Dynamic Changes in Concurrent Systems: Modelling and Verification

Eric Badouel, Javier Oliver

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Dynamic Changes in Concurrent Systems: Modelling and Verification

Eric Badouel and Javier Oliver

N° 3708

June 1999

THÈME 1

ISSN 0249-6399

Dynamic Changes in Concurrent Systems: Modelling and Verification

Eric Badouel * and Javier Oliver †

Thème 1 — Réseaux et systèmes
Projet Paragraphe

Rapport de recherche n3708 — June 1999 — 37 pages

Abstract: In this paper we address the issues of modelling and verification of concurrent systems subject to dynamic changes using Petri net formalisms. As far as the expressivity of the model is concerned a built-in and decentralized mechanism for handling changes is looked for. At the same time the basic decidable properties of Petri nets (Place Boundedness, Reachability, Deadlock, and Liveness) should remain decidable for the extended model. The gain in terms of modelling power is usually paid by a loss of decidable properties. A trade-off needs to be found between expressivity and computability. In a previous study we have introduced a class of high level Petri nets, called reconfigurable nets, that can dynamically modify their own structure by rewriting some of their components. These nets were used for modelling computer supported cooperative work (CSCW) and more precisely workflow systems. In this study we restrict our attention to the subclass of reconfigurable nets, termed reversible, whose structure modifying rules are formally inversible. Such a net may be viewed as the cascaded composition of an automaton with a parametric Petri net; and under some additional assumption it is equivalent to a stratified Petri net. Place boundedness, reachability, deadlock, and liveness are decidable properties of reversible reconfigurable nets. From a practical point of view however the choice of a particular model (Petri nets, selfmodifying or stratified Petri nets, reconfigurable nets ...) depends strongly on the nature of the problem to be modelized.

Key-words: Reconfigurable Nets, dynamic changes, CSCW, Workflow Systems, Boundedness, Self-Modifying Petri Nets

(Résumé : tsvp)

This work was undertaken during a visit of Javier Oliver at Irisa.

* Email: Eric.Badouel@irisa.fr
† Email: joliver@dsic.upv.es

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : 02 99 84 71 00 - International : +33 2 99 84 71 00
Télécopie : 02 99 84 71 71 - International : +33 2 99 84 71 71

Changements dynamiques dans les systèmes concurrents : modélisation et vérification

Résumé : Nous nous intéressons dans cette étude au problème de la modélisation et de la vérification de systèmes concurrents sujets à des changements dynamiques. Le formalisme de base est celui des réseaux de Petri. En ce qui concerne son expressivité on souhaiterait un modèle ayant un mécanisme de prise en compte des changements dynamiques qui soit à la fois interne et local. Par ailleurs les propriétés de base –telles l'accessibilité, le caractère borné d'une place, le blocage, la vivacité– devraient continuer à être des propriétés décidables pour cette classe étendue de réseaux de Petri. D'ordinaire le gain en terme d'expressivité se traduit par une perte en terme de propriété décidable et un compromis doit être trouvé entre expressivité et calculabilité. Dans une étude précédente, nous avons introduit une classe de réseaux de Petri de haut niveau, appelés réseaux reconfigurables, qui peuvent changer dynamiquement leur propre structure en réécrivant certain de leur composants. Ces réseaux peuvent servir à modéliser les changements de modes opératoires dans les systèmes de travail coopératif et plus spécifiquement dans les systèmes à flots de tâches. Dans la présente étude on se restreint à la sous classe des réseaux reconfigurables, dits réversibles, pour lesquels chaque règle de modification structurelle peut être formellement inversée. Un tel réseau peut être vu comme une composition en cascade d'un automate et d'un réseau de Petri paramétré, et sous certaines hypothèses est équivalent à un réseau de Petri stratifié. Par ailleurs toutes les propriétés mentionnées ci-dessus sont décidables pour les réseaux reconfigurables réversibles. D'un point de vue pratique cependant c'est la nature particulière de chaque problème qui détermine le choix d'un modèle particulier (réseaux de Petri, réseaux automodifiants ou stratifies, réseaux reconfigurables ...).

Mots-clé : Réseaux reconfigurables, changements dynamiques, CSCW, systèmes à flot de tâches, caractère borné, réseaux de Petri automodifiants

1 Introduction

In recent years many studies have considered using Petri nets for modelling *Computer Supported Cooperative Work* (CSCW) [12, 8]. These applications, also called *groupware* applications, concern distributed activities. They involve systems of agents (computer systems or humans) which cooperate within an heterogeneous and geographically distributed environment. The simplest instances of cooperative work are the *Workflow Management Systems*. These systems support the realization of work procedures by groups of collaborating agents by coordinating the flow of tasks within the distributed system.

The design of CSCW systems raises some important modelling issues like dynamic changes of activities, task migration, superimposition of different levels of activities and the notion of multiple operating modes. The Petri net formalism does not offer a direct way to address these features and for that reason modelling any realistic CSCW system directly with Petri net may be a difficult task! It is therefore tempting to introduce extensions of Petri nets especially designed so as to allow for an easy formalization of such or such feature. The gain in term of modelling power is usually paid by a loss of decidable properties. A trade-off needs to be found between expressivity and computability.

In this paper we address the issues of *modelling* and *verification* of concurrent systems subject to dynamic changes.

As already mentionned, making concurrent systems adaptable to changes is one of the main challenge in CSCW systems design. It is important that the mechanism for change be explicitly represented into the model so that at each stage of product development, designers can experiment the effect of structural changes, e.g. by using prototypes. This means that structural changes are taken into account from the very begining of the design process rather than handled by an external and global system, e.g. some exception handling mechanism, designed and added to the model describing the system normal behaviour. Thus we favour an *internal* and *incremental* over an *external* and *uniform* description of changes, and a *local* over a *global* handling of changes. This approach is compatible with the bottom-up modular synthesis of Petri nets where a complex system is derived from successive refinements of places or transitions by sub-systems.

Valk' self-modifying nets [15, 16] is an early attempt of an extension of Petri net model with a built-in mechanism for handling changes. It is moreover quite a natural generalization of Petri nets. In contrast with Petri net, the pre-condition and post-condition of a transition t are not given as vectors but as matrices ${}^\bullet t, t^\bullet : P \times P \to I\!N$. Transition t is then enabled in marking $M \in I\!N^P$ if and only if for each place p_i the condition $M(p_i) \geq \sum_j {}^\bullet t(p_i, p_j) M(p_j)$ is satisfied. When enabled this transition can fire and produce the new marking M' such that $M'(p_i) = M(p_i) - \sum_j {}^\bullet t(p_i, p_j) M(p_j) + \sum_j t^\bullet(p_i, p_j) M(p_j)$, that is to say $M' = \Lambda(t) \cdot M$ where the *transfert* matrix Λ is given by $\Lambda(p_i, p_j) = \delta_{i,j} - {}^\bullet t(p_i, p_j) + t^\bullet(p_i, p_j)$ with $\delta_{i,j} = 1$ if $i = j$ and $\delta_{i,j} = 0$ if $i \neq j$. The notion of systems of replacement/multiplication of matrices takes the place of the notion of systems of replacement/addition of vectors that characterize Petri nets.

Usually flow relations depends only upon a limited number of places that we shall term *control places*. The content of these places in turn are affected only by certain transitions

termed *changes of configuration* by contrast to the other *ordinary* transitions. With every marking M is associated a Petri net $N_M = (P, T, {}^\bullet()_M, ()^\bullet{}_M)$ obtained by evaluation of the flow relations in marking M: ${}^\bullet t_M(p_i) = {}^\bullet t(p_i, -) \cdot M$ and ${}^\bullet t_M(p_i) = {}^\bullet t(p_i, -) \cdot M$. This Petri net N_M is the *configuration* of the self-modifying net N in marking M. As long as no control places are modified the self-modifying net behaves as its current configuration. Self-modifying nets are therefore well adapted for situations in which several modes of operation coexist.

As indicated in Fig. 1 Petri net with inhibitors arcs can be implemented by self-modifying nets. This formalism is thus Turing powerful and there is no hope to obtain general automatic verification tools for the full class of self-modifying nets. Moreover it seems difficult to find interesting sub-classes of self-modifying nets with decidable properties (boundedness, liveness, reachability, ...) except for *Post self-modifying nets* [15] where all preconditions are constant. Indeed, Dufourd et al. [10] prove that boundedness is already undecidable for Petri nets with *reset arcs* [1], which is a simple case of self-modification (see Fig. 1). Another simple case of self-modification is related to transfert arcs [7]. In [10] Dufourd et al. prove that place-boundedness is undecidable for Petri nets with transfert arcs even though boundedness is decidable for that class. However, in order to lessen our previous assertion, one should mention that Coverability and Termination which are both undecidable properties of self-modifying nets are decidable both for Petri nets with reset arcs and for Petri nets with transfert arcs.

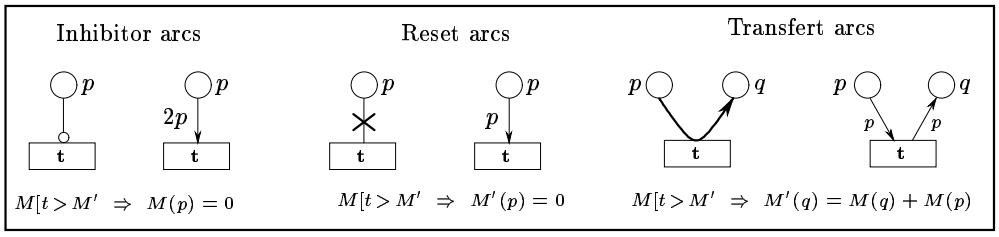


Figure 1: implementation of inhibitor, reset, and transfert arcs in self-modifying nets

There is however a gap between what is theoretically expressible in terms of self-modifying Petri nets and the manner in which we intend to use this model for modelling concurrent systems adaptable to changing situations. Even though the use of reset arcs and transfert arcs arises quite naturally in that context it is possible as we shall see shortly to obtain much more tractable models with very similar features. As stated above, we intend to use self-modifying nets for modelling a system consisting of a bunch of Petri nets, called its *configuration*, together with some mechanism (using the so-called control places and the changes of configuration) allowing the system to evolve from one configuration to another one under certain circumstances. In order to fit in more straightforwardly with this manner of describing such systems we introduce the model of *Automaton controlled Petri Nets* (ACPN in short). An ACPN is a parametric family of Petri nets, which means a Petri net whose

flow arc inscriptions are linear functions of certain parameters, together with some control given by a finite automaton whose states determine the current values of these parameters. It is therefore the cascaded composition (or semi direct product) of a (control) automaton with a parametric Petri net. In that respect it is quite similar to a Vector Addition System with States (VASS) which is a direct product of an automaton with a Petri net. We shall indeed be able to associate each ACPN with some weakly equivalent VASS. This notion of weak equivalence is enough to ensure that both models are equivalent with respect to some fundamental properties as Place Boundedness, Reachability, and Liveness. Since all these properties are decidable for VASS they are also decidable for ACPN.

Even though ACPN have a definite advantage over the self-modifying nets in terms of *validation* of properties they have serious drawbacks from a *modelling* standpoint. Actually the handling of changes of configurations is described globally and externally by the control automaton where on the contrary it was given locally and internally in self-modifying nets where control places and changes of configurations (which are respectively places and transitions of the system) are used. This means that modelling with ACPN cannot allow for the modular bottom-up synthesis technique. This model will therefore be inadequate for modelling large systems, but may be used as a target formalism on which verification may be performed. This situation is quite similar to Petri nets where a Petri net is used for the modular synthesis of a concurrent system while verification is performed on its state graph.

In order to obtain a local description of the changes of configurations one may consider modelling the control automaton itself as a (say one-safe) Petri net thus obtaining a two-level stratified Petri net. Stratified Petri nets [4] are self-modifying nets whose places $p_1, \ldots, p_n$ may be given in an order such that the flow relation inscriptions of the arcs connected to place p_i involve only the places p_j $(j < i)$ which appear strictly before p_i. This means that the transfert matrix of a stratified Petri net is a lower triangular matrix with 1 coefficients on the diagonal. A stratified Petri net can be viewed as the cascaded composition of Petri nets; and actually the marking graph of a stratified Petri net is the cascaded composition of the marking graphs of Petri nets. It was shown in [4] that stratified Petri nets may be synthesized from their marking graph using the region technique. As far as verification is concerned however stratified Petri nets suffer from the same limitations as the whole class of self-modifying Petri nets.

In a previous study [5] we have introduced a class of high level Petri nets, called reconfigurable nets, that can dynamically modify their own structure by rewriting some of their components. A reconfigurable Petri net is a Petri net with local structural modifying rules performing the replacement of one of its subnets by another subnet. The tokens in a deleted place are transfered to a created one. These nets were used for modelling dynamic changes within workflow systems as in [11]. It was shown that boundedness of a reconfigurable net can be decided by constructing a simplified form of Karp and Miller's coverability tree, however this construction did not allow to decide whether a given place of the net is bounded. The technique used by van der Aalst [2] for verifying soundness of workflow nets can be generalized to reconfigurable nets. Finally a translation of a reconfigurable net into an equivalent self-modifying net was also presented. The class of reconfigurable nets therefore

appears as a subclass of self-modifying nets for which boundedness can be decided. However this translation uses features like reset arcs whereas boundedness of the class of Petri nets with reset arcs is undecidable. This suggests that a simpler translation might be possible at least for particular subclasses of reconfigurable nets.

In this study we restrict our attention to the subclass of reconfigurable nets, termed reversible, whose structure modifying rules are formally inversible. This assumption is quite natural since the system that we intend to modelize is supposed to run forever and thus should not have degraded behaviour. Such a net may be viewed as the cascaded composition of an automaton with a parametric Petri net, i.e. as an ACPN, thus Boundedness, Reachability and Liveness are decidable for reversible reconfigurable net. Finally under some additional assumption a reversible reconfigurable net is shown equivalent to a stratified Petri net.

The remainder of the paper is organized as follows. In Section 2 we introduce the model of *Automaton Controlled Petri Nets* (ACPN). We show that any ACPN may be associated with some weakly equivalent *Vector Addition System with States* (VASS) and that consequently place-boundedness, reachability, deadlock and liveness are decidable properties of ACPN. In Section 3 we recall the definition of *Reconfigurable Nets* and consider its subclass of *Reversible Reconfigurable Nets* for which a fundamental property is established. This fundamental property is used in Section 4 to prove that the marking graph of a reversible reconfigurable net is equivalent to the marking graph of some ACPN. Thus place-boundedness, reachability, deadlock and liveness are also decidable properties of reversible reconfigurable nets. The fundamental property of reversible reconfigurable nets is used in Section 5 to establish a translation of some reversible reconfigurable nets into stratified Petri nets. In Section 6 we give some examples of concurrent systems subject to dynamic changes. They are respectively modelized using reversible reconfigurable nets, Petri nets, and stratified Petri nets showing that the choice of a particular model much depends on the nature of the problem to be modelized.

2 Automaton Controlled Petri Nets

In this Section the notion of Automaton Controlled Petri Net (ACPN) is introduced. A weak equivalence between ACPNs and Vector Addition Systems with States (VASS [14]) is indicated from which it follows that Place-boundedness, Reachability, Deadlock and Liveness are decidable properties of ACPNs. We indicate in particular a definition of the coverability tree of an ACPN which is the counterpart of the coverability tree of the associated VASS.

