Representations of Petri net interactions

Paweł Sobociński

ECS, University of Southampton, UK

Abstract. We introduce a novel compositional algebra of Petri nets, as well as a stateful extension of the calculus of connectors. These two formalisms are shown to have the same expressive power.

Introduction

In part owing to their intuitive graphical representation, Petri nets [28] are often used both in theoretical and applied research to specify systems and visualise their behaviour. On the other hand, process algebras are built around the principle of compositionality: their semantics is given *structurally* so that the behaviour of the whole system is a function of the behaviour of its subsystems. Indeed, Petri nets and process calculi differ in how their underlying semantics is defined: Petri nets via some kind of globally defined transition system of "firing" transitions, and process calculi via an inductively generated (SOS [27]) labelled transition system. As a consequence, the two are associated with different modelling methodologies and reasoning techniques.

There has been much research concentrating on relating the two domains. This paper continues this tradition by showing that a certain class of Petri nets has, in a precise way, the same expressive power as a process calculus.

Technically, we introduce a compositional extension of Condition/Event nets with consume/produce loops. A net is associated with left and right interfaces to which its transitions may connect. Composition of two such nets along a common boundary occurs via a kind of synchronisation of transitions. This notion of compositionality is related to the concept of open nets [4–6].

On the other hand, the process calculus can be considered an extension of (an SOS presentation of) stateless connectors [9] with a very simple notion of state: essentially a one-place buffer. A related extension was considered in [3].

The operations of well-known process algebras have influenced research on Petri nets and various translations have been considered. In the 1990s there was a considerable amount of research that, roughly speaking, related and adapted the operations of the CCS [23] and related calculi to Petri nets. An example of this is the Petri Box calculus [7, 20] and, to a lesser extent, the combinators of Nielsen, Priese and Sassone [26]. More recently, Cerone [11] defined several translations from C/E nets to the Circal process algebra, that like CCS is based on a binary composition and hiding operators. Other recent related work has included endowing Petri nets with labelled transition systems, using techniques and intuitions originating from process calculi, see [21, 24, 29].

Conversely, there has also been considerable work on translating process calculi to Petri nets: representative examples include [10, 12, 14, 31]. Recently [15] suggests a set of operations for open nets to which an SOS semantics is assigned.

The operations of the calculus presented in this paper are fundamentally different to those utilised in the aforementioned literature. Indeed, they are closer in nature to those of tile logic [13] and Span(Graph) [18] than to the operations of CCS. More recently, similar operations have been used by Reo [2], glue for component-based systems [8] and the wire calculus [30]. Indeed, in [17] Span(Graph) is used to capture the state space of P/T nets; this work is close in spirit to the translation from nets to terms given in this paper.

Different representations of the same concept can sometimes serve as an indication of its canonicity. Kleene's theorem [19, 22] is a well-known example: on the one hand graphical structures with a globally defined semantics (finite automata) are shown to have the same expressive power as a language with an inductively-defined semantics (regular expressions).

Structure of the paper. Nets with boundaries are introduced in §1 and the relevant process calculus, for the purposes of this paper dubbed the "Petri calculus", is introduced in §2. The translation from nets to process calculus terms is given in §3. A reverse translation is given in §4. Future work is discussed in §5.

1 Nets

Definition 1 For the purposes of this paper a Petri net is a 4-tuple $N = (P, T, {}^\circ-, -^\circ)$ where[1]:

- $*$ P is a set of *places*;
- $*$ T is a set of *transitions*;
- $*$ ${}^\circ-, -^\circ \colon T \to 2^P$ are functions.

N is *finite* when both P and T are finite sets.

The obvious notion of net homomorphisms $f\colon N \to M$ is a pair of functions $f_T\colon T_N \to T_M$, $f_P\colon P_N \to P_M$ such that ${}^\circ-_N\,;\, 2^{f_P} = f_T\,;\, {}^\circ-_M$ and $-^\circ{}_N\,;\, 2^{f_P} = f_T\,;\, -^\circ{}_M$, where $2^{f_P}(X) = \bigcup_{x\in X}\{f_P(x)\}$. For a transition $t \in T$, ${}^\circ t$ and t° are called, respectively, its *pre-* and *post-sets*. Notice that Definition 1 allows transitions with empty pre- and post-sets; this option, while counterintuitive for ordinary nets, will be necessary for nets with boundaries, introduced in §1.1.

Transitions t, u are *independent* when ${}^\circ t \cap {}^\circ u = \varnothing$ and $t^\circ \cap u^\circ = \varnothing$. Note that this notion of independence is quite liberal and allows so-called contact situations. Moreover, a place p can be both in ${}^\circ t$ and t° for some transition t; some authors refer to this as a consume/produce loop; the notion of *contextual net* [25] is related. A set U of transitions is mutually independent when, for all $t, u \in U$, if $t \neq u$ then t and u are independent. Given a set of transitions U let ${}^\circ U = \bigcup_{u\in U} {}^\circ u$ and $U^\circ = \bigcup_{u\in U} u^\circ$.

[1] In the context of C/E nets some authors call places *conditions* and transitions *events*.

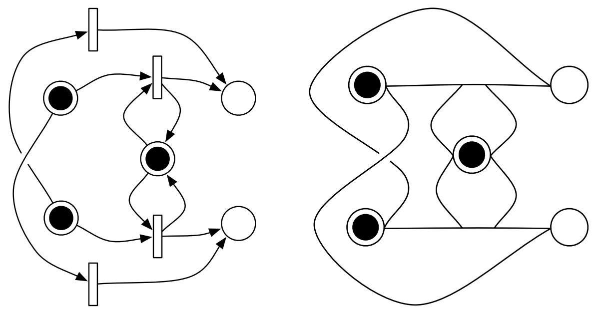


Fig. 1. Traditional and alternative graphical representations of a net.

Definition 2 (Semantics) *Let* $N = (P, T, {}^{\circ}-, -^{\circ})$ *be a net,* $X, Y \subseteq P$ *and* $t \in T$. *Write:*

$$(N, X) \rightarrow_{\{t\}} (N, Y) \quad \stackrel{\text{def}}{=} \quad {}^{\circ}t \subseteq X,\ t^{\circ} \subseteq Y \ \&\ X \backslash {}^{\circ}t = Y \backslash t^{\circ}.$$

For $U \subseteq T$ *a set of mutually independent transitions, write:*

$$(N, X) \rightarrow_{U} (N, Y) \quad \stackrel{\text{def}}{=} \quad {}^{\circ}U \subseteq X,\ U^{\circ} \subseteq Y \ \&\ X \backslash {}^{\circ}U = Y \backslash U^{\circ}.$$

Note that, for any $X \subseteq P$, $(N, X) \xrightarrow{\varnothing} (N, X)$. *States of this transition system will be referred to as* markings *of* N.

