Petri Net Process Decomposition with Application to Validation [1]

by **F.L. Ţiplea** [a] and **J. Desel** [b]

[a] Faculty of Computer Science
"Al. I. Cuza" University of Iaşi
6600 Iaşi, Romania
e-mail: fltiplea@infoiasi.ro

[b] Mathematisch–Geographische Fakultät
Lehrstuhl für Informatik
Katholische Universität Eichstätt
85072 Eichstätt, Germany
e-mail: joerg.desel@ku-eichstaett.de

Abstract

The aim of this paper is to show how any process of a Petri net which is a composition of two nets can be decomposed in processes of the component nets, and vice versa. Therefore, we introduce the concept of a process sample of a net with respect to a subnet and we claim that, under some requirements, jumping nets are suitable to generate all the process samples of a net. Then we give sufficient conditions under which a jumping Petri net can be simulated by a Petri net. Finally, a methodology for validation of Petri net models based on process decomposition, is proposed.

1 Introduction and Preliminaries

In designing a Petri net model for a complex system, it is desirable to decompose the system in modules, based for example on their functionality, to design Petri net models for each module, and then to compose these nets in order to get a Petri net model for the entire system. The behaviour of the resulting net should be a composition of behaviours of the component parts. The aim of this paper is to provide a framework for this approach, based on processes and jumping nets.

Among the many contributions with related topics we like to mention [4]. As we will do in our approach, composition of modules is based on fusion of places in [4]. The corresponding composition of processes representing the behaviour of a Petri net is defined by the *intersection* of process sets of the respective modules, i.e., each process of a single module also specifies the detailed behaviour (possible or not) of other modules. In contrast to [4], we abstract from details of other modules when defining the processes of a module; hence our composition operator of processes is based on the *union* of cuts.

In Section 2 we show how processes of nets can be decomposed in processes of smaller nets (subnets), and in Section 3 we argue that jumping nets can be successfully used as a tool for a compact generation of Petri net processes. Section 4 gives some conditions under which jumping Petri nets can be simulated by Petri nets. Finally, Section 5 presents some application of process decomposition to the validation of Petri net models.

In the remaining of this section we will fix the terminology and notation that we use in this paper (for details concerning Petri nets and processes the reader is referred to [1], [2], [5], [6]).

[1] This research was carried out while the first author was visiting Katholische Universität Eichstätt by a grant from DAAD.

The sum of two functions f_i from A_i into the set $\mathbf{N}$ of nonnegative integers, $i = 1, 2$, denoted by $f_1 + f_2$, is defined as $f_1(a)$ for all $a \in A_1 \Leftrightarrow A_2$, $f_2(a)$ for all $a \in A_2 \Leftrightarrow A_1$, and $f_1(a) + f_2(a)$ for all $a \in A_1 \cap A_2$. For a function $f : A \to B$ and a subset $C \subseteq A$, $f|_C$ denotes the restriction of f to C, and $f(C)$ the set of all images of $c \in C$ by f. By f^{-1} we denote the inverse-image function associated to f, that is $f^{-1}(b) = \{a \in A | f(a) = b\}$, for all $b \in B$. For a binary relation R, R^+ denotes the transitive closure of R.

A (finite) *P/T-net* is a 4-tuple $\Sigma = (S, T, F, W)$, where S and T are two finite sets (of *places* and *transitions*, resp.), $S \cap T = \emptyset$, $F \subseteq (S \times T) \cup (T \times S)$ is the *flow relation* and $W : (S \times T) \cup (T \times S) \to \mathbf{N}$ is the *weight function* of Σ, satisfying $W(x, y) = 0$ iff $(x, y) \notin F$. When $W(x, y) \leq 1$ for all $(x, y) \in F$, we may (and will) simplify the 4-tuple (S, T, F, W) to the 3-tuple (S, T, F). A *marking* of Σ is a function $M : S \to \mathbf{N}$, sometimes identified with a vector $M \in \mathbf{N}^{|S|}$. For $x \in S \cup T$ we set ${}^\bullet x = \{y | (y, x) \in F\}$, $x^\bullet = \{y | (x, y) \in F\}$, and ${}^\bullet x^\bullet = {}^\bullet x \cup x^\bullet$. In this paper we assume that each transition t of a P/T-net satisfies ${}^\bullet t \neq \emptyset \neq t^\bullet$.

