Petri Nets and Other Models of Concurrency

Mogens Nielsen and Vladimiro Sassone

ABSTRACT. This paper retraces, collects, and summarises contributions of the authors — in collaboration with others — on the theme of Petri nets and their categorical relationships to other models of concurrency.

CONTENTS

1991 *Mathematics Subject Classification.* Primary 68Q55, 68Q10, 68Q05.

Key words and phrases. Semantics and Models of Concurrency, Noninterleaving, Petri Nets.

Introduction

Concurrency theory is based on a number of different formal models of computation, with *Petri nets* [**66, 67**], or just *nets*, as a prominent example. Other models include the event structures of Winskel [**94**], the trace structures of Mazurkiewicz [**43**], the asynchronous and the concurrent transition systems of Bednarczyk [**4**], Shields [**83**] and Stark [**84**], just to name a few. Similarly, concurrency deals with an abundance of notions for behavioural equivalence, with the bisimulation of Milner [**50**], trace equivalence of Hoare [**30**], and pomset equivalence of Pratt [**69**] as prime examples.

During the past decade, attempts have been made in order to understand the relationships between the confusingly many different concepts within concurrency theory, and many of these are based on the language of *category theory*. Our main goal in this paper is to survey some of the main ideas behind this categorical approach to concurrency, and at the same time to present some particular categorical results for nets.

The first part of our paper is devoted to some categorical results on the *behaviour* of nets and their *relations* with other models, whereas the second part focuses on a categorical approach to the algebraic *structure* of net processes. In our presentation we have chosen to treat (only) three different classes of net systems: the *elementary net systems* of Thiagarajan [**87**], the *semiweighted net systems* [**48**], and *place/transition systems* [**72**], but the approaches presented here are, we claim, applicable to any class of net systems.

Let us start by a few general comments on the role of category theory in our treatment of the behaviour of net systems. Fist of all, how do we relate nets to other models for concurrency? Any model for concurrency is meant to model the behaviour of distributed systems at a certain level of abstraction, focusing on certain aspects of the behaviour, deliberately abstracting from others. Here, we shall attempt to classify models according to their 'level of abstraction', and in stating and proving such relationships we shall use the language of category theory — in particular the notion of *adjunction*. In many contexts this has proven to be a convenient language succinctly expressing such relationships, abstracting away from the details of the often very different mathematical formalisms of the individual models. As the reader will see, nets relate nicely to most of our chosen models, in the sense that one of the models '*embed*' into the other; 'embedding' is formalised here by special adjunctions called *coreflections*.

As the reader will see, adjunctions and hence coreflections between two categories of models, M_0 and M_1, consists of ways of translating from one model to the other, satisfying certain properties. Formally, an adjunction is expressed in terms of two functors $\mathsf{L}\colon \mathsf{M}_0 \to \mathsf{M}_1$ and $\mathsf{R}\colon \mathsf{M}_1 \to \mathsf{M}_0$, and a coreflection is a way of saying that M_0 embeds into M_1 — with L telling us how to *embed* M_0 into M_1, and R how to *project* M_1 onto M_0. This will be our formal way of saying that 'M_0 is an *abstract* version of M_1'.

In Part 1, we shall show examples of such embeddings between our classes of net systems and those of event structures, trace structures, domains, and transition systems. These result are part of a greater picture of relationships between models for concurrency, see e.g. [**97, 80, 79**]. We have chosen to present a few results in some detail, at the expense of the range of models covered.

It is important to notice that all our categories are based on notions of morphisms which should be thought of as '*simulations*'. This view is supported by the fact that

they are all '*behaviour respecting*', as formalised in concrete theorems. This means that the existence of a morphism may be seen as a demonstration that one object (implementation) satisfies another (specification), and hence morphisms may also play a role in formal verification.

Once adjunctions are established between models, one may start comparing and transferring behavioural concepts from one model to another, formally via the adjoints L and R. In the final section of Part 1, we shall present on such example based on [**35, 63**], introducing a general way of understanding Milner's seminal notion of bisimulation [**50**] across a range of different models, including net systems.

It must be noted that there is more to the categorical view of models than we present here. For instance, universal constructs like products and coproducts serve as basis for giving semantics to process algebras. The reader is referred to [**97**] for more detail.

In Part 2 we restrict attention to the level of single nets in order to analyse the structure of their spaces of computations, i.e., the *algebraic structure* of their processes. Of course, we keep using categorical tools, following an approach that can be said '*in the small*', as opposed to the one in Part 1 that — dealing with the totality of nets — is '*in the large*'.

The idea is, given a net N, to describe in abstract terms its *concatenable processes*, a notion introduced in [**18**] to account for sequential composition of processes. The existence of an operation of concatenation leads easily to a category of concatenable processes of N, where objects are states (markings) and arrows are (concatenable) processes. It turns out that such a category is a *symmetric monoidal category* whose tensor product is the parallel composition of processes [**18**]. The relevance of this result is that it describes Petri net behaviours as *algebras* in a remarkable way.

Here we recall some of the results of [**77, 18, 45, 75**] providing, in particular, a construction that associates to each net N a symmetric monoidal category $P(N)$ isomorphic to the category of concatenable processes of N. Such an approach is completely abstract, *axiomatic*, in that it is formulated in terms of *universal* constructions. Namely, as we shall see, $P(N)$ is the *free symmetric strict monoidal* category on the net N modulo two simple additional axioms. The exposition is based on [**77**].

Most of the results presented here are based on work by the authors in co-operation with colleagues. Our main contribution here has been to collect and reformulate existing results, and to add a few new results as an attempt to obtain a uniform and coherent exposition. The results on elementary net systems and their relationships to other models in Part 1 is based on various works by G. Rozenberg, P.S. Thiagarajan, and G. Winskel in collaboration with Nielsen. The work on unfolding semiweighted nets and nets as monoidal categories are due to Sassone in collaboration with M. Meseguer, U. Montanari. And finally, Section 4 on nets and bisimulation is adopted from work A. Joyal, and G. Winskel and Nielsen.

Part 1. ON THE BEHAVIOUR OF NETS

1. Petri Nets, Hoare Structures, and Trace Structures

We start out be considering some fundamental and simple classes of Petri nets and their relationships to other models for concurrency. The theory of nets was originally a

strong source of inspiration behind the introduction of traces by Mazurkiewicz in [**43**]. Also, the relationship between traces and nets have been extensively studied, see in particular the survey papers by Rozenberg and Thiagarajan in [**73, 87**]. The presentation here is based on joint work with Rozenberg and Thiagarajan, [**59**], in which proofs and details may be found.

1.1. Elementary net systems. Elementary net systems were introduced by Thiagarajan [**87**] as a fundamental class of nets. His definitions were as follows.

DEFINITION. A *condition/event net* (CE for short) is a triple (B,E,F) where B and E are disjoint sets of, respectively, *conditions* and *events*, $F \subseteq (B \times E) \cup (E \times B)$, the *flow relation*, admits no isolated elements, i.e.,

$$\mathit{domain}(F) \cup \mathit{range}(F) = B \cup E,$$

where $\mathit{domain}(F) = \{x \mid \exists y.\ (x,y) \in F\}$ and $\mathit{range}(F) = \{y \mid \exists x.\ (x,y) \in F\}$.

Let $N = (B,E,F)$ be a CE. Then $X_N = B \cup E$ is the set of *elements* of N. Let $x \in X_N$. It will be convenient to use the following notation.

$$\begin{array}{rcll} {}^\bullet x & = & \{y \mid (y,x) \in F\} & \text{(the set of } \textit{pre-elements} \text{ of } x\text{)} \\ x^\bullet & = & \{y \mid (x,y) \in F\} & \text{(the set of } \textit{post-elements} \text{ of } x\text{)} \end{array}$$

This 'dot' notation is extended to subsets of X_N in the obvious way. For $e \in E$ we shall call ${}^\bullet e$ the set of *pre-conditions* of e and we shall call $e^\bullet$ the set of *post-conditions* of e.

DEFINITION. A CE net is said to be *simple* if for all $x,y \in X_N$ such that ${}^\bullet x = {}^\bullet y$ and $x^\bullet = y^\bullet$, we have $x = y$.

DEFINITION. An *elementary net system* is a quadruple $N = (B,E,F,c_{in})$ where

- (B,E,F) is a simple net called the *underlying net* of N.
- $c_{in} \subseteq B$ is the *initial case* of N.

Thus a simple CE net may be viewed as a directed bipartite graph with no isolated or confused elements, and an elementary net system is a simple net together with a 'state' specified as subset of B-elements.

Presenting an elementary net system as a graph, following standard practise, the B-elements will be drawn as circles, the E-elements as boxes, the elements of the flow relation, F, as directed arcs, and the initial case will be indicated by dots (tokens) on its members. Figure 1 is an example of a net.

As a model for concurrency, B-elements are used to denote the (local) atomic states (or resources) called *conditions* and the E-elements are used to denote (local) atomic changes-of-states called *events*. The flow relation models the effect on conditions by an occurrence of an event, in the form of a *fixed* neighbourhood relation between the conditions and events of a system.

The dynamics of an elementary net system are simple. A state (usually called a *case*) of the system consists of a set of conditions holding concurrently. An event can occur at a case if all its pre-conditions and none of its post-conditions hold at the case. When an event occurs each of its pre-conditions ceases to hold and each of its post-conditions begins to hold. Let us formalise this dynamics of net systems.

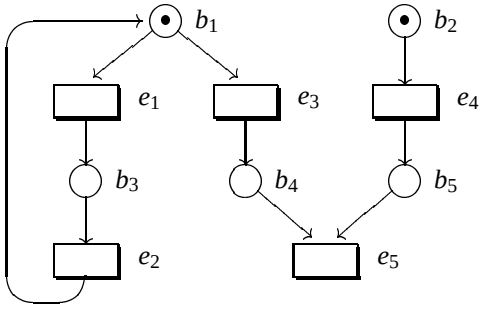


FIGURE 1

DEFINITION. Let $N = (B, E, F)$ be a net. Then $\longrightarrow_N \subseteq Pow(B) \times E \times Pow(B)$ is the *(elementary) transition relation* generated by N, and is given by

$$\longrightarrow_N = \{(k, e, k') \mid k \setminus k' = {}^\bullet e \quad \& \quad k' \setminus k = e^\bullet\}$$

DEFINITION. Let $N = (B, E, F, c_{in})$ be an elementary net system.

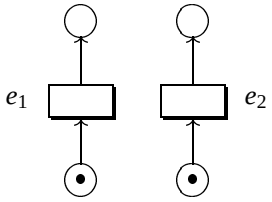
- C_N, the *state space of* N, is the least subset of $Pow(B)$ containing c_{in} such that, if $c \in C_N$ and $(c, e, c') \in \longrightarrow_N$, then $c' \in C_N$. (Note that, whenever possible, we use N to denote both the net system and its underlying net.)
- $CG_N = (C_N, \longrightarrow_{C_N})$, where $\longrightarrow_{C_N} = \longrightarrow_N \cap (C_N \times E \times C_N)$, is the *case graph* associated with N.

The case graph of N describes the dynamics of N by giving, for any possible state, the diagram of the possible state-transitions.

Basic concepts concerning the behaviour of distributed systems such as causality, choice, concurrency, and confusion ('glitch') can now be cleanly defined — and separated from each other — with the help of net systems. The interested reader is referred to Thiagarajan [**87**] for details. Here we just bring out a few important behavioural concepts.

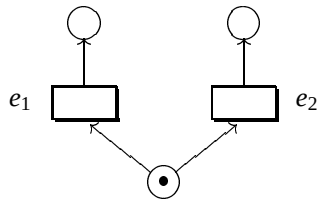
EXAMPLE. Let us illustrate by means of a few small examples how nets can be used to model concurrency, nondeterminism, and enabling.

(1) **Concurrency:**



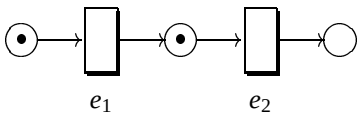
The events e_1 and e_2 can occur concurrently, in the sense that they both have concession and are independent in not having any pre or post conditions in common.

(2) **Conflict:**



Either one of events e_1 and e_2 can occur, but not both. This shows how nondeterminism can be represented in a net.

(3) **Contact:**



The event e_2 has concession. The event e_1 does not — its post condition holds — and it can only occur after e_2. This illustrates contact. In general, there is *contact* at a marking M when for some event e

$$^\bullet e \subseteq M \quad \& \quad e^\bullet \cap (M \setminus {}^\bullet e) \neq \varnothing.$$

As a further example, a critical region may be described as the elementary net system in Figure 2, where the condition in the center represents a kind of semaphore controlling the access (p's and v's events) to critical regions (c_0 and c_1) by the two processes.

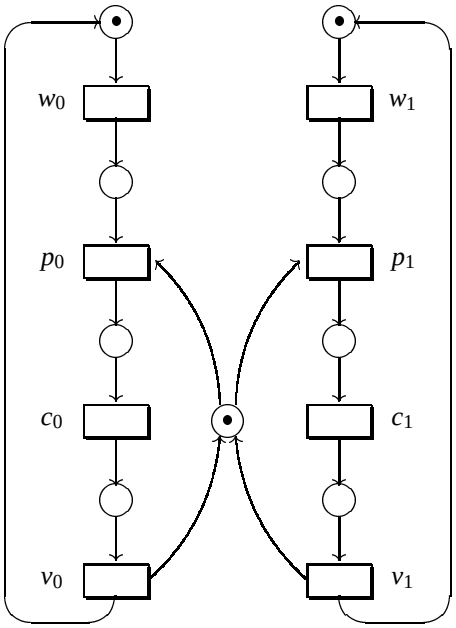


FIGURE 2

1.2. Trace structures. An traditional way to describe the behaviour of a system is to consider all the admissible sequences of event occurrences, the so-called *traces* of the system. Essentially, this amounts to giving a formal language whose alphabet is a set of events and whose strings represent the potential evolution of the system. Trace structures, introduced originally by Mazurkiewicz [**43**] as a model for concurrency, arose from a simple, yet powerful new idea: equip the alphabet of formal languages with an extra structure of *independence*, interpreted as computational independence between atomic *actions*. We recall this development starting with the following simpler notion.

DEFINITION. A *Hoare structure* is a pair (H,Σ) where Σ is an alphabet (of atomic actions), and H is a nonempty, prefix closed subset of the monoid Σ^*.

Actually, such structures are called traces in [**30**], but we prefer to reserve the word traces for the structures that will follow. Building on the definition of the transition relation we may associate an obvious Hoare structure with an elementary net system.

DEFINITION. The set FS_N of *firing sequences* of $N = (B,E,F,c_{in})$ is the subset of E^* defined inductively as follows.

- $\varepsilon \in FS_N$ and $c_{in}\,[\![\varepsilon\rangle\, c_{in}$, for ε the empty sequence;
- $$\frac{\rho \in FS_N \quad \text{and} \quad c_{in}\,[\![\rho\rangle\, c \quad \text{and} \quad c \xrightarrow{e}_N c'}{\rho e \in FS_N \quad \text{and} \quad c_{in}\,[\![\rho e\rangle\, c'}.$$

Observe that $[\![\ \rangle$ is the natural 'extension' of $\longrightarrow_N$ to $\{c_{in}\} \times E^* \times C_N$.