The left diagram in Fig. 1 demonstrates the traditional graphical representation of a (marked) net. Places are circles; a marking is represented by the presence or absence of tokens. Each transition $t \in T$ is a rectangle; there are directed edges from each place in ${}^{\circ}t$ to t and from t to each place in t°. This graphical language is a particular way of drawing hypergraphs; the right diagram in Fig. 1 exemplifies another graphical representation, more suitable for representing the notion of nets introduced in this paper. Places are again circles, but each has exactly two *ports*: one on the left and one on the right. Transitions are undirected *links*—each link can connect to any number of ports. Connecting t to the right port p signifies that $p \in {}^{\circ}t$, connecting t to the left port means that $p \in t^{\circ}$. Variants of link graphs have been used to characterise various free monoidal categories: see for instance [1, 16].

1.1 Nets with boundaries

Let $\underline{k}$, $\underline{l}$, $\underline{m}$, $\underline{n}$ range over finite ordinals: $\underline{n} \stackrel{\text{def}}{=} \{0, 1, \ldots, n-1\}$.

Definition 3 *Let* $m, n \in \mathbb{N}$. *A (finite)* net with boundaries $N \colon m \to n$, *is a sextuple* $(P, T, {}^{\circ}-, -^{\circ}, {}^{\bullet}-, -^{\bullet})$ *where:*

- $*$ $(P, T, {}^{\circ}-, -^{\circ})$ *is a finite net;*

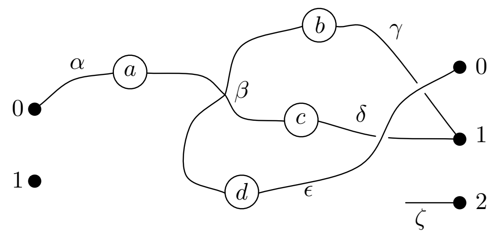


Fig. 2. Representation of a net with boundaries $2 \to 3$. Here $T = \{\alpha,\ \beta,\ \gamma,\ \delta,\ \epsilon,\ \zeta\}$ and $P = \{a,\ b,\ c,\ d\}$. The non-empty values of $^\circ-$ and $-^\circ$ are: $\alpha^\circ = \{a\}$, $^\circ\beta = \{a\}$, $\beta^\circ = \{b, c, d\}$, $^\circ\gamma = \{b\}$, $^\circ\delta = \{c\}$. The non-empty values of $^\bullet-$ and $-^\bullet$ are: $^\bullet\alpha = \{0\}$, $\gamma^\bullet = \{1\}$, $\delta^\bullet = \{1\}$, $\zeta^\bullet = \{2\}$.

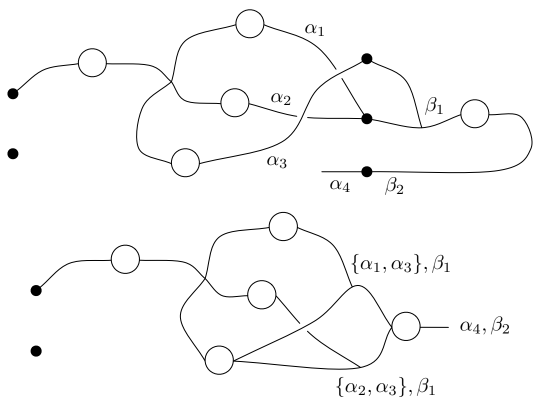


Fig. 3. Illustration of composition of two nets with boundaries.

* $^\bullet-\colon T \to 2^{\underline{m}}$, $-^\bullet\colon T \to 2^{\underline{n}}$ *are functions.*

We refer to m *and* n *as, respectively, the left and right* boundaries *of* N*. An example is pictured in Fig. 2.*

Henceforward we shall usually refer to nets with boundaries as simply nets.

The obvious notion of homomorphism between two nets with equal boundaries extends that of ordinary nets: given nets $N, M\colon m \to n$, $f\colon N \to M$ is a pair of functions $f_T\colon T_N \to T_M$, $f_P\colon P_N \to P_M$ such that $^\circ-_N\ ;\ 2^{f_P} = f_T\ ;\ ^\circ-_M$, $-^\circ{}_N\ ;\ 2^{f_P} = f_T\ ;\ -^\circ{}_M$, $^\bullet-_N = f_T\ ;\ ^\bullet-_M$ and $-^\bullet{}_N = f_T\ ;\ -^\bullet{}_M$. A homomorphism is an isomorphism iff its two components are bijections; we write $N \cong M$ when there is an isomorphism from N to M.

The notion of independence of transitions extends to nets with boundaries in the obvious way: $t, u \in T$ are said to be *independent* when

$$^\circ t \cap {}^\circ u = \varnothing, \quad t^\circ \cap u^\circ = \varnothing, \quad {}^\bullet t \cap {}^\bullet u = \varnothing \quad \text{and} \quad t^\bullet \cap u^\bullet = \varnothing.$$

Let $M : l \to m$ and $N : m \to n$ be nets. In order to define the composition along their shared boundary, we must first introduce the concept of *synchronisation*: a pair (U, V), with $U \subseteq T_M$ and $V \subseteq T_N$ mutually independent sets of transitions such that:

* $U \cup V \neq \varnothing$;
* $U^\bullet = {}^\bullet V$.

