**Theoretical Informatics and Applications**
Theoret. Informatics Appl. **37** (2003) 17–38
DOI: 10.1051/ita:2003006

# ON THE ANALYSIS OF PETRI NETS AND THEIR SYNTHESIS FROM PROCESS LANGUAGES

Ludwik Czaja[1]

**Abstract.** Processes in Place/Transition (P/T) nets are defined inductively by a peculiar numbering of place occurrences. Along with an associative sequential composition called catenation and a neutral process, a monoid of processes is obtained. The power algebra of this monoid contains all process languages with appropriate operations on them. Hence the problems of analysis and synthesis, analogous to those in the formal languages and automata theory, arise. Here, the analysis problem is: for a given P/T net with an initial marking find the set of all processes the net may evoke. The synthesis problem is: given a process language $L$ decide if there exists a marked net whose evolutions (represented by processes) are collected in $L$ and, in the positive case, find such net and its initial marking. The problems are posed and given a general solution.

**Mathematics Subject Classification.** 68Q85.

## 1. Introduction with a notation of nets

Processes in a Petri net represent its possible evolutions. Intuitively, such a "trace" of evolution stems from recording any transition at a time of its firing, along with its neighbourhood, *i.e.* entry and exit arrows and places. In this way a process is a structure of occurrences of net constituents. Thus, a net again, though of a simplified structure: acyclic and, customarily, with no place-branching. In the process construct proposed here, the latter restriction is partly abandoned, by permitting forward but no backward place-branches. This is an effect of a peculiar numbering of place occurrences, that is their identification. Any arrow

*Keywords and phrases.* Petri net, process language, analysis and synthesis of nets.

[1] Institute of Informatics, Warsaw University, ul. Banacha 2, 02-097 Warsaw, Poland; e-mail: lczaja@mimuw.edu.pl

leaving (entering) a place in a process represents removal (insertion) of a token from (in) respective place in the underlying net. Many works on processes in nets concern elementary (1-safe) nets, *e.g.* [5, 20, 26, 29, 30, 32] to mention a few. In some papers, *e.g.* [1, 3, 12, 15–17, 24, 31], we find various definitions of process for Place/Transition nets, that is nets with finite or infinite place capacity. In all of these definitions a process is constructed for a *given* net. The approach admitted here is different: processes are thought to constitute a semantic domain common to *all* P/T nets. Their construction proceeds independently of any underlying net, much in the way words are built from letters by juxtaposition, independently of any generative device like automaton, grammar or fix-point equations (a similar, in this respect, but more abstract construction of process is in [33]). The role of a letter plays here atomic process, which is the earliest occurrence of a transition with its pre and post places. The role of juxtaposition plays sequential composition of processes, called catenation. In this way, our process is a sort of an occurrence net rather than a process in the standard net theory. When extended to maximality, it corresponds to the notion of unfolding in that theory.

The approach allows for collecting diverse processes into sets – process languages. Some of such collections, sometimes even single processes as that in Figure 4.1, do not enjoy the existence of generative nets, while others do. Hence the synthesis problem: given a process language $L$ decide if there exists a marked net whose evolutions are collected in $L$ and, in the positive case, find such net and its initial marking. The problem is given a general solution in Section 3 as well as the analysis problem: for a given marked net find the set of all its evolutions. Note that the term "analysis" is used in the sense analogous to the one encountered in automata theory rather than in Petri nets theory, where it refers to proving properties. As usually, the motivation for analysis is to provide a formal framework for discovering properties of dynamic systems by examination of objects they generate, process languages evoked by nets in particular. Many works are concerned with this problem [14,22,23,25,27] are but a few examples. A computer aided tool RELVIEW [2] transforms a given net into an algebraic relation (thus, performs a sort of analysis), then derives some static and dynamic properties of the net from this relation. On the other hand, the synthesis provides a mechanism for construction of systems starting from atomic processes and a specification of their permissible (partial) ordering by any measure of time. Here, such specification is expressed by process languages. It should be noticed that there are a number of other kinds of net synthesis, for instance, from incomplete specification given by (pairs of) regular languages and from deterministic stack automata [11], from transition systems [4,14,28] from the so-called prexes (predicate expressions) [19], etc. In fact, analysis and synthesis consist in moving between descriptive formalisms.

In this paper we limit our considerations to Place/Transition nets with unbounded capacity of places and arrow weights equal 1 and to moving between nets and process languages. Extension of this approach to fully general P/T nets turns out straightforward and has been shown in [18] as an exercise. This is achieved by defining transition $t$ as a pair of multisets ${}^\bullet t, t^\bullet : \mathbb{X} \to \mathbb{N}$ (instead of sets) with $\mathbb{X}$ – a universe of places and ${}^\bullet t(x)$ and $t^\bullet(x)$ interpreted as weights of arrows

$x \to t$ and $t \to x$ respectively. Then, accordingly redefining the concept of process: atomic processes involve, in this case, weighted arrows. Catenation of processes remains as it is here. For simplicity, we limit the considerations to the usage of the elementary calculus of sets instead of multisets. In general case, some results must be slightly redefined, *e.g.* the analysis remains correct for the pure transitions only (with no self-loops). The approach is motivated by simplicity of the proposed solution to the analysis and synthesis tasks, due to the non-standard construction of process. It is not difficult to find examples of process languages having no generating nets. A trivial example is any set of processes which is not prefix-closed. Also, the set of processes in Figures 2.2–2.4 with all their prefixes is not net-generable. A not so trivial example is the set of all finite prefixes of the infinite process in Figure 4.1. Specific examples of non-net-generable process languages may be provided by the aid of Theorem 3.3.

Further in this Introduction a notation of nets and its graphical presentation is outlined. Section 2 exposes definition of finite process, catenation of processes, discusses their peculiarities and provides some properties of these concepts. Section 3 is the main one: its subject is nets with initial marking and analysis/synthesis problems. Section 4 contains hints toward extension of the former results to infinite processes and Section 5 – some remarks, possibly for further investigations.

## Nets

Let $\mathbb{X}$ be a non-empty set (universe of *net-places* for all nets). A *transition* over $\mathbb{X}$ is a pair $t = ({}^\bullet t, t^\bullet)$ with the sets ${}^\bullet t, t^\bullet \subseteq \mathbb{X}$, ${}^\bullet t \neq \emptyset \neq t^\bullet$, called, respectively, a *pre-set* and *post-set* of $t$. Transition $t$ is *pure* iff ${}^\bullet t \cap t^\bullet = \emptyset$, which means it does not involve any loop. By $\mathbb{T}$ the universe of all transitions is denoted, the set of all finite (infinite) sequences of transitions, empty sequence $\lambda$ including, is $\mathbb{T}^*$ ($\mathbb{T}^\omega$). Let $\mathbb{T}^\infty = \mathbb{T}^* \cup \mathbb{T}^\omega$. *Unmarked* nets are subsets of $\mathbb{T}$, denoted T, possibly with indices. A *marking* is a function $M$: $\mathbb{X} \to \mathbb{N}$ where $\mathbb{N}$ is the set of all natural numbers, 0 including. Markings will be treated as multisets over $\mathbb{X}$ and usual operations +, - and comparison $\leq$ applied component-wise to them. The set $\mathbb{N}^{\mathbb{X}}$ of all markings is denoted by $\mathbb{M}$. A *marked net* is a pair $\mathcal{N} = (T, M)$ with $T \subseteq \mathbb{T}$, $M \in \mathbb{M}$. Semantics of a transition $t$ is a binary relation $[[t]] \subseteq \mathbb{M} \times \mathbb{M}$ defined by: $(M, M') \in [[t]]$ iff ${}^\bullet t \leq M \wedge M' = M - {}^\bullet t + t^\bullet$. Semantics of a net $T \subseteq \mathbb{T}$ is a binary relation $[[T]] = \bigcup\limits_{t \in T} [[t]]$. Additionally $[[\emptyset]] = \emptyset$. The reflexive and transitive closure $[[T]]^*$ is a *reachability* relation in $\mathbb{M}$. A sequence of transitions $U = t_1 t_2 t_3 ... \in \mathbb{T}^\infty$ is a *firing sequence* if there are markings $M_0, M_1, M_2, ...$ such that $(M_{j-1}, M_j) \in [[t_j]]$for $j = 1, 2, 3, ...$ . Directly from these definitions we get:

**Proposition 1.1.** $[[T_1 \cup T_2]] = [[T_1]] \cup [[T_2]]$, $[[T_1 \cap T_2]] = [[T_1]] \cap [[T_2]]$, *for any unmarked nets* $T_1, T_2 \subseteq \mathbb{T}$.

A graphical representation of transitions and nets explains the following example. Transitions $t = (\{b\}, \{a\})$, $u = (\{a\}, \{b, e\})$, $v = (\{d, e\}, \{c\})$, $w = (\{c\}, \{d\})$, are drawn in Figure 1.1.

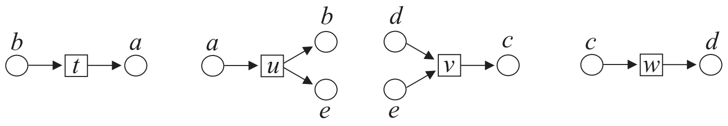


Figure 1.1

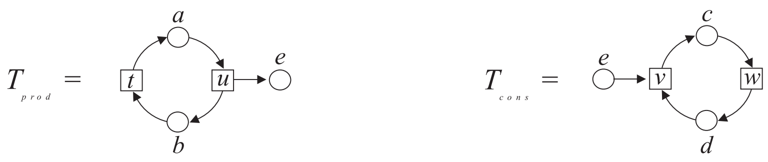


Figure 1.2

Nets $T_{\text{prod}} = \{t, u\}$, $T_{\text{cons}} = \{v, w\}$, representing a system of producers and consumers respectively, are drawn in Figure 1.2.

The set of transitions $T_{\text{prodcons}} = T_{\text{prod}} \cup T_{\text{cons}} = \{t, u, v, w\}$ representing a producers/consumers system with unbounded buffer $e$ is drawn as the net in Figure 1.3.

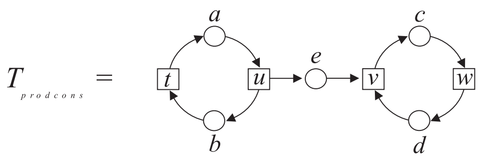


Figure 1.3

## 2. Monoid of finite processes

**Definition 2.1** (Atomic process. Set $PR_0$)**.** Let $\mathbb{X}$ be a set (net-places). An *atomic process* is a singleton (one-element) set $\varepsilon = \{(A \times \{0\}, B \times \{1\})\}$, where $A, B \subseteq \mathbb{X}$, $A \neq \emptyset \neq B$. Sets $A \times \{0\}, B \times \{1\}$ are called a *pre-set* and *post-set* of $\varepsilon$ respectively. Denote ${}^\bullet\varepsilon = A$, $\varepsilon^\bullet = B$, $Y_\varepsilon = A \times \{0\} \cup B \times \{1\}$ and define a function $I_\varepsilon : \mathbb{X} \to \{0, 1\}$ by:

$$I_\varepsilon(x) = \begin{cases} 1 & \text{if} \quad x \in \varepsilon^\bullet \\ 0 & \text{if} \quad x \notin \varepsilon^\bullet. \end{cases}$$

Say that $\varepsilon$ corresponds to transition $t = (A, B) \in \mathbb{T}$ and, to stress this, write $\varepsilon_t$. The set of all atomic processes, atoms in short, is denoted $PR_0$.