A *marked P/T-net* is a pair $\gamma = (\Sigma, M_0)$, where Σ is a P/T-net and M_0, the *initial marking* of γ, is a marking of Σ. A *labelled marked P/T-net* is a 3-tuple $\gamma = (\Sigma, M_0, l)$, where the first two components form a marked P/T-net and l, the *labelling function* of γ, assigns to each transition either a letter or the empty word λ. In the sequel we shall often use the term "Petri net" whenever we refer to a structure γ as above. In all the above definitions Σ is called the *underlying net* of γ. A marking (place, transition, arc, weight, resp.) of a Petri net γ is the marking (place, transition, arc, weight, resp.) of the underlying net of γ.

Pictorially, a Petri net γ is represented by a graph. The places are denoted by circles and transitions by boxes; the flow relation is represented by arcs. The arc $f \in F$ is labelled by $W(f)$ whenever $W(f) > 1$. The initial marking M_0 is represented by $M_0(s)$ tokens in the circle representing the place s, and the labelling function is denoted by letters in the boxes representing transitions.

Let γ be a Petri net and M a marking of it. The *transition rule* states that a transition t is *enabled* at M, denoted $M[t\rangle_\gamma$, if $M(s) \geq W(s, t)$ for all $s \in S$. If t is enabled at M then t may *occur* yielding a new marking M' given by $M'(s) = M(s) \Leftrightarrow W(s, t) + W(t, s)$, for all $s \in S$; we abbreviate this by $M[t\rangle_\gamma M'$. The transition rule is extended to sequences w of transition in the usual way. If $M_0[w\rangle_\gamma M$ then M is called *reachable*; $[M_0\rangle_\gamma$ denotes the set of all reachable markings (including M_0 itself). The notation "$[\cdot\rangle_\gamma$" will be simplified to "$[\cdot\rangle$" whenever γ is understood from context.

A net $N = (B, E, F)$ is called an *occurrence net* if $|{}^\bullet b| \leq 1$ and $|b^\bullet| \leq 1$ for all $b \in B$, and F^+ is acyclic (i.e. for all $x, y \in B \cup E$ if $(x, y) \in F^+$ then $(y, x) \notin F^+$). Usually the elements of B are called *conditions* whereas the elements of E are called *events*. The *partially ordered set induced* by N is $(B \cup E, \prec_N)$, where $\prec_N = F^+$. The *initial* (*final*) *cut* of N is ${}^\circ N = \{b \in B | {}^\bullet b = \emptyset\}$ ($N^\circ = \{b \in B | b^\bullet = \emptyset\}$). A *$V$-labelled occurrence net* is a couple $\pi = (N, p)$, where N is an occurrence net and p is a total function from $B \cup E$ into an alphabet V. The above definitions (partial order, cut, initial and final cut) are transferred to labelled occurrence nets π; the corresponding notations are obtained by changing "N" into "π" (e.g. $\prec_\pi$, ${}^\circ\pi$, π°). Let $\Sigma = (S, T, F, W)$ be a P/T-net, $\pi = (N, p)$ an $(S \cup T)$-labelled occurrence net such that $p(B) \subseteq S$ and $p(E) \subseteq T$, and C a subset of conditions of π. Define the *marking induced by C in Σ* by $M_C(s) = |p^{-1}(s) \cap C|$, for all $s \in S$.

There are two alternative definitions of a process of a Petri net, axiomatic and inductive, and it is well-known that for finite Petri nets they yields exactly the same objects ([1]). We adopt here the axiomatic definition. A *process* of $\gamma = (\Sigma, M_0)$ is a $(S \cup T)$-labelled occurrence net $\pi = (N, p)$ satisfying:

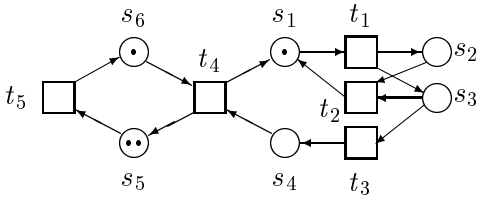
(i) $p(B) \subseteq S$, $p(E) \subseteq T$;
(ii) $M_0(s) = |p^{-1}(s) \cap {}^o N|$ for each $s \in S$;
(iii) $W(s, p(e)) = |p^{-1}(s) \cap {}^\bullet e|$ and $W(p(e), s) = |p^{-1}(s) \cap e^\bullet|$ for each $e \in E$ and $s \in S$.