The set of synchronisations inherits an ordering from the subset relation, ie $(U',\, V') \subseteq (U,\, V)$ when $U' \subseteq U$ and $V' \subseteq V$. A synchronisation is said to be *minimal* when it is minimal with respect to this order. Let

$$T_{M;N} \stackrel{\text{def}}{=} \{(U,\, V) \mid U \subseteq T_M,\, V \subseteq T_N,\, (U,\, V) \text{ a minimal synchronisation}\}$$

Notice that any transition in M or N not connected to the shared boundary m is a minimal synchronisation in the above sense. Define[2] ${}^\circ-, -^\circ : T_{M;N} \to 2^{P_M+P_N}$ by letting ${}^\circ(U,\, V) = {}^\circ U \cup {}^\circ V$, $(U,\, V)^\circ = U^\circ \cup V^\circ$. Define ${}^\bullet- : T_{M;N} \to 2^{\underline{l}}$ by ${}^\bullet(U,\, V) = {}^\bullet U$ and $-^\bullet : T_{M;N} \to 2^{\underline{n}}$ by $(U,\, V)^\bullet = V^\bullet$. The *composition* of M and N, written $M; N : l \to n$, has:

* $T_{M;N}$ as its set of transitions;
* $P_M + P_N$ as its set of places;
* ${}^\circ-, -^\circ : T_{M;N} \to 2^{P_M+P_N}$, ${}^\bullet- : T_{M;N} \to 2^{\underline{l}}$, $-^\bullet : T_{M;N} \to 2^{\underline{n}}$ as above.

An example of a composition of two nets is illustrated in Fig. 3.

Proposition 4

(i) Let $M, M' : k \to n$ and $N, N' : n \to m$ be nets with $M \cong M'$ and $N \cong N'$. Then $M \,;\, N \cong M' \,;\, N'$
(ii) Let $L : k \to l$, $M : l \to m$, $N : m \to n$ be nets. Then $(L \,;\, M) \,;\, N \cong L \,;\, (M \,;\, N)$

□

We need to define one other binary operation on nets. Given nets $M : k \to l$ and $N : m \to n$, their *tensor product* is, intuitively the net that results from putting the two nets side-by-side. Concretely, $M \otimes N : k + m \to l + n$ has:

* set of transitions $T_M + T_N$;
* set of places $P_M + P_N$;
* ${}^\circ-, -^\circ, {}^\bullet-, -^\bullet$ defined in the obvious way.

1.2 Semantics

Throughout this paper we use two-labelled transition systems. Labels are words in $\{0, 1\}^*$ and are ranged over by α, β. Write $\#\alpha$ for the length of a word α. The intuitive idea is that a transition $p \xrightarrow[\beta]{\alpha} q$ signifies that a system in state p can, in a single step, synchronise with α on its left boundary, β on it right boundary and change its internal state to q.

[2] We use $+$ to denote disjoint union.

Definition 5 (Transitions) For $k,\, l \in \mathbb{N}$, a $(k,\, l)$-*transition* is a two-labelled transition of the form $\xrightarrow[\beta]{\alpha}$ where $\alpha, \beta \in \{0,1\}^*$, $\#\alpha = k$ and $\#\beta = l$. A $(k,\, l)$ labelled transition system ($(k,\, l)-$LTS) is a transition system that consists of $(k,\, l)$-transitions.

Definition 6 (Bisimilarity) A simulation on a $(k,\, l)-$LTS is a relation S on its set of states that satisfies the following: if $(v, w) \in S$ and $v \xrightarrow[\beta]{\alpha} v'$ then $\exists w'$ s.t. $w \xrightarrow[\beta]{\alpha} w'$ and $(v', w') \in S$. A bisimulation is a relation S where both S and S^{-1} are simulations. Bisimilarity is the largest bisimulation relation.

For any $k \in \mathbb{N}$, there is a bijection $\ulcorner - \urcorner : 2^k \to \{0,1\}^k$ with

$$\ulcorner U \urcorner_i \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } i \in U \\ 0 & \text{otherwise} \end{cases} .$$

Definition 7 (Semantics) *Let $N : m \to n$ be a net and $X, Y \subseteq P_N$. Write:*

$$(N,\, X) \xrightarrow[\beta]{\alpha} (N, Y) \quad \stackrel{\text{def}}{=} \quad \exists \text{ mutually independent } U \subseteq T_N \text{ s.t.} \\ (N, X) \to_U (N, Y),\ \alpha = \ulcorner {}^\bullet U \urcorner \ \&\ \beta = \ulcorner U^\bullet \urcorner \quad (1)$$

Notice that $(N, X) \xrightarrow[0^n]{0^m} (N, X)$.

We conclude this section with a brief remark on the relationship between nets with boundaries and open nets [4, 6]. While open nets are based on P/T nets, a similar construction can be carried out for the variant of net given by Definition 1. Composition in open nets is based on a pushout construction in a category of open-net morphisms. It is not difficult to show that this open net composition can be captured by a composition of nets with boundaries. We omit the details here.

2 Petri calculus

Here we give the syntax and the structural operational semantics of a simple process calculus, which, for the purposes of this paper, we shall refer to as the *Petri calculus*. It results, roughly, from adding a one-place buffer to the calculus of stateless connectors [9]. The syntax does not feature any binding nor primitives for recursion.

$$P \ ::= \ \bigcirc \mid \odot \mid \mathsf{I} \mid \mathsf{X} \mid \Delta \mid \nabla \mid \bot \mid \top \mid \Lambda \mid \mathsf{V} \mid \downarrow \mid \uparrow \mid P \otimes P \mid P \,;\, P$$

There is an associated sorting. Sorts are of the form $(k,\, l)$, where $k, l \in \mathbb{N}$. The inference rules are given in Fig. 4. Due to their simplicity, a simple induction confirms uniqueness of sorting: if $\vdash\ P : (k,\, l)$ and $\vdash\ P : (k',\, l')$ then $k = k'$ and $l = l'$. We shall only consider sortable terms.

Structural inference rules for operational semantics are given in Fig. 5. The rule (Refl) guarantees that any term is always capable of "doing nothing"; note

$$\frac{}{\vdash \bigcirc : (1,1)} \quad \frac{}{\vdash \odot : (1,1)} \quad \frac{}{\vdash \mathsf{I} : (1,1)} \quad \frac{}{\vdash \mathsf{X} : (2,2)}$$

$$\frac{}{\vdash \Delta : (1,2)} \quad \frac{}{\vdash \nabla : (2,1)} \quad \frac{}{\vdash \bot : (1,0)} \quad \frac{}{\vdash \top : (0,1)}$$

$$\frac{}{\vdash \wedge : (1,2)} \quad \frac{}{\vdash \vee : (2,1)} \quad \frac{}{\vdash \downarrow : (1,0)} \quad \frac{}{\vdash \uparrow : (0,1)}$$

$$\frac{\vdash P : (k,l) \quad \vdash R : (m,n)}{\vdash P \otimes R : (k{+}m,\, l{+}n)} \quad \frac{\vdash P : (k,n) \quad \vdash R : (n,l)}{\vdash P ; R : (k,l)}$$