Definition 1 (Parametric Petri Net) *A parametric Petri net is a structure* $(\Pi, P, E, Pre, Post, M_0)$ *where* Π, P, *and* E *are pairwise disjoint finite sets whose elements are respectively termed parameters, places, and events.* $Pre, Post : P \times E \to \mathbb{N}^{\Pi}$ *are respectively termed the preset matrix and postset matrix.* $M_0 \in \mathbb{N}^P$ *the initial marking. If we let* ${}^\bullet\Lambda, \Lambda^\bullet : \mathbb{N}^{\Pi} \to (P \times E \to \mathbb{N})$ *be given by* ${}^\bullet\Lambda(\alpha)(p,e) = Pre(p,e) \cdot \alpha$ *and* $\Lambda^\bullet(\alpha)(p,e) = Post(p,e) \cdot \alpha$ *for* $\alpha \in \mathbb{N}^{\Pi}$, $e \in E$ *and* $p \in P$; *then a parametric Petri net amounts to a family of Petri*

nets $N_\alpha = (P, E, {}^\bullet\Lambda(\alpha), \Lambda^\bullet(\alpha))$ *indexed by* $\alpha \in \mathbb{N}^\Pi$. *The firing relation in* N_α *is given by*

$$M\ [e/\alpha\rangle\ M' \quad \Leftrightarrow \quad \forall p \in P \begin{cases} (i) & M(p) \geq Pre(p,e) \cdot \alpha \\ (ii) & M'(p) = M(p) - Pre(p,e) \cdot \alpha + Post(p,e) \cdot \alpha \end{cases}$$

Definition 2 (Automaton Controlled Petri Net) *An automaton controlled Petri net is a triple* $\mathcal{N} = (A, \Delta, N)$ *where* $A = (S, E, T, s_0)$ *is a finite deterministic automaton with set of states* S, *set of events* E, *transition relation* $T \subseteq S \times E \times S$, *and initial state* $s_0 \in S$. $N = (\Pi, P, E, Pre, Post, M_0)$ *is a parametric Petri net with the same set of events as the automaton.* $\Delta : S \to \mathbb{N}^\Pi$, *termed the state encoding function, gives the interface between the control part (the automaton) and the controlled part (the parametric Petri net). Actually* N *can be construed as the family of nets* N_α *controlled by the automaton* A. *The markings of* $\mathcal{N}$ *are the pairs* $(M, s) \in \mathbb{N}^P \times S$ *and its transition relation between markings is given by*

$$(M, s) \xrightarrow{e} (M', s') \quad \Leftrightarrow \quad s \xrightarrow{e} s' \ \wedge \ M\ [e/\Delta(s)\rangle\ M'$$

Its marking graph is the automaton, denoted $N \rtimes_\Delta A$, *obtained by restricting its transition relation to the set of markings reachable from the initial marking* (M_0, s_0).

An automaton controlled Petri net then consists of two systems composed in cascade: an automaton A and a parametric Petri net N. The interface between the two systems is given by the state encoding function that may be presented in matrix form $\Delta : S \times \Pi \to \mathbb{N}$. We abbreviate $N \rtimes_\Delta A$ to $N \rtimes A$ when Δ is a *trivial* state encoding, i.e. when $S = \Pi$ and Δ is the identity matrix. Automaton Controlled Petri nets are very similar to Vector Addition System with States.

Definition 3 (Vector Addition System with States (VASS)) *A Vector Addition System with States* $(V, Arcs, \ell)$ *consists of a finite oriented graph* $G = (V, Arcs)$, *an integer* $m \geq 1$ *and a map* $\ell : Arcs \to \mathbb{Z}^m$, *i.e. a VASS is an oriented graph labelled in* $\mathbb{Z}^m$. *A marking is a pair* (M, v) *made of a vector with non-negative entries* $M \in \mathbb{N}^m$ *and a vertex* $v \in V$. *The notation* $(M, v) \xrightarrow{a} (M', v')$ *means that* a *is an arc from* v *to* v' *and* $M' = M + \ell(a)$. *The marking* (M', v') *is said to be reachable from marking* (M, v) *if* $(M, v) \xrightarrow{*} (M', v')$ *where* $\xrightarrow{*}$ *is the reflexive and transitive closure of the relation of one-step reachability given by :* $(M, v) \to (M', v') \Leftrightarrow \exists a \in A \quad (M, v) \xrightarrow{a} (M', v')$. *If the VASS comes equipped with some initial marking* (M_0, v_0) *its marking graph is the graph whose labelled arcs are the triples* $(M, v) \xrightarrow{a} (M', v')$ *where* (M, v) *(and thus* (M', v')*) are markings reachable from the initial marking.*

Clearly an ACPN $\mathcal{N} = (A, \Delta, N)$ where N is a *pure* Petri net is equivalent to the VASS $\mathcal{N}^\circ = (V, Arcs, \ell)$ whose vertices are the states of A: $V = S$ and one has one arc in $Arcs$ from s to s' labelled by the vector $V(e, s) \in \mathbb{Z}^n$ given by

$$V(e, s)(i) = [Post(p_i, e) - Pre(p_i, e)] \cdot \Delta(s)$$

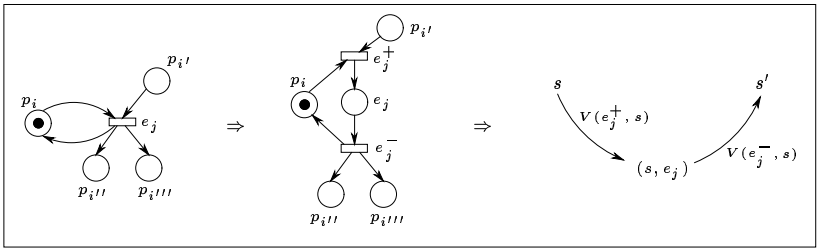


Figure 2: from impure net to pure net

where $P = \{p_1, \ldots, p_n\}$ is some fixed enumeration of the set of places of N. Actually $\mathcal{N}$ and $\mathcal{N}^\circ$ have the same markings and $(M, s) \xrightarrow{e} (M', s')$ in the marking graph of $\mathcal{N}$ if and only if $(M, s) \overset{V(e,s)}{\rightarrow} (M', s')$ in the marking graph of $\mathcal{N}^\circ$.

Suppose that net N is impure which means that both $Pre(p, e) \cdot \Delta(s)$ and $Post(p, e) \cdot \Delta(s)$ are non zero for some place p, event e and state s, otherwise stated the configuration of N associated with state s ($N_{\Delta(s)}$) is an impure Petri net. Now one can easily transform any Petri net into a pure Petri net by splitting its events. This transformation which is illustrated in the left of Fig. 2 consists in splitting each event e into two events, the "begining of e" denoted e^+, and the "end of e" denoted e^-. Each input place of event e in the original net is an input place of e^+ with the same weight. Similarly each output place of event e in the original net is an output place of e^- with the same weight. Finally we have in the split net an extra place for each event of the original net (bearing for simplicity the name of that event) this place is an output place for e^+ and an input place for e^- with single weights, its content gives the number of instances of event e that are currently executed. The split net gives a finer description of the phenomenon being modelized in that by breaking the atomicity of events it allows for an explicit description of the co-occurrence of events. However if one is not interested in the independence of events, which is the case when one want to check properties like reachability, boundedness, liveness or deadlock, then the split net is equivalent to the original net (see [3]). One may think to apply this operation for each configuration of N and to use the construction given above for pure ACPN. However the atomicity of events require that each firing of an event takes place in some particular configuration of the system, and this property is not preserved by splitting of events, since e^- can occur in a different configuration than the corresponding e^+. In order to escape this problem we shall enforce event e^+ to occur immediately after e^-. In this way we totally loose any information about the concurrency of events, but as we already mentionned this has no effect on the properties we are interested in. Thus each ACPN $\mathcal{N} = (A, \Delta, N)$ is associated with a VASS $\mathcal{N}^\circ = (V, Arcs, \ell)$ with components as follows. Vertices are either states of the ACPN or pairs (s, e) made of a state and of an event enabled in this state: $V = S \cup \{(s, e) \in S \times E \mid \exists s' \in S \quad s \xrightarrow{e} s'\}$. For each transition $s \xrightarrow{e} s'$ we have two arcs in the VASS: one from vertex s to vertex (s, e) labelled $V(e^+, s)$ and another one from vertex (s, e) to vertex s' labelled $V(e^-, s)$ where $V(e^+, s)$ and $V(e^-, s)$ are the vectors of $\mathbb{Z}^{n+k}$

given by

$$V(e^+,s)(i) = \begin{cases} -Pre(p_i,e)\cdot\Delta(s) & \text{if} \quad 1 \le i \le n \\ 1 & \text{if} \quad i > n \quad \text{and} \quad e = e_{i-n} \\ 0 & \text{if} \quad i > n \quad \text{and} \quad e \neq e_{i-n} \end{cases}$$

and

$$V(e^-,s)(i) = \begin{cases} Post(p_i,e)\cdot\Delta(s) & \text{if} \quad 1 \le i \le n \\ -1 & \text{if} \quad i > n \quad \text{and} \quad e = e_{i-n} \\ 0 & \text{if} \quad i > n \quad \text{and} \quad e \neq e_{i-n} \end{cases}$$

where $P = \{p_1,\ldots,p_n\}$ and $E = \{e_1,\ldots,e_k\}$ are some fixed enumerations of the set of places and events of the ACPN. Then the markings of $\mathcal{N}$ are also markings of $\mathcal{N}^\circ$ and $(M,s) \xrightarrow{e} (M',s')$ in the marking graph of $\mathcal{N}$ if and only if we have the sequence of transitions $(M,s) \overset{V(e^+,s)}{\rightarrow} (M''\cup\{e\},s) \overset{V(e^-,s)}{\rightarrow} (M',s')$ in the marking graph of $\mathcal{N}^\circ$ where $M'' : P \to \mathbb{N}$. Moreover $(M,s) \overset{V(e^+,s)}{\rightarrow} (M''\cup\{e\},s)$ is the only transition leading to marking $(M''\cup\{e\},s)$ and $(M''\cup\{e\},s) \overset{V(e^-,s)}{\rightarrow} (M',s')$ is the only transition from this marking because $V(e^+,s)$ which add one token to place e and $V(e^-,s)$ which remove one token from this place are the only vectors that modify the value of place e. If we restrict attention to the markings reachable from some initial marking $(M_0,s_0) \in \mathbb{N}^P \times S$ it follows that the reachable markings of $\mathcal{N}^\circ$ are the reachable markings of $\mathcal{N}$ together with some marking of the form $(M''\cup\{e\},s)$ where $M'' : P \to \mathbb{N}$, $e \in E$, and $s \in S$. Thus the ACPN and its associated VASS are equivalent with respect to the following properties: Place-Boundedness, Reachability, Deadlock and Liveness, since these properties are decidable for VASS (see [14]).

Proposition 4 *Place-Boundedness, Reachability, Deadlock and Liveness are decidable properties for ACPN.*

In particular one can decide place boundedness of an ACPN by constructing the coverability tree of its associated VASS. We can simplify the construction of this coverability tree by amalgamating the successive arcs associated with $V(e^+,s)$ and $V(e^-,s)$. We conclude this section by a description of this coverability tree.

We define an order relation between markings of an ACPN by letting

$$(M,s) \sqsubseteq (M',s') \;\Leftrightarrow\; (s = s') \;\wedge\; (\forall p \in P \; M(p) \le M'(p))$$

Similar to what is done for Petri nets, a finite approximation of the reachability tree called the coverability tree can be constructed. For that purpose a new value denoted ω and representing "an arbitrary large" integer is added to the set of $\mathbb{N}$ of non-negative integers; with the order relation and the addition and substraction operation extended by letting $\forall n \in \mathbb{N}, \quad n < \omega \quad \omega + n = n + \omega = \omega + \omega = \omega, \quad \omega - n = \omega, \quad$ and $\quad n - \omega$ is undefined.

Definition 5 (Generalized markings) *A generalized marking of an ACPN $\mathcal{N}$ is a pair $(\tilde{M},s)$ where $\tilde{M}$ is a map $\tilde{M} : P \to \mathbb{N}\cup\{\omega\}$ and $s \in S$ is a state.*

The previous order relation is then extended to generalized markings :

$$(\tilde{M}, s) \sqsubseteq (\tilde{M}', s') \;\Leftrightarrow\; (s = s') \;\wedge\; (\forall p \in P\ \tilde{M}(p) \leq \tilde{M}'(p))$$

Definition 6 (Coverability tree of an ACPN) *The coverability tree of an ACPN $\mathcal{N}$ is constructed by the following algorithm:*

- *Initially the tree is reduced to its root labelled (M_0, s_0) and tagged as "new" vertex.*
- *While "new" vertices exist, do the following:*
  - *Select a new vertex V, let (M, s) be its label.*
  - *If the label is distinct from all the labels of the nodes on the path from the root to vertex V then for every firing $(M, s) \xrightarrow{e} (M', s')$ do the following:*
    - *Create a new vertex V' and an arc from V to V' labelled e and tag this vertex V' "new".*
    - *Label vertex V' with generalized configuration $(\tilde{M}', s')$ defined as follows. If there exists some node V'' on the path from the root to vertex V whose label (M'', s'') is such that $(M'', s'') \sqsubseteq (M', s')$ and $M''(p) < M'(p)$, then we let $\tilde{M}'(p) = \omega$. Otherwise, we let $\tilde{M}'(p) = M'(p)$.*
  - *Withdraw V from the set of "new" vertices.*

Proposition 7 *The coverability tree of an Automaton Controlled Petri Net is finite. An Automaton Controlled Petri Net is bounded if and only if no vertex of its coverability tree is labeled by (M, s) in which M contains an ω component. More precisely if (M, s) is some generalized marking labelling a node of the coverability tree, then for all integer N there exists a reachable marking associated with the same state: (M', s) (i.e. $M' : P \to \mathbb{N}$ has no ω component) such that:*

1. $M(p_i) = \omega \quad \Rightarrow \quad M'(p_i) \geq N$;
2. $M(p_i) \in \mathbb{N} \quad \Rightarrow \quad M'(p_i) = M(p_i)$.

Proof: Follows from the analogous result for VASS [14]. ∎

3 Reversible Reconfigurable Nets

Let us recall from [5] the definition of a reconfigurable net.

