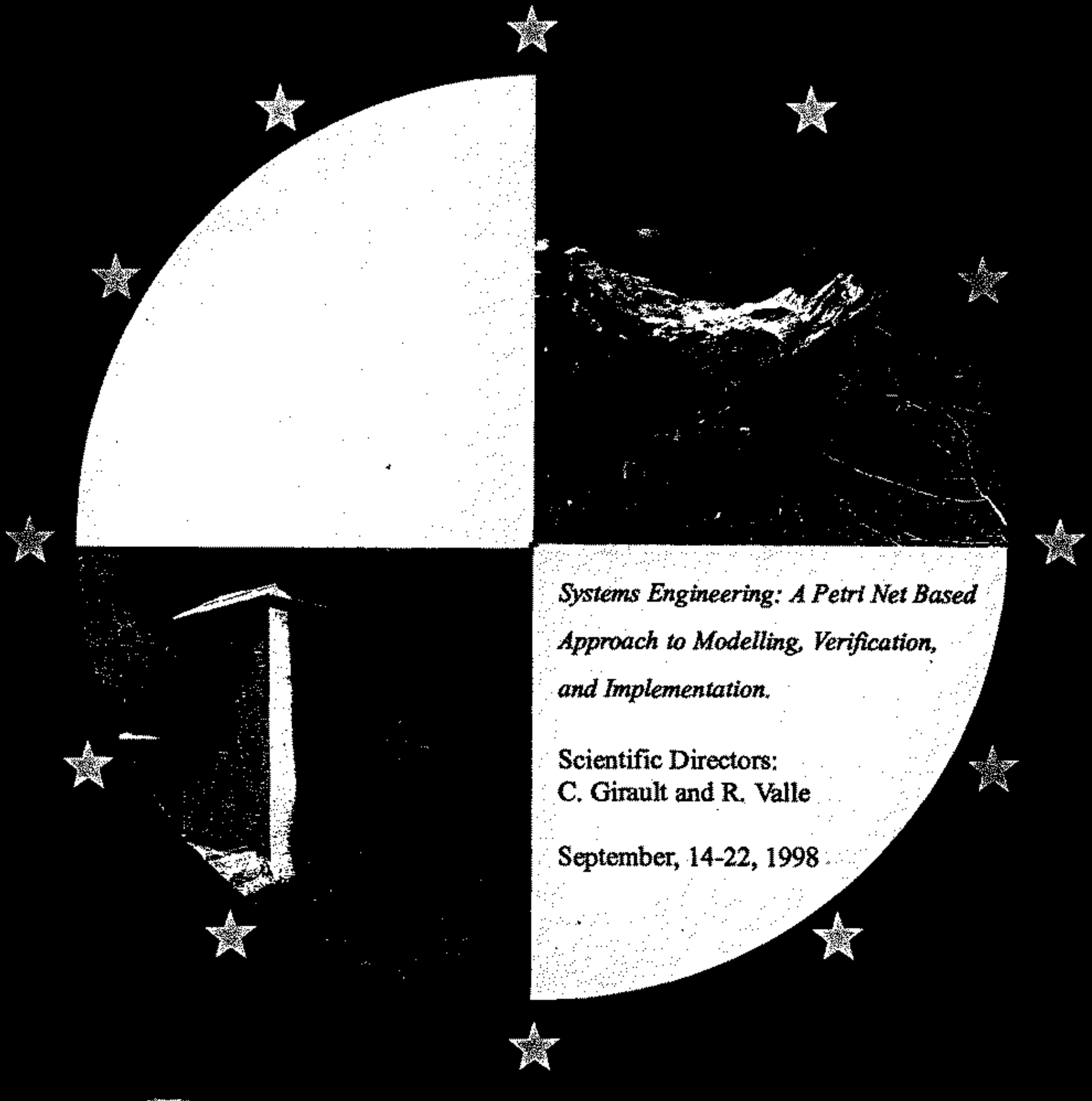
Human Capital and Mobility

MATCH

Advanced Schools



Systems Engineering: A Petri Net Based Approach to Modelling, Verification, and Implementation.

Scientific Directors:
C. Girault and R. Valle

September, 14-22, 1998

Residencia Universitaria de Jaca
Universidad de Zaragoza (Spain)

Systems Engineering: A Petri Net Based Approach to Modelling, Verification, and Implementation

MATCH
Human Capital and Mobility CHRX-CT94-0452

Draft for the Summer School
Jaca (Spain), 14-23 September, 1998

MATCH (Modelling and Analysis of Time Constrained and Hierarchical Systems) is a European Union Human Capital and Mobility initiative. Partners of this initiative that have contributed to this book are from the Universities of Eindhoven (W. Aalst, T. Basten, M. Graaf, L. Somers, M. Voorhoeve), Hamburg (B. Farwer, R. Mackenthun, D. Moldt, M. O. Stehr, R. Valk), Paris (P. Barril, A. Diagne, C. Dutheillet, M. P. Gervais, S. Haddad, J. M. Ilié, F. Kordon, D. Poitrenaud, I. Vernier), and Zaragoza (J. M. Colom, J. Ezpeleta, M. Silva, E. Teruel).

1ª Edición: Septiembre 1998

Depósito Legal: 2392/98

Imprime:
KRONOS
Menéndez Pelayo, 4
50009 Zaragoza

Contents

Chapter 16

Structural Methods

16.1 Introduction

A salient feature of concurrent systems is the complexity of their behaviour compared to the structure of the model. This is usually known as the *state space explosion problem*. More precisely, the associated transition systems, or reachability graph, of a given Petri net model is, when finite, much bigger than the net model itself. This fact disuades us from studying these systems by enumeration of their reachable states, although this is frequently the kind of analysis that is carried out in practice, what can limit the problems to be dealt with quite dramatically.

A fruitful approach to the problem has been the so called *structure theory*. The idea is to get useful information about the behaviour reasoning on the structure of the model. Structure theory investigates the relationship between the behaviour of a net system and its structure, i.e. the linear algebraic and graph theoretic objects and properties associated to the net and the initial marking. The study of this relationship usually leads to a deep understanding of the system. The ultimate goals of structure theory are usually phrased as the *analysis problem*, that is, trying to obviate the aforementioned state space explosion problem, developing analysis methods that not require the construction of the state space, and the *synthesis problem*, aiming at the design of refinement and composition operators that are known to preserve properties of interest. In this chapter we concetrate in the analysis problem inside the structure theory.

When general concurrent systems are considered, typical structural techniques give necessary or sufficient conditions on the studied properties. Nevertheless, the most satisfactory results are obtained when the scope is limited to restricted classes of systems and particular properties. The behaviour of general concurrent systems is of course richer, but sensible limitations lead to useful subclasses, both able to model some practical systems and to give insight on the relationships between the behaviour and the structure in more general ones. The typical restictions that are imposed aim at limiting the interplay between synchronisations and conflicts. On one hand, these restrictions facilitate the analysis. On the other, some modellin capabilities are lost. The designer must find a compromise between modelling power and availability of powerful analysis tools, while one of the theoretician's goals is obtaining better results for increasingly larger subclasses.

In this chapter two intimately related families of structural analysis techniques will be considered:

- *Graph Theory.* Objects like paths, circuits, handles, bridges, siphons, traps, etc. and their relationships are investigated. Tipically, only ordinary nets are considered, and the major results are obtained for particular properties, mainly boundedness, liveness, and reversibility, and restricted [4, 11, 19, 22, 46]

- *Linear Algebra and Convex Geometry.* These are techniques based on the state equation and/or the flows and semiflows. The semiflows can be used to prove properties like boundedness, mutual exclusion, liveness, etc. More generally, the state equation can be used as a basic description of the system in order to prove or disprove the existence of markings or firing sequences fulfilling some given conditions, eventually expressed as logic formulas [8, 18, 38]. Typically, results for general P/T net systems are obtained [8, 7, 29, 30, 31, 40, 42], some of which may become specially powerful when applied to restricted subclasses combining graph theory based arguments [17, 18, 20, 43].

To facilitate the analysis of a large and complex system it can be transformed (tipically reduced) preserving the properties to be analysed []. Transformation rules somehow preserve the behaviour while they are often supported by structural arguments as simple, and efficient, aufficient conditions. Net system reductions are presented in the next section with special emphasis in the implicit place concept.

16.2 Net system reductions

In order to paliate the state space explosion problem several techniques has been introduced to obtain *reduced state spaces.* As an example we can cite the stubborn set method [47, 48]. These techniques work directly in the construction phase of the reachability graph maintaining the original net model. In this section we review a different kind of reduction techniques named *net system reductions.* These reductions proceed transforming the net structure and, sometimes, the initial marking.

From an operational point of view, the approach is based on the definition of a kit or catalog of *reduction rules*, each one preserving a subset of properties (liveness, boundedness, reversibility, etc) to be analysed. A reduction rule characterises a type of subnet system (*locality principle*) to be substituted by another (simpler) subnet system.

The preconditions to be fulfilled have a *behavioural* and/or *structural* formulation. Behavioural preconditions can be more powerful for a given initial marking, but their verification is usually much more complex. So preconditions presented here are based on structural considerations and properties of the initial marking (i.e. the initial marking is considered as a parameter).

The design of a catalog of reduction rules is based on a tradeoff between completeness (i.e. transformation capabilities) and usefulness (i.e. applicability).

Given a catalog of reduction rules, analysis by reduction (the transformation procedure) is iterative by nature: Given the property (or properties) to be analyzed, the subset of rules that preserve it (them) is applied until the reduced system becomes irreducible. The irreducible system may be so simple that the property under study is trivially checked (see Figure 16.2.d). In other cases, the irreducible net is just "simpler" to analyse using another analysis technique (e.g. we can obtain a reduced state space on which it is possible to analyse the property that has been preserved in the reduction process). In other words, techniques to analyse net system models are complementary, not exclusive.

Reduction rules are transformation rules interesting for net analysis. When considered in the reverse sense they become expansion rules, interesting for net synthesis: stepwise refinements (or top-down) approach. Examples of this approach can be found in the context of synthesis of live and bounded Free Choice systems [20] or in the definition of subclasses of nets by the recursive application of classical expansion rules as the case of Macroplace/Macrotransition systems [16]. Using this approach, with adequate expansion rules, the model will verify by construction the specification. This is interesting when compared with the more classical approach based on the iteration of the design and analysis phases until the specification is satisfied. The iterative process has two basic disadvantages:

1) the lack of general criteria for modifying (correcting) a model which does not meet the requirements.

2) the operational difficulty inherent to the validation phase.

Nevertheless, since no kit of reduction rules is complete (i.e. able to fully reduce any system), it is not possible to synthesize an arbitrary system by such stepwise refinements.

A very basic kit of reduction rules is presented. Additional details are given only for the rule of implicit places, which are redundancies in the net system model: if an implicit place is removed, then (illusory) synchronizations disappear and other reduction rules can be applied.

16.2.1 A basic kit of reduction rules

Figure 16.1 presents graphically the structural and marking conditions for a kit of very particular cases of reduction rules. It is not difficult to observe that they preserve properties such as liveness and the bound of places (thus boundedness),

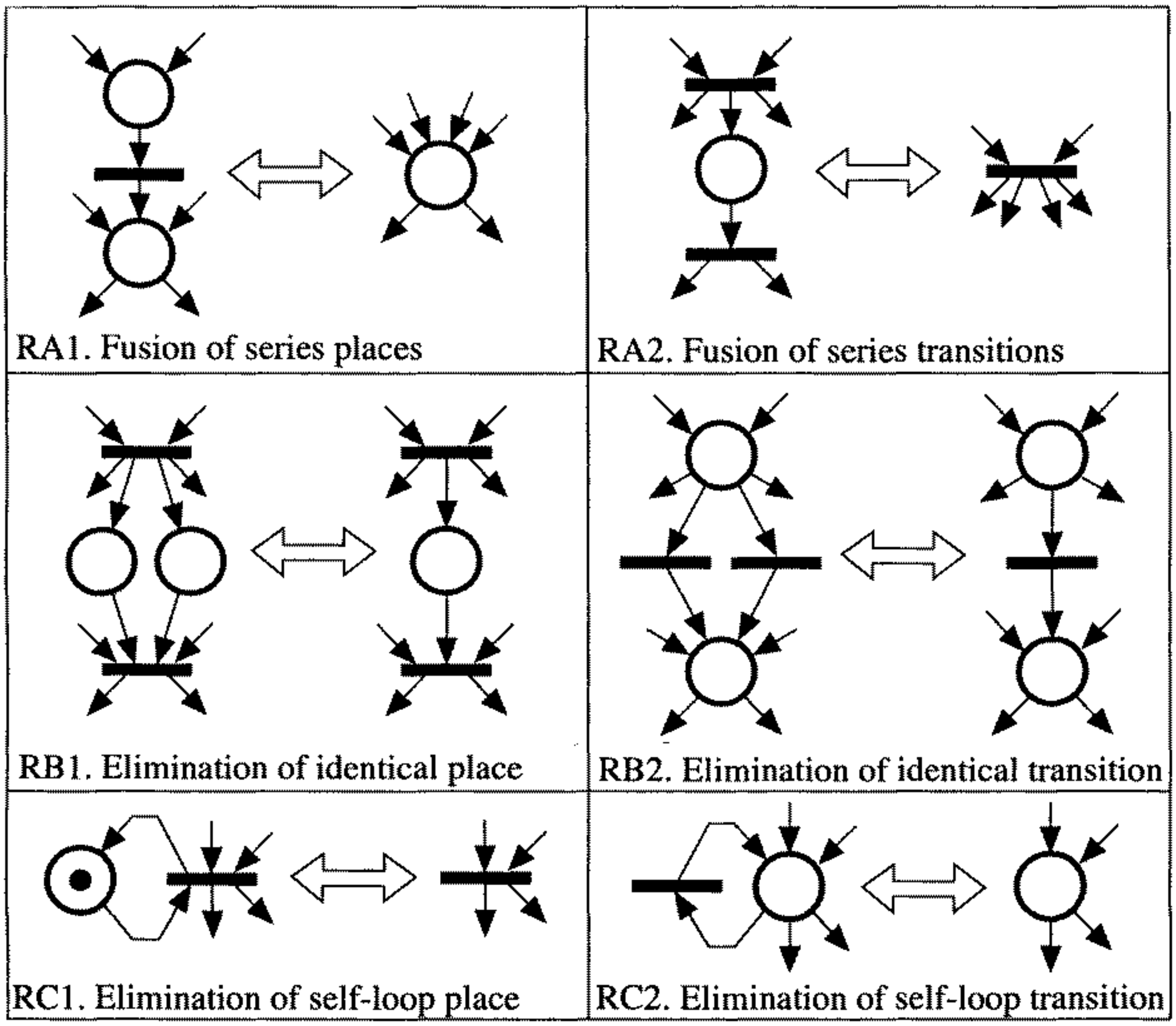

RA1. Fusion of series places
RA2. Fusion of series transitions
RB1. Elimination of identical place
RB2. Elimination of identical transition
RC1. Elimination of self-loop place
RC2. Elimination of self-loop transition

Figure 16.1: A basic reduction kit.

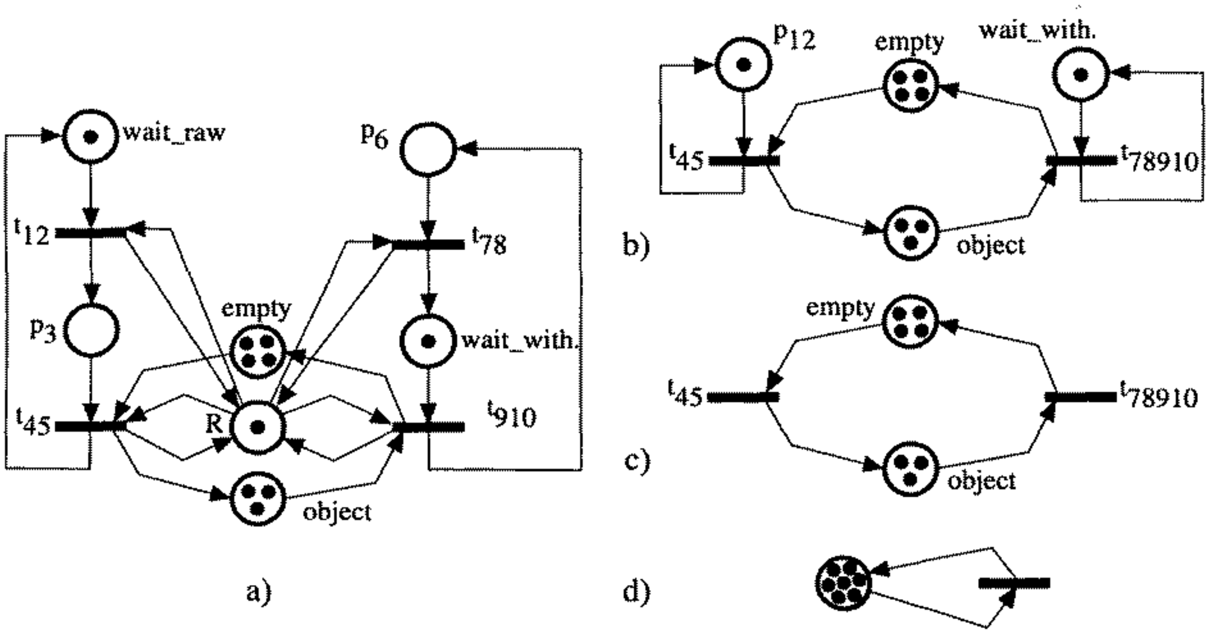


Figure 16.2: The reduction process shows (see (d)) that the net system in figure 6.8 is live, 7-bounded and reversible.

- *RA1* is a particular case of the *macroplace rule* [37].
- *RA2* is a particular case of the *transition fusion rule* [2]
- *RB1* and *RC1* are particular cases of the *implicit place rule* [38, 40] (to be considered later in more detail). Observe that *RC1* can be trivially generalized creating several self-loops in which the place always appears. Liveness, the bound of places, and reversibility are preserved. Moreover if the place contains several tokens, liveness, boundedness (in general not the bound of the net system) and reversibility are preserved.
- *RB2* and *RC2* are particular cases of *identical* and *identity transition rules* [2], respectively.

