

One of the most interesting features of this kind of models is their composability. Two S^{4}PR nets can be composed into a new S^{4}PR model via fusion of the common resources. Since multiple resource reservation is allowed, S^{4}PR nets are not ordinary, that is, the weight of the arcs from the resources to the state machines (or vice versa) is not necessarily equal to one.

Definition 7.1 [38] Let $I_{\mathcal{N}}$ be a finite set of indices. An S^{4}PR net is a connected generalized pure P/T net $\mathcal{N} = \langle P, T, \mathbf{C} \rangle$ where

1. $P = P_0 \cup P_S \cup P_R$ is a partition such that
   a. [idle places] $P_0 = \bigcup_{i \in I_{\mathcal{N}}} \{p_{0_i}\}$
   b. [process places] $P_S = \bigcup_{i \in I_{\mathcal{N}}} P_i$, where
   $$\forall i \in I_{\mathcal{N}} : P_i \neq \emptyset \text{ and } \forall i, j \in I_{\mathcal{N}} : i \neq j, P_i \cap P_j = \emptyset$$
   c. [resource places] $P_R = \{r_1, ..., r_n\}, n > 0$
2. $T = \bigcup_{i \in I_{\mathcal{N}}} T_i$, where $\forall i \in I_{\mathcal{N}}, T_i \neq \emptyset$, and $\forall i, j \in I_{\mathcal{N}}, i \neq j, T_i \cap T_j = \emptyset$
3. For each $i \in I_{\mathcal{N}}$, the subnet generated by $\{p_{0_i}\} \cup P_{S_i}, T_i$ is a strongly connected state machine such that every cycle contains p_{0_i}
4. For each $r \in P_R$, there exists a unique minimal p-semiflow $\mathbf{y_r} \in \mathbb{N}^{|P|}$ such that $\{r\} = \|\mathbf{y_r}\| \cap P_R$, $\|\mathbf{y_r}\| \cap P_0 = \emptyset$, $\|\mathbf{y_r}\| \cap P_S \neq \emptyset$, and $\mathbf{y_r}[r] = 1$
5. $P_S = \bigcup_{r \in P_R} (\|\mathbf{y_r}\| \setminus \{r\})$

■

Figure 7.1 depicts a net system belonging to the S^{4}PR class. Places R and S are the *resource places*, which represent two different types of resources, while the number of tokens in them represents the quantity of free instances of each type of resource. If we remove these places, we get two isolated state machines. These state machines represent two different production plans in the FMS.

At the initial state, the unique tokens in each state machine are located at the so-called *idle place* (here, A and E). In general, the idle place can be seen as a mechanism which limits the maximum number of concurrent parts being processed in the same production plan. The rest of places model the various stages of the production plans as far as resource reservation is concerned. Consequently, tokens in these places represent parts which are being processed at the corresponding stages.

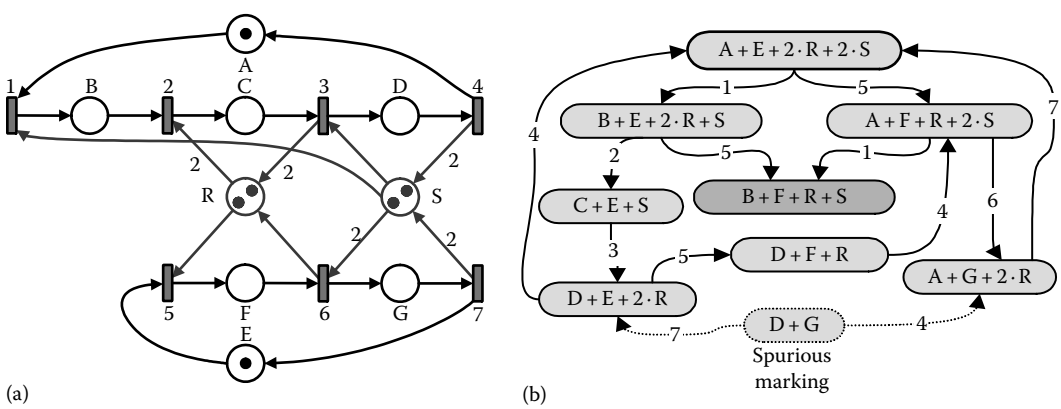

FIGURE 7.1 **(See colour insert.)** (a) An S^{4}PR net system. Despite being non-live, no minimal siphon is ever insufficiently marked. (b) Reachability graph of the net in (a).

Meanwhile, the transitions represent the acquisition or release of resources by the processes along their evolution through the production plan. Every time a transition fires, the total amount of resources available is altered while the part advances to the next stage. The weight of an arc connecting a resource with a transition models the number of instances which are allocated or released when a part advances.

For instance, place R could model a set of free robotic arms used to process parts in the stage C of the first production plan (two arms are needed per each part processed there) as well as in the stage F of the second production plan (only one arm needed per part processed). Consequently, if transition 5 is fired from the initial marking, then one robotic arm will be allocated and one part will visit stage F. Still, there will remain one robotic arm to be used freely by other processes.

Finally, it is worth noting that moving one isolated token of a state machine (by firing its transitions) until the token reaches back the idle state leaves the resource place markings unaltered. Thus, the resource usage is conservative. Colom [3] presents a more detailed example of the modelling of a manufacturing cell through an S^4PR net. That example features processes with alternate routings based on online decisions and processing stages in which several types of resources are used simultaneously.

The next definition formalizes the fact that there should exist enough free resource instances in the initial state so that every production plan is realizable:

Definition 7.2 [38] Let $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C}\rangle$ be an S^4PR net. An initial marking $\mathbf{m_0}$ is acceptable for $\mathcal{N}$ iff $\|\mathbf{m_0}\| = P_0 \cup P_R$ and $\forall p \in P_S, r \in P_R : \mathbf{m_0}[r] \geq \mathbf{y_r}[p]$. ■

This definition of admissible initial marking leads to an archetype of RASs for which there is no activity at the system startup and every production path is executable in isolation from the initial state. Furthermore, these RAS models hold a number of properties of good behaviour which are usually desirable in an FMS. The following property is a basic yet relevant one:

Lemma 7.1 [37] *Every S^4PR net is conservative and consistent.*

Observe that conservative and consistent nets are called well-formed nets because this is a necessary condition for structural liveness and structural boundedness.

The conservative nature of the net ensures that no resource is created or destroyed, as all resource places are covered by a p-semiflow. Indeed, the support of the unique minimal p-semiflow $\mathbf{y_r}$ indicates the process stages (process places) where resources of type r are used conservatively. On the other hand, the net consistency ensures that all production paths that constitute the production plans are reproducible.

Another interesting property of S^4PR net systems is that liveness equals reversibility. In other words, it is assumed that, given an intermediary execution state of the system, the initial state can always be recovered successfully unless some undesirable behaviour happens leading the system to a deadlock.

Lemma 7.2 [28] *Let $\langle \mathcal{N}, \mathbf{m_0}\rangle$ be an S^4PR net system with an acceptable initial marking. $\langle \mathcal{N}, \mathbf{m_0}\rangle$ is live iff $\mathbf{m_0}$ is a home state* (*i.e., the system is reversible*).

The earlier result is particularly interesting considering that in this type of RASs, triggering new processes can never increase the number of available resources in the system. Indeed, if we are not able to move all active processes towards termination, then the system has reached a deadlock from which it cannot be recovered autonomously [37].

Another interesting feature of this kind of nets is that if we have enough copies of a type of resource, then the latter is not a constraint to the system of processes and can be removed from the analysis of the RAP. This intuitive idea is captured in PNs by means of the concept of structurally implicit place (SIP). An SIP is a place whose row in the incidence matrix can be obtained as a non-negative linear combination of other rows of the incidence matrix. This property, essentially a structural property, implies that if we have the freedom to select the initial marking for these places, we can make them implicit places. Then, they can be removed from the net maintaining the same set of occurrence sequences of transitions. The next result points out this structural property.

Lemma 7.3 [25] *Every resource place of an S^4PR net is structurally implicit.*

In fact, all rows of the incidence matrix in the aforementioned non-negative linear combination correspond to process places. This linear combination of process places is proven to be unique [25]. According to the theory of SIPs [12], if we are considering SIPs in structurally bounded nets, then we can compute a finite initial marking for an SIP making it an implicit place.

This apparently instrumental result allows us to obtain interesting conclusions. The first one says that we must analyse if the resources of an S^4PR net system are implicit or not, because if some of them are implicit, we can remove them and then we can simplify further analysis (e.g., less siphons) or even we can fall in a subclass where we can find stronger results that can be applied. The second conclusion says that we can conclude that these nets are structurally live and structurally bounded.

Lemma 7.4 [25] *Every S^4PR net is structurally live and structurally bounded.*

The earlier result is quite straightforward: the net is conservative; therefore, it is structurally bounded. All resource places are SIPs therefore, if we consider an initial marking where (1) all resource places have an initial marking making each one implicit and (2) all idle places contain at least one token, then the net is live. In effect, if the resource places are implicit, then we can remove them and the language of occurrence sequences remains unchanged. Therefore, the resulting net is composed by a set of strongly connected state machines, each one containing at least one token, and therefore each one is live. If our net is live, it is structurally live. Finally, observe that this kind of initial marking is an admissible initial marking.

The discussion is obviously relevant from a physical point of view. As intuition suggests, it is always possible to make the system live by incrementing the number of available resources at the system start-up. This assumption, however, works well globally, but unexpected behaviour can arise when not all resource places are considered. Indeed, liveness is not monotonic even for more simple types of RASs, such as those which can be modelled by means of the S^3PR class:

Definition 7.3 An S^3PR [5] is an S^4PR net where $\forall p \in P_S : |{}^{\bullet\bullet}p \cap P_R| = 1$ and $({}^{\bullet\bullet}p \cap P_R = p^{\bullet\bullet} \cap P_R)$. ■

In these models, it is assumed that the production plans consist of production stages in which one single resource is allocated each time. S^3PR nets have been widely exploited in the field of FMSs thanks to a constructive simplicity which does not prevent them from being applicable to many plant configurations, plus the fact of having a simpler structural liveness characterization, as discussed in the next subsection. In spite of the seeming simplicity of this kind of systems, the following negative result regarding liveness monotonicity applies:

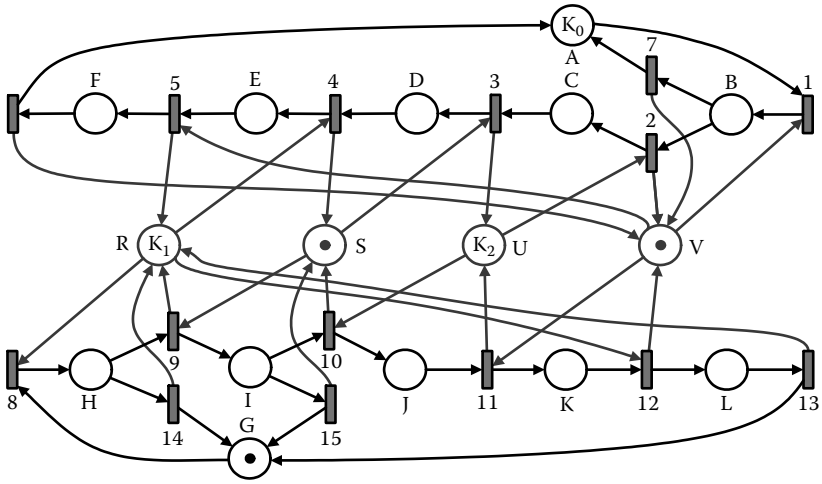

FIGURE 7.2 **(See colour insert.)** S^3PR net which is non-live iff $(K_0 \geq K_1, K_3 \geq 2) \vee (K_0 \cdot K_1 \cdot K_3 = 0)$. Note that $\mathbf{m_0}$ is an acceptable initial marking if $(K_0 \cdot K_1 \cdot K_3 \neq 0)$.

Property 7.1 *There exists an S^3PR net such that liveness is not monotonic, with respect to the marking of either the idle/process places or that of the resource places, that is liveness is not always preserved when those are increased.*

The net depicted in Figure 7.2 illustrates this fact:

- With respect to P_R: The system in Figure 7.2 is live with $K_0 = K_1 = K_3 = 1$ and non-live with $K_0 = K_1 = 1, K_3 = 2$ (however, it becomes live again if the marking of R, S and V is increased enough so as to make every resource place an implicit place).
- With respect to P_0: The system in Figure 7.2 is live with $K_0 = 1, K_1 = K_3 = 2$ and non-live with $K_0 = K_1 = K_3 = 2$.

Note that liveness is monotonic with respect to the marking of the resource places for every net belonging to the L-S^3PR class [11]. L-S^3PR is a subclass of S^3PR in which all production plans are composed of one single production path:

Definition 7.4 An *L-S^3PR net* [6] is an S^3PR net where $\forall p \in P_S : |{}^\bullet p| = |p^\bullet| = 1$. ■

L-S^3PR has some interesting properties derived from the aforementioned structural restriction [6,11]. Those include liveness monotonicity with respect to resource places. However, from S^3PR nets upwards, there is a discontinuity zone (i.e., a range of initial markings where the property is fluctuating) between the point where the resource places are empty enough so that every transition is dead (also held for lower markings) and the point where every resource place is implicit (liveness is preserved if their marking is increased). Markings within these bounds fluctuate between liveness and non-liveness. The location of those points also depends on the marking of the idle/process places: the more tokens in them, the farther the saturation point (i.e., the upper bound).

The forthcoming remark is straightforward from Definitions 7.3 and 7.4. It is worth noting that Definition 7.2 is consistent with the respective definitions of acceptable initial marking provided in the literature [5,6] for the aforementioned subclasses.

Remark 7.1 L-$S^3PR \subseteq S^3PR \subseteq S^4PR$.