Fig. 4. Sort inference rules.

$$\frac{}{\bigcirc \xrightarrow[0]{1} \odot}\ (\textsc{TkI}) \quad \frac{}{\odot \xrightarrow[1]{0} \bigcirc}\ (\textsc{TkO1}) \quad \frac{}{\odot \xrightarrow[1]{1} \odot}\ (\textsc{TkO2}) \quad \frac{}{\mathsf{I} \xrightarrow[1]{1} \mathsf{I}}\ (\textsc{Id}) \quad \frac{a,b \in \{0,1\}}{\mathsf{X} \xrightarrow[ba]{ab} \mathsf{X}}\ (\textsc{Tw})$$

$$\frac{}{\Delta \xrightarrow[11]{1} \Delta}\ (\Delta) \quad \frac{}{\nabla \xrightarrow[1]{11} \nabla}\ (\nabla) \quad \frac{}{\bot \xrightarrow{1} \bot}\ (\bot) \quad \frac{}{\top \xrightarrow[1]{} \top}\ (\top) \quad \frac{(a \in \{0,1\})}{\wedge \xrightarrow[(1-a)a]{1} \wedge}\ (\wedge a) \quad \frac{(a \in \{0,1\})}{\vee \xrightarrow[1]{(1-a)a} \vee}\ (\vee a)$$

$$\frac{P \xrightarrow[\mathbf{c}]{\mathbf{a}} Q \quad R \xrightarrow[\mathbf{b}]{\mathbf{c}} S}{P;R \xrightarrow[\mathbf{b}]{\mathbf{a}} Q;S}\ (\textsc{Cut}) \quad \frac{P \xrightarrow[\mathbf{b}]{\mathbf{a}} Q \quad R \xrightarrow[\mathbf{d}]{\mathbf{c}} S}{P \otimes R \xrightarrow[\mathbf{bd}]{\mathbf{ac}} Q \otimes S}\ (\textsc{Ten}) \quad \frac{P : (k,l)}{P \xrightarrow[0^l]{0^k} P}\ (\textsc{Refl})$$

Fig. 5. Structural rules for operational semantics.

that this is the only rule that applies to $\downarrow$ and $\uparrow$. Each of the rules $(\wedge a)$ and $(\vee a)$ actually represent two rules, one for $a = 0$ and one for $a = 1$.

Bisimilarity on the transition system obtained via the inference rules in Fig. 5 is a congruence. This is important, because it allows us to replace subterms with bisimilar subterms without affecting the behaviour of the overall term. This fact will be relied upon in several proofs.

Proposition 8 *If $P \sim P'$ then, for any R:*

(i) $(P \,;\, R) \sim (P' \,;\, R)$;
(ii) $(R \,;\, P) \sim (R \,;\, P)$;
(iii) $(P \otimes R) \sim (P' \otimes R)$;
(iv) $(R \otimes P) \sim (R \otimes P')$.

□

A *process* is a bisimulation equivalence class of a term. We write $[t] : (m, n)$ for the process that contains $t : (m, n)$.

2.1 Circuit diagrams

In subsequent sections it will often be convenient to use a graphical language for terms in the Petri calculus. Diagrams in the language will be referred to as *circuit diagrams*. We shall be careful, when drawing diagrams, to make sure that each diagram can be converted to a syntactic expression by "scanning" the diagram from left to right. The following result justifies the usage.

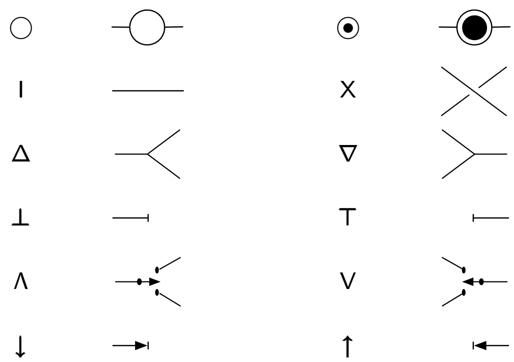


Fig. 6. Circuit diagram components.

Lemma 9

(i) Let $P : (k, l)$, $Q : (l, m)$, $R : (m, n)$. Then

$$(P \,;\, Q) \,;\, R \sim P \,;\, (Q \,;\, R);$$

(ii) Let $P : (k, l)$, $Q : (m, n)$, $R : (t, u)$. Then

$$(P \otimes Q) \otimes R \sim P \otimes (Q \otimes R);$$

(iii) Let $P : (k, l)$, $Q : (l, m)$, $R : (n, t)$, $S : (t, u)$. Then

$$(P \,;\, Q) \otimes (R \,;\, S) \sim (P \otimes R) \,;\, (Q \otimes S).$$

Proof. Straightforward, using the inductive presentation of the operational semantics. □

Each of the language constants is represented by a circuit component listed in Fig. 6. For the translation of §3 we need to construct four additional kinds of compound terms, for each $n > 0$:

$$\mathsf{I}_n : (n,\, n) \quad d_n : (0,\, 2n) \quad e_n : (2n,\, 0) \quad \Delta_n : (n,\, 2n) \quad \nabla_n : (2n,\, n)$$

with operational semantics characterised by:

$$\frac{\alpha\in\{0,1\}^n}{\mathsf{I}_n \xrightarrow[\alpha]{\alpha} \mathsf{I}_n} \quad \frac{\alpha\in\{0,1\}^n}{d_n \xrightarrow[\alpha\alpha]{} d_n} \quad \frac{\alpha\in\{0,1\}^n}{e_n \xrightarrow{\alpha\alpha} e_n} \quad \frac{\alpha\in\{0,1\}^n}{\Delta_n \xrightarrow[\alpha\alpha]{\alpha} \Delta_n} \quad \frac{\alpha\in\{0,1\}^n}{\nabla_n \xrightarrow[\alpha]{\alpha\alpha} \nabla_n} \tag{2}$$

First, $\mathsf{I}_n = \bigotimes_n \mathsf{I}$. Now because d_n and e_n, as well as Δ_n and ∇_n are symmetric, here we only construct d_n and Δ_n. Each is defined recursively:

$$d_1 = \mathsf{T} \,;\, \Delta \qquad d_{n+1} = d_n \,;\, (\mathsf{I}_n \otimes d_1 \otimes \mathsf{I}_n); (\mathsf{I}_{n+1} \otimes \mathsf{X}_n)$$

$$\mathit{\Delta}_1 = \mathsf{\Delta} \qquad \mathit{\Delta}_{n+1} = (\mathsf{\Delta} \otimes \mathit{\Delta}_n) \,;\, (\mathsf{I} \otimes \mathsf{X}_n \otimes \mathsf{I}_n)$$

where also $\mathsf{X}_n : (n+1,\, n+1)$ is defined recursively:

$$\mathsf{X}_1 = \mathsf{X} \qquad \mathsf{X}_{n+1} = (\mathsf{X}_n \otimes \mathsf{I}) \,;\, (\mathsf{I}_n \otimes \mathsf{X}).$$

An easy induction on the derivation of a transition confirms that these construction produce terms whose semantics is characterised by (2).