**Example 2.1.**

$$\begin{aligned}
\varepsilon_t &= \{(\{\langle b,0\rangle\},\{\langle a,1\rangle\})\}\\
\varepsilon_u &= \{(\{\langle a,0\rangle\},\{\langle b,1\rangle,\langle e,1\rangle\})\}\\
\varepsilon_v &= \{(\{\langle d,0\rangle,\langle e,0\rangle\},\{\langle c,1\rangle\})\}\\
\varepsilon_w &= \{(\{\langle c,0\rangle\},\{\langle d,1\rangle\})\}.
\end{aligned}$$

Pictorially, these atoms, corresponding to transitions *t, u, v, w* from Figure 1.1, are represented in Figure 2.1.

FIGURE 2.1

**Definition 2.2** (Finite process. Set $PR_{\text{fin}}$).

1. Atomic process $\varepsilon$ and the empty set $\emptyset$ are finite processes. Assume $Y_\emptyset = \emptyset$ and $I_\emptyset(x) = 0$ for each $x \in \mathbb{X}$.
2. Let $\alpha$ be a finite process already defined and $\varepsilon$ an atomic process. Thus, the set $Y_\alpha$ and the function $I_\alpha$: $\mathbb{X} \to \mathbb{N}$ are defined. Then, the $\varepsilon$-successor of $\alpha$ defined as $\text{suc}_\varepsilon(\alpha) = \alpha \cup \varepsilon^\alpha$ is a finite process, where $\varepsilon^\alpha$ is $\varepsilon$ with each pair $\langle x,k\rangle \in Y_\varepsilon$ ($k \in \{0,1\}$) replaced by $\langle x, k + I_\alpha(x)\rangle$ (say: $\varepsilon^\alpha$ is a *shift* of $\varepsilon$ by $\alpha$). Obviously, $\text{suc}_\varepsilon(\emptyset) = \varepsilon$. Define:

$$\alpha \otimes \varepsilon = \text{suc}_\varepsilon(\alpha)$$

$Y_{\alpha\otimes\varepsilon} = Y_\alpha \cup Y_{\varepsilon^\alpha}$ where $Y_{\varepsilon^\alpha}$ is $Y_\varepsilon$ with each pair $\langle x,k\rangle \in Y_\varepsilon$ ($k \in \{0,1\}$) replaced by $\langle x, k + I_\alpha(x)\rangle$

$$I_{\alpha\otimes\varepsilon}(x) = I_\alpha(x) + I_\varepsilon(x)$$

$Y_\alpha$ is the set of *process-places* in $\alpha$. If $\langle x,k\rangle \in Y_\alpha$ then the process-place $\langle x,k\rangle$ is an *occurrence* of the net-place $x$ in $\alpha$ and $k$ is an *occurrence number* of $x$. $I_\alpha(x)$ is the greatest occurrence number of $x$ in $\alpha$, that is, $I_\alpha(x) = \sup\{k | \langle x,k\rangle \in Y_\alpha\}$. The set of all finite processes is denoted $PR_{\text{fin}}$. Therefore, $PR_{\text{fin}}$ is the least set with $PR_0 \cup \{\emptyset\} \subseteq PR_{\text{fin}}$ and closed under all the $\varepsilon$-successors.

**Remarks.** (1) This definition allows to unconditionally compute successors of any finite process. This corresponds to the situation that processes from $PR_{\text{fin}}$ represent evolutions of nets with unbounded place-capacity and infinite number of tokens in each place. Thus, the universe of processes $PR_{\text{fin}}$ plays similar part as a set of all (finite) words over a given alphabet in the theory of formal languages. Here, $PR_0$ plays, obviously, part of the alphabet. This is not so with processes generated by marked nets and with infinite processes – which will be considered later.

(2) The computational features of the process construct are explained later, following a number of examples and simple formal properties.

(3) A different but equivalent definition of process is in [10]. It encompasses both finite and infinite processes.

Examples of three finite processes composed of atoms from Figure 2.1 are in Figure 2.2, Figure 2.3, Figure 2.4; obviously, these processes represent three different evolutions of the net in Figure 1.3.

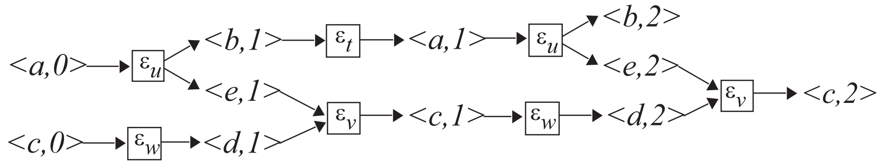


FIGURE 2.2. This process requires initially at least one token in places $a$ and $c$.

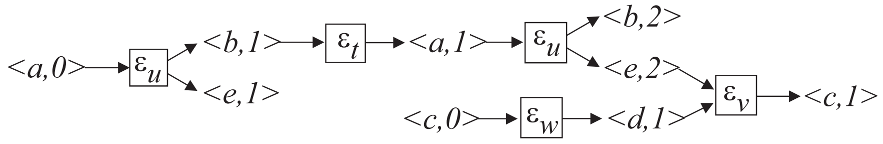


FIGURE 2.3. This process requires initially at least one token in places $a$ and $c$.

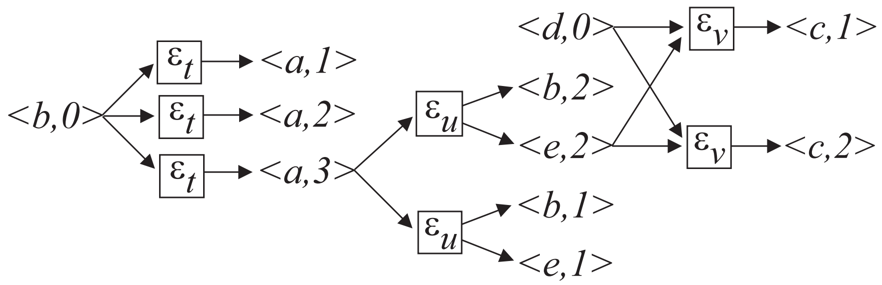


FIGURE 2.4. This process requires initially at least three tokens in place $b$ and two in $d$.

**Definition 2.3** (Catenation $\otimes$ of finite processes. Neutral process **1**)**.** Let $\alpha, \beta \in PR_{\text{fin}}$ and $\varepsilon \in PR_0$. Catenation $\otimes$ is defined inductively by:

$$\begin{aligned}
&\alpha \otimes \emptyset = \alpha \\
&\alpha \otimes \varepsilon = \text{suc}_\varepsilon(\alpha) \\
&\alpha \otimes \text{suc}_\varepsilon(\beta) = \text{suc}_\varepsilon(\alpha \otimes \beta).
\end{aligned}$$

Since $\emptyset \otimes \alpha = \alpha$, the empty set is neutral for $\otimes$. Thus, we use denotation **1** for $\emptyset$.

**Remarks.** Symbol $\otimes$ has been used for catenation in a number of papers (on similar concept of process for various kind of nets) preceding the present one, *e.g.* [8–10] among others. In [8,9], where processes (in 1-safe nets) with a norm have been considered, the symbol $\odot$ is used for a certain redefined sort of catenation.

**Proposition 2.1.** *Catenation of finite processes is associative:* $\alpha \otimes (\beta \otimes \gamma) = (\alpha \otimes \beta) \otimes \gamma$.

For the proof, valid for finite and infinite processes, see Theorem 4.1.

**Corollary 2.1.** $\mathcal{M}_{\text{fin}} = (PR_{\text{fin}}, \otimes, \mathbf{1})$ *is a monoid.*

**Definition 2.4** (**Shift** $\beta^\alpha$)**.** For processes $\beta, \alpha$ let $\beta^\alpha$ be $\beta$ with each pair $\langle x, k\rangle \in Y_\beta$ replaced by $\langle x, k{+}I_\alpha(x)\rangle$. That is, $\beta^\alpha$ is $\beta$ with net-place occurrence numbers incremented or *shifted* by values of $I_\alpha$.

Definitions 2.2–2.4 and Proposition 2.1 directly imply:

**Proposition 2.2.** *Any finite process is either* **1** *or a catenation of a number of atomic processes and:*

(a) *if* $m \le n$ *then* $\varepsilon_1 \otimes \varepsilon_2 \otimes \ldots \otimes \varepsilon_m \subseteq \varepsilon_1 \otimes \varepsilon_2 \otimes \ldots \otimes \varepsilon_n$ *for any atomic processes* $\varepsilon_j$*;*

(b) $\bigcup_{j=1}^{n} \varepsilon_1 \otimes \varepsilon_2 \otimes \ldots \otimes \varepsilon_j = \varepsilon_1 \otimes \varepsilon_2 \otimes \ldots \otimes \varepsilon_n$*;*

(c) $\alpha \otimes \beta = \alpha \cup \beta^\alpha$ *for any* $\alpha, \beta \in PR_{\text{fin}}$.

Note that if $\varepsilon_t, \varepsilon_u, \varepsilon_v, \varepsilon_w$ are atoms in Figure 2.1 then the process in Figure 2.2 is: $\varepsilon_u \otimes \varepsilon_w \otimes \varepsilon_t \otimes \varepsilon_v \otimes \varepsilon_u \otimes \varepsilon_w \otimes \varepsilon_v$, the process in Figure 2.3 is: $\varepsilon_u \otimes \varepsilon_t \otimes \varepsilon_w \otimes \varepsilon_u \otimes \varepsilon_v$ and the process in Figure 2.4 is: $\varepsilon_t \otimes \varepsilon_t \otimes \varepsilon_t \otimes \varepsilon_u \otimes \varepsilon_u \otimes \varepsilon_v \otimes \varepsilon_v$.

**Proposition 2.3.** $\varepsilon \otimes \delta = \delta \otimes \varepsilon \Leftrightarrow \varepsilon = \delta \vee {}^\bullet\varepsilon \cap \delta^\bullet = \varepsilon^\bullet \cap {}^\bullet\delta = \varepsilon^\bullet \cap \delta^\bullet = \emptyset$ *for any* $\varepsilon, \delta \in PR_0$.