For the elementary net representation of the familiar example of mutual exclusion, we get the following Hoare structure

$$\{\varepsilon, w_0, w_1, w_0w_1, w_1w_0, w_0p_0, w_1p_1, w_0w_1p_0, \dots\}.$$

One of the essential aspects of nets is that they allow an explicit representation of the distributed nature of computations. For instance, in the mutual exclusion example of Figure 2, the independence between actions w_0 and w_1 is represented, following our intuitive understanding of the net, by the disjointness of their local effects. However, as with Hoare structures in general, firing sequences 'hide' aspects of the behaviour of a net system to do with parallel or independent activities. To bring out this deficiency more clearly, we follow the original way of introducing independence between events of elementary net systems. In net theory this relation is most often referred to as the *concurrency relation*.

DEFINITION. For $e_1 \neq e_2 \in E$ and $c \in C_N$, say that e_1 and e_2 can occur *concurrently* at c — written $c[\![\{e_1,e_2\}\rangle$ or, when c can be omitted, also e_1 co e_2 — if $c[\![e_1\rangle$, $c[\![e_2\rangle$, and $({}^\bullet e_1 \cup e_1^\bullet) \cap ({}^\bullet e_2 \cup e_2^\bullet) = \varnothing$.

Thus e_1 and e_2 can occur concurrently at c iff they can occur individually and their neighbourhoods are disjoint. *Conflict* is clearly the 'dual' notion: e_1 and e_2 are in conflict at c if $c[\![e_1\rangle$, $c[\![e_2\rangle$, *but not* $c[\![\{e_1,e_2\}\rangle$, i.e., at c either e_1 may occur or e_2 may occur but not both. The choice as to whether e_1 or e_2 will occur is assumed to be resolved by the 'environment' of the system. For the system N of Figure 1, for instance, at the initial case e_1 and e_4 can occur concurrently. Consequently, the

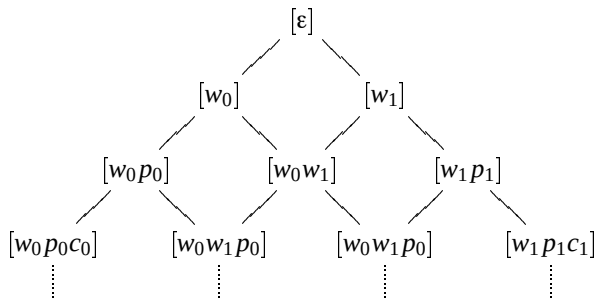


FIGURE 3

firing sequences $e_1e_2e_4$ and $e_4e_1e_2$ and $e_1e_4e_2$ all represent the *same* (non-sequential) stretch of behaviour of N. Also, e_1 and e_3 are in conflict at the initial case. Hence the firing sequences $e_1e_2e_4$ and $e_3e_4e_5$ represent two conflicting (alternative) stretches of behaviour of N.

The idea suggested by Mazurkiewicz [**43**] is to allow the modelling of such independent activities of components of system by introducing the extra structure of an independence relation I on the action alphabet. For nets, following our intuition we would relate two actions as independent if and only if they involve concurrent events. Based on an independence alphabet, the behaviour of a system will be modeled in terms of traces, i.e., of *equivalence classes* of

$$\simeq_I \ : \text{the least congruence on } E^* \text{ such that} \quad e_0e_1 \simeq_I e_1e_0 \quad \text{whenever} \quad e_0 \, I \, e_1.$$

In our running example $w_0w_1p_0 \simeq_I w_0p_0w_1$ and the equivalence class of $w_0w_1p_0$ is $[w_0w_1p_0] = \{w_0w_1p_0, w_0p_0w_1, w_1w_0p_0\}$.

Now, Hoare structures generalise from subsets of Σ^* to subsets of the monoid of traces, denoted $M(\Sigma, I)$. The prefix ordering of Hoare structures generalise to a prefix ordering of traces, defined in terms of the following preorder on strings:

$$s \lesssim_I t \quad \text{if and only if} \quad \exists u.\, su \simeq_I t$$

which induces the following partial order (*prefix order*) on traces:

$$\sqsubseteq_I = \lesssim_I / \simeq_I,$$

that is,

$$[s] \sqsubseteq_I [t] \quad \text{if and only if} \quad \exists u, v.\, s \simeq_I u \lesssim_I v \simeq_I t.$$

In our example $[\varepsilon] \sqsubseteq_I [w_0] \sqsubseteq_I [w_0w_1] \sqsubseteq_I [w_0p_0w_1]$, and the initial traces and their prefix ordering are as shown in Figure 3. Notice that the extra modelling power boils down to the presence of traces like $[w_0w_1]$ in our example, representing actions w_0 and w_1 in any (unspecified) order, and interpreted as their concurrent or independent occurrences.

We are now ready for our formal definition of trace structures. Conceptually, we follow [**44**] where a trace structure is defined to be a prefix closed, proper subset of the monoid $M(\Sigma, I)$. However, only for technical reasons we prefer in our formal definition to work with such structures in terms of consistent subsets of Σ^* — with the traces a derived notion, as in Proposition 1.1 below.

DEFINITION. A *trace structure* is a triple $T = (M, \Sigma, I)$ where (Σ, I) is an *independence alphabet*, i.e. $I \subseteq \Sigma \times \Sigma$ is *irreflexive* and *symmetric*, and $M \subseteq \Sigma^*$ is such that for all $t, t' \in \Sigma^*$ and $a, b \in \Sigma$:

$$\begin{array}{lll} \textbf{\textit{consistency:}} & t \approx_I t' \in M & \Longrightarrow t \in M; \\ \textbf{\textit{prefix closure:}} & ta \in M & \Longrightarrow t \in M; \\ \textbf{\textit{properness:}} & ta, tb \in M \;\&\; a\, I\, b & \Longrightarrow tab \in M. \end{array}$$

We use the notation $M/\approx_I$ for the traces of T, i.e.,

$$M/\approx_I = \{[w] \mid w \in M\}.$$

We may think of a trace structure as a prefix closed set of traces, in the sense that from the axioms of consistency and prefix closure above, we get the following.

PROPOSITION 1.1. *Given a trace structure $T = (M, \Sigma, I)$ then $(M/\approx_I, \sqsubseteq_I)$ satisfies*

- $w \in M$ *if and only if* $[w] \in M/\approx_I$;
- $[w] \sqsubseteq_I [w'] \in M/\approx_I$ *implies* $[w] \in M/\approx_I$.

As will be expected by now, the information concerning concurrency and conflict-resolution hidden by Hoare structures may be retrieved by associating with a net a trace structure with concurrency as the appropriate independence relation.

THEOREM 1.2. *Let $N = (B, E, F, c_{in})$ be an elementary net system and let the independence relation associated with N be*

$$I = \{(e_1, e_2) \mid e_1, e_2 \in E \;\&\; ({}^\bullet e_1 \cup e_1^\bullet) \cap ({}^\bullet e_2 \cup e_2^\bullet) = \varnothing\}.$$

Then $nt(N) = (FS_N, E, I)$ is a trace structure.

PROOF. The required properties follow from definition. In particular, $nt(N)$ is consistent and proper by definition of the (elementary) transition relation. □

For the net system N from Figure 1, Figure 4 shows an initial portion of the associated poset of traces.

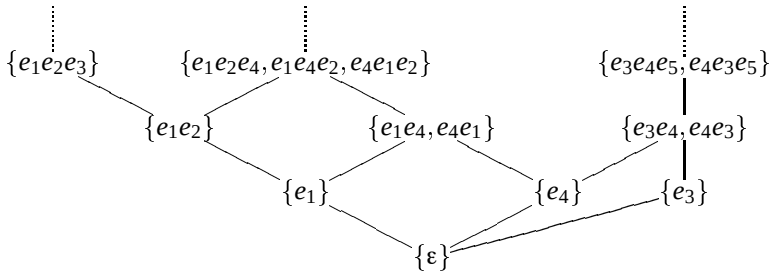


FIGURE 4

The beauty of the trace semantics is its simplicity. One of the classical results from concurrency theory is that the trace semantics is 'consistent' with an alternative way of defining the behaviour of net systems in terms of unfoldings into processes (or occurrence nets). Several results of this type have been shown [**9**]. The presentation

that follows is adapted from [**59**]. For the sake of convenience we shall assume here that N is *contact-free*. In other words, we shall assume,

$$\forall c \in C_N.\ \forall e \in E.\ {}^\bullet e \subseteq c \implies e^\bullet \cap (c \setminus {}^\bullet e) = \varnothing.$$

As we shall see later, this does not involve any loss of generality, at least for the study of behavioural issues.

The theoretic development of Petri nets, focusing on the noninterleaving aspects of concurrency, brought to the foreground various notions of process, e.g. [**68, 26, 7, 45, 18**]. Generally speaking, these are structures accounting for the *causal relationships* which rule the occurrence of events in computations. Thus, ideally, processes are simply computations in which explicit information about such causal connections is added. Abstractly, the processes of a net N are ordered sets whose elements are labelled by events of N. Concretely, in order to describe exactly which sets of events give rise to processes, one takes a process of $N = (B, E, F, c_{in})$ will be a *labelled net* of the form $\tilde{N} = (\tilde{B}, \tilde{E}, \tilde{F}, \pi)$, where $(\tilde{B}, \tilde{E}, \tilde{F})$ is a restricted kind of a net (viz., finite, conflict-free, acyclic) called a *causal* or *process* net, and the labelling function $\pi\colon \tilde{B} \cup \tilde{E} \to B \cup E$ is required to connect the structure of $\tilde{N}$ to that of N in a suitable way. For a definition of a process along these lines see Part 2, or, e.g., [**73**].

Here we shall define processes with the help of firing sequences. This will enable us to build up the finite processes of N inductively. For a similar development of the process notion, see [**9**].

For each firing sequence ρ, we will define a process $N_\rho = (B_\rho, E_\rho, F_\rho, \pi_\rho)$. In doing so it will be convenient to keep track of the conditions that hold in N after the run represented by the firing sequence ρ. This set of conditions will be encoded as c_ρ.

DEFINITION. Let N be (B, E, F, c_{in}). Then $N_\rho = (B_\rho, E_\rho, F_\rho, \pi_\rho)$ is defined inductively on the length of $\rho \in FS_N$ as follows.

Case $\rho = \varepsilon$**:** Then $N_\varepsilon = (\varnothing, \varnothing, \varnothing, \varnothing)$ and $c_\varepsilon = \{(b, \phi) \mid b \in c_{in}\}$.

Case $\rho = \rho' e$**:** Assume that $N_{\rho'} = (B_{\rho'}, E_{\rho'}, F_{\rho'}, \pi_{\rho'})$. Then $N_\rho = (B_\rho, E_\rho, F_\rho, \pi_\rho)$ where, for $X = \{(b, D) \mid b \in {}^\bullet e \ \&\ (b, D) \in c_{\rho'}\}$ and $Y = \{(b, \{(e, X)\}) \mid b \in e^\bullet\}$, we have

$$\begin{aligned} E_\rho &= E_{\rho'} \cup \{(e, X)\}, \\ B_\rho &= B_{\rho'} \cup X \cup Y, \\ F_\rho &= F_{\rho'} \cup (X \times \{(e, X)\}) \cup (\{(e, X)\} \times Y), \\ \pi_\rho &= \lambda(z, Z) \in B_\rho \cup E_\rho.\ z. \end{aligned}$$

Finally, $c_\rho = (c_{\rho'} \setminus X) \cup Y$.

It will turn out that N_ρ as defined above is a labelled net. For $\rho = e_1 e_2 e_4 e_3$ in the system N of Figure 1 we show N_ρ in Figure 5. For convenience we have displayed π_ρ by writing the value of $\pi_\rho(x)$ besides the graphical representation of x for each $x \in B_\rho \cup E_\rho$.

In order to establish a relationship between the traces of N and its processes it is necessary to define an ordering relation over the processes of N.

DEFINITION. Let N be (B, E, F, c_{in}).

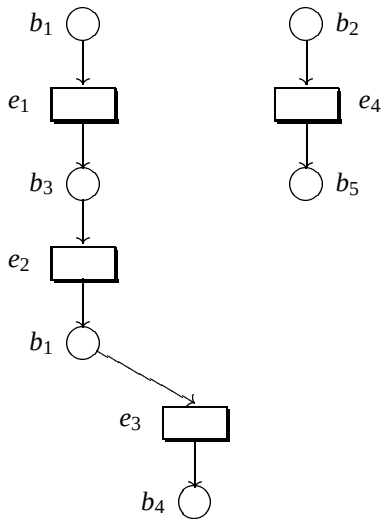


FIGURE 5

- ▷ The set of finite processes of N is $P_N = \{N_\rho \mid \rho \in FS_N\}$, for N_ρ as in the previous definition.
- ▷ $\sqsubseteq \subseteq P_N \times P_N$ is defined by

$$(B_\rho, E_\rho, F_\rho, \pi_\rho) \sqsubseteq (B_{\rho'}, E_{\rho'}, F_{\rho'}, \pi_{\rho'}) \quad \text{if} \quad B_\rho \subseteq B_{\rho'} \;\&\; E_\rho \subseteq E_{\rho'} \;\&\; F_\rho \subseteq F_{\rho'}.$$

Clearly $\sqsubseteq$ is a partial ordering relation. The main result relating trace semantics to processes is the following.

THEOREM 1.3. *For N any elementary net system $(P_N, \sqsubseteq)$ and the ordering of the traces from $nt(N)$, i.e., $(FS_N/\approx_I, \sqsubseteq_I)$, are isomorphic posets.*

PROOF. In [**59**] it is proved that $f\colon FS_N/\approx_I \to P$ given by $f([\rho]) = N_\rho$ is an isomorphism. □

1.3. A categorical way to relationships. In the last section we attempted to show connections between net systems and other structures. Although it is apparent that nets a more general, expressive, and powerful model, we lack at this stage a way to make precise any statement in this sense. Is there a formal way of saying that traces 'embeds' into nets, that 'nets generalise' them naturally? More generally, how can we relate nets to the other models? How do we establish relationships between models?

As we discussed in the introduction, we tend to classify models can for concurrency according to their '*level of abstraction*' (see, e.g., [**80, 97, 79**]), that is, according to those aspects of the behaviour of distributed systems they focus on and those they deliberately abstract from. In stating and proving relationships between models viewed under this perspective, the language of *category theory* as proven in many contexts to be very useful, as it is capable of *abstracting* away from unwanted details of the individual models and, therefore, of expressing the more essential aspects *succinctly* and in great *generality*. Let us review very briefly a few key steps behind these ideas.

First, all the models are introduced as a class of *objects*, e.g., the class of net systems or the class of trace structures, equipped with a notion of 'behaviour-preserving' (i.e., simulation) *morphism*, making each model into a category. The role of the morphisms is to make explicit (*if* and) *how* each single object relates to all the others. In particular, as behaviour is preserved, (*if* and) *how* it can be simulated. This makes explicit that central to our objects and, therefore, to the respective categories, is the dynamic notion of behaviour. Also, the very notion adopted for 'simulation' determines what aspects of behaviour are important, i.e., what can be ignored by a successful simulation (the aspects abstracted away) and what instead must be preserved (the aspects focused on). In other terms, the adopted notion of morphism define the abstraction level of the model.