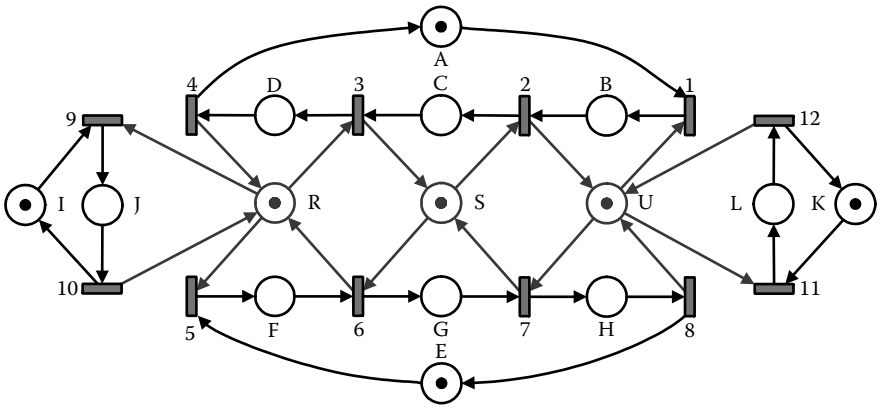

FIGURE 7.3 **(See colour insert.)** Non-live but deadlock-free L-S^{3}PR net system.

Nowadays, the most general superclass for modelling RASs with sequential processes is the S*PR class [8]. S*PR nets extend the expressive power of the processes to that of state machines, that is the process subnets are ordinary state machines with potentially intricate internal cycles. Although deadlock avoidance techniques have been proposed for this class, a structural liveness characterization is still lacking.

Finally, one of the characteristic properties of the S^{n}PR family is that deadlock freeness does not imply liveness [11]. In this sense, they are trickier than other well-known PN classes such as strongly connected free choice systems [16] or bounded strongly connected equal conflict systems [36], where both properties are equivalent. Figure 7.3 shows that this holds even for L-S^{3}PR nets.

Property 7.2 [11] *There exists an L-S^{3}PR net system with an acceptable initial marking such that it is deadlock-free but not live.*

This L-S^{3}PR net system has no deadlock but two reachable livelocks:

$$\text{Livelock1} \equiv \{[\mathrm{B}+\mathrm{G}+\mathrm{I}+\mathrm{K}+\mathrm{R}], [\mathrm{B}+\mathrm{G}+\mathrm{J}+\mathrm{K}]\}$$

$$\text{Livelock2} \equiv \{[\mathrm{C}+\mathrm{F}+\mathrm{I}+\mathrm{K}+\mathrm{U}], [\mathrm{C}+\mathrm{F}+\mathrm{I}+\mathrm{L}]\}$$

Nevertheless, these livelocks are captured by insufficiently marked siphons.

In terms of the system behaviour, this means that some processes can remain active while some others cannot progress any longer. This obviously hardens the liveness analysis.

7.2.3 Main Analysis Results

Traditionally, empty or insufficiently marked siphons have been a fruitful structural element for characterizing non-live RASs [5,31,37]. The more general the net class, however, the more complex the siphon-based characterization is. The following results can be easily obtained from previously published works. The originality here is to point out the strict conditions that the siphons must fulfil.

Theorem 7.1 [5] *Let $\langle \mathcal{N}, \mathbf{m_0} \rangle$ be an S^{3}PR net system with an acceptable initial marking. $\langle \mathcal{N}, \mathbf{m_0} \rangle$ is non-live iff $\exists \mathbf{m} \in RS(\mathcal{N}, \mathbf{m_0})$ and a minimal siphon $D : \mathbf{m}[D] = \mathbf{0}$.*

Although the previous theorem is originally enunciated for general empty siphons [5] (i.e., not explicitly minimal), it is obvious that Theorem 7.1 trivially holds from there. Indeed, suppose that the empty siphon D is not minimal. Then there must exist a minimal siphon $D' \subset D$. Since $\mathbf{m}[D'] = \mathbf{0}$, an empty minimal siphon exists. The sufficient part is straightforward.

For instance, the S^3PR net system in Figure 7.2 is non-live with $K_0 = K_1 = 1$, $K_3 = 2$. From this acceptable initial marking, the marking $[E + K + S + 2 \cdot U]$ can be reached by firing σ, where $\sigma = 1\ 8\ 2\ 9\ 3\ 10\ 4\ 11$. This firing sequence empties the siphon $\{B, H, F, L, R, V\}$.

However, this characterization is sufficient, but not necessary, in general, for S^4PR net systems. For instance, the S^4PR net system in Figure 7.1 is non-live but no siphon is ever emptied. Hence, the liveness characterization, and particularly the notion of *empty siphon*, had to be generalized.

The following theorem presents a liveness characterization for S^4PR nets. This characterization is fully behavioural. The structural causes for non-liveness are related to the existence of certain siphons and this will be presented later. An instrumental definition will be introduced first.

Definition 7.5 [38] Given a marking $\mathbf{m}$ in an S^4PR net system, a transition t is said to be

- $\mathbf{m}$-process-enabled ($\mathbf{m}$-process-disabled) iff it has (not) a marked input process place, that is $t \in (\|\mathbf{m}\| \cap P_S)^\bullet$ (i.e., $t \notin (\|\mathbf{m}\| \cap P_S)^\bullet$)
- $\mathbf{m}$-resource-enabled ($\mathbf{m}$-resource-disabled) iff its input resource places have (not) enough tokens to fire it, that is, $\mathbf{m}[P_R, t] \geq \mathbf{Pre}[P_R, t]$ (i.e., $\mathbf{m}[P_R, t] \ngeq \mathbf{Pre}[P_R, t]$)

■

Theorem 7.2 [38] *Let $\langle \mathcal{N}, \mathbf{m_0} \rangle$ be an S^4PR net system with an acceptable initial marking. $\langle \mathcal{N}, \mathbf{m_0} \rangle$ is non-live iff $\exists \mathbf{m} \in RS(\mathcal{N}, \mathbf{m_0})$ such that the set of $\mathbf{m}$-process-enabled transitions is non-empty and each one of these transitions is $\mathbf{m}$-resource-disabled.*

Theorem 7.2 relates non-liveness to the existence of a marking in which all active processes are blocked. Their output transitions need resources that are not available. These needed resources cannot be generated (released by the corresponding processes) by the system (the transitions are dead) because there exists a set of circular waits between the blocked processes.

This concept of circular waits is captured in the model by the existence of a siphon (in PN terms) whose resource places are the places preventing the firing of the process-enabled transitions. The following theorem characterizes non-liveness in terms of siphons establishing the bridge between behaviour and model structure.

Theorem 7.3 [38] *Let $\langle \mathcal{N}, \mathbf{m_0} \rangle$ be an S^4PR net system with an acceptable initial marking. $\langle \mathcal{N}, \mathbf{m_0} \rangle$ is non-live iff $\exists \mathbf{m} \in RS(\mathcal{N}, \mathbf{m_0})$ and a siphon D such that* (1) *there exists at least one $\mathbf{m}$-process-enabled transition,* (2) *every $\mathbf{m}$-process-enabled transition is $\mathbf{m}$-resource-disabled by resource places in D and* (3) *process places in D are empty at $\mathbf{m}$.*

Most analysis and control techniques in the literature are based on the computation of this structural element: the so-called *bad siphon*. A bad siphon (often also called *strict siphon* in the literature [22]) is a siphon which is not the support of a p-semiflow. If bad siphons become (sufficiently) emptied, their output transitions die since the resource places of the siphon cannot regain tokens anymore, thus revealing the *deadly embrace*. Many control techniques thus rely on the insertion of monitor places, that is controllers in the real system, which limit the leakage of tokens from the bad siphons.

Such a siphon D is said to be *insufficiently marked at* $\mathbf{m}$, generalizing the notion of empty siphon. In contrast to the S^3PR class, it is worth noting the following fact about *minimal* siphons in S^4PR net systems, which emerges because of their minimal p-semiflows not being strictly binary.

Property 7.3 *There exists an S^4PR net system with an acceptable initial marking which is non-live but every siphon characterizing the non-liveness is non-minimal, that is minimal siphons are insufficient to characterize non-liveness.*

For instance, the S^4PR net system in Figure 7.1 is non-live, but there is no minimal siphon containing both resource places R and S. Note that the siphon $D = \{R, S, D, G\}$ becomes insufficiently marked at $\mathbf{m}$, where $\mathbf{m} \equiv [B + F + R + S]$, but it contains the minimal siphon $D' = \{S, D, G\}$. D' is not insufficiently marked for any reachable marking. It is also worth noting that no siphon is ever emptied.

Thus, non-minimal siphons must be considered in order to deal with deadlocks in systems more complex than S^3PR.

7.3 STRUCTURAL DEADLOCK PREVENTION BASED ON THE ADDITION OF NEW VIRTUAL RESOURCES

7.3.1 Synthesis Flow for Liveness Enforcing

Traditionally, PN-based deadlock prevention techniques for RASs have been based on the exploitation of the siphon as a structural element that captures the causal essence of a deadlock through its emptying. As discussed in Section 7.2, this emptying of a siphon is manifested in absolute terms for the most simple RAS net classes (L-S^3PR, S^3PR), while for more complex types of systems (S^4PR), deadlocks are characterized through the concept of insufficiently marked siphon, in which some tokens may permanently remain in the siphon yet still being insufficient to allow an eventual firing of certain transitions.

Fruitful results have been obtained in the field of FMSs by exploiting such concept. Siphon inspection allows determining which system processes may block each other and because of which allocations of resources. This allows introducing monitors in the system which inhibit the possibility of the system reaching such undesired states. These controllers are naturally represented in the PN as new resource places introducing GMECs [15] that forbid certain firing sequences.

Most works that approach the problem of live FMSs synthesis from this perspective usually exploit the fact that these monitors are represented by resource places in the PN. For sufficiently general net classes for RASs, such as S^4PR, this means that the PN that models the resulting system after the addition of the monitor still belongs to the same class of nets. In this case, it is possible to approach the augmented model using the same tools. This has facilitated the introduction of iterative correction methods which address possible deadlocks in a gradual manner until obtaining a live system [38].

Often, each monitor that is added to the model is capable of removing a family of markings that belong to the same terminal strongly connected component, be it from the reachability space of the net or at least from the solution space of its net state equation. In terms of the net structure, this is achieved by monitoring the transitions that subtract tokens from the siphon, so that it is not possible to drain the siphon beyond a certain threshold from which it cannot regain its initial marking. In other words, these transitions which potentially empty the siphon will request tokens from the new monitor place and therefore always have a minimum of two input resource places, wherein at least one of them belongs to the siphon. That is why in the case of more restrictive subclasses such as S^3PR, the addition of monitor places takes us out of the class studied, since in them, each transition

has at most one input resource place (indeed, only one resource is allocated to each active process in S^3PR nets) [5].

On the other hand, the addition of the monitor may cause the appearance of new siphons that include this new resource place in them. Those new siphons can eventually be insufficiently marked, making it necessary to consider the need to control them. Behaviourally, this can be explained by the conversion of non-terminal strongly connected components that inevitably lead to deadlock into terminal strongly connected components that contain now actual deadlocks, having removed the leaf components that hung from them. Consequently, iterative approaches need to consider not only the siphons of the original net but also those new siphons which appear when others are being controlled.

Furthermore, this kind of techniques have a major handicap. Although the strategy of addition of monitors works fine for simple RAS classes, such as those that are structurally safe [13], such approaches have deeper problems in more general circumstances. It not only happens that sometimes there does not exist a single place to cut a terminal strongly connected component in a complete and isolated way, but occasionally the elimination of one of its markings through a monitor place requires removing legal markings from other strongly connected components. This happens even for RAS classes whose net models are ordinary and their processes are linear, that is, which lack online decisions [11]. Later in Section 7.3.3, we will see an example of this type of phenomenon.

This circumstance is due to the fact that monitor places introduce linear marking constraints (i.e., the weighted sum of tokens in a set of places is limited by some number), while some of those unwanted markings may be trapped in the convex hull of the space of legal markings. This means that (to make a clean excision) we need a non-linear constraint that can cut this kind of markings. In this context, Iordache and Antsaklis [19] propose the introduction of disjunctive constraints through control mechanisms that are no longer a single place but instead a subnet which overcomes the aforementioned obstacle. A major difficulty in this kind of approaches is, again, the fact that the subnet introduced to control the siphon may leave the resulting PN out of the class of networks that can be analysed with the classic theory of siphons in RASs.

Another common approach for liveness enforcing through the addition of monitors is based on the exploration and manipulation of the state space of the net, leaving aside siphons as a structural means of approaching non-liveness. In this context, it is essential that bad states are categorized, taking into account not only those in which there exist dead transitions but also those markings which are doomed to deadlock. Nevertheless, the biggest obstacle that this kind of approaches manifest is to handle the state explosion problem, aggravated especially in highly concurrent RASs.

Due to the computational complexity of dealing with the reachability set, Garcìa-Vallés et al. [13] propose the use of symbolic representations of sets of markings through ordered binary decision diagrams (OBDDs) to obtain a compact representation of the set of forbidden markings. This ultimately allows the definition of a temporal logic and the efficient computation of the control logic (i.e., the set of monitors) through model checking techniques.

A different subfamily of approaches on the problem relies on the theory of regions (or variations of it) to tackle the state explosion problem and the addition of monitors that forbid bad states [14,39]. Nazeem and Reveliotis [30] address the classification of the reachability space through non-linear classifiers, tackling the problem of bad markings that become trapped in non-convex regions of legal markings.

Finally, there have been a set of techniques based on the structure of the Petri net as a graph. Typically, they consider the relations among resources and they look for dangerous relations, as circular waiting relations among resources. The Petri net structure typically reflects the relations of use among resources (which are the main problem when dealing with deadlocks) and, in this sense, several approaches have been presented trying to exploit this. One of the most common approaches is based on the usage of digraphs [10]. In this vein, Fanti et al. [9] approach the RAP in railway networks originally modelled by means of coloured Petri nets by way of digraphs. The resulting control logic is incorporated to the original net system in the form of coloured monitor places. Digraph-based deadlock avoidance policies are approached in Chapter 8.

7.3.2 Approaches Based on Siphon Computation

The approaches based on the relation among bad states and structural components of the PN have their starting point in [5]. That paper presented a characterization of the liveness problem in S^3PR in terms of empty siphons. Many papers have later tried to either extend those results to more general classes of PNs or to explore them [7,17,18,20,21,31,33,34,37,38].