*Proof.* For $\varepsilon = \delta$ it holds obviously, so let us assume $\varepsilon \neq \delta$ and prove $\varepsilon \otimes \delta = \delta \otimes \varepsilon \Leftrightarrow {}^\bullet\varepsilon \cap \delta^\bullet = \varepsilon^\bullet \cap {}^\bullet\delta = \varepsilon^\bullet \cap \delta^\bullet = \emptyset$. Note that $\varepsilon \otimes \delta = \delta \otimes \varepsilon \Rightarrow (\varepsilon = \varepsilon^\delta \wedge \delta = \delta^\varepsilon)$. Indeed, $\varepsilon \subseteq \varepsilon \otimes \delta$ and $\delta \subseteq \delta \otimes \varepsilon$, thus $\varepsilon \cup \delta \subseteq \varepsilon \otimes \delta \cup \delta \otimes \varepsilon = \varepsilon \cup \delta^\varepsilon$, which implies $\delta = \delta^\varepsilon$ (since both $\varepsilon \cup \delta$ and $\varepsilon \cup \delta^\varepsilon$ are two-element sets). Similarly, $\varepsilon = \varepsilon^\delta$. If the two-element sets $\varepsilon \cup \delta^\varepsilon$ and $\delta \cup \varepsilon^\delta$ are distinct then obviously either $\varepsilon \neq \varepsilon^\delta$ or$\delta \neq \delta^\varepsilon$. Thus $(\varepsilon = \varepsilon^\delta \wedge \delta = \delta^\varepsilon) \Leftrightarrow \varepsilon \otimes \delta = \delta \otimes \varepsilon$. Equations $\varepsilon = \varepsilon^\delta$ and $\delta = \delta^\varepsilon$ imply: ${}^\bullet\varepsilon \cap \delta^\bullet = \emptyset$ $(x \in {}^\bullet\varepsilon \cap \delta^\bullet$ prevents $\varepsilon = \varepsilon^\delta)$, $\varepsilon^\bullet \cap {}^\bullet\delta = \emptyset$ $(x \in \varepsilon^\bullet \cap {}^\bullet\delta$

prevents $\delta = \delta^\varepsilon$) and $\varepsilon^\bullet \cap \delta^\bullet = \emptyset$ ($x \in \varepsilon^\bullet \cap \delta^\bullet$ prevents both $\varepsilon = \varepsilon^\delta$ and $\delta = \delta^\varepsilon$). Therefore, $\varepsilon \otimes \delta = \delta \otimes \varepsilon \Rightarrow {}^\bullet\varepsilon \cap \delta^\bullet = \varepsilon^\bullet \cap {}^\bullet\delta = \varepsilon^\bullet \cap \delta^\bullet = \emptyset$. Now, suppose $\varepsilon \otimes \delta \neq \delta \otimes \varepsilon$, that is either $\varepsilon \neq \varepsilon^\delta$ or $\delta \neq \delta^\varepsilon$. If $\varepsilon \neq \varepsilon^\delta$ then either ${}^\bullet\varepsilon \cap \delta^\bullet \neq \emptyset$ or $\varepsilon^\bullet \cap \delta^\bullet \neq \emptyset$, if $\delta \neq \delta^\varepsilon$ then either $\varepsilon^\bullet \cap {}^\bullet\delta \neq \emptyset$ or $\varepsilon^\bullet \cap \delta^\bullet \neq \emptyset$. Therefore, ${}^\bullet\varepsilon \cap \delta^\bullet = \varepsilon^\bullet \cap {}^\bullet\delta = \varepsilon^\bullet \cap \delta^\bullet = \emptyset \Rightarrow \varepsilon \otimes \delta = \delta \otimes \varepsilon$. □

Proposition 2.3 directly implies:

**Theorem 2.1.** *If $\alpha = \varepsilon_1 \otimes \varepsilon_2 \otimes ... \otimes \varepsilon_n = \delta_1 \otimes \delta_2 \otimes ... \otimes \delta_m$ where $\varepsilon_j$ ($j = 1, 2, ..., n$) and $\delta_k$ ($k = 1, 2, ..., m$) are atomic processes, then sequence $\delta_1, \delta_2, ..., \delta_m$ is a permutation of sequence $\varepsilon_1, \varepsilon_2, ..., \varepsilon_n$ (hence $n = m$). Two consecutive atoms $\varepsilon_j, \varepsilon_{j+1}$ may be swapped without affecting process $\alpha$ iff $\varepsilon_j = \varepsilon_{j+1} \vee {}^\bullet\varepsilon_j \cap \varepsilon_{j+1}^\bullet = \varepsilon_j^\bullet \cap {}^\bullet\varepsilon_{j+1} = \varepsilon_j^\bullet \cap \varepsilon_{j+1}^\bullet = \emptyset$.*

Let us discuss peculiarity of the notion of process formalised in Definition 2.2 and a relationship between processes and firing sequences.

## Action (in)dependence – A general remark

Consider the net in Figure 2.5 with initial and final markings $M_0$ and $M$ respectively, such that $M_0(a) = 4$, $M_0(b) = M_0(c) = 0$, $M(a) = 0$, $M(b) = M(c) = 2$.

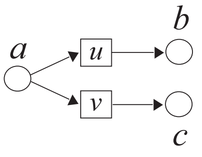


Figure 2.5

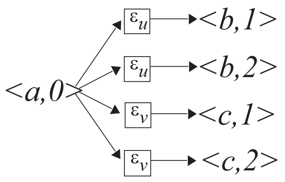


Figure 2.6

The net evolves into the process in Figure 2.6 and this *one* process is described by each of the following *six* firing sequences: $uuvv, uvuv, uvvu, vvuu, vuvu, vuuv$. Transitions $u$ and $v$ are independent in the sense that order of their firing is irrelevant for any marking reachable from $M_0$. This is so, however, only if the notion of actions' independence is understood with respect to the marking as the system's state. To clarify this point of view, let us consider the net in Figure 2.7, interpreted by actions inscribed into transitions. Let the initial marking be $M_0$ with $M_0(a) = M_0(b) = 1$, $M_0(c) = M_0(d) = 0$.

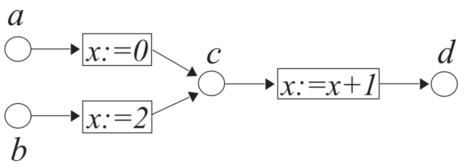


Figure 2.7

Actions $x := 0$ and $x := 2$ are mutually independent with respect to the state being the marking only, because any marking reachable from $M_0$, in particular the final marking $M$ with $M(a) = M(b) = 0$, $M(c) = M(d) = 1$, does not depend on the order of their execution. However, actions $x := 0$ and $x := 2$ are mutually dependent with respect to the final state composed of $M$ augmented by the value of integer variable $x$, as shows the table in Figure 2.8.

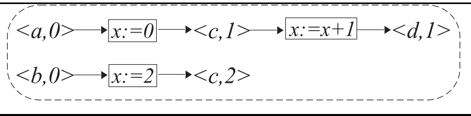
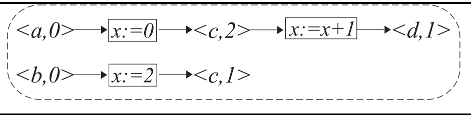
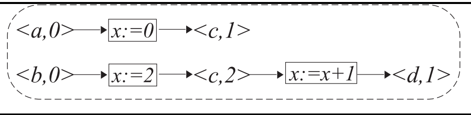
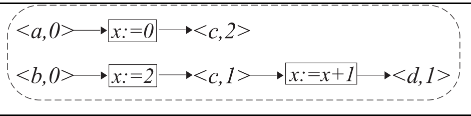
| Firing sequence | Final state | Process |
|---|---|---|
| $x := 0;\ x := x + 1;\ x := 2$ | $M,\ x = 2$ |  |
| $x := 2;\ x := 0;\ x := x + 1$ | $M,\ x = 1$ |  |
| $x := 0;\ x := 2;\ x := x + 1$ | $M,\ x = 3$ |  |
| $x := 2;\ x := x + 1;\ x := 0$ | $M,\ x = 0$ |  |

FIGURE 2.8

Obviously, under a different interpretation, transitions $(\{a\}, \{c\})$ and $(\{b\}, \{c\})$ may become independent with respect to the final state $(M, x)$. For instance, if $x := 0$ is replaced by $x := x$ and $x := x + 1$ by $x := x^2 - 2$ then the final state becomes always $(M, 2)$.

Summarising, the net structure alone enforces (in)dependence of transitions with respect to the state restricted to its marking component. Their (in)dependence with respect to the whole state is subject, in general, to the net structure and interpretation. Regarding the example in Figure 2.8, the reader is encouraged to compare construction of P/T net process in [1, 3, 12, 15, 16] with that given in Definition 2.2: with the obvious meaning of isomorphism between processes (*e.g.* isomorphic are the first two processes as well as the last two), two isomorphism classes of our processes correspond to processes presented in the mentioned references.

## Causal dependence in processes

Causal dependence and concurrency of events are directly inferred from the process construct. Events are process constituents intended to be occurrences of transitions of an underlying net. They are atomic processes suitably renumbered, thus of the form $\{(A \times \mathbb{N}_A, B \times \mathbb{N}_B)\}$ with $A, B \subseteq \mathbb{X}$, $\mathbb{N}_A, \mathbb{N}_B \subseteq \mathbb{N}$. Let a process $\alpha$ be factorised into atomic processes (Prop. 2.2): $\alpha = \varepsilon_1 \otimes \varepsilon_2 \otimes ... \otimes \varepsilon_n$. To each $\varepsilon_j$ and number $j$ there corresponds a *representation* $\varepsilon_j^{\alpha_{j-1}}$ (w.r.t. the factorisation) of an event where $\alpha_{j-1}$ is the prefix $\varepsilon_1 \otimes \varepsilon_2 \otimes ... \otimes \varepsilon_{j-1}$ $(2 \le j \le n)$ of $\alpha$ – recall Definition 2.2 ($\varepsilon_1$ represents event $\{({}^\bullet\varepsilon_1 \times \{0\}, \varepsilon_1^\bullet \times \{1\})\}$). Thus, for a fixed factorisation, $\varepsilon_1 \otimes \varepsilon_2 \otimes ... \otimes \varepsilon_n$ each event is uniquely determined by a certain pair $(\varepsilon_j, j)$. It follows from Theorem 2.1 that any other factorisation determines the same set of events. Event $\xi$ *precedes* $\eta$ in a process $\alpha$ $(\xi \prec_\alpha \eta)$ iff for each factorisation $\varepsilon_1 \otimes \varepsilon_2 \otimes ... \otimes \varepsilon_n$ of $\alpha$, if $\xi$ and $\eta$ are represented by $(\varepsilon_i, i)$ and $(\varepsilon_j, j)$ respectively, then $i < j$. Events $\xi$ and $\eta$ are *concurrent* in a process $\alpha$ iff $\neg(\xi \prec_\alpha \eta \vee \eta \prec_\alpha \xi)$ and $\xi \neq \eta$. For instance, in the process $\alpha$ in Figure 2.9 $\{(\{\langle c,0\rangle\}, \{\langle d,1\rangle\})\} \prec_\alpha \{(\{\langle d,1\rangle\}, \{\langle e,1\rangle\})\}$, while event $\{(\{\langle a,0\rangle\}, \{\langle b,1\rangle\})\}$ is concurrent to both former events.