Definition 8 *A reconfigurable net is a structure $N = (P, T, F, R)$ where $P = \{p_1, \ldots, p_m\}$ is a non empty and finite set of places, $T = \{t_1, \ldots, t_n\}$ is a non empty and finite set of transitions disjoint from P ($P \cap T = \emptyset$), $F : (P \times T) \cup (T \times P) \to \mathbb{N}$ is a weighted flow relation, and $R = \{r_1, \ldots, r_k\}$ is a finite set of structure modifying rules. A structure*

modifying rule is a map $r : P_1 \to P_2$ whose domain and codomain are disjoint subsets of places ($P_1, P_2 \subseteq P$ and $P_1 \cap P_2 = \emptyset$). A marking of net N is a map $M : P \to \mathbb{N} \cup \{\alpha\}$ where $\alpha \notin \mathbb{N}$, when $M(p) = \alpha$ place p is said not to exists in marking M whereas $M(p) = n \in \mathbb{N}$ expresses that p exists in marking M and has value n. We let $E = T \cup R$ denote the set of events of the reconfigurable net. We let $M\ [e\rangle\ M'$ denote the fact that event e is enabled in marking M and that the net reaches marking M' when firing this event. This transition relation is defined as follows. A transition $t \in T$ is enabled in marking M if:

$$\forall p \in P \qquad M(p) \neq \alpha \Rightarrow M(p) \geq F(p,t)$$

When transition t is fired in marking M, the resulting transition $M\ [t\rangle\ M'$ is such that $\forall p \in P$

$$\begin{array}{lcl} M(p) = \alpha & \Rightarrow & M'(p) = \alpha \\ M(p) \neq \alpha & \Rightarrow & M'(p) = M(p) - F(p,t) + F(t,p) \end{array}$$

A structure modifying rule $r \in R$ is enabled in marking M if:

$$\begin{array}{ll} \forall p \in P_1 & M(p) \neq \alpha \\ \forall p \in P_2 & M(p) = \alpha \end{array}$$

The firing of this enabled rule r produces the new marking M' defined as:

$$\begin{array}{ll} \forall p \in P_1 & M'(p) = \alpha \\ \forall p \in P_2 & M'(p) = \sum\{M(q) \mid q \in P_1 \ \wedge\ r(q) = p\} \\ \forall p \in P \backslash (P_1 \cup P_2) & M'(p) = M(p) \end{array}$$

A marked reconfigurable net is a reconfigurable net together with an initial marking.

The firing policy of transitions is like in the Petri net obtained by discarding the non existing places. This Petri net is called a *configuration* of the reconfigurable net. As long as no structure modifying rule take place, the reconfigurable net behaves exactly like this Petri net. Structure modifying rules produce a structure change in the net by removing existing places and creating new ones, thus moving the system from one configuration to another one. When a place is removed, the tokens of this place do not disappear, but they are moved to other places of the net. Hence, the number of tokens remains constant through the application of structure modifying rules. The set of places that exists in a marking M, let $D(M) = \{p \in P |\ M(p) \neq \alpha\}$, is termed the *domain* of M. Two markings are said to be equivalent when they have the same domain: $M_1 \equiv M_2 \Leftrightarrow D(M_1) = D(M_2)$. A *mode of operation* is an equivalence class for $\equiv$, it can be identify with a subset $D \subseteq P$ of places. We let $\mathcal{O}(N, M)$ denote the set of modes of operation of (reachable markings of) the marked reconfigurable net (N, M). Roughly speaking a reversible reconfigurable net can be seen as a bunch of Petri nets: its configurations. The configurations of a reconfigurable net correspond to the various *modes* of operation of the system. The structure modifying rules allow to switch from one mode of operation to another one while it modifies the current marking accordingly by displacing the tokens from the vanishing places to the created ones.

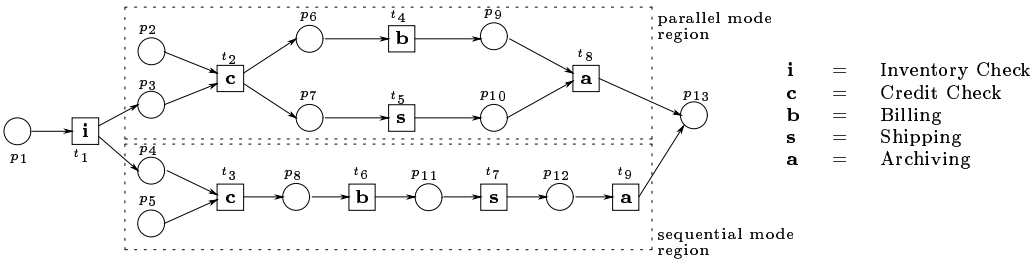


Figure 3: a reversible reconfigurable net

This reconfigurable net of Fig. 3 describes how to proceed an order request from a customer, there are two modes of operation corresponding to distinct regions in the graphical representation of the net, one in which the Billing and Shipping operations are processed sequentially and the other in which they are processed in parallel. The structure modifying rule $r : \{p_4; p_2\} \to \{p_3; p_5\}$ given by $r(p_4) = p_3$ and $r(p_2) = p_5$ permit to switch from the sequential mode of operation to the parallel mode of operation. Conversely the structure modifying rule r^{-1} realizes the switching in the converse direction. Fig. 4 represents a fragment of the marking graph of this reconfigurable net, a place is graphically represented in a given state if and only if that place exists in the current marking (i.e. its value is different from α). The reconfigurable net of Fig. 3 is *reversible* in that all of its structure modifying rules are bijections.

Definition 9 *A reversible reconfigurable net is a reconfigurable net all of whose structure modifying rules are bijections.*

The first property trivially verified by reversible reconfigurable nets, and which justify the chosen terminology, is the co-determinism of their marking graphs:

Observation 10 *Reversible reconfigurable nets have co-deterministic marking graphs:*

$$M' \ [e\rangle \ M \ \wedge \ M'' \ [e\rangle \ M \ \Rightarrow \ M' = M''$$

Every (bijective) structure modifying rule $r : P_1 \to P_2$ induces a permutation φ_r of the set of places given by:

$$\varphi_r(p) = \begin{cases} r(p) & \text{if} \quad p \in P_1 \\ r^{-1}(p) & \text{if} \quad p \in P_2 \\ p & \text{if} \quad p \in P \setminus (P_1 \cup P_2) \end{cases}$$

Every sequence of events $u \in E^*$ induces also a permutation φ_u given by:

$$\varphi_u = \begin{cases} id_P & \text{if} \quad u \in T^* \\ \varphi_r \circ \varphi_v & \text{if} \quad u = vr, \ \ v \in E^* \ \text{ and } \ r \in R \end{cases}$$

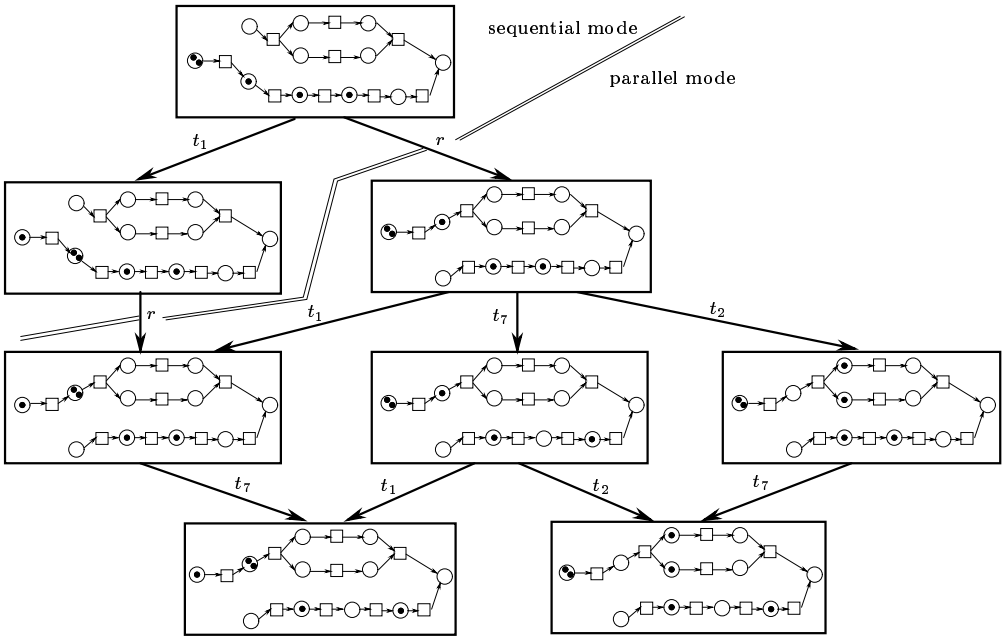


Figure 4: (part of) the marking graph of the reconfigurable net of Fig. 3

Proposition 11 (Fundamental Property of Reversible Reconfigurable Nets) *Let $N = (P, T, F, R)$ be a reversible reconfigurable net and $u \in E^*$ a sequence of events. Let $D_u^1, D_u^2 \subseteq P$, $\sigma_u : P \to \mathbb{N}$ and $\Delta_u : P \to \mathbb{Z}$ be given by:*

$$D_u^1 = \begin{cases} \emptyset & \text{if} \quad u \in T^* \\ P_1 & \text{if} \quad u = r \in R \quad r : P_1 \to P_2 \\ D_v^1 \cup \varphi_v^{-1}(D_e^1) & \text{if} \quad u = ve, \ v \in E^* \ \text{and} \ e \in E \end{cases}$$

$$D_u^2 = \begin{cases} \emptyset & \text{if} \quad u \in T^* \\ P_2 & \text{if} \quad u = r \in R \quad r : P_1 \to P_2 \\ D_v^2 \cup \varphi_v^{-1}(D_e^2) & \text{if} \quad u = ve, \ v \in E^* \ \text{and} \ e \in E \end{cases}$$

$$\sigma_u(p) = \begin{cases} 0 & \text{if} \quad u \in R^* \\ F(p,t) & \text{if} \quad u = t \in T \\ max\{\sigma_v(p); \sigma_e(\varphi_v(p)) - \Delta_v(p)\} & \text{if} \quad u = ve, \ v \in E^* \ \text{and} \ e \in E \end{cases}$$

$$\Delta_u(p) = \begin{cases} 0 & \text{if} \quad u \in R^* \\ F(t,p) - F(p,t) & \text{if} \quad u = t \in T \\ \Delta_v(p) + \Delta_e(\varphi_v(p)) & \text{if} \quad u = ve, \ v \in E^* \ \text{and} \ e \in E. \end{cases}$$

For markings M and M' one has

$$M \ [u\rangle \ M' \ \Leftrightarrow \begin{cases} (i) & D_u^1 \subseteq D(M) \quad \text{and} \quad D(M) \cap D_u^2 = \emptyset \\ (ii) & \forall p \in D(M) \quad M(p) \geq \sigma_u(p) \\ (iii) & D(M') = \varphi_u(D(M)) \\ (iv) & \forall p \in D(M) \quad M'(\varphi_u(p)) = M(p) + \Delta_u(p) \end{cases}$$

Proof: Let $M \xrightarrow{u} M'$ when conditions *(i)* to *(iv)* are satisfied. This set of transitions is deterministic. More precisely, we write $M \xrightarrow{u} M' \Leftrightarrow M \xrightarrow{u} \wedge M' = M \cdot u$ where $M \xrightarrow{u} \Leftrightarrow (i) \wedge (ii)$ and $M' = M \cdot u \Leftrightarrow (iii) \wedge (iv)$. The ternary relation $R(M, u, M') \Leftrightarrow [M' = M \cdot u]$ is, as suggested by its notation, functional: given M and u, the conditions *(iii)* and *(iv)* completely caracterize the element M'. In particular such an M' always exists, and thus $M \xrightarrow{u}$ is logically equivalent to $\exists M' \quad M \xrightarrow{u} M'$. We prove $M \ [u\rangle \ M' \ \Leftrightarrow \ M \xrightarrow{u} M'$ by induction on the length of u. The base cases are when $|u| \leq 1$:

- Let $u = \epsilon$ be the empty word. Then by convention ϵ is enabled in every marking M and $M \ [\epsilon\rangle \ M' \ \Rightarrow \ M' = M$. Since $D_\epsilon^1 = D_\epsilon^2 = \emptyset$ and $\sigma_\epsilon(p) = 0$, we deduce that $M \xrightarrow{\epsilon}$ always holds. Since $\varphi_\epsilon = id_P$ and $\Delta_\epsilon(p) = 0$, we deduce $M \cdot \epsilon = M$.

- Let $u = t \in T$ be a transition. Since $D_t^1 = D_t^2 = \emptyset$, and $\sigma_t(p) = F(p,t)$, $M \xrightarrow{t} \Leftrightarrow \forall p \in D(M) \quad M(p) \geq F(p,t)$. Since $\Delta_t(p) = F(t,p) - F(p,t)$ and $\varphi_t = id_P$, $M' = M \cdot t \Leftrightarrow D(M') = D(M) \ \wedge \ \forall p \in D(M) \quad M'(p) = M(p) + \Delta_t(p)$. Altogether $M \xrightarrow{t} M' \ \Leftrightarrow \ M \ [t\rangle \ M'$.

- Let $u = r \in R$ be a structure modifying rule which is a bijection $r : P_1 \to P_2$. Since $D^1_r = P_1$, $D^2_r = P_2$, and $\sigma_r(p) = 0$, $M \xrightarrow{r} \Leftrightarrow P_1 \subseteq D(M) \wedge D(M) \cap P_2 = \emptyset$. Since $\Delta_r(p) = 0$, $M' = M \cdot r \Leftrightarrow D(M') = \varphi_r(D(M)) \wedge \forall p \in D(M) \quad M'(\varphi_r(p)) = M(p)$, i.e. $M' = M \cdot r \Leftrightarrow M = M' \circ \varphi_r$. Altogether $M \xrightarrow{r} M' \Leftrightarrow M\ [r\rangle\ M'$.