From this standpoint, the notion of *functor* is the first tool category theory makes available to us in order to check the sanity of our translations from one model to another. Essentially, it requires us to map objects to objects *preserving* all the existing relationships, i.e., all the existing simulations. In other words, it requires to map also behaviours to behaviours.

Tools much more refined than functors are the notions of *adjunction* and *coreflection*, central to many papers on models of concurrency and, in particular, to our presentation here. Let us briefly comment on their formal definition and the intuitive way to understand them. Technically, an adjunction between categories M_0 and M_1, consists of ways of mapping from one to the other and back, satisfying certain properties. Formally, (see [**42**] for alternative characterisations) we shall express an adjunction in terms of two functors $\mathsf{L}\colon \mathsf{M}_0 \to \mathsf{M}_1$ (the *left adjoint*) and $\mathsf{R}\colon \mathsf{M}_1 \to \mathsf{M}_0$ (the *right adjoint* of the adjunction) satisfying (see Figure 6):

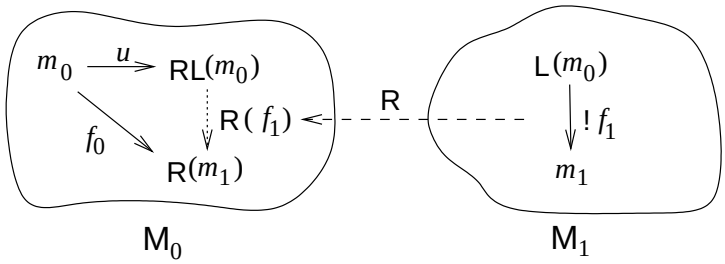


FIGURE 6

> for *each* object m_0 of M_0, there is a morphism $u\colon m_0 \to \mathsf{R} \circ \mathsf{L}(m_0)$ (the *unit* at m_0) such that for *each* object m_1 of M_1 and *each* morphism $f_0\colon m_0 \to \mathsf{R}(m_1)$, then there is a *unique* morphism $f_1\colon \mathsf{L}(m_0) \to m_1$, such that $f_0 = \mathsf{R}(f_1) \circ u$.

In other terms, for each $m_0 \in \mathsf{M}_0$, $\mathsf{L}(m_0)$ is a 'special' object of M_1 in the sense that all the maps from m to objects of the kind $\mathsf{R}(m_1)$ in M_0 come exactly and unambiguously from maps from $L(m_0)$ to m_1 in M_1. Moreover, all such maps can be factored in the image of R via $u\colon m_0 \to \mathsf{R} \circ \mathsf{L}(m_0)$, a 'special' map that m_0 comes equipped with. Reading system for object and simulation for morphism, this definition has an evident significance in computational terms.

If all units of an adjunction are *isomorphisms*, then the adjunction is called a *coreflection*. This essentially means that no information is lost moving from M_0 to M_1, as the identity of objects is retained and recovered back by R. It follows from the definition that the left adjoint L of a coreflection is always *full* and *faithful*, i.e., an *embedding*. In other words, we may think of M_0 as a coreflective full subcategory of M_1 (the one identified by the image of L), and of L as the inclusion $\mathsf{M}_0 \hookrightarrow \mathsf{M}_1$, whereas R tells us how to project M_1 back onto M_0.

Paraphrasing this situation in terms of categories of models, behaviours and simulations, we can say that R selects for each $m \in \mathsf{M}_1$ its *best possible abstract 'approximation'* in M_0. That is, an object $\mathsf{R}(m) \in \mathsf{M}_0$ together with a simulation $\mathsf{R}(m) \to m$ such that any other $\mathsf{R}(m') \to m$ factors as $\mathsf{R}(m') \to \mathsf{R}(m) \to m$.

So much for the formal definition. In the following the existence of a coreflection of M_0 into M_1 will be our formal way of saying that 'M_0 is an *abstract* version of M_1'.

We therefore start by turning our models into a categories by defining appropriate notions of morphisms. Morphisms of languages are simply functions on their alphabets which send strings in one language to strings in another.

DEFINITION. A function $\lambda\colon \Sigma \to \Sigma'$ extends to strings by defining

$$\widehat{\lambda}(s\alpha) = \widehat{\lambda}(s)\lambda(\alpha).$$

A *morphism* of Hoare structures $(H,\Sigma) \to (H',\Sigma')$ consists of a function $\lambda\colon \Sigma \to \Sigma'$ such that $\forall s \in H.\ \widehat{\lambda}(s) \in H'$.
We write H for the category of Hoare structures with the above understanding of morphisms, where composition is our usual composition of functions.

Before we continue, let us comment briefly on our choice of morphisms — on Hoare structures as well as on all other models considered in this paper. In much of the literature, more liberal notions of morphisms are used, based on partial (rather than total) functions on the labelling sets. These more general morphisms have the advantage that many useful combinators (e.g., parallel composition) may be expressed as universal constructions in the corresponding categories of models. Furthermore, they may be thought of as specifying correctness properties: the 'correctness' of the mutual exclusion example, for instance, follows by the fact that the partial function λ from the alphabet of actions, which is undefined for $\{w_0, w_1\}$ and the identity function for all other action symbols, *is* a morphism from the Hoare structure of the mutual exclusion example to the Hoare structure consisting of all prefixes of the regular language $(p_0c_0v_0 + p_1c_1v_1)^*$. However, we have chosen here to restrict ourselves to morphisms based on total functions, purely as an attempt to simplify our presentation technically.

Similarly, morphisms between trace structures are morphisms between the underlying languages which preserve independence.

DEFINITION. A *morphism* of trace structures $(M,\Sigma,I) \to (M',\Sigma',I')$ consists of a function $\lambda\colon \Sigma \to \Sigma'$ which

***preserves independence*:** $\alpha\, I\, \beta$ implies $\lambda(\alpha)\, I'\, \lambda(\beta)$, for all $\alpha,\beta \in \Sigma$;

***preserves strings*:** $s \in M$ implies $\widehat{\lambda}(s) \in M'$, for all strings s.

This, with the usual composition of functions defines T, the category of trace structures.

It is easy to see that morphisms of trace structures preserve traces and the ordering between them.

PROPOSITION 1.4. *Let $\lambda\colon (M,\Sigma,I) \to (M',\Sigma',I')$ be a morphism of trace structures. If $s \lesssim_I t$ in the trace structure (M,Σ,I) then $\widehat{\lambda}(s) \lesssim_{I'} \widehat{\lambda}(t)$ in (M',Σ',I').*

It follows that $\widehat{\lambda}$ defines a monote function from $(M/\approx_I, \sqsubseteq_I)$ to $(M'/\approx_{I'}, \sqsubseteq_{I'})$. Concerning nets, we consider the following definition.

DEFINITION. Let $N = (B,E,F,c_{in})$ and $N' = (B',E',F',c'_{in})$ be elementary net systems. A *morphism* $(\beta,\eta)\colon N \to N'$ consists of a relation $\beta \subseteq B \times B'$, such that β^{op} is a partial function $B' \rightharpoonup B$, and a function $\eta\colon E \rightharpoonup E'$ such that

$$\forall (b,b') \in \beta.\ b \in c_{in} \iff b' \in c'_{in}, \qquad \beta({}^\bullet e) = {}^\bullet\eta(e), \qquad \beta(e^\bullet) = \eta(e)^\bullet.$$

Thus morphisms on nets preserve initial cases and events when defined. A morphism $(\beta,\eta)\colon N \to N'$ expresses how occurrences of events and conditions in N induce occurrences in N'. Morphisms on nets preserve behaviour.

PROPOSITION 1.5. *Let $N = (B,E,F,c_{in})$ and $N' = (B',E',F',c'_{in})$ be elementary nets and $(\beta,\eta)\colon N \to N'$ a morphism.*

- *If $c\ [\![e\rangle\ c'$ in N, then $f_\beta(c)\ [\![\eta(e)\rangle\ f_\beta(c')$ in N', for $f_\beta(c) = \beta(c) \cup (c'_{in} \smallsetminus \beta(c_{in}))$.*
- *If ${}^\bullet e_1^\bullet \cap {}^\bullet e_2^\bullet = \varnothing$ in N, then ${}^\bullet\eta(e_1)^\bullet \cap {}^\bullet\eta(e_2)^\bullet = \varnothing$ in N'.*

PROOF. It is easily seen that ${}^\bullet\eta(e) = \beta({}^\bullet e)$ and that $\eta(e)^\bullet = \beta(e^\bullet)$ for all events e of N. Observe too that because β^{op} is a partial function, β in addition preserves intersections and set differences. These observations mean that $\beta(c)\ [\![\eta(e)\rangle\ \beta(c')$ in N' follows from the assumption that $c\ [\![e\rangle\ c'$ in N, and that independence is preserved. □

DEFINITION. Let EN denote the category of elementary net systems and their morphisms under the obvious componentwise composition of morphisms, e.g., the composition of $(\beta_0,\eta_0)\colon N_0 \to N_1$ and $(\beta_1,\eta_1) : N_1 \to N_2$ is $(\beta_1 \circ \beta_0, \eta_1 \circ \eta_0)\colon N_0 \to N_2$

This choice of morphisms for elementary net systems may not be as obvious and intuitively clear as the those for the other models we consider. Indeed alternative categories of net systems have been studied — see, e.g., [**93, 45, 12, 48, 97**]. Here we just remark that Proposition 1.5 proves that these morphisms preserve behaviour (and concurrency), a fact that has been explored by, e.g. [**11**], where such morphism have been used to express correctness properties. Also, we note that the derived notion of isomorphism becomes identity up to names of conditions and events.

THEOREM 1.6. *The construction that maps $N = (B,E,F,c_{in})$ to the Hoare structure (FS_N,E) extends to a functor nh from EN to H.*

THEOREM 1.7. *The trace semantics nt extends to a functor nt from EN to T.*

However, these functors are *not* part of any adjunction. Following our discussion above, one would expect a formal result embedding T in EN, but for this to be the case it turns out that one needs a more abstract semantics. The reason why *nt* is too concrete is that it preserves information about event 'identities'. As we shall see in the next section, forgetting these will help yielding a 'nice' (read 'universal') unfolding of EN into event structures.

2. Petri Nets, Event Structures, and Domains

Consider again the prefix ordering of traces introduced above. What can be said about their structure and properties? In this section we shall provide a characterisation of such orderings in terms of a well-known class of *Scott domains* [**81, 6**]. Moreover, in the process of doing so, we shall also show that they arise exactly as the orderings associated with the dynamics of another well-known model for concurrency: the *event structures*, originally introduced in [**57**].

2.1. Event structures. The prefix ordering of the strings of a Hoare structure — which is in fact a *tree* ordering — may also be viewed as a structure over action occurrences, where individual occurrences may be either *ordered*, i.e., following each other in time in the same computation, or *not*, i.e., belong to different computations. Event structures may be seen as a generalisation of such structures, allowing a third relationship between occurrences, that of *concurrency*, i.e., belonging to the same computation, but without any *causal/temporal* ordering.

DEFINITION. Define an *event structure* to be a structure $(E, \leq, \#)$ consisting of a set E, of *events* which are partially ordered by $\leq$, the *causal dependency relation*, and a binary, symmetric, irreflexive relation $\# \subseteq E \times E$, the *conflict relation*, which satisfy for all $e, e', e'' \in E$

$$\{e' \mid e' \leq e\} \text{ is finite}, \qquad e \# e' \leq e'' \implies e \# e''.$$

Say two events $e, e' \in E$ are *concurrent*, and write $e \operatorname{co} e'$, if $\neg(e \leq e' \text{ or } e' \leq e \text{ or } e \# e')$. Write $\mathbb{W}$ for $\# \cup 1_E$, i.e., the reflexive closure of the conflict relation.

The finiteness assumption restricts attention to discrete processes where an event occurrence depends only on finitely many previous occurrences. The axiom on the conflict relation expresses that if two events causally depend on events in conflict then they too are in conflict.

Guided by our interpretation we can formulate a notion of computation state of an event structure $(E, \leq, \#)$. Taking a computation state of a process to be represented by the set x of events which have occurred in the computation, we expect that

$$e' \in x \quad \& \quad e \leq e' \quad \implies \quad e \in x,$$

i.e., if an event has occurred then all events on which it causally depends have occurred too, and also that

$$\forall e, e' \in x.\ \neg(e \# e'),$$

i.e., two conflicting events cannot occur together in the same computation.

DEFINITION. Let $(E, \leq, \#)$ be an event structure. Its *configurations*, $D(E, \leq, \#)$, are those subsets $x \subseteq E$ which are

***conflict-free*:** $\forall e, e' \in x.\ \neg(e \# e')$;

***downwards-closed*:** $\forall e, e'.\ e' \leq e \in x \implies e' \in x$.

In particular, define $\lfloor e \rfloor = \{e' \in E \mid e' \leq e\}$, which is a configuration, as it is downward-closed and conflict-free. Write $D^0(E, \leq, \#)$ for the set of finite configurations.

The important relations associated with an event structure can be recovered from its finite configurations (or indeed similarly from its configurations).

PROPOSITION 2.1. *Let $(E,\leq,\#)$ be an event structure. Then*

- $e \leq e'$ *if and only if* $\forall x \in D^0(E,\leq,\#).\ e' \in x \implies e \in x$;
- $e \# e'$ *if and only if* $\forall x \in D^0(E,\leq,\#).\ e \in x \implies e' \notin x$;
- $e \operatorname{co} e'$ *if and only if* $\exists x,x' \in D^0(E,\leq,\#)$ *such that*

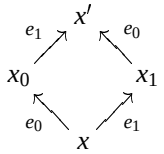
$$e \in x \smallsetminus x' \ \&\ e' \in x' \smallsetminus x \ \&\ x \cup x' \in D^0(E,\leq,\#).$$

Events manifest themselves as atomic jumps from one configuration to another, and later it will follow that we can regard such jumps as transitions in the case graph associated with a net system.

DEFINITION. Let $(E,\leq,\#)$ be an event structure and x,x' be configurations. Write

$$x \xrightarrow{e} x' \quad \text{if and only if} \quad e \notin x \ \&\ x' = x \cup \{e\}.$$

PROPOSITION 2.2. *Two events e_0,e_1 of an event structure are in the concurrency relation* co *if and only if there exist configurations x,x_0,x_1,x' such that*



Morphisms on event structures are defined as follows [**92, 91**]:

DEFINITION. Let $ES = (E,\leq,\#)$ and $ES' = (E',\leq',\#')$ be event structures. A *morphism* from ES to ES' consists of a function $\eta\colon E \to E'$ on events which satisfies

$$x \in D(ES) \implies \eta(x) \in D(ES')$$
$$\forall e_0,e_1 \in x.\ \eta(e_0) = \eta(e_1) \implies e_0 = e_1.$$

A morphism $\eta\colon ES \to ES'$ between event structures expresses how behaviour in ES determines behaviour in ES'. The function η expresses how the occurrence of events in ES implies the simultaneous occurrence of events in ES'; the fact that $\eta(e) = e'$ can be understood as expressing that the event e' is a 'component' of the event e and, in this sense, that the occurrence of e implies the simultaneous occurrence of e'. If two distinct events in ES have the same image in ES' under η then they cannot belong to the same configuration.