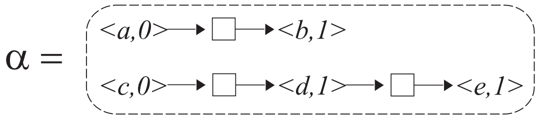


FIGURE 2.9

It should be noticed that sometimes a process may keep the information that one of two events occurred first despite their appearance as concurrent. An example is the process in Figure 2.4 where event $\xi = \{(\{\langle d,0\rangle, \langle e,2\rangle\}, \{\langle c,1\rangle\})\}$ precedes $\eta = \{(\{\langle d,0\rangle, \langle e,2\rangle\}, \{\langle c,2\rangle\})\}$, though it looks as if transition $v$ in Figure 1.3 might have fired twice "at the same time" (the so-called self-concurrence). Their time-ordering is irrelevant from the computational point of view. This is an artefact being an outcome of numbering of place occurrences. Obviously, interchanging $\langle c,1\rangle$ and $\langle c,2\rangle$ in Figure 2.4 does not affect the process, thus its graph. This artefact is one of some features by which our construct of process differs from other constructs known in the literature.

## A correspondence between firing sequences and processes

As the forthcoming Theorem 2.2 states, although many firing sequences (thus observations of a system activity) may correspond to one and the same process, a given firing sequence corresponds to exactly one. This is not the case with (at least some, known to me) standard models of processes. Consider, for instance, again the table in Figure 2.8. Neglecting occurrence numbers of places in processes in the second and third rows, two standard (*i.e.* constructed as in *e.g.* [1, 16] and many other publications) distinct processes are obtained. But both processes correspond to the firing sequence in the second row (as well as to the one in the third row).

## Forward place-branching

The next feature of our processes is forward place-branching which is not at all a nondeterminism: processes developed here are essentially deterministic. Let us explain it in more detail. Recall again the inductive Definition 2.2. If $\alpha$ is a process constructed "so far" and $\varepsilon^\alpha$ – an event then a certain process-place $\langle x,k\rangle$ from the pre-set of $\varepsilon^\alpha$ (but not from its post-set!) may turn out identical with a certain process-place occurring in the "interior" of $\alpha$ (*i.e.* such that one or more arrows emanate from it in $\alpha$). After attachment of $\varepsilon^\alpha$ to $\alpha$, one more arrow will emanate from $\langle x,k\rangle$ in the extended process $\alpha\cup\varepsilon^\alpha$, because each process-place $\langle x,k\rangle$ should occur uniquely in the graphical presentation of process (notice that this way of mapping a set of events onto a picture is exactly the same as mapping a set of transitions – thus a net – onto a picture as shown in Sect. 1). An example is in Figure 2.4: from the process-place $\langle b,0\rangle$ three arrows emanate and from each of process-places $\langle a,3\rangle$, $\langle d,0\rangle$ and $\langle e,2\rangle$ two arrows emanate. This is interpreted as follows. If the net-place $b$ in Figure 1.3 contains three tokens at least, then transition $t=(\{b\},\{a\})$ may fire three times; events $\{(\{\langle b,0\rangle\},\{\langle a,1\rangle\})\}$, $\{(\{\langle b,0\rangle\},\{\langle a,2\rangle\})\}$ and $\{(\{\langle b,0\rangle\},\{\langle a,3\rangle\})\}$ are three consecutive occurrences (firings) of $t$; the occurrence numbers 1, 2 and 3 of the net-place $a$ account for the "time-order" of the firings. If the net-place $a$ contains two tokens at least then transition $u=(\{a\},\{b,e\})$ may fire twice; events $\{(\{\langle a,3\rangle\},\{\langle b,1\rangle,\langle e,1\rangle\})\}$ and $\{(\{\langle a,3\rangle\},\{\langle b,2\rangle,\langle e,2\rangle\})\}$ are two consecutive occurrences of $u$; the occurrence numbers 1 and 2 of the net-places $b$ and $e$ account for the "time-order" of the firings. If the net-places $d$ and $e$ contain at least two tokens each then transition $v=(\{d,e\},\{c\})$ may fire twice; events $\{(\{\langle d,0\rangle,\langle e,2\langle\},\{\langle c,1\rangle\})\}$ and $\{(\{\langle d,0\rangle,\langle e,2\rangle\},\{\langle c,2\rangle\})\}$ are two consecutive occurrences of $v$; the occurrence numbers 1 and 2 of the net-place $c$ account for the "time-order" of the firings.

To sum up this peculiar feature deviating our processes from the standard ones, let us note that although Definition 2.2 is independent of any underlying net, it is motivated by the "token game" rules as follows. Each time a transition $\tau$ is fired, **new** occurrences of its post-set places are generated as pairs of the form $\langle x,k\rangle$ where $k$ accounts for how many times a token entered the net-place $x$ until the process reached a stage ("cut" – in the standard terminology) with $\langle x,k\rangle$. In other words, a **new** process-place $\langle x,k\rangle$ ($k>0$) is generated when a token enters $x$. On the other hand, **no new** occurrence of a place $x$ from the pre-set of $\tau$ is generated if it has been generated "so far", except for the case when $x$ did not take part in the net's activity yet ($\langle x,0\rangle$ is then generated). In other words, **no new** occurrence of a place $x$ is generated when a token exits $x$ – save for the first exit. Therefore, to each process-place one arrow at the most may enter (no backward process-place branching), while it may happen that more than one arrow exits it (possible forward process-place branching).

## Order of appending atoms

One may also ask why the two arrows emanate from $\langle a,3\rangle$ but not from $\langle a,1\rangle$ or $\langle a,2\rangle$ in Figure 2.4? This is another artefact of the model. It stems from the process construction: the successor $\mathrm{suc}_\varepsilon(\alpha)$ is computed in such a way that the occurrence numbers of places in the event $\varepsilon^\alpha$ become $I_\alpha(x)$ if $x\in{}^\bullet\varepsilon$ and $1+I_\alpha(x)$ if $x\in\varepsilon^\bullet$ where $I_\alpha(x)$ is the greatest occurrence number of the net-place $x$ in $\alpha$. The meaning of this artefact might be seen as follows: to obtain process $\mathrm{suc}_\varepsilon(\alpha)$, among all occurrences $\langle x,k\rangle$ in $\alpha$ of the same net-place $x$, the latest is chosen to make the link between $\alpha$ and $\mathrm{suc}_\varepsilon(\alpha)$. It is worth noting that occurrence number $k$ has nothing to do with an order of tokens' entering or leaving net-place $x$ (tokens are indistinguishable) but reflects an order of appending atoms in the course of process construction.

## Firing sequences *vs.* processes

Now, let us, in general, compare P/T net's behaviour represented by firing sequences and processes. The unbounded capacity of places with as many tokens as needed (nets without initial marking, in fact) imply that the notions of firing sequence and arbitrary sequence of transitions coincide. That is why the $\varepsilon$-successor (Def. 2.2) is unconditionally defined. One may, thus, correctly define a process generated by a sequence of transitions as a function pr: $\mathbb{T}^* \to PR_{\mathrm{fin}}$ inductively specified by:

$$\mathrm{pr}(\lambda) = \emptyset,\ \mathrm{pr}(Ut) = \mathrm{pr}(U) \otimes \{({}^\bullet t \times \{0\}, t^\bullet \times \{1\})\} \text{ for } U \in \mathbb{T}^*, t \in \mathbb{T}.$$

Let two finite firing sequences $U$ and $V$ be given. Define a binary relation $\sim\ \subseteq \mathbb{T}^* \times \mathbb{T}^*$: $U \sim V$ iff $U = WuvW' \wedge V = WvuW' \wedge (u = v \vee {}^\bullet u \cap v^\bullet = u^\bullet \cap {}^\bullet v = u^\bullet \cap v^\bullet = \emptyset)$ for some firing sequences $W$, $W'$ and transitions $u, v$. Let $\overset{*}{\sim}$ (equivalence) be the reflexive and transitive closure of $\sim$. Readers familiar with Mazurkiewicz traces [13, 21] will recognise equivalence classes of $\overset{*}{\sim}$ as his traces with the independence relation $\sim \setminus \mathrm{id}$ (id – identity relation).

**Theorem 2.2.** *For any firing sequences $U, V \in \mathbb{T}^*$:*

(a) $\mathrm{pr}(UV) = \mathrm{pr}(U) \otimes \mathrm{pr}(V)$*, that is,* pr *is a homomorphism from the monoid* $(\mathbb{T}^*, ., \lambda)$ *into* $(\mathrm{PR}_{\mathrm{fin}}, \otimes, \emptyset)$*;*

(b) $U \overset{*}{\sim} V \Leftrightarrow \mathrm{pr}(U) = \mathrm{pr}(V)$*, that is, the quotient monoid* $(\mathbb{T}^*/_{\overset{*}{\sim}}, ., \{\lambda\})$ *and* $(PR_{\mathrm{fin}}, \otimes, \emptyset)$ *are isomorphic.*

*Proof of (a).* Base: for $U$ and $V$ being $\lambda$ or a transition, (a) holds obviously. Inductive hypotheses: $V = V't$, $\mathrm{pr}(UV') = \mathrm{pr}(U) \otimes \mathrm{pr}(V')$, $\mathrm{pr}(V't) = \mathrm{pr}(V') \otimes \mathrm{pr}(t)$. Then, $\mathrm{pr}(UV) = \mathrm{pr}(UV't) =$ (by pr definition) $\mathrm{pr}(UV') \otimes \{({}^\bullet t \times \{0\}, t^\bullet \times \{1\})\}$ =(by hypotheses and pr definition) $\mathrm{pr}(U) \otimes \mathrm{pr}(V') \otimes \mathrm{pr}(t) = \mathrm{pr}(U) \otimes \mathrm{pr}(V)$.

*Proof of (b).* ($\Rightarrow$)-part. Suppose $U \sim V$, that is $U = WuvW' \wedge V = WvuW' \wedge (u = v \vee {}^\bullet u \cap v^\bullet = u^\bullet \cap {}^\bullet v = u^\bullet \cap v^\bullet = \emptyset)$ for some firing sequences $W$, $W'$ and transitions $u, v$. By (a): $\mathrm{pr}(U) = \mathrm{pr}(W) \otimes \varepsilon \otimes \delta \otimes \mathrm{pr}(W')$ and $\mathrm{pr}(V) = \mathrm{pr}(W) \otimes \delta \otimes \varepsilon \otimes \mathrm{pr}(W')$ with $\varepsilon = \mathrm{pr}(u)$, $\delta = \mathrm{pr}(v)$, $\varepsilon, \delta \in \mathrm{PR}_0$. If $u = v$ then $\mathrm{pr}(U) = \mathrm{pr}(V)$, so let $u \neq v$. Since ${}^\bullet\varepsilon = {}^\bullet u$, $\varepsilon^\bullet = u^\bullet$ for any atomic process $\varepsilon = \{({}^\bullet u \times \{0\}, u^\bullet \times \{1\})\}$, with $u \in \mathbb{T}$, thus (by assumption $U \sim V$) ${}^\bullet\varepsilon \cap \delta^\bullet = \varepsilon^\bullet \cap {}^\bullet\delta = \varepsilon^\bullet \cap \delta^\bullet = \emptyset$, hence, by Proposition 2.3: $\varepsilon \otimes \delta = \delta \otimes \varepsilon$, therefore, $\mathrm{pr}(U) = \mathrm{pr}(V)$. By inductive reasoning, we get $U \overset{*}{\sim} V \Rightarrow \mathrm{pr}(U) = \mathrm{pr}(V)$.