A first family of approaches to the problem of synthesizing live FMSs from a structural perspective is based on the computation of all siphons that may be involved in a deadlock, in order to control them. In the context of S^3PR nets, this often results in the need for algorithms for computing the set of minimal siphons since, for this kind of models, any deadlock situation is associated with at least one minimal empty siphon. This is essentially the approach originally presented in the seminal work by Ezpeleta et al. [5].

Some work has been done later to avoid the computation of all minimal siphons, trying to reduce the computational complexity by only controlling a significant subset of them [22]. The core of this kind of approaches is to try to obtain truly independent siphons by means of its interactions via transitions (dependency is measured by linear combinations of transitions related to the siphon). Other works try to avoid the computation of all minimal siphons through the introduction of hybrid strategies in which the state space of the net is also explored [32].

The pruning graph [1] is a powerful artefact that not only allows the computation of every minimal siphon in a S^4PR net but also provides a concise insight on how these siphons are constructed. For S^3PR nets, minimal siphons are enough to characterize deadlock situations.

In this graph, nodes represent the minimal siphons containing a single resource place, while the arcs represent the pruning relations between those 'seed' siphons. Particularly, Cano et al. [1] prove that such *seed* siphons are unique and that every minimal siphon of an S^4PR net containing more than one resource place can be obtained by observation of the pruning relations between the set of minimal siphons containing each of its resource places in isolation. Note that two minimal siphons are in a pruning relation if their union contains places that are non-essential, that is there exists a place such that for all its output transitions, it holds that either they do not have input from the siphon or they do not return tokens to the siphon.

For more restrictive classes of nets, such as S^3PR or $SOAR^2$, it is even possible to obtain the set of minimal siphons by purely algebraic methods for the different combinations of the 'seed' minimal siphons [35]. But in general, the technique requires the search and identification of strongly connected subgraphs under certain conditions to ratify the minimality of the resulting siphon. The pruning graph will be revisited in Section 7.4 in the context of a different family of synthesis techniques for a subclass of S^4PR.

An alternative route for structural control based on siphons passes through the introduction of iterative synthesis strategies based on the individual search of potentially dangerous siphons and their control through monitors. This scheme is repeated until a live system is obtained, which sometimes means avoiding the computation of a significant number of siphons.

In this vein, several studies have appeared that address the computation of problematic siphons and monitor places through techniques based on ILP [17,37,38]. In the next subsection, we will discuss an approach based on this last family of techniques, which will be illustrated through examples.

7.3.3 Managing Siphons for the Computation of Virtual Resources

Tricas et al. [38] present an iterative algorithm for deadlock prevention based on ILP which is reviewed in the following. This innovative result is grounded on the structural characterization already presented in Theorem 7.3 and deployed in the context of supervisory control of FMSs for the rather general class of S^4PR nets. With the help of the net state equation, a set of ILP problems (ILPPs) can be constructed which prevents the costly exploration of the state space. As far as we know, other works on ILP-based liveness enforcing depart from a similar strategy even though the

objective function to be optimized may differ [22]. To illustrate the algorithm, the S^{4}PR net system in Figure 7.1 will be used.

In each iteration, this algorithm searches for a bad siphon. If found, a control place is suggested to prevent that siphon from ever becoming insufficiently marked. Such control place will be a virtual resource, in such a way that the resulting PN remains into the S^{4}PR class. Thanks to this, a new iteration of the algorithm can be executed. The algorithm terminates as soon as there do not exist more siphons to be controlled, that is the system is live.

Prior to the introduction of the algorithm and its related ILPPs, some basic notation must be established.

In the following, for a given insufficiently marked siphon D, $D_{\mathrm{R}} = D \cap P_{\mathrm{R}}$ and $\mathbf{y}_{D_{\mathrm{R}}} = \sum_{r \in D_{\mathrm{R}}} \mathbf{y}_r$. Notice that $\mathbf{y}_{D_{\mathrm{R}}}$ expresses the total amount of resource units belonging to D (in fact, to D_{R}) used by each active process in their process places. Also, $\mathbf{sb}[p]$ denotes the structural bound* of p [4]. Finally the following holds:

Definition 7.6 [38] Let $\langle \mathcal{N}, \mathbf{m_0} \rangle$ be an S^{4}PR net system. Let D be a siphon of $\mathcal{N}$. Then, $\mathcal{T}h_D = \|\mathbf{y}_{D_{\mathrm{R}}}\| \setminus D$ is the *set of thieves* of D, that is the set of process places of the net that use resources of the siphon and do not belong to that siphon. ■

The next system of restrictions relates the liveness characterization introduced in Theorem 7.2 with the ILPPs which are used in the forthcoming algorithm. Essentially, the structural characterization is reformulated into a set of linear restrictions given a reachable marking and a related bad siphon.

Proposition 7.1 [38] *Let $\langle \mathcal{N}, \mathbf{m_0} \rangle$ be an S^{4}PR net system. The net is non-live if and only if there exist a siphon D and a marking $\mathbf{m} \in \mathrm{RS}(\mathcal{N}, \mathbf{m_0})$ such that the following set of inequalities has, at least, one solution:*

$$
\begin{cases}
\mathbf{m}[P_{\mathrm{S}}] \gneq \mathbf{0} & & \text{-- } \exists t \in T : t \texttt{ is } \mathbf{m}\texttt{-process-enabled} \\
\forall t \in T \setminus P_0{}^{\bullet} : & \textit{with } \{p\} = {}^{\bullet}t \cap P_{\mathrm{S}}, & \\
 & \mathbf{m}[p] \geq e_t & \text{-- } e_t\texttt{=0 if } t \texttt{ is } \mathbf{m}\texttt{-process-disabled} \\
 & e_t \geq \frac{\mathbf{m}[p]}{\mathbf{sb}[p]} & \text{-- } e_t\texttt{=1 if } t \texttt{ is } \mathbf{m}\texttt{-process-enabled} \\
\forall r \in D_{\mathrm{R}}, \forall t \in r^{\bullet} \setminus P_0{}^{\bullet} : & \frac{\mathbf{m}[r]}{\mathbf{Pre}[r,t]} \geq e_{rt} & \text{-- } e_{rt}\texttt{=0 if } t \texttt{ is } \mathbf{m}\texttt{-resource-disabled by } r \\
 & e_{rt} \geq \frac{\mathbf{m}[r] - \mathbf{Pre}[r,t] + 1}{\mathbf{m_0}[r] - \mathbf{Pre}[r,t] + 1} & \text{-- } e_{rt}\texttt{=1 if } t \texttt{ is } \mathbf{m}\texttt{-resource-enabled by } r \\
\forall r \in P_{\mathrm{R}} \setminus D_{\mathrm{R}}, \forall t \in r^{\bullet} \setminus P_0{}^{\bullet} : & e_{rt} = 1 & \text{-- } e_{rt}\texttt{=1 if } r \notin D \\
\forall t \in T \setminus P_0{}^{\bullet} : & \sum_{r \in {}^{\bullet}t \cap P_{\mathrm{R}}} e_{rt} < |{}^{\bullet}t \cap P_{\mathrm{R}}| + 1 - e_t & \text{-- } \texttt{if } t \texttt{ is } \mathbf{m}\texttt{-process-enabled then} \\
 & e_t \in \{0,1\} & \qquad t \texttt{ is } \mathbf{m}\texttt{-resource-disabled by } D_R \\
\forall r \in D_{\mathrm{R}}, \forall t \in r^{\bullet} \setminus P_0{}^{\bullet} : & e_{rt} \in \{0,1\} &
\end{cases}
\tag{7.1}
$$

The following proposition introduces a set of additional restrictions on the system (7.1) that characterize the condition of siphon for the set of places whose respective variables v_p equal zero (observe that, for notational simplicity, we use v_p for process places and v_r for resource places). Note that the minimality of the siphon is not required, which makes sense considering that no minimal siphon characterizes liveness for the class of S^{4}PR nets, as introduced in Section 7.2. Therefore, the new proposition captures the characterization introduced in Theorem 7.3 with a system of linear inequalities.

* sb[p] is the max. of the following ILPP: $sb[p] = \max \mathbf{m}[p]$ s.t. $\mathbf{m} = \mathbf{m_0} + \mathbf{C} \cdot \boldsymbol{\sigma}, \mathbf{m} \geq \mathbf{0}, \boldsymbol{\sigma} \in \mathbb{N}^{|T|}$.

Proposition 7.2 [38] *Let $\langle \mathcal{N}, \mathbf{m_0} \rangle$ be an S^{4}PR net system. The net is non-live if and only if there exist a siphon D and a marking $\mathbf{m} \in \mathrm{RS}(\mathcal{N}, \mathbf{m_0})$ such that the following set of inequalities has a solution with $D = \{p \in P_\mathrm{S} \cup P_\mathrm{R} \mid v_p = 0\}$:*

$$
\begin{cases}
\forall p \in P \setminus P_0, \forall t \in {}^\bullet p : v_p \geq \sum_{q \in {}^\bullet t} v_q - |{}^\bullet t| + 1 & \text{-- } D \texttt{ is a siphon} \\
\sum_{p \in P \setminus P_0} v_p < |P \setminus P_0| & \text{-- } |D| \texttt{ > 1} \\
\mathbf{m}[P_\mathrm{S}] \gneq \mathbf{0} & \text{-- } \exists t \in T : t \texttt{ is } \mathbf{m}\texttt{-process-enabled} \\
\forall t \in T \setminus P_0{}^\bullet : & \textit{with } \{p\} = {}^\bullet t \cap P_\mathrm{S}, \\
\qquad \mathbf{m}[p] \geq e_t & \text{-- } e_t\texttt{=0 if } t \texttt{ is } \mathbf{m}\texttt{-process-disabled} \\
\qquad e_t \geq \frac{\mathbf{m}[p]}{\mathbf{sb}[p]} & \text{-- } e_t\texttt{=1 if } t \texttt{ is } \mathbf{m}\texttt{-process-enabled} \\
\forall r \in P_\mathrm{R}, \forall t \in r^\bullet \setminus P_0{}^\bullet : \ \frac{\mathbf{m}[r]}{\mathbf{Pre}[r,t]} + v_r \geq e_{rt} & \text{-- } e_{rt}\texttt{=0 if } t \texttt{ is } \mathbf{m}\texttt{-res.-disab. by } r \texttt{ and } r \in D \\
\qquad e_{rt} \geq \frac{\mathbf{m}[r] - \mathbf{Pre}[r,t] + 1}{\mathbf{m_0}[r] - \mathbf{Pre}[r,t] + 1} & \text{-- } e_{rt}\texttt{=1 if } t \texttt{ is } \mathbf{m}\texttt{-resource-enabled by } r \\
\qquad e_{rt} \geq v_r & \text{-- } e_{rt}\texttt{=1 if } r \notin D \\
\forall t \in T \setminus P_0{}^\bullet : \ \sum_{r \in {}^\bullet t \cap P_\mathrm{R}} e_{rt} < |{}^\bullet t \cap P_\mathrm{R}| + 1 - e_t & \text{-- } \texttt{if } t \texttt{ is } \mathbf{m}\texttt{-process-enabled then} \\
\forall p \in P \setminus P_0 : \quad v_p \in \{0, 1\} & \qquad t \texttt{ is } \mathbf{m}\texttt{-resource-disabled by } D_R \\
\forall t \in T \setminus P_0{}^\bullet : \quad e_t \in \{0, 1\} & \\
\forall r \in P_\mathrm{R}, \forall t \in r^\bullet \setminus P_0{}^\bullet : \quad e_{rt} \in \{0, 1\} &
\end{cases}
\tag{7.2}
$$

Thanks to the addition of the net state equation as another linear restriction, the following theorem constructs an ILPP which can compute a marking and a bad siphon holding system (7.2). Nevertheless, that marking can be a spurious solution of the state equation. Since this kind of nets can have killing spurious solutions (i.e., spurious solutions which are non-live when the original net system is live), then the theorem establishes a necessary but not sufficient condition. This is usually not a problem when the objective is to obtain a live system: the only consequence can be that some harmless, unnecessary control places are added. These control places would forbid some markings which are not really reachable.

Since one siphon must be selected, the ILPP selects that with a minimal number of places, hoping that controlling the smallest siphons first may prevent controlling the bigger ones. Other works present analogous techniques with a different objective function for this ILPP [17].

Theorem 7.4 [38] *Let $\langle \mathcal{N}, \mathbf{m_0} \rangle$ be an S^{4}PR net system. If the net is non-live, then there exist a siphon D and a marking $\mathbf{m} \in \mathrm{PRS}(\mathcal{N}, \mathbf{m_0})$ such that the following set of inequalities has, at least, one solution with $D = \{p \in P_\mathrm{S} \cup P_\mathrm{R} \mid v_p = 0\}$:*

$$
\begin{aligned}
&\max \textstyle\sum_{p \in P \setminus P_0} v_p \\
&s.t. \quad \mathbf{m} = \mathbf{m_0} + \mathbf{C} \cdot \boldsymbol{\sigma} \\
&\qquad\ \ \mathbf{m} \geq \mathbf{0}, \boldsymbol{\sigma} \in \mathbb{N}^{|T|} \\
&\qquad\ \ \text{System (2)}
\end{aligned}
$$

The previous theorem can compute a marking $\mathbf{m}$ and a related bad siphon D. However, siphon D can be related with a high number of deadlocks, and not only with that represented with $\mathbf{m}$. For that reason, the aim is to compute a control place able to cut every *unwanted* marking which the siphon D is related to. Consequently, two different strategies are raised from the observation of the set of unwanted markings: (1) adding a place that introduces a lower bound of the number of available resources in the siphon for every reachable marking (*D-resource place*), or (2) adding a place that introduces an upper bound of the number of active processes which are withdrawing tokens from the siphon (*D-control place*).