An interesting remark is the analogy between rules at the same row in Figure 16.1: Basically rules *RX2* are obtained from rules *RX1* by changing the role of places and transitions (*duality*) and reversing the arrows (important only for rules *RA*).

Example 16.1 Let us consider now the net system in Figure 6.8.b. The subnet defined by $\mathrm{op}_1 - t_3 -$ wait_dep. verifies the precondition of rule *RA1*. Thus it can be reduced to a place, p_3 (Fig. 16.2.a). The same holds for $\mathrm{op}_2 - t_6 -$ wait_free that is reduced to p_6 (Fig. 16.2.a). The subnets $t_1 -$ load $- t_2$, $t_4 -$ deposit $- t_5$, $t_7 -$ unload $- t_8$, and $t_9 -$ withdraw $- t_{10}$ can be reduced according to *RA2* (see t_{12}, t_{45}, t_{78} and t_{910} in Fig. 16.2.a). Place R in Fig. 16.2.a is implicit (one of the trivial generalizations mentioned for *RC1*). Thus it can be removed, and wait_draw $- t_{12} - p_3$ and $t_{910} - p_6 - t_{78}$ can be reduced to p_{12} and t_{78910}, respectively (see Figure 16.2.b). Places p_{12} and wait_with. are implicit (*RC1*) in Figure 16.2.b, thus the net system in Figure 16.2.c is obtained. Playing the token game, a place (e.g. object) can became empty in Figure 16.2.c and $t_{45} -$ object $- t_{78910}$ can be reduced (*RA2*) to a single transition (Fig. 16.2.d). Therefore, the original net system is live, 7-bounded and reversible.

16.2.2 Implicit places

A place in a net system is a constraint to the firing of its output transitions. If the removal of a place does not change the behaviour of the original net system, it represents a redundancy in the system and it can be removed. A place whose removal preserves the behaviour of the system is called an *implicit place.* Two notions of behaviour equivalence are used to define implicit places. The first one considers that the two net systems have the same behaviour if they present the same fireable sequences. That is, this place can be removed without changing the *sequential observation* of the behaviour of the net system (i.e. the set of fireable sequences). Implicit places under this equivalence notion are called *sequential implicit places* (SIP). The second notion of equivalence imposes that the two net systems must have the same sequences of steps. In this case the implicit places are called *concurrent implicit places* (CIP) and their removal does not change the possibilities of simultaneous occurrences of transitions in the original net system. Implicit places model false synchronisations on their output transitions.

Definition 16.1 *Let $\mathcal{S} = \langle \mathcal{N}, \mathbf{m_0} \rangle$ be a net system and $\mathcal{S}' = \langle \mathcal{N}', \mathbf{m_0}' \rangle$ the net system resulting from removing place p from $\mathcal{S}$. The place p is a*

1. *Sequential Implicit Place (SIP) iff* $\mathrm{L}(\mathcal{N}, \mathbf{m_0}) = \mathrm{L}(\mathcal{N}', \mathbf{m_0}')$, *i.e., the removing of place p preserves all firing sequences of the original net.*

2. *Concurrent Implicit Place(CIP) iff* $\mathrm{LS}(\mathcal{N}, \mathbf{m_0}) = \mathrm{LS}(\mathcal{N}', \mathbf{m_0}')$, *i.e., the removing of place p preserves all sequences of steps of the original net.*

It is easy to see that if a place p is a CIP then it is also a SIP (since the preservation of the sequences of steps implies the preservation of the firing sequences). Nevertheless, the contrary is not true in general. Let us consider, for example, the net in Figure 6.2. The place p_6 is a SIP since its removal does not change the set of firing sequences (the reachability graphs of the original net system and the net system without place p_6 are the same). But the place p_6 is not a CIP because after its removal transitions b and c can occur simultaneously and in the original net system they are sequentialised (i.e. the steps are not preserved). A SIP with self-loops, in order to be a CIP may require more tokens in its initial marking than those making it a SIP (in our example p_6 to be CIP requires two tokens in the initial marking). In [8] it is proven that a self-loop free SIP is also a CIP.

Let p be an CIP of the net system $\mathcal{S}$ and $\mathcal{S}'$ the net system $\mathcal{S}$ without the CIP p. Let σ be a fireable sequence of steps in $\mathcal{S}$, such that $\mathbf{m_0} \xrightarrow{\sigma} \mathbf{m}$. The sequence σ is also fireable in the net system $\mathcal{S}'$, i.e., $\mathbf{m_0}' \xrightarrow{\sigma} \mathbf{m}'$. This is because the removal of an CIP preserves the fireable sequences of steps of the net system. A trivial consequence of this is that the reached markings in $\mathcal{S}$ and $\mathcal{S}'$, firing the same sequence σ, are strongly related: $\forall q \in P \setminus \{p\}$, $\mathbf{m}[q] = \mathbf{m}'[q]$. Moreover, if $\mathbf{s}$ is a step enabled at $\mathbf{m}'$ the following holds: $\mathbf{m}' \geq \mathbf{Pre}' \cdot \mathbf{s} \Longrightarrow \mathbf{m}[p] \geq \sum_{t \in (p^\bullet \cap \|\mathbf{s}\|)} \mathbf{s}[t] \cdot \mathbf{Pre}[p,t]$. If p is a SIP the previous property can be writen in the following way: $\forall t \in p^\bullet$, $\mathbf{m}' \geq \mathbf{Pre}'[P', t] \Longrightarrow \mathbf{m}[p] \geq \mathbf{Pre}[p,t]$.

The elimination of a CIP or a SIP preserves: deadlock-freeness, liveness and marking mutual exclusion properties; but it does not preserve: boundedness and reversibility. Moreover, the elimination of a CIP preserves the firing mutual exclusion property, but this is not true for SIPs.

Example 16.2 The net system in Fig. 16.3.a is unbounded (p_4 is the unique unbounded place) and non-reversible (also because of p_4). Place p_4 is a CIP. Removing p_4 the system becomes bounded and reversible! On the other hand, place p_6 in Figure 6.2 imposes firing mutual exclusion between b and c. Being p_6 a SIP, the reduction rule does not preserve firing mutual exclusion. According to the definition, fireable sequences are preserved.

Sometimes it is practical to impose an additional condition to the definition of implicit places, asking their marking to be redundant (computable) with

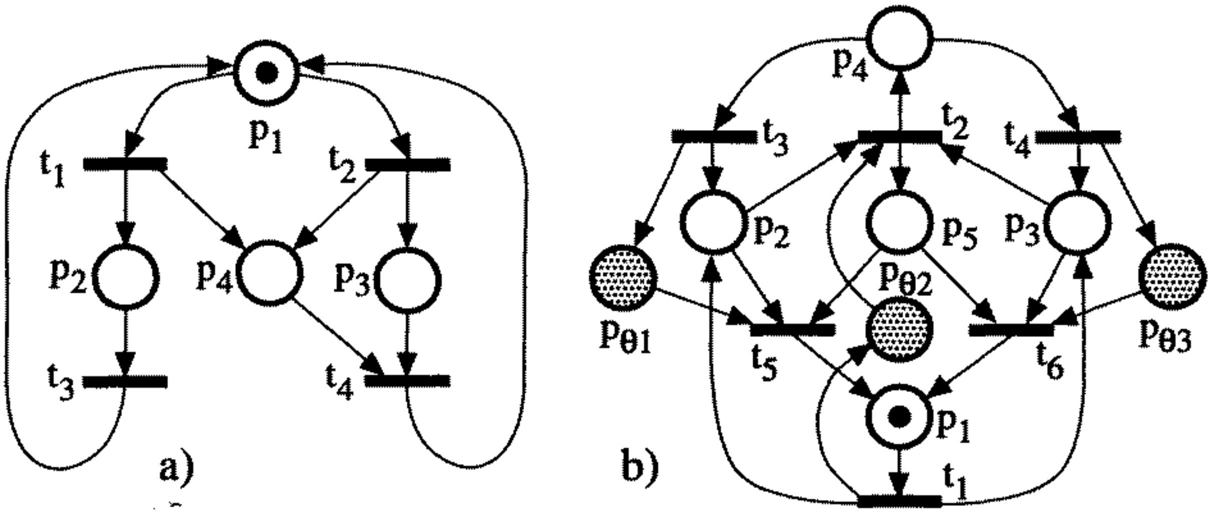


Figure 16.3: a) Place p_4 is firing implicit but not marking implicit. Removing p_4 the "false" synchronisation in t_4 disappears. b) The places in the set $\{p_{\theta 1}, p_{\theta 2}, p_{\theta 3}\}$ (or $\{p_2, p_3, p_5\}$) are CIPs.

respect to (from) the marking of the other places in the net (i.e. a marking redundancy property). Let us consider the CIP $p_{\theta 1}$ of the net system, $\mathcal{S}$, depicted in Figure 16.3.b. This place is CIP and its marking can be computed from the marking of places p_1, p_2 and p_5: $\forall \mathbf{m} \in \mathrm{RS}(\mathcal{S})$, $\mathbf{m}[p_{\theta 1}] = \mathbf{m}[p_1] + \mathbf{m}[p_2] + \mathbf{m}[p_5] - 1$. This class of places will be called *marking implicit places*. Nevertheless, the marking of some implicit places cannot be exclusively computed from the marking of the other places in the net. These places will be called *firing implicit places*. As an example consider the CIP p_4 in Figure 16.3.a: $\forall \mathbf{m} \in \mathrm{RS}(\mathcal{S})$, such that $\mathbf{m_0} \xrightarrow{\sigma} \mathbf{m}$, $\mathbf{m}[p_4] = \mathbf{m}[p_3] + \sigma[t_1])$. The classification of the implicit places into marking and firing implicit places can be applied to the two previously defined classes: CIP and SIP. Because of the additional condition, marking implicit places preserve the state space (i.e., the reachability graph of the net system with and without p are isomorphous), therefore they preserve boundedness and reversibility, too.

Implicit places presented until now are in a behavioural setting. In order to do the verification we must resort to algorithms based on the reachability graph with the inherent limitations and the high associated complexity. The structural formulation of the implicit place reduction rule requires the statement of a structure based condition to be fulfilled by the implicit place and the characteristics of its initial marking. Places satisfying the structure based condition will be called *structurally implicit places*, that is, places that become implicit provided they are marked with enough tokens.

Definition 16.2 *Let $\mathcal{N}$ be a net. A place p of $\mathcal{N}$ is a structurally implicit place iff there exists a subset $I_p \subseteq P \setminus \{p\}$ such that $\mathbf{C}[p,T] \geq \sum_{q \in I_p} y_q \cdot \mathbf{C}[q,T]$, where y_q is a nonnegative rational number (i.e. $\exists \mathbf{y} \geq 0$, $\mathbf{y}[p] = 0$ such that $\mathbf{y} \cdot \mathbf{C} \leq \mathbf{C}[p,T]$ and $I_p = ||\mathbf{y}||$.*

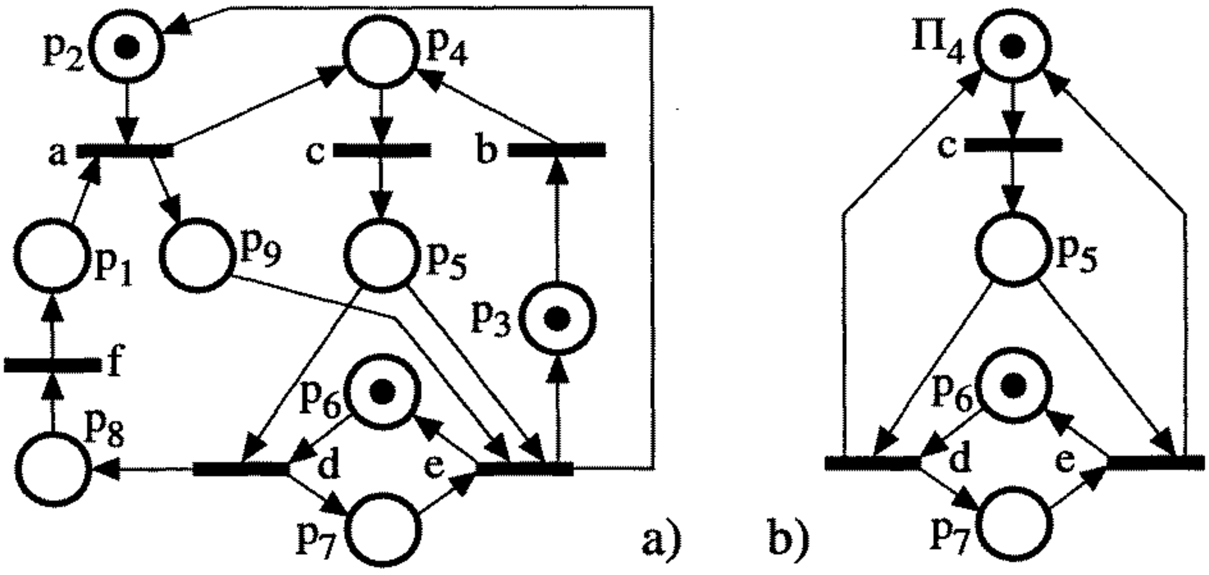


Figure 16.4: Places p_9 and p_2 (or p_2 and p_7) are implicits

Obviously, the above structural condition can be checked in polynomial time. The next property gives the initial marking conditions to be satisfied by a structurally implicit place to become a SIP or a CIP. This condition is based on the solution of a Linear Programming Problem (LPP 16.1) that computes an upper bound of the minimal initial marking of a structurally implicit place to be SIP or CIP in the net system $\langle\mathcal{N}, \mathbf{m_0}\rangle$. Because, LPPs are of polynomial time complexity [34], the evaluation of this condition has this complexity.

Property 16.3 *Let $\langle\mathcal{N}, \mathbf{m_0}\rangle$ be a net system. A structurally implicit place p of $\mathcal{N}$, with initial marking $\mathbf{m_0}[p]$, is a SIP (CIP) if $\mathbf{m_0}[p] \geq z$, where z is the optimal value of the LPP 16.1 with $\alpha = 1$ ($\alpha = \max\{\sum_{t \in p^\bullet} \mathbf{s}[t] | \mathbf{s} \in \mathrm{LS}(\mathcal{N}, \mathbf{m_0})\}$).*

$$\begin{array}{rll} z = & \mathit{min.} & \mathbf{y} \cdot \mathbf{m_0} + \alpha \cdot \mu \\ & \mathit{s.t.} & \mathbf{y} \cdot \mathbf{C} \leq \mathbf{C}[p,T] \\ & & \mathbf{y} \cdot \mathbf{Pre}[P,t] + \mu \geq \mathbf{Pre}[p,t] \quad \forall t \in p^\bullet \\ & & \mathbf{y} \geq 0, \mathbf{y}[p] = 0 \end{array} \qquad (16.1)$$

If the optimal solution of the LPP 16.1, for a structurally implicit place p, verifies that $\mathbf{y} \cdot \mathbf{C} = \mathbf{C}[p,T]$, then p is a marking implicit place and the following holds: $\forall \mathbf{m} \in \mathrm{RS}(\mathcal{N}, \mathbf{m_0})$, $\mathbf{m}[p] = \mathbf{y} \cdot \mathbf{m} + \alpha \cdot \mu$.

Observe that a structurally implicit place can become implicit for any initial marking of places $P \setminus \{p\}$ if we have the freedom to select an adequate initial marking for it. This property is not true for CIPs (or SIPs) that are not structurally implicit places. For example, the place p_{10} in Figure 2.4.a is a CIP but it is not a structurally implicit place. Moreover, the place p_{10} is not implicit if we change the initial marking of place p_4 from 0 to 1.

Example 16.3 Solving the LPP 16.1 for the place p_9 in Fig. 16.4.a with $\alpha = 1$ we obtain $z = 0$, for the optimal solution: $\mathbf{y} = [0, 0, 1, 1, 1, 0, 1, 0,$

0] and $\mu = -1$. Moreover, $\mathbf{C}[p_9, T] = \mathbf{C}[p_3, T] + \mathbf{C}[p_4, T] + \mathbf{C}[p_5, T] + \mathbf{C}[p_7, T]$. Because $\mathbf{m_0}[p] \geq z = 0$, p_9 is a SIP (since p_9 is self-loop free place it is also a CIP) and can be removed. Being p_9 a marking implicit place we can write: $\forall \mathbf{m} \in \mathrm{RS}(\mathcal{N}, \mathbf{m_0})$, $\mathbf{m}[p_9] = \mathbf{m}[p_3] + \mathbf{m}[p_4] + \mathbf{m}[p_5] + \mathbf{m}[p_7] - 1$.

Once p_9 is removed, a similar computation for p_2 can be done and p_2 is also shown to be a CIP. Figure 16.4.b shows a reduced net system. It can be obtained reducing $p_3 - b - p_4$ into a place (say p_{34}) (*RA1*) and finally $p_8 - f - p_1 - a - p_{34}$ into Π_4. Using the kit in Figure 16.1, plus the implicit place rule no more reductions can be performed: the net system is irreducible with respect to that kit of reduction rules. The rule *RA1* allows to fuse Π_4 and p_5. The new place is implicit, so it can be removed. Then a cycle with $p_6 - d - p_7 - e - p_6$ remains. Finally it can be reduced to a basic net, $p_6 - t_{de} - p_6$, with one token. Therefore the original net system is live, bounded. It is also reversible, but we cannot guarantee this because of the fusion of $p_3 - b - p_4$ into p_{34}.