2.2 Relational forms

For $\theta \in \{\mathsf{X}, \mathsf{\Delta}, \mathsf{\nabla}, \perp, \top, \wedge, \vee, \downarrow, \uparrow\}$ let T_θ denote the set of terms generated by the following grammar:

$$T_\theta \;::=\; \theta \mid \mathsf{I} \mid T_\theta \otimes T_\theta \mid T_\theta \,;\, T_\theta.$$

We shall use t_θ to range over terms of T_θ. We now identify two classes of terms of the Petri calculus: the *relational forms*.

Definition 10 A term $t : (k,\, l)$ is in *right relational form* when

$$t = t_\perp \,;\, t_{\mathsf{\Delta}} \,;\, t_{\mathsf{X}} \,;\, t_\vee \,;\, t_\uparrow.$$

Dually, t is said to be in *left relational form* when

$$t = t_\downarrow \,;\, t_\wedge \,;\, t_{\mathsf{X}} \,;\, t_{\mathsf{\nabla}} \,;\, t_\top.$$

The following result spells out the significance of the relational forms.

Lemma 11 *For each function $f \colon \underline{k} \to 2^{\underline{l}}$ there exists a term $\rho_f : (k,\, l)$ in right relational form, the dynamics of which are characterised by the following:*

$$\frac{}{\rho_f \xrightarrow[\ulcorner V \urcorner]{\ulcorner U \urcorner} \rho_f} \;\Leftrightarrow\; U \subseteq \underline{k} \text{ s. t. } \forall u, v \in U.\; u \neq v \Rightarrow f(u) \cap f(v) = \varnothing \;\&\; V = f(U)$$

The symmetric result holds for functions $f \colon \underline{k} \to 2^{\underline{l}}$ and terms $t : (l,\, k)$ in left relational form. Write $\lambda_f : (l,\, k)$ for any term in left relational form that corresponds to f in the above sense.

Proof. Any function $f : \underline{k} \to 2^{\underline{l}}$ induces a triple $(\underline{m},\, l_f \colon \underline{m} \to \underline{k},\, r_f \colon \underline{m} \to \underline{l})$ where l_f and r_f are jointly injective, ie the function $(l_f, r_f) \colon \underline{m} \to \underline{k} \times \underline{l}$ is injective, and $f(i) = \bigcup_{j \in l_f^{-1}(i)} r_f(j)$ where $l_f^{-1}(i) = \{j \mid l_f(j) = i\}$. Any two such triples are isomorphic as spans of functions. It is not difficult to verify that any function $l_f \colon \underline{m} \to \underline{k}$ gives rise to a term t_{l_f} of the form $t_\perp \,;\, t_{\mathsf{\Delta}} \,;\, t_{\mathsf{X}}$, the semantics of which are characterised by $t_{l_f} \xrightarrow[\ulcorner l_f^{-1}(U) \urcorner]{\ulcorner U \urcorner} t_{l_f}$ for any $U \in \underline{k}$ where for all $u, v \in U$, $l_f^{-1}(u) \cap l_f^{-1}(v) = \varnothing$. Also, any function $r_f \colon \underline{m} \to \underline{l}$ gives rise to a term t_{r_f} of the form $t_{\mathsf{X}} \,;\, t_\vee \,;\, t_\downarrow$, the semantics of which are $t_{r_f} \xrightarrow[\ulcorner W \urcorner]{\ulcorner V \urcorner} t_{r_f}$ where $\forall w \in W$ there exists unique $v \in V$ such that $r_f(v) = w$. It thus suffices to let $\rho_f = t_{l_f} \,;\, t_{r_f}$. $\square$

A simple example is given in Fig. 7. Note that not all terms $t : (k,\, l)$ in right relational form are bisimilar to ρ_f for some $f \colon \underline{k} \to 2^{\underline{l}}$; a simple counterexample is $\mathsf{\Delta} \,;\, \vee : (1,\, 1)$.

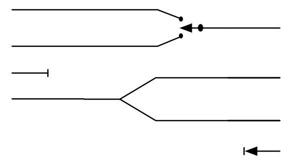


Fig. 7. Right relational form of $f\colon \underline{4} \to 2^{\underline{4}}$ defined $f(0), f(1) = \{0\}$, $f(2) = \varnothing$ and $f(3) = \{1, 2\}$.

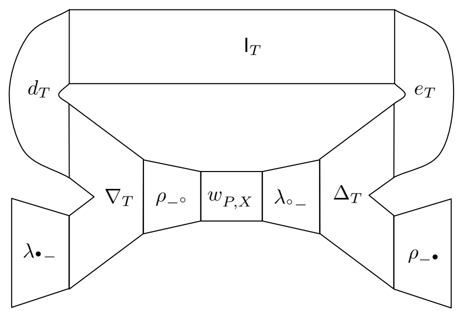


Fig. 8. Diagrammatic representation of the translation from a marked net to a term.

3 Translating nets to Petri calculus terms

Here we present a translation from nets with boundaries, defined in §1, to the process calculus defined in §2. Let $N\colon m \to n = (P, T, {}^\circ-, -^\circ, {}^\bullet-, -^\bullet)$ be a finite net with boundary and $X \subseteq P$ a marking. Assume, without loss of generality, that $P = \underline{p}$ and $T = \underline{t}$ for some $p, t \in \mathbb{N}$. Let

$$w_{P,\,X} : (p,\, p) \overset{\text{def}}{=} \bigotimes_{i<p} m_i \quad \text{where} \quad m_i \overset{\text{def}}{=} \begin{cases} \odot & \text{if } i \in X \\ \bigcirc & \text{otherwise} \end{cases}$$

The following technical result will be useful for showing that the encoding of this section is correct.