In order to define the initial marking of such places, two constants must be computed which are the result of two ILPPs. These ILPPs evaluate every unwanted marking that a bad siphon is related to:

Definition 7.7 [38] Let $\langle \mathcal{N}, \mathbf{m_0} \rangle$ be an S^{4}PR net system. Let D be an insufficiently marked siphon, and $m_D^{\max}$ and $m_D^{\min}$ are defined as follows, with $v_p = 0$ iff $p \in D$:

$$\begin{array}{rl} m_D^{\max} = \max & \sum_{r \in D_{\mathrm{R}}} \mathbf{m}[r] \\ s.t. & \mathbf{m} = \mathbf{m_0} + \mathbf{C} \cdot \boldsymbol{\sigma} \\ & \mathbf{m} \geq \mathbf{0}, \boldsymbol{\sigma} \in \mathbb{N}^{|T|} \\ & \mathbf{m}[P_{\mathrm{S}} \setminus \mathcal{T}h_D] = \mathbf{0} \\ & \text{System (1)} \end{array} \qquad \begin{array}{rl} m_D^{\min} = \min & \sum_{p \in \mathcal{T}h_D} \mathbf{m}[p] \\ s.t. & \mathbf{m} = \mathbf{m_0} + \mathbf{C} \cdot \boldsymbol{\sigma} \\ & \mathbf{m} \geq \mathbf{0}, \boldsymbol{\sigma} \in \mathbb{N}^{|T|} \\ & \mathbf{m}[P_{\mathrm{S}} \setminus \mathcal{T}h_D] = \mathbf{0} \\ & \text{System (1)} \end{array}$$

■

The next definition establishes the connectivity and the initial marking of the control place proposed for a given bad siphon D, both whether that place is a D-process place or a D-resource place.

Definition 7.8 [38] Let $\langle \mathcal{N}, \mathbf{m_0} \rangle$ be a non-live S^{4}PR net system. Let D be an insufficiently marked siphon and $m_D^{\max}$ and $m_D^{\min}$ as in Definition 7.7. Then, the associated D-resource place, p_D, is defined by means of the addition of the following incidence matrix row and initial marking: $\mathbf{C}^{p_D}[p_D, T] = -\sum_{p \in \mathcal{T}h_D} \mathbf{y}_{D_{\mathrm{R}}}[p] \cdot \mathbf{C}[p, T]$, and $\mathbf{m_0}^{p_D}[p_D] = \mathbf{m_0}[D] - (m_D^{\max} + 1)$. The associated D-process place, p_D, is defined by means of the addition of the following incidence matrix row and initial marking: $\mathbf{C}^{p_D}[p_D, T] = -\sum_{p \in \mathcal{T}h_D} \mathbf{C}[p, T]$, and $\mathbf{m_0}^{p_D}[p_D] = m_D^{\min} - 1$. ■

Finally, we can state the algorithm that computes the control places for a given S^{4}PR net system. In those cases in which a D-resource place with an admissible initial marking cannot be computed, the algorithm proposes the corresponding D-process place, which always has an admissible initial marking [38].

Theorem 7.5 [38] *Let $\langle \mathcal{N}, \mathbf{m_0} \rangle$ be an S^{4}PR net system. Algorithm* 7.1 *applied to $\langle \mathcal{N}, \mathbf{m_0} \rangle$ terminates. The resulting controlled system, $\langle \mathcal{N}^{\mathrm{C}}, \mathbf{m_0^C} \rangle$, is a live S^{4}PR net system such that* $\mathrm{RS}(\mathcal{N}^{\mathrm{C}}, \mathbf{m_0^C}) \subseteq \mathrm{RS}(\mathcal{N}, \mathbf{m_0})$.

Let us now apply Algorithm 7.1 to the net depicted in Figure 7.1. There exists one deadlock ($\mathbf{m} \equiv [\mathrm{B} + \mathrm{F} + \mathrm{R} + \mathrm{S}]$) and two insufficiently marked siphons in $\mathbf{m}$, $D_1 = \{\mathrm{R}, \mathrm{S}, \mathrm{D}, \mathrm{G}\}$ and $D_2 = D_1 \cup \{\mathrm{C}\}$. None of these is minimal. When applied step 1 of Algorithm 7.1, the ILPP of Theorem 7.4 returns $D = D_1$, since D_1 has less places than D_2. In step 2, we compute $m_D^{\max} = \mathbf{m}[\mathrm{R}] + \mathbf{m}[\mathrm{S}] = 2$.

Algorithm 7.1 [38] Synthesis of live S^{4}PR net systems

1. Compute an insufficiently marked siphon using the ILPP of Theorem 7.4.
2. Compute $m_D^{\max}$ (Definition 7.7).
   (a) If the associated D-resource place (Definition 7.8) has an acceptable initial marking according to Definition 7.2, then let p_D be that place, and go to step 3.
   (b) Else, compute $m_D^{\min}$ (Definition 7.7). Let p_D be the associated D-process place (Definition 7.8).
3. Add the control place p_D.
4. Go to step 1, taking as input the partially controlled systems, until no insufficiently marked siphons exist.

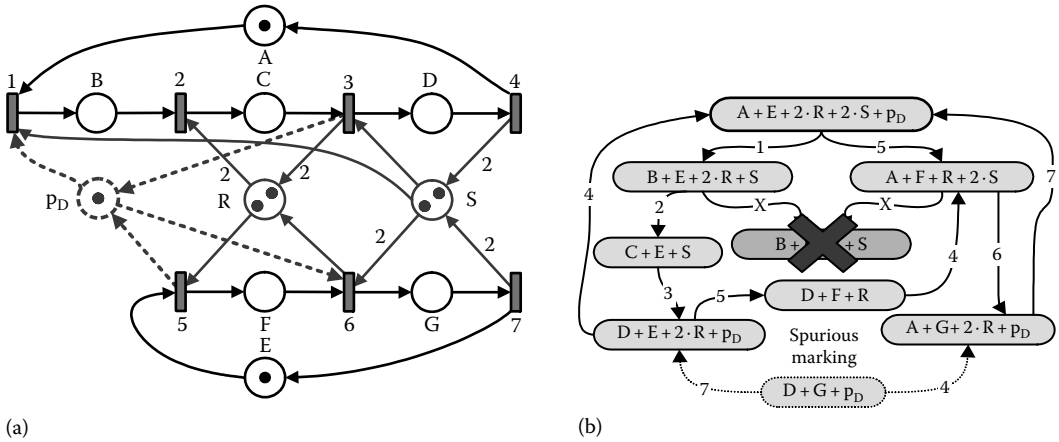

FIGURE 7.4 **(See colour insert.)** (a) The controlled system after applying Algorithm 7.1 on the net in Figure 7.1. (b) Reachability graph of the net in (a).

Since the associated D-resource place has not an acceptable initial marking (only one token in it is insufficient at $\mathbf{m_0}$), then we compute $m_D^{\min} = \mathbf{m}[B] + \mathbf{m}[C] + \mathbf{m}[F] = 2$. In step 3, we add the associated D-process place p_D to the net. And finally, we go back to step 1. But now the net is live and the ILPP of Theorem 7.4 has no solution, so the algorithm finishes after its first iteration. The resulting controlled system is depicted in Figure 7.4.

However, these techniques have certain limitations. For general S^{4}PR nets, adding extra monitors to cut 'bad' states off and enforce liveness may also entail the removal of some legal markings. This happens when some unwanted marking can be obtained by a linear combination of two reachable markings which are legal (i.e., those which are not doomed to deadlock and therefore ideally should not be forbidden). Then the unwanted marking is trapped in the convex hull of the space of legal markings. That means that any additional linear constraint that could be added to effectively forbid that marking necessarily would involve removing some legal extreme points. In other words, any GMEC which eliminates the unwanted marking also requires removing some legal marking.

This type of limitation is addressed by García-Vallés et al. [13] from the standpoint of general PN; in fact, the net in Figure 7.5 is a variation of an illustrative example presented in that work. Hereinafter, we will address the correction of this net system by means of Algorithm 7.1. This net contains a unique (minimal) bad siphon $D = \{\mathrm{R, S, C, F}\}$ which is insufficiently marked at the following reachable markings:

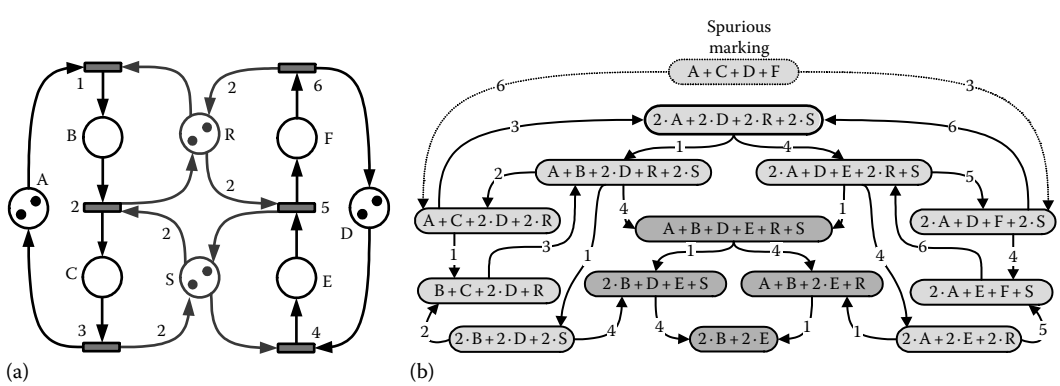

FIGURE 7.5 **(See colour insert.)** (a) A non-live S^{4}PR net system with a non-convex permissible marking space. (b) Reachability graph of the net in (a).

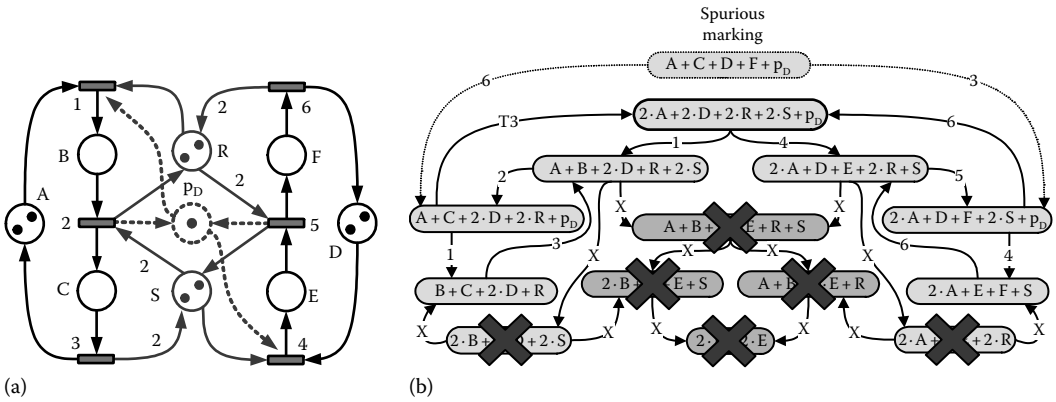

FIGURE 7.6 **(See colour insert.)** (a) The controlled system after applying Algorithm 7.1 on the net in Figure 7.5. (b) Reachability graph of the net in (a).

$$[\mathrm{A}+\mathrm{B}+\mathrm{D}+\mathrm{E}+\mathrm{R}+\mathrm{S}] \equiv \mathbf{m_1}$$

$$[2\cdot\mathrm{B}+\mathrm{D}+\mathrm{E}+\mathrm{S}]$$

$$[\mathrm{A}+\mathrm{B}+2\cdot\mathrm{E}+\mathrm{R}]$$

$$[2\cdot\mathrm{A}+2\cdot\mathrm{D}] \equiv \mathbf{m_2}$$

Indeed, the siphon D is empty at $\mathbf{m_2}$, which is a deadlock. Note that the other three markings are doomed to deadlock and therefore should be forbidden as well. When applied step 1 of Algorithm 7.1, the ILPP of Theorem 7.4 returns D. In step 2, we compute $m_D^{\max} = 2$ (this value is obtained because of $\mathbf{m_1}$). In step 3, we add the associated D-resource place p_D to the net. And finally, we go back to step 1. But now the net is live and the ILPP of Theorem 7.4 has no solution, so the algorithm terminates. The resulting controlled system is depicted in Figure 7.6.

As shown in Figure 7.6, the proposed solution includes the addition of a resource place p_D cutting the four unwelcome markings that are shaded in a different colour in both figures. However, this place also forbids two additional markings which do not inevitably lead to deadlock: $[2\cdot\mathrm{B}+2\cdot\mathrm{D}+2\cdot\mathrm{S}]$ and $[2\cdot\mathrm{A}+2\cdot\mathrm{E}+2\cdot\mathrm{R}]$. Note that $\mathbf{m_0}$ is still reachable from those markings.

This last observation might suggest that the new resource place is not the optimal so as to cut the unwanted states off. However, it is easy to check that the marking $[\mathrm{A}+\mathrm{B}+\mathrm{D}+\mathrm{E}+\mathrm{R}+\mathrm{S}]$, which inevitably leads to deadlock, lies in a midpoint between the two legal markings which have been eliminated. Certainly,

$$\frac{[2\cdot\mathrm{B}+2\cdot\mathrm{D}+2\cdot\mathrm{S}]+[2\cdot\mathrm{A}+2\cdot\mathrm{E}+2\cdot\mathrm{R}]}{2} = [\mathrm{A}+\mathrm{B}+\mathrm{D}+\mathrm{E}+\mathrm{R}+\mathrm{S}]$$

Therefore, any linear constraint added to remove such marking inevitably requires the elimination of any of those two legal markings.

This example also illustrates another limitation of these techniques: by adding linear constraints that merely inhibit reachable markings, and therefore possible firing sequences, the system concurrence is reduced. In many cases, this negatively affects the system performance. The proposed control logic depicted in Figure 7.6 not only inhibits completely the possibility that a process of type 'left' (i.e., a non-idle token on the left process subnet) and a process of type 'right' (i.e., a non-idle token on the right process subnet) ever coexist in the system, but even the circumstances under which two processes of the same type concur are reduced.