16.3 Linear algebraic techniques

Analysis techniques based on linear algebra allow the verification of properties of a general net system. The key idea is simple, and it has been already commented previously: Let $\mathcal{S}$ be a net system with incidence matrix $\mathbf{C}$. If $\mathbf{m}$ is reachable from $\mathbf{m_0}$ by firing sequence σ, then $\mathbf{m} = \mathbf{m_0} + \mathbf{C} \cdot \boldsymbol{\sigma}$. Therefore the set of natural solutions, $(\mathbf{m}, \boldsymbol{\sigma})$, of this state equation defines a linearisation of the reachability set $\mathrm{RS}(\mathcal{S})$ denoted $\mathrm{LRS}^{SE}(\mathcal{S})$. This set can be used to analyse properties like marking and submarking reachability and coverability, firing concurrency, conflict situations, deadlock-freeness, mutual exclusion, k-boundedness, existence of frozen tokens, synchronic relations, etc. To do so, the properties are expressed as formulas of a first order logic having linear inequalities as atoms, where the reachability or fireability conditions are relaxed by satisfiability of the state equation. These formulas are verified checking existence of solutions to systems of linear equations that are automatically obtained from them [8]. For instance, if $\forall \mathbf{m} \in \mathrm{RS}(\mathcal{S}) : \mathbf{m}[p] = 0 \vee \mathbf{m}[p'] = 0$; then places p and p' are in mutual exclusion. This is verified checking absence of (natural) solutions to $\{\mathbf{m} = \mathbf{m_0} + \mathbf{C} \cdot \boldsymbol{\sigma} \wedge \mathbf{m}[p] > 0 \wedge \mathbf{m}[p'] > 0\}$. *Integer Linear Programming Problems* [35] where the state equation is included in the set of constraints can be posed to express optimization problems, like the computation of marking bounds, synchronic measures, etc. [8, 40]. This approach is a generalization of the classical reasoning using linear invariants [29, 31], and it deeply bridges the domains of net theory and convex geometry resulting in a unified framework to understand and enhance structural techniques [8] (see Subsection 6.3.2).

Unfortunately, it usually leads to only semidecision algorithms (i.e., only necessary or only sufficient conditions) because, in general, $\mathrm{RS}(\mathcal{S}) \subset \mathrm{LRS}^{SE}(\mathcal{S})$. The undesirable solutions are named *spurious*.

Example 16.4 (Existence of spurious solutions and their consequences in the analysis) Let us consider the net system depicted in Figure 2.3. The corresponding net state equation has the following marking spurious solutions: $\mathbf{m}_1 = 2 \cdot p_4$, $\mathbf{m}_2 = 2 \cdot p_2$, $\mathbf{m}_3 = 2 \cdot p_3$, $\mathbf{m}_4 = 2 \cdot p_5$, $\mathbf{m}_5 = p_2 + p_4$, $\mathbf{m}_6 = p_3 + p_4$. The first four solutions allow to conclude that p_2, p_3, p_4 and p_5 are 2-bounded, while they are really 1-bounded (check it). The solutions $\mathbf{m}_2$, $\mathbf{m}_3$ and $\mathbf{m}_4$ are total deadlocks. Then using the state equation we cannot conclude that the system in Fig. 2.3 is deadlock-free.

Spurious solutions can be removed using certain structural techniques, consequently improving the quality of the linear description of the system [10]. For example, it is clear that adding implicit places, a new system model with identical behaviour is obtained. For some net systems, if the implicit places are chosen carefully, the state equation of the new system may have no integer spurious solution preventing to conclude on the bound of a place or the deadlock freeness of the system.

Example 16.5 (Elimination of spurious solutions) The net system in Figure 16.3.b has been obtained adding the implicit places $p_{\theta 1}$, $p_{\theta 2}$ and $p_{\theta 3}$ to that in Figure 2.3. The above mentioned spurious solutions, $\mathbf{m}_i, i = 1 \ldots 6$; are not solutions of the new state equation. Moreover, we can conclude now that the new (and original) net system(s) were 1-bounded and deadlock-free!

Anyway the algorithms based on linear algebra do decide in many situations, and they are relatively efficient, specially if the integrality of variables is disregarded. (This further relaxation may spoil the quality, although in many cases it does not [14, 40].) Moreover, these techniques allow in an easy way an initial marking *parametric* analysis (e.g. changing the number of customers, size of resources, initial distribution of customers and/or resources, etc). The application of these techniques to the analysis of the boundedeness and deadlock-freeness properties is illustrated in subsections 16.3.1 and 16.3.2, repectively.

In temporal logic terms, the above outlined approach is well suited for *safety* properties ("some bad thing never happens"), but not so much for *liveness* properties ("some good thing will eventually happen"). For instance, the formula expressing reversibility would be $\forall \mathbf{m} \in \mathrm{LRS}^{SE}(\mathcal{S}) : \exists \boldsymbol{\sigma}' \gneqq 0 : \mathbf{m_0} = \mathbf{m} + \mathbf{C} \cdot \boldsymbol{\sigma}'$, but this is neither necessary nor sufficient for reversibility. The general approach to linearly verify these liveness properties is based on the verification of safety properties that are necessary for them to hold, together with some inductive reasoning [26]. For instance, deadlock-freeness is necessary for transition liveness, and the existence of some *decreasing potential function* proves reversibility [39] (see subsection 16.3.4).

Another important contribution of linear techniques to liveness analysis has been the derivation of *ad hoc* simple and efficient semidecision conditions. In subsection 16.3.3, we present one of these conditions based on a rank upper bound of the incidence matrix, which was originally conceived when computing the *visit ratios* in certain subclasses of net models [6].

The following subsections study marking bounds and boundedness, deadlock-freeness, structural liveness and liveness, and reversibility.

16.3.1 Bounds and boundedness

The study of the bound of a place p, $\mathbf{b}(p)$, through linear algebraic techniques, requires the linearisation of the reachability set in the definition of $\mathbf{b}(p)$ by means of the state equation of the net. In this subsection we assume that $\mathbf{m} \in \mathbb{R}^n$ and $\boldsymbol{\sigma} \in \mathbb{R}^m$. This linearisation of the definition of $\mathbf{b}(p)$ leads to a new quantity called the *structural bound* of p, $\mathbf{sb}(p)$:

$$\mathbf{sb}(p) = \sup\{\mathbf{m}(p) | \mathbf{m} = \mathbf{m_0} + \mathbf{C} \cdot \boldsymbol{\sigma} \geq 0, \boldsymbol{\sigma} \geq 0\} \tag{16.2}$$

Let $\mathbf{e_p}$ be the *characteristic vector* of p: $\mathbf{e_p}[q] :=$ if $q = p$ then 1 else 0. The structural bound of p, $\mathbf{sb}(p)$, can be obtained as the optimal solution of the following Linear Programming Problem (LPP):

$$\begin{array}{rll} \mathbf{sb}(p) = & \text{max.} & \mathbf{e_p} \cdot \mathbf{m} \\ & \text{s.t.} & \mathbf{m} = \mathbf{m_0} + \mathbf{C} \cdot \boldsymbol{\sigma} \geq 0 \\ & & \boldsymbol{\sigma} \geq 0 \end{array} \tag{16.3}$$

Therefore $\mathbf{sb}(p)$ can be computed in polynomial time. In sparse-matrix problems (matrix $\mathbf{C}$ is usually sparse), good implementations of the classical *simplex method* leads to quasi-linear time complexities.

Because $\text{RS}(\mathcal{S}) \subset \text{LRS}^{SE}(\mathcal{S})$, in general, we have that $\mathbf{sb}(p) \geq \mathbf{b}(p)$ (recall example 16.4). Therefore, if we are investigating the k-boundedness of a place (i.e. $\mathbf{m}[p] \leq k$), we have a sufficient condition in polynomial time: if $\mathbf{sb}(p) \leq k$ then $\mathbf{b}(p) \leq k$ (i.e. p is k-bounded).

In the sequel we argue on classical results from linear programming and convex geometry theories. We assume the reader is aware of these theories (see, for example, [33, 34]); otherwise all the needed arguments are compiled and adapted in [40]. The important point here is to convey the idea that other theories are helpful to understand in a deep and general framework many sparse results on net systems' behaviours. The dual linear programming problem of 16.3 is the following (see any text on linear programming to check it):

$$\begin{array}{rll} \mathbf{sb}(p)' = & \text{min.} & \mathbf{y} \cdot \mathbf{m_0} \\ & \text{s.t.} & \mathbf{y} \cdot \mathbf{C} \leq 0 \\ & & \mathbf{y} \geq \mathbf{e_p} \end{array} \tag{16.4}$$

The LPP 16.3 has always a feasible solution ($\mathbf{m} = \mathbf{m_0}$, $\boldsymbol{\sigma} = 0$). Using duality and boundedness theorems from linear programming theory, both LPPs 16.3 and 16.4 are bounded (thus p is structurally bounded) and $\mathbf{sb}(p) = \mathbf{sb}(p)'$ iff there exists a *feasible solution* for the LPP 16.4: $\mathbf{y} \geq \mathbf{e_p}$ such that $\mathbf{y} \cdot \mathbf{C} \leq 0$.

The reader can easily check that LPP 16.4 makes in polynomial time an "implicit search" for the structural bound of p on a set of structural objects including all the p-semiflows. In this sense, we can say that analysis methods based on the state equation are more general than those based on linear invariants. That is, the dual LPPs of those based on the state equation consider not only the p-semiflows but other structural objects as $\mathbf{y} \geq 0$ such that $\mathbf{y} \cdot \mathbf{C} \lneq 0$. On the other hand we must say that the computational effort using the linear invariants is greater than using the state equation, since the computation of the minimal p-semiflows (in some cases, an exponential number!) must be done previously to the study of the property.

From the above discussion and using the *alternatives theorem* (an algebraic form of the *Minkowski-Farkas lemma*) the following properties can be proved:

Property 16.4 *The following three statements are equivalent:*

1. *p is structurally bounded, i.e. p is bounded for any $\mathbf{m_0}$.*
2. *There exists $\mathbf{y} \geq \mathbf{e_p}$ such that $\mathbf{y} \cdot \mathbf{C} \leq 0$. (place-based characterization)*
3. *For all $\mathbf{x} \geq 0$ such that $\mathbf{C} \cdot \mathbf{x} \geq 0$, $\mathbf{C}[p,T] \cdot \mathbf{x} = 0$. (transition-based characterization)*

Property 16.5 *The following three statements are equivalent:*

1. *$\mathcal{N}$ is structurally bounded, i.e. $\mathcal{N}$ is bounded for any $\mathbf{m_0}$.*
2. *There exists $\mathbf{y} \geq \mathbf{1}$ such that $\mathbf{y} \cdot \mathbf{C} \leq 0$. (place-based characterization)*
3. *For all $\mathbf{x} \geq 0$ such that $\mathbf{C} \cdot \mathbf{x} \geq 0$, $\mathbf{C} \cdot \mathbf{x} = 0$; i.e. $\nexists\, \mathbf{x} \geq 0$ s.t. $\mathbf{C} \cdot \mathbf{x} \gneqq 0$. (transition-based characterization)*

16.3.2 Deadlock-freeness (and liveness)

Deadlock-freeness concerns the existence of some activity from any reachable state of the system. It is a necessary condition for liveness, although in general not sufficient. When no part of the system can evolve, it is said that the system has reached a state of total deadlock (or *deadlock* for short). In net system terms, a deadlock corresponds to a marking from which no transition is fireable. In order to study deadlock-freenes by means of linear algebraic techniques, the property must be expressed as a formula of a first order logic having linear inequalities as atoms, where the reachability or fireability conditions are relaxed by satisfiability of the state equation. The formula to express that a marking is a deadlock consists of a condition for every transition expressing that it is disabled at such marking. This condition consists of several inequalities, one per input place of the transition (expressing that the marking of such place is less than the corresponding weight) linked by the "∨" connective (because lack of tokens in a single input place disables the transition). We give below a basic general sufficient condition for deadlock-freeness based on the absence of solutions satisfying simultaneously the net state equation and the formula expressing the total deadlock condition commented above.

Proposition 16.6 *Let $\langle \mathcal{N}, \mathbf{m_0} \rangle$ be a net system. If there doesn't exist any solution $(\mathbf{m}, \boldsymbol{\sigma})$, for the system*

$$\begin{aligned} &\mathbf{m} = \mathbf{m_0} + \mathbf{C} \cdot \boldsymbol{\sigma} \\ &\mathbf{m} \geq 0, \boldsymbol{\sigma} \geq 0 \\ &\bigvee_{p \in {}^\bullet t} \mathbf{m}[p] < \mathbf{Pre}[p,t]; \forall t \in T \end{aligned} \tag{16.5}$$

then $\langle \mathcal{N}, \mathbf{m_0} \rangle$ *is deadlock-free.*

Obviously, the deadlock conditions are non linear, because they are expressed using the "$\vee$" connective. Anyway we can express the above condition by means of a set of linear systems as follows. Let $\alpha : T \to P$ be a mapping that assigns to each transition one of its input places. If there doesn't exist α such that the system

$$\begin{aligned} &\mathbf{m} = \mathbf{m_0} + \mathbf{C} \cdot \boldsymbol{\sigma} \\ &\mathbf{m} \geq 0, \boldsymbol{\sigma} \geq 0 \\ &\mathbf{m}[\alpha(t)] < \mathbf{Pre}[\alpha(t), t]; \forall t \in T \end{aligned} \tag{16.6}$$

has a solutîon, then $\langle \mathcal{N}, \mathbf{m_0} \rangle$ is deadlock-free. The problem is that we have to check it for *every* mapping α of input places to transitions so we have to check $\prod_{t \in T} |{}^\bullet t|$ systems of linear inequalities. If every transition has exactly one input place (e.g. State Machines) then only one system needs to be checked, but in general the number might be large. Nevertheless it is possible to reduce the number of systems to be checked, preserving the set of *integer solutions.* For this purpose, the work [42] presents four simplification rules of the deadlock condition using information obtained from the net system, and a simple net transformation obtaining an equivalent one wrt. the deadlock-freeness property where the enabling conditions of transitions can be expressed linearly. As a result, deadlock-freeness of a wide variety of net systems can be proven by verifying absence of solutions to a single system of linear inequalities. Even more, in some subclasses it is known that there are no spurious solutions being deadlocks, so the method decides on deadlock-freeness [43]. The following example presents the deadlock-freenes analysis of the net system in figure 6.8 applying this technique.

Example 16.6 (Deadlock-freeness analysis and simplification rules) Let us consider the net system in Figure 6.8. The direct application of the method described in proposition 16.6 requires to check $\prod_{t \in T} |{}^\bullet t| = 36$ linear *systems as that presented in 16.6. Nevertheless, below we show that we can* reduce the deadlock-freeness analysis on this net to check a unique linear system applying the simplification rules presented in [42]. Solving the LPP 16.3 for the places of the net system we obtain the following: $\mathbf{sb}(\mathrm{p}) = 1$, for all $p \in P \setminus \{\mathrm{empty}, \mathrm{object}\}$; and $\mathbf{sb}(\mathrm{empty}) = \mathbf{sb}(\mathrm{object}) = 7$ (the same can be obtained from the linear invariants in Eqs 6.1-6.4). The transitions t_1, t_4, t_7 and t_9 are those presenting complex conditions giving rise to the large number of linear systems. The simplification of these consitions is as follows:

a) The non-fireability condition of t_1 is $(\mathbf{m}[\mathrm{wait\_raw}] = 0) \vee (\mathbf{m}[\mathrm{R}] = 0)$. Taking into account that $\mathbf{sb}(\mathrm{wait\_raw}) = \mathbf{sb}(\mathrm{R}) = 1$, we can apply a particularization of rule 3 in [42] to replace the previous complex condition by a unique linear inequality: Let t be a transition such that each input place verifies that its structural bound is equal to the weight of its output arc joining it to t. The non fireability condition for transition t at a marking $\mathbf{m}$ is $\sum_{p\in {}^\bullet t} \mathbf{m}[p] \le \sum_{p\in P} \mathbf{Pre}[p,t]-1$. That is, the amount of tokens in the input places of t is less than the needed. Therefore, for the transition t_1 this linear condition is: $\mathbf{m}[\text{wait\_raw}]+\mathbf{m}[\text{R}] \le 1$.

b) The non-fireability condition of t_7 is $(\mathbf{m}[\text{wait\_free}]=0) \vee (\mathbf{m}[\text{R}]=0)$. In a similar way to the case of transition t_1 we replace this condition by $\mathbf{m}[\text{wait\_free}]+\mathbf{m}[\text{R}] \le 1$, since $\mathbf{sb}(\text{wait\_free})=\mathbf{sb}(\text{R})=1$ and rule **3** in [42] can be applied.

c) The non-fireability condition of t_4 is $(\mathbf{m}[\text{wait\_dep.}]=0) \vee (\mathbf{m}[\text{R}]=0) \vee (\mathbf{m}[\text{empty}]=0)$. Since $\mathbf{sb}(\text{wait\_dep.})=\mathbf{sb}(\text{R})=1$ and $\mathbf{sb}(\text{empty})=7$ (i.e. only one input place of t_7 has a **sb** greater than the weight of the arc) rule 4 of [42] can be applied. Then, the previous complex condition is replaced by the following linear condition:

$$\begin{aligned}&\mathbf{sb}(\text{empty})\cdot(\mathbf{m}[\text{wait\_dep.}]+\mathbf{m}[\text{R}])+\mathbf{m}[\text{empty}] \le \\ &\qquad \mathbf{sb}(\text{empty})\cdot(\mathbf{Pre}[\text{wait\_dep.},T]+\mathbf{Pre}[\text{R},T])+\mathbf{Pre}[\text{empty},T]-1\end{aligned}$$

i.e. $7(\mathbf{m}[\text{wait\_dep.}]+\mathbf{m}[\text{R}])+\mathbf{m}[\text{empty}] \le 14$.

d) The non-fireability condition of t_9 can be reduced to the following linear condition by similar reasons to the case of transition t_4: $7(\mathbf{m}[\text{wait\_with.}]+\mathbf{m}[\text{R}])+\mathbf{m}[\text{object}] \le 14$.

Applying the previously stated simplifications, the deadlock-freeness analysis for the net system in Figure 6.8 is reduced to verify that there doesn't exist any solution $(\mathbf{m},\boldsymbol{\sigma})$, for the following single linear system (the reader can check that the system has no solutions).

$$\begin{array}{ll}\mathbf{m}=\mathbf{m_0}+\mathbf{C}\cdot\boldsymbol{\sigma} & \\ \mathbf{m}\ge 0, \boldsymbol{\sigma}\ge 0 & \\ \mathbf{m}[\text{wait\_raw}]+\mathbf{m}[\text{R}]\le 1; & \text{for } t_1 \\ \mathbf{m}[\text{load}]=0; & \text{for } t_2 \\ \mathbf{m}[\text{op}_1]=0; & \text{for } t_3 \\ 7(\mathbf{m}[\text{wait\_dep.}]+\mathbf{m}[\text{R}])+\mathbf{m}[\text{empty}]\le 14; & \text{for } t_4 \\ \mathbf{m}[\text{deposit}]=0; & \text{for } t_5 \\ \mathbf{m}[\text{op}_2]=0; & \text{for } t_6 \\ \mathbf{m}[\text{wait\_free}]+\mathbf{m}[\text{R}]\le 1; & \text{for } t_7 \\ \mathbf{m}[\text{unload}]=0; & \text{for } t_8 \\ 7(\mathbf{m}[\text{wait\_with.}]+\mathbf{m}[\text{R}])+\mathbf{m}[\text{object}]\le 14; & \text{for } t_9 \\ \mathbf{m}[\text{withdrawal}]=0; & \text{for } t_{10}\end{array} \tag{16.7}$$

Linear invariants may also be used to prove *deadlock-freeness*. Using the linear invariants in Eqs. (6.1-6.4), we shall prove that our net system in Figure 6.8 is deadlock-free.