($\Leftarrow$)-part. Suppose $\mathrm{pr}(U) = \mathrm{pr}(V)$. Let $U = u_1 u_2 ... u_n$ and $V = v_1 v_2 ... v_n$, $u_j, v_j \in \mathbb{T}$. By (a): $\mathrm{pr}(U) = \varepsilon_1 \otimes \varepsilon_2 \otimes ... \otimes \varepsilon_n$, $\mathrm{pr}(V) = \delta_1 \otimes \delta_2 \otimes ... \otimes \delta_n$ where $\varepsilon_j = \mathrm{pr}(u_j)$,

$\delta_j = \mathrm{pr}(v_j)$, $\varepsilon_j, \delta_j \in \mathrm{PR}_0$ and by pr definition: $\varepsilon_j = \{({}^\bullet u_j \times \{0\}, u_j^\bullet \times \{1\})\}$, $\delta_j = \{({}^\bullet v_j \times \{0\}, v_j^\bullet \times \{1\})\}$ $(j = 1, 2, ..., n)$. By assumption $\mathrm{pr}(U) = \mathrm{pr}(V)$ and Theorem 2.1, the sequence $\delta_1, \delta_2, ..., \delta_n$ is a permutation of $\varepsilon_1, \varepsilon_2, ..., \varepsilon_n$ that is, a composition of transpositions (swaps) of some adjacent $\varepsilon'$s occurring in the latter sequence and every such transposition preserves the equation $\mathrm{pr}(U) = \mathrm{pr}(V)$. If, $\varepsilon_1 \otimes \varepsilon_2 \otimes ... \otimes (\varepsilon_k \otimes \varepsilon_{k+1}) \otimes ... \otimes \varepsilon_n = \varepsilon_1 \otimes \varepsilon_2 \otimes ... \otimes (\varepsilon_{k+1} \otimes \varepsilon_k) \otimes ... \otimes \varepsilon_n$ then $\varepsilon_k \otimes \varepsilon_{k+1} = \varepsilon_{k+1} \otimes \varepsilon_k$ thus, by Proposition 2.3: $\varepsilon_k = \varepsilon_{k+1} \vee {}^\bullet\varepsilon_k \cap \varepsilon_{k+1}^\bullet = \varepsilon_k^\bullet \cap {}^\bullet\varepsilon_{k+1} = \varepsilon_k^\bullet \cap \varepsilon_{k+1}^\bullet = \emptyset$. This implies $u_k = u_{k+1} \vee {}^\bullet u_k \cap u_{k+1}^\bullet = u_k^\bullet \cap {}^\bullet u_{k+1} = u_k^\bullet \cap u_{k+1}^\bullet = \emptyset$, that is, $u_1 u_2 ... u_k u_{k+1} ... u_n \sim u_1 u_2 ... u_{k+1} u_k ... u_n$. Therefore, by composing such permitted transpositions, we get $U \overset{*}{\sim} V$. □

## 3. Marked net processes. Analysis and synthesis

The processes in Section 2 have been defined without relating them to a particular P/T net. The approach was like that in the theory of formal languages, where words are defined regardless of their generative devices like automata, grammars or fix-point equations. An example of this approach, closer to the field of concurrency, is the trace theory [13]. Such mutual independence of processes and generative nets, of their initial marking in particular, makes the $\varepsilon$-successor (Def. 2.2) operation unconditionally performable: it is always possible to attach an atom to a finite process $\alpha$. Note that this is as if a net whose evolution the process might represent were marked with infinite number of tokens in each place. In fact, for any net-place $x$ the difference between the number of arrows going out of and coming into each of its occurrence $\langle x, k\rangle \in Y_\alpha$ in each prefix of the process $\alpha$ must never exceed the initial number of tokens in $x$ in the underlying net. Before defining the set of finite processes generated by marked P/T nets let us introduce denotations:

Let a finite process $\alpha = \varepsilon_1 \otimes \varepsilon_2 \otimes ... \otimes \varepsilon_n$, $\varepsilon_j \in PR_0$, $j = 1, 2, ..., n$ be given (recall Prop. 2.2). Characteristic function of a set $A$ is defined as $\chi_A(x) = 1$ for $x \in A$ and $\chi_A(x) = 0$ for $x \notin A$. Denote:

$O_\alpha(x) = \sum_{j=1}^{n} \chi_{{}^\bullet\varepsilon_j}(x)$ number of arrows going out of all $\langle x, k\rangle$ in $\alpha$

$I_\alpha(x) = \sum_{j=1}^{n} \chi_{\varepsilon_j^\bullet}(x)$ number of arrows coming into all $\langle x, k\rangle$ in $\alpha$.

For $\alpha = \mathbf{1}$ assume $O_\alpha(x) = I_\alpha(x) = 0$ for any $x$.

$FOLD(\alpha) = \bigcup_{j=1}^{n} \{({}^\bullet\varepsilon_j, \varepsilon_j^\bullet)\}$. Folding of $\alpha$, that is, the set of projections of transitions of $\alpha$ ($\alpha$ is treated as a net over $\mathbb{X} \times \mathbb{N}$) onto $\mathbb{X}$. For instance, projection of process transition $(\{\langle a, 0\rangle\}, \{\langle b, 1\rangle\})$ over $\mathbb{X} \times \mathbb{N}$ onto $\mathbb{X}$ is the net transition $(\{a\}, \{b\})$ over $\mathbb{X}$.

For $\alpha = \mathbf{1}$ assume $FOLD(\alpha) = \emptyset$.

Functions $O_\alpha$, $I_\alpha$ and $FOLD$ are correctly defined since they do not depend on factorisation of $\alpha$ into atoms. $O_\alpha$, $I_\alpha$ are treated as multisets and operations $+$, $-$ and comparison $\leq$ on multisets will be applied to them.

A finite process $\beta$ is a *prefix* of $\alpha$ iff $\alpha = \beta \otimes \gamma$ for a certain finite process $\gamma$. $PREF(\alpha)$ denotes the set of all prefixes of $\alpha$ and for $L \subseteq PR_{\text{fin}} : PREF(L) = \bigcup\limits_{\alpha \in L} PREF(\alpha)$. Let us recall that a marked net is $\mathcal{N} = (T, M)$ where $T \subseteq \mathbb{T}$, and $M\colon\ \mathbb{X} \to \mathbb{N}$ is an initial marking.

**Definition 3.1** (Atomic and finite processes generated by a marked net)**.** Let a marked net $\mathcal{N} = (T, M)$ be given. The set of atomic and, respectively, finite processes generated by $\mathcal{N}$ is:

$$PR_0[\mathcal{N}] = \{\varepsilon \in PR_0 |\ \mathrm{F}OLD(\varepsilon) \subseteq T \wedge O_\varepsilon - I_\varepsilon \leq M\}$$
$$PR_{\text{fin}}[\mathcal{N}] = \{\alpha \in PR_{\text{fin}} |\ \mathrm{F}OLD(\alpha) \subseteq T \wedge \forall \beta \in PREF(\alpha) : O_\beta - I_\beta \leq M\}\cdot$$

**Remarks.** (1) $M(x) - (O_\alpha(x) - I_\alpha(x))$ is a number of tokens collected in the place $x$ in effect of generating process $\alpha$. $M - (O_\alpha - I_\alpha)$ is a marking reached from $M$ in effect of generating process $\alpha$.

(2) It should be noticed that Definition 3.1 would not be correct for nets with arbitrary arrow weights. In that case it should be strenghten by assumig each transition to be pure, *i.e.* ${}^\bullet t \cap t^\bullet = \emptyset$.

For instance, consider the net $T_{\text{prodcons}}$ in Figure 1.3 with marking $M = \begin{array}{|c|c|c|c|c|}\hline a & b & c & d & e \\ \hline 1 & 0 & 1 & 0 & 0 \\ \hline \end{array}$. Let $\gamma, \delta, \eta$ be processes depicted in Figures 2.2, 2.3 and 2.4 respectively. Then $\gamma, \delta \in PR_{\text{fin}}[T_{\text{prodcons}}, M]$, but $\eta \notin PR_{\text{fin}}[T_{\text{prodcons}}, M]$. Indeed, it may be checked that $O_\beta - I_\beta \leq M$ for each prefix $\beta$ of $\gamma$ and $\delta$, but *e.g.* $O_\eta - I_\eta \nleq M$. For $M' = \begin{array}{|c|c|c|c|c|}\hline a & b & c & d & e \\ \hline 0 & 3 & 0 & 2 & 0 \\ \hline \end{array}$ we have for each prefix $\beta$ of $\eta$ : $O_\beta - I_\beta \leq M'$ thus $\eta \in PR_{\text{fin}}[T_{\text{prodcons}}, M']$.

Recall from the Petri net theory that for an unmarked net $T$ (with all the arrow weights equal 1), the incidence matrix $\underline{T}$ of $T$ is a matrix with rows corresponding to places, columns - to transitions (ordered in a fixed way) in $T$ and

$$\underline{T}[x, t] = \begin{cases} +1 & \text{if } x \in t^\bullet \backslash {}^\bullet t \\ -1 & \text{if } x \in\ {}^\bullet t \backslash t^\bullet \\ 0 & \text{else.} \end{cases}$$

Given a marked net $\mathcal{N} = (T, M)$, let $\alpha \in PR_{\text{fin}}[\mathcal{N}]$ with $\alpha = \varepsilon_1 \otimes \varepsilon_2 \otimes \ldots \otimes \varepsilon_n$, $\varepsilon_j \in PR_0[\mathcal{N}]$. Let $\underline{\alpha}$ denote a vector whose components $\underline{\alpha}[t]$ correspond to transitions $t \in T$ ordered as in the matrix $\underline{T}$ and $\underline{\alpha}[t] =$ number of indices $j$ in $\varepsilon_1 \otimes \varepsilon_2 \otimes \ldots \otimes \varepsilon_n$ with $({}^\bullet \varepsilon_j, \varepsilon_j^\bullet) = t$. That is, $\underline{\alpha}[t]$ is a number saying how many times the transition $t$ has been fired in effect of generating process $\alpha$. Note that due to the Theorem 2.1, $\underline{\alpha}$ does not depend on factorisation of $\alpha$ into atoms.

**Example 3.1.** For the net $T_{\text{prodcons}}$ in Figure 1.3 and process $\alpha$ in Figure 2.2:

$\underline{T_{\text{prodcons}}} =$

| | $t$ | $u$ | $v$ | $w$ |
|---|---|---|---|---|
| $a$ | $+1$ | $-1$ | $0$ | $0$ |
| $b$ | $-1$ | $+1$ | $0$ | $0$ |
| $c$ | $0$ | $0$ | $+1$ | $-1$ |
| $d$ | $0$ | $0$ | $-1$ | $+1$ |
| $e$ | $0$ | $+1$ | $-1$ | $0$ |

$\underline{\alpha} =$

| $t$ | $u$ | $v$ | $w$ |
|---|---|---|---|
| 1 | 2 | 2 | 2 |

**Theorem 3.1.** *Let a marked net* $\mathcal{N} = (T, M)$ *and a process* $\alpha \in PR_{\text{fin}}[\mathcal{N}]$ *be given. Then* $\underline{T}{\cdot}\underline{\alpha} = I_\alpha - O_\alpha$.