7.3.4 Non-Redundant Virtual Resources

A possible undesired consequence of using this type of approach is that of obtaining a set of places wherein some of them may be redundant for the control of the net, in the sense that their removal would preserve the language of firing sequences of the net. In other words, some of the control places may be implicit places.

As introduced in Lemma 7.3, every resource place of an S^4PR net is structurally implicit. Therefore, every monitor place added to prevent a siphon from becoming insufficiently marked is also an SIP. This does not necessarily mean that these places are implicit, as this depends on their initial marking. In fact, usually the last resource place added with this type of iterative approach is not an implicit place. This is due to the fact that it must cut at least one bad marking that was allowed so far. Provided that at least one such marking is not a spurious marking, we can state that the introduction of the monitor place restricts the language of firing sequences of the net, and thus the place is not implicit.

However, the introduction of a new resource place, even not being implicit, can make redundant (i.e., implicit) some of the previously existing monitor places in the net [37]. From the standpoint of the control engineer, this is an undesirable situation that can lead to the introduction of unnecessary control mechanisms involving a cost overrun or overcomplicate the system maintenance.

For this reason, post-processing techniques should be applied to cut implicit places after obtaining the control logic. Despite the NP-completeness of determining, in the general case, the minimum initial marking that makes an SIP implicit [12], there still exist efficient techniques based on sufficient conditions for determining if a control place with a given initial marking is implicit. García-Vallés and Colom [12] propose a technique based on linear programming which allows to determine it with a polynomial time cost in the worst case.

Such kind of techniques can be used iteratively on the set of control places in order to eliminate redundancies. In the worst case, this involves solving the earlier linear programming problem at most as many times as control places exist in the net, which still implies a cost in polynomial time in the size of the net.

7.3.5 Putting All Together

Throughout this section, an overview on some cutting-edge approaches on structural-based synthesis techniques based on the addition of monitors has been presented, with a detailed look on an iterative technique based on ILP.

This kind of techniques have their origin in the classic control theory, in the sense that the objective is to constrain the behaviour of the FMS by monitoring the system so as to forbid unwanted states or event sequences. Such strategies are especially valuable when trying to strictly comply with the original design of the FMS, that is the production plans are fixed, and it is not possible or desirable to increase the number of system resources or privatize their use.

Nonetheless, such an approach still presents some disadvantages. One obvious drawback exists on the fact that by constraining the language of event occurrences (i.e., firing sequences in the net model), the system concurrency is reduced. This can be detrimental to the overall system performance.

Another metric that can be degraded is the resource utilization rate. In effect, since the control logic prevents some resources from being used under certain circumstances (i.e., when a deadlock may occur), the involved resources may result underused in the long term.

Another problem which was illustrated in Section 7.3.3 is that the strategy of adding linear restrictions through monitors to enforce liveness may inevitably imply that some legal behaviour is prevented from happening. This was already illustrated through the net in Figures 7.5 and 7.6.

Finally, the computation cost of computing all siphons that can be problematic so as to control them (and remove the redundant control logic) can be especially demanding for systems in which there exist a high number of types of resources.

In the next section, we present a different family of structural synthesis techniques which somehow may transgress some of the principles of classic control theory but can provide a different category of solutions which can be complementary in certain scenarios.

7.4 STRUCTURAL DEADLOCK PREVENTION BASED ON THE ADDITION OF COPIES OF A RESOURCE TYPE

7.4.1 Principles of the Technique

In the previous section, we have reviewed the structural synthesis methods directly derived from the characterization of the liveness property. In this context, the reason for the non-liveness appearing is in the bad siphons with a small number of tokens inside. The synthesis strategy consists of the control of the siphon in such a way that the number of tokens inside the siphon always is greater than or equal to a value guaranteeing the liveness of the transitions covered by the siphon. Therefore, the basis of the strategy is a control-based strategy where we introduce extra constraints to the transitions allocating new copies of the resources of the siphon that prevent that the number of these copies go under a dangerous value. Given the iterative nature of the method, the constraints are constructed as new (virtual) resources of the system whose availability represents the constraints to the allocation of the original resources. This procedure allows us to maintain the net controlled in an intermediate iteration within the class of nets for which a siphon-based characterization of the liveness property exists, and so we can proceed with the next iteration.

We must point out here that these techniques adhere to some common rules or characteristics. The infringement of such rules gives rise to the new techniques introduced later in this section. The rules or characteristics shared by those techniques based on virtual resources are as follows:

- The controlled net respects the original state machines (the production plans) and the initial marking of the state machines (the initial marking of the idle places). For the designer, this means that the original system is completely maintained and so he/she can identify his/her original design.
- The same happens with respect to the original resources of the net system. The controlled net contains the same resource types (resource places) as in the original net, and the initial marking of these resource places is the same (the number of copies of each type of original resource is maintained).
- Liveness enforcing by means of virtual resources is based on the addition of some places that represent additional constraints to the transitions in charge of the allocation of some original resources when a request of a process (a token in a process place) arrives to the controller of the resource. The effect of these additional virtual resources is to forbid some occurrence sequences of transitions of the original net to prevent an excessive decay in the marking of a bad siphon. In other words, it is the classical approach inside control theory, where the goal is to forbid states or occurrence sequences. Therefore, the constraints reduce the concurrency inside the system because some occurrence sequences are forbidden and the resource utilization rate is reduced because some states where resources are in use are forbidden.

Observe that these three characteristics are shared by all techniques inspired in the use of siphons. Nevertheless, in this section, for a particular subclass of nets, we present a technique in which the three previous characteristics are not respected. That is, we propose a technique where the central point is the addition of copies of the original resources. It is well known that in bounded systems, if we are able to increase the number of copies of resources, then all deadlock problems disappear because the resource places become implicit places and can be removed. Nevertheless, we try to add a minimum number of copies of resources because they can be expensive; then in order to minimize

this, we specialize the original copies of a type of resource in the sense that certain copies only can be used by a subset of processes and the other copies of resources are used only by the rest of processes. This idea will be implemented by splitting a type of resource (a resource place) into two new types of resources each one used in a private way by a disjoint subset of processes of the original system. Therefore, we increase the number of copies of the original resources but increasing the degree of privatization of the use (or, in other words, reducing the degree of resource sharing). We will see that this technique, from a structural point of view, breaks the original bad siphons, and then if we are able to break all bad siphons, the net cannot have deadlocks.

At our best knowledge, this strategy has not been considered until the PhD thesis of Rovetto [35]. The positive effect in the global behaviour of the system is that concurrency is not reduced. In fact, all original states of the system remain reachable and new states are added representing the recovery states from the deadlocks to the safe states of the system. Therefore, these techniques are not based on the forbidden state strategy coming from control theory.

In the following, the non-live S^3PR net depicted in Figure 7.7a is used to illustrate the different techniques of liveness enforcing presented in this work and the different effect produced by each one in the behaviour of the corrected net. The net in Figure 7.7a is non-live because of the reachable marking $[B + E]$ (a total deadlock) that appears in the reachability graph depicted on the right of the PN. Figure 7.7b illustrates the result of the application of the liveness enforcing technique presented in Section 7.3.3. In effect, in this net, there exists the bad siphon $\{C, F, R, S\}$ that becomes empty at the marking $[B + E]$. Therefore, from this siphon and the initial marking of the net, the method computes the virtual resource place CP1 depicted in the figure with an initial marking equal to 1. This virtual resource (constraint) produces the removal of the deadlock marking $[B + E]$ as the reachability graph on the right of the figure points out. In other words, CP1 forbids the reachability of the state $[B + E]$.

In this section, we propose the increase in the number of copies of some original resource place. A first approach would advocate for increasing in one unit the initial marking of one of the resources involved in the formation of the deadlock. In this case, the two resources R and S are involved in the deadlock because they belong to the unique minimal bad siphon of the net: $\{C, F, R, S\}$. In order to enforce liveness, the resource R is selected, and one extra token is added to the initial marking, that is, one extra copy of the resource type R is added to the system as it is illustrated in Figure 7.7c. The effects on the behaviour of the corrected model is that all states of the original net are maintained in the corrected net or at least we can state a bijection between the two sets of states. The other effect is that new states appear in the reachability graph allowing to recover the system from the deadlock state to a safe state. The result is that the net becomes live. Nevertheless, the reader can observe that with this technique, the bad siphons of the original net persist in the corrected net model, and then from a structural point of view, the problem has not been fixed. In effect, if you increase the initial marking of the idle place A making it equal to two tokens, then the net system becomes non-live.

The method proposed in this section is illustrated by means of the net in Figure 7.7d. We increase the number of copies of resources, but in order to fix the problem at a structural level, this extra copy is of a resource type different to the pre-existing copy. In other words, the resource type R is split into two different resource types R1 and R2, each one containing one copy (one copy more than in the original situation with only the resource type R), and each one of these resource types is used in a private way from only one process plan. The resulting behaviour is very similar to the previous one: the original states are maintained and new states appear allowing to recover from the deadlock state to a safe state. Nevertheless, the reader can observe that with this approach the split of R has broken the original bad siphon and now there are not bad siphons in the net. This means that if we have an admissible initial marking in the net, the net will be live.

Finally, it is interesting to observe that the techniques based on the addition of copies of resources allow to maintain or even increase the existing concurrency in the original net. The other interesting effect is that the method of virtual resources constrains the allocation of resources sequentializing

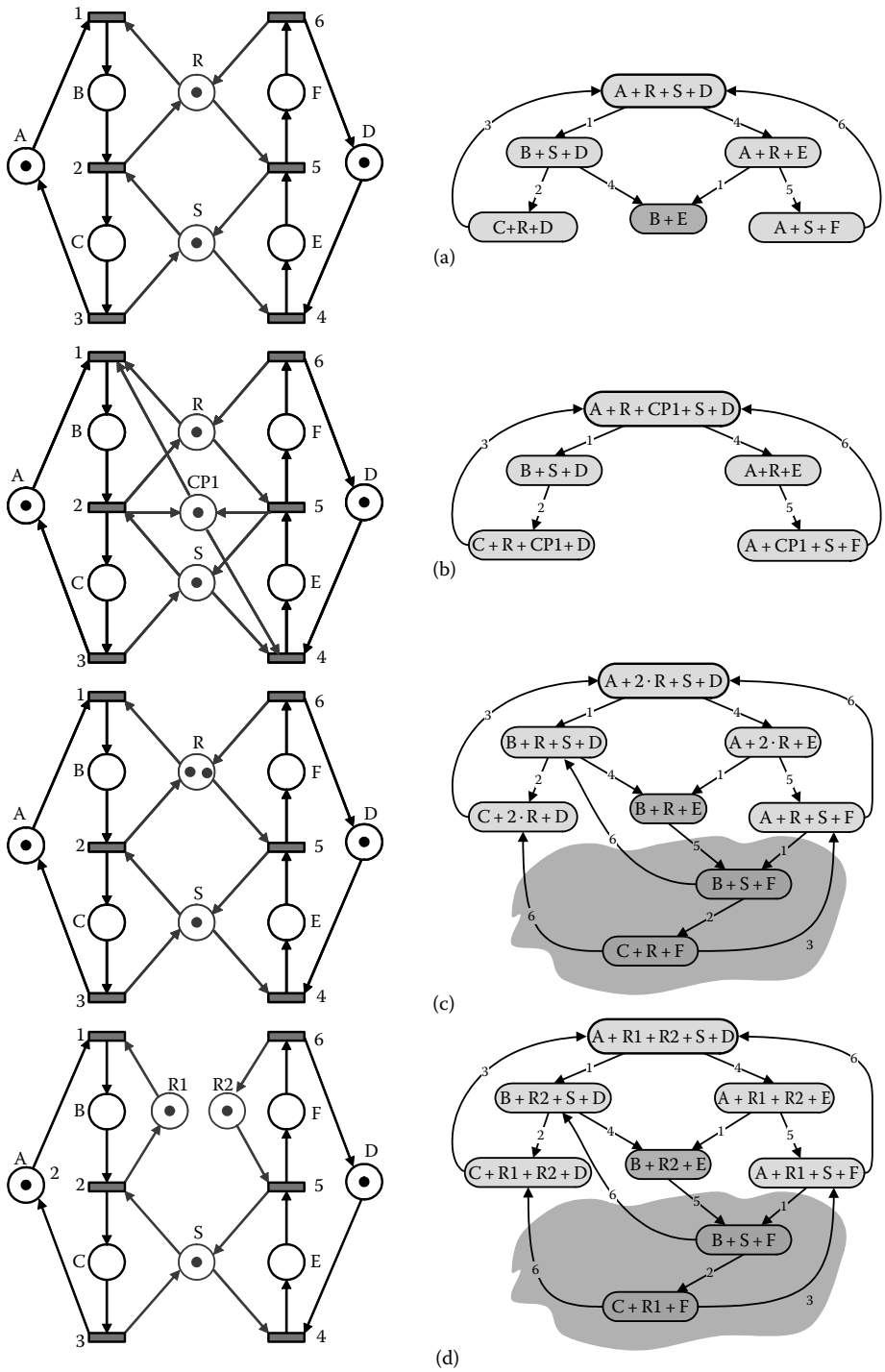

FIGURE 7.7 **(See colour insert.)** (a) S^3PR net which is non-live. (b) The net in (a) enforced to be live by the addition of the virtual resource CP1 computed from the bad siphon $\{C, F, R, S\}$. (c) The net in (a) that becomes live by the addition of an extra copy of the type of resource R. (d) The net in (a) that becomes live by splitting the resource type R into two new resource types, R1 and R2, and each one is used in a private way by one of the process plans.

the processes and therefore the resource utilization ratio is lower than in the case of the additional copies of existing resources.

The implementation of this technique requires the analysis of the structure of the net identifying the zones in the net where a copy of a resource is used in a continuous way. These zones are the candidates to use the copies of the resource they need in a private way. In the following section, in order to illustrate the approach, we introduce a subclass of S^4PR nets named $SOAR^2$ nets that were introduced in [35] to model minimal* adaptive routing algorithms of AGV systems. In this class of nets, we define the structural objects named zones (in fact structural regions of the net where a copy of a type of resource is used in a continuous way), the relations between the different zones and the relations of the zones with the siphons of the net. The latter allow, to identify the zones where a privatization of the use of the copies of some resources would break all the siphons of the net. Taking into account that the method will maintain the corrected net inside the class of the S^4PR nets, and the corrected net has an admissible initial marking, then the net will be live.