If there exists a deadlock, no transition can be fired. Let us try to construct a marking in which no transition is fireable. When a unique input place of a transition exists, that place must be unmarked. So $\mathbf{m}[\text{load}] = \mathbf{m}[\text{op}_1] = \mathbf{m}[\text{deposit}] = \mathbf{m}[\text{op}_2] = \mathbf{m}[\text{unload}] = \mathbf{m}[\text{withdrawal}] = 0$, and the linear invariants in Eqs (6.1-6.4) reduce to:

$$\mathbf{m}[\text{wait\_raw}] + \mathbf{m}[\text{wait\_dep.}] = 1 \tag{16.8}$$
$$\mathbf{m}[\text{wait\_free}] + \mathbf{m}[\text{wait\_with.}] = 1 \tag{16.9}$$
$$\mathbf{m}[\text{empty}] + \mathbf{m}[\text{object}] = 7 \tag{16.10}$$
$$\mathbf{m}[\text{R}] = 1 \tag{16.11}$$

Since R should always be marked at the present stage, to prevent the firing of t_1 and t_7, places wait_raw and wait_free should be unmarked. The linear invariants are reduced once more, leading to:

$$\mathbf{m}[\text{wait\_dep.}] = 1 \tag{16.12}$$
$$\mathbf{m}[\text{wait\_with.}] = 1 \tag{16.13}$$
$$\mathbf{m}[\text{empty}] + \mathbf{m}[\text{object}] = 7 \tag{16.14}$$
$$\mathbf{m}[\text{R}] = 1 \tag{16.15}$$

Since $\mathbf{m}[\text{wait\_dep.}] = \mathbf{m}[\text{wait\_with.}] = 1$, to avoid the firing of t_4 and t_9, $\mathbf{m}[\text{empty}] + \mathbf{m}[\text{object}] = 0$ is needed. This contradicts Eq (16.14), so the net system is deadlock-free. A more compact, algorithmic presentation of the above deadlock-freeness proof is:

```
if $\mathbf{m}[\text{load}] + \mathbf{m}[\text{op}_1] + \mathbf{m}[\text{deposit}] + \mathbf{m}[\text{op}_2] + \mathbf{m}[\text{unload}] + \mathbf{m}[\text{withdrawal}] \geq 1$
  then one of $t_2$,$t_3$,$t_5$,$t_6$,$t_8$ or $t_{10}$ is fireable
  else if $\mathbf{m}[\text{wait\_raw}] + \mathbf{m}[\text{wait\_free}] \geq 1$
         then one of $t_1$or $t_7$ is fireable
         else one of $t_4$ or $t_9$ is fireable
```

As a final remark, we want to point out that liveness can be proved for the net system in Figure 6.8. Liveness implies deadlock-freeness, but the reverse is not true in general. Nevertheless, if the net is consistent and it has only one minimal t-semiflow, as it happens in the example, where the unique minimal t-semiflow is $\mathbf{1}$; then any infinite behaviour must contain all transitions with relative firings given by such t-semiflow. Thus deadlock-freeness implies, in this case, liveness.

16.3.3 Structural liveness and liveness

A necessary condition for a transition t to be live in a system $\langle \mathcal{N}, \mathbf{m_0} \rangle$ is its eventual infinite fireability, i.e. the existence of a firing repetitive sequence σ_R containing t: $\exists \sigma_R \in \mathrm{L}(\mathcal{N}, \mathbf{m_0})$ such that $\mathbf{m_0} \xrightarrow{\sigma_R} \mathbf{m} \geq \mathbf{m_0}$ and $\boldsymbol{\sigma}_R[t] > 0$.

Using the state equation as a linearisation of the reachability set, an *upper bound* of the number of times t can be fired in $\langle \mathcal{N}, \mathbf{m_0} \rangle$ is given by the following LPP ($\mathbf{e_t}[u] :=$ if $u = t$ then 1 else 0):

$$\begin{array}{rll} \mathbf{sr}(t) = & \text{max.} & \mathbf{e_t} \cdot \boldsymbol{\sigma} \\ & \text{s.t.} & \mathbf{m} = \mathbf{m_0} + \mathbf{C} \cdot \boldsymbol{\sigma} \geq 0 \\ & & \boldsymbol{\sigma} \geq 0 \end{array} \tag{16.16}$$

The dual of (LPP 16.16) is:

$$\begin{array}{rll} \mathbf{sr}(t)' = & \text{min.} & \mathbf{y} \cdot \mathbf{m_0} \\ & \text{s.t.} & \mathbf{y} \cdot \mathbf{C} \leq -\mathbf{e_t} \\ & & \mathbf{y} \geq 0 \end{array} \tag{16.17}$$

We are interested on characterizing when $\mathbf{sr}(t)$ goes to infinity. The LPP 16.16 has $\mathbf{m} = \mathbf{m_0}$ and $\boldsymbol{\sigma} = 0$ as a feasible solution. Using first duality and unboundedness theorems from linear programming and later the alternatives theorem, the following properties can be stated:

Property 16.7 *The following three statements are equivalent:*

1. *t is structurally repetitive (i.e. there exists a "large enough" $\mathbf{m_0}$ such that t can be fired infinitely often).*

2. *There does not exist $\mathbf{y} \geq 0$ such that $\mathbf{y} \cdot \mathbf{C} \leq -\mathbf{e_t}$ (place-based perspective }*

3. *There exists $\mathbf{x} \geq \mathbf{e_t}$ such that $\mathbf{C} \cdot \mathbf{x} \geq 0$ { transition-based perspective }*

Property 16.8 *The following three statements are equivalent:*

1. *$\mathcal{N}$ is structurally repetitive (i.e. all transitions are structurally repetitive).*

2. *There does not exist $\mathbf{y} \geq 0$ such that $\mathbf{y} \cdot \mathbf{C} \lneq 0$*

3. *There exists $\mathbf{x} \geq \mathbf{1}$ such that $\mathbf{C} \cdot \mathbf{x} \geq 0$*

Aditionally, the following classical results can be stated [31, 5, 38]:

Property 16.9 *Let $\mathcal{N}$ be a net and $\mathbf{C}$ its incidence matrix.*

1. **if** *$\mathcal{N}$ is structurally live* **then** *$\mathcal{N}$ is structurally repetitive.*

2. **if** *$\mathcal{N}$ is structurally live and structurally bounded* **then** *$\mathcal{N}$ is conservative ($\exists \mathbf{y} \geq \mathbf{1}$ such that $\mathbf{y} \cdot \mathbf{C} = 0$) and consistent ($\exists \mathbf{x} \geq \mathbf{1}$ such that $\mathbf{C} \cdot \mathbf{x} = 0$).*

3. **if** *$\mathcal{N}$ is connected, consistent and conservative* **then** *it is strongly connected.*

4. **if** *$\mathcal{N}$ is live and bounded* **then** *$\mathcal{N}$ is strongly connected and consistent.*

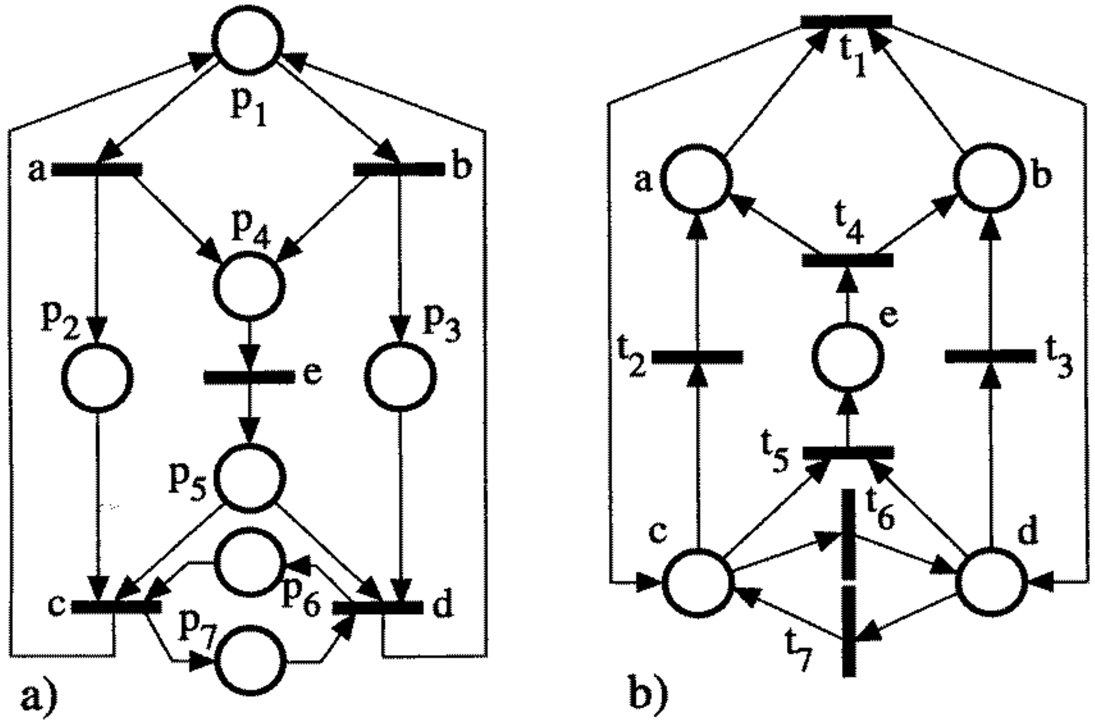


Figure 16.5: Two conservative and consistent, struturally non-live nets: (a) rank(**C**) = 4, |EQS| = 3, thus $\mathcal{N}$ is not structurally live; (b) rank(**C**) = 4, |EQS| = 4, |CCS| = 3, thus no answer.

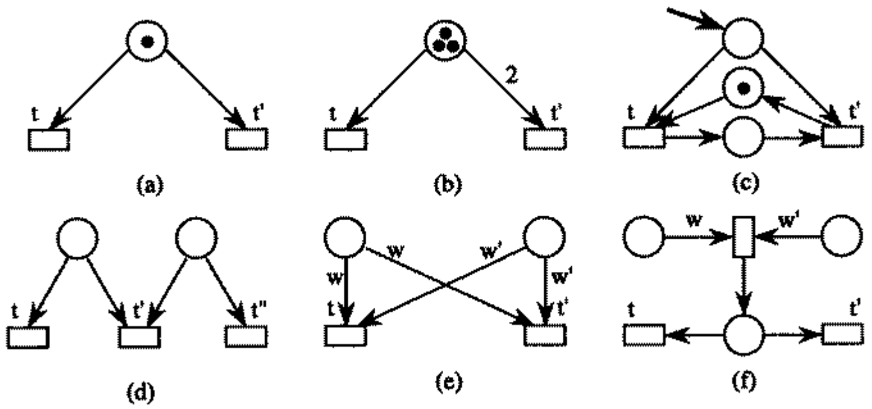


Figure 16.6: Conflicts and structural conflicts.

Net structures in Figure 16.5 are consistent and conservative, but there does not exist a live marking for them. A more careful analysis allows to improve the above result with a *rank condition* on the incidence matrix of $\mathcal{N}$, **C**. Before the introduction of this improved result we need to introduce certain structural objects related with conflicts.

Conflicts in sequential systems they are clearly the situation in which two actions are enabled so one must be chosen to occur. For instance, Figure 16.6 (a) shows a conflict between t and t'. Things become more complicated in the case of concurrent systems, where the fact that two transitions are enabled does not necessarily imply that we must choose one. Sometimes, the "sequential" definition there is a conflict when two transitions are enabled and the occurrence of one disables the other — is suitable, namely in 1-bounded systems. But in other cases a new definition is needed. Consider now the marking that puts two tokens in the place of Figure 16.6 (a). Neither the occurrence of t disables t' nor the converse, but the firing of one decreases the enabling degree of the other: so to say, each token must decide which way to go. Formally, *there is a conflict situation when the enabling vector is not an enabled step.* In Figure 16.6 (b), neither the occurrence of t or t' disables the other, but the firing of t' decreases the enabling degree of t from three to one. The enabling vector is $3t + t'$, while the (maximal) enabled steps are $3t$ and $t + t'$. By the way, this example shows that *conflict does not imply absence of concurrency*: t and t' are involved in a conflict, but they could occur concurrently, as in $t + t'$.

The very basic net construct used to model conflicts is a place with more than one output transition, i.e., a distributor place. In fact, distributor places are needed to model conflicts, but the converse is not true. Due to the regulation circuit in Figure 16.6 (c), t and t' are never in effective conflict although they share an input place. The output transitions of a distributor place are said to be in *structural conflict relation* ($\langle t_i, t_j \rangle \in$ SCf when ${}^\bullet t_i \cap {}^\bullet t_j \neq \emptyset$). This relation is reflexive and symmetric, but not transitive. Its transitive closure is named *coupled conflict relation*, and it partitions the transitions of a net into *coupled conflict sets* (CCS(t) denotes the coupled conflict set containing t). In Figure 16.6 (d) t and t'' are not in structural conflict relation but they are in coupled conflict relation, through t'.

Very often in the literature, our structural conflicts are called simply "conflicts", but we prefer to add the adjective structural to better distinguish from the behavioural, hence dynamical, notion of (effective) conflict, which depends on the marking. As we have noted, a structural conflict makes possible the existence of an effective conflict, but it does not guarantee it, e.g,. Figure 16.6 (d), except for the case of *equal conflicts*, where all the transitions in structural conflict have the same precondition. Transitions t and t' are said to be in *equal conflict relation*, $\langle t, t' \rangle \in EQ$, when $t = t'$ or $\mathbf{Pre}[P, t] = \mathbf{Pre}[P, t'] \neq \mathbf{0}$. This equivalence relation partitions the transitions into *equal conflict sets*. The equal conflict set containing t is denoted by EQS(t). Figure 16.6 (e) shows an equal conflict set.

Relating the rank of the incidence matrix with the number of (coupled o equal) structural conflicts in a net improves the previous conditions on structural liveness:

Property 16.10 *Let $\mathcal{N}$ be a net and* **C** *its incidence matrix.*

1. **if** *$\mathcal{N}$ is live and bounded* **then** *$\mathcal{N}$ is strongly connected, consistent, and* $\text{rank}(\mathbf{C}) \leq |\text{SEQS}| - 1$.

2. **if** *$\mathcal{N}$ is conservative, consistent, and* $\text{rank}(\mathbf{C}) = |\text{SCCS}| - 1$ **then** *$\mathcal{N}$ is structurally live and structurally bounded.*

The condition in property 16.10.1 has been proven to be also sufficient for some subclasses of nets [13, 44, 45]. Observe that, even for structurally bounded nets, we do not have a complete characterization of structural liveness. Since |SCCS| $\leq$ |SEQS|, there is still a range of nets which satisfy neither the necessary nor the sufficient condition to be structurally live and structurally bounded! The added rank condition allows to state that the net in Figure 16.5.a is structurally non-live. Nevertheless, nothing can be said about structural liveness of the net in Figure 16.5.b.

Property 16.10 is purely structural (i.e., the initial marking is not considered at all). Nevertheless, it is clear that a too small initial marking (e.g. the empty marking) make non live any net structure. A less trivial lower bound for the initial marking based on marking linear invariants is based on fireability of every transition. If $t \in T$ is fireable at least once, for any p-semiflow $\mathbf{y}$, $\mathbf{y} \cdot \mathbf{m_0} \geq \mathbf{y} \cdot \mathbf{Pre}[P, t]$. Therefore:

Property 16.11 *If* $\langle \mathcal{N}, \mathbf{m_0} \rangle$ *is a live system, then* $\forall \mathbf{y} \geq 0$ *such that* $\mathbf{y} \cdot \mathbf{C} = 0$*,* $\mathbf{y} \cdot \mathbf{m_0} \geq \max_{t \in T}(\mathbf{y} \cdot \mathbf{Pre}[P, t]) \geq 1$

Unfortunately no characterization of liveness exists in linear algebraic terms. The net system in Figure 6.1.b adding a token to p_5 is consistent, conservative, fulfills the rank condition and all p-semiflows are marked, but it is non live.

16.3.4 Reversibility (and liveness)

Let us use now a *Liapunov-stability-like* technique to prove that the net system in Figure 6.8 is reversible. It serves to illustrate the use of marking linear invariants and some inductive reasonings to analyze liveness properties.

As a preliminary consideration that makes easier the rest of the proof, the following simple property will be used: Let $\langle \mathcal{N}, \mathbf{m_1} \rangle$ be a reversible system and $\mathbf{m_0}$ reachable from $\mathbf{m_1}$ (i.e., $\exists \sigma \in L(\mathcal{N}, \mathbf{m_1})$ such that $\mathbf{m_1} \xrightarrow{\sigma} \mathbf{m_0}$). Then $\langle \mathcal{N}, \mathbf{m_0} \rangle$ is reversible.

Assume $\mathbf{m_1}$ is like $\mathbf{m_0}$ (Figure 6.8), but making:$\mathbf{m_1}[\text{wait\_raw}] = \mathbf{m_1}[\text{empty}] = 0$, $\mathbf{m_1}[\text{wait\_dep.}] = 1$ and $\mathbf{m_1}[\text{object}] = 7$.