Morphisms of event structures preserve the concurrency relation. This is a simple consequence of Proposition 2.2, showing how the concurrency relation holding between events appears as a 'little square' of configurations.

PROPOSITION 2.3. *Let E be an event structure with concurrency relation* co *and E' an event structure with concurrency relation* co$'$. *Let $\eta\colon E \to E'$ be a morphism of event structures. Then, for any events e_0,e_1 of E,*

$$e_0 \operatorname{co} e_1 \implies \eta(e_0) \operatorname{co}' \eta(e_1).$$

Morphisms between event structures can be described more directly in terms of the causality and conflict relations of the event structure.

PROPOSITION 2.4. *A morphism of event structures from $(E,\leq,\#)$ to $(E',\leq',\#')$ is a function $\eta\colon E\to E'$ such that*

- $\lfloor\eta(e)\rfloor\subseteq\eta(\lfloor e\rfloor)$,
- $\eta(e_0)\mathrel{\mathbb{W}'}\eta(e_1)\quad\Longrightarrow\quad e_0\mathrel{\mathbb{W}}e_1$.

Let E denote the category of event structures with morphism as described above and composition named composition of functions.

2.2. Event structures and domains. Let us turn our attention to the class of partial orders corresponding with the orderings of configurations of event structures. The characterisation given below in terms of special Scott domains has been originally formulated in [**57**].

In the following, we shall need a few standard definitions from domain theory. For $(D,\sqsubseteq)$ a partial order and X a subset of D, we write as usual $\bigsqcup X$ for the least upper bound of X, when it exists.

DEFINITION. Let $(D,\sqsubseteq)$ be a partial order. A *complete prime* of D is an element $p\in D$ such that

$$p\sqsubseteq\bigsqcup X\implies\exists x\in X.\ p\sqsubseteq x$$

for any set X for which $\bigsqcup X$ exists.

DEFINITION. For $(D,\sqsubseteq)$ a partial and $d_0,d_1\in D$, we say that d_1 *covers* d_0, in symbols $d_0<d_1$, if and only if $d_0\sqsubset d_1$ and, for every d,

$$d_0\sqsubseteq d\sqsubseteq d_1\implies d=d_0\text{ or }d=d_1.$$

DEFINITION. Let $(D,\sqsubseteq)$ be a partial order. We say that D is

- *bounded complete* if all subsets $X\subseteq D$ which have an upper bound in D have a *least* upper bound $\bigsqcup X$ in D.
- *coherent* if all subsets $X\subseteq D$ which are *pairwise* bounded (i.e., such that all pairs of elements $d_0,d_1\in X$ have upper bounds in D), have *least* upper bounds $\bigsqcup X$ in D. (Note that coherence implies bounded completeness).
- *prime algebraic* if

  $$x=\bigsqcup\{p\sqsubseteq x\mid p\text{ is a complete prime}\},$$

  for all $x\in D$. If furthermore the sets

  $$\{p\sqsubseteq q\mid p\text{is a complete prime}\}$$

  are always finite when q is a complete prime, then D is said to be *finitary*.

A *prime algebraic domain* domain is a bounded complete and prime algebraic partial order.

THEOREM 2.5. *Let $(E,\leq,\#)$ be an event structure. The partial order $(\mathcal{D}(E,\leq,\#),\subseteq)$, that we shall indicate simply as $\mathcal{D}(E,\leq,\#)$, is a coherent, finitary, prime algebraic domain whose complete primes are the $\{\lfloor e\rfloor\mid e\in E\}$.*

PROOF. See [**57, 95**]. □

Conversely, any coherent, finitary, prime algebraic domain is associated with the partial order of configurations of event structures.

THEOREM 2.6. *Let $(D,\sqsubseteq)$ be a coherent, finitary, prime algebraic domain. Define $Pr(D,\sqsubseteq)$ as the event structure $(E,\leq,\#)$*

$$\begin{array}{rcl} E & = & \text{the complete primes of } (D,\sqsubseteq) \\ \leq & = & \text{is the restriction of } \sqsubseteq \text{ to } E \\ \# & = & \{(x,y) \in E \times E \mid x \sqcup y \text{ does not exist in } D\} \end{array}$$

Then $(D,\sqsubseteq)$ and $D(E,\leq,\#)$ are isomorphic partial orders.

PROOF. See [**57**]. □

Actually, the relationship between event structures and coherent, finitary prime algebraic domains is very strong, in that they are *equivalent*: one can be used to represent the other. This may be formalised also in terms of a categorical equivalence between D and a category of coherent, finitary prime algebraic domains equipped with stable functions as morphisms.

THEOREM 2.7. *Let D denote the category of coherent, finitary prime algebraic domains with morphism functions $f\colon (D_0,\sqsubseteq_0) \to (D_1,\sqsubseteq_1)$ satisfying:*

***additivity*:** *for all $x,y \in D_0$ such that $x \sqcup y$ exists, $f(x \sqcup y) = f(x) \sqcup f(y)$;*

***stability*:** *for all $x,y \in D_0$ such that $x \sqcup y$ exists, $f(x \sqcap y) = f(x) \sqcap f(y)$;*

***covering preserving*:** *for all $x,y \in D_0$. if $x <_0 y$ then $f(x) <_1 f(y)$.*

Then D and E are equivalent categories.

PROOF. One can prove that D and Pr can be extended to functors that form an equivalence of categories. See [**95**]. □

Getting back to trace structures, we may now formulate a functor, which in essence performs the abstraction from identities of events mentioned previously, and based on this a universal form of unfolding elementary nets into D (and hence E).

THEOREM 2.8. *Given a trace structure $T = (M,\Sigma,I)$ then $td(T) = (M/\approx_I, \sqsubseteq_I)$ is a coherent, finitary prime algebraic domain, and td extends to a functor from T to D.*

PROOF. See [**97**]. □

The following is the result announced at the end of the previous section.

THEOREM 2.9. *$td \circ nt$ is the right adjoint of a coreflection between EN and D.*

PROOF. The proof of this is rather involved, but may be found in [**61**], and for more general forms of net systems in [**97**] and [**33**]. □

2.3. Semiweighted nets. Having introduced event structures and used them as a 'bridge' across elementary net systems, trace structures and domains, we now set out to study the relationships between nets and event structures directly, by means of a so-called *unfolding* construction.

Such an approach to net systems was devised in [**57, 94**] for the category *safe nets* and extended in [**48, 46**] to the more generous category of *semiweighted nets*. Informally, it consists of a coreflection of event structures into nets to whose right adjoint, the 'unfolding', is obtained by *'unrolling'* the *'dynamic'* structure of nets to the *'static'* structure of event structures. In other words, it amounts to 'compiling' transitions to events, so yielding a fine-grain description of the *causal* interaction between the components of a computation, as such interactions must be resolved to the global, static relations of causality and conflict of event structures. Under this translation events are to be thought of as unique occurrences of transitions which bear unique, static causal links to each other.

The unfolding construction factors via a coreflection through Occ, the category of *occurrence nets* [**57**], so yielding the following global picture, where $\hookleftarrow$ is the inclusion functor, and the lower arrows are left adjoints.

$$\mathsf{SWNets} \underset{U(\_)}{\overset{\hookleftarrow}{\longrightarrow}} \mathsf{Occ} \underset{E(\_)}{\overset{N(\_)}{\hookleftarrow}} \mathsf{E} \underset{D(\_)}{\overset{Pr(\_)}{\hookleftarrow}} \mathsf{D}$$

In presenting these results, we shall follow closely [**48**]. Let us start by generalising the class of nets we consider.

The first generalisation is to pass from condition/event to place/transition nets, i.e., to allow markings to be *multisets* (rather than sets) of places. The state of a net is now thought as a distribution of resources ('tokens') in places. Differently from conditions, that may simply hold or not, resources may be absent, but also present in multiple copies.

The flow relation F of elementary nets can be equivalently formalised by a pair of functions ${}^\bullet(\_), (\_)^\bullet : E \to Pow(B)$ assigning to each event its pre- and post-set of conditions. In the same way, the structure of a place/transition net can be equivalently described by generalising F to a *multi*relation or by a pair of functions assigning pre- and post-*multi*sets to transitions. Following [**45**], we choose here this second way.

In the following we shall denote by $\mu(S)$ the monoid of multisets of S. Recall that a multiset is a function from S to ω and that the sum $\mu_1 + \mu_2$ is the multiset μ such that $\mu(s) = \mu_1(s) + \mu_2(s)$, for all $s \in S$. Often, we represent $\mu \in \mu(S)$ as a formal sum $\sum \mu(s_i) \cdot s_i$ where only the $s_i \in S$ such that $\mu(s_i) > 0$ appear; the empty multiset will be denoted by 0.

DEFINITION. A *place/transition net* (PT for short) is a structure

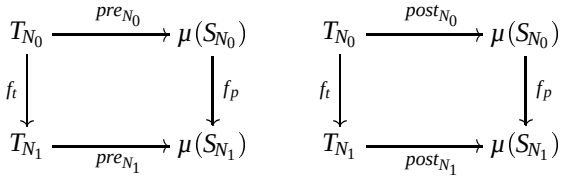
$$N = (pre_N, post_N : T_N \to \mu(S_N))$$

where S_N is a set whose elements are called *places*, T_N is a set whose elements are called *transitions*, pre_N and $post_N$ are functions which assign to each transition, respectively, a *source* (or *pre-set*) and a *target* (or *post-set*) multiset of places, For $t \in T_N$, we write $t : u \to v$ to indicate that $pre_N(t) = u$ and $post_N(t) = v$.

A *morphism* of nets $f\colon N_0 \to N_1$ consists of a pair of functions

$$\langle f_t\colon T_{N_0} \to T_{N_1}, f_p\colon \mu(S_{N_0}) \to \mu(S_{N_1})\rangle,$$

where the place component f_p is a *monoid homomorphism*, which respect initial marking and source and target, i.e., the two diagrams below commute.



Explicitly, f_t and f_p are such that:

$$pre_{N_1} \circ f_t = f_p \circ pre_{N_0}, \quad \text{and} \quad post_{N_1} \circ f_t = f_p \circ post_{N_0}$$

This, with the obvious componentwise composition of morphisms, defines the category PTNets.

A PT net is thus a graph whose arcs are the transitions and whose nodes are the multisets on the set of places, i.e., *markings* of the net. As usual, transitions have pre- and post-sets, i.e., sources and targets, in which each place has only finitely many tokens, i.e., finite multiplicity. The same applies to markings. Finally, morphisms of PT nets are graph morphisms in the precise sense of preserving source and target of transitions. In addition, they respect the monoidal structure of multisets, which simply boils down to saying that $f_p(0) = 0$ and that $f_p(\mu_1 + \mu_2) = f_p(\mu_1) + f_p(\mu_2)$ for each pair of multisets $\mu_1, \mu_2 \in \mu(S_{N_0})$.

In the following we shall consider the category of those PT nets whose initial markings and whose post-sets are sets, as opposed to multisets. Since weights are allowed only on the arcs from places to transitions, they are referred to as *semiweighted* nets [**48**].

DEFINITION. A PT net N is *semiweighted* (SW for short) is for all $t \in T_N$, $post_N(t)$ is a set. Moreover, we assume the standard constraint that $pre_N(t) \neq 0$. A *semiweighted net systems* is a semiweighted net N together with an initial marking u_N that is a *set*.
A *morphism* of semiweighted net systems $f\colon N_0 \to N_1$ is a morphism of the underlying place/transition nets that, in addition, preserves the initial marking, i.e., $f_p(u_{N_0}) = u_{N_1}$.
Semiweighted net systems and their morphisms define the category SWNets.

Notice that disallowing transitions with empty pre-set is a step necessary in any behavioural construction involving nets, as such transitions are highly degenerated; in particular, any number of parallel copies of them can fire at any marking.

Starting from the primitive $t\colon u \to v$ — to be read as t performs a computation *consuming* the tokens in u and *producing* the tokens in v — the notion of *firing* and *state space* is extended to PT nets as follows. A finite number of transitions can be composed in parallel to form a *step*, which, therefore, is a finite multiset of transitions. We write $u \,[\![\alpha\rangle\, v$ to denote a step α with source u and target v. The set $S(N)$ of steps of N is generated by the rules:

$$\frac{u \in \mu(S)}{u \,[\![0\rangle\, u} \qquad \frac{t\colon u \to v \text{ in } N \text{ and } w \text{ in } \mu(S)}{(u+w)\,[\![t\rangle\,(v+w) \text{ in } S(N)} \qquad \frac{u \,[\![\alpha\rangle\, v \text{ and } u' \,[\![\beta\rangle\, v' \text{ in } S(N)}{(u+u')\,[\![\alpha+\beta\rangle\,(v+v') \text{ in } S(N)}.$$

A finite number of steps from the initial marking can be sequentially composed thus yielding a *step sequence*. The set of step sequences, denoted $SS(N)$, is given by:

$$\frac{u_N\ [\![\alpha_0\rangle\ v_0, \ldots, u_n\ [\![\alpha_n\rangle\ v_n \text{ in } S(N) \text{ and } u_i = v_{i-1},\ i = 1, \ldots, n}{u_N\ [\![\alpha_0\rangle [\![\alpha_1\rangle \cdots [\![\alpha_n\rangle\ v_n \text{ in } SS(N)}.$$

The set $R(N)$ of *reachable markings* of N is the set of markings which are target of some step sequence, i.e.,

$$R(N) = \{v \mid \exists\, (u_N\ [\![\alpha_0\rangle \cdots [\![\alpha_n\rangle\ v) \text{ in } SS(N)\}.$$

An important class of nets is that of *occurrence nets*, introduced originally in [**57**] by 'unfolding' safe nets into a suitable 'collection' of their processes (as defined for elementary net systems in Section 1.2). Occurrence nets are elementary nets with a nicely stratified structure whose minimal elements constitute the initial marking.

DEFINITION. An *occurrence net* is an semiweighted net Θ such that $pre_\Theta(t)$ is a set for all $t \in T_\Theta$, and

i) $\forall a \in S_\Theta,\ |{}^\bullet a| \leq 1$, and $a \in u_\Theta$ if and only if ${}^\bullet a = \varnothing$;

ii) $\prec$ is irreflexive, where $\prec$ is the transitive closure of the (flow) relation

$$\prec^1 = \{(a,t) \mid a \in S_\Theta, t \in T_\Theta, t \in a^\bullet\} \cup \{(t,a) \mid a \in S_\Theta, t \in T_\Theta, t \in {}^\bullet a\};$$

moreover, $\forall t \in T_\Theta$, $\{t' \in T_\Theta \mid t' \prec t\}$ is finite;

iii) the binary '*conflict*' relation $\#$ on $T_\Theta \cup S_\Theta$ is irreflexive, where

$$\forall t_1, t_2 \in T_\Theta,\ t_1 \#_m t_2 \iff pre_\Theta(t_1) \cap pre_\Theta(t_2) \neq \varnothing\ \ \&\ \ t_1 \neq t_2,$$

$$\forall x, y \in T_\Theta \cup S_\Theta,\ x \# y \iff \exists t_1, t_2 \in T_\Theta,\ t_1 \#_m t_2\ \ \&\ \ t_1 \preceq x\ \ \&\ \ t_2 \preceq y,$$

and $\preceq$ is the reflexive closure of $\prec$.