7.4.2 $SOAR^2$ Class of Nets and Related Properties

The $SOAR^2$ class of nets is a strict subclass of the S^4PR nets. The acronym stands for '*S^4PR with ordered allocation and release of resources*'. This class was introduced in [35] to model minimal adaptive routing algorithms of objects or data between a source station/node and a destination station/node throughout a system for the guidance of automated vehicles or an interconnection network composed of communication channels. Therefore, the constraints imposed to define the class from the S^4PR class come from the application domain and are justified by the application domain.

The main properties characterizing this subclass with respect to the general class of S^4PR are the following:

- The order in which the resources (rails or channels) are requested and allocated following a given route is the same order in which these resources are released.
- A process in an intermediate step can have allocated simultaneously a set of resources.
- In a change of the state of a process (occurrence of a transition), only one single operation of allocation or release over a unique copy of a type of resource can be executed. In other words, it is not possible to have concurrency between operations of allocation/release of copies of resources.
- For each type of resource, there is only a unique copy of resource belonging to this type of resource.

The translation of the previous characteristics to PN terms related to the superclass of S^4PR gives rise to the following structural constraints and to some constraints for the admissible initial markings valid for these nets:

1. In each circuit of each state machine of the net, the resources are released in the same order that they were allocated.
2. In each transition of the net, it is only possible to find a unique resource place connected to the transition. If the arc connecting the place inputs in the transition, this represents an allocation operation. If the arc connecting the place outputs from the transition, this represents a release operation.
3. A net of the $SOAR^2$ class is an ordinary PN, that is the weight of all arcs is equal to one.
4. The admissible initial marking of all resource places in $SOAR^2$ nets is equal to 1.

* A routing is considered minimal if the number of nodes between source and destination is the lowest possible.

In the earlier constraints on the class of S^4PR to obtain the $SOAR^2$ class, the three last constraints are more or less easy to understand and to manage. Nevertheless, the first constraint requires the introduction of some new objects named the *zone of continuous use of a copy of resource* or simply *resource zone*. The properties of these structural objects individually considered and the relations defined over the set of resource zones are the basis of the new deadlock prevention technique intuitively introduced in the previous subsection. Because of the lack of space, we will introduce these concepts with the aid of some examples.

A zone j of continuous use of the resource r in the ith state machine of the $SOAR^2$ net is a maximal set of process places that are holders of the resource r, $Z^r_{i,j}$, such that the subnet generated by the zone and its input and output transitions is a connected state machine. In the left $SOAR^2$ net of Figure 7.8, it is possible to find the following set of resource zones: $Z^R_{1,1} = \{B, C, D\}$, $Z^S_{1,1} = \{C, D, E\}$ and $Z^U_{1,1} = \{D, E, F\}$. In a similar way, the zones of the right $SOAR^2$ net of Figure 7.8 are $Z^R_{1,1} = \{B, C\}$, $Z^S_{1,1} = \{C, D, E\}$ and $Z^U_{1,1} = \{E, F\}$. For a resource r, we can find zones that belong to different state machines, and obviously they will be disjoint because they belong to disjoint state machines. It is also possible to find, for a resource r, several zones belonging to the same state machine. Obviously, in this last case, the zones must be disjoint. Another important property is that the idle place cannot

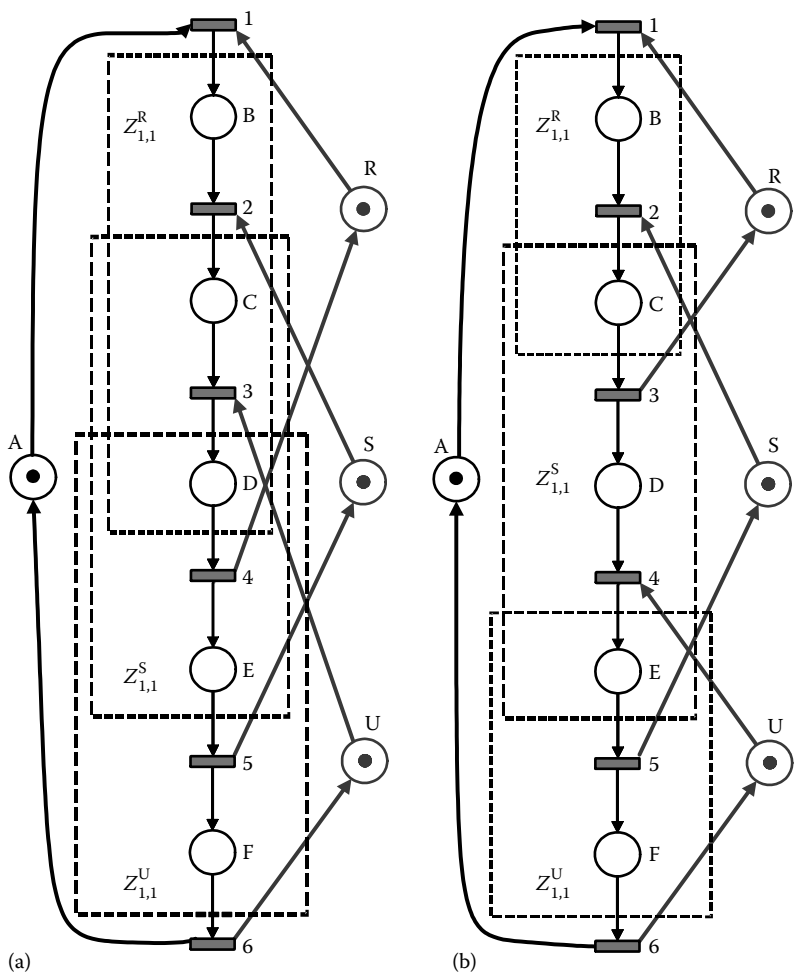

FIGURE 7.8 **(See colour insert.)** (a) and (b): Two $SOAR^2$ nets, each one containing a unique state machine showing the resource zones contained.

belong to any zone because the idle place cannot be a holder place of a resource (recall that the SOAR^2 class is a strict subclass of the S^{4}PR class of nets). The set of all resource zones of a net will be denoted as $\mathcal{Z}$, and the set of zones associated to a resource r will be denoted as $\mathcal{Z}^r$.

The first important relation between zones is the overlapping relation between resource zones. We say that a set of zones $A \subseteq \mathcal{Z}$, $A \neq \emptyset$ is a set of overlapping zones if and only if $\bigcap_{Z \in A} Z \neq \emptyset$. That is, the set of places belonging to the intersection of a set of zones are simultaneously holder places of all resource places associated to the zones of the set. In other words, the places of the intersection are process places and a token inside one of these places (a process) is using a copy of each resource associated to the zones of the set. Therefore, the computation of the maximal sets of overlapping zones is important for the study of the ordering relations in the allocation and release of the set of resources that simultaneously are held by a process inside one of the places in the intersection. For example, for the left net in Figure 7.8, there is only one maximal set of resource zones: $\{Z^{\mathrm{R}}_{1,1}, Z^{\mathrm{S}}_{1,1}, Z^{\mathrm{U}}_{1,1}\}$. Observe that $Z^{\mathrm{R}}_{1,1} \cap Z^{\mathrm{S}}_{1,1} \cap Z^{\mathrm{U}}_{1,1} = \{\mathrm{D}\}$. In other words, when we have a token inside the place D, the process is using simultaneously the resources R, S and U. Observe that in order to reach this situation, the allocation of this resources happened in the order $\mathrm{R} - \mathrm{S} - \mathrm{U}$. The moving of the token outside the set of overlapping zones releases the three resources in the same order: the condition imposed for SOAR^2 nets. However, in the right net in Figure 7.8, there are two maximal sets of overlapping resource zones: $\{Z^{\mathrm{R}}_{1,1}, Z^{\mathrm{S}}_{1,1}\}$ and $\{Z^{\mathrm{S}}_{1,1}, Z^{\mathrm{U}}_{1,1}\}$. In these two maximal sets, the allocation order of the resources is respected in the release process.

The order relation in the allocation of resources and its further release induces an order relation on the set of resource zones that is called pruning relation between zones. Taking into account that it is a total order relation in a maximal set of overlapping zones, we are only interested in the maximal chains of ordered elements in the set. This order relation can be represented by means of a graph. In Figure 7.9, the reader can find a SOAR^2 net and in the first column of the table the maximal sets of overlapping zones. The second column represents the maximal chains of ordered elements in the set. This ordering relations can be represented in a graphical way in the pruning graph of resource zones.

Observe that the union of all zones associated to a same resource gives as result the set of holders of the resource that together with the resource allow to obtain the support of the minimal p-semiflow associated to the resource. The agglomeration, in the pruning graph of resource zones, of all zones of a same resource in a single node allows to obtain the pruning graph of resources used to characterize the siphons of S^{4}PR in [1]. This agglomeration operation in the pruning graph of the resource zones is illustrated in Figure 7.10. The left figure represents the pruning graph of the resource zones, wherein the nodes inside the shadowed boxes correspond to the zones associated to the same resource and are the nodes to be fused in a single node. Therefore, each box gives rise to a node of the right figure in Figure 7.10. The arcs in the right figure have their origin in the order relation (pruning relation) between two zones associated to different resources but overlapped.

We must recall that a necessary condition to obtain a siphon containing a given set of resources is that the subgraph of the pruning graph containing the nodes associated to the considered resources and the arcs among them is a strongly connected subgraph. Therefore, if we transform our net in such a way that the resulting pruning graph is acyclic, then we cannot obtain siphons with more than one resource (i.e., the bad siphons of the net causing the non-liveness problems). That is, the acyclic pruning graph obtained after the transformation evidences that the liveness property has been enforced.

7.4.3 New Deadlock Prevention Policy Based on the Specialization of Resources

In this section, we present a new deadlock prevention technique for the previously introduced class of SOAR^2 nets. We have presented the advantages of the technique from the point of view of the maintenance (or even increase) of the concurrency of the system and the improvement of the resource utilization ratio in comparison with other deadlock prevention techniques. Nevertheless, there is another reason for the introduction of this technique: the implementation of the control places added to enforce liveness. In effect, in the methods presented in the previous section, the new

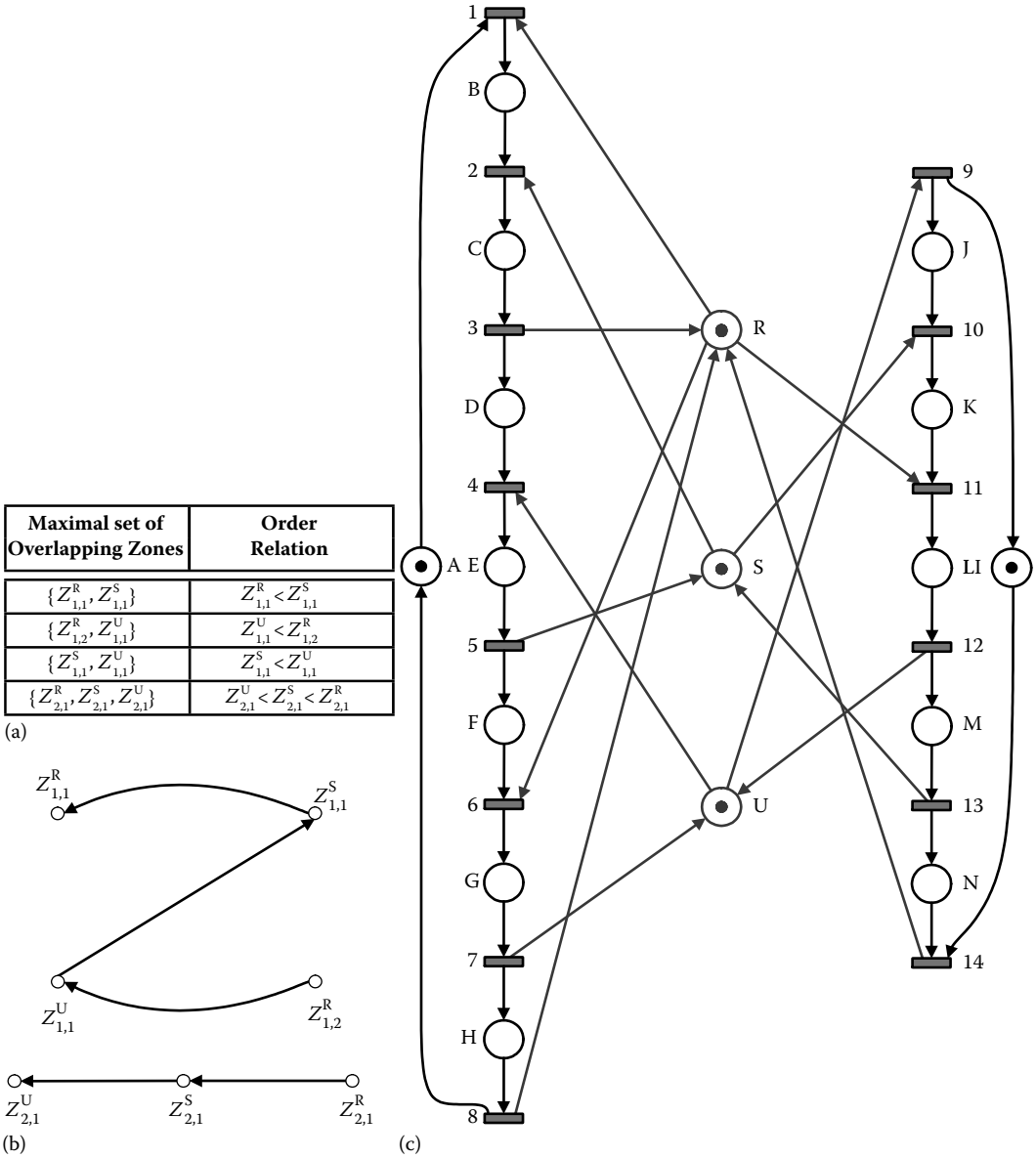

FIGURE 7.9 **(See colour insert.)** (a) Order relation between resource zones of the net in (c). (b) Maximal sets of overlapping resource zones of the net in (c). (c) A SOAR^2 net.

control place computed for the liveness enforcing property can be difficult to implement because our new virtual resource is used to control the number of tokens inside a siphon and the arcs of this virtual place need to be connected to several transitions that are not local. This non-locality of the transitions connected to the new resource place can give rise to new problems with respect to a distributable implementation. The technique we summarize in the following respects this locality principle because the area of intervention in order to enforce the liveness property is constrained to a resource zone and then the implementation issues related to distributability are solved.