Let us prove first that $\langle \mathcal{N}, \mathbf{m_1} \rangle$ is reversible. Let $\mathbf{w}$ be a non-negative place weighting such that $\mathbf{w}[p_i] = 0$ iff p_i is marked in $\mathbf{m_1}$. Therefore, $\mathbf{w}[\text{wait\_dep.}] = \mathbf{w}[\text{R}] = \mathbf{w}[\text{object}] = \mathbf{w}[\text{wait\_with.}] = 0$ and $\mathbf{w}[p_j] > 0$ for all the other places. The function $\mathbf{v}(\mathbf{m}) = \mathbf{w} \cdot \mathbf{m}$ has the following properties: $\mathbf{v}(\mathbf{m}) \geq 0$ and $\mathbf{v}(\mathbf{m_1}) = 0$

For the system in Figure 6.8 a stronger property holds: $\mathbf{v}(\mathbf{m}) = 0 \iff \mathbf{m} = \mathbf{m_1}$. This can be clearly seen because $\mathbf{w} \cdot \mathbf{m} = 0 \iff \mathbf{m}[\text{wait\_raw}] = \mathbf{m}[\text{load}] = \mathbf{m}[\text{op}_1] = \mathbf{m}[\text{deposit}] = \mathbf{m}[\text{empty}] = \mathbf{m}[\text{op}_2] = \mathbf{m}[\text{wait\_free}] = \mathbf{m}[\text{unload}] = \mathbf{m}[\text{withdrawal}] = 0$. Even more, it is easy to check the following: $\mathbf{m_1}$ is the present marking $\iff t_9$ is the unique fireable transition.

If there exists (warning: in Liapunov-stability criteria the universal quantifier is used!) a finite firing sequence (i.e., a finite trajectory) per reachable marking $\mathbf{m}_i$ such that $\mathbf{m}_i \xrightarrow{\sigma_k} \mathbf{m}_{i+1}$ and $\mathbf{v}(\mathbf{m}_i) > \mathbf{v}(\mathbf{m}_{i+1})$, in a finite number of transition firings $\mathbf{v}(\mathbf{m}) = 0$ is reached. Because $\mathbf{v}(\mathbf{m}) = 0 \iff \mathbf{m} = \mathbf{m_1}$, a proof that $\mathbf{m_1}$ is reachable from any marking has been obtained (i.e, $\langle \mathcal{N}, \mathbf{m_1} \rangle$ is reversible).

Premultiplying the net state equation by $\mathbf{w}$ we obtain the following condition:if $\sigma_k = t_j$ then $[\mathbf{w} \cdot \mathbf{m}_{i+1} < \mathbf{w} \cdot \mathbf{m}_i] \iff \mathbf{w} \cdot \mathbf{C}[P, t_j] < 0$

Now, removing in Figure 6.8 the places marked at $\mathbf{m}_1$ (i.e., wait_dep., R, object, wait_with.) and fireable transitions (i.e., t_9) an acyclic net is obtained, so there exists an $\mathbf{w}$ such that $\mathbf{w} \cdot \mathbf{C}[P, t_j] < 0, \forall j \neq 9$.

For example, taking as weights the levels in the acyclic graph we have:

$$\mathbf{w}[\text{op}_1] = \mathbf{w}[\text{unload}] = 1 \tag{16.18}$$

$$\mathbf{w}[\text{load}] = \mathbf{w}[\text{wait\_free}] = 2 \tag{16.19}$$

$$\mathbf{w}[\text{wait\_raw}] = \mathbf{w}[\text{op}_2] = 3 \tag{16.20}$$

$$\mathbf{w}[\text{deposit}] = \mathbf{w}[\text{withdrawal}] = 4 \tag{16.21}$$

$$\mathbf{w}[\text{empty}] = 5 \tag{16.22}$$

and $\mathbf{w} \cdot \mathbf{C} = [-1, -1, -1, -1, -1, -1, -1, -1, +4, -1]$. In other words, the firing of any transition, except t_9, decreases $\mathbf{v}(\mathbf{m}) = \mathbf{w} \cdot \mathbf{m}$.

Using the algorithmic deadlock-freedom explanation in previous sections, the reversibility of $\langle \mathcal{N}, \mathbf{m}_1 \rangle$ is proven (observe that the p-invariants in Eqs (6.1-6.2-6.3-6.4) remain for $\mathbf{m}_1$):

```
if $\mathbf{m}[\text{load}] + \mathbf{m}[\text{op}_1] + \mathbf{m}[\text{deposit}] + \mathbf{m}[\text{op}_2] + \mathbf{m}[\text{unload}] + \mathbf{m}[\text{withdrawal}] \geq 1$
  then $\mathbf{v}(\mathbf{m})$ can decrease firing $t_2$,$t_3$,$t_5$,$t_6$,$t_8$ or $t_{10}$
  else if $\mathbf{m}[\text{wait\_raw}] + \mathbf{m}[\text{wait\_free}] \geq 1$
          then $\mathbf{v}(\mathbf{m})$ can decrease firing $t_1$or $t_7$
          else $\mathbf{v}(\mathbf{m})$ can decrease firing $t_4$ or $t_9$ is the unique fireable transition
               (iff $\mathbf{m}_1$ is the present marking)
```

Because $\mathbf{m}_0$ is reachable from $\mathbf{m}_1$ (e.g. firing $\sigma = (t_9 t_{10} t_6 t_7 t_8)^5 t_4 t_5$), $\langle \mathcal{N}, \mathbf{m}_0 \rangle$ is a reversible system.

Once again liveness of the system in Figure 6.8 can be proved, because the complete sequence (i.e. containing all transitions) $\sigma = t_1 t_2 t_3 t_4 t_5 t_9 t_{10} t_6 t_7 t_8$ can be fired. Since the system is reversible, no transition loses the possibility of firing (i.e., all transitions are live).

16.4 Siphons and traps

By means of graph theory based reasoning it is possible to characterize many properties of net subclasses. *Siphons* (also called *structural deadlocks*, or more simply *deadlocks*) and *traps* are easily recognizable subsets of places that generate very particular subnets.

Definition 16.12 *Let $\mathcal{N} = \langle P, T, F \rangle$ be an ordinary net.*

1. *A* siphon *is a subset of places, Σ, such that the set of its input transitions is contained in the set of its output transitions: $\Sigma \subseteq P$ is a siphon $\iff$ ${}^\bullet\Sigma \subseteq \Sigma^\bullet$.*

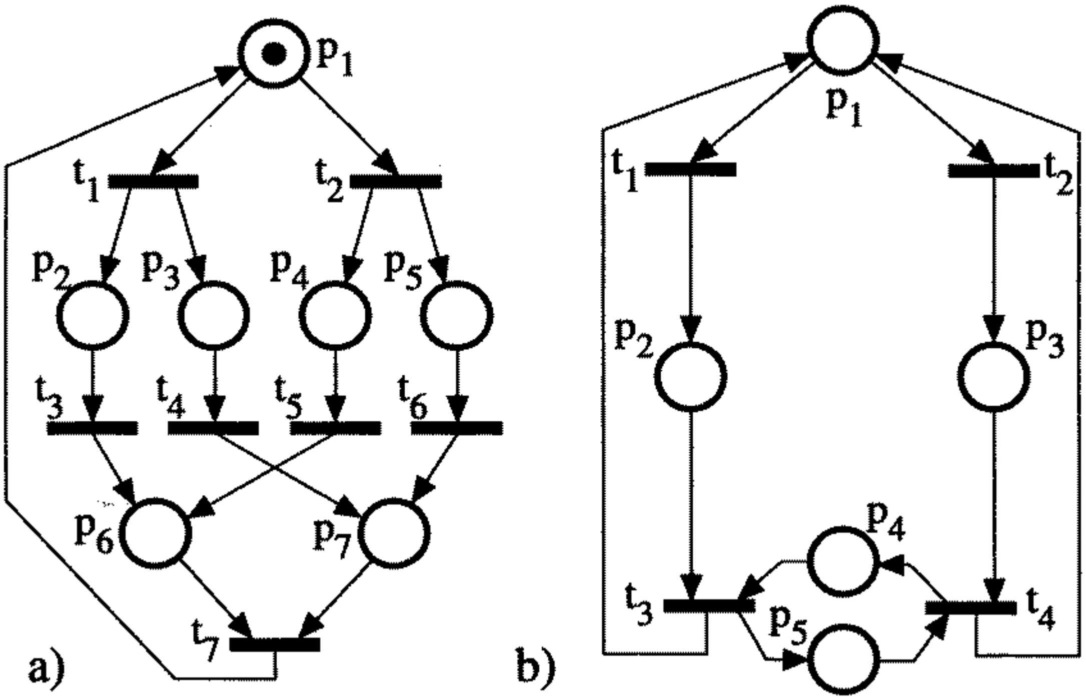


Figure 16.7: Two consistent and conservative free choice nets: (a) Structurally live rank(**C**) = 5, |EQS| = 5; (b) Structurally non-live rank(**C**) = 3, |EQS| = 2.

2. *A* trap *is a subset of places, θ, such that the set of its output transitions is contained in the set of its input transitions: $\theta \subseteq P$ is a trap $\Longleftrightarrow \theta^\bullet \subseteq {}^\bullet\theta$.*

$\Sigma = \{p_1, p_2, p_4, p_5, p_6\}$ is a siphon for the net in Figure 16.7.a: ${}^\bullet\Sigma = \{t_7, t_1, t_2, t_3, t_5\}$, while $\Sigma^\bullet = {}^\bullet\Sigma \cup \{t_6\}$. Σ contains a trap, $\theta = \Sigma \setminus \{p_5\}$. In fact θ is also a siphon (it is minimal: removing any number of places no siphon can be obtained).

Siphons and traps are reverse concepts: A subset of places of a net $\mathcal{N}$ is a siphon iff it is a trap on the reverse net, $\mathcal{N}^{-1}$ (i.e. that obtained reversing the arcs, its flow relation, F).

The following property "explains" why structural deadlocks or siphons (think on "soda siphons") and traps are the names of the above concepts.

Property 16.13 *Let $\langle\mathcal{N}, \mathbf{m_0}\rangle$ be an ordinary net system.*

1. *If $\mathbf{m} \in \mathrm{RS}(\mathcal{N}, \mathbf{m_0})$ is a deadlock state, then $\Sigma = \{p | \mathbf{m}[p] = 0\}$ is an unmarked (empty) siphon.*

2. *If a siphon is (or becomes) unmarked, it will remain unmarked for any possible net system evolution. Therefore all its input and output transitions are dead. So the system is not-live (but can be deadlock-free).*

3. *If a trap is (or becomes) marked, it will remain marked for any possible net system evolution (i.e. at least one token is "trapped").*

If a trap is not marked at $\mathbf{m_0}$, and the system is live, $\mathbf{m_0}$ will not be recoverable from those markings in which the trap is marked. Thus:

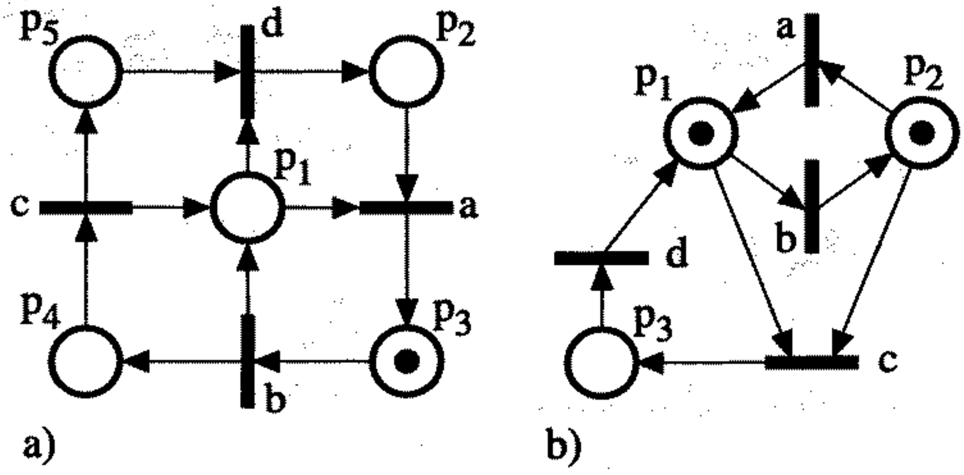


Figure 16.8: For the two nets, the MST-property does not hold, but: (a) The simple net is live and bounded; (b) The non-simple net is non-live (although deadlock-free) and bounded.

Corollary 16.14 *If a live net system is reversible, then* $\mathbf{m_0}$ *marks all traps.*

Remark For live and bounded free choice systems a stronger property holds: Marking all traps is a necessary and sufficient condition for reversibility [3]. The net system in Figure 16.7.a is reversible. Nevertheless, if $\mathbf{m_0} = [0, 1, 0, 0, 1, 0, 0]$, the new system is live and bounded but non reversible: The trap $\theta = \{p_1, p_3, p_4, p_6, p_7\}$ is not marked at $\mathbf{m_0}$.

A siphon which contains a marked trap will never become unmarked. So this more elaborate property can be of helpful for some liveness characterizations.

Definition 16.15 *Let* $\mathcal{N}$ *be an ordinary net. The system* $\langle \mathcal{N}, \mathbf{m_0} \rangle$ *has the* Marked-Siphon-Trap property, *MST-property, if each siphon contains a marked trap at* $\mathbf{m_0}$.

A siphon (trap) is *minimal* if it does not contain another siphon (trap). Thus, siphons in the above statement can be constrained to be minimal without any loss of generality.

The MST-property guarantees that all siphons will be marked. Thus no dead marking can be reached, according with property 16.13.1. Therefore:

Property 16.16 *If* $\langle \mathcal{N}, \mathbf{m_0} \rangle$ *has the MST-property, the system is deadlock-free.*

Figure 16.8 presents some limitations of the MST-property for liveness characterization.

Remark The MST-property is sufficient for liveness in simple net systems and necessary and sufficient for free-choice net systems. As a corollary, the *liveness monotonicity* result is true for the case of live free-choice systems: If $\langle \mathcal{N}, \mathbf{m_0} \rangle$ is a live free-choice system, then for all $\mathbf{m_0}' \geq \mathbf{m_0}$, $\langle \mathcal{N}, \mathbf{m_0}' \rangle$ is also live. The previous result does not apply to Simple Net systems. The system in Figure 6.1.b is simple, $\Sigma = \{p_1, p_2, p_7\}$ is a siphon ($^\bullet\Sigma = \{t_3, t_4, t_1\}$, $\Sigma^\bullet = {}^\bullet\Sigma \cup \{t_2\}$) that does not contain any trap. If we assume $\mathbf{m_0}[p_5] = 1$, t_2 can be fired and Σ becomes empty, leading to non-liveness.

16.5 Analysis of net subclasses

In this section we quickly overview some of the analytical results for certain subclasses that we define in Subsection 16.5.1. We organise the material around properties instead of describing the results for each subclass, what would lead to abundant redundancies. (Of course, properties of large subclasses such as EQ systems, are inherited by their subclasses such as FC or DF systems.)

Our intention is to show how the restrictions imposed by subclasses' definitions, at the price of losing some modelling capabilities, facilitate the analysis. The designer must find a compromise between modelling power and availability of powerful analysis tools, while one of the theoretician's goals is obtaining better results for increasingly larger subclasses.

The general idea behind the structure theory of net subclasses is to investigate properties that every net system in the subclass enjoys, instead of analysing each particular system. These general properties are useful in two ways:

- The designer knows that her/his system (if it belongs to an appropriate subclass) behaves "well" (e.g., liveness monotonicity, existence of home states).

- General analysis methods become more applicable or more conclusive (e.g., model checking for FC, liveness analysis for all the subclasses considered).

The technical development of the presented results, and many other details that are out of the scope of this very succint presentation, can be found in [15], [36], [41], [45].

16.5.1 Some Syntactical Subclasses

Historically, subclasses of ordinary nets have received special attention because powerful results were early obtained for them. In this presentation some of them appear as subclasses of their weighted generalisations for the sake of concision. Regarding the modelling power, clearly some subclasses have less than others if the former are properly included in the latter. Also the weighted generalisations have more modelling power than their ordinary counterparts since, in general, the ordinary implementations of weights do not preserve the (topological) class membership.

Join-free and State Machines

A P/T net $\mathcal{N}$ is *join-free (JF)* when no transition is a join, i.e., $|^\bullet t| \leq 1$ for every t. With these nets, proper synchronisations cannot be modelled. $\mathcal{N}$ is a *weighted P-net* when every transition has one input and one output place, i.e., $|{}^\bullet t| = |t^\bullet| = 1$ for every t. An ordinary weighted P-net is a P-net or *state machine (SM)*, a name due to the fact that when marked *with only one token* each place represents a possible global state of the (sequential) system. With more than one token concurrency appears: an SM with k tokens represents k instances of the same sequential process evolving in parallel. Given an adequate stochastic interpretation, strongly connected SM correspond to closed Jackson Queueing Networks.

Distributor-free and Marked Graphs

A P/T net $\mathcal{N}$ is *distributor-free (DF)* when no place is a distributor, i.e., $|p^\bullet| \leq 1$ for every p. With these nets, conflicts cannot be modelled. They are also called *structurally persistent* because the structure enforces persistency, that is, the property that a transition can only be disabled by its own firing. $\mathcal{N}$ is a *weighted T-net* when every place has one input and one output transition, i.e., $|{}^\bullet p| = |p^\bullet| = 1$ for every p. An ordinary weighted T-net is a T-net or *marked graph (MG)*, a name due to a representation as a graph where the nodes are the transitions and the arcs joining them are marked (that is, places have been obviated). As some examples, MG can model activity ordering systems, generalising PERT graphs, job-shop systems with fixed production routing and machine sequencing, flow lines, Kanban systems, etc. For instance, the net in Figure 16.9 (a) is a MG. Given an adequate stochastic interpretation, strongly connected MG correspond to Fork/Join Queueing Networks with Blocking.

Equal Conflict and Free Choice

A P/T net $\mathcal{N}$ is *equal conflict (EQ)* when every pair of transitions in structural conflict are in equal conflict, i.e., they have the same pre-incidence function: ${}^\bullet t \cap {}^\bullet t' \neq \emptyset$ implies $\mathbf{Pre}[P, t] = \mathbf{Pre}[P, t']$. An ordinary EQ net is an *(extended) free choice net (FC)*. Free choice nets play a central role in the theory of net systems because there are powerful results for their analysis and synthesis while they allow the modelling of systems allowing both conflicts and synchronisations. It is often said that FC can be seen as MG enriched with SM-like conflicts or, equivalently, SM enriched with MG-like synchronisations. However, they cannot model mutex semaphores or resource sharing, for instance. The net in Figure 16.9 (b) is FC. The fundamental property of EQ systems is that whenever a marking enables some transition t, then it enables every transition in $\mathrm{EQS}(t) = \mathrm{CCS}(t)$. It can be said that the structural and behavioural notions of conflict coincide. It is also said that conflicts and synchronisations are neatly separated, because it is easy to transform the net so that no output of a distributor place is a join: Figure 16.6 (f) is the result of transforming (e).