In order to obtain processes of labelled Petri nets $\gamma = (\Sigma, M_0, l)$ we consider each process $\pi = (N, p)$ of (Σ, M_0) and replace the function p by p', where $p'(x) = p(x)$ for every condition x, and

$p'(x) = (l \circ p)(x)$ for every event x. The set of all processes of a Petri net γ is denoted by $\Pi(\gamma)$.

2 Process Decomposition

We recall an example from [3] concerning a Petri net model $\gamma = (\Sigma, M_0)$ of a vending machine for beverages (Figure 2.1).



$s_1 = ready$ $t_1 = insert$
$s_2 = counter$ $t_2 = reject$
$s_3 = inserted$ $t_3 = accept$
$s_4 = accepted$ $t_4 = dispense$
$s_5 = warm$ $t_5 = brew$
$s_6 = cold$

Figure 2.1

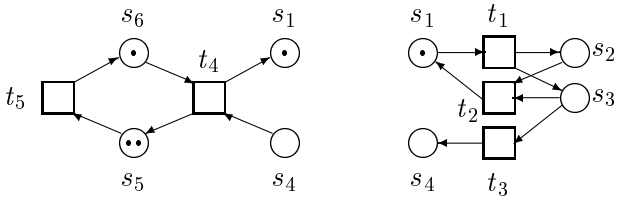


Figure 2.2

Initially, the machine is ready for the insertion of a coin. An inserted coin is checked (counterfeit is rejected). When a coin is accepted, a beverage is dispensed and the control is returned to the state *ready*.

The Petri net model in Figure 2.1 may be viewed as composed by two main parts: the *control part* (concerned with accepting/rejecting coins) and the *dispense part* (concerned with dispensing of a beverage). These two parts are connected by means of the places s_1 and s_4. We may separate them into two Petri nets γ_0 and γ_1 by multiplying the places s_1 and s_4 togheter with their initial markings (Figure 2.2).

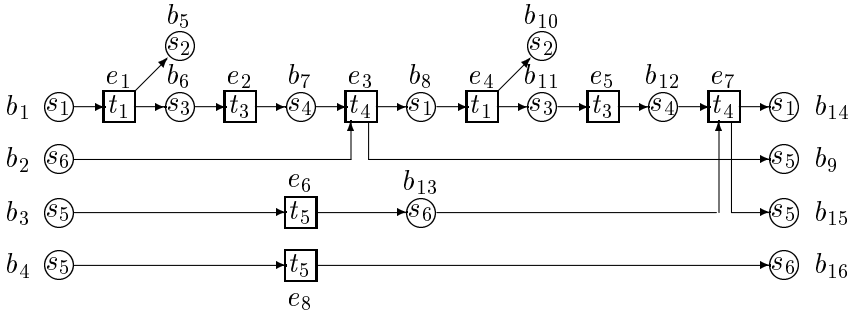


Figure 2.3

Let us consider now the process π of γ pictorially represented in Figure 2.3. This process can be split into two parts (occurrence nets) π_0 and π_1 as in Figure 2.4, according to the decomposition of γ. The initial cut of π_0 generates a marking of γ_0 with two tokens in s_4, which is neither reachable in γ nor in γ_0, and similarly for π_1 with respect to s_1. However, π_0 (π_1) can become a process of γ_0 (γ_1) if we increase the initial marking of γ_0 (γ_1) by two tokens in s_4 (one token in s_1). This fact is not a fortuitous one but it is a particular case of process decomposition as we will see later. We recall first some concepts from [9].