*Proof.* Let $\alpha = \varepsilon_1 \otimes \varepsilon_2 \otimes ... \otimes \varepsilon_n \in PR_{\text{fin}}[\mathcal{N}]$, for some $\varepsilon_j \in PR_0$. By definition of $\underline{T}$: $\underline{T}[x,t] = \chi_{t^\bullet}(x) - \chi_{{}^\bullet t}(x)$. Thus, the inner product $\sum_{t\in T} \underline{T}[x,t] \cdot \underline{\alpha}[t]$ equals $\sum_{t\in T} \chi_{t^\bullet}(x) \cdot \underline{\alpha}[t] - \sum_{t\in T} \chi_{{}^\bullet t}(x) \cdot \underline{\alpha}[t]$. But $\sum_{t\in T} \chi_{t^\bullet}(x) \cdot \underline{\alpha}[t]$ is the number of tokens added to place $x$ in effect of generating process $\alpha$ and $\sum_{t\in T} \chi_{{}^\bullet t}(x){\cdot}\underline{\alpha}[t]$ is the number of tokens removed from $x$ in effect of generating $\alpha$. Hence, $\sum_{t\in T} \chi_{t^\bullet}(x) \cdot \underline{\alpha}[t] = \sum_{j=1}^n \chi_{\varepsilon_j^\bullet}(x) = I_\alpha(x)$ and $\sum_{t\in T} \chi_{{}^\bullet t}(x) \cdot \underline{\alpha}[t] = \sum_{j=1}^n \chi_{{}^\bullet \varepsilon_j}(x) = O_\alpha(x)$. Therefore $(\underline{T}{.}\underline{\alpha})(x) = I_\alpha(x) - O_\alpha(x)$. □

Now, we are in a position to state and prove analysis and synthesis problems for marked P/T nets and languages of finite processes. Let a monoid $\mathcal{M}_{\text{fin}} = (PR_{\text{fin}}, \otimes, \mathbf{1})$ be given. Its power algebra $\mathcal{S}_{\text{fin}} = (\mathcal{P}(PR_{\text{fin}}), \cup, \otimes, \emptyset, \mathbf{1})$ (where $\mathcal{P}(PR_{\text{fin}})$ is the powerset of $PR_{\text{fin}}$) is induced by $\mathcal{M}_{\text{fin}}$ as usually, that is by extending catenation of processes to their sets: $A \otimes B = \{\alpha \otimes \beta |\ \alpha \in A \wedge \beta \in B\}$ for $A, B \subseteq PR_{\text{fin}}$. Sets $A, B$ are *languages of (finite) processes.* The algebra $\mathcal{S}_{\text{fin}}$ is an *$\omega$-complete semiring* and a number of its laws may be found *e.g.* in [7–10]. Let us mention the following:

**Proposition 3.1.** *For any countable indexed family of process languages* $\{A_j\}_{j\in I}$ *(I is an index set) and a process language B:* $B \otimes \bigcup_{j\in I} A_j = \bigcup_{j\in I} B \otimes A_j$.

As usually, the closure (a counterpart of the Klenee star) based on catenation $\otimes$ is defined: for a process language $A \subseteq PR_{\text{fin}}$ denote: $A^0 = \{\mathbf{1}\}$, $A^{k+1} = A^k \otimes A$, $A^\otimes = \bigcup_{k=0}^{\infty} A^k$.

The *analysis problem* is: given a marked net $\mathcal{N}$ find $PR_{\text{fin}}[\mathcal{N}]$.

**Theorem 3.2** (Analysis)**.** *For a given marked net* $\mathcal{N} = (T, M)$*:*

$$PR_{\text{fin}}[\mathcal{N}] = \{\alpha \in PR_0[T]^\otimes |\ \ \forall \beta \in PREF(\alpha) :\ -\underline{T} \cdot \underline{\beta} \le M\}$$

*where* $PR_0[T] = \{\varepsilon \in PR_0|\ \ FOLD(\varepsilon) \subseteq T\}$ *is the set of atoms (see Def. 2.1 and Ex. 2.1) corresponding to transitions in the net T.*

*Proof.* $PR_{\text{fin}}[\mathcal{N}] = \{\alpha \in PR_{\text{fin}}|\ \ FOLD(\alpha) \subseteq T \wedge \forall \beta \in PREF(\alpha)$: $O_\beta - I_\beta \le M\}$ = (by Th. 3.1) $\{\alpha \in PR_{\text{fin}}|\ \ FOLD(\alpha) \subseteq T \wedge \forall \beta \in PREF(\alpha)$: $-\underline{T} \cdot \underline{\beta}$

$\le M\}$. Thus, it remains to check that $PR_0[T]^{\otimes} = \{\alpha \in PR_{\text{fin}} |\ FOLD(\alpha) \subseteq T\}$. Indeed, $PR_0[T]^{\otimes} = \{\alpha \in PR_{\text{fin}} |\ \exists \varepsilon_1, \varepsilon_2, ..., \varepsilon_n \in PR_0[T]\colon \alpha = \varepsilon_1 \otimes \varepsilon_2 \otimes ... \otimes \varepsilon_n\} \cup \{\mathbf{1}\} = \{\alpha \in PR_{\text{fin}} |\ FOLD(\alpha) \subseteq T\}$ because $FOLD(\alpha) = \bigcup_{j=1}^{n} FOLD(\varepsilon_j)$ and $FOLD(\varepsilon_j) \subseteq T$. □

**Corollary.** $PR_{\text{fin}}[T, M]$ *is the greatest set* $X$ *of processes* $\alpha$ *satisfying:*

$$\begin{aligned} FOLD(\alpha) &\subseteq T \\ X &= PREF(X) \\ X &= \{\alpha |\ -\underline{T} \cdot \underline{\alpha} \le M\} \cdot \end{aligned}$$

For example, if $\mathcal{N} = (T_{\text{prodcons}},\ M)$ with $T_{\text{prodcons}} = \{t, u, v, w\}$ as in Figure 1.3 and $M = \begin{array}{|c|c|c|c|c|} \hline a & b & c & d & e \\ \hline 1 & 0 & 1 & 0 & 0 \\ \hline \end{array}$ then $PR_0[T_{\text{prodcons}}] = \{\varepsilon_t, \varepsilon_u, \varepsilon_v, \varepsilon_w\}$ as in Figure 2.1 and $PR_{\text{fin}}[\mathcal{N}] = \{\alpha \in \{\varepsilon_t, \varepsilon_u, \varepsilon_v, \varepsilon_w\}^{\otimes} |\ \forall \beta \in PREF(\alpha)\colon -\underline{T_{\text{prodcons}}} \cdot \underline{\beta} \le M\}$ where matrix $\underline{T_{\text{prodcons}}}$ is in Example 3.1.

The *synthesis problem* is: given a process language $L \subseteq PR_{\text{fin}}$ decide if there exists a marked net $\mathcal{N}$ such that $L = PR_{\text{fin}}[\mathcal{N}]$ and in the positive case find this net. Say then that $L$ is *net-definable*.

**Remark.** Here, we leave aside a formal meaning of the sentence "given a process language" (but see Sect. 5). The reader may assume any constructive specification of $L$, *e.g.* by a graph grammar or fix-point equations like in [9,10]. What matters is that the predicate $\alpha \in L$ should be decidable.

**Proposition 3.2.** *For a given process language* $L$ *define the marked net* $\mathcal{N} = (T, M)$*:* $T = \bigcup_{\alpha \in L} FOLD(\alpha)$, $M = \inf\{M' |\ M_\alpha \le M'$ *for each* $\alpha \in L\}$ *where* $M_\alpha = \inf\{M' |\ -\underline{T}.\underline{\beta} \le M'$ *for each prefix* $\beta$ *of* $\alpha\}$ *(*inf *means infimum of this set, i.e. the greatest lower bound w.r.t.* $\le$*). Then:*

(*a*) $L \subseteq PR_{\text{fin}}[\mathcal{N}]$*;*

(*b*) *the net* $\mathcal{N}$ *is minimal in the sense that* $T \subseteq T'$ *and* $M \le M'$ *for any marked net* $\mathcal{N}' = (T', M')$ *such that* $L \subseteq PR_{\text{fin}}[\mathcal{N}']$.

*Proof of (a).* In accordance with Section 1, $T$ is an unmarked net. If $\alpha \in L$ then $FOLD(\alpha) \subseteq T$. By Theorem 3.1: $M_\alpha = \inf\{M' |\ O_\beta - I_\beta \le M'$ for each prefix $\beta$ of $\alpha\}$, hence, by Definition 3.1: $\alpha \in PR_{\text{fin}}[T, M_\alpha]$, thus $\alpha \in PR_{\text{fin}}[T, M]$ (because $M_\alpha \le M$). Therefore $L \subseteq PR_{\text{fin}}[\mathcal{N}]$.

*Proof of (b).* Evident by definition of $\mathcal{N}$: removing a transition from $T$ or a token from any place in $T$ (*i.e.* making $T$ or $M$ smaller than constructed) would exclude from $PR_{\text{fin}}[\mathcal{N}]$ at least one process $\alpha \in L$. □

In general $L \ne PR_{\text{fin}}[\mathcal{N}]$ for $\mathcal{N}$ defined in Proposition 3.2. For instance, if $L$ contains exactly the process in Figure 2.2 and all its prefixes then $\mathcal{N} = (T_{\text{prodcons}},\ M)$. But $\mathcal{N}$ generates much more processes than that in Figure 2.2.

**Theorem 3.3** (Synthesis). *Let a process language $L$ be given and let $\mathcal{N} = (T, M)$ be a marked net constructed in Proposition 3.2. Then: $L$ is net-definable iff $L = PR_{\text{fin}}[\mathcal{N}]$.*

*Proof.* If $L = PR_{\text{fin}}[\mathcal{N}]$ then $L$ is net-definable by definition. Let $L$ be net-definable, that is $L = PR_{\text{fin}}[\mathcal{N}']$ for a certain marked net $\mathcal{N}' = (T', M')$. By Proposition 3.2(a): $PR_{\text{fin}}[\mathcal{N}'] \subseteq PR_{\text{fin}}[\mathcal{N}]$. By Proposition 3.2(b) $\mathcal{N}$ is minimal among all $\mathcal{N}'$ such that $L \subseteq PR_{\text{fin}}[\mathcal{N}']$, hence $T \subseteq T'$ and $M \leq M'$. But $(T \subseteq T' \wedge M \leq M') \Rightarrow PR_{\text{fin}}[\mathcal{N}] \subseteq PR_{\text{fin}}[\mathcal{N}']$. Indeed, if $\alpha \in PR_{\text{fin}}[\mathcal{N}]$ then, by Definition 3.1: $FOLD(\alpha) \subseteq T \wedge \forall \beta \in PREF(\alpha)$: $O_\beta - I_\beta \leq M$. Thus $FOLD(\alpha) \subseteq T' \wedge \forall \beta \in PREF(\alpha)$: $O_\beta - I_\beta \leq M'$, hence $\alpha \in PR_{\text{fin}}[\mathcal{N}']$ implying $PR_{\text{fin}}[\mathcal{N}] \subseteq PR_{\text{fin}}[\mathcal{N}']$. Concluding, $L = PR_{\text{fin}}[\mathcal{N}'] = PR_{\text{fin}}[\mathcal{N}]$, which ends the proof. □

For example, if $L$ contains exactly processes in Figures 2.2–2.4 and all their prefixes, then $L$ is not net-definable, since the set $PR_{\text{fin}}[\mathcal{N}]$ with $\mathcal{N}$ constructed in Proposition 3.2, contains much more processes than $L$ does.