This defines the category Occ as a full subcategory of SWNets.

Elements x and y of Θ by are *concurrent*, x co y, if they are related neither by $\preceq$ nor by $\#$. For X a set of elements, we write $\mathrm{Co}(X)$ to mean that x co y for all $x, y \in X$.

Thanks to the stratified structure of the nets in Occ, for them we can define the concepts of *depth* of elements and, consequently, of *subnet of depth n*. Essentially, this will allow us to work on such nets by induction.

DEFINITION. Let Θ be a net in Occ. The *depth* of elements in $T_\Theta \cup S_\Theta$ is defined inductively by:

- $\mathrm{depth}(b) = \begin{cases} 0 & \text{if } b \in u_\Theta; \\ \mathrm{depth}(t) & \text{if } {}^\bullet b = \{t\}; \end{cases}$
- $\mathrm{depth}(t) = \max\{\mathrm{depth}(b) \mid b \prec t\} + 1.$

DEFINITION. Given a net Θ in Occ define its *subnet* of depth n, $\Theta^{(n)}$, as

- $T_{\Theta^{(n)}} = \{t \in T_\Theta \mid \mathrm{depth}(t) \leq n\}$;
- $S_{\Theta^{(n)}} = \{b \in S_\Theta \mid \mathrm{depth}(b) \leq n\}$;
- $pre_{\Theta^{(n)}}$ and $post_{\Theta^{(n)}}$ are the restrictions of pre_Θ and $post_\Theta$ to $T_{\Theta^{(n)}}$;
- $u_{\Theta^{(n)}} = u_\Theta$.

Clearly, $\Theta^{(n)}$ is a net in Occ, whenever Θ is such. For each $n \leq m$ there exists a morphism $in_{n,m}\colon \Theta^{(n)} \to \Theta^{(m)}$ whose components are both set inclusions. In the following we shall call such net morphisms simply *inclusions*. Observe that, if $\langle f,g\rangle\colon \Theta_0 \to \Theta_1$ is an inclusion, we obviously have $u_{\Theta_0} = u_{\Theta_1}$ and, for each $t \in T_{\Theta_0}$, $pre_{\Theta_0}(t) = pre_{\Theta_1}(t)$ and $post_{\Theta_0}(t) = post_{\Theta_1}(t)$.

The sequence of nets $\Theta^{(n)}$, $n \in \omega$, can be seen as a sequence of finite approximations which, together with the corresponding inclusions, determines Θ uniquely (up to isomorphisms). We shall formalise this intuition by means of the categorical notion of colimit. The following results will allow us to define the unfolding of a net in terms of finite unfoldings, viz., its subnets of depth n. We first need to show that Occ possesses the required colimits. Consider the category $\underline{\omega} = \{0 \to 1 \to 2 \to 3 \cdots\}$ and the class D of diagrams $D\colon \underline{\omega} \to \mathsf{Occ}$ such that $D(n \to n+1) = in_n\colon D(n) \to D(n+1)$ is an inclusion. For such a class we have the following results. The reader is referred to [**42**] for the definition of the categorical concepts involved.

PROPOSITION 2.10. *For any $D \in D$, the colimit of D in Occ exists.*

PROOF. Consider the net $\Theta = (pre_\Theta, post_\Theta\colon T_\Theta \to \mu(S_\Theta), u_\Theta)$ where

$$T_\Theta = \bigcup_n T_{D(n)}, \qquad S_\Theta = \bigcup_n S_{D(n)}, \qquad u_\Theta = u_{D(0)},$$

$$pre^i_\Theta(t) = pre^i_{D(n_t)}(t), \qquad post^i_\Theta(t) = post^i_{D(n_t)}(t),$$

where n_t above denotes any $n \in \omega$ such that $t \in T_{D(n)}$.

Clearly, Θ is well-defined, is a net, and belongs to Occ. Then taking for each $n \in \omega$ $\mu_n\colon D(n) \to \Theta$ to be the obvious inclusion, we have that μ is the limiting cocone. □

PROPOSITION 2.11. *Given a net Θ in Occ, let $D_\Theta\colon \underline{\omega} \to \mathsf{Occ}$ be the functor such that $D_\Theta(n) = \Theta^{(n)}$ and $D_\Theta(n \to n+1) = in_{n,n+1}\colon \Theta^{(n)} \to \Theta^{(n+1)}$. Then $\Theta = \mathrm{Colim}(D_\Theta)$.*

PROOF. It is enough to observe that the colimit construction for diagrams in D in the proof of the previous proposition gives a family $\mu_n\colon D(n) \to \Theta$, $n \in \omega$, where $\mu_n\colon \Theta^{(n)} \to \Theta$ is the inclusion of $\Theta^{(n)}$ in Θ. □

2.4. Unfolding semiweighted nets. We are now ready to define the *unfolding* of SW nets in terms of occurrence nets and show that it is a functor from SWNets to Occ which is right adjoint to the inclusion of Occ in SWNets.

We start by giving the object component of such a functor. To this end, given a net N, we define a sequence occurrence nets, whose nth element approximates the unfolding of N up to depth n, i.e., it reflects *all* the possible behaviours of N up to (step) sequences of length at *most* n. Clearly, the unfolding of N will be defined as the colimit of an appropriate $\underline{\omega}$-diagram built on the sequence of approximating nets.

The purpose of the following inductive definition is to generate all the possible instances of places and transitions of N by decorating them with their '*history*'. Places in the approximating nets represent instances of places of N: precisely, they are pairs (x,b), where $b \in S_N$ and x is a set encoding the history of this particular instance of b. Analogously, the transitions are pairs (B,t) where $t \in T_N$ and the set B represents the history of the instance of t.

DEFINITION. Let $N = (pre_N, post_N \colon T_N \to \mu(S_N), u_N)$ be a net in SWNets. We define the nets $U(N)^{(k)} = (pre_k, post_k \colon T_k \to \mu(S_k), u_k)$, for $k \in \omega$, where (cf. Figure 7)

- $S_0 = \{ (\varnothing, b) \mid b \in u_N \}$;
- $T_0 = \varnothing$, and pre_0 and $post_0$ with the obvious definitions;
- $u_0 = \sum S_0$;

and for $k > 0$,

- $$\frac{B = \{ (x_j, b_j) \mid j \in J \} \subseteq S_{k-1}, \quad \mathrm{Co}(B), \quad \sum_{j\in J} b_j = pre_N(t) \text{ for } t \in T_N}{(B,t) \in T_k \quad \text{and} \quad pre_k(B,t) = \sum B}$$
- $$\frac{t_0 = (B,t) \in T_k, \quad post_N(t) = \sum_{j\in J} b_j}{\big(\{t_0\}, b_j\big) \in S_k, \ \forall j \in J, \quad \text{and} \quad post_k(t_0) = \sum_j \big(\{t_0\}, b_j\big)}$$
- $u_k = \sum_j (\varnothing, b_j) = \sum S_0 = u_0$.

Therefore $U(N)^{(0)}$ consists of the initial marking of N, and, informally speaking, $U(N)^{(n+1)}$ is obtained, inductively, by generating a new transition for each possible subset of *concurrent* places of $U(N)^{(n)}$ whose corresponding multiset of places of N is the source of some transition t of N; the target of t is then decorated with its history and added to $U(N)^{(n+1)}$.

Clearly, we shall take $U(N)$ to be the colimit of the sequence of the $U(N)^{(n)}$, $n \in \omega$. To do that, we first need to prove the following lemma.

LEMMA 2.12. *For all $n \in \omega$, $U(N)^{(n)}$ is an occurrence net of depth n. Moreover, for each $n \in \omega$ there is an inclusion $in_n \colon U(N)^{(n)} \to U(N)^{(n+1)}$.*

PROOF. That $U(N)^{(n)}$ has depth n and that there exists an inclusion from $U(N)^{(n)}$ to $U(N)^{(n+1)}$ is obvious from the definition. We have to show that $U(N)^{(n)}$ is an occurrence net. For each $t \in T_n$, $pre_n(t)$ and $post_n(t)$ are multisets where all the elements have multiplicity one, i.e., sets. The same happens for u_n.

i) For each $(x,b) \in S_n$, ${}^\bullet(x,b) = x$ which is either the empty set or a singleton. So $|{}^\bullet(x,b)| \leq 1$. Moreover, $(x,b) \in u_n$ if and only if $x = \varnothing$ if and only if ${}^\bullet(x,b) = \varnothing$.

ii) By definition of $U(N)^{(n)}$, whenever $x \prec^1 y \prec^1 z$, $\mathrm{depth}(z) = \mathrm{depth}(x) + 1$. Since $x, z \in T_n$ or $x, z \in S_n$ implies that there exists at least one y such that $x \prec y \prec^1 z$ we have $\mathrm{depth}(x) < \mathrm{depth}(z)$. So $x \neq z$ and $\prec$ is irreflexive. Observe that this, together with (*i*), implies that, in each reachable marking, every place has multiplicity at most one. In fact, since that happens in u_n, since each place has only one pre-event and each transition occurs at most once in any computation, there is no way to generate multiple tokens in a place. Moreover, $\{t' \in T_n \mid t' \prec t\}$ is obviously finite for all $t \in T_n$.

iii) Recall that $x \# x$ if and only if $\exists t, t' \in T_n,\ t \neq t'$ and $t \#_m t'$ such that $t \preceq x$ and $t' \preceq x$. So, by (*i*), x cannot be a place, otherwise we would have backward branching. This means that there exist $b, b' \in pre_n(x)$, $b \neq b'$ such that $b \mathrel{\mathrm{co}} b'$, i.e., $x = (B,t)$ with *not* $\mathrm{Co}(B)$, that is impossible.

The other conditions of occurrence nets obviously hold. □

DEFINITION. We define $U(N)$ to be the colimit of the diagram $D\colon \underline{\omega} \to \mathsf{DecOcc}$ such that $D(n) = U(N)^{(n)}$ and $D(n \to n+1) = in_n$. By Lemma 2.12 D belongs to $\mathcal{D}$ and so, by Proposition 2.10, the colimit exists and is a decorated occurrence net.

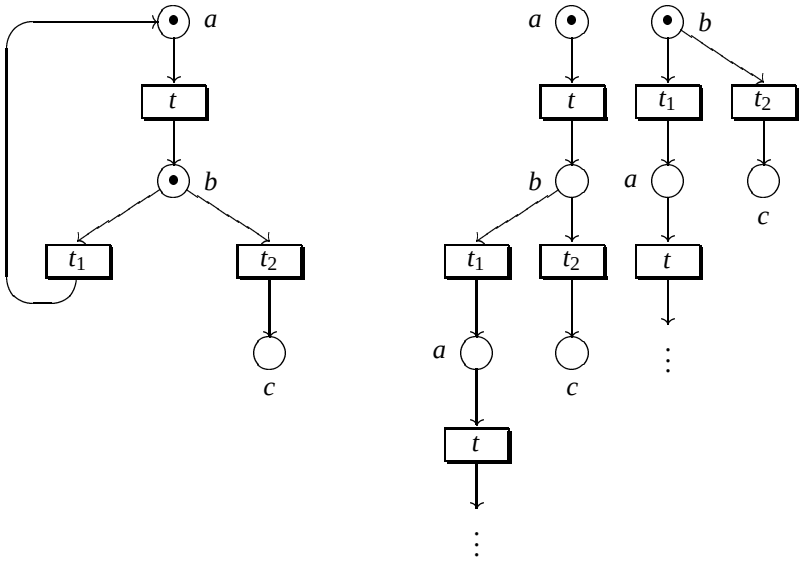


FIGURE 7. An SW net N and (part of) its unfolding $U(N)$

The correspondence between elements of the unfolding and elements of the original net is formalised by the folding morphism, which will also be the counit of the adjunction.

PROPOSITION 2.13. *Consider the map* $\varepsilon\colon U(N) \to N$ *defined by*

- $\varepsilon_t(B,t) = t$;
- $\varepsilon_p(0) = 0$;
- $\varepsilon_p(\sum_i (x_i, y_i)) = \sum_i y_i$.

Then, ε_N *is a morphism in* SWNets, *called the folding of* $U(N)$ *into* N.

PROOF. Since the transitions of $U(N)$ are of the form $t_0 = (B,t)\colon \sum B \to \sum C$, where $B = \{(x_j, b_j) \mid j \in J\} \subseteq S_{U(N)}$, $C = \{(\{t_0\}, c_k) \mid k \in K\}$, $t \in T_N$, $\sum_{j\in J} b_j = pre_N(t)$, and $\sum_{k\in K} c_k = post_N(t)$, we immediately obtain

$$\varepsilon_p(pre_{U(N)}(B,t)) = pre_N(\varepsilon_t(B,t)),$$

and analogously for *post*. Since $u_{U(N)} = \sum_{b\in S_N} u_N(b) \cdot (\varnothing, b)$, we have $\varepsilon_p(u_{U(N)}) = \sum_{b\in S_N} u_N(b) \cdot b = u_N$. □

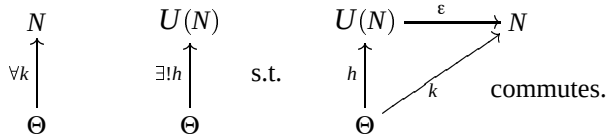
The next lemma is the final ingredient we need to prove that $U(\_)$ is right adjoint to the inclusion. The missing details can be found in [**48**].

LEMMA 2.14. *Let* Θ_0 *and* Θ_1 *be occurrence nets and let* $f\colon \Theta_0 \to \Theta_1$ *be a morphism. Then, for each* $t_0 \in T_{\Theta_0}$, *we have* $\mathrm{Co}\left(pre_{\Theta_0}(t_0)\right)$ *and* $\mathrm{Co}\left(f_p(pre_{\Theta_0}(t_0))\right)$.

PROOF. Since, by definition of occurrence nets, $\{t' \preceq t\}$ is finite, we have *not* $\mathrm{Co}(pre_{\Theta_0}(t_0))$ iff $\exists b, b' \in pre_{\Theta_0}(t_0)$ such that $b \# b'$. This would mean that $\exists t, t' \in T_{\Theta_0},\ t \neq t'$ and $t \#_m t'$ such that $t \preceq b$ and $t' \preceq b'$. Thus, since $t \preceq t_0$ and $t' \preceq t_0$, we would have $t_0 \# t_0$ which is impossible since Θ_0 is a occurrence net. Furthermore, $f_p(pre_{\Theta_0}(t_0)) = pre_{\Theta_1}(f_t(t_0))$, which is the pre-set of a transition of a occurrence net and so, by the first part of this proposition, $\mathrm{Co}(f_p(pre_{\Theta_0}(t_0)))$. □

THEOREM 2.15. *The pair* $\langle \hookrightarrow, U(\_) \rangle : \mathsf{Occ} \rightharpoonup \mathsf{SWNets}$ *constitutes an adjunction.*

PROOF. Let N be a SW net and $U(N)$ its unfolding. We show that the folding $\varepsilon\colon U(N) \to N$ is universal, i.e., for any occurrence net Θ and any morphism $k\colon \Theta \to N$ in SWNets, there exists a unique $h\colon \Theta \to U(N)$ in Occ such that $k = \varepsilon \circ h$.