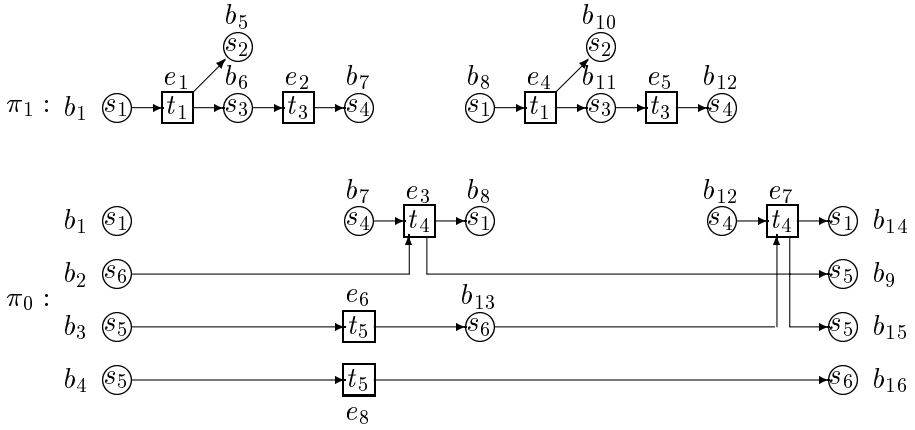


Figure 2.4

The only kind of *subnets* we will consider in this paper are those generated by subset of transitions. That is, if $\Sigma = (S, T, F, W)$ is a net and $T_1 \subseteq T$, then by the *subnet generated by* T_1 we understand the net $\Sigma_1 = (S_1, T_1, F_1, W_1)$, where $S_1 = {}^\bullet T_1^\bullet$ and F_1 and W_1 are the corresponding restrictions of F and W to S_1 and T_1. If Σ_1 is such a subnet, then the subnet generated by $T \Leftrightarrow T_1$ will be called the *difference* of Σ and Σ_1, and it will be denoted by $\Sigma \Leftrightarrow \Sigma_1$. These two concepts can be naturally extended to (labelled) marked Petri nets. For example, the Petri nets γ_0 and γ_1 in Figure 2.2 are subnets of γ in Figure 2.1, generated by $T_0 = \{t_4, t_5\}$ and $T_1 = \{t_1, t_2, t_3\}$, resp.; moreover, $\gamma_0 = \gamma \Leftrightarrow \gamma_1$.

Let S^c be some finite set (of places), and let $PN(S^c)$ be the class of (labelled) marked Petri nets possesing at least the places in S^c. Without loss of generality assume that, for each two Petri nets in $PN(S^c)$, the set S^c contains all common elements (places and transitions) of the nets. Consider further the set $PN(S^c, M_0^c) = \{\gamma \in PN(S^c) | M_0|_{S^c} = M_0^c\}$, where $M_0^c \in \mathbf{N}^{S^c}$.

For any two Petri nets γ_1 and γ_2, either both in $PN(S^c)$ or both in $PN(S^c, M_0^c)$, their (componentwise defined) union will be called the *catenation (along* S^c*)* of γ_1 and γ_2, and will be denoted by $\gamma_1 \circ \gamma_2$; the set S^c will be called the *set of connecting places*. In the case $S^c = \emptyset$, the Petri net $\gamma_1 \circ \gamma_2$ consists of two disjoint subnets, γ_1 and γ_2. The net γ in Figure 2.1 is the catenation along $\{s_1, s_4\}$ of γ_0 and γ_1 in Figure 2.2.

The definition of catenation of Petri nets can be naturally extended to labelled occurrence nets by requiring supplementary $p_1(s) = p_2(s)$ for all $s \in S^c$ (p_1 and p_2 are the corresponding labelling functions).

Before stating the main result of this section we will adopt one more notation. For a Petri net $\gamma = (\Sigma, M_0, l) \in PN(S^c, M_0^c)$ and a marking $M \in \mathbf{N}^{S^c}$, we denote by $(\gamma + M)$ the Petri net $(\Sigma, M_0 + M, l) \in PN(S^c, M_0^c + M)$.

Theorem 2.1 (Process decomposition theorem)
Let $\gamma_0, \gamma_1 \in PN(S^c, M_0^c)$. *For any process* $\pi \in \Pi(\gamma_0 \circ \gamma_1)$ *there are two markings* $M', M'' \in \mathbf{N}^{S^c}$ *and two processes* $\pi_0 \in \Pi(\gamma_0 + M')$ *and* $\pi_1 \in \Pi(\gamma_1 + M'')$ *such that* $\pi = \pi_0 \circ \pi_1$ *(where this composition of processes is along some suitable set of common conditions).*

The process π_0 (π_1) in the theorem above will be called a *sample* of π w.r.t. γ_0 (γ_1). The set of all process samples of γ w.r.t. γ_0 will be denoted by $\Pi(\gamma, \gamma_0)$. The processes π_0 and π_1 in Figure 2.4 are processes of $(\gamma_0 + (0, 2))$ and $(\gamma_1 + (1, 0))$, respectively (the order on connecting places is s_1, s_4). Therefore, π_0 is a sample of π w.r.t. γ_0, and π_1 is a sample of π w.r.t. γ_1.