Lemma 12 $w_{P,X} \xrightarrow[\ulcorner W \urcorner]{\ulcorner Z \urcorner} Q$ *iff* $Q = w_{P,Y}$*,* $W \subseteq X$*,* $Z \subseteq Y$ *and* $X \backslash W = Y \backslash Z$*.*

Proof. Examination of rules (TkI), (TkO1) and (TKO2), together with the rule (Ten). □

The translation of N can now be expressed as:

$$T_{N,X} \overset{\text{def}}{=} (d_T \otimes \lambda_{\bullet-})\,;(\mathsf{I}_T \otimes (\nabla_T\,;\rho_{-\circ}\,;w_{P,X}\,;\lambda_{\circ-}\,;\Delta_T));(e_T \otimes \rho_{-\bullet}).$$

A circuit diagram representation of the above term is illustrated in Fig. 8.

The encoding preserves and reflects semantics in a very tight manner, as shown by the following.

Theorem 13 *Let N be a finite net. The following hold:*

(i) if $(N,X) \xrightarrow[\beta]{\alpha} (N,Y)$ then $T_{N,X} \xrightarrow[\beta]{\alpha} T_{N,Y}$;
(ii) conversely, if $T_{N,X} \xrightarrow[\beta]{\alpha} Q$ then there exists Y such that $Q = T_{N,Y}$ and $(N,X) \xrightarrow[\beta]{\alpha} (N,Y)$.

Proof. (i) If $(N,X) \xrightarrow[\beta]{\alpha} (N,Y)$ then there exists a set $U \subseteq \underline{t}$ of mutually independent transitions such that $(N,X) \rightarrow_U (N,Y)$, with $\alpha = \ulcorner {}^\bullet U \urcorner$ and $\beta = \ulcorner U^\bullet \urcorner$. Using the conclusion of Lemma 12, we have

$$w_{P,X} \xrightarrow[\ulcorner {}^\circ U \urcorner]{\ulcorner U^\circ \urcorner} w_{P,Y}.$$

Now, using the conclusion of Lemma 11 and (Cut) we obtain transition

$$\rho_{-\circ} \; ; w_{P,X} \; ; \lambda_{\circ -} \xrightarrow[\ulcorner U \urcorner]{\ulcorner U \urcorner} \rho_{-\circ} \; ; w_{P,Y} \; ; \lambda_{\circ -}$$

and subsequently

$$\nabla_T \; ; \rho_{-\circ} \; ; w_{P,X} \; ; \lambda_{\circ -} \; ; \Delta_T \xrightarrow[\ulcorner U \urcorner \ulcorner U \urcorner]{\ulcorner U \urcorner \ulcorner U \urcorner} \nabla_T \; ; \rho_{-\circ} \; ; w_{P,Y} \; ; \lambda_{\circ -} \; ; \Delta_T$$

Certainly $\mathsf{I}_T \xrightarrow[\ulcorner U \urcorner]{\ulcorner U \urcorner} \mathsf{I}_T$, thus using the semantics of d_T and e_T we obtain:

$$T_{N,X} \xrightarrow[\ulcorner U^\bullet \urcorner]{\ulcorner {}^\bullet U \urcorner} T_{N,Y}$$

as required.
(ii) If $T_{N,X} \xrightarrow[\beta]{\alpha} Q$ then $Q = (d_T \otimes \lambda_{\bullet -}) \; ; Q_1 \; ; (e_T \otimes \rho_{-\bullet})$ and

$$I_T \otimes (\nabla_T \; ; \rho_{-\circ} \; ; w_{P,X} \; ; \lambda_{\circ -} \; ; \Delta_T) \xrightarrow[\ulcorner U' \urcorner \ulcorner U' \urcorner \ulcorner V' \urcorner]{\ulcorner U \urcorner \ulcorner U \urcorner \ulcorner V \urcorner} Q_1$$

For some $U, V, U', V' \subseteq \underline{t}$ with $\alpha = \ulcorner {}^\bullet V \urcorner$ and $\beta = \ulcorner V'^\bullet \urcorner$. The structure of (Ten) and the semantics of I_T imply that $U = U'$ and $Q_1 = \mathsf{I}_T \otimes Q_2$ with

$$\nabla_T \; ; \rho_{-\circ} \; ; w_{P,X} \; ; \lambda_{\circ -} \; ; \Delta_T \xrightarrow[\ulcorner U \urcorner \ulcorner V' \urcorner]{\ulcorner U \urcorner \ulcorner V \urcorner} Q_2$$

Now the semantics of Δ_T implies that $U = V$ and conversely, the semantics of ∇_T that $U = V'$, moreover $Q_2 = \nabla_T \; ; Q_3 \; ; \delta_T$ with

$$\rho_{-\circ} \; ; w_{P,X} \; ; \lambda_{\circ -} \xrightarrow[\ulcorner U \urcorner]{\ulcorner U \urcorner} Q_3$$

Finally, using the conclusion of Lemma 11, we obtain $Q_3 = \rho_{-\circ} \; ; Q_4 \; ; \lambda_{\circ -}$ and

$$w_{P,X} \xrightarrow[\ulcorner {}^\circ U \urcorner]{\ulcorner U^\circ \urcorner} Q_4$$

In particular, we obtain that $Q_4 = w_{P,Y}$ and $(N,X) \xrightarrow[\beta]{\alpha} (N,Y)$. □

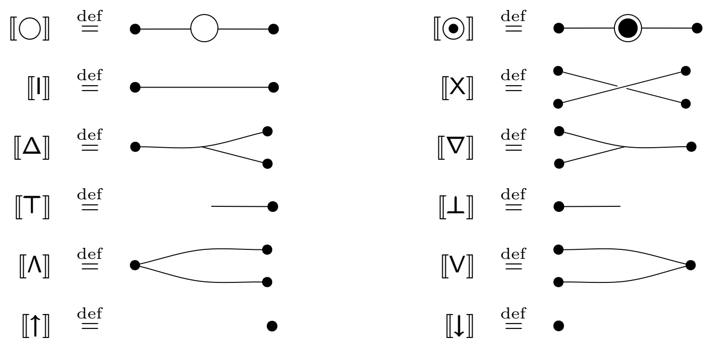
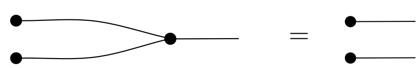


Fig. 9. Translation from calculus constants to nets with marking.