We assume $M\ [u\rangle\ M' \Leftrightarrow M \xrightarrow{u} M'$ for $|u| < n$ with $n \geq 1$. Let $u = v.e$ be a word of length n, where $v \in E^*$ and $e \in E$. $M\ [u\rangle\ M' \Leftrightarrow \exists M''\ M\ [v\rangle\ M'' \wedge M''\ [e\rangle\ M'$. By inductive assumption $M\ [u\rangle$ is therefore equivalent to $M \xrightarrow{v} \wedge\ M \cdot v \xrightarrow{e}$, i.e. to the following conditions

$$
\begin{array}{ll}
(i) & D^1_v \subseteq D(M) \quad \text{and} \quad D(M) \cap D^2_v = \emptyset \\
(ii) & \forall p \in D(M) \quad M(p) \geq \sigma_v(p) \\
(i)' & D^1_e \subseteq D(M'') \quad \text{and} \quad D(M'') \cap D^2_e = \emptyset \\
(ii)' & \forall p \in D(M'') \quad M''(p) \geq \sigma_e(p) \\
 & \text{where marking } M'' \text{ is given by:} \\
(iii) & D(M'') = \varphi_v(D(M)) \\
(iv) & \forall p \in D(M) \quad M''(\varphi_v(p)) = M(p) + \Delta_v(p)
\end{array}
$$

Since $D(M'') = \varphi_v(D(M))$, $(i) \wedge (i)' \Leftrightarrow D^1_v \cup \varphi_v^{-1}(D^1_e) \subseteq D(M) \wedge [D^2_v \cup \varphi_v^{-1}(D^2_e)] \cap D(M) = \emptyset$, it is therefore equivalent to

$$(i)'' \quad D^1_u \subseteq D(M) \quad \text{and} \quad D(M) \cap D^2_u = \emptyset$$

$(ii) \wedge (ii)' \Leftrightarrow$ [by *(iii)* and *(iv)*] $\forall p \in D(M) \quad M(p) \geq \sigma_v(p) \wedge M(p) + \Delta_v(p) \geq \sigma_e(\varphi_v(p)) \Leftrightarrow \forall p \in D(M) \quad M(p) \geq max\{\sigma_v(p); \sigma_e(\varphi_v(p)) - \Delta_v(p)\} = \sigma_u(p)$. Thus $(ii) \wedge (ii)' \Leftrightarrow (ii)''$ where:

$$(ii)'' \quad \forall p \in D(M) \quad M(p) \geq \sigma_u(p)$$

Thus $M\ [u\rangle$ if and only if $M \xrightarrow{u}$. Now both transition systems are deterministic, and by the base cases $M\ [e\rangle\ M' \Leftrightarrow M \xrightarrow{e} M'$ for every $e \in E$, thus $M\ [u\rangle\ M' \Leftrightarrow M \xrightarrow{u} M'$ as required. ∎

We use the fundamental property of reversible reconfigurable nets in Section 4 and in Section 5. In the first one, we show that any reversible reconfigurable net is equivalent to some automaton controlled Petri net and we derive therefrom a definition of coverability trees for reversible reconfigurable nets. This definition improves the coverability tree construction introduced in [5] for reconfigurable nets in that it not only allows to decide on boundedness but also on place-boundedness. In Section 5, we identify a subclass of reversible reconfigurable nets that can be simulated by a stratified Petri net [4], that is to say by a self-modifying Petri net for which a stratification of the set of places into layers exists so that the flow relations attached to a place involve only the content of places of lower layers. The modifying Petri nets that we have used to simulate reconfigurable nets in [5] are not stratified. This feature was essential since reconfigurable nets unlike stratified Petri nets are in general not co-deterministic in the sense that we cannot for each firing $M\ [e\rangle\ M'$ deduce marking M

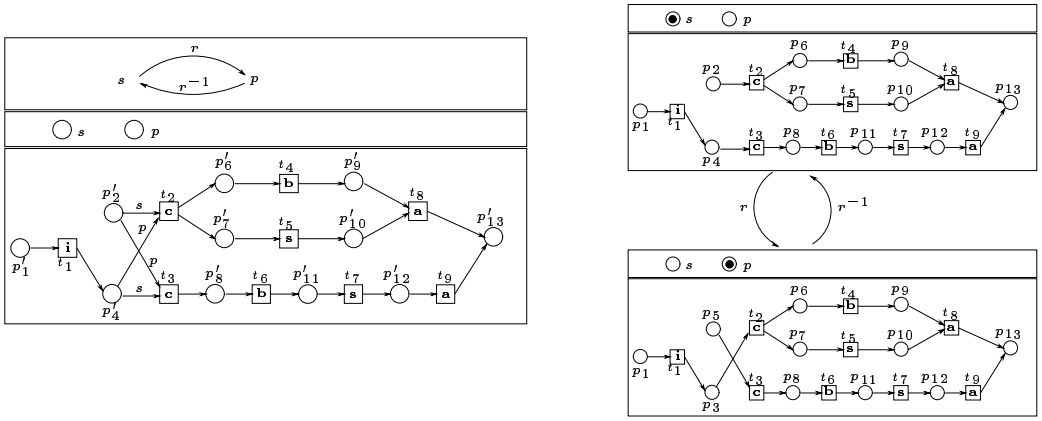


Figure 5: the reversible reconfigurable net of Fig. 3 as an automaton controlled Petri net

from the data of event e and marking M'. As we noticed already, reconfigurable nets are co-deterministic however if all structure modifying rules are assumed to be bijective, i.e. if it is a reversible reconfigurable net.

4 Reversible Reconfigurable Net and Automaton Controlled Petri Nets

It is only for convenience that we have assumed that the automaton and the parametric Petri Net have the same set of events. This is usually not the case, but can safely be assumed up to the adjunction of *idle events*. An idle event of the control part (automaton) is an $e \in E$ such that $\forall s \in S \quad s \xrightarrow{e} s$. An idle event of the controlled part (parametric Petri net) is an $e \in E$ such that $\forall p \in P \quad Pre(p,e) = Post(p,e) = 0$. Fig. 5 gives the automaton controlled Petri net corresponding to the reversible reconfigurable net of Fig. 3, notice that the idle events are not represented as it will always be the case. If A and B are deterministic automata with the same alphabet of events we let $A \preccurlyeq B$ and say that A is covered by B when there exists a saturating morphism of automata from B to A, i.e. a map σ from the set of states of B to the set of states of A which relates the corresponding initial states and such that $s_1 \xrightarrow{e} s_2 \Rightarrow \sigma(s_1) \xrightarrow{e} \sigma(s_2)$ and $\sigma(s_1) \xrightarrow{e} s'_2 \Rightarrow \exists s_2 \; s_1 \xrightarrow{e} s_2 \wedge \sigma(s_2) = s'_2$. If every state is accessible from the initial state in both automata, then such a morphism when it exists is unique and it is a surjection. Moreover the equivalence generated by the covering relation (i.e. the least equivalence relation that contains $\preccurlyeq$) identifies exactly those automata having the same language (where every state is a terminal state).

Proposition 12 *The marking graph of a marked reversible reconfigurable net is covered by (and thus equivalent to) the marking graph of an automaton controlled Petri net with trivial*

state encoding:

$$\mathbf{mg}(N, M_0) \preccurlyeq N(\Phi) \rtimes A(\Phi)$$

If $N = (P, T, F, R)$, *then* $A(\Phi) = (S, E, T', s_0)$ *and* $N(\Phi) = (\Pi, P', E, Pre, Post, M'_0)$ *where* $S = \Pi$ *is the group* Φ *of permutations generated by the permutations* φ_r *associated with some structure modifying rule* $r \in R$, *and* s_0 *is the neutral element of that group.* $E = T \cup R$ *is the set of events of* N. $P' = D(M_0) = \{p \in P | M_0(p) \neq \alpha\}$ *is the domain of marking* M_0, *and* M'_0 *is the restriction of* M_0 *to its domain:* $\forall p \in P' = D(M_0) \quad M'_0(p) = M_0(p)$. *The non idle events of the control part are the structure modifying rules* $r \in R$ *with*

$$\varphi \xrightarrow{r} \varphi' \quad \Leftrightarrow \quad r : P_1 \to P_2 \ \wedge \ P_1 \subseteq \varphi(P') \ \wedge \ P_2 \cap \varphi(P') = \emptyset \ \wedge \ \varphi' = \varphi_r \circ \varphi$$

The non idle events of the controlled part are the transitions $t \in T$ *of the reconfigurable net with* $Pre(p,t) = \sum F(\varphi(p), t) \cdot \varphi$ *and* $Post(p,t) = \sum F(t, \varphi(p)) \cdot \varphi$, *i.e.*

$$M \ [t/\varphi\rangle \ M' \quad \Leftrightarrow \quad \forall p \in P' \begin{cases} (i) & M(p) \geq F(\varphi(p), t) \\ (ii) & M'(p) = M(p) - F(\varphi(p), t) + F(t, \varphi(p)) \end{cases}$$

Proof: By Prop. 11 the transition relation of N is given by

$$M \ [u\rangle \ M' \ \Leftrightarrow \begin{cases} (i) & D^1_u \subseteq D(M) \quad \text{and} \quad D(M) \cap D^2_u = \emptyset \\ (ii) & \forall p \in D(M) \quad M(p) \geq \sigma_u(p) \\ (iii) & D(M') = \varphi_u(D(M)) \\ (iv) & \forall p \in D(M) \quad M'(\varphi_u(p)) = M(p) + \Delta_u(p) \end{cases}$$

Suppose M is a reachable marking, i.e. $M_0 \ [u\rangle \ M$ for some firing sequence $u \in E^*$. The marking $M/u = M \circ \varphi_u$ is such that $D(M/u) = D(M_0)$, and marking M can be retrieved from M/u and φ_u using the identity $M = M/u \circ \varphi_u^{-1}$. Let us therefore consider the pairs (M, φ) in which φ belongs to the group Φ generated by the permutations φ_r for r a structure modifying rule, and M is a marking with domain $P' = D(M_0)$. And we associate such a pair (M, φ) with the marking $M \circ \varphi^{-1}$. Now

$$M \circ \varphi^{-1} \ [u\rangle \ M' \circ \varphi'^{-1} \ \Leftrightarrow \begin{cases} (i) & D^1_u \subseteq \varphi(P') \quad \text{and} \quad \varphi(P') \cap D^2_u = \emptyset \\ (ii) & \forall p \in P' \quad M(p) \geq \sigma_u(\varphi(p)) \\ (iii) & \varphi'(P') = \varphi_u(\varphi(P')) \\ (iv) & \forall p \in P' \ \ M'(\varphi'^{-1} \circ \varphi_u \circ \varphi(p)) = M(p) + \Delta_u(\varphi(p)) \end{cases}$$

Notice that $M \circ \varphi^{-1} \ [u\rangle \ M' \circ \varphi'^{-1}$ does not entail that $\varphi' = \varphi_u \circ \varphi$. Let $(M, \varphi) \xrightarrow{u} (M', \varphi')$ when $M \circ \varphi^{-1} \ [u\rangle \ M' \circ \varphi'^{-1}$ and $\varphi' = \varphi_u \circ \varphi$. Thus

$$(M, \varphi) \xrightarrow{u} (M', \varphi') \Leftrightarrow \begin{cases} (i) & D^1_u \subseteq \varphi(P') \quad \text{and} \quad \varphi(P') \cap D^2_u = \emptyset \\ (ii) & \forall p \in P' \quad M(p) \geq \sigma_u(\varphi(p)) \\ (iii) & \varphi' = \varphi_u \circ \varphi \\ (iv) & \forall p \in P' \quad M'(p) = M(p) + \Delta_u(\varphi(p)) \end{cases}$$

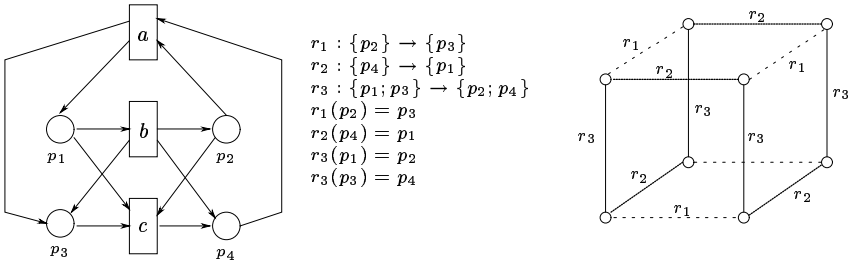


Figure 6: a reversible reconfigurable net

When $u = t \in T$ is a transition of N these conditions reduces to

$$(M,\varphi) \xrightarrow{t} (M',\varphi') \Leftrightarrow \begin{cases} (i) & \square \\ (ii) & \forall p \in P' \quad M(p) \geq F(\varphi(p),t) \\ (iii) & \varphi' = \varphi \\ (iv) & \forall p \in P' \quad M'(p) = M(p) - F(\varphi(p),t) \\ & \qquad\qquad\qquad\quad +F(t,\varphi(p)) \end{cases}$$

When $u = r \in R$ is a structure modifying rule, the above reduces to

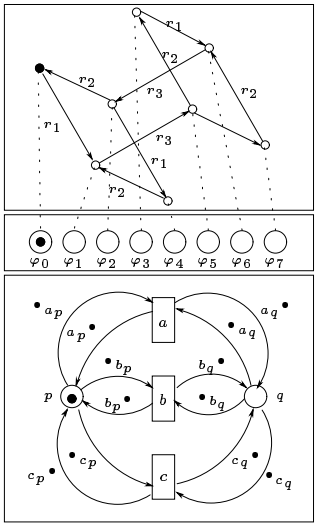
$$(M,\varphi) \xrightarrow{r} (M',\varphi') \Leftrightarrow \begin{cases} (i) & P_1 \subseteq \varphi(P') \quad \text{and} \quad P_2 \cap \varphi(P') = \emptyset \\ (ii) & \square \\ (iii) & \varphi' = \varphi_r \circ \varphi \\ (iv) & M' = M \end{cases}$$

Therefore the restriction of the set of transitions $(M,\varphi) \xrightarrow{e} (M',\varphi')$ to the set of configurations (M,φ) reachable from the initial configuration (M_0, id) is the automaton controlled Petri net $N(\Phi) \rtimes A(\Phi)$. Now the mapping $f : N(\Phi) \rtimes A(\Phi) \to \mathbf{mg}(N, M_0)$ given by $f(M,\varphi) = M \circ \varphi^{-1}$ is, by definition of $M \circ \varphi^{-1}\ [u\rangle\ M' \circ \varphi'^{-1}$, a morphism of automata. It is a saturating morphism, because $M \circ \varphi^{-1}\ [e\rangle\ M'$, which is the conjunction of the following conditions

$$\begin{array}{ll} (i) & D_e^1 \subseteq \varphi(P') \quad \text{and} \quad \varphi(P') \cap D_e^2 = \emptyset \\ (ii) & \forall p \in P' \quad M(p) \geq \sigma_e(\varphi(p)) \\ (iii) & D(M') = \varphi_e(\varphi(P')) \\ (iv) & \forall p \in P' \quad M'(\varphi_e \circ \varphi(p)) = M(p) + \Delta_e(\varphi(p)) \end{array}$$

means that one has the transition $(M,\varphi) \xrightarrow{e} (M' \circ \varphi_e \circ \varphi, \varphi_e \circ \varphi)$ whose target is sent to M': $f(M' \circ \varphi_e \circ \varphi, \varphi_e \circ \varphi) = M'$. ∎

Fig. 6 represents a reversible reconfigurable net with three structure modifying rules. Let σ_j be the permutation of $\{1;2;3\}$ associated with the permutation φ_{r_j}, i.e. $\varphi_{r_j}(p_i) = p_{\sigma_j(i)}$. Then $\sigma_1 = (2,3)$, $\sigma_2 = (1,4)$, and $\sigma_3 = (1,2)(3,4)$ and Φ is isomorphic to the group with



$\varphi_j(p_i) = p_{\sigma_j(i)}$ where

σ_0	=	/1 2 3 4/	=	id
σ_1	=	/1 3 2 4/	=	$(2,3)$
σ_2	=	/4 2 3 1/	=	$(1,4)$
σ_3	=	/2 1 4 3/	=	$(1,2)(3,4)$
σ_4	=	/4 3 2 1/	=	$(1,4)(2,3)$
σ_5	=	/2 4 1 3/	=	$(1,2,4,3)$
σ_6	=	/3 1 4 2/	=	$(1,3,4,2)$
σ_7	=	/3 4 1 2/	=	$(1,3)(2,4)$

$$
\begin{aligned}
{}^\bullet a_p &= \varphi_2+\varphi_3+\varphi_4+\varphi_5 \\
a_p^\bullet &= \varphi_0+\varphi_1+\varphi_6+\varphi_7 \\
{}^\bullet b_p &= \varphi_0+\varphi_1 \\
b_p^\bullet &= \varphi_2+\varphi_3+\varphi_4+\varphi_5+\varphi_6+\varphi_7 \\
{}^\bullet c_p &= \varphi_0+\varphi_1+\varphi_3+\varphi_5+\varphi_6+\varphi_7 \\
c_p^\bullet &= \varphi_2+\varphi_4 \\
{}^\bullet a_q &= \varphi_0+\varphi_2+\varphi_5+\varphi_7 \\
a_q^\bullet &= \varphi_1+\varphi_3+\varphi_4+\varphi_6 \\
{}^\bullet b_q &= \varphi_3+\varphi_6 \\
b_q^\bullet &= \varphi_0+\varphi_1+\varphi_2+\varphi_4+\varphi_5+\varphi_7 \\
{}^\bullet c_q &= \varphi_0+\varphi_1+\varphi_2+\varphi_3+\varphi_4+\varphi_6 \\
c_q^\bullet &= \varphi_5+\varphi_7
\end{aligned}
$$