Asymmetric Choice, or Simple

A P/T net $\mathcal{N}$ is *asymmetric choice (AC)*, sometimes called *Simple*, when it is ordinary and $p^\bullet \cap p'^\bullet \neq \emptyset$ implies $p^\bullet \subseteq p'^\bullet$ or viceversa. In these nets, the

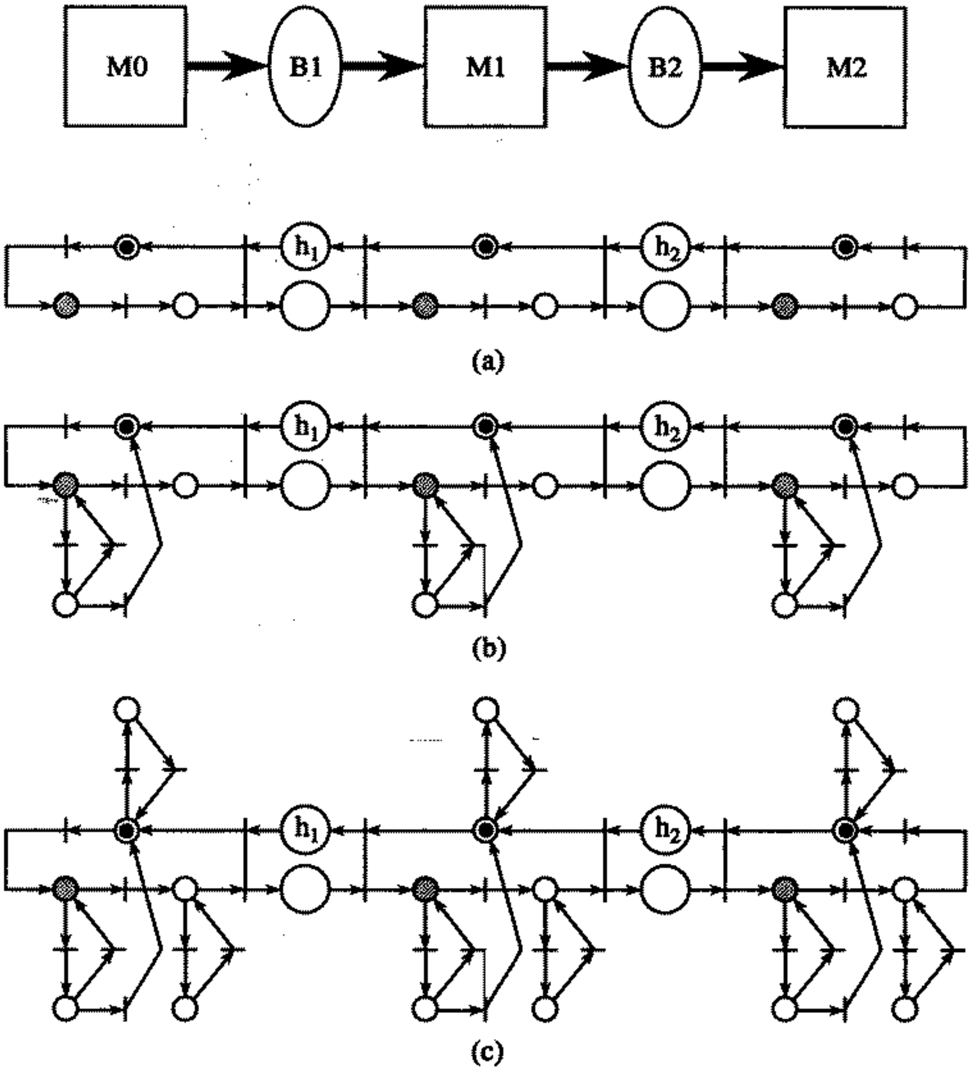


Figure 16.9: Modelling a flow line with three machines and two buffers. Each buffer is modelled by two places, for the parts and "holes", respectively (the later initially marked with h_i holes). Each machine is modelled by a state machine, initially idle, where the "working-state" is shaded; they follow a blocking after service policy (they start their work even if there are no holes in the output buffer, so they might stay blocked before unloading). The different models consider: (a) reliable machines, (b) machines with operation dependent failures (may fail only when working), and (c) machines with time dependent failures (may fail at any time). Scrapping (part is discarded) is possible in the case of unreliable machines.

conflict relation is transitive. They generalise FC, and allow modelling to a certain extent resource sharing. The net in Figure 16.9 (c) is AC.

The above subclasses are defined through a global constraint on the topology. Their relations are illustrated in the graph of Figure 16.10, where a directed arrow connecting two subclasses indicates that the source properly includes the destination, and the constructs depicted illustrate the typical situations that distinguish each subclass.

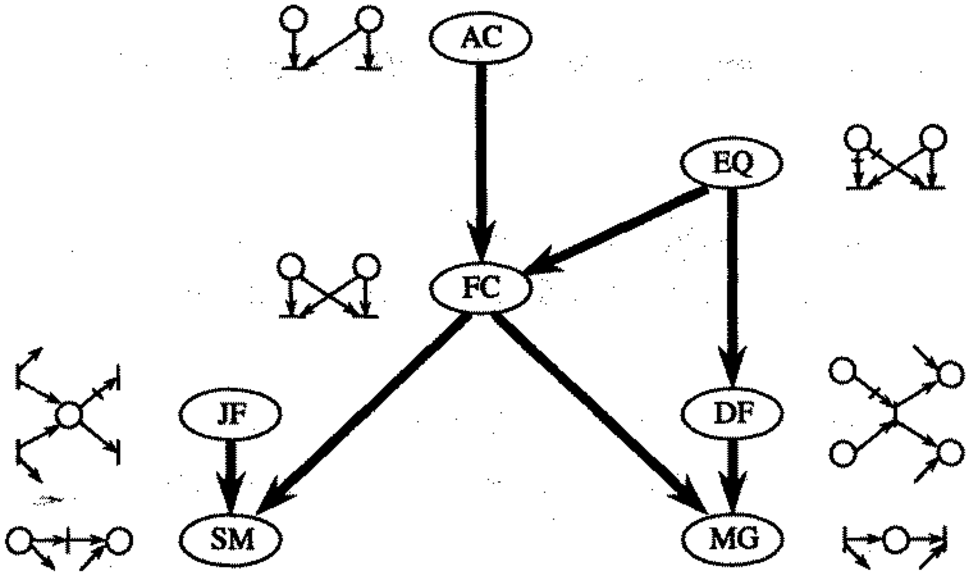


Figure 16.10: Relations between some basic syntactical subclasses.

Modular Subclasses

Subclasses can also be defined in a modular way, by giving some modules and how interconnecting them. Very often the modules are monomarked SM, representing sequential systems which run in parallel communicating in a restricted fashion. A few examples follow.

Superposed automata systems (SA) are composed by monomarked SM synchronised by transition merging, that is, via rendez-vous. They lead to general — although structured — bounded systems models. For instance, all the nets in Figure 16.9 are SA (if the capacities of the buffers are one).

Systems of buffer-cooperating functional entities are modules (depending on the kind of modules we obtain different subclasses) synchronised by message-passing through buffers in a restricted fashion. A P/T system $\mathcal{S}$ is in this class when:

- $P = B \uplus \biguplus_i P_i$, $T = \biguplus_i T_i$. The net systems $\mathcal{S}_i$ generated by P_i and T_i are the *functional entities* or modules, and the places of B are the *buffers*.
- For every $b \in B$, there exists i such that $b^\bullet \in T_i$, that is, buffers are output private. Moreover if $t, t' \in T_i$ are in EQ relation in $\mathcal{N}_i$, then $\mathbf{Pre}[b, t] = \mathbf{Pre}[b, t']$, that is, buffers do not modify the EQ relations of the modules. These restrictions on buffers prevent competition.

In case the modules are monomarked SM we obtain *deterministically synchronised sequential processes (DSSP)*. In case they are EQ systems we obtain systems of cooperating EQ systems. These can be buffer-interconnected again, leading to a hierarchical class of systems, recursively defined, that is called $\{SC\}^*EQS$, standing for systems of cooperating systems of cooperating ... EQ sytems. They allow the modelling of hierarchically coupled cooperating systems. The net systems in Figure 16.9 (a) and (b) can be seen as — rather trivial —

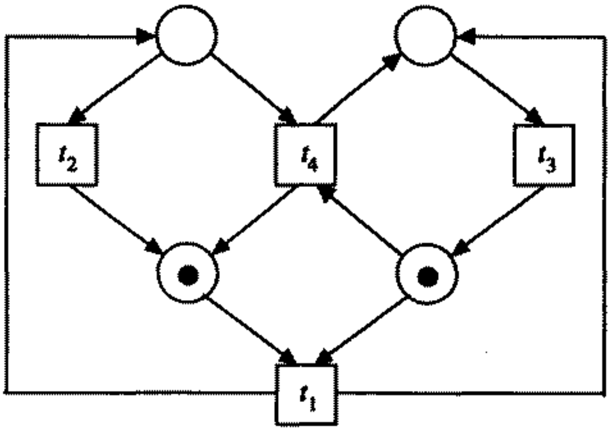


Figure 16.11: A net system where local fairness does not guarantee impartiality, and which can exhibit monopoly situations.

examples of buffer-cooperating systems, where the places modelling the buffers are precisely the buffers, while each machine is modelled by an SM.

Systems of Simple Sequential Processes with Resources (S^3PR) are SM synchronised by a restricted resource sharing. The restrictions impose that there is a place in each SM which is contained in every cycle and does not use any resource (an "unavoidable idle state"), and that every other place uses one (possibly shared) resource. They allow the modelling of rather general flexible manufacturing systems, or similar systems where resource sharing is essential, as it shall be shown in Chapter **??**

16.5.2 Fairness and monopolies

In some systems, *impartiality* (or *global fairness*, that is, every transition appears infinitely often in infinite sequences) can be achieved *locally* (every solution of a — local — conflict that is effective infinitely often is taken infinitely often):

Theorem 16.17 *Let S be a bounded strongly connected EQ system or DSSP. A sequence $\sigma \in \mathrm{L}(S)$ is globally fair iff it is locally fair.*

This property is not true in general. Take for instance the net system in Figure 16.11. The sequence $\sigma = \{t_1\, t_2\, t_3\}^\omega$ is locally fair (actually, during the occurrence of σ no conflict is effective at all), but it is not globally fair since t_4 never occurs. Conversely, the sequence $\sigma = \{t_1\, t_3\, t_4\, t_3\, t_1\, t_2\, t_3\}^\omega$ is globally fair but not locally since whenever t_2 and t_4 are in conflict t_4 wins.

The equivalence of local and global fairness has two important consequences. The first one is equivalence of liveness and deadlock-freeness, what facilitates the analysis of liveness because it suffices to check the weaker property of deadlock-freeness:

Theorem 16.18 *Let S be a bounded strongly connected EQ system or DSSP. Then S is live iff it is deadlock-free.*

The second consequence is relevant for the eventual interpretation of the model. Assume, for instance, that the system in Figure 16.11 is interpreted so that transitions occur after a deterministic delay equal to their index. Then, the system behaves repeating the occurrence of $t_1\, t_2\, t_3$, never giving a chance to t_4, despite it was perfectly live in the autonomous model: the interpretation has destroyed liveness leading to a *monopoly* situation (the "resources" needed by t_4 are "monopolized" by t_2).

This can never happen to a bounded strongly connected EQ system or DSSP, assuming the interpretation allows progress (i.e., a transition that is continuously enabled eventually occurs): by imposing a fair conflict resolution policy, which can be done in a distributed fashion provided structurally conflicting transitions are allocated together, it is guaranteed that no action in the system becomes permanently disabled if the autonomous model was live.

16.5.3 Confluence and directedness

Persistent systems, which include structurally persistent ones (DF) enjoy a strong *confluence* property: whenever from a given marking we reach two different markings by firing two distinct sequences, then we can complete both such sequences, each with the firings left with respect to the other, reaching in any case the same marking [28]. Confluence is closely related to determinacy [27]: interpreting sequences as executions and transition occurrences as operations, when from a given point two different executions may occur, depending on operation times or other external matters, each operation in one execution will eventually occur in the other (assuming progress), possibly in a different order and with a different timing.

Moreover, confluence facilitates checking liveness (non-termination) of persistent systems: it suffices to find a repeatable sequence that contains every transition. This is because such a repeatable sequence allows to construct a sequence greater than any given sequence σ fireable from the initial marking, and this proves that σ can be continued to enable the repeatable sequence.

Stepping out from persistent systems, the presence of effective conflicts may destroy confluence. *Directedness* is a weaker property that states that a common successor of arbitrary reachable markings always exist, and which holds for some subclasses:

Theorem 16.19 *Let $\mathcal{S}$ be a live EQ system or DSSP. Let $\mathbf{m_a}, \mathbf{m_b} \in \mathrm{RS}(\mathcal{S})$. Then $\mathrm{RS}(\mathcal{N}, \mathbf{m_a}) \cap \mathrm{RS}(\mathcal{N}, \mathbf{m_b}) \neq \emptyset$.*

Informally, directedness means that the effect of a particular resolution of a conflict is not "irreversible": there is a point where the evolution joints with that which would have been if the decision had been other. The existence of *home states*, i.e., states that can be ultimately reached after whichever evolution, follows from directedness and boundedness:

Theorem 16.20 *Live and bounded EQ systems or DSSP have home states.*

The system in Figure 6.3.b is an example of a live and 1-bounded system without home states.

This is an important property for many reasons:

- The system is known to have states to return to, which is often required in reactive systems. Chosing one such state as the initial one makes the system *reversible*, i.e., $\mathbf{m_0}$ can always be recovered.

- Model checking is largely simplified, since there is only one terminal strongly connected component in the reachability graph.

- Under a Markovian interpretation (e.g., as in *generalized stochastic Petri nets* [1]), *ergodicity* of the marking process is guaranteed; otherwise, simulation or computation of steady state performance indices could be meaningless.

16.5.4 Reachability and the state equation

As it was discussed in Section 16.3, reachable markings are solutions to the state equation but, in general, not conversely: some solutions of the state equation may be "spurious". This limits the use of the state equation as a convenient algebraic representation of the state space.

Fortunately stronger relations between reachable markings and solutions to the state equation are available for some subclasses:

Theorem 16.21 *Let $\mathcal{S}$ be a P/T system with reachability set* RS *and linearised reachability set wrt. the state equation* $\mathrm{LRS}^{\mathrm{SE}}$.

1. *If $\mathcal{S}$ is a live weighted T-system, or a live and consistent source private DSSP, then* $\mathrm{RS} = \mathrm{LRS}^{\mathrm{SE}}$. *Moreover, if it is a live MG, then the integrality constraints can be disregarded.*

2. *If $\mathcal{S}$ is a bounded, live, and reversible DF system, then $\mathbf{m} \in \mathrm{RS}$ iff $\mathbf{m} \in \mathrm{LRS}^{\mathrm{SE}}$ and the unique minimal T-semiflow of the net is fireable at $\mathbf{m}$.*

3. *If $\mathcal{S}$ is a live, bounded, and reversible FC system, then $\mathbf{m} \in \mathrm{RS}$ iff $\mathbf{m} \in \mathrm{LRS}^{\mathrm{SE}}$ (integrality constraints can be disregarded) and every trap is marked at $\mathbf{m}$.*

4. *If $\mathcal{S}$ is a live EQ system or a live and consistent DSSP, and $\mathbf{m_a}, \mathbf{m_b} \in \mathrm{LRS}^{\mathrm{SE}}$, then $\mathrm{RS}(\mathcal{N}, \mathbf{m_a}) \cap \mathrm{RS}(\mathcal{N}, \mathbf{m_b}) \neq \emptyset$.*

We can take advantage of the above statements in a diversity of situations. For instance, the reachability characterisation for live MG allows to analyse some of their properties through linear programming. Even the last, and weakest, statement in the above theorem — a directedness result at the level of the linearised reachability graph — can be very helpful: since, in particular, it implies that there are no spurious deadlocks in live EQ systems, or live and consistent DSSP, the deadlock-freeness analysis technique presented in Subsection 16.3.2 — which in these cases requires a single equation system — allows to decide the property.

Figure I.2.3 shows an example of a live and 1-bounded system with spurious deadlocks.

16.5.5 Analysis of liveness and boundedness

One of the properties that supports the claim that "good" behavior should be easier to achieve in some subclasses than in general systems is liveness monotonicity wrt. the initial marking. This means that liveness, provided that the net is "syntactically" correct as we shall precise later, is a matter of having enough tokens in the buffers (customers, resources, initial data, etc.), differently to what happens in general systems where the addition of tokens may well cause deadlocks due to poorly managed competition. For instance, in the net system of Figure 6.1.b adding a token (in p_5) to the initial marking destroys liveness.

Theorem 16.22 *Let* $\langle \mathcal{N}, \mathbf{m_0} \rangle$ *be a live EQ system or DSSP. The EQ system or DSSP* $\langle \mathcal{N}, \mathbf{m_0} + \Delta\mathbf{m_0} \rangle$*, where* $\Delta\mathbf{m_0} \geq \mathbf{0}$ *is live too.*

Very often, a net system is required to be live and bounded. As we saw in Section 6.3.1 the verification of liveness can be very hard, so we want to avoid it when possible. In some cases we are able to decide using structural methods alone; in other cases we can characterise the nets that can be lively and boundedly marked, so the costful enumeration analysis needs to be used only when there is a chance of success.