Consider the diagram in Occ given by $D_\Theta(n) = \Theta^{(n)}$, the subnet of Θ of depth n and $D_\Theta(n \to n+1) = in_n\colon \Theta^{(n)} \to \Theta^{(n+1)}$. We define a sequence of morphisms of nets $h_n\colon \Theta^{(n)} \to U(N)$, such that for each n, $h_n = h_{n+1} \circ in_n$. Since $\Theta = \mathrm{Colim}(D_\Theta)$, there is a unique $h\colon \Theta \to U(N)$ such that $h \circ \mu_n = h_n$ for each n. At the same time, we show that

$$\forall n \in \omega,\ k \circ \mu_n = \varepsilon \circ h_n$$

and that the h_n form the unique sequence of morphisms $h_n\colon \Theta^{(n)} \to U(N)$ such that this holds. Thus we have

$$\forall n \in \omega,\ k \circ \mu_n = \varepsilon \circ h \circ \mu_n$$

and, by the universal property of the colimit, $k = \varepsilon \circ h$. To show the uniqueness of h, let h' be such that $k = \varepsilon \circ h'$. Then we have $k \circ \mu_n = \varepsilon \circ h' \circ \mu_n$. But h_n is the unique morphism for which this happens. Therefore, for each n, $h_n = h' \circ \mu_n$ and so, again by the universal property of the colimit, $h = h'$.

Let us now define h_n and therefore $h\colon \Theta \to U(N)$, and show that the h_n, $n \in \omega$, form the unique sequence of morphisms for which (1) above holds.

depth 0. This is a special case of the inductive step, and we omit it (see [**48**].)

depth n+1. Let us suppose that we have defined $h_n\colon \Theta^{(n)} \to U(N)$ and that it is a morphism. Suppose that for each $m \leq n$, h_m is the unique morphism such that $\varepsilon \circ h_m = k \circ \mu_m$. Let h_{n+1} be h_n on the elements of depth less or equal to n. Now, we define h_{n+1} on the elements of depth $n+1$. Let $t_1 \in T_\Theta$ such that $\mathrm{depth}(t_1) = n+1$ and $k(t_1) = t$. Since $pre_\Theta(t_1)$ is a set of elements of depth less or equal to n, $h_n(pre_\Theta(t_1))$ is defined. Since h_n is a morphism, by Lemma 2.14, we have $\mathrm{Co}(h_n(pre_\Theta(t_1)))$. Moreover, since $\varepsilon \circ h_n = k \circ \mu_n$, we have that

$$pre_N(t) = k(pre_\Theta(t_1)) = \varepsilon \circ h_n(pre_\Theta(t_1)) = \textstyle\sum_{j \in J} b_j, \quad \text{for } J \text{ such that } \{(x_j, b_j) \mid j \in J\} = h_n(pre_\Theta(t_1)).$$

Therefore $t_0 = \big(h_n(pre_\Theta(t_1)), t\big) = \big(h_{n+1}(pre_\Theta(t_1)), t\big) \in T_{U(N)}$. Now, since h_{n+1} has to make the diagram commute, $h_{n+1}(t_1)$ must be of the form (B, t) and, since it has to be a

morphism, it must be $pre_{U(N)}((B,t)) = \sum B = h_{n+1}(pre_\Theta(t_1))$. Therefore $h_{n+1}(t_1) = t_0$. Observe that there is only one choice for $h_{n+1}(t_1)$, given k and h_n by inductive hypothesis. Obviously, $\varepsilon \circ h_{n+1}(t_1) = t = k(t_1) = k \circ \mu_{n+1}(t_1)$. Now, let $post_\Theta(t_1) = \sum_i a_i$. Suppose that $k(a_i) = \sum_j m_j^i b_j^i$. Since $k(post_\Theta(t_1)) = post_N(k(t_1))$, we have $post_N(k(t_1)) = \sum_{i,j} m_j^i b_j^i$, with all b_j^i distinct. It follows that $m_j^i = 1$ and thus in $U(N)$ we have the places $\bigcup_{i,j}\{(\{t_0\}, b_l^j)\}$. We define

$$h_{n+1}(a_i) = \sum_j \{(\{t_0\}, b_l^j)\}$$

and, as before, conclude that $\varepsilon \circ h_{n+1}(a_i) = \sum_j b_j^i = k(a_i) = k \circ \mu_{n+1}(a_i)$.

Observe that $h_{n+1}(a_i)$ is completely determined by k and by the conditions of decorated occurrence net morphisms.

Finally, we have to show that h_{n+1} is a morphism $\Theta^{(n+1)} \to U(N)$. But this task is really trivial because, by its own construction, h_{n+1} preserves source, target and initial marking. □

THEOREM 2.16. *$\langle \hookrightarrow, U(\_)\rangle$ is a coreflection* $\mathsf{Occ} \rightharpoonup \mathsf{SWNets}$.

It is worth observing that when N is a *safe* net, $U(N)$ is (isomorphic to) the unfolding of N defined in [**57, 94**]. In other words, $\langle \hookrightarrow, U(\_)\rangle$ restricts to the coreflection $\mathsf{Occ} \rightharpoonup \mathsf{Safe}$ presented in *loc. cit*.

Our final step in relating SW nets to event structures and domains is to fill the gap between occurrence nets and event structures. To this aim, we conclude this section recalling the definitions of the functors forming the coreflection $\langle N, E\rangle \colon \mathsf{E} \rightharpoonup \mathsf{Occ}$ as studied in [**57, 94**].

DEFINITION. Let Θ be an occurrence net. Then, $E(\Theta)$ is the event structure

$$(T_\Theta, \preceq, \#),$$

where $\preceq$ and $\#$ are the restriction to T_Θ, the set of transitions of Θ, of, respectively, the *flow* ordering $\preceq_\Theta$ and the *conflict* relation $\#_\Theta$ implicitly defined by Θ.
For $f \colon N_0 \to N_1$ a morphism in Occ, we take $E(f)$ to be $f_t \colon E(N_0) \to E(N_1)$, which clearly gives a functor $E \colon \mathsf{Occ} \to \mathsf{E}$.

Consider now an event structure $(E, \leq, \#)$. As a notation, for a subset A of E, we write $\#A$ to mean that $a \# a'$ for all $a \neq a' \in A$ Similarly, $e < A$ means that $e < e'$ for all $e' \in A$. Then, we can define define

$$N(E, \leq, \#) = (pre, post \colon E \to \mu(M \cup B), \sum M),$$

where

$$\begin{aligned} M &= \{(\varnothing, A) \mid A \subseteq E \text{ and } \#A\}; \\ B &= \{(e, A) \mid e \in E,\ \#A \text{ and } e < A\}; \\ pre(e) &= \{(c, A) \in B \cup M \mid e \in A\}; \\ post(e) &= \{(e, A) \in B\}. \end{aligned}$$

Then, we have the following.

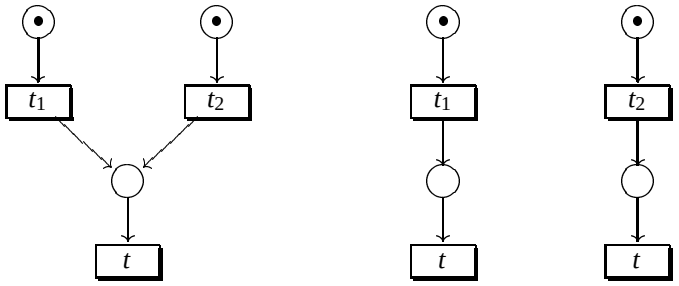
THEOREM 2.17. *For each event structure E, $N(E)$ is an occurrence net such that $EN(E) = E$. Moreover, N extends to a functor that is left adjoint to E, so yielding a coreflection whose unit is the identity function $E \to EN(E) = E$.*

PROOF. See [**95**]. □

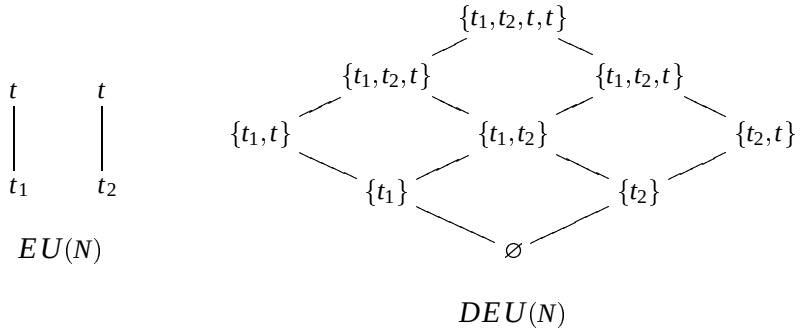
Although $NE(\Theta)$ and Θ are not isomorphic in Occ, it is worth observing that they are 'behaviourally' such. In particular, an inspection of the definition will prove that $NE(N)$ is *place-saturated* version of N, i.e., that is obtained by adding to N all the places that it is possible to add without duplicating any or altering the behaviour.

The coreflection between Occ and SWNets can of course be composed with the coreflection between Occ and E and with the equivalence (*a fortiori* a coreflection!) of D and E. The following example shows the structures associated by this chain of coreflections to a simple, well-known, (non-safe) semiweighted net.

EXAMPLE.



Observe that the unfolding contains two concurrent copies of t. These correspond to the occurrences of t in two possible 'causal contexts', namely t caused by t_1 and t caused by t_2. In the picture below, which shows the event structure and the prime algebraic domain associated to N, the four events so arising are labelled by the transition they correspond to.



3. Petri Nets and Transition Systems

Looking back at the notion of case graph CG_N in Section 1.1, we certainly know that — to a certain extent and with a certain degree of precision — nets can be described by some sort of transition system. The details, however, are far from trivial and in this section we set out to study them by presenting a coreflection between elementary net systems and so-called *elementary transition systems* due to [**59, 60**].

Intuitively, this result identifies a category of transition systems which may be seen as an abstract behavioural characterisation of elementary net systems, formally stated as a coreflection 'embedding' a category of certain transition systems into EN. This was the first of a series of behavioural characterisation results for nets, in the sense that several similar coreflections have been shown for more general classes of nets, see e.g. [**97, 53**]. We restrict ourselves here to elementary nets, though we believe that the ideas from [**53**] generalise smoothly to provide a corresponding result for SWNets.

3.1. Transition systems. Transition systems are a frequently used model of parallel processes. They consist of a set of states, with an initial state, together with transitions between states which are labelled to specify the kind of events they represent.

DEFINITION. A *transition system* is a structure $(S, i, L, tran)$ where

- S is a set of *states* with *initial state* i,
- L is a set of *labels*,
- $tran \subseteq S \times L \times S$ is the *transition relation.*

A transition $(s, a, s') \in tran$ is often indicated as $s \xrightarrow{a} s'$.

DEFINITION. Let $T_0 = (S_0, i_0, L_0, tran_0)$ and $T_1 = (S_1, i_1, L_1, tran_1)$ be transition systems. A *morphism* $f\colon T_0 \to T_1$ is a pair $f = (\sigma, \lambda)$ where

- $\sigma\colon S_0 \to S_1$, a function between sets of states,
- $\lambda\colon L_0 \to L_1$, a function between sets of labels,

are such that $\sigma(i_0) = i_1$ and $(s, \alpha, s') \in tran_0$ implies $(\sigma(s), \lambda(\alpha), \sigma(s')) \in tran_1$.

The intention behind the definition of morphism is, as usual in this paper, to guarantee that the relevant behavioural notions are preserved. In case of transition systems this amounts to saying that the effect of a transition $s \xrightarrow{\alpha} s'$ in T_0 is matched in T_1 by the effect of a corresponding transition $\sigma(s) \xrightarrow{\lambda(\alpha)} \sigma(s')$. To complete the definition, morphism are required to preserve initial states.

Transition system and their morphisms form a *category* TS in which composition is defined componentwise — i.e., composing $(\sigma, \lambda)\colon T_0 \to T_1$ and $(\sigma', \lambda')\colon T_1 \to T_2$ yields $(\sigma' \circ \sigma, \lambda' \circ \lambda)\colon T_0 \to T_2$ — and the identity morphism of T is $(1_{T_S}, 1_{T_L})$, where 1_X is the identity function on the set X.

The definition below refines the notion of case graph of an elementary net systems by recasting it in terms of transition systems. In the following, for $N = (B, E, F, c_{in})$ an elementary net system, we shall say that an event $e \in E$ is '*active*' if it has an occurrence $(c, e, c') \in \longrightarrow_{C_N}$.

DEFINITION. Let $N = (B, E, F, c_{in})$ an elementary net system. The *reachable case graph* associated with N is the transition system $TS_N = (C_N, c_{in}, E_N, \longrightarrow_{C_N})$, where E_N is the set of *active events* of N.

The reachable case graph, or simply the transition system, associated with the net system of Figure 1 is presented in Figure 8.

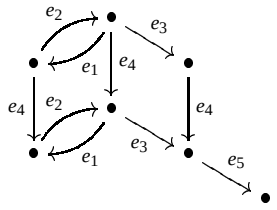


FIGURE 8

With the definition of TS given above, it is quite easy to see that the case graph construction of elementary net systems extends to a *functor* from EN to TS (see below). In order for this functor to be part (left adjoint) of a coreflection, however, we first need to identify those transition systems that arise as reachable case graphs of elementary nets. Such transition systems can be characterised in terms of *regions* by Ehrenfeucht and Rozenberg [**20**].

DEFINITION. Let $T = (S, i, L, tran)$ be a transition system. Then $r \subseteq S$ is a *region* of T if and only if the following two conditions are satisfied for all $(s_0, e, t_0), (s_1, e, t_1) \in tran$:

- ▷ if $s_0 \in r$ & $t_0 \notin r$, then $s_1 \in r$ & $t_1 \notin r$;
- ▷ if $s_0 \notin r$ & $t_0 \in r$, then $s_1 \notin r$ & $t_1 \in r$.

So, a region is a subset of states, such that for all $e \in L$, *all* occurrences of an e-labelled transitions have the same '*crossing*' relationship with respect to r (leaving or entering). As an example, consider again the transition system from Figure 8. It is easy to check, for instance, that the singleton set consisting of the state entered by the e_5-labelled transition is a region, whereas the singleton set consisting of the initial state is *not*.

It follows trivially that both $\varnothing$ and S are regions of T. They are called the *trivial regions*, and we use R_T as notation for the set of *non-trivial* regions of T. Also, for $s \in S$, we use R_s to denote the set of non-trivial regions of T containing s. More precisely

$$R_s = \{r \mid s \in r \text{ and } r \in R_T\}.$$

Yet another crucial notation concerns the set of *pre-* and *post-regions* of a label. Formally, for $e \in L$, we define

the *pre-regions* of e: $\quad {}^\circ e = \{r \in R_T \mid \exists (s, e, s') \in tran.\ s \in r \,\&\, s' \notin r\}$;

the *post-regions* of e: $\quad e^\circ = \{r \in R_T \mid \exists (s, e, s') \in tran.\ s \notin r \,\&\, s' \in r\}$.

Some useful properties of regions can now be stated.