Figure 7: the net of Fig. 6 as an automaton controlled Petri net

generators $\sigma_1, \sigma_2, \sigma_3$ and with relators $\sigma_1\cdot\sigma_1$, $\sigma_2\cdot\sigma_2$, $\sigma_3\cdot\sigma_3$, $\sigma_1\cdot\sigma_2\cdot\sigma_1\cdot\sigma_2$, and $\sigma_1\cdot\sigma_3\cdot\sigma_2\cdot\sigma_3$. Thus the generators are involutive, σ_1 and σ_2 commutes ($\sigma_1\cdot\sigma_2 = \sigma_2\cdot\sigma_1$) and σ_3 exchanges σ_1 and σ_2 ($\sigma_1\cdot\sigma_3 = \sigma_3\cdot\sigma_2$ and $\sigma_2\cdot\sigma_3 = \sigma_3\cdot\sigma_1$). The group Φ is thus isomorphic to the semi-direct product $(\mathbb{Z}_2\times\mathbb{Z}_2)\rtimes_\theta\mathbb{Z}_2$ where action $\theta:\mathbb{Z}_2\to\mathbf{Aut}(\mathbb{Z}_2\times\mathbb{Z}_2)$ is given by $\theta(1)(i,j) = (j,i)$ and where the generators σ_1, σ_2 and σ_3 are respectively mapped to $(1,0,0)$, $(0,1,0)$ and $(0,0,1)$. The Cayley graph of Φ is represented on the right-hand side of Fig. 6. The automaton $A(\Phi)$ is obtained by keeping those transitions $\varphi\xrightarrow{r}\varphi'$ of the Cayley graph of Φ (i.e. $\varphi' = \varphi_r\circ\varphi$) such that $P_1\subseteq\varphi(P')$ and $P_2\cap\varphi(P') = \emptyset$ where $r: P_1\to P_2$ and $P' = D(M_0)$. Fig. 7 represents the net of Fig. 6 as an automaton controlled Petri net. It can equivalently be represented (see Fig. 8) as the automaton $A(\Phi)$ where each node of this automaton is labelled by the corresponding configuration of the net $N(\Phi)$. A reversible reconfigurable net and its associated are equivalent with respect to the following properties: Place-boundedness, reachability, liveness, and deadlock. Thus,

Proposition 13 *Place-boundedness, reachability, liveness, and deadlock are decidable properties of reversible reconfigurable nets.*

The only point in the above statement that is not immediately clear is about the place-boundedness property. Indeed different places of the reversible reconfigurable net are represented by the same place in the associated automaton controlled Petri net. However the state component raises the ambiguity: by Prop. 7 the place p of a reversible reconfigurable net is bounded if and only if for each vertex of the coverability tree of the associated ACPN it is the case that, if (M,φ) is the label of this vertex, $M(\varphi^{-1}(p))\neq\omega$.

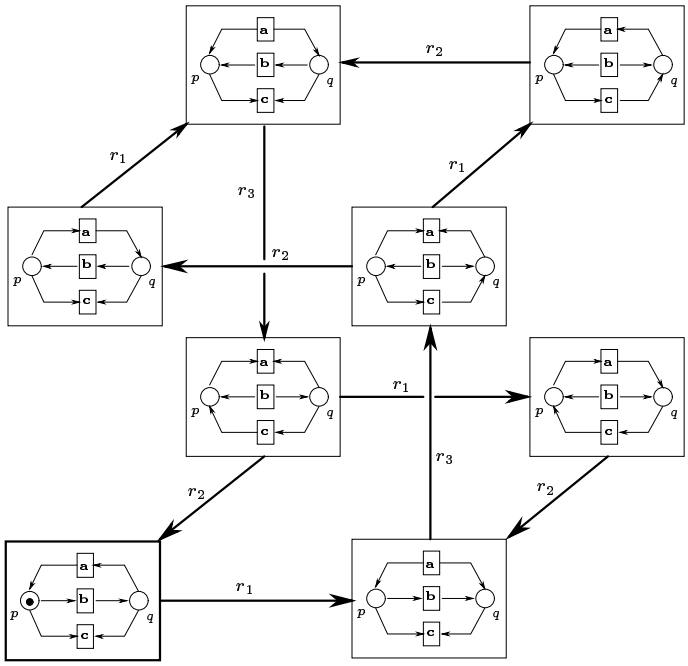


Figure 8: another representation of the cascade of Fig. 7

In order to conclude this section we rephrase the definition of the coverability tree of the ACPN associated with a reversible reconfigurable net. We recall that the order between (generalized) markings is given by

$$(M,\varphi) \sqsubseteq (M',\varphi') \quad \Leftrightarrow \quad \varphi = \varphi' \quad \text{and} \quad \forall p \in \Omega_0 \quad M(p) \leq M'$$

Thus a component ω is added in the coverability construction when one reaches a vertex V' associated with generalized marking (M',φ') such that there exists some vertex V on the path from the root to V' labelled with generalized marking (M,φ) such that $(M,\varphi) \sqsubseteq (M',\varphi')$ and $M(p) < M'(p)$ for some p. Thus $\varphi = \varphi'$ and $\varphi_u = id$ where u is the label of the path from V to V'. The algorithm is then as follows where a sequence is termed *iterating* when the induced permutation is the identity mapping.

Definition 14 *The coverability tree of a marked reversible reconfigurable net (N, M_0) is constructed by the following algorithm:*

- *Initially the tree is reduced to its root labelled M_0 and tagged as a "new" vertex.*
- *While "new" vertices exist, do the following:*
  - *Select a new vertex V, let M be its label.*
  - *If M is distinct from all the labels of the nodes on the path from the root to vertex V then for every firing $M\ [e\rangle\ M'$ do the following:*
    - *Create a new vertex V' and an arc from V to V' labelled e and tag this vertex V' "new".*
    - *Label vertex V' with generalized marking $\tilde{M}'$ defined as follows. If there exists some node V'' on the path from the root to vertex V whose label M'' is such that $M'' \sqsubseteq M'$ and $M''(p) < M'(p)$ and such that the label of the path from V'' to V' is an iterating sequence of events, then we let $\tilde{M}'(p) = \omega$. Otherwise, we let $\tilde{M}'(p) = M'(p)$.*
  - *Withdraw V from the set of "new" vertices.*

5 Reversible Reconfigurable Nets and Stratified Petri Nets

In [5] a translation of a reconfigurable net into an equivalent self-modifying net was represented. Self-modifying nets [15, 16] are generalizations of place/transition nets where the flow relation between a place and a transition depends on the marking.

Definition 15 *A self-modifying net is a structure $N = (P,T,F)$ where $P = \{p_1,\ldots,p_m\}$ is a non empty and finite set of places, $T = \{t_1,\ldots,t_n\}$ is a non empty and finite set of transitions disjoint from P, and $F : (P \times T) \cup (T \times P) \to \mathbb{N}^{P_*}$ is the flow relation where $P_* = P \cup \{*\}$ and $* \notin P$. A vector $\varphi \in \mathbb{N}^{P_*}$ can be represented by a formal sum $\varphi = \lambda_0 + \sum_{i=1}^{m} \lambda_i \cdot p_i$ where the constant coefficient is the entry corresponding to the fictituous*

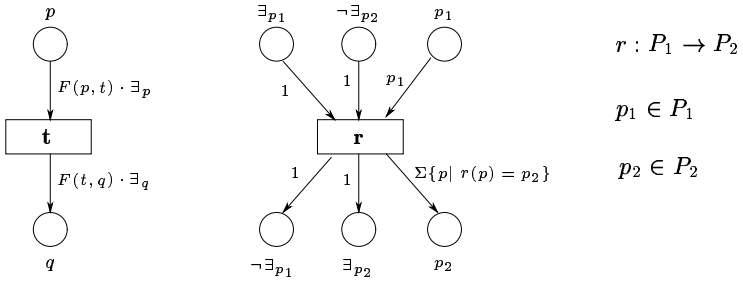


Figure 9: translating a reconfigurable net into an equivalent self-modifying net

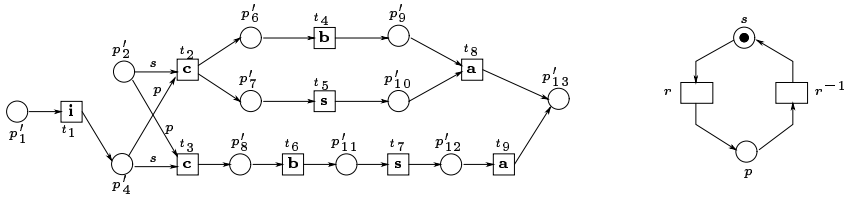


Figure 10: a stratified Petri net equivalent to the reconfigurable net of Fig. 3

place: $\lambda_0 = \varphi(*)$ *and* $\lambda_i = \varphi(p_i)$. *A marking of net* N *is a map* $M : P \to \mathbb{N}$. *If* $M \in \mathbb{N}^P$ *is a marking and* $\varphi \in \mathbb{N}^{P_*}$, *we let* $\varphi(M) = \lambda_0 + \sum_{i=1}^{m} \lambda_i \cdot M(p_i)$ *denote the evaluation of the affine function* φ *in marking* M. *We let* $M\ [t\rangle\ M'$ *when transition* t *is enabled in marking* M *and leads to marking* M'. *This transition relation is given by:*

$$M\ [t\rangle\ M' \ \Leftrightarrow \forall p \in P\ \ M(p) \geq F(p,t)(M) \ \ \wedge \ \ M' = M - F(p,t)(M) + F(t,p)(M)$$

A marked self-modifying net is a self-modifying net together with an initial marking.

The translation that is sketched in Fig. 9 involves reset arcs. This fact may surprise in view of Dufourd's result according to which the boundedness of Petri nets with reset arcs is undecidable. In many cases however such reset arcs are not needed, for instance Fig. 10 shows a self-modifying net equivalent to the reversible reconfigurable net of Fig. 3. The net of Fig. 10 is even stratified [4] which means there exists a stratification of the set of places into layers so that the flow relations attached to a place involve only the content of places of lower layers.

Definition 16 *A stratified Petri net is a self-modifying net* $N = (P, T, F)$ *whose set of places* P *can be partially ordered so that if* $F(p,t)$ *or* $F(t,p)$ *is of the form* $\lambda_0 + \sum_{i=1}^{m} \lambda_i \cdot p_i$, *then* $\lambda_i = 0$ *for every indices* i *such that* $p \leq p_i$.

More precisely, since each structure modifying rule is a bijection, the number of places that exists in a current mode of operation of a reversible reconfigurable net remains unchanged when a structure modifying rule occurs. Therefore a place p that exists in the initial marking

can represent the set of places $\varphi(p)$ for φ ranging in the group of permutations of the set of places generated by the permutations φ_r associated with some structure modifying rule r. For instance the place p'_2 (respectively the place p'_4) in Fig. 10 stands for the place p_2 (resp. the place p_4) of Fig. 3 when the system is in the sequential mode of operation; and it stands for the place p_5 (resp. p_3) when the system is in the parallel mode of operation. The flow relations attached to that place are no longer constant but depend on the current mode of operation. A reversible reconfigurable net can roughly be viewed as a parametric Petri net, whose instances are its configurations, controlled by an automaton that describes how to switch from one configuration to another one.

Observation 17 *An automaton $A = (S, E, T, s_0)$ is isomorphic to the marking graph of a stratified Petri net if and only if*

$$A \cong N_k \rtimes (N_{k-1} \rtimes \dots N_2 \rtimes (N_1 \rtimes B_E) \dots)$$

where $B_E = (\{*\}, E, \{*\} \times E \times \{*\}, *)$ is a bouquet and N_1 to N_k are parametric Petri nets.

In particular the marking graph of a Petri net is of the form $N \rtimes B_E$, i.e. a Petri net can be construed as N_α for a parametric Petri Net N with one parameter ($\Pi = \{*\}$) where $\alpha(*) = 1$. The meaning of symbol $\rtimes$ was introduced in Def. 2.

In Prop. 12 we fail to have an isomorphism because the domain of the initial marking may be mapped to the same set by different permutations of Φ. If we exclude nets for which this phenomenon occurs we actually have an isomorphism as shown by the following proposition.

Definition 18 *A reversible reconfigurable net is termed simple if the only permutation of Φ (the group generated by the permutations φ_r associated with structure modifying rule $r \in R$) that maps the domain of the initial marking to itself is the identity.*

Let $\Omega_0 = D(M_0)$ be the domain of the initial marking, then equivalently the net is simple if $\forall \varphi, \psi \in \Phi \quad \varphi(\Omega_0) = \psi(\Omega_0) \Rightarrow \varphi = \psi$.