## 4. Monoid of finite and infinite processes

Recall Definition 2.2. and Proposition 2.2. Any atomic process is a singleton set and any finite process is a union of atomic processes with occurrences of places suitably renumbered. It is represented as a catenation (a total operation!) of atomic processes. Given an infinite sequence of atomic processes $\varepsilon_1, \varepsilon_2, ..., \varepsilon_n, ...$, define $\alpha_n = \varepsilon_1 \otimes \varepsilon_2 \otimes ... \otimes \varepsilon_n$. Obviously $\alpha_n \subseteq \alpha_{n+1}$ for each $n \geq 1$, thus we have a non-decreasing chain of sets $\alpha_1 \subseteq \alpha_2 \subseteq ... \subseteq \alpha_n \subseteq ...$ each being a finite process. It is known that such chains have the least upper bounds in the universe of objects the chains are composed of. For the above chain it is $\alpha = \bigcup\limits_{n \geq 1} \alpha_n$ called an infinite join. It is referred to as an *infinite process* provided that $\alpha_n \subset \alpha_{n+1}$ (strict inclusion) for all $n$. Define:

$Y_\alpha = \bigcup\limits_{n \geq 1} Y_{\alpha_n}$ (this is the set of all pairs $\langle x, k \rangle$, *i.e.* process places in $\alpha$)

$I_\alpha(x) = \sum\limits_{n \geq 1} I_{\varepsilon_n}(x)$ (note: this time $I_\alpha(x)$ may equal $\infty$).

For the infinite process in Figure 4.1 there is no generating Place/Transition net, that is such, whose evolution this process might represent.

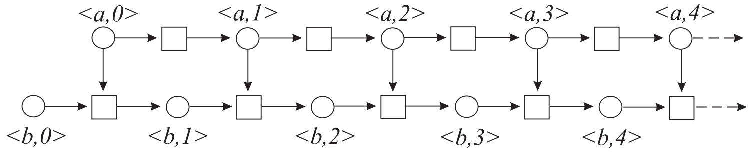


Figure 4.1

Indeed, a minimal (see Prop. 3.2) marked net generating this process is depicted in Figure 4.2.

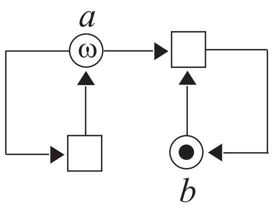


FIGURE 4.2

However, to generate this process, place $a$ would have to be marked with infinitely many tokens, which is not a marking of Place/Transition nets.

Before defining a catenation of arbitrary, also infinite processes, let us assume some denotations:

For processes $\beta, \alpha$ let ${}^\bullet\beta^\bullet = \{x \in \mathbb{X} |\ \exists k\colon\ \langle x, k\rangle \in Y_\beta\}$ (${}^\bullet\beta^\bullet$ is a projection of $Y_\beta$ onto $\mathbb{X}$) and let $\beta^\alpha$ be $\beta$ with each pair $\langle x, k\rangle \in Y_\beta$ replaced by $\langle x, k + I_\alpha(x)\rangle$ provided that $\forall x \in {}^\bullet\beta^\bullet\colon\ I_\alpha(x) < \infty$. This guarantees a resonable catenating: $\alpha \otimes \beta$. Note that $I_\alpha(x) = \sup\{k | \langle x, k\rangle \in Y_\alpha\}$ (the greatest occurrence number of $x$ in $\alpha$ – if it is finite and $\infty$ otherwise). To make catenation $\otimes$ of arbitrary processes always defined, let us introduce a "chaotic" process $\Omega = \mathcal{P}(\mathbb{Y}) \times \mathcal{P}(\mathbb{Y})$ where $\mathbb{Y} = \mathbb{X} \times \mathbb{N}$. The set $\Omega$ is called *chaos* because some of its subsets are processes, but, besides, arbitrary sets of pairs $(A, B)$ with $A \subseteq \mathbb{Y}$, $B \subseteq \mathbb{Y}$, are in $\Omega$. Obviously $Y_\Omega = \mathbb{Y}$ and $I_\Omega(x) = \infty$ for any $x \in \mathbb{X}$. By $PR_\Omega$ is denoted the set of all finite and infinite processes along with the chaos $\Omega$. Note that $\Omega$ plays part of annihilating (zero) process for catenation: $\Omega \otimes \alpha = \alpha \otimes \Omega = \Omega$, *i.e.* it absorbs any process which it is catenated with, since $\alpha \subseteq \Omega$ for each $\alpha$. On the other hand, two non-$\Omega$ processes may result in $\Omega$ when catenated: it happens when the left process contains infinitely many occurrences of a place and this place occurs at least once in the right process. In this way the catenation is extended to arbitrary processes and remains a total operation. Such construct of zero-process $\Omega$ turned out beneficial to some techniques used in a calculus of process languages developed *e.g.* in [6, 7, 9, 10]. So, we come to the following:

**Definition 4.1** (Catenation of arbitrary processes)**.** For $\alpha, \beta \in PR$:

$$\alpha \otimes \beta = \begin{cases} \alpha \cup \beta^\alpha \text{ if } & \forall x \in {}^\bullet\beta^\bullet : \quad I_\alpha(x) < \infty \\ \Omega \text{ else.} \end{cases}$$

**Example 4.1.**

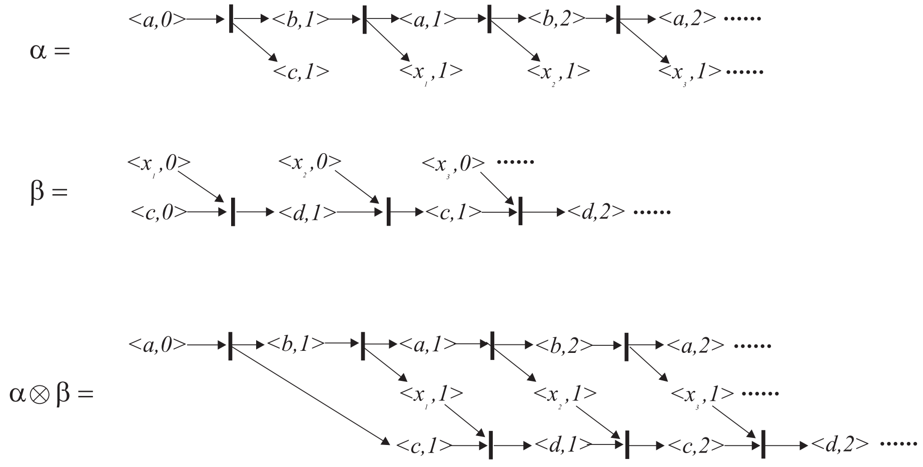


FIGURE 4.3

Note that $\beta \otimes \alpha = \Omega$, because $I_\beta(c) = \infty$ and $c \in {}^\bullet\alpha^\bullet$.

**Theorem 4.1.** *Catenation of arbitrary (finite or infinite) processes is associative.*

*Proof.* Firstly, we show that $\alpha \otimes (\beta \otimes \gamma) = \Omega \Leftrightarrow (\alpha \otimes \beta) \otimes \gamma = \Omega$. It may be assumed that $\alpha \neq \Omega$, $\beta \neq \Omega$, $\gamma \neq \Omega$, since otherwise the equivalence holds trivially. If $\alpha \otimes (\beta \otimes \gamma) = \Omega$ then $\beta \otimes \gamma = \Omega$, which means that $I_\beta(x) = \infty$ for a certain $x \in {}^\bullet\gamma^\bullet$. But then $I_{\beta\otimes\gamma}(x) = \infty$, hence $(\alpha \otimes \beta) \otimes \gamma = \Omega$. The reverse implication is shown analogously. Therefore, let us assume $\alpha \otimes (\beta \otimes \gamma) \neq \Omega$, thus also $(\alpha \otimes \beta) \otimes \gamma \neq \Omega$. Since $\alpha \otimes (\beta \otimes \gamma) = \alpha \cup \beta^\alpha \cup (\gamma^\beta)^\alpha$ and $(\alpha \otimes \beta) \otimes \gamma = \alpha \cup \beta^\alpha \cup \gamma^{\alpha \cup \beta^\alpha}$, what suffices to be verified is $(\gamma^\beta)^\alpha = \gamma^{\alpha \cup \beta^\alpha}$. If a pair $\langle x, k\rangle$ is in $\gamma$ (*i.e.* $\langle x, k\rangle \in Y_\gamma$) then, in effect of shifting $\gamma^\beta$, it transforms into the pair $\langle x, k + I_\beta(x)\rangle$ in $\gamma^\beta$ and, further, into the pair $\langle x, k + I_\beta(x) + I_\alpha(x)\rangle$ in $(\gamma^\beta)^\alpha$. On the other hand, since $\langle x, k\rangle$ converts into $\langle x, k + I_{\alpha\cup\beta^\alpha}(x)\rangle$ in $\gamma^{\alpha\cup\beta^\alpha}$, it remains to check whether $I_{\alpha\cup\beta^\alpha}(x) = I_\beta(x) + I_\alpha(x)$. If $x$ does not occur either in $\alpha$ or $\beta$ then the latter equation holds obviously. Let $x$ occur in $\alpha$ and in $\beta$. By definition of the shift $\beta^\alpha$ we get: $I_{\alpha\cup\beta^\alpha}(x) = \max(I_\alpha(x), I_{\beta^\alpha}(x)) = I_{\beta^\alpha}(x) = I_\beta(x) + I_\alpha(x)$, which ends the proof. $\square$

**Corollary.** *$\mathcal{M}_\Omega = (PR_\Omega, \otimes, \mathbf{1})$ is a monoid.*

With a certain caution for infinite objects, one may state some analogues to Theorems 2.1 and 2.2 for arbitrary processes. Also, a trivial sort of analysis/synthesis for unmarked P/T nets and arbitrary processes may be established as follows. For a process language $A \subseteq PR$ let:
$A^\otimes = \{\alpha \in PR |\ \exists n > 0,\ \alpha_j \in A\ \ (1 \leq j \leq n){:}\ \ \alpha = \alpha_1 \otimes \alpha_2 \otimes ... \otimes \alpha_n\} \cup \{\mathbf{1}\}$

$A^{\circledast} = \{\alpha \in PR |\ \exists \alpha_j \in A\ (1 \le j < \infty)\colon \alpha = \bigcup\limits_{j=1}^{\infty} \alpha_1 \otimes \alpha_2 \otimes \ldots \otimes \alpha_j\}$ (see Prop. 2.2) $A^{\circledast} = A^{\otimes} \cup A^{\circledast}$ and for an unmarked net $T \subseteq \mathbb{T}$ let $PR_0[T]$ denote the set of atomic processes as in Definition 2.1 with $\mathbb{X}$ restricted to net-places occurring in $T$ (see also Th. 3.2). Then the set $PR_\Omega[T]$ of all processes into which $T$ may evolve is $PR_0[T]^{\circledast}$. On the other hand, for a given process language $L \subseteq PR_\Omega$ there exists an unmarked net evolving into processes in $L$ iff $L = (L \cap PR_0)^{\circledast}$ and this net is computed as $T = \bigcup\limits_{\alpha \in L} FOLD(\alpha)$ (*cf.* [10]). However, the analysis/synthesis problems for marked nets and infinite processes require more complex logical means, since in this case the matrices and vectors may contain $\infty$ as its entries. Likewise, dealing with permutations of infinite sequences (recall Th. 2.2) requires stronger logical means. One enters, in this case, considerations of effectivity (computability) of some operations and relations (*e.g.* relation $\overset{*}{\sim}$). Moreover, firing sequence $U$ in Theorem 2.2 would have to be assumed finite, otherwise $UV$ would not be defined. This paper is limited to finite processes for these problems. Process languages for 1-safe (elementary) nets, along with analysis/synthesis, may be found in [6, 7].