3 Jumping Nets – A Tool for Process Sample Generation

Given a Petri net $\gamma = \gamma_0 \circ \gamma_1$, where $\gamma_0, \gamma_1 \in PN(S^c, M_0^c)$, we may ask how to generate all the process samples of γ w.r.t. γ_0. Of course, we may generate all the processes of γ and then split these processes as in the proof of Theorem 2.1 (top-down). We shall describe in the sequel an alternative method based on *jumping nets* which allows a bottom-up construction of processes. First, let us look again to the nets in Figure 2.4. From the π_0's point of view (in the context of γ), the conditions b_7 and b_{12} have been "pumped" by the environment (by π_1). However, for these two conditions, γ_0 has to pay with other two conditions, b_1 and b_8 (labelled by s_1). More precisely, γ_0 gives a condition labelled by s_1 to γ_1 and receives, not necessary immediately, a condition labelled by s_4. This exchange of conditions can be formally described by the binary relation

$$R_0 = \{((1,0),(0,0)),((0,0),(1,0)),((0,0),(0,1)),((1,0),(0,1))\}$$

on $\mathbf{N}^{S^c}$. For example, the pair $((1,0),(0,1)) \in R_0$ states that whenever γ_0 has produced a condition labelled by s_1 then it can exchange it with a condition labelled by s_4. A computation step associated to the couple (γ_0, R_0) is an extension of the usual occurrence rule of γ_0 by:

$$M\, R_0\, M' \quad \Leftrightarrow \quad M(s_1) = 1,\ M(s_4) = 0,\ M'(s_1) = 0,\ M'(s_4) = 1$$

for all markings M and M'. For example, the following is a sequence of computation steps in (γ_0, R_0):

$$(1,0,2,1)R_0(0,1,2,1)[t_4\rangle(1,0,3,0)[t_5\rangle(1,0,2,1)R_0(0,1,2,1)[t_4\rangle(1,0,3,0)[t_5\rangle(1,0,2,1).$$

We shall prove that the set of all process samples of γ w.r.t. γ_0 is isomorphic to the set of processes of (γ_0, R_0), which are to be defined. The reader can discuss in a similar way the case of γ_1 endowed with the relation $R_1 = \{((0,1),(1,0))\}$.

Now, we will introduce the concept of a jumping Petri net in a slightly different way than in [7] and [8].

Definition 3.1 *A* jumping Petri net *(*marked jumping Petri net, labelled marked jumping Petri net*) is a couple $\mathcal{J} = (\gamma, R)$, where γ is a P/T-net (marked P/T-net, labelled marked P/T-net) and R is a finite union of sets, each of which being a binary relation on $\mathbf{N}^{S'}$ for some $S' \subseteq S$.*

The elements of R are called *jumps* of $\mathcal{J}$. A jump (M, M'), where $M, M' \in \mathbf{N}^{S'}$ and $S' \subseteq S$, is called *local on* S'. For technical reasons is necessary to extend jumps to the set S of places of γ, as follows:

$$M\, R\, M' \quad \Leftrightarrow \quad M|_{S'}\, R\, M'|_{S'} \text{ and } M|_{S-S'} = M'|_{S-S'},$$

for all markings $M, M' \in \mathbf{N}^S$, where $S' \subseteq S$ and $(M|_{S'}, M'|_{S'})$ is a local jump on S'.

A computation step in a jumping Petri net $\mathcal{J} = (\gamma, R)$ is performed either by a transition, in the usual way, or by a jump. That is,

$$M[x\rangle M' \quad \Leftrightarrow \quad \text{either } x \in T \text{ and } M[x\rangle_\gamma M', \text{ or } x \in R \text{ and } M\, R\, M'.$$

Local jumps which do not affect all places of a Petri net can occur concurrently with each other or concurrently with transition occurrences. This is formally reflected in the following definition of processes of jumping Petri nets (for convenience we will adopt an inductive definition).

Let $\mathcal{J} = (\gamma, R)$ be a marked jumping Petri net, t a transition and $r = (M, M')$ a local jump on a subset $S' \subseteq S$. Then:

- an *elementary occurrence net* associated to t is a labelled occurrence net $\pi = (N, p)$ with the properties: π contains only one event e labelled by t, $W(s,t)$ preconditions and $W(t,s)$ postconditions of e labelled by s, for all $s \in S$, and no other element;
- an *elementary occurrence net* associated to r is a labelled occurrence net $\pi = (N, p)$ with the properties: π contains only one event e labelled by r, $M(s)$ preconditions and $M'(s)$ postconditions of e labelled by s, for all $s \in S'$, and no other element. Pictorially, the event e will be drawn by a double box;
- an *initial occurrence net* of γ is an occurrence net (N, p) which does not contain any event and, for each $s \in S$, it contains exactly $M_0(s)$ conditions labelled by s.