Theorem 16.23 *Let* $\mathcal{N}$ *be an EQ or DSSP net. A marking* $\mathbf{m_0}$ *exists such that* $\langle \mathcal{N}, \mathbf{m_0} \rangle$ *is a live and bounded EQ system or DSSP iff* $\mathcal{N}$ *is strongly connected, conservative (or consistent), and* $\mathrm{rank}(\mathbf{C}) = |\mathrm{SEQS}| - 1$*. Moreover, in EQ systems, liveness of the whole system is equivalent to liveness of each P-component (the P-subnets generated by the minimal P-semiflows).*

Particular cases of the above result are well-known in net theory. For instance, in the ordinary case, the P-components of a FC net are strongly connected SM, which are live iff they are marked, so the liveness criterion can be stated as "there are no unmarked P-semiflows". In the case of MG, which are always consistent and $\mathrm{rank}(\mathbf{C}) = |\mathrm{SEQS}| - 1 = |T| - 1$, the existence of a live and bounded marking is equivalent to strong connectedness. Since their P-components are their circuits, liveness can be checked removing the marked places and verifying that the remaining net is acyclic.

16.6 Invariants and Reductions for Coloured Petri nets

16.6.1 Invariants

As in ordinary Petri nets, one of the main aspects of the structural verification of CPNs is the generation of invariants. But before developping techniques for such a problem, some points must be clarified:

- how can we express a invariant of CPNs and especially a linear invariant ?
- how a family of (linear) invariants can be characterized as a generative family of invariants ?
- how can we build a (generative) family of (linear) invariants ?

The aim of this section is to successively answer to these three questions in a concise way.

Presentation of linear invariants

The choice of an adequate definition of invariants should meet the following requirements. At first, an invariant of an high-level Petri net must be a high-level invariant. For instance, if we model processes by colours, it means that an invariant should express properties of the behaviour of one particular process but also of the behaviour of any process, or else of the behaviour of any process except a particular one etc. On the other hand, the definition should enable mathematical developments leading to efficient algorithms for computing invariants.

One can try to follow the definition of ordinary invariants, i.e.: a invariant is a weighted sum of the marking of the places left invariant by the firing of any transition. However such a definition involves two hidden extensions :

- what can be the weights on place marking ?
- there are multiple ways to fire a transition (as much as the size of the colour domain of the transition)

The essential point in the definition below is that the weights are colour functions. It can be interpreted as applying the colour function on the place marking corresponds to extract the relevant part of information contained in this marking for a given invariant. In order to be mathematically sound this function must have for domain the colour domain of the place and for codomain a common domain for the weights of the same invariant. This codomain may be viewed as the interpretation domain of the invariant and this requirement ensures that the weighted sum of the marking places is well defined.

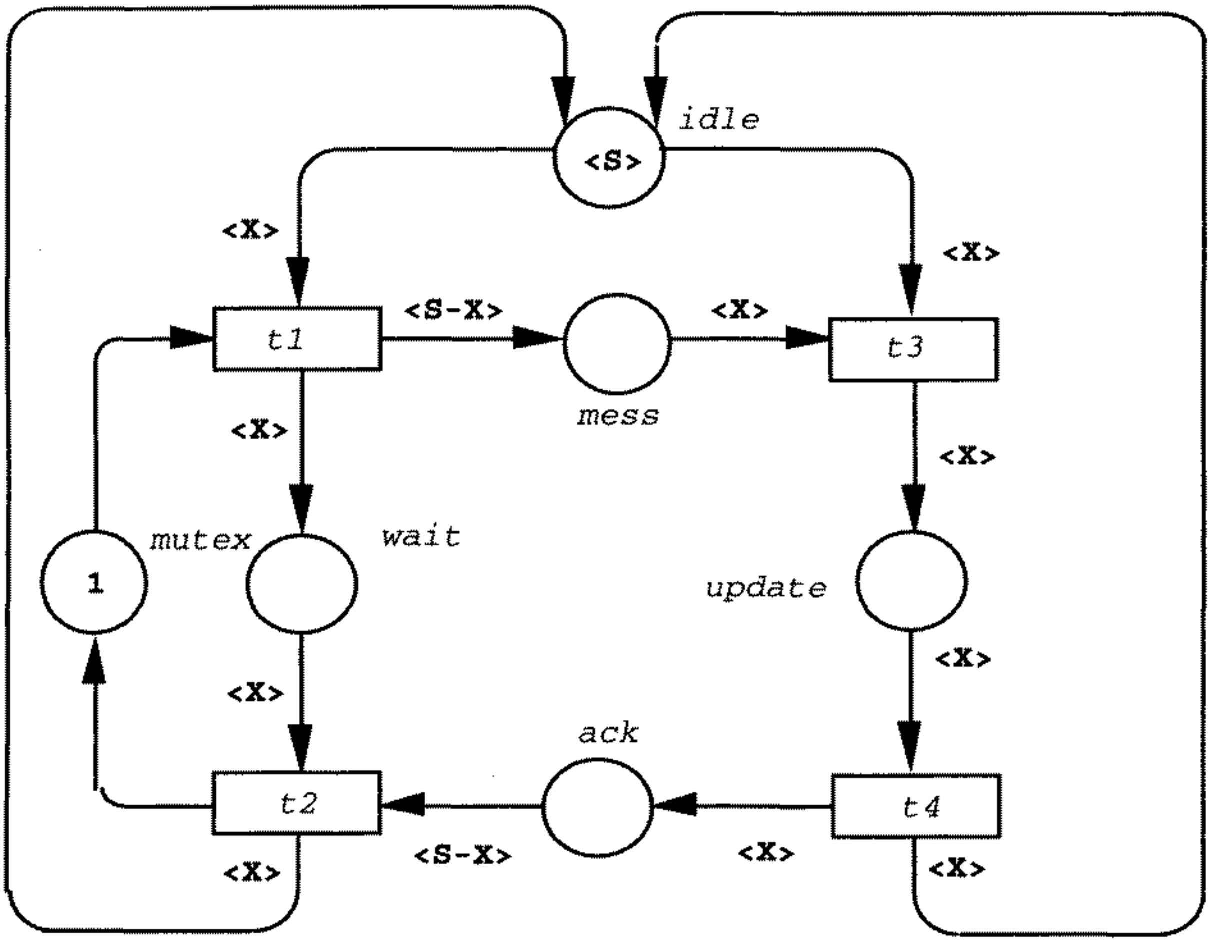


Figure 16.12: A CPN model of a replicated database

Definition 16.24 *A linear invariant* $\mathbf{v}$ *of a coloured Petri net* $\mathcal{N}$ *is defined by :*

- *cd*($\mathbf{v}$) *the colour domain of the invariant,*
- $\forall p \in P$, $\mathbf{v}(p)$ *a function from* $\mathbb{Z}^{cd(p)}$ *to* $\mathbb{Z}^{cd(\mathbf{v})}$,

such that :

$$\forall \mathbf{m} \textit{ reachable marking}, \sum_{p \in P} \mathbf{v}(p)(\mathbf{m}(p)) = \sum_{p \in P} \mathbf{v}(p)(\mathbf{m}_0(p))$$

The net of figure 16.12 models a database management with multiple copies. The access grant of the database is centralized and submitted to the mutual exclusion.

The database is shared by a set of sites represented by the colour domain *Sites*. In order to modify the database an idle site (in place *idle*) must get the grant (a neutral token in place *mutex*) and once it has modified the file, it sends messages to the others sites. The whole action is modelled by the transition t_1 and the contents of the message is not modelled. Then the other sites update their own database (transition t_3) and send an acknowledgment (transition t_4).

Once the active site has received all the acknowledgments, it releases the grant (transition t_2).

In order to simplify the net, accessing and modifying the database is modelled by a single transition (indivisible step) while the updating of the other sites is modelled by a place (divisible step).

Initially there is a token per site in place *idle* and a neutral token in place *mutex*.

Let us give a first example of a linear invariant :

$$\begin{array}{lcl} cd(\mathbf{v}) & = & Sites \\ \mathbf{v} & = & \langle x\rangle.idle + \langle x\rangle.wait + \langle x\rangle.update \end{array}$$

This linear invariant describes the behaviour of any site : either the site is idle, either it waits for the acknowledgments or it updates its database.

Computation of linear invariants

Now we focus on the computation of family of linear invariants. Ideally, we want a generative family of invariants i.e. any linear invariant should be obtained by a linear combination of the invariants of the family. Keeping in mind what is a linear invariant of a well-formed net, we allow the coefficients of the combination to be functions with the same requirements on domain and codomain for these functions. It should be mentionned that this is the only way to keep a generative family of reasonnable size and moreover as we shall see in what follows to obtain significant flows (directly like items of the family or by linear combination).

We will not give in this subsection any algorithm but instead we will show on the previous example how to compute a generative family of invariants. Then we will interpret our family of invariants.

The corner stone of all the algorithms is to handle the incidence matrix in a similar way as it is done in the Gaussian elimination. However the elimination rules must be applied under conditions which ensure that no linear invariant will be "forgotten".

We start with the incidence matrix $\mathbf{C}$ of our example :

$$\mathbf{C} = \begin{array}{cc} \begin{array}{cccc} t1 & t2 & t3 & t4 \end{array} & \\ \left(\begin{array}{cccc} \langle x\rangle & -\langle x\rangle & 0 & 0 \\ -\langle x\rangle & \langle x\rangle & -\langle x\rangle & \langle x\rangle \\ \langle s-x\rangle & 0 & -\langle x\rangle & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & \langle x\rangle & -\langle x\rangle \\ 0 & -\langle s-x\rangle & 0 & \langle x\rangle \end{array}\right) & \begin{array}{l} \langle x\rangle.wait \\ \langle x\rangle.idle \\ \langle x\rangle.mess \\ 1.mutex \\ \langle x\rangle.update \\ \langle x\rangle.ack \end{array} \end{array}$$

On the right, the initial family of invariants is shown i.e. each place weight by the identity function of its colour domain. Now we proceed by a standard rule : adding to a line another one which has been premultiplied by a function.

the only restriction with this rule is the consistency of domains and codomains of functions. The result is shown below :

$$\begin{array}{ccccl}
t1 & t2 & t3 & t4 & \\
\langle x\rangle & -\langle x\rangle & 0 & 0 & \langle x\rangle.wait \\
0 & 0 & -\langle x\rangle & \langle x\rangle & \langle x\rangle.idle + \langle x\rangle.wait \\
0 & \langle s-x\rangle & -\langle x\rangle & 0 & \langle x\rangle.mess - \langle s-x\rangle.wait \\
0 & 0 & 0 & 0 & 1.mutex + 1.wait \\
0 & 0 & \langle x\rangle & -\langle x\rangle & \langle x\rangle.update \\
0 & -\langle s-x\rangle & 0 & \langle x\rangle & \langle x\rangle.ack
\end{array}$$

We apply on this new matrix a second rule which eliminates a line for which one of its coefficient is the only non nul coefficient of its column (here the first line). We require that this coefficient is an injective mapping.

$$\begin{array}{ccccl}
t1 & t2 & t3 & t4 & \\
0 & 0 & -\langle x\rangle & \langle x\rangle & \langle x\rangle.idle + \langle x\rangle.wait \\
0 & \langle s-x\rangle & -\langle x\rangle & 0 & \langle x\rangle.mess - \langle s-x\rangle.wait \\
0 & 0 & 0 & 0 & 1.mutex + 1.wait \\
0 & 0 & \langle x\rangle & -\langle x\rangle & \langle x\rangle.update \\
0 & -\langle s-x\rangle & 0 & \langle x\rangle & \langle x\rangle.ack
\end{array}$$

The third rule we apply is identical to a Gaussian elimination rule i.e. delete a null column (here *t1*).

$$\begin{array}{cccl}
t2 & t3 & t4 & \\
0 & -\langle x\rangle & \langle x\rangle & \langle x\rangle.idle + \langle x\rangle.wait \\
\langle s-x\rangle & -\langle x\rangle & 0 & \langle x\rangle.mess - \langle s-x\rangle.wait \\
0 & 0 & 0 & 1.mutex + 1.wait \\
0 & \langle x\rangle & -\langle x\rangle & \langle x\rangle.update \\
-\langle s-x\rangle & 0 & \langle x\rangle & \langle x\rangle.ack
\end{array}$$

We can iterate this process for eliminating the column *t2* using the property that $\langle s-x\rangle$ is injective.

$$\begin{array}{cccl}
t2 & t3 & t4 & \\
0 & -\langle x\rangle & \langle x\rangle & \langle x\rangle.idle + \langle x\rangle.wait \\
0 & -\langle x\rangle & \langle x\rangle & \langle x\rangle.mess - \langle s-x\rangle.wait + \langle x\rangle.ack \\
0 & 0 & 0 & 1.mutex + 1.wait \\
0 & \langle x\rangle & -\langle x\rangle & \langle x\rangle.update
\end{array}$$

The elimination of *t3* ends the algorithm as it eliminates at the same time *t4*. We obain the following family of invariants :

- The state of any site : either the site is idle, either it waits for the acknowledgments or it updates its database.

$$\langle x\rangle.wait + \langle x\rangle.idle + \langle x\rangle.update$$

- The state of the database : either the grant is present or a site is waiting for completing its transaction.

$$1.wait + 1.mutex$$

- The synchronization between sites : if a site is waiting then either any other site has a message for the current transaction, either it updates its copy, or it has send its acknowledgment.

$$\langle x\rangle.mess + \langle x\rangle.ack + \langle x\rangle.update - \langle s - x\rangle.wait$$

Some other significant linear invariants may also be obtained by combining the previous invariants. For instance, the following invariant says that either the grant is present and all the sites are idle or the grant is absent and the idle sites are exactly those who will receive a message or have sent their acknowledgment.

$$\langle x\rangle.idle - \langle s\rangle.mutex - \langle x\rangle.mess - \langle x\rangle.ack$$

Additional remarks

The computation of linear invariants we have described in the previous section can not be straightforwardly extended to general coloured nets. We emphasize below the three problems one must cope to provide a general algorithm :

- How to compose lines in order to cancel items of the matrix ?
- How to ensure that the last coefficient of a column is injective ?
- How to handle the two previous operations in a parametrized way i.e. independantly of the size of the colour domains (in our example, the number of sites) ?

A general algorithm for coloured nets has been proposed in [12]. The key point of the algorithm is the intensive use of generalized semi-inverses. Its only restriction is to fix the size of the colour domains.

On the other hand, with restrictions to subclasses of coloured nets, it is possible to obtain algorithms which handle the parameters [25], [24] [32]. These algorithms transform the incidence matrix of functions into a set of matrices with coefficients taken in a ring of poynomials. The variables correspond to the parameters of the net. At this point, it is enough to apply a Gaussian-like elimination on these matrices. Each vector solution is finally transformed (in an inverse way) to a linear invariant.

16.6.2 Reductions

A reduction of nets is defined by some conditions of application and a method of transformation such that the behaviour of the reduced net has the same behaviour as the original one w.r.t. to generic properties if the orignal net fulfills the conditions. Reductions theory has been mainly developed by Gérard Berthelot [2] who has proposed ten reductions covering a wide area of applications cases. Introducing a new reduction is interesting if :

- it covers a frequent behavioural situation (as will be described in the next subsection),
- the application conditions can be checked in a efficient way (e.g. by examination of the net structure or by linear invariant computation).

The generalization to high-level nets has been proposed by different authors [9] , [21] and [23]. We will follow the last reference to introduce some reductions. Then we will illustrate them on the example of the database management and at last we will discuss about a methodology to define new reductions.

Some reductions

We limit ourselves to two reductions for CPNs (more information may be found in the previous reference) which are enough for our example. The style of definition will be unformal : the general interpretation of the reduction, the definition of application conditions and the method of transformation with for each item a corresponding interpretation. The set of properties preserved by these reductions is essentially the same one for the two reductions and covers liveness, boudedness, home state existence, unavoidable state existence, etc.

IMPLICIT PLACE SIMPLIFICATION

The reduction deletes a place which never forbids alone the firing of transition. Such a place occurs in two cases : either it is a redundant place which explicitely models an information implicitely contained in the other places or the place was not originally implicit but other reductions has transformed it.

The existence of an implicit place is ensured by a particular invariant. Such reduction illustrates an indirect use of the invariants for the model verification.

Definition 16.25 *Let $\mathcal{N}$ be a coloured Petri net, a place p is implicit iff :*

1. *there exists a linear invariant* $\mathbf{v}$ *such that :*
   - $cd(\mathbf{v}) = cd(p)$
   - $\mathbf{v}(p)$ *is the identity function of* $\mathrm{Bag}(cd(p))$
   - $\forall p' \neq p, \forall c \in cd(p'), -\mathbf{v}(p')(c) \in \mathrm{Bag}(cd(p'))$
2. $\forall t \in T, \forall c \in cd(t), \sum_{p' \in P} \mathbf{v}(p')(\mathbf{Pre}(p', t)(c)) \geq \sum_{p' \in P} \mathbf{v}(p')(\mathbf{m_0}(p'))$

The condition *2* ensures that in the initial marking, p will not forbid alone the firing of any transition due to the positivity constraints included in the condition *1*. The fact that **v** is a linear invariant ensures that the condition *2* is also true for all reachable markings.

The transformation deletes the place and the bordering arcs.

Definition 16.26 *The reduced net $\mathcal{N}_r = \langle P', T', \mathbf{Pre}', \mathbf{Post}', C', cd' \rangle$ with initial marking $\mathbf{m_0}'$ obtained from the net $\mathcal{N}$by the simplification of the implicit place p is defined by :*

- $P' = P - \{p\}$
- $T' = T$
- $C' = C$
- $\forall t \in T', \forall p' \in P', cd(t) = cd(t)$ *and* $cd(p') = cd(p)$
- $\forall t \in T', \forall p' \in P', \mathbf{Pre}'(p', t) = \mathbf{Pre}(p', t)$ *and* $\mathbf{Post}(p', t) = \mathbf{Post}(p', t)$
- $\forall p' \in P', \mathbf{m_0}'(p') = \mathbf{m_0}(p')$

POST-AGGLOMERATION OF TRANSITIONS

The principle of the post-agglomeration is the following. Suppose we are given H a set of transitions which represent global actions and which lead to an intermediate state representated by a token in the place p. Suppose that the way of going out of this intermediate state is to fire a transition f which represents a local transition (i.e. without synchronization). Then the firing of any transition of H could be immediatly followed by the firing of f without modifying the global behaviour of the net. Doing this, the place p may be deleted and the firings of transition f may be included in the firing of any transition of H. However one must take some care in order to define the new items of the **Pre** and the **Post** matrices. For simplicity, we present here a restricted version of this reduction.

We begin by introducing the concept of a safe colour function required by the next definition. A safe function produces (or consumes depending on the arc) at most one token per colour of the place. In a modelling, the functions are the most of the time safe functions.