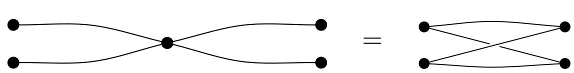
4 Translating Petri calculus terms to nets

Each of the constants of the Petri calculus has a corresponding net with the same semantics: this translation is given in Fig. 9. The naive way of extending this translation to all terms would then be to let $[\![t_1 \,;\, t_2]\!] = [\![t_1]\!] \,;\, [\![t_2]\!]$ and $[\![t_1 \otimes t_2]\!] = [\![t_1]\!] \otimes [\![t_2]\!]$. The naive translation does not reflect behaviour, essentially because of three problematic compositions that involve $\wedge$ and/or $\vee$. First, consider the net that would result from translating the term $\vee \,;\, \bot : (2, 0)$:



According to the inductive system in Fig. 5, the non-trivial transitions of the operational semantics of $\vee \,;\, \bot$ are: $\vee \,;\, \bot \xrightarrow{10} \vee \,;\, \bot$ and $\vee \,;\, \bot \xrightarrow{01} \vee \,;\, \bot$. Now $[\![\vee \,;\, \bot]\!]$ has the above transitions, but also an extra transition: $[\![\vee \,;\, \bot]\!] \xrightarrow{11} [\![\vee \,;\, \bot]\!]$. The second problematic composition is $\top \,;\, \wedge$, which is symmetric to the above situation.

The third and final problematic composition amongst constants arises when translating the term $\vee \,;\, \wedge : (2, 2)$. Here the net composition of the translated components is:



Now the non-trivial derivable transitions are

$$(\vee \,;\, \wedge) \xrightarrow[01]{01} (\vee \,;\, \wedge),\ (\vee \,;\, \wedge) \xrightarrow[10]{10} (\vee \,;\, \wedge),\ (\vee \,;\, \wedge) \xrightarrow[10]{01} (\vee \,;\, \wedge),\ (\vee \,;\, \wedge) \xrightarrow[01]{10} (\vee \,;\, \wedge).$$

Again, the encoding introduces an additional transition

$$[\![\vee \,;\, \wedge]\!] \xrightarrow[11]{11} [\![\vee \,;\, \wedge]\!].$$

The solution, then, is to first transform each term t into a bisimilar term t' in a form which allows compositional translation into a bisimilar net $N_{t'}$.

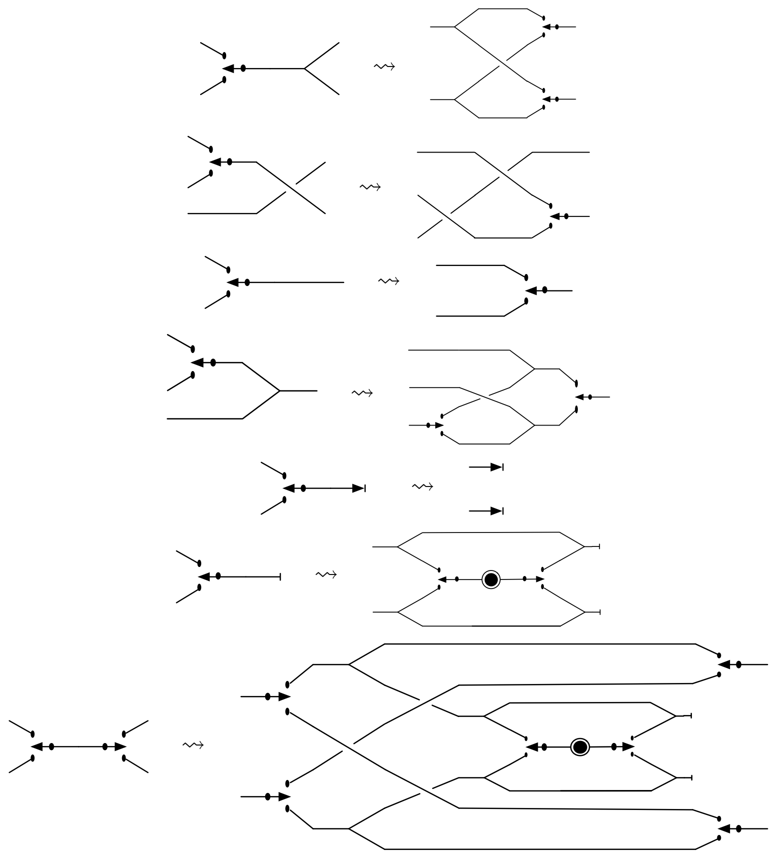

(3)
(4)
(5)
(6)
(7)
(8)
(9)

Fig. 10. Rewriting system for $\vee$.

The initial transformation is best understood via the circuit diagram representation of a term, the soundness of which is ensured by Lemma 9. We say that a term is in *composable form* when, in its circuit diagram:

(i) any occurrence of $\vee$ is connected on the right to either the right boundary, another occurrence of $\vee$, $\bigcirc$ or $\odot$;
(ii) any occurrence of $\wedge$ is connected on the left to either the left boundary, another occurrence of $\wedge$, $\bigcirc$ or $\odot$;

If a term t can can be transformed into the above form then it follows that it can be written as $t_1 = t_\wedge \,;\, t_2 \,;\, t_\vee$, where in t_2 any occurrence of $\wedge$ and $\vee$ is within a subterm of the form $t_\vee \,;\, \bigcirc \,;\, t_\wedge$ (*), or $t_\vee \,;\, \odot \,;\, t_\wedge$ (**). Terms of the form (*) and (**) translate into correct nets, by a case straightforward analysis, the translation can be continued compositionally to obtain a net N_{t_1} with marking X_{t_1} such that $(N_{t_1}, X_{t_1}) \sim t_1$.

Theorem 14 *For each term t there exists a net N_t such that $t \sim N_t$.*

Proof. By the above reasoning, it suffices to show that a term can be transformed into composable form. For this we apply transformations to individual occurrences of $\vee$ and $\wedge$ until the requirements of composable form are met. The rules for $\vee$ are given in Fig. 10. Rules (8) and (9) deal with $\vee$'s problematic compositions. The other rules "push $\vee$ to the right". The complete rewriting system is obtained by including the symmetric versions of (3), (4), (5), (6), (7) and (8) for $\wedge$. □

5 Conclusion and future work

We showed that the class of nets with boundaries has the same expressiveness as a simple process calculus with operations that are fundamentally different from those of CCS, but closely related to operations of coordination languages. As future work it will be interesting to capture the expressive power of other classes of nets, for instance P/T nets with boundaries, with extensions of the process calculus presented here.