PROPOSITION 3.1. *Let $T = (S, i, L, tran)$ be a transition system. Then*

i) *$r \subseteq S$ is a region if and only if $\overline{r} = S \setminus r$ is a region.*
ii) *Let $e \in L$. Then $e^\circ = \{\overline{r} \mid r \in {}^\circ e\}$.*
iii) *Let $(s, e, s') \in tran$. Then $R_s \setminus R_{s'} = {}^\circ e$ and $R_{s'} \setminus R_s = e^\circ$.*

Another important property of regions is that they are preserved by inverse morphism.

PROPOSITION 3.2. *Let $f = (\sigma, \lambda)$ be a morphism from T_0 to T_1. If r is a region of T_1 then $\sigma^{-1}(r)$ is a region of T_0. Furthermore, for every $e \in L_{T_0}$, we have $\sigma^{-1}(r) \in {}^\circ e$ (respectively $\sigma^{-1}(r) \in e^\circ$) if and only if $r \in {}^\circ\lambda(e)$ (respectively to $r \in \lambda(e)^\circ$).*

Using the notion of regions, we may now define the transition systems which arise as case graphs of elementary nets, following [**20**].

DEFINITION. A transition system $T = (S, i, L, tran)$ is said to be *elementary* if it satisfies the following conditions:

(A1) $\forall e \in L.\ \exists (s, e, s') \in tran$, where $s \neq s'$;

(A2) $\forall s \in S \setminus \{i\}.\ \exists s_0, s_1, \ldots, s_{n+1} \in S$ and $e_0, e_1, \ldots, e_n \in L$ such that
$$i = s_0, \quad s_{n+1} = s, \quad \text{and} \quad (s_i, e_i, s_{i+1}) \in tran \text{ for } 0 \leq i \leq n;$$

(A3) $\forall e, e' \in L.\ {}^\circ e = {}^\circ e' \implies e = e'$;

(A4) $\forall s, s' \in S.\ R_s = R_{s'} \implies s = s'$;

(A5) $\forall s \in S.\ \forall e \in L.\ {}^\circ e \subseteq R_s \ \&\ e^\circ \cap R_s = \varnothing \implies \exists (s, e, s') \in tran$.

We let ETS denote the full subcategory of TS with elementary transition systems as objects.

Notice that the axioms presented here are a cleaned up version of the axioms presented in [**59**], but in this context, the two sets of axioms can be proved to be equivalent. The two first axioms just state that each label and each state can be *reached* form the initial state by a finite number of transitions. The next two enforce certain obvious regional *separation* properties, for labels and states respectively. And the final axiom enforces a relationship between regional properties of labels and the transition relation.

3.2. Elementary nets and transition systems. It turns out that ETS is exactly the category of case graphs associated with EN, a result which we shall formalise in this section. For full proofs of the theorems quoted, we refer to [**59**]

THEOREM 3.3. *The construction that maps $N = (B, E, F, c_{in})$ to the its reachable case graph $TS_N = (C_N, c_{in}, E_N, \longrightarrow_{C_N})$ extends to a functor nt from EN to ETS.*

PROOF. Easy. It follows essentially from Proposition 1.5. □

More importantly, *nt* is the left adjoint component of a coreflection whose a right adjoint, $tn\colon \mathsf{ETS} \to \mathsf{EN}$, operates on objects as detailed below.

DEFINITION. Let $T = (S, i, L, tran)$ be a transition system. The *net structure* associated to T is the net $N_T = (R_T, L, F_T, R_i)$, where

- $(r, e) \in F_T$ if and only if $r \in {}^\circ e$;
- $(e, r) \in F_T$ if and only if $r \in e^\circ$.

Essentially, this definition 'reads' a transition systems as the case graph of some net. Thus, as expected, the labels of T become events of N_T and the regions become the conditions, with the pre- and post-sets obviously given by pre- and post-regions.

PROPOSITION 3.4. *The net structure $N_T = (R_T, L, F_T, R_i)$ associated to the transition system $T = (S, i, L, tran)$ is an elementary net system.*

PROOF. Easy. □

As anticipated, N_T is the object part of the right adjoint of *nt*.

THEOREM 3.5. *The construction that maps a transiton system T to its net structure N_T extends to a functor tn from* ETS *to* EN.

PROOF. A morphism $(\sigma, \lambda) \colon T_0 \to T_1$ is mapped to $tn(\sigma, \lambda) = (\beta, \lambda) \colon N_{T_0} \to N_{T_1}$, where $\beta^{-1}(r_1) = \sigma^{-1}(r_1)$. With this definition, the proof essentially follows from Proposition 3.2. □

THEOREM 3.6. *Functors tn and nt form a coreflection* ETS $\rightharpoonup$ EN *with tn as left adjoint.*

PROOF. This proof is rather involved. We refer the interested reader to [**59**]. □

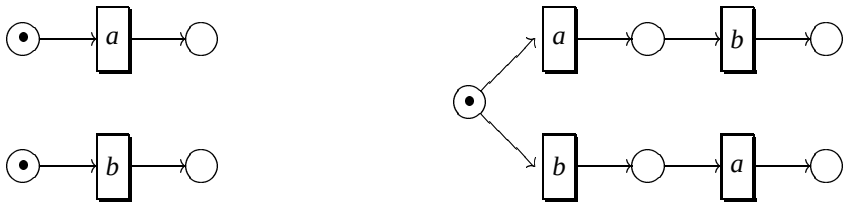
One interesting part of the construction is that, exactly as in the case of event structures, $tn \circ nt(N)$ is a '*condition-saturated*' version of N, in the formal sense that any further addition of a condition will change the behaviour (case graph) of the net. As a consequence, the saturated net will have, for each condition b, a condition representing the *complement* of b — formally the set complement $\bar{r}$ of the region r associated with b (cf. Proposition 3.1*(i)*). Hence, as a corollary of this construction, we have that any elementary net system is behaviourally equivalent to a contact-free system.

4. Petri Nets and Bisimulations

The coreflections that we studied in the previous sections establish a web of formal relationships between nets and other models, enabling us to place nets in a broader picture of models for concurrency and, often, allowing the translation of concepts to between models. This section aims at providing an illustration of this point. It is an exposition of a the categorical approach to bisimulation obtained from spans of open maps as defined in [**35**] — to which we refer for the missing proofs — with an additional treatment of SWNets in the general picture. The open map approach presented here has also been applied successfully to capture other familiar behavioural equivalences on nets, e.g., Hoare's trace equivalence [**30**] and Milner's weak bisimulation [**49, 50**], both of which may be obtained by slightly changing the notion of path extension from the one presented here [**55**]. Also, the open morphism approach has been applied successfully to different categories of models, e.g., probabilistic systems [**55**], and timed systems [**34**].

Once again, the idea here is to put forward that the categorical view of models for concurrency provides guidelines for definitions of concepts, like behavioural equivalences, consistent across a range of models. In particular, the notion of bisimulation derived for nets comes automatically equipped with a number of essential properties. The categorical approach here contrasts with the more common alternative of searching for a sensible candidate for bisimulation on nets and, having found one of then checking it possesses these essential properties.

4.1. Labelled models and their relationship. Like most models for concurrency, nets [**65**] and asynchronous transition systems [**54**], or more precisely their labelled versions, have been used as models for process languages like CCS, [**50**]. As an illustration, following [**65**], the (strongly bisimilar) CCS expressions $a.nil \mid b.nil$ and $a.b.nil + b.a.nil$ are represented by the following rather different nets.



$a.nil \mid b.nil$ $\qquad$ $a.b.nil + b.a.nil$

There is a general way of introducing labels to models in such a way that one may carry over adjunctions between unlabelled models to their labelled counterparts, following [**35**]. Here we sketch the idea, applicable to the categories of nets and event structures. We assume a category X of structures each of which possesses a distinguished set of events and where morphisms have as a component a function between sets of events. In any such setting, morphisms may be lifted uniformly to a category X_L with labels.

- ▷ The objects of X_L consist of structures (X, l) where X is an object of X, and $l : E \to L$ is a (total) labelling function from E the events of X to the labelling set L.
- ▷ The morphisms of X_L from (X, l) to (X', l') correspond to morphisms $f : X \to X'$ of X of which the event component η preserves labels, i.e. $l' \circ \eta = l$.

Correspondingly, for a set of labels L, we denote the fibres over L in the labelled versions of our categories of nets and event structures by EN_L, SWNets_L and E_L, respectively. Similarly the category of transition systems over label set L, with morphisms having the identity as label component, will be denoted TS_L, and its full subcategory of *synchronisation trees* — that are precisely the tree-'shaped' transition systems — by S_L.

It follows for general reasons [**97**] (and is easy to see) that the coreflection between nets and event structures lift to a coreflection between the labelled versions. The modified adjoints are essentially the adjoints presented in the previous sections, simply carrying the label parts across from one model to the other. Furthermore, this coreflection is part of greater collection [**97, 80, 79**] of which here we are interested in the small portion shown in the diagram below.

$$\mathsf{S}_L \underset{es}{\overset{se}{\hookrightarrow}} \mathsf{E}_L \underset{EU(\_)}{\overset{N(\_)}{\hookrightarrow}} \mathsf{SWNets}_L$$

Concerning synchronisation trees, we remark that, as hinted at in Section 2.1, they can be identified with those labelled event structures having *empty* co-relation, i.e., those whose every two events are either causally dependent or conflicting. This gives

rise to an embedding $se\colon \mathsf{S}_L \hookrightarrow \mathsf{E}_L$ which, actually, admits a coreflection right adjoint whose action on objects yields the tree of event sequences ordered by prefix.

4.2. Path-lifting morphisms. Following [**35**], a computation path represents a particular run or history of a process. For transition systems, a computation path is traditionally taken to be a finite sequence of transitions. For a labelling set L, define the category of *branches* Bran_L to be the full subcategory of transition systems, with labelling set L, with objects those finite synchronisation trees with exactly one (maximal) branch; so the objects of Bran_L are essentially *strings* over alphabet L. A computation path in a transition system T, with labelling set L, can then be represented by a morphism

$$p\colon P \to T$$

in TS_L from an object P of Bran_L.

How should we represent a computation path of a net or an event structure? To take into account the explicit concurrency exhibited by an event structure, it is reasonable to represent a computation path as a morphism from a partial order of labelled events, that is from a *pomset* [**69**]. Observe that pomsets with labels in L can be identified with special kinds of labelled event structures in E_L: those with *empty* conflict relation. Then, define the category of pomsets Pom_L, with respect to a labelling set L, to be the full subcategory of E_L whose objects consist exclusively of *finite* pomsets. A computation path in an event structure E, with labelling set L, is a morphism

$$p\colon P \to E$$

in E_L from an object P of Pom_L.

What about computation paths in nets? The answer is that our embeddings of event structures into nets allow us to view *pomsets as labelled nets*! Let us illustrate the point more concretely. The left adjoint $N(\_)$ of the coreflection $\mathsf{E}_L \to \mathsf{SWNets}_L$ embeds labelled event structures, and so pomsets, in labelled SW nets. This enables us to identify pomsets P in Pom_L with their images $N(P)$ as labelled saturated nets in SWNets_L. Now, we can take a computation path in a net N, with labelling set L, to be a morphism

$$p\colon P \to N$$

in SWNets_L from a 'pomset' P, with labelling set L — where P actually stands for labelled saturated net in SWNets_L corresponding to a pomset. In future, when discussing nets, we will deliberately confuse pomsets with their image in SWNets_L under the embedding.

Generally, assume a category of models M (this can be any of the categories of labelled structures we are considering) and a choice of path category, a subcategory $\mathsf{P} \hookrightarrow \mathsf{M}$ consisting of path objects (these could be branches, or pomsets) together with morphisms expressing how they can be extended. Define a *computation path* in an object X of M to be a morphism

$$p\colon P \to X,$$

in M, where P is an object in P. A morphism $f\colon X \to Y$ in M takes such a path p in X to the path $f \circ p\colon P \to Y$ in Y. The morphism f expresses the sense in which Y simulates X: any computation path in X is matched by the computation path $f \circ p$ in Y.

We might demand a stronger condition of a morphism $f\colon X \to Y$ expressed succinctly in the following *path-lifting condition*: whenever, for $m\colon P \to Q$ a morphism in P, a 'square'

$$\begin{array}{ccc} P & \xrightarrow{p} & X \\ {\scriptstyle m}\downarrow & & \downarrow{\scriptstyle f} \\ Q & \xrightarrow[q]{} & Y \end{array}$$

in M commutes, i.e., $q \circ m = f \circ p$, meaning the path $f \circ p$ in Y can be extended via m to a path q in Y, then there is a morphism p' such that in the diagram

$$\begin{array}{ccc} P & \xrightarrow{p} & X \\ {\scriptstyle m}\downarrow & \overset{p'}{\nearrow} & \downarrow{\scriptstyle f} \\ Q & \xrightarrow[q]{} & Y \end{array}$$

the two 'triangles' commute, i.e., $p' \circ m = p$ and $f \circ p' = q$, meaning the path p can be extended via m to a path p' in X which matches q. When the morphism f satisfies this condition we shall say it is P-*open*.

It is easily checked that P-open morphisms include all the identity morphisms (in fact, all isomorphisms) of M and are closed under composition there; in other words they form a subcategory of M.

For transition systems, Bran_L-open morphisms are already familiar.

PROPOSITION 4.1. *With respect to a labelling set L, the Bran_L-open morphisms of TS_L are the 'zig-zag morphisms' of* [**88**]*, the 'p-morphism' of* [**82**]*, the 'abstraction homomorphisms' of* [**14**]*, and the 'pure morphisms' of* [**5**]*, the 'transition-preserving homomorphisms' of* [**23**]*, i.e., those label-preserving morphisms $(\sigma, 1_L)\colon T \to T'$ on transition systems over labelling set L with the property that for all reachable states s of T*

$$\text{if } \sigma(s) \xrightarrow{a} s' \text{ in } T' \text{ then } s \xrightarrow{a} u \text{ in } T \text{ and } \sigma(u) = s', \text{ for } u \text{ a state of } T.$$

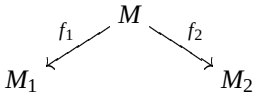
DEFINITION. Let P be a category in a category of models M. Objects X_1, X_2 of M are P-*bisimilar* iff they are in the equivalence generated by being related by a P-open map.

For the interleaving models of transition systems and synchronisation trees with path category P taken to be branches, P-bisimulation coincides with Milner's *strong bisimulation* [**50, 49**].

THEOREM 4.2. *Two transition systems (and so synchronisation trees), over the same labelling set L, are Bran_L-bisimilar iff they are strongly bisimilar in the sense of* [**50**].

In many cases, including the ones considered here, P-bisimilarity between two objects have a particularly simple presentation.

THEOREM 4.3. *If the category* M *has pullbacks, then* M_1 *and* M_2 *are* P*-bisimilar iff there is a span of* P*-open morphisms* f_1, f_2*:*



PROOF. It follows since pullbacks of P-open morphisms are P-open. □

PROPOSITION 4.4. *The categories* S_L, E_L *and* Occ_L *have pullbacks.*

PROOF. One shows that Occ_L has pullbacks. Then, using the facts that right adjoints preserve limits, and pullbacks in particular, and that there are coreflections from categories S_L and E_L to Occ_L, we obtain pullbacks in any of these as images under the right adjoints of the pullback in Occ_L of diagrams transported into Occ_L by the left adjoints. □

We conclude this section presenting a few general facts from [**35**] about how open morphisms and bisimilarity are preserved and reflected by functors, especially when part of a coreflection. For notational simplicity we shall assume the left adjoints of the coreflections are inclusions. It follows that for the coreflections of Section 4.1, in which the categories of models share the same choice of path category, open morphisms and bisimilarity are preserved in both directions of the adjunction.