The basis of the method to correct the model consists of the transformation of the PN in such a way that the pruning graph of resources of the transformed net becomes acyclic. Therefore, and according to the characterization of the minimal siphons of a S^{4}PR net on the pruning graph of resources, there

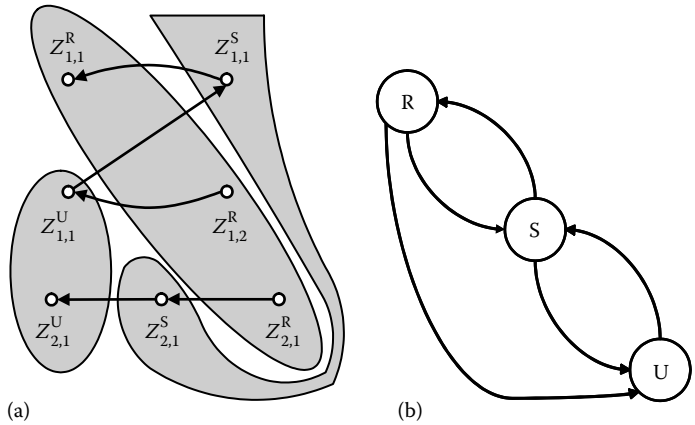

FIGURE 7.10 (a) Pruning graph of the resource zones of the SOAR^2 net in Figure 7.9. (b) Agglomerated pruning graph of the net.

exist no minimal siphons containing more than one resource since there are not strongly connected subgraphs containing more than one node. This is the basis of the method because in this way we enforce the net to be live since the PN has an admissible initial marking and there are not bad siphons in the net.

In order to reach this final goal, the designer must follow the steps stated in the following for SOAR^2 nets:

Step 1: *Construction of the pruning graph of the resource zones of the net.* The net in Figure 7.11, used as illustrative example in this subsection, is a non-live SOAR^2 net. In effect, the graph on the right hand of the figure is the pruning graph of resources that, as the reader can observe, contains one cycle. In this case, the cycle exists because of a bad siphon containing the resource places R and S: the minimal siphon $D = \{\mathrm{R}, \mathrm{S}, \mathrm{C}, \mathrm{D}, \mathrm{G}, \mathrm{H}\}$. This siphon is emptied by firing the transitions 1 and 5 from the initial marking. In general, the existence of cycles in the resource pruning graph does not characterize the existence of bad siphons, but if we can make acyclic the graph, then it is ensured that no bad siphon exists, and the resulting net is live. To accomplish this, we first must compute all the zones, and after that, we must compute the maximal set of overlapping zones. This lets us construct the pruning graph of resource zones, which contains four resource zones. The resulting graph is depicted in Figure 7.12.

Step 2: *Construction of the pruning graph of resources of the SOAR^2 net from the pruning graph of the resource zones.* By aggregating the resource zones which are shadowed in their pruning graph on the right hand of Figure 7.12, we obtain the pruning graph of resources which is depicted in Figure 7.11. As already explained in step 1, if the resulting pruning graph of resources is acyclic, then the net system is live for any acceptable initial marking and no changes are required. Otherwise, the pruning graph of resources provides valuable information on how liveness can be enforced.

Step 3: *Computation of a minimal number of arcs of the pruning graph of resources whose removal makes it acyclic.* Essentially, we must strive to select a minimal set of arcs whose removal requires as few changes as possible in the original net. This is often related to the intuitive idea of selecting a set of arcs which involve the smallest possible portion of the original net. Such information can be extracted from the pruning graph of the resource zones. In this vein, it must be remarked that a unique arc in the pruning graph of resources may map into several arcs of the pruning graph of resource zones. These arcs would connect different pairs of zones involving the same pair of resources. In some cases, the removal of some arc may require the duplication of some places

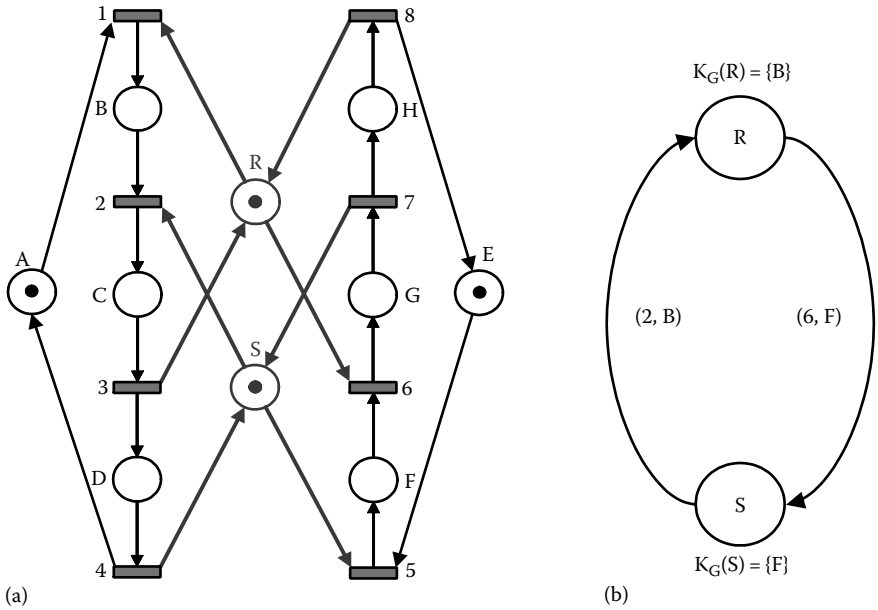

FIGURE 7.11 **(See colour insert.)** (a) A non-live SOAR^2 net. (b) Corresponding pruning graph of resources of the net.

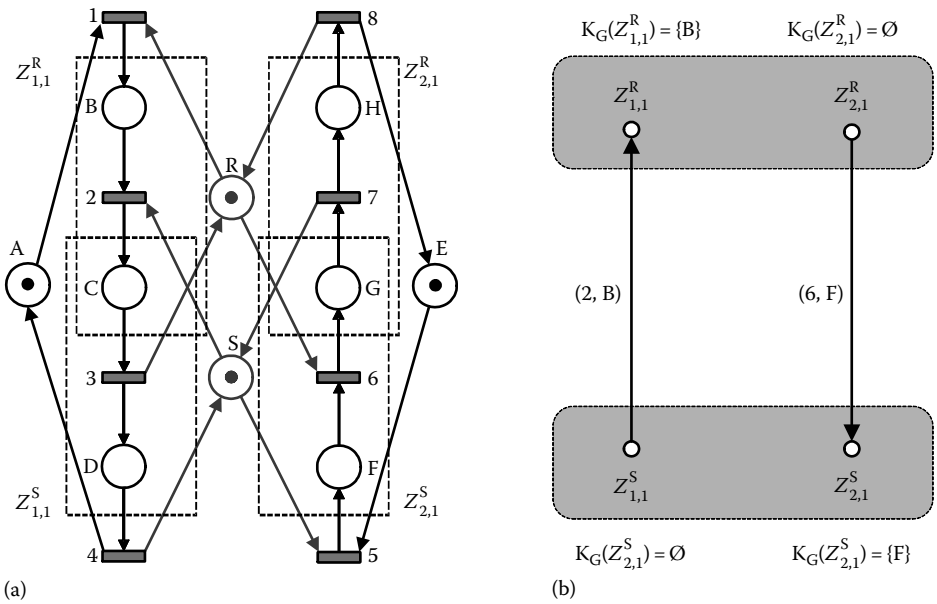

FIGURE 7.12 **(See colour insert.)** (a) Resource zones of the SOAR^2 net in Figure 7.11. (b) Corresponding pruning graph of resource zones of the same net.

and transitions, so a proper selection of the set of arcs is fundamental to avoid this. Due to the symmetry of the net of Figure 7.11, we can select any of the two arcs of the pruning graph of resources to make it acyclic. In this example, we select the arc drawn from S to R.

Step 4: *Removal of the arcs of the pruning graph of resources by the addition of virtual resources between the overlapping zones that generate these arcs.* Having obtained a candidate set of arcs, the net is transformed in order to incorporate the new resource places which break the siphons,

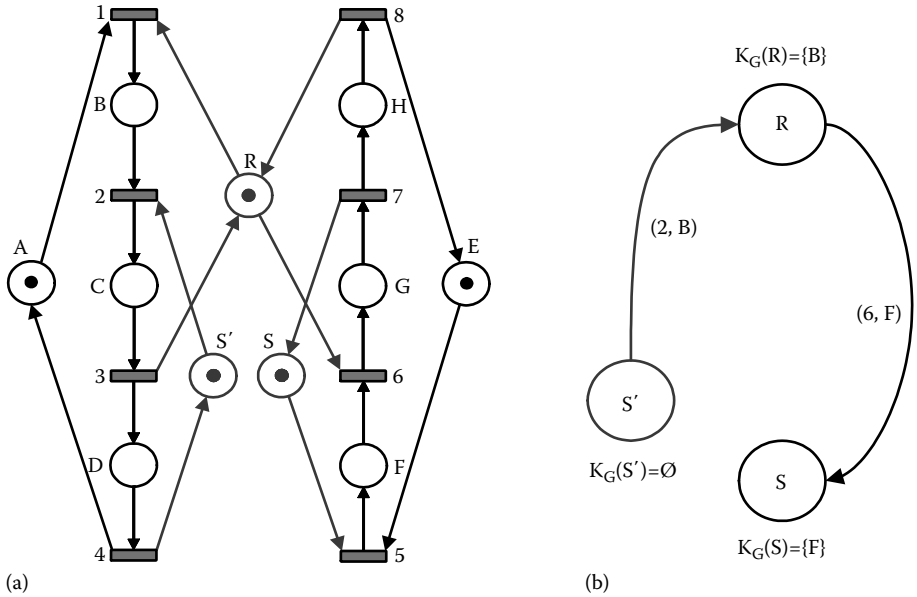

FIGURE 7.13 **(See colour insert.)** (a) A live SOAR^2 net obtained from the net in Figure 7.11 by splitting the resource S into two resource types used in a private way by each state machine. (b) Corresponding pruning graph of resources of the net.

and the pruning graph of resources is updated accordingly. In the example, the source node S in the selected arc is replaced by a new resource place S'. In the original net, this means that the resource place S is split in such a way that S' will be used in the old resource zone $Z^{S}_{1,1}$ (now: $Z^{S'}_{1,1}$) while S will still be used in the resource zone $Z^{S}_{2,1}$.

7.5 CONCLUSION

From the perspective of a system engineer, the synthesis techniques of live FMSs based on formal methods such as PNs are a powerful tool. These allow the system engineer to have efficient methods to correct deficiencies in the system in a fully automated and transparent manner. In this chapter, we presented a review of some cutting-edge synthesis techniques based on the structure of the RAS model of an FMS.

In particular, two structural deadlock prevention techniques based on PNs have been presented in greater depth. The first one stems from a standpoint which is closer to the classical approach in control theory. That is, the appearance of deadlocks is prevented through the inhibition of certain event occurrences leading to them. The latter is accomplished through the addition of monitors and entails a restriction on the behaviour of the system and on the concurrence of the production processes that coexist in it.

A second approach has been presented which has its origin in the techniques of deadlock prevention in AGV systems for the construction of deadlock-free minimal adaptive routing algorithms. This type of approach departs from the classical approach proposed in control theory, since it is assumed that the production plans can be retrofitted or that the resources in the system can be increased and privatized. By applying this kind of techniques, we obtain a local solution to the problems of deadlocks that does not degrade the system concurrency.

Although the two methodological approaches have been presented separately, these techniques are not mutually exclusive. Depending on the will of the system designer and the type of resources involved, one or another technique may or may not be applicable. Thus, in the FMS can coexist

resources that we are willing to increase and privatize, while for others it may just be possible (or more reasonable) using an approach closer to classical control theory. In this sense, further research must be done to properly integrate both perspectives into a cohesive methodology. This will be the subject of future work.

7.6 PETRI NET CONCEPTS AND NOTATIONS

We assume the reader is familiar with basic PN concepts. They can be found, for instance, in [29].