Definition 3.2 *Let $\mathcal{J} = (\gamma, R)$ be a marked jumping Petri net. The* set of processes *of $\mathcal{J}$, denoted by $\Pi(\mathcal{J})$, is the smallest set with the properties:*

(1) $\Pi(\mathcal{J})$ contains all the initial occurrence nets associated to $\mathcal{J}$;

(2) if $\pi_1 \in \Pi(\mathcal{J})$ and π_2 is an elementary occurrence net associated to a transition t such that ${}^\circ\pi_2 \subseteq \pi_1^\circ$, then the catenation along ${}^\circ\pi_2$ of π_1 and π_2, whenever it is possible, is in $\Pi(\mathcal{J})$;

(3) if $\pi_1 \in \Pi(\mathcal{J})$ and π_2 is an elementary occurrence net associated to a local jump $r = (M, M')$ on a subset S' of places such that $|\pi_1^\circ \cap p^{-1}(s)| = M(s)$ for all $s \in S'$, then the catenation along ${}^\circ\pi_2$ of π_1 and π_2, whenever it is possible, is in $\Pi(\mathcal{J})$.

In cases (2) and (3) we say that π_1 is extended, *by catenation, by π_2.*

It is clear that for any such π there is at least a sequence $\pi_0, \pi_1, \ldots, \pi_m = \pi$, where π_0 is an initial occurrence net and π_{i+1} may be constructed from π_i as described in Definition 3.2, for all $0 \leq i \leq m \Leftrightarrow 1$. Processes of labelled jumping Petri nets are obtained as for labelled Petri nets.

Example 3.1 *Let $\mathcal{J}_0 = (\gamma_0, R_0)$ and $\mathcal{J}_1 = (\gamma_1, R_1)$, where γ_0 and γ_1 are the nets in Figure 2.2, and R_0 and R_1 are the relations given in the beginning of this section. Then, π_0 (π_1) in Figure 3.1 is a process of $\mathcal{J}_0$ ($\mathcal{J}_1$), where $r_0 = ((1,0),(0,1))$ ($r_1 = ((0,1)(1,0))$).*

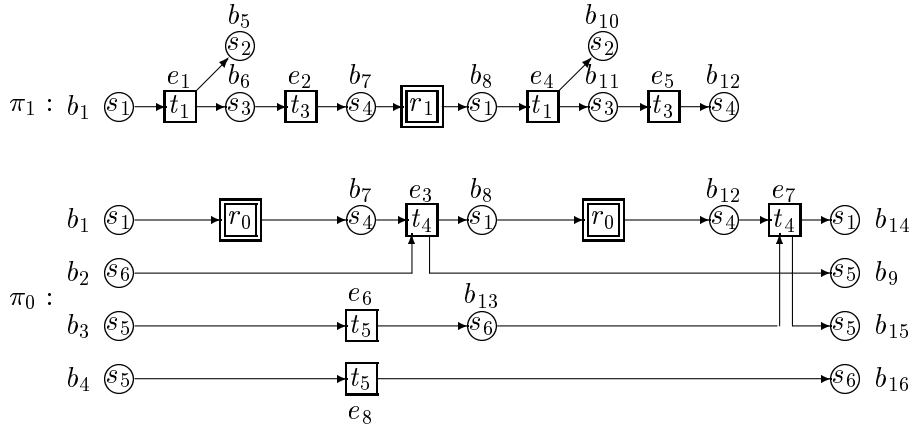


Figure 3.1

Now let us move our attention to the generation of process samples by jumping nets. First let us note that a subnet γ_1 of a Petri net γ may *induce* some jumps for the net $\gamma \Leftrightarrow \gamma_1$. Moreover, it may induce the same jump at different markings of γ. If the jumps induced by γ_1 do not depend on the internal configuration of γ_1, but just on the marking of the connecting places, then γ is called *γ_1-context free.*

Definition 3.3 *Let $\gamma_0, \gamma_1 \in PN(S^c, M_0^c)$, $\gamma = \gamma_0 \circ \gamma_1$, and let M be a reachable marking of γ.*

(1) A jump $(M^c, \overline{M}^c)$ on S^c is induced by γ_1 at M in γ *if there is a marking $\overline{M} \in [M_0\rangle_\gamma$ such that:*

(i) $M|_{S^c} = M^c$ and $\overline{M}|_{S^c} = \overline{M}^c$;

(ii) $\overline{M}$ is reachable from M only by occurrences of transitions in γ_1, but at least by one occurrence.