## 5. Concluding remarks

Process languages, *i.e.* subsets of $PR_\Omega$ may be specified by various generative mechanisms, not only by nets. Consider, for instance, the set of all finite (for simplicity) processes evoked by the net $T_{\text{prodcons}}$ from Figure 1.3 with initial marking $M = \begin{array}{|c|c|c|c|c|}\hline a & b & c & d & e \\ \hline 1 & 0 & 1 & 0 & 0 \\ \hline\end{array}$ (see example following Th. 3.2). One may construct a (non-regular) grammar $G$ (with $T_{\text{prodcons}}$ as the set of terminal symbols) generating a certain set $\mathcal{L}(G)$ being the intersection of two sets:

(1) $\{u\}\{tu\}^*\{\lambda, t\} \parallel \{w\}\{vw\}^*\{\lambda, v\}$ where $\parallel$ is the shuffle (interleaving) operation on; string languages;

(2) $\{X \in \{t, u, v, w\}^* |\ \underline{Y}(u) \ge \underline{Y}(v)$ for each prefix $Y$ of $X\}$ where $\underline{Y}(u)$ is the number of occurrences of $u$ in the string $Y$.

Each process $\alpha \in PR_{\text{fin}}[T^*_{\text{prodcons}}, M]$ is represented by a string $Z \in \mathcal{L}(G)$ such that $\alpha = \text{pr}(Z)$ and each string from $\mathcal{L}(G)$ represents a process from $PR_{\text{fin}}[T^*_{\text{prodcons}}, M]$ (recall Th. 2.2 but also Th. 2.1). It follows from Theorem 3.2 that for any finite net $T$ with places marked by infinitely many tokens, the set of processes is $PR_0[T]^{\otimes}$, thus specifiable by a regular grammar. Some properties of process languages specified by grammars (in the form of fix-point equations) are in [9, 10].

The analysis/synthesis problems for nets consist in finding transformations between marked nets and a *representation* of corresponding process languages. Results of Section 3 gave a general outline of the solution only: representation of process languages is not limited there to one constructive formalism. For instance, notation for the outcome of analysis (Th. 3.2) was borrowed from semiring $\mathcal{S}_{\text{fin}}$, linear algebra and set calculus. Furthermore, synthesis (Th. 3.3), which requires decision for $\alpha \in L$, was left without constructive specification of the language $L$. A linguistic (or algebraic) constructive solution to the analysis/synthesis problems would consist in providing algorithms of transformation (preserving process languages) between nets and grammars (or fix-point equations) with catenation $\otimes$. This work is in progress. The literature on passing between various system description tools is quite rich – some references are cited in the Introduction. Another direction of development of approach sketched in this paper is search for possibly efficient algorithms for the reachability problem. Theorem 2.2 accounts for reduced set of processes in comparison with the set of firing sequences. This, along with peculiar process construction is expected to provide some useful results in algorithmic verification of reachability, that is of checking $(M, M') \in [[T]]^*$ (see Introduction), by exploitation of the so-called partial order reduction techniques. This issue also enjoys rich literature, it is however outside the scope of this paper, thus not mentioned in the References.

*Acknowledgements.* I am deeply indebted to the anonymous referees whose remarks made the final version of this work essentially improved.

## References

[1] E. Best and C. Fernandez, *Non-sequential Processes, A Petri Net View.* Springer, Berlin, Heidelberg, New York, Tokyo, *EATCS Monogr. Theoret. Comput. Sci.* **13** (1988).

[2] R. Berghammer, B. Karger and C. Ulke, Relation-Algebraic Analysis of Petri Nets with RELVIEW, in *Tools and Algorithms for the Construction and Analysis of Systems, Second Int. Workshop, TACAS'96*, Passau, Germany. Springer-Verlag, *Lecture Notes in Comput. Sci.* **1055** (1996) 49-69.

[3] E. Best and R. Devillers, Sequential and Concurrent Behaviour in Petri Net Theory. *Theoret. Comput. Sci.* **55** (1987) 87-136.

[4] N. Busi and G.M. Pinna, Synthesis with Inhibitor Arcs, in *Proc. of CONCUR'97*, Warsaw, Poland. Springer-Verlag, *Lecture Notes in Comput. Sci.* **1243** (1997) 151-165.

[5] L.A. Castelano, *Beta Processes of C/E Systems*, Advances in Petri Nets. Springer-Verlag, *Lecture Notes in Comput. Sci.* **222** (1985) 83-100.

[6] L. Czaja, Net-Definability of Process Languages. *Fund. Inform.* **37** (1999) 213-223.

[7] L. Czaja, Process Languages and Nets. *Theoret. Comput. Sci.* **238** (2000) 161-181.

[8] L. Czaja and M. Kudlek, Lematy Iteracyjne dla Rownosciowo Definiowalnych Jezykow Procesow (in Polish), in *Proc. of symposium Theoretical Informatics, Methods of Analysis of Incomplete and Distributed Information.* Technical University Bialystok (1999) 8-23.

[9] L. Czaja and M. Kudlek, Rational, Linear and Algebraic Process Languages and Iteration Lemmata. *Fund. Inform.* **43** (2000) 49-60.

[10] L. Czaja and M. Kudlek, $\omega$-Process Languages for Place/Transition Nets. *Fund. Inform.* **47** (2001) 217-229.

[11] P. Darondeau, *Deriving unbounded Petri nets from formal languages.* IRISA, Internal Report No. 1172 (1998).

[12] J. Desel and W. Reisig, *Place/Transition Petri Nets*, in Lecture Notes on Petri Nets 1: Basic Models, edited by R.W. Rozenberg. Springer-Verlag, *Lecture Notes in Comput. Sci.* **1491** (1998) 122-173.

[13] V. Diekert and G. Rozenberg, *The Book of Traces.* World Scientific, Singapore, New Jersey, London, Hong Kong (1995).

[14] J. Esparza and M. Silva, On the Analysis and Synthesis of Free Choice Systems, in *Advances in Petri Nets 1990*. Springer-Verlag, *Lecture Notes in Comput. Sci.* **483** (1991) 243-286.
[15] U. Goltz and W. Reisig, The Non-sequential Behaviour of Petri Nets. *Inform. and Comput.* **57** (1983) 125-147.
[16] R. Gorrieri, *Refinement, Atomicity and Transactions for Process Description Languages*, Ph.D. Thesis TD-2/91. Università degli Studi di Pisa Dipartimento di Informatica (1991).
[17] W.E. Kotov, *Petri Nets* (in Russian). NAUKA, Moscow (1984).
[18] A. Krzywicki, *Processes in Place/Transition Nets with weights* (in Polish), M.Sc. Thesis. Institute of Informatics, Warsaw University (2001).
[19] T. Kuzak, *Sequential Behaviour of Nets with Unbounded Capacity of Places* (in Polish), Ph.D. Thesis. Institute of Computer Science Foundations, Polish Academy of Sciences (1987).
[20] I.A. Lomazova, On Occurrence Net Semantics for Petri Nets with Contacts, in *Proc. of Fundamentals of Computation Theory, 11th International Symposium, FCT'97*, Krakow, Poland. Springer-Verlag, *Lecture Notes in Comput. Sci.* **1279** (1997) 317-328.
[21] A. Mazurkiewicz, *Trace theory*, edited by W. Brauer *et al.*, Petri Nets, Applications and Relationship to other Models of Concurrency. Springer, Berlin-Heidelberg-New York, *Lecture Notes in Comput. Sci.* **255** (1987) 279-324.
[22] A. Mazurkiewicz, *Concurrency, Modularity and Synchronization.* Mathematical Foundations of Computer Science, Porabka–Kozubnik, Poland. Springer-Verlag, *Lecture Notes in Comput. Sci.* **379** (1989) 577-598.
[23] A. Mazurkiewicz, *Introduction to Trace Theory*, in [13], pp. 3-41.
[24] J. Meseguer, U. Montanari and V. Sassone, On the Semantics of Place/Transition Petri Nets. *Math. Struct. Comput. Sci.* **7** (1997) 359-397.
[25] M. Nielsen and G. Winskel, *Trace Structures and other Models of Concurrency*, in [13], pp. 271-305.
[26] E. Ochmanski, Occurrence traces – Processes of elementary net systems, in *Advances in Petri Nets 88*. Springer-Verlag, *Lecture Notes in Comput. Sci.* **340**, 331-342.
[27] E. Ochmanski, *Recognizable Trace Languages*, in [13], pp. 168-204.
[28] M. Pietkiewicz–Koutny, Transition Systems of Elementary Net Systems with Inhibitor Arcs, in *18th International Conference on Application and Theory of Petri Nets*, Toulouse, France. Springer-Verlag, *Lecture Notes in Comput. Sci.* **1248** (1997) 310-327.
[29] W. Reisig, *Petri Nets. An Introduction.* Springer, Berlin-Heidelberg-New York, Tokyo, *EATCS Monogr. Theoret. Comput. Sci.* **4** (1985).
[30] G. Rozenberg, *Behaviour of elementary net systems*, edited by W. Brauer, Petri nets: Central models and their properties; advances in Petri nets; proceedings of an advanced course, Bad Honef. Springer-Verlag, *Lecture Notes in Comput. Sci.* **254** (1986) 8-19.
[31] G. Rozenberg and J. Engelfriet, *Elementary Net Systems*, in Lectures on Petri Nets 1: Basic Models, edited by W. Reisig and G. Rozenberg. Springer-Verlag, *Lecture Notes in Comput. Sci.* **1491** (1998) 12-121.
[32] P.H. Starke, *Petri-Netze. Grundlagen, Anwendungen, Theorie.* VEB Deutscher Verlag der Wissenschaften, Berlin (1980) Polish translation WNT (1987).
[33] J. Winkowski, Behaviours of Concurrent Systems. *Theoret. Comput. Sci.* **12** (1980) 39-60.

Communicated by D. Niwinski.
Received June, 2001. Accepted December, 2002.