Definition 16.27 *A safe colour function from* Bag(E) *to* Bag(F) *is safe iff :*

$$\forall c \in E, \forall c' \in F, f(c)(c') \leq 1$$

Definition 16.28 *Let $\mathcal{N}$be a coloured Petri net, p a place which has for output the transition f et for inputs the set H of transitions with $f \notin H$, one can post-agglomerate f with H iff :*

1. $\forall h \in H, \mathbf{Post}(p, h)$ *is a safe function*

2. $cd(f) = cd(p)$ *and* $\mathbf{Pre}(p, f)$ *is the identity function*

3. *Initially* p *is unmarked*

4. *Any firing of* f *produces tokens*

5. *The only input place of* f *is* p

The condition *1* ensures that for a given colour, a transition of H produces one token per firing. The condition *2* ensures that such a token can be consumed by the firing of f with the same colour. The condition *3* could be overcomed by substituting $\mathbf{m_0}$ by a set of initial markings after emptying the place p. Nevertheless the place p is seldom marked. The condition *4* is necessary for preserving boudedness equivalence. The last condition with condition *2* is the crucial one as it ensures the immediate firing of f once p is marked.

The transformation deletes the place p and the bordering arcs. Moreover any function on output arcs of a transition of H is obtained as the sum of the old function and the combined effect of the output of this transition followed by adequate firings of f (this explains the composition of functions).

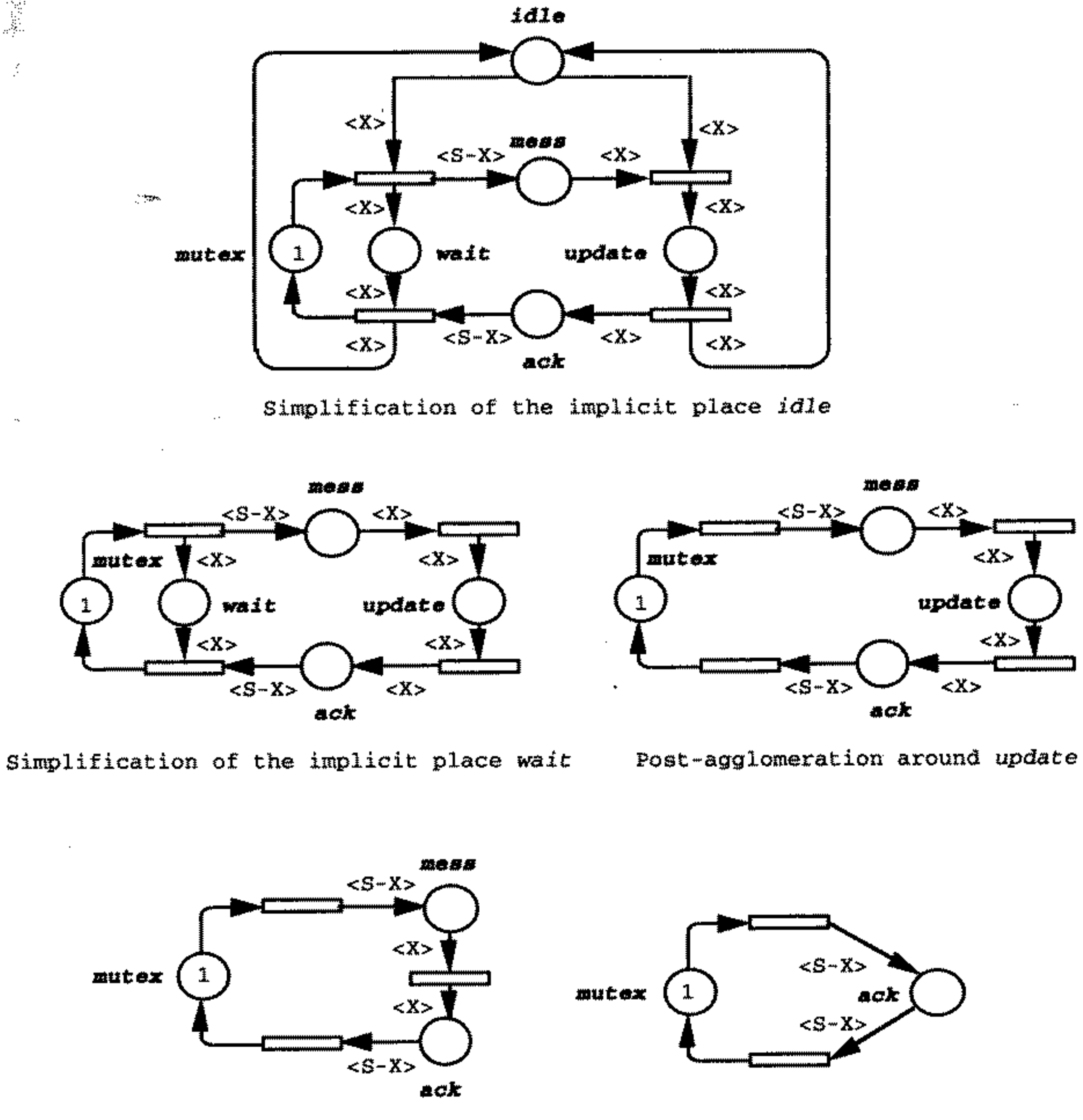
Definition 16.29 *The reduced net* $\mathcal{N}_r = \langle P', T', \mathbf{Pre}', \mathbf{Post}', C', cd' \rangle$ *with initial marking* $\mathbf{m_0}'$ *obtained from the net* $\mathcal{N}$ *by the post-agglomeration of the set of transitions* H *and the transition* f *is defined by :*

- $P' = P - \{p\}$
- $T' = T - \{f\}$
- $C' = C$
- $\forall t \in T', \forall p' \in P', cd(t) = cd(t)$ *and* $cd(p') = cd(p)$
- $\forall t \in T' - H, \forall p' \in P', \mathbf{Pre}'(p', t) = \mathbf{Pre}(p', t)$ *and* $\mathbf{Post}'(p', t) = \mathbf{Post}(p', t)$
- $\forall h \in H, \forall p' \in P', \mathbf{Pre}'(p', h) = \mathbf{Pre}(p', h)$ *and* $\mathbf{Post}'(p', h) = \mathbf{Post}(p', h) + \mathbf{Post}(p', f) \circ \mathbf{Post}(p, h)$
- $\forall p' \in P', \mathbf{m_0}'(p') = \mathbf{m_0}(p')$

Application to the example

The figure 16.13 shows the reduction process of the model of database management. We could have reduced the initial net until we obtain a single transition using other reductions than the ones presented here. Nevertheless the final net presented in the figure is small enough to analyze its behaviour. The net language is the set of prefixes of $(\bigcup_{c \in Sites} t1(c).t2(c))^*$. Its interpretation is straightforward : the behaviour of the net is an infinite sequence of database modifications done in mutual exclusion.

Let us give some explanations on the applications of the reductions :



Simplification of the implicit place *idle*

Simplification of the implicit place *wait*

Post-agglomeration around *update*

Post-agglomeration around *mess*

Figure 16.13: The reduction of the CPN model of a replicated database

ow associated to the implicit place *idle* is the one we have obtained the combination of the generative family in the subsection 16.6.1.

- The flow associated to the implicit place *wait* is :

$$1.wait - \frac{1}{n-1}.mess - \frac{1}{n-1}.update - \frac{1}{n-1}.ack$$

  where n is the number of sites

- During the post-agglomration around *mess*, the valuation of the arc from *t1* to *ack* is obtained by composition of the functions $\langle s - x\rangle$ and $\langle x\rangle$.

Methodology for obtaining high-level reductions

In this part, we give the way one can use to define a new sound reductions for CPNs starting from a reduction of Petri nets. This methodology includes two steps : the specification and the validation of a new reduction.

Specification of a high-level reduction

As for the ordinary reduction, one must specify application conditions and transformation rules. During this specification, the conditions must be decomposed in two kinds :

- Structural conditions closest as possible as the ones for the ordinary reduction.
- Functionnal conditions that must be the weakest possible in order to obtain a good structure for the unfolded Petri net corresponding to the CPN.

The transformation rule must follow these two principles :

- It must not increase the size of the colour domain (since in this case there is a hidden extension of the net).
- It enables only significant operations for the colour functions (like for instance composition or inverse) in order to keep the new functions manageable.

Validation of a high-level reduction

Once defined the reduction, the proof that it is sound must be done in the following way :

Unfolding Show that in the ufolded net a set of ordinary reductions fulfill their conditions.

Reductions sequence Find an order of these reductions such that a reduction is still applicable once the previous ones have been applied.

Folding Show the ordinary reduced net may be folded in order to give the reduced CPN given by the transformation rule.

Bibliography

[1] M. Ajmone-Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets.* Wiley, 1995.

[2] G. Berthelot. Transformations and decompositions of nets. In W. Brauer, W. Reisig, and G. Rozenberg, editors, *Petri Nets: Central Models and Their Properties. Advances in Petri Nets 1986. Part I*, volume 254 of *Lecture Notes in Computer Science*, pages 359–376. Springer Verlag, Berlin, 1987.

[3] E. Best, L. Cherkasova, J. Desel, and J. Esparza. Characterization of home states in free choice systems. Berichte 7/90, Hildesheimer Informatik, Hildesheim, Germany, July 1990.

[4] E. Best and P. S. Thiagarajan. Some classes of live and safe Petri nets. In K. Voss et al., editors, *Concurrency and Nets*, pages 71–94. Springer, 1987.

[5] G.W. Brams. *Réseaux de Petri: théorie et pratique (2 vols.).* Masson, Paris, 1983.

[6] J. Campos, G. Chiola, and M. Silva. Properties and performance bounds for closed free choice synchronized monoclass queueing networks. *IEEE Transactions on Automatic Control*, 36(12):x–y, December 1991. [Special issue on *Multidimensional Queueing Networks*].

[7] J. M. Colom, J. Campos, and M. Silva. On liveness analysis through linear algebraic techniques. In *Procs. of the AGM of Esprit BRA 3148 (DEMON)*, 1990.

[8] J.M. Colom. *Análisis estructural de Redes de Petri, programación lineal y geometría convexa.* PhD thesis, Departamento de Ingeniería Eléctrica e Informática, Universidad de Zaragoza, Zaragoza, España, June 1989.

[9] J.M. Colom, J. Martínez, and M. Silva. Packages for validating discrete production systems modeled with Petri nets. In P. Borne and S.G. Tzafestas, editors, *Applied Modelling and Simulation of Technological Systems*, pages 529–536. Elsevier Science Publishers B.V. (North-Holland), 1987.

[10] J.M. Colom and M. Silva. Improving the linearly based characterization of P/T nets. In G. Rozenberg, editor, *Advances in Petri Nets 1990*, volume 483 of *Lecture Notes in Computer Science*, pages 113–145. Springer Verlag, Berlin, 1991.

[11] F. Commoner, A. W. Holt, S. Even, and A. Pnueli. Marked directed graphs. *Journal on Computer Systems Science*, 5:72–79, 1971.

[12] J.M. Couvreur. The general computation of flows for coloured nets. In *Proc. 11th Int. Conference on Application and Theory of Petri Nets*, Paris, France, June 1990.

[13] J. Desel. A proof of the rank theorem for Extended Free Choice nets. In K. Jensen, editor, *Application and Theory of Petri Nets 1992*, volume 616 of *Lecture Notes in Computer Science*, pages 134–153. Springer Verlag, Berlin, 1992.

[14] J. Desel and J. Esparza. Reachability in cyclic Extended Free Choice nets. *Theoretical Computer Science*, 114:93–118, 1993.

[15] J. Desel and J. Esparza. *Free Choice Petri Nets*, volume 40 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1995.

[16] A. Desrochers, H. Jungnitz, and M. Silva. An approximation method for the performance analysis of manufacturing systems based on GSPNs. In *Procs of the Third International Conference on Computer Integrated Manufacturing and Automation Technology (CIMAT'92)*, pages 46–55. IEEE Computer Society Press, 1992.

[17] J. Esparza. A solution to the covering problem for 1-bounded conflict-free Petri nets using linear programming. *Information Processing Letters*, 41:313–319, 1992.

[18] J. Esparza. Model checking using net unfoldings. *Science of Computer Programming*, 23:151–195, 1994.

[19] J. Esparza and M. Silva. Circuits, handles, bridges and nets. In G. Rozenberg, editor, *Advances in Petri Nets 1990*, volume 483 of *Lecture Notes in Computer Science*, pages 210–242. Springer, 1991.

[20] J. Esparza and M. Silva. On the analysis and synthesis of free choice systems. In G. Rozenberg, editor, *Advances in Petri Nets 1990*, volume 483 of *Lecture Notes in Computer Science*, pages 243–286. Springer Verlag, Berlin, 1991.

[21] H.J. Genrich. Equivalence transformations of Pr/Tr nets. In *Proc. 9th European Workshop on Application and Theory of Petri Nets*, Venezia, Italy, June 1988.

M. H. T. Hack. Analysis of production schemata by Petri nets. Master's thesis, M.I.T., Cambridge, MA, USA, 1972. (Corrections in *Computation Structures Note* 17, 1974).

[23] S. Haddad. Generalization of reduction theory to coloured nets. In *Proc.* 9^{th} *European Workshop on Application and Theory of Petri Nets*, Venezia, Italy, June 1988.

[24] S. Haddad and J.M. Couvreur. Towards a general and powerful computation of flows for parameterized coloured nets. In *Proc.* 9^{th} *European Workshop on Application and Theory of Petri Nets*, Venezia, Italy, June 1988.

[25] S. Haddad and C. Girault. Algebraic structure of flows of a regular coloured net. In *Proc.* 7^{th} *European Workshop on Application and Theory of Petri Nets*, Oxford, England, June 1986.

[26] C. Johnen. Algorithmic verification of home spaces in P/T systems. In *Procs IMACS 1988. 12th World Congress on Scientific Computation*, pages 491–493, 1988.

[27] R. M. Karp and R. E. Miller. Properties of a model for parallel computations: Determinacy, termination, queueing. *SIAM Journal on Applied Mathematics*, 14(6):1390–1411, 1966.

[28] L. H. Landweber and E. L. Robertson. Properties of conflict-free and persistent Petri nets. *Journal of the ACM*, 25(3):352–364, 1978.

[29] K. Lautenbach. Linear algebraic techniques for Place/transition nets. In W. Brauer et al., editor, *Petri Nets: Central Models and their Properties. Advances in Petri Nets 1986*, volume 254 of *Lecture Notes in Computer Science*, pages 142–167. Springer Verlag, Berlin, 1987.

[30] G. Memmi. *Fuites et Semi-flots dans les Réseaux de Pétri.* PhD thesis, Univ. Pierre et Marie Curie (Paris VI), December 1978.

[31] G. Memmi and G. Roucairol. Linear algebra in net theory. In W. Brauer, editor, *Net Theory and Applications*, volume 84 of *Lecture Notes in Computer Science*, pages 213–223. Springer Verlag, Berlin, 1980.

[32] G. Memmi and J. Vautherin. Analysisng nets by the invariant method. In W. Brauer, W. Reisig, and G. Rozenberg, editors, *Petri Nets: Central Models and their Properties. Advances in Petri Nets 1986, Part I*, volume 254 of *Lecture Notes in Computer Science*, pages 300–336. Springer, 1987. Collected in [?].

[33] K.G. Murty. *Linear Programming.* John Wiley and Sons, New York, 1983.

[34] G.L. Nemhauser, A.H. Rinnoy Kan, and M.J. Todd. *Optimization*, volume 1 of *Handbook in Operations Research and Management Science.* North Holland, Amsterdam, 1989.

Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization.* John Wiley and Sons, 1988.

[36] L. Recalde, E. Teruel, and M. Silva. Modeling and analysis of cooperating processes with Petri nets. Research report, Dep. Informática e Ingeniería de Sistemas, Universidad de Zaragoza, María de Luna, 3, 50015 Zaragoza, Spain, 1996.

[37] M. Silva. Sur le concept de macroplace et son utilisation pour l'analyse des réseaux de Petri. *R.A.I.R.O. Automatique/Systems Analysis and Control*, 15(4):335–345, Avril 1981.

[38] M. Silva. *Las redes de Petri en la Automática y la Informática.* Editorial AC, Madrid, 1985.

[39] M. Silva. Logical controllers. In *Proceedings of IFAC Symposium on Low Cost Automation*, pages 157–166, 1989.

[40] M. Silva and J.M. Colom. On the computation of structural synchronic invariants in P/T nets. In G. Rozenberg, editor, *Advances in Petri Nets 1988*, volume 340 of *Lecture Notes in Computer Science*, pages 386–417. Springer Verlag, Berlin, 1988.

[41] E. Teruel, J. M. Colom, and M. Silva. Choice-free Petri nets: A model for deterministic concurrent systems with bulk services and arrivals. *IEEE Trans. on Systems, Man, and Cybernetics*, 1997. To appear.

[42] E. Teruel, J.M. Colom, and M. Silva. Linear analysis of deadlock-freeness of Petri net models. In *Procs. of the European Control Conference, ECC'93*, pages 513–518, Groningen, The Netherlands, June 1993.

[43] E. Teruel and M. Silva. Liveness and home states in Equal Conflict systems. In M. Ajmone Marsan, editor, *Application and Theory of Petri Nets 1993*, volume 691 of *Lecture Notes in Computer Science*, pages 415–432. Springer Verlag, Berlin, 1993.

[44] E. Teruel and M. Silva. Well-formedness Equal Conflict systems. In R. Valette, editor, *Application and Theory of Petri Nets 1994*, volume 815 of *Lecture Notes in Computer Science*, pages 491–510. Springer Verlag, Berlin, 1994.

[45] E. Teruel and M. Silva. Structure theory of equal conflict systems. *Theoretical Computer Science*, 153(1-2):271–300, 1996.

[46] P. S. Thiagarajan and K. Voss. A fresh look at free choice nets. *Information and Control*, 61(2):85–113, 1984.

[47] A. Valmari. Stubborn sets for reduced state space generation. In G. Rozenberg, editor, *Advances in Petri Nets 1990*, volume 483 of *Lecture Notes in Computer Science*, pages 491–515. Springer Verlag, Berlin, 1991.

A. Valmari. A stubborn attack on state explosion. *Formal Methods in System Design*, 1(4):297–322, 1992.