(2) γ is called γ_1-context free *if for any jump $(M^c, \overline{M}^c)$ induced by γ_1 and for any reachable marking M in γ, if $M|_{S^c} = M^c$ then γ_1 induces $(M^c, \overline{M}^c)$ at M.*

The jumps induced by the Petri net γ_1 in Figure 2.2 are those from the set R_0, and the only jump induced by γ_0 is that from R_1. The Petri net γ in Figure 2.1 is both γ_0- and γ_1-context free.

Let $\mathcal{J} = (\gamma, R)$ be a jumping Petri net. From each process π of $\mathcal{J}$ we define a new occurrence net by removing all the events labelled by jumps (and the corresponding arcs). Let $\Pi'(\mathcal{J})$ be the

set of all these occurrence nets. The occurrence nets in Figure 2.4 are obtained, as described above, from the processes in Figure 3.1.

Theorem 3.1 (Process sample generation theorem)
Let $\gamma_0, \gamma_1 \in PN(S^c, M_0^c)$ and $\gamma = \gamma_0 \circ \gamma_1$. If γ is γ_1-context free then $\Pi(\gamma, \gamma_0) = \Pi'(\mathcal{J})$, where $\mathcal{J} = (\gamma_0, R_0)$ and R_0 is the set of jumps induced by γ_1 in γ.

Theorem 3.1 gives us a specification tool for all the process samples of a Petri net γ w.r.t. a subnet γ_1 in the case that γ is γ_1-context free.

4 Jumping Petri Nets Versus Petri Nets

It is clear that any Petri net can be viewed as a jumping Petri net by taking the empty set as the set of jumps. Consequently, processes of Petri nets are particular cases of processes of jumping Petri nets. In [7] it has been proved that jumping Petri nets are strictly more powerful, from the interleaving semantics point of view, than Petri nets even if finite sets of jumps are considered. Clearly, this result holds true for the case of process semantics as well. In some particular cases, as we shall see below, jumping Petri nets can be simulated by Petri nets.

Definition 4.1 *Let S be a non-empty set and R be a finite union of sets, each of which being a binary relation on $\subseteq \mathbf{N}^{S'} \times \mathbf{N}^{S'}$ for some set $S' \subseteq S$. We say that R is* Δ-finite *if there is a finite set V of vectors with integer components such that for any $(M, M') \in R$ we have $M' \Leftrightarrow M \in V$.*

Remark 4.1 *All jumping Petri nets obtained by decomposing bounded Petri nets (that is, Petri nets with a finite set of reachable markings) are Δ-finite.*

Definition 4.2 *Let $\mathcal{J}$ be a jumping Petri net. Its set of jumps R is said to be* complete *if for any reachable marking M of γ and for any jump $(M_1, M_2) \in R$, if (M_1, M_2) is local on $S' \subseteq S$ and $M|_{S'} \geq M_1$ then there is a marking $M' \in \mathbf{N}^{S'}$ such that $(M|_{S'}, M') \in R$ and $M' \Leftrightarrow M|_{S'} = M_2 \Leftrightarrow M_1$.*

In order to compare processes of jumping Petri nets with processes of Petri nets we need the concept of (j, λ)-*isomorphism*. A process $\pi_1 = (N_1, p_1)$ of a labelled jumping Petri net is (j, λ)-*isomorphic* to a process $\pi_2 = (N_2, p_2)$ of a labelled Petri net if there is a bijection $\varphi : B_1 \cup E_1 \to B_2 \cup E_2$ such that:

(1) $p_1(x) = p_2(\varphi(x))$ for all $x \in B_1$ and for all $x \in E_1$ with the property that $p_1(x)$ is not a jump; if $p_1(x)$ is a jump then $p_2(\varphi(x))$ is λ;
(2) $x \prec_{\pi_1} y$ iff $\varphi(x) \prec_{\pi_2} \varphi(y)$ for all $x, y \in B_1 \cup E_1$.

This notion of isomorphism of processes is a slight modification of the classical one: it is an isomorphism of occurrence nets preserving all the condition-labels and all the non-jump event-labels; the events labelled by jumps are mapped into events labelled by λ (this justifies our terminology of (j, λ)-isomorphism).