Acknowledgment. The author thanks Jennifer Lantair and the anonymous referees for helpful suggestions.

References

1. S. Abramsky. Abstract scalars, loops and free traced and strongly compact closed categories. In *Algebra and coalgebra in computer science (CALCO '05)*, volume 3629 of *LNCS*, pages 1–31. Springer, 2005.
2. F. Arbab. Reo: a channel-based coordination model for component composition. *Math. Struct. Comp. Sci.*, 14(3):1–38, 2004.
3. F. Arbab, R. Bruni, D. Clarke, I. Lanese, and U. Montanari. Tiles for Reo. In *Algebraic Development Techniques (WADT '08)*, volume 5486 of *LNCS*. Springer, 2008.
4. P. Baldan, A. Corradini, H. Ehrig, and R. Heckel. Compositional modelling of reactive systems using open nets. In *Concurrency Theory (CONCUR '01)*, volume 2154 of *LNCS*, pages 502–518, 2001.
5. P. Baldan, A. Corradini, H. Ehrig, and R. Heckel. Compositional semantics for open Petri nets based on deterministic processes. *Math. Struct. Comp. Sci.*, 15(1):1–35, 2005.
6. P. Baldan, A. Corradini, H. Ehrig, R. Heckel, and B. König. Bisimilarity and behaviour-preserving reconfigurations of Petri nets. *Log. Meth. Comput. Sci.*, 4(4):1–41, 2008.
7. E. Best, R. Devillers, and J. G. Hall. The box calculus: A new causal algebra with multi-labelled communication. In *Advances in Petri Nets 1992*, volume 609 of *LNCS*, pages 21–69. Springer, 1992.
8. S. Bliudze and J. Sifakis. A notion of glue expressiveness for component-based systems. In *Concurrency Theory (CONCUR '08)*, volume 5201 of *LNCS*, pages 508–522. Springer, 2008.
9. R. Bruni, I. Lanese, and U. Montanari. A basic algebra of stateless connectors. *Theor. Comput. Sci.*, 366:98–120, 2006.

10. N. Busi and R. Gorrieri. A Petri net semantics for the π-calculus. In *Concurrency theory (CONCUR '95)*, volume 962, pages 145–159. Springer, 1995.
11. A. Cerone. Implementing Condition/Event nets in the circal process algebra. In *Fundamental Approaches to Software Engineering (FASE '02)*, volume 2306 of *LNCS*, pages 49–63. Springer, 2002.
12. P. Degano, R. D. Nicola, and U. Montanari. A distributed operational semantics for CCS based on C/E systems. *Acta Inform.*, 26:59–91, 1988.
13. F. Gadducci and U. Montanari. The tile model. In *Proof, Language and Interaction: Essays in Honour of Robin Milner*, pages 133–166. MIT Press, 2000.
14. U. Goltz. CCS and Petri nets. In *Semantics of Systems of Concurrent Processes*, volume 469 of *LNCS*. Springer, 1990.
15. J. F. Groote and M. Voorhoeve. Operational semantics for Petri net components. *Theor. Comput. Sci.*, 379(1-2):1–19, 2007.
16. A. Joyal and R. Street. The geometry of tensor calculus, I. *Adv. Math.*, 88:55–112, 1991.
17. P. Katis, N. Sabadini, and R. F. C. Walters. Representing P/T nets in Span(Graph). In *Algebraic Methodology and Software Technology (AMAST '97)*, number 1349 in LNCS, pages 307–321. Springer, 1997.
18. P. Katis, N. Sabadini, and R. F. C. Walters. Span(Graph): an algebra of transition systems. In *Algebraic Methodology and Software Technology (AMAST '97)*, volume 1349 of *LNCS*, pages 322–336. Springer, 1997.
19. S. C. Kleene. Representation of events in nerve nets and finite automata. In *Automata Studies*, pages 3–41. Princeton University Press, 1956.
20. M. Koutny, J. Esparza, and E. Best. Operational semantics for the Petri box calculus. In *Concurrency Theory (CONCUR '94)*, volume 836 of *LNCS*, pages 210–225. Springer, 1994.
21. J. J. Leifer and R. Milner. Transition systems, link graphs and Petri nets. *Math. Struct. Comp. Sci.*, 16(6):989–1047, 2006.
22. R. McNaughton and H. Yamada. Regular expressions and state graphs for automata. *IEEE Trans. Electronic Computers*, 9:39–47, 1960.
23. R. Milner. *A Calculus of Communicating Systems*, volume 92 of *LNCS*. Springer, 1980.
24. R. Milner. Bigraphs for Petri nets. In *Lectures on Concurrency and Petri Nets*, volume 3098 of *LNCS*, pages 686–701, 2003.
25. U. Montanari and F. Rossi. Contextual nets. *Acta Inform.*, 32(6):545–596, 1995.
26. M. Nielsen, L. Priese, and V. Sassone. Characterizing behavioural congruences for Petri nets. In *Concurrency Theory (CONCUR '95)*, volume 962 of *LNCS*, pages 175–189. Springer, 1995.
27. G. D. Plotkin. A structural approach to operational semantics. *J. Logic Algebr. Progr.*, 60-61:17–139, 2004. Originally appeared as Technical Report DAIMI FN-19, University of Aarhus, 1981.
28. W. Reisig. *Petri nets: an introduction.* EATCS Monographs on Theoretical Computer Science. Springer Verlag, 1985.
29. V. Sassone and P. Sobociński. A congruence for Petri nets. In *Petri Nets and Graph Transformation (PNGT '04)*, volume 127 of *ENTCS*, pages 107–120, 2005.
30. P. Sobociński. A non-interleaving process calculus for multi-party synchronisation. In *Interaction and Concurrency (ICE '09)*, volume 12 of *EPTCS*, 2009.
31. R. van Glabbeek and F. Vaandrager. Petri net models for algebraic theories of concurrency. In *PARLE, Parallel Architectures and Languages Europe, Volume II*, volume 259 of *LNCS*. Springer, 1987.