A PN (or place/transition net) is a three-tuple $\mathcal{N} = \langle P, T, W\rangle$ where P and T are two non-empty disjoint sets whose elements are called *places* and *transitions*. $W : (P \times T) \cup (T \times P) \longrightarrow \mathbb{N}$ is the *flow relation*: if $W(x, y) > 0$, then we say that there is an arc from x to y, with weight $W(x, y)$. Ordinary nets are those for which $W : (P \times T) \cup (T \times P) \rightarrow \{0, 1\}$. Given a net $\mathcal{N} = \langle P, T, W\rangle$ and a node $x \in P \cup T$, ${}^\bullet x = \{y \in P \cup T \mid W(y, x) > 0\}$, and $x^\bullet = \{y \in P \cup T \mid W(x, y) > 0\}$. An ordinary net will be called a *state machine* if and only if $\forall t \in T,\ |{}^\bullet t| = 1$ *and* $|t^\bullet| = 1$. A *marking* is a mapping $\mathbf{m} : P \longrightarrow \mathbb{N}$. The pair $\langle \mathcal{N}, \mathbf{m_0}\rangle$, where $\mathcal{N}$ is a net and $\mathbf{m_0}$ is an (initial) marking, is called a *net system*. A transition $t \in T$ is *enabled* for a marking $\mathbf{m}$ if and only if $\forall\ p \in {}^\bullet t,\ \mathbf{m}[p] \geq W(p, t)$; this fact will be denoted as $\mathbf{m} \xrightarrow{t}$; when fired (in the usual way), this gives a new marking $\mathbf{m}'$; this will be denoted as $\mathbf{m} \xrightarrow{t} \mathbf{m}'$. A marking $\mathbf{m}'$ is *reachable* from another marking $\mathbf{m}$ if and only if there exists a firing sequence $\sigma = t_1 t_2 ... t_n$ so that $\mathbf{m} \xrightarrow{t_1} \mathbf{m}_1 \xrightarrow{t_2} \mathbf{m}_2 \cdots \mathbf{m}_{n-1} \xrightarrow{t_n} \mathbf{m}'$ and is denoted as $\mathbf{m} \xrightarrow{\sigma} \mathbf{m}'$. The set of *reachable* markings from $\mathbf{m_0}$ in $\mathcal{N}$ is denoted as $RS(\mathcal{N}, \mathbf{m_0})$. A PN is self-loop-free if and only if $\neg(W(x, y) > 0 \wedge W(y, x) > 0)$. A self-loop-free PN $\mathcal{N} = \langle P, T, W\rangle$ can be alternatively represented as $\mathcal{N} = \langle P, T, \mathbf{C}\rangle$ where $\mathbf{C}$ is the net *flow matrix*: a $P \times T$ integer matrix so that $\mathbf{C} = P_0 st - \mathbf{Pre}$ where for each $t \in T, p \in P$, $P_0 st[p, t] = W(t, p)$ $\mathbf{Pre}[p, t] = W(p, t)$.

We consider structural analysis techniques based on linear algebra using a *linear relaxation* of the behaviour of a net system. Let $\langle \mathcal{N}, \mathbf{m_0}\rangle$ be a net system and σ a fireable sequence of transitions from $\mathbf{m_0}$. The (integer) linear relaxation looks as follows: $\mathbf{m_0} \xrightarrow{\sigma} \mathbf{m} \Rightarrow \mathbf{m} = \mathbf{m_0} + \mathbf{C} \cdot \overline{\sigma} \geq 0, \overline{\sigma} \geq 0$ where $\mathbf{m}$ is reachable from $\mathbf{m_0}$ firing σ, $\overline{\sigma}$ is the Parikh (or firing count) vector of σ and $\mathbf{C}$ the incidence matrix of the net, $\mathcal{N}$. This linear system is known as the *state equation* of the net system. Unfortunately, the reverse of the earlier implication is not true. More precisely, the state equation has integer solutions, $(\mathbf{m}, \sigma)$, not reachable on the net system. We call them *spurious* solutions.

A set of places D is a *siphon* if and only if ${}^\bullet D \subseteq D^\bullet$. A marked PN $\langle \mathcal{N}, \mathbf{m_0}\rangle$ is *deadlock-free* if and only if $\forall\ \mathbf{m} \in \mathrm{RS}(\mathcal{N}, \mathbf{m_0}),\ \{t \in T \mid \mathbf{m} \xrightarrow{t}\} \neq \emptyset$. A transition t is *live* if and only if $\forall\ \mathbf{m} \in \mathrm{RS}(\mathcal{N}, \mathbf{m_0}),\ \exists\ \mathbf{m}' \in \mathrm{RS}(\mathcal{N}, \mathbf{m})$ so that $\mathbf{m}' \xrightarrow{t}$. A marked net is live if and only if every transition is live. We will say that a transition t is *dead for a reachable marking* $\mathbf{m}$ if and only if there is no reachable marking from $\mathbf{m}$ that enables t. *Semiflows* are natural annullers of matrix $\mathbf{C}$. Right and left natural annullers are called *t-semiflows* ($\mathbf{x} \in \mathbb{N}^{|T|}$ such that $\mathbf{C} \cdot \mathbf{x} = 0$) and *p-semiflows* ($\mathbf{y} \in \mathbb{N}^{|P|}$ such that $\mathbf{y} \cdot \mathbf{C} = 0$), respectively. A net is said to be *conservative* (*consistent*) if there exists a p-semiflow $\mathbf{y} > 0$ (a t-semiflow $\mathbf{x} > 0$).

REFERENCES

1. E.-E. Cano, C.-A. Rovetto and J.-M. Colom. An algorithm to compute the minimal siphons in S^{4}PR nets. *Discrete Event Dynamic Systems*, 22(4):403–428, 2012.
2. F. Chu and X. Xie. Deadlock analysis of Petri nets using siphons and mathematical programming. *IEEE Transactions on Robotics and Automation*, 13(6):793–804, December 1997.
3. J.-M. Colom. The resource allocation problem in flexible manufacturing systems. In W.-M.-P. van der Aalst and E. Best, eds., *Proceedings of the 24th International Conference on Applications and Theory of Petri Nets*, vol. 2679 of *LNCS*, pp. 23–35, Eindhoven, the Netherlands, Springer–Verlag, June 2003.
4. J.-M. Colom and M. Silva. Improving the linearly based characterization of P/T nets. *Advances in Petri Nets 1990*, 483:113–145, 1991.

5. J. Ezpeleta, J.-M. Colom and J. Martínez. A Petri net based deadlock prevention policy for flexible manufacturing systems. *IEEE Transactions on Robotics and Automation*, 11(2):173–184, April 1995.
6. J. Ezpeleta, F. García-Valles and J.-M. Colom. A class of well structured Petri nets for flexible manufacturing systems. In J. Desel and M. Silva, eds., *Proceedings of the 19th International Conference on Application and Theory of Petri Nets*, vol. 1420 of *LNCS*, pp. 64–83, Lisbon, Portugal, Springer–Verlag, June 1998.
7. J. Ezpeleta and L. Recalde. A deadlock avoidance approach for non-sequential resource allocation systems. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 34(1), January 2004.
8. J. Ezpeleta, F. Tricas, F. García-Vallés and J.-M. Colom. A banker's solution for deadlock avoidance in FMS with flexible routing and multiresource states. *IEEE Transactions on Robotics and Automation*, 18(4):621–625, August 2002.
9. M.-P. Fanti, A. Giua and C. Seatzu. Monitor design for colored Petri nets: An application to deadlock prevention in railway networks. *Control Engineering Practice*, 14(10):1231–12472, October 2006.
10. M.-P. Fanti and M.-C. Zhou. Deadlock control methods in automated manufacturing systems. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 34(1):5–22, January 2004.
11. F. García-Vallés. Contributions to the structural and symbolic analysis of place/transition nets with applications to flexible manufacturing systems and asynchronous circuits. PhD thesis, University of Zaragoza, Zaragoza, Spain, April 1999.
12. F. García-Vallés and J.-M. Colom. Implicit places in net systems. In P. Bucholz and M. Silva, eds., *IEEE 8th International Workshop on Petri Nets and Performance Models (PNPM'99)*, pp. 104–113, Zaragoza, Spain, IEEE Computer Society Press, September 1999.
13. F. García-Vallés, F. Tricas, J.-M. Colom and J. Ezpeleta. Optimal control of discrete event systems. In *Proceedings of the 4th International Workshop on Discrete Event Systems (WODES'98)*, Cagliary, Italy, IEEE Control, IEEE, August 1998.
14. A. Ghaffari, N. Rezg and X. Xie. Design of a live and maximally permissive Petri net controller using the theory of regions. *IEEE Transactions on Robotics and Automation*, 19(1):137–142, February 2003.
15. A. Giua, F. DiCesare and M. Silva. Generalized mutual exclusion constraints on nets with uncontrollable transitions. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC'92)*, vol. 2, pp. 974–979, Chicago, IL, October 1992.
16. D. Hillen. Relationship between deadlock-freeness and liveness in free-choice nets. *Newsletter*, (19): 28–32, February 1985.
17. H. Hu, M.C. Zhou and Z.W. Li. Supervisor optimization for deadlock resolution in automated manufacturing systems with Petri nets. *IEEE Transactions on Automation Science and Engineering*, 8(4):794–804, October 2011.
18. Y.-S. Huang, M.-D. Jeng, X. Xie and D.-H. Chung. Siphon-based deadlock prevention policy for flexible manufacturing systems. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 36(6):1248–1256, 2006.
19. M.V. Iordache and P.J. Antsaklis. Petri net supervisors for disjunctive constraints. In *Proceedings of the 2007 American Control Conference*, pp. 4951–4956, New York, July 2007.
20. M.D. Jeng, X.L. Xie and S.L. Chung. ERCN* merged nets for modeling degraded behavior and parallel processes in semiconductor manufacturing systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 34(1):102–112, 2004.
21. M.D. Jeng, X.L. Xie and M.Y. Peng. Process nets with resources for manufacturing modeling and their analysis. *IEEE Transactions on Robotics*, 18(6):875–889, 2002.
22. Z.-W. Li and M.-C. Zhou. Control of elementary and dependent siphons in Petri nets and their application. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 38(1):133–148, January 2008.
23. Z.W. Li and M.C. Zhou. *Deadlock Resolution in Automated Manufacturing Systems: A Novel Petri Net Approach*. New York, Springer, 2009.
24. H. Liao, J. Stanley, Y. Wang, S. Lafortune, S.A. Reveliotis and S. Mahlke. Deadlock-avoidance control of multithreaded software: An efficient siphon-based algorithm for gadara petri nets. In *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC 2011), 2011 50th IEEE Conference on*, pp. 1142–1148, Orlando, FL, IEEE, December 2011.

25. J.-P. López-Grao. Contributions to the deadlock problem in multithreaded software applications observed as Resource Allocation Systems. PhD thesis, University of Zaragoza, Zaragoza, Spain, 2012.
26. J.-P. López-Grao and J.-M. Colom. Resource Allocation Systems: Some complexity results on the S^4PR class. In E. Najm, J.-F. Pradat-Peyre, and V.V. Donzeau-Gouge, eds, *Formal Techniques for Networked and Distributed Systems – FORTE 2006*, vol. 4229 of *LNCS*, pp. 323–338, Paris, France, Springer, 2006.
27. J.-P. López-Grao and J.-M. Colom. On the deadlock analysis of multithreaded control software. In *Proceedings of the 16th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'2011)*, Toulouse, France, IEEE, September 2011.
28. J.-P. López-Grao and J.-M. Colom. A Petri net perspective on the Resource Allocation Problem in software engineering. *Transactions On Petri Nets and Other models of Concurrency V (ToPNoC V)*, 6900:181–200, 2012.
29. T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, 1989.
30. A. Nazeem and S. Reveliotis. Designing compact and maximally permissive deadlock avoidance policies for complex resource allocation systems through classification theory: The nonlinear case. *IEEE Transactions on Automatic Control*, 57(7):1670–1684, July 2012.
31. J. Park and S.-A. Reveliotis. Deadlock avoidance in sequential resource allocation systems with multiple resource acquisitions and flexible routings. *IEEE Transactions on Automatic Control*, 46(10):1572–1583, October 2001.
32. L. Piroddi, R. Cordone and I. Fumagalli. Combined siphon and marking generation for deadlock prevention in Petri nets. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 39(3):650–661, May 2009.
33. S.-A. Reveliotis. *Real-Time Management of Resource Allocation Systems: A Discrete Event Systems Approach*. New York, Springer Science+Business Media, Inc., December 2004.
34. S.-A. Reveliotis. Implicit siphon control and its role in the liveness-enforcing supervision of sequential resource allocation systems. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 37(3):319–328, May 2007.
35. C.-A. Rovetto. Métodos basados en Redes de Petri para el diseño de algoritmos de encaminamiento adaptativos mínimos libres de bloqueos. PhD thesis, University of Zaragoza, Zaragoza, September 2011.
36. E. Teruel and M. Silva. Liveness and home states in equal conflict systems. In M. Ajmone Marsan, ed, *Proceedings of the 14th International Conference on Application and Theory of Petri Nets*, vol. 691, pp. 415–432, Chicago, IL, Springer-Verlag, 1993.
37. F. Tricas. Deadlock analysis, prevention and avoidance in sequential resource allocation systems. PhD thesis, University of Zaragoza, Zaragoza, Spain May 2003.
38. F. Tricas, F. García-Vallés, J.-M. Colom and J. Ezpeleta. A Petri net structure-based deadlock prevention solution for sequential resource allocation systems. In *Proceedings of the 2005 International Conference on Robotics and Automation (ICRA)*, pp. 272–278, Barcelona, Spain, IEEE, April 2005.
39. M. Uzam and M.-C. Zhou. An iterative synthesis approach to petri net-based deadlock prevention policy for flexible manufacturing systems. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 37(3):362–371, May 2007.
40. Y. Wang, H. Liao, S.A. Reveliotis, T. Kelly, S. Mahlke and S. Lafortune. Gadara nets: Modeling and analyzing lock allocation for deadlock avoidance in multithreaded software. In *Proceedings of the 49th IEEE Conference on Decision and Control*, pp. 4971–4976, Atlanta, GA, IEEE, December 2009.
41. X.L. Xie and M.D. Jeng. ERCN-merged nets and their analysis using siphons. *IEEE Transactions on Robotics and Automation*, 29(4):692–703, 1999.

FURTHER READING

Since the seminal work in which the class S^3PR was introduced, most PN models for RASs focus in the study of systems with sequential processes and on-line routing decisions [5,6,8,28,31,35,37,38]. This chapter was essentially devoted to the presentation of two different families of structural techniques for dealing with deadlocks in systems of this nature.

The reader can find some insight on the problem of integrating assembly/disassembly operations in RAS models in works such as [7,20,21,33,41]. In general, structural liveness enforcing approaches can be computationally demanding in this scenario, as evidenced for augmented marked graphs in [2]. An insight on the computational complexity of liveness enforcing on the RAS context is given in [26].

A complementary reading on the abstraction and modelling of FMSs as RASs by way of PN can be found in [3]. The books [23,33] also focus on the RAP from this perspective. Besides, the latter book also features some approximation to AGV transportation systems from the point of view of RAS modelling through PN. Recently, AGVs have been comprehensively tackled in [35]. Other application domains in which similar methodological approaches have been deployed include multiprocessor interconnection networks [35] and multithreaded software [24,27,28,40].