Proposition 19 *The marking graph of a marked simple reversible reconfigurable net is isomorphic to the marking graph of an automaton controlled Petri net:*

$$\mathbf{mg}(N, M_0) \cong N(\Phi) \rtimes A(\Phi)$$

Moreover, in that case one has also an isomorphism

$$N(\Phi) \rtimes A(\Phi) \cong N(\Phi) \rtimes_\Delta A(\mathcal{O})$$

where $A(\mathcal{O}) = (S, E, T', s_0)$ is the automaton whose states are the modes of operations of the marked net: $S = \mathcal{O}(N, M_0)$, whose initial state $s_0 = \Omega_0$ is the mode of operation $\Omega_0 = D(M_0)$ of the initial marking. The non idle events of $A(\mathcal{O})$ are the structure modifying rules $r \in R$ with

$$\Omega \xrightarrow{r} \Omega' \quad \Leftrightarrow \quad r : P_1 \to P_2 \ \wedge \ P_1 \subseteq \Omega \ \wedge \ P_2 \cap \Omega = \emptyset \ \wedge \ \Omega' = \varphi_r(\Omega)$$

The state encoding function matrix $\Delta : \mathcal{O}(N, M_0) \times \Phi \to \mathbb{N}$ *is given by*

$$\Delta(\Omega, \varphi) = \textbf{ if } \Omega = \varphi(\Omega_0) \textbf{ then } 1 \textbf{ else } 0$$

Proof: We saw that the map $f : N(\Phi) \rtimes A(\Phi) \to \mathbf{mg}(N, M_0)$ given by $f(M, \varphi) = M \circ \varphi^{-1}$ is a saturating morphism of automata. Since the corresponding automata are deterministic and accessible the saturating morphism is surjective and it is an isomorphism if and only if it is injective. Suppose $f(M, \varphi) = f(M', \varphi')$, let Ω be the domain of $M \circ \varphi^{-1} = M' \circ \varphi'^{-1}$. Then $\Omega = \varphi(\Omega_0) = \varphi'(\Omega_0)$ where $\Omega_0 = D(M_0)$ is the domain of the initial marking and it follows by simplicity that $\varphi = \varphi'$ and thus $M = M'$ which shows that f is injective. Hence

$$\mathbf{mg}(N, M_0) \cong N(\Phi) \rtimes A(\Phi)$$

Now, one has a bijective correspondance between the reachable configurations of $N(\Phi) \rtimes A(\Phi)$ and the reachable configurations of $N(\Phi) \rtimes_\Delta A(\mathcal{O})$ taking (M, φ) to $(M, \varphi(\Omega_0))$ and conversely (M, Ω) to $(M, \Delta(\Omega))$. This correspondance is an isomorphism of automata, since if $\varphi = \Delta(\Omega)$ and $\varphi' = \Delta(\Omega')$ one has

$$\begin{array}{lll}
(M, \Omega) \xrightarrow{e} (M', \Omega') & \text{in} & N(\Phi) \rtimes_\Delta A(\mathcal{O}) \\
\Leftrightarrow \quad [\text{if } e = t] & M\ [t/\Delta(\Omega)\rangle\ M' & \text{in } N(\Phi) \quad \wedge \quad \Omega' = \Omega \\
\quad\ [\text{if } e = r] & M = M' \quad \wedge\ P_1 \subseteq \Omega\ \wedge\ P_2 \cap \Omega = \emptyset\ \wedge\ \Omega' = \varphi_r(\Omega) & \\
\Leftrightarrow \quad [\text{if } e = t] & M\ [t/\varphi\rangle\ M' & \text{in } N(\Phi) \quad \wedge\ \varphi' = \varphi \\
\quad\ [\text{if } e = r] & M = M' \quad \wedge\ P_1 \subseteq \varphi(\Omega_0)\ \wedge\ P_2 \cap \varphi(\Omega_0) = \emptyset\ \wedge\ \varphi' = \varphi_r \circ \varphi & \\
\Leftrightarrow \quad (M, \varphi) \xrightarrow{e} (M', \varphi') & \text{in} & N(\Phi) \rtimes A(\Phi)
\end{array}$$

∎

The advantage of this second presentation (see Fig. 11) is that the automaton $A(\mathcal{O})$ is isomorphic to the marking graph of a 1-safe Petri net. In this manner one has a distributed representation of the control. More precisely two structural modification rules are independent if they are concerned with disjoint subsets of places of the controlled parametric Petri net.

Because of the presence of a non trivial state encoding function, the automaton controlled Petri net $N(\Phi) \rtimes_\Delta A(\mathcal{O})$ does not lead directly to a representation by a stratified Petri net. It is necessary to encode the parameters $\varphi \in \Phi$ of net $N(\Phi)$ by places of a stratified Petri net with marking graph isomorphic to $A(\mathcal{O})$.

Proposition 20 *Let* $\mathcal{N} = (P, T, F, M_0)$ *be a 1-safe marked Petri net and* $X \subseteq 2^P$ *a set of 'properties' of* $\mathcal{N}$*. There exists a 1-safe marked stratified Petri net* $\mathcal{N}(X) = (\tilde{P}, T, \tilde{F}, \tilde{M}_0)$ *such that (i)* $\mathcal{N}(X)$ *is a conservative extension of* $\mathcal{N}$ *in the sense that* $P \subseteq \tilde{P}$*,* $\tilde{F}(p, t) = F(p, t)$ *and* $\tilde{M}_0(p) = M_0(p)$ *for every* $p \in P$ *and the mapping* $(\tilde{M} \subseteq \tilde{P}) \mapsto (\tilde{M} \cap P \subseteq P)$ *induces an isomorphism between the respective marking graphs of* $\mathcal{N}$ *and* $\mathcal{N}(X)$*, and (ii) each property of* X *is represented in* $\mathcal{N}(X)$*: for each* $x \in X$ *there exists a place* $p_x \in \tilde{P}$ *such that for every reachable marking* $\tilde{M}$ *of* $\mathcal{N}(X)$ *one has* $\tilde{M}(p_x) = 1$ *if and only if* $x \subseteq \tilde{M} \cap P$*.*

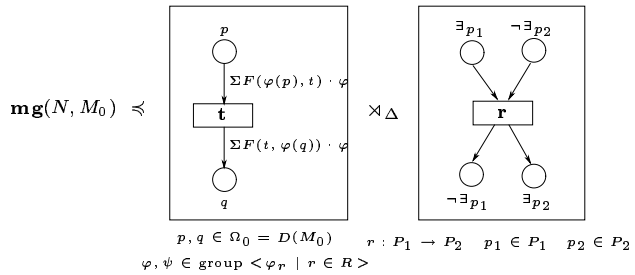


Figure 11: a reversible reconfigurable net as a cascade composition of Petri nets

Proof: Let us consider first the case where $X = 2^P$. We let $\tilde{P} = X = 2^P$ and we define $\tilde{F}(\tilde{p}, t)$ and $\tilde{F}(t, \tilde{p})$ for $\tilde{p} \in \tilde{P}$ as follows. The flow relation $\tilde{F}$ is an extension of F, i.e. $\tilde{F}(\{p\}, t) = F(p, t)$ and $\tilde{F}(t, \{p\}) = F(t, p)$ for $p \in P$. Let $x \in \tilde{P}$ with cardinality at least two, if $\exists p \in x\ F(p, t) = 1$ and $\forall p \in x\ F(t, p) = 0$ then we let $\tilde{F}(x, t) = \{p \in x \mid F(p, t) = 0\}$ (i.e. $\tilde{F}(x, t) = y \in \tilde{P}$ with $y \subset x$) else we let $\tilde{F}(x, t) = 0$. Symmetrically, if $\exists p \in x\ F(t, p) = 1$ and $\forall p \in x\ F(p, t) = 0$ then we let $\tilde{F}(t, x) = \{p \in x \mid F(t, p) = 0\}$ else we let $\tilde{F}(t, x) = 0$. Finally we let $\tilde{M}_0(x) = 1$ if $\forall p \in x\ M_0(p) = 1$ else we let $\tilde{M}_0(x) = 0$. Let $\mathcal{N}(X)$ denote the stratified Petri net $(\tilde{P}, T, \tilde{F}, \tilde{M}_0)$ so obtained. If we identify $p \in P$ with the one-element set $\{p\} \in \tilde{P}$, we have that $P \subseteq \tilde{P}$, and $\tilde{F}(p, t) = F(p, t)$ and $\tilde{M}_0(p) = M_0(p)$ for every $p \in P$. We shall prove that the map that takes marking M of $\mathcal{N}$ to marking $\tilde{M}$ of $\mathcal{N}(X)$ given by

$$\tilde{M}(x) = \begin{cases} 1 & \text{if} \quad \forall p \in x \quad M(p) = 1 \\ 0 & \text{otherwise} \end{cases}$$

is an isomorphism between their marking graphs. It follows therefrom that the stratified Petri net $\mathcal{N}(X)$ is 1-safe. Moreover the accessible markings of $\mathcal{N}(X)$ are of the form $\tilde{M}$ for M some accessible marking of $\mathcal{N}$ and the inverse isomorphism follows from the identity $M = \tilde{M} \cap P$ where markings of 1-safe nets are viewed as sets of places. Thus place x in $\mathcal{N}(X)$ represents x viewed as a property $x \subseteq P$ of $\mathcal{N}$ since $\tilde{M}(x) = 1$ if and only if $\forall p \in x \quad M(p) = 1$, i.e. $x \subseteq M = \tilde{M} \cap P$ for every accessible marking $\tilde{M}$. Thus if the map $M \mapsto \tilde{M}$ is an isomorphism, the net $\mathcal{N}(X)$ is a conservative extension of $\mathcal{N}$ in which every property of X is represented. Since marking graphs of nets are deterministic and reachable, in order to verify that $M \mapsto \tilde{M}$ is an isomorphism it is sufficient to check that *(i)* a transition t is enabled in marking M if and only if it is enabled in $\tilde{M}$ and *(ii)* if $M\ [t\rangle\ N$ and $\tilde{M}\ [t\rangle\ \mathbf{N}$, then $\mathbf{N} = \tilde{N}$; which amounts to prove that for every transition $t \in T$ and property $x \subseteq P$ one has

$$\begin{array}{ll} (i) & M\ [t\rangle\ \Rightarrow \tilde{F}(x, t) \cdot \tilde{M} \leq \tilde{M}(x) \\ (ii) & M\ [t\rangle\ N\ \Rightarrow \tilde{N}(x) = \tilde{M}(x) - \tilde{F}(x, t) \cdot \tilde{M} + \tilde{F}(t, x) \cdot \tilde{M} \end{array}$$

We proceed by case analysis:

1. Assume there exists $p \in x$ such that $F(p,t) = 1$ and $q \in x$ such that $F(t,q) = 1$. Then if $M\ [t\rangle\ M'$ we necessarily have $M(q) = 0$ and $M'(p) = 0$ and therefore $\tilde{M}(x) = \tilde{M}'(x) = 0$. Since $\tilde{F}(x,t) = \tilde{F}(t,x) = 0$ conditions *(i)* and *(ii)* are satisfied.

2. Assume for all $p \in x$ $F(t,p) = 0$ and there exists some $q \in x$ such that $F(q,t) = 1$. Then $\tilde{F}(x,t) = y$ where $y = \{p \in x \mid F(p,t) = 0\}$ and $\tilde{F}(t,x) = 0$. If $M\ [t\rangle\ M'$ then $\tilde{M}(x) = \tilde{M}(y)$ because every input places of t are marked in M, and $\tilde{M}'(x) = 0$. Conditions *(i)* and *(ii)* are thus satisfied.

3. Assume for all $p \in x$ $F(p,t) = 0$ and there exists some $q \in x$ such that $F(t,q) = 1$. Then $\tilde{F}(t,x) = y$ where $y = \{p \in x \mid F(t,p) = 0\}$ and $\tilde{F}(x,t) = 0$. If $M\ [t\rangle\ M'$ then $\tilde{M}(x) = 0$ and $\tilde{M}'(x) = \tilde{M}'(y)$ because every output places of t are marked in M'. Conditions *(i)* and *(ii)* are thus satisfied.

4. Assume for all $p \in x$ $F(p,t) = F(t,p) = 0$, then $\tilde{F}(x,t) = \tilde{F}(t,x) = 0$ and $\tilde{M}(x) = \tilde{M}'(x)$ for every pair of markings M and M' such that $M\ [t\rangle\ M'$. Conditions *(i)* and *(ii)* are thus satisfied.

Now if $X \subset P$, the net previously defined for $X = 2^P$, i.e. $\mathcal{N}(2^P)$, satisfies the requirements of the proposition. However another and smaller solution is obtained by the restriction of $\mathcal{N}(2^P)$ to the set of places $\tilde{P} = \{y \subseteq P \mid \exists x \in X \quad y \Subset x\}$ where $\Subset$ is the reflexive and transitive closure of the relation $\ll$ given by $y \ll x \Leftrightarrow \exists t \in T \cdot$

$$(\forall p \in x\ F(t,p) = 0\ \wedge\ \exists p \in x\ F(p,t) = 1\ \wedge\ y = \{p \in x \mid F(p,t) = 0\}) \quad \vee$$
$$(\forall p \in x\ F(p,t) = 0\ \wedge\ \exists p \in x\ F(t,p) = 1\ \wedge\ y = \{p \in x \mid F(t,p) = 0\})$$

i.e. we first let in $\mathcal{N}(X)$ all places of $\mathcal{N}(2^P)$ that are properties in X and iteratively add all others places of $\mathcal{N}(2^P)$ that appears as flow arc inscriptions of places already incorporated in $\mathcal{N}(X)$. It is in some sense the stratified subnet of $\mathcal{N}(2^P)$ induced by X. ∎

Since $\mathcal{N}(X)$ is smaller than $\mathcal{N}(Y)$ when $X \subseteq Y$, there is less 'loss of concurrency' when replacing net $\mathcal{N}$ by $\mathcal{N}(X)$ than when it is replaced by $\mathcal{N}(Y)$. I.e. the internalization of more properties is paid by a loss in concurrency. At one extreme if we want all subsets of places to be represented then the resulting system becomes completely sequential.

From Prop. 19 and Prop. 20 it follows that

Proposition 21 *Any marked simple reversible reconfigurable net can be associated with a marked stratified Petri net with isomorphic marking graph and whose set of transitions is the set of events of the original reconfigurable net.*

The corresponding stratified Petri net is represented in Fig. 12. More precisely if $\mathcal{N} = (P, T, R, F, M_0)$ is a marked simple reversible reconfigurable net, the associated marked stratified Petri net is $\mathcal{N}' = (P', T', F', M_0')$ defined as follows. $P' = \Omega_0 \cup \exists P \cup \neg\exists P \cup \mathcal{O}(N, M_0)\downarrow$ where Ω_0 is the domain of the initial marking, $\exists P$ and $\neg\exists P$ are disjoint copies of P whose respective typical elements are noted $\exists_p$ and $\neg\exists_p$ for p ranging in P, and

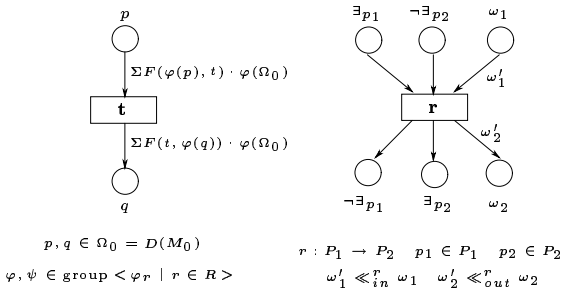


Figure 12: the stratified Petri net equivalent to a simple reversible reconfigurable net

$\mathcal{O}(N, M_0)\downarrow \;\; = \{\gamma \subseteq P \mid \exists \Omega \in \mathcal{O}(N, M_0) \quad \gamma \Subset \Omega\}$ is the downward closure of the set $\mathcal{O}(N, M_0)$ of modes of operation for the reflexive and transitive closure $\Subset$ of the relation $\ll = \bigcup_{r \in R} (\ll^r_{in} \cup \ll^r_{out})$ where

$$\gamma' \ll^r_{in} \gamma \Leftrightarrow [(P_1 \cup \neg P_2) \cap \gamma \neq \emptyset \;\; \wedge \;\; (\neg P_1 \cup P_2) \cap \gamma = \emptyset \;\; \wedge \;\; \gamma' = \gamma \setminus (P_1 \cup \neg P_2)]$$

and

$$\gamma' \ll^r_{out} \gamma \Leftrightarrow [(\neg P_1 \cup P_2) \cap \gamma \neq \emptyset \;\; \wedge \;\; (P_1 \cup \neg P_2) \cap \gamma = \emptyset \;\; \wedge \;\; \gamma' = \gamma \setminus (\neg P_1 \cup P_2)]$$