Theorem 4.1 *Let $\mathcal{J} = (\gamma, R)$ be a labelled marked jumping Petri net. If R is Δ-finite and complete then there is a labelled marked Petri net γ' such that $\Pi(\mathcal{J})$ and $\Pi(\gamma')$ are (j, λ)-isomorphic.*

5 Application of Process Decomposition to Validation

One aim of simulation is to *validate* the model w.r.t. some desired behavioural properties (that is, to check whether the desired properties are reflected or not in the simulated runs of the model). As we could expect, many problems are encountered when dealing with simulation of a Petri net model: fairness, alternatives (solving conflicts), termination conditions, visualization and property checking

for large processes, etc. All these problems could be grouped into two main classes: *generation* and *analysis* of processes (for a detailed discussion the reader is referred to [3]). Therefore, it turns out to be an important task to look for an adequate tool to represent and generate processes as well as for an efficient strategy for analyzing them. In this section we will show how the mechanism developed in the last two sections can be used for validation of Petri net models.

Suppose that a net γ can be decomposed as follows:

$$\begin{array}{lcll} \gamma & = & \gamma_0 \circ \gamma_0' & (S^{c,0}) \\ \gamma_0' & = & \gamma_1 \circ \gamma_1' & (S^{c,1}) \\ \cdots & & & \\ \gamma_{n-2}' & = & \gamma_{n-1} \circ \gamma_n & (S^{c,n-1}) \end{array}$$

(the sets of connecting places in brackets). Formally, we may write

$$\gamma = \gamma_0 \circ (\gamma_1 \circ \cdots \circ (\gamma_{n-1} \circ \gamma_n) \cdots).$$

We suppose that the net γ_n is of reazonable small size such that it supports an ad hoc validation. Then, we can define the jumping net $\mathcal{J}_{n-1}$ in order to generate process samples and validate γ_{n-1} in the context of γ_n. If this step is successfully performed then we may consider valid the net $(\gamma_{n-1} \circ \gamma_n)$ and continue validation. The main problem we encounter when dealing with the construction of jumping Petri nets (as above) is to find a convenient way to describe the set of jumps. In fact, this problem has two main aspects:

1. decide whether is it possibe to define a jumping Petri net $\mathcal{J}_i$ (as above);
2. if the answer to the question above is positive, then find a convenient way to describe the set of jumps.

For the first question, the only good result we have is that from Theorem 3.1. The answer to the second question seems to be more complicated and we do not have yet any answer.

However, in practice it could be not necessary to construct *a priori* the jumping Petri nets $\mathcal{J}_i$. We may take γ_i and generate processes of it until a jump is necessary. Then, we take the current marking of connecting places and, togheter with the current marking of the internal places of γ_i' we generate a jump for γ_i (and save the current marking on the internal places of γ_i').

References

[1] E. Best, C. Fernandez: *Nonsequential Processes. A Petri Net Point of View*, EATCS Monographs on Theoretical Computer Science, Springer-Verlag, 1988.

[2] J. Desel, W. Reisig: *Place Transition Systems*, in: *Lectures on Petri Nets I: Basic Models*, Lecture Notes in Computer Science 1491, 1998, 122–173.

[3] J. Desel: *Validation of System Models Using Partially Ordered Runs*, to appear in: *Business Process Management-Models, Techniques and Empirical Studies* (W.M.P. van der Aalst, J. Desel, A. Oberweis, eds.), Lecture Notes in Computer Science, 1999.

[4] E. Kindler: *A Compositional Partial Order Semantics for Petri Net Components*, in: *Proc. of the 18th International Conference on Application and Theory of Petri Nets* (P. Azema, G. Balbo, eds.), Lecture Notes in Computer Science 1248, 1997, 235–252.

[5] W. Reisig: *Petri Nets. An Introduction*, EATCS Monographs on Theoretical Computer Science, Springer-Verlag, 1985.

[6] W. Reisig: *Elements of Distributed Algorithms. Modeling and Analysis with Petri Nets*, Springer-Verlag, 1998.

[7] F.L. Ţiplea, T. Jucan: *Jumping Petri Nets*, Foundations of Computing and Decision Sciences 19, 1994, 319–332.

[8] F.L. Ţiplea, E. Mäkinen: *Jumping Petri Nets. Specific Properties*, Fundamenta Informaticae 32, 1997, 373–392.

[9] F.L. Ţiplea, J. Desel: *Proving Correctness of Petri Net Transformations by Replacement Techniques*, submitted.