$T' = T \cup R$ is the set of events of $\mathcal{N}$. The flow relation is given by the following identities where $t \in T$ and $r : P_1 \to P_2 \in R$

$$F'(p,t) = \begin{cases} \sum_{\varphi \in \Phi} F(\varphi(p), t) \cdot \varphi(\Omega_0) & \text{if} \quad p \in \Omega_0 \\ 0 & \text{otherwise} \end{cases}$$

$$F'(t,p) = \begin{cases} \sum_{\varphi \in \Phi} F(t, \varphi(p)) \cdot \varphi(\Omega_0) & \text{if} \quad p \in \Omega_0 \\ 0 & \text{otherwise} \end{cases}$$

$$F'(r,p) = \begin{cases} 1 & \text{if} \quad p \in P_1 \cup \neg P_2 \\ \gamma' & \text{if} \quad p = \gamma \in \mathcal{O}(N, M_0)\downarrow \;\; \text{and} \;\; \gamma' \ll^r_{in} \gamma \\ 0 & \text{otherwise} \end{cases}$$

$$F'(p,r) = \begin{cases} 1 & \text{if} \quad p \in \neg P_1 \cup P_2 \\ \gamma' & \text{if} \quad p = \gamma \in \mathcal{O}(N, M_0)\downarrow \;\; \text{and} \;\; \gamma' \ll^r_{out} \gamma \\ 0 & \text{otherwise} \end{cases}$$

The initial marking M'_0 is given by

$$M'_0(p) = \begin{cases} M_0(p) & \text{if} \quad p \in \Omega_0 \\ 1 & \text{if} \quad p = \exists_q \;\; \text{where} \;\; q \in \Omega_0 \;\; \text{or} \;\; p = \neg\exists_q \;\; \text{where} \;\; q \notin \Omega_0 \\ 0 & \text{if} \quad p = \exists_q \;\; \text{where} \;\; q \notin \Omega_0 \;\; \text{or} \;\; p = \neg\exists_q \;\; \text{where} \;\; q \in \Omega_0 \\ 1 & \text{if} \quad p = \gamma \in \mathcal{O}(N, M_0)\downarrow \;\; \text{and} \;\; \gamma \subseteq M_0 \\ 0 & \text{if} \quad p = \gamma \in \mathcal{O}(N, M_0)\downarrow \;\; \text{and} \;\; \gamma \not\subseteq M_0 \end{cases}$$

6 Some Examples

In this section we give some examples of concurrent systems subject to dynamical changes. They are respectively modelized by a reversible reconfigurable net (Section 6.1), a Petri net (Section 6.2), and a stratified Petri net (Section 6.3) thus witnessing that the choice of a particular model much depends on the nature of the problem to be modelized.

6.1 Sending Information Packages in a Transmission Net

Reversible reconfigurable nets are well adapted to describe large distributed systems where dynamic changes may occur in different locations and have local effects, even if these changes may not be totally independent. When on the contrary changes require a global synchronization of the system, as in the example of the following section, then Petri nets will be better suited. Let us present a small sized example but which hopefully illustrate this kind of situation. We consider a transmission net that receives from two machines (represented by transitions t_1 and t_8 in Fig. 13) blocks of information to be carried to a main building

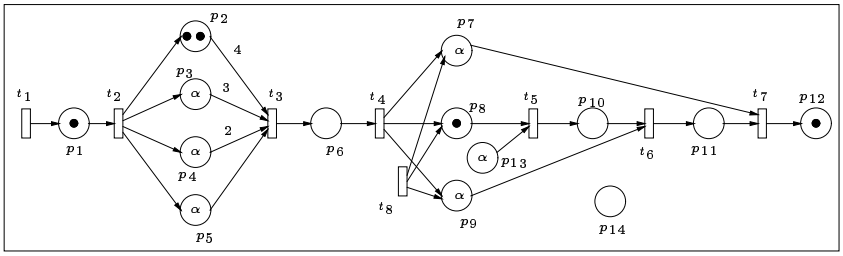


Figure 13: a marked reversible reconfigurable net

(represented by place p_{12}).

This net has two different parts delimited by places p_1, p_6, and p_{12}. In the first part, there exist four possibilities depending on the decision of an operator. He can decide to send the packages in ones, in twos, in threes or in fours which are respectively represented by the paths $t_2p_5t_3$, $t_2p_4t_3$, $t_2p_3t_3$, $t_2p_2t_3$. In the second part, the net offers three possibilities for the forwarding of data in the net:

1. packages follow the normal path through $t_4p_8t_5p_{10}t_6p_{11}t_7p_{12}$;
2. packages avoid transition t_5 and follow the path $t_4p_9t_6p_{11}t_7p_{12}$; finally,
3. packages can go directly through the path $t_4p_7t_7p_{12}$.

The system is described by the reversible reconfigurable net of Fig. 13 which consists of 14 places, 8 transitions and 6 structure modifying rules $R = \{r_1, r_2, r_3, r_4, r_5, r_6\}$ where

$$\begin{array}{ll} r_1 = \{p_5\} \rightarrow \{p_2\} & r_2 = \{p_4\} \rightarrow \{p_3\} \\ r_3 = \{p_3\} \rightarrow \{p_2\} & r_4 = \{p_7, p_{13}\} \rightarrow \{p_8, p_{14}\} \\ r_5 = \{p_9\} \rightarrow \{p_7\} & r_6 = \{p_8, p_{14}\} \rightarrow \{p_9, p_{13}\} \end{array}$$

The initial marking represented in Fig. 13 is

$$M_0 = \{1, 2, \alpha, \alpha, \alpha, 0, \alpha, 1, \alpha, 0, 0, 1, \alpha, 0\}$$

This net has 12 different configurations shown in Fig. 14. The corresponding ACPN is

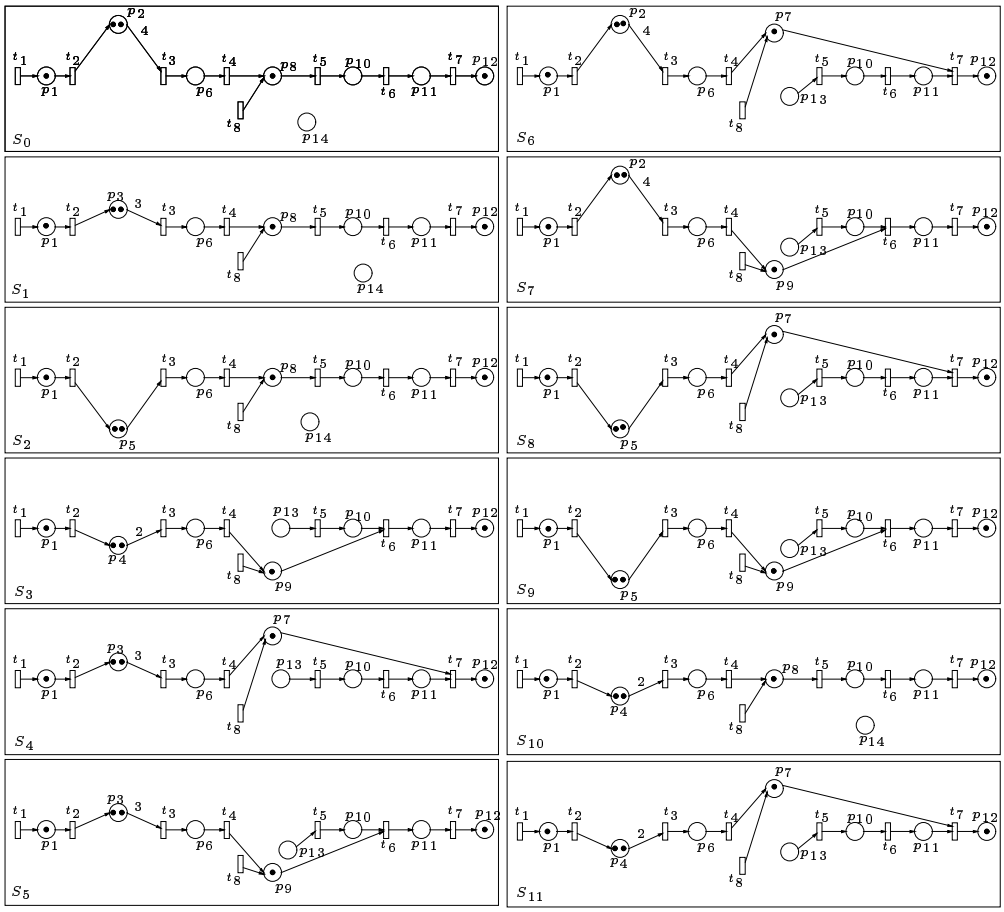


Figure 14: The 12 configurations of the reversible reconfigurable net of Fig. 13

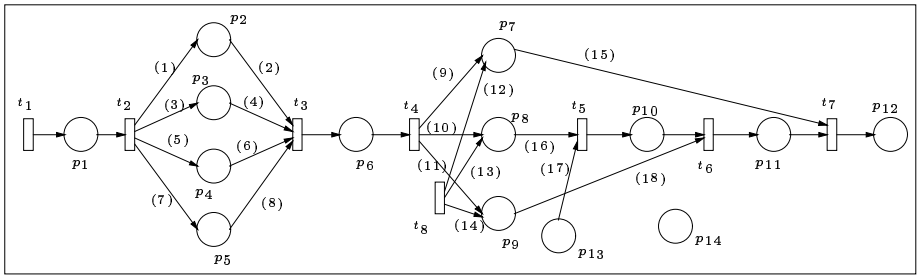


Figure 15: ACPN of the reversible reconfigurable net of Fig. 13

shown in Fig. 15. Numbered arc weights in the ACPN of Fig. 15 are:

(1)	$S_0 + S_6 + S_7$	(2)	$4S_0 + 4S_6 + 4S_7$
(3)	$S_1 + S_4 + S_5$	(4)	$3S_1 + 3S_4 + 3S_6$
(5)	$S_3 + S_{10} + S_{11}$	(6)	$2S_3 + 2S_{10} + 2S_{11}$
(7)	$S_2 + S_8 + S_9$	(8)	$S_2 + S_8 + S_9$
(9)	$S_4 + S_6 + S_8 + S_{11}$	(10)	$S_0 + S_1 + S_2 + S_{10}$
(11)	$S_3 + S_5 + S_7 + S_9$	(12)	$S_4 + S_6 + S_8 + S_{11}$
(13)	$S_0 + S_1 + S_2 + S_{10}$	(14)	$S_3 + S_5 + S_7 + S_9$
(15)	$S_4 + S_6 + S_8 + S_{11}$	(16)	$S_0 + S_1 + S_2 + S_{10}$
(17)	$S_3 + S_4 + S_5 + S_6 + S_7 + S_8 + S_9 + S_{11}$	(18)	$S_3 + S_5 + S_7 + S_9$

The corresponding automaton not shown here may be calculated from the set of configurations and the set of structure modifying rules.

6.2 An Assembly Shop with Different Modes of Operations

By contrast with the previous section, the example considered below requires a synchronization of the whole system before any reconfiguration can take place. For that reason reconfigurable nets are inadapted and a solution using Petri nets will be favoured. The system is a factory consisting of assembly shops communicating pieces through conveyor belts. Each assembly shop contains some machines in order to process pieces, some stores and a robot that takes pieces to and from conveyor belts and loads and unloads machines and stores. Depending on the product that the factory is currently producing each assembly shop follows some particular mode of operation. Figure 16 displays an assembly shop with three different modes of operations. In Mode I, the robot receives a piece on belt A, it loads machine M_1 with that piece so that some operation can be processed, then the piece is temporarily stored until it can be loaded to machine M_2 for another operation and then forwarded to another assembly shop via belt B. The places of the associated Petri net, on the left of Fig. 16, corresponds to the following operations

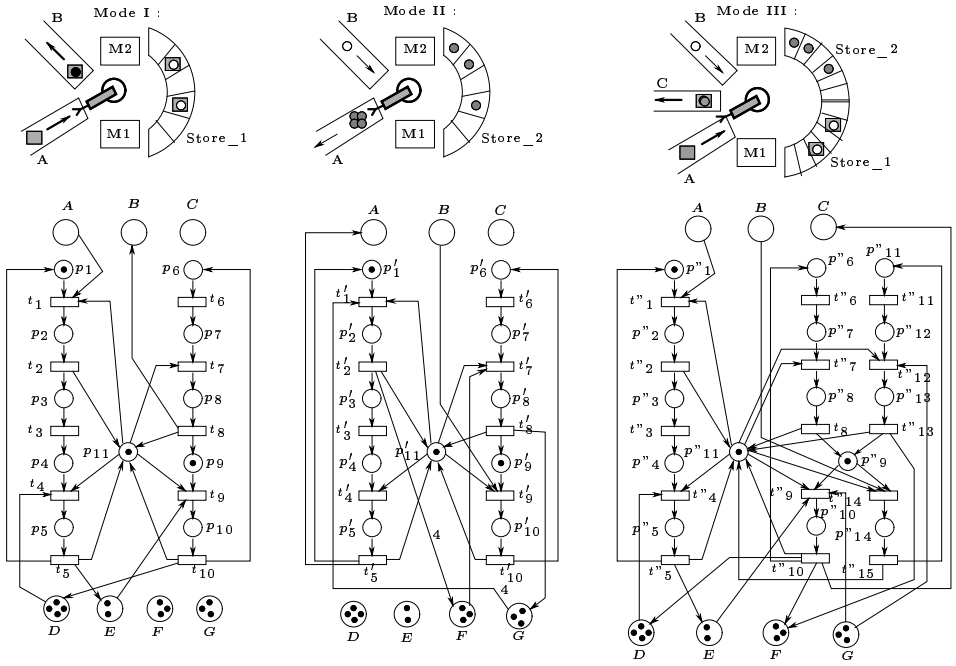


Figure 16: an assembly shop with different modes of operations

p_1	:	machine M_1 waits for a piece
p_2	:	the robot loads machine M_1 with a piece received from belt A
p_3	:	machine M_1 is working
p_4	:	machine M_1 ready to unload
p_5	:	the robot unloads machine M_1 and store the piece in $Store_1$
p_6	:	machine M_2 is working
p_7	:	machine M_2 ready to unload
p_8	:	the robot unloads machine M_2 and send the piece to belt B
p_9	:	machine M_2 waits for a piece
p_{10}	:	the robot loads machine M_2 with a piece from $Store_1$
A	:	pieces ready to be taken from belt A
B	:	pieces on belt B
C	:	pieces on belt C
D	:	number of free slots in $Store_1$
E	:	number of pieces in $Store_1$
F	:	number of free slots in $Store_2$
G	:	number of pieces in $Store_2$

If we forget about the places A, B, C, D, E, F, and G associated with the conveyors belts and the stores, this Petri net consists of three state machine components associated respectively with machines M_1 and M_2 and the robot, in particular all the places p_i are one-safe and can be interpreted as properties. Then the property $stable = p_1 \wedge p_{11} \wedge p_9$ which states that machines M_1 and M_2 are in their idle state, waiting for a piece to be loaded, and the robot is available is a *stable* property meaning that from any accessible marking it is possible to reach some marking verifying this property. The other two modes of operation, displayed on the center and right parts of Fig. 16, are similarly defined. In Mode II, pieces arrive from belt B, they are loaded first on machine M_2 that perform some transformation on it and then put into $Store_2$. Four pieces may be picked up, if available, from this store to be assembled on machine M_1 in order to form a new piece which is sent to belt A. In Mode III, machine M_1 can process the same operation as in Mode I on pieces coming from belt A, the resulting pieces are stored in $Store_1$. Machine M_2 can process the same operation as in Mode II on pieces coming from belt B, the resulting pieces are stored in $Store_2$. Machine M_2 can also assemble one piece coming from each of the stores and send the resulting piece on belt C. Notice that each store always contains the same kind of pieces, that belt C is used only on Mode III, and that the direction of the conveyor belts may change when swithching from one mode of operation to another one. Suppose now that a change in the factory requires that this assembly shop switches from Mode I to Mode II. In order not to confuse different kind of pieces, and also because the direction of some conveyor belts may change, it is important that, before the change can actually take place, the system has reached a stable state in which each machine is idle with no piece loaded and that there is no piece remaining on the conveyor belts, only the contents of the stores are irrelevant. For that reason the change requires a synchronisation of the whole system. In order not to complicate too much this example we shall assume that there is a mechanism for withdrawing pieces from the coveyor

belts when the system asks for a change (the pieces are put back on the corresponding belt as soon as the system comes back to an adequate configuration). The change is then performed in two stages as illustrated in Fig. 17. First the system asks for a change from

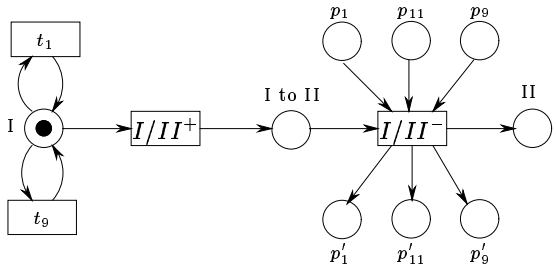


Figure 17: changing from mode I to mode II

Mode I to Mode II (transition I/II^+) by moving the token from place I to place "I to II". In doing so transitions t_1 and t_9 which test that the system is in Mode I are inhibited, the machines M_1 and M_2 thus refuse to proceed any new piece before the change actually takes place. The Petri net correponding to Mode I will therefore necessarily reach the stable state where places p_1, p_{11}, and p_9 contain one token each (the other places p_i are empty). The change can then take place (firing of transition I/II^-). Then the Petri net correponding to Mode I is inactivated since all its places p_i are empty while the Petri net correponding to Mode II becomes active. Of course transitions I/II^+ and I/II^- must do the similar changes for all those assembly shops that are concerned by the change. We think that this example, even though quite simple, is representative of a some kind of dynamic changes in distributed systems.

6.3 Flows in a Network

Our last example is also representative of a whole class of systems where changes arise from the actions of agents that regulate some distributed system. Stratified Petri nets seem to be a well-adapted formalism for this kind of situations. We consider a workflow system consisting of agents exchanging data by asynchronous message passing. The system is described by a finite acyclic directed graph with one source node and one sink node. Nodes of the graph represent agents whereas the arcs of the graph represent communication channels between them. In order to proceed correctly, a task should enter the system at its source and follow any path to the sink. So different paths correspond to various manners of completing the task. A flow of this graph gives a repartition of paths from source to sink. For that purpose it is convenient to add one fictituous arc ℓ_0 from the sink to the source. We recall that a flow is then a map assigning a non-negative integral weight to each arc so that in each node the total weight of the entering arcs is equal to the total weight of the exiting arcs. For instance the flow represented in Fig. 18 corresponds to a certain policy of repartition for the tasks entering the system: for each six tasks entering the systems two are directed to node 2 and four to node 3; node 5 should receive three tasks from node 3 for each task received from from node 2, and these four tasks should all be directed to node 8; etc. Each such flow

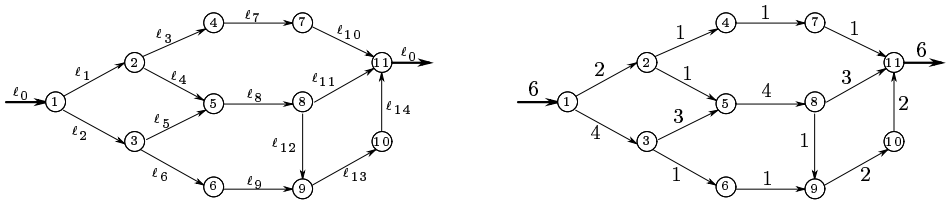


Figure 18: a flow (on the right-hand side) of the workflow system (on the left-hand side)

determines a load rate for each node, for instance in the example of Fig. 18 the load rate of node 8 is $\frac{4}{6} = \frac{2}{3}$ whereas the load rate of node 7 is $\frac{1}{6}$. In case there are modifications in the performance of some nodes one may wish to modify the current flow. For instance by decreasing of one unit the weights of the arcs ℓ_4, ℓ_8, and ℓ_{11}, and at the same time increasing by one unit the weights of the arcs ℓ_3, ℓ_7, and ℓ_{10}. The difference between two flows is a cycle. As it is well-known from graph theory (see [6]) the cycles of a graph form a $\mathbb{Z}$-module a generating set of which is associated with the chords of some spanning tree. Figure 19 gives some generating set for the cycles of the graph of Fig. 18. For instance the cycle σ adds one unit to the weight of ℓ_2 and ℓ_5 and substracts one unit to the weight of ℓ_1 and ℓ_4. The cycles may then be viewed as the transitions of a Petri net whose places are associated with the arcs of the graph. A marking is thus a non-negative integral weight of the arcs. Figure 20 shows the Petri net (in bold face and in superimposition with the graph) resulting from the generating set of Fig. 19. We have added however an extra cycle σ_6 and an extra place $\overline{\ell}_0$ whose initial content gives an upper bound on the total weight of a flow (i.e. the value of the fictituous arc ℓ_0). One may wish to verify that the set of accessible

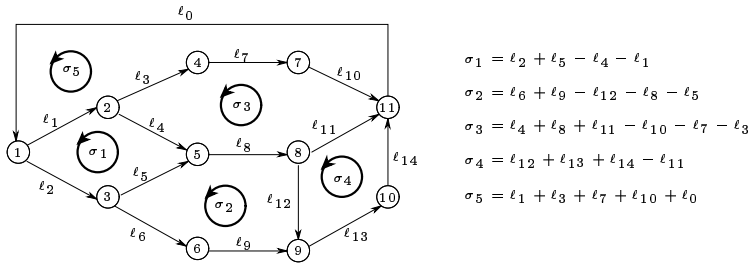


Figure 19: a generating set of cycles

markings of Fig. 20 is the set of the flows of the graph whose total weight does not exceed the initial value of $\overline{\ell}_0$. If that initial value is 10 as indicated in the figure the marking graph

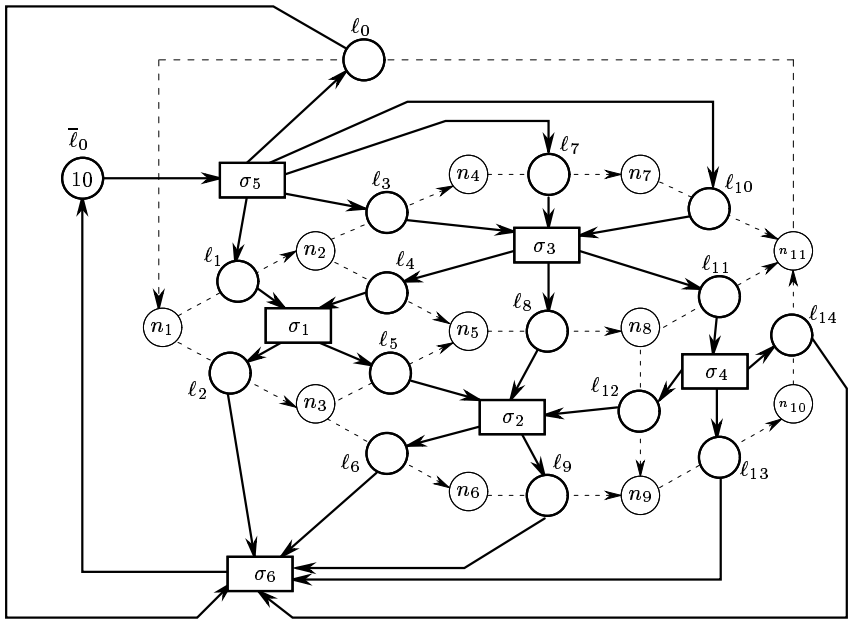
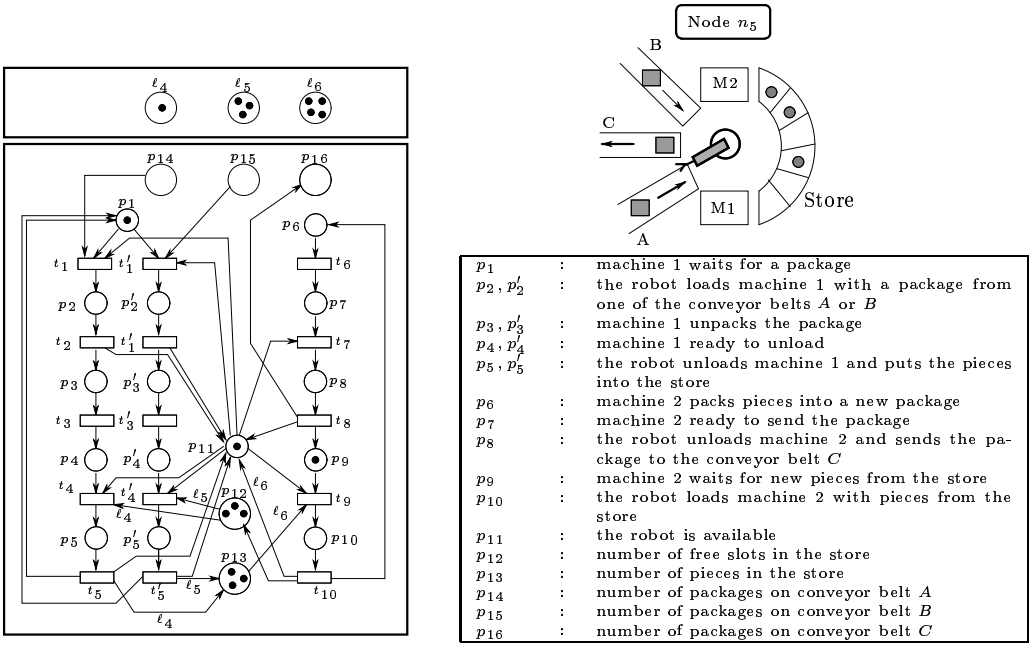


Figure 20: the net associated with the generating set of cycles of Fig. 19

has already 5005 states and 21 164 transitions. Now the whole system can be modelized by a stratified Petri net consisting of the Petri net of Fig. 20 together with a parametric Petri net attached to each node of the graph whose parameters are the places associated with the arcs connected to that node. The Petri net associated with Node n_5 is for instance given in Fig. 21. It can be viewed as an assembly shop, similar to the example of Section 6.2, that receives and sends packages whose sizes are given by the values of parameters. When there



p_1	:	machine 1 waits for a package
p_2, p_2'	:	the robot loads machine 1 with a package from one of the conveyor belts A or B
p_3, p_3'	:	machine 1 unpacks the package
p_4, p_4'	:	machine 1 ready to unload
p_5, p_5'	:	the robot unloads machine 1 and puts the pieces into the store
p_6	:	machine 2 packs pieces into a new package
p_7	:	machine 2 ready to send the package
p_8	:	the robot unloads machine 2 and sends the package to the conveyor belt C
p_9	:	machine 2 waits for new pieces from the store
p_{10}	:	the robot loads machine 2 with pieces from the store
p_{11}	:	the robot is available
p_{12}	:	number of free slots in the store
p_{13}	:	number of pieces in the store
p_{14}	:	number of packages on conveyor belt A
p_{15}	:	number of packages on conveyor belt B
p_{16}	:	number of packages on conveyor belt C

Figure 21: the behaviour of a node of the network

are several outcomming arcs then packages must be sent in turn on each of the outcomming conveyor belts in order to respect the ratios given by the current flow.

References

[1] Araki, T., and Kasami, T., *Some decision problems related to the reachability problem for Petri nets.* Theoretical Computer Science, 3(1) (1977) 85–104.

[2] van der Aalst, W.M.P., *Verification of Workflow Nets.* Proceedings of icatpn'97, volume 1248 of Lecture Notes in Computer Science, Springer Verlag (1997) 407–426.

[3] Badouel, E., *Splitting of Actions, Higher-Dimensional Automata and Net Synthesis.* Inria Research Report No 3013 (1996).

[4] Badouel, E., and Darondeau, Ph., *Stratified Petri Nets.* Proceedings of fct'97, volume 1279 of Lecture Notes in Computer Science, Springer Verlag (1997) 117–128.

[5] Badouel, E., and Oliver, J., *Reconfigurable Nets, a Class of High Level Petri Nets Supporting Dynamic Changes within Workflow Systems.* Inria Research Report PI-1163 (1998).

[6] Berge, C., *Graphes et hypergraphes.* Dunod (1970). English translation: *Graphs and Hypergraphs.* North Holland (1973).

[7] Ciardo, G., *Petri nets with marking-dependent arc cardinality: Properties and analysis.* Proceedings of icatpn'94, volume 815 of Lecture Notes in Computer Science, Springer Verlag (1994) 179–198.

[8] De Michelis, G., and Ellis, C.A., *Computer Supported Cooperative Work and Petri Nets.* Third Advanced Course on Petri Nets, Dagstuhl Castle, Germany (1996), volume 1492 of Lecture Notes in Computer Science, Springer Verlag (1998) 125–153.

[9] Diestel, R., *Graph Theory.* volume 173 of *Graduate Texts in Mathematics.* Springer Verlag (1996).

[10] Dufourd, C., and Finkel, A., and Schnoebelen, P., *Reset nets between decidability and undecidability.* In Proc. of icalp'98, volume 1443 of Lecture Notes in Computer Science, Springer Verlag (1998) 103–115.

[11] Ellis, C., Keddara, K., and Rozenberg, G., *Dynamic Change within Workflow Systems.* Proceedings of the Conference on Organizational Computing Systems, acm Press, New York (1995) 10–21.

[12] Ellis, C.A, and Nutt, G.J., *Modeling Enactment of Workflow Systems.* Proceedings of icatpn'93, volume 691 of Lecture Notes in Computer Science, Springer Verlag (1993) 1–16.

[13] Karp, R.M., and Miller, R.E., *Parallel program schemata.* Journal of Computer and System Sciences vol. 3 (1969) 147–195.

[14] Reutenauer, Ch., *Aspects mathématiques des réseaux de Petri.* Masson, Paris (1989). English translation by I. Craig: *The Mathematics of Petri Nets.* Prentice-Hall, New York (1990).

[15] Valk, R., *Self-Modifying Nets, a Natural Extension of Petri Nets.* Proceedings of icalp'78, volume 62 of Lecture Notes in Computer Science, Springer Verlag (1978) 464–476.

[16] Valk, R., *Generalizations of Petri Nets.* Proceedings of mfcs'81, volume 118 of Lecture Notes in Computer Science, Springer Verlag (1981) 140–155.

Unit´e de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unit´e de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unit´e de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unit´e de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unit´e de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399