LEMMA 4.5. *Let* M *be a coreflective subcategory of* X *with* R *right adjoint to the inclusion function* $\mathsf{M} \hookrightarrow \mathsf{X}$ *and* P *a subcategory of* M*.*

i) *A morphism* f *of* M *is* P*-open in* M *if and only if* f *is* P*-open in* X*.*
ii) *The components of the counit* $\varepsilon_X : \mathsf{R}(X) \to X$ *are* P*-open in* X*.*
iii) *A morphism* f *is* P*-open in* X *if and only if* $\mathsf{R}(f)$ *is* P*-open in* M*.*

COROLLARY 4.6. *Let* M *be a coreflective subcategory of* X *with* R *right adjoint to the inclusion functor* $\mathsf{M} \hookrightarrow \mathsf{X}$ *and* P *a subcategory of* M*.*

i) M_1 *and* M_2 *are* P*-bisimilar in* M *if and only if they are* P*-bisimilar in* X*.*
ii) M_1 *and* M_2 *are* P*-bisimilar in* X *if and only if* $\mathsf{R}(M_1)$ *and* $\mathsf{R}(M_2)$ *are* P*-bisimilar in* M*.*

PROOF. *(i)* Directly from *(i)* of Lemma 4.5.

(ii) '*only if*': By Lemma 4.5*(iii)*, a span of open morphisms in X has, as image under R, a span of open morphisms in M. Thus P-bisimilarity of M_1 and M_2 in X implies P-bisimilarity of $\mathsf{R}(M_1)$ and $\mathsf{R}(M_2)$ in M.

'*if*': Suppose $\mathsf{R}(M_1)$ and $\mathsf{R}(M_2)$ in M are P-bisimilar in M via a span of open morphisms $f_1 : M \to \mathsf{R}(M_1)$, $f_2 : M \to \mathsf{R}(M_2)$ in M. By Lemma 4.5*(i)*, f_1 and f_2 form a span of open morphisms in X. The components of the counits of the coreflection $\varepsilon_1 : \mathsf{R}(M_1) \to M_1$ and $\varepsilon_2 : \mathsf{R}(M_2) \to M_2$ are open by Lemma 4.5*(ii)*. Hence the compositions $\varepsilon_1 \circ f_1$ and $\varepsilon_2 \circ f_2$ form a span of open morphisms in X showing the P-bisimilarity of M_1 and M_2 in X. □

4.3. Pom_L-bisimulation for nets. As seen in Lemma 4.1 and Theorem 4.2, for transition systems the general definition of P-open morphism and P-bisimilarity coincide with familiar notions: the equivalence of strong bisimilarity central to Milner's work. Here we explore how the general definitions specialise to the models of event structures and nets, with nonsequential observations in the form of pomsets. We focus our attention on SWNets, but the answers provided follow closely those obtained for other classes of net systems in [**63**].

We start by characterising Pom_L-open morphisms of SWNets_L.

PROPOSITION 4.7. *The Pom_L-open morphisms of SWNets_L are precisely those which satisfy the 'zig-zag' condition of Proposition 4.1 and which, in addition, reflect consecutive independence, i.e., those $f\colon N_1 \to N_2$ satisfying:*

- *f_t is total and label preserving;*
- *whenever $f_p(\mu) \xrightarrow{e'} \nu'$ in N_2, for μ reachable, then there exists $\mu \xrightarrow{e} \nu$ in N_1 such that $f_t(e) = e'$ and $f_p(\nu) = \nu'$;*
- *whenever $\mu \xrightarrow{e} \mu'$ and $\mu' \xrightarrow{e'} \mu''$ in N_1, for μ reachable, and $f_t(e)$ co $f_t(e')$ in N_2, then e co e' in N_1.*

PROOF. Let $f\colon N_1 \to N_2$ be an open morphism in SWNets_L. The function f_t is total and label preserving from definition of morphisms in SWNets_L, and by considering linear pomsets, where causal dependency is a total order, it is clear as in Proposition 4.1 that f satisfies the 'zig-zag' condition. The only nontrivial part is the reflection of consecutive independence. Let μ be a reachable marking and let

$$\mu \xrightarrow{e} \mu' \quad \text{and} \quad \mu' \xrightarrow{e'} \mu'',$$

be two consecutive transitions in N_1. Consider the corresponding transitions

$$f_p(\mu) \xrightarrow{f_t(e)} f_p(\mu') \quad \text{and} \quad f_p(\mu') \xrightarrow{f_t(e')} f_p(\mu'')$$

of N_2, and assume that $f_t(e)$ and $f_t(e')$ are independent in N_2. Assume further that $l(e) = l(f_t(e)) = a$ and $l(e') = l(f_t(e')) = a'$.

Because μ is reachable there is a chain of transitions

$$c_{in} = \mu_0 \xrightarrow{e_1} \mu_1 \xrightarrow{e_2} \cdots \xrightarrow{e_n} \mu_n = \mu$$

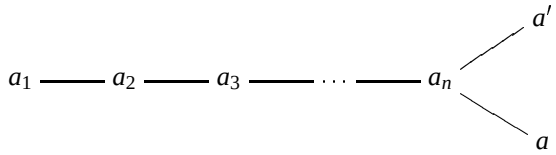
in N_1 from its initial marking c_{in}. Let $l(e_i) = a_i$, and take P to be the linear pomset with $n+2$ elements, ordered and labelled as indicated in the following pomset

$$a_1 \leq a_2 \leq \cdots \leq a_n \leq a \leq a'$$

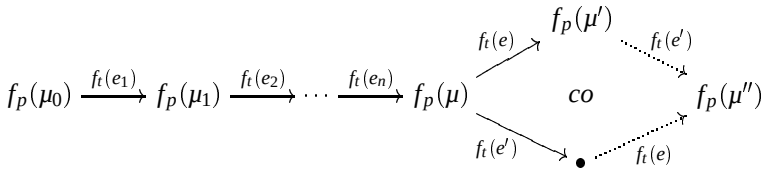
Let $p\colon P \to N_1$ be that morphism in SWNets_L which maps this chain of transitions to

$$\mu_0 \xrightarrow{e_1} \mu_1 \xrightarrow{e_2} \cdots \xrightarrow{e_n} \mu \xrightarrow{e} \mu' \xrightarrow{e'} \mu''$$

in N_1. Let Q be the pomset differing from P only in that the a and a' labelled elements are unordered, i.e., the pomset corresponding to the following graph.



Let $q\colon Q \to N_2$ be that morphism in SWNets_L mapping these transitions to



in N_2, where the dotted arrows represent the concurrency diamond. Letting $m\colon P \to Q$ be the obvious morphism of pomsets, we observe the commuting diagram:

$$\begin{array}{ccc} P & \xrightarrow{p} & X \\ {\scriptstyle m}\downarrow & & \downarrow{\scriptstyle f} \\ Q & \xrightarrow[q]{} & Y \end{array}$$

But f is open, so we obtain a morphism $p'\colon Q \to T$ such that the two 'triangles' commute in the following diagram.

$$\begin{array}{ccc} P & \xrightarrow{p} & X \\ {\scriptstyle m}\downarrow & \overset{p'}{\nearrow} & \downarrow{\scriptstyle f} \\ Q & \xrightarrow[q]{} & Y \end{array}$$

Because p' preserves independence, we see that e and e' are independent in T. So because f is open it satisfies the 'zig-zag' condition and reflects consecutive independence.

The proof in the other direction is omitted; we refer to [**35**] for a similar proof involving asynchronous transitions systems [**4**]. □

We now turn to the question of bisimulation. As shown in [**35**], taking *pomsets* as the path category P yields in the case of *event structures* a reasonable strengthening of a previously studied equivalence: the *history-preserving bisimulation* [**71, 24**]. Its definition below depends on the simple but important remark that a *configuration* of an event structure can be regarded as a *pomset*, with causal dependency relation and labelling got by restricting those of the event structure.

DEFINITION. A *history-preserving bisimulation* between event structures E_1 and E_2 consists of a set H of triples (x_1, f, x_2), where x_i is a configuration of E_i and $f\colon x_1 \cong x_2$ is an isomorphism of pomsets, such that $(\varnothing, \varnothing, \varnothing) \in H$ and, whenever $(x_1, f, x_2) \in H$,

i) if $x_1 \xrightarrow{a} x_1'$ in E_1, then $\exists\, x_2 \xrightarrow{a} x_2'$ in E_2 and $(x_1', f', x_2') \in H$ with $f \subseteq f'$;

ii) if $x_2 \xrightarrow{a} x_2'$ in E_2 then $\exists\, x_1 \xrightarrow{a} x_1'$ in E_1 and $(x_1', f', x_2') \in H$ with $f \subseteq f'$.

We say a history-preserving bisimulation H is *strong* if whenever $(x, f, y) \in H$

I) if $x' \subseteq x$, for x' a configuration of E_1, then $(x', f', y') \in H$, for $f' \subseteq f$ and $y' \subseteq y$;

II) if $y' \subseteq y$, for y' a configuration of E_2, then $(x', f', y') \in H$, for $f' \subseteq f$ and $x' \subseteq x$.

THEOREM 4.8. *Let E_1, E_2 be event structures with labelling sets L. The following are equivalent:*

i) *E_1 and E_2 are Pom_L-bisimilar in E_L.*
ii) *E_1 and E_2 are strong history-preserving bisimilar.*

PROOF. See [**63**]. □

Via the coreflection between event structures and Petri nets, from Corollary 4.6 we can draw characterisations of Pom_L-bisimilarity on nets.

THEOREM 4.9. *Let N_1 and N_2 be nets with labelling sets L. The following are equivalent.*

i) *The nets N_1 and N_2 are Pom_L-bisimilar in SWNets_L.*
ii) *The unfoldings to event structures $\mathcal{EU}(N_1)$ and $\mathcal{EU}(N_2)$ are strong history-preserving bisimilar.*

So, for general reasons, the notion of bisimilarity for nets agrees with the notion of bisimilarity for the associated case graphs and unfoldings (where it amounts to strong history-preserving bisimilarity). Results expressing agreements of this kind would probably be required of any notion of bisimilarity, and, without the help of some categorical machinery, would seem to require separate proofs. Of course, having characterised Pom_L-bisimilarity on nets as strong history-preserving bisimilarity of their unfoldings to event structures, it is possible to produce a characterisation in terms of nets and their 'processes' along the lines of [**89**].

Many attempts have been made to define bisimilarity for noninterleaving models like Petri nets, and the idea of parameterising the definition on a notion of observation has been used in other attempts, e.g., [**16**]. One of the advantages of Pom_L-bisimilarity is, as shown, e.g., by Theorem 4.9, its *robustness* across a range of models. Another issue is the the *sensitivity* of Pom_L-bisimilarity for nets to the particular choice of path category Pom_L, as the notion of Pom_L-bisimilarity might in fact seem questionable to those who view general pomsets as not observable. To answer such a question, let us define a pomset to be an *almost totally ordered multiset* if it is of one of the two simple forms considered in the proof of Proposition 4.7, i.e., allowing at *most* two (maximal) elements to be unordered. Note that in the range of subclasses of *pomsets* considered in the literature (see [**69**]), this one is as close to Bran_L as one can get! The following result shows that restricting ourselves to such pomsets does not change the notion of P-bisimilarity.

COROLLARY 4.10. *Let Atom_L denote the full subcategory of Pom_L consisting of the almost totally ordered multisets.*

i) *A morphism in SWNets_L is Pom_L-open if and only if it is Atom_L-open.*
ii) *Two nets are Pom_L-bisimilar iff they are Atom_L-bisimilar.*

PROOF. Clearly (ii) follows from (i), so we concentrate on a proof of (i).

The *'only if'* part of (i) follows immediately from definition of open maps. By inspecting the proof of Proposition 4.7, we observe that a morphism in SWNets_L is Pom_L-open if it is Atom_L-open. □

Here, we have illustrated how to introduce bisimilarity for SW nets, using open maps as in [**35**]. Much of the theory developed since then on this approach to concurrency may be transferred to the setting of nets. As examples, we mention the logical and game theoretic characterisations from [**56**], and the treatment of higher order models in [**96**]. But many questions are still left open.

Part 2. ON THE STRUCTURE OF NETS

5. Petri nets as monoids

A very prominent role in the semantic theory of Petri nets is played by various notions of process, as, e.g. [**68, 26, 7, 45, 18**]. This, as we already pointed out in Section 1.2, is because processes are structures capable of accounting, not only for the mere occurrence of events in a computation, but also for the *causal relationships* which ruled such occurrences. In other terms, processes are *noninterleaving* structures and, as such, very suited to describe Petri nets.

A parallel and extremely successful line of research in concurrency, rooted in the very ideas of denotational semantics, is the one following the *algebraic* approach. Here the focus is on *structural* and *compositional* aspects of systems and behaviours, and the leading idea is to describing them by means of a few basic building blocks and a small number of *combinators* [**30, 49, 29, 50**]. The appeal of this approach is that it tends to devise neat algebraic structures that capture the *essential* nature of the class of systems considered.

In this section, we shall focus on a line of research — detailed, e.g., in [**45, 18, 47, 77, 78, 75, 76**] — aimed at recasting Petri *net processes* in lieu of ideas from *process algebras* and *categorical algebra*. In particular, we shall focus on Petri net *concatenable processes*, introduced in [**18**] to account, as their name indicates, for the issue of process concatenation. We start by briefly reconsidering the ideas that lead to their definition. The exposition will follow [**77**] closely.

5.1. Concatenable processes. Ideally, Petri net processes are simply computations in which explicit information about cause/effect relationship between event occurrences is added. More precisely, as we describe causality by means of partial orderings, the processes of a net N are ordered sets whose elements are labelled by transitions of N. In order to describe exactly which multisets of transitions form 'legal' processes, it is very convenient to define a process of N to be a map $\pi\colon \Theta \to N$ which maps transitions to transitions and places to places respecting the 'bipartite graph structure' of nets. Here Θ is a *finite deterministic occurrence net*, i.e., roughly speaking, a finite, conflict-free, 1-safe, acyclic net. The role of π is to 'label' the places and the (partially ordered) transitions of Θ with places and transitions of N in a way compatible with the structure of N.

Given this definition, one can assign the correct *source* and *target* states to a process $\pi\colon \Theta \to N$ by considering the multisets of places of N which are the image via π of the

places of Θ with, respectively, empty pre-set and empty post-set (henceforth referred to as *minimal* and *maximal* places of Θ). Now, the simple minded attempt to concatenate a process $\pi_1 \colon \Theta_1 \to N$ with source u to a process $\pi_0 \colon \Theta_0 \to N$ with target u by *gluing* the maximal places of Θ_0 with the minimal places of Θ_1 in a way which preserves the labellings fails immediately. In fact, if more than one place of u is labelled by a single place of N, there are many ways to put in one-to-one correspondence the maximal places of Θ_0 and the minimal places of Θ_1 respecting the labels, i.e., there are many possible concatenations of π_0 and π_1, each of which gives a possibly different process of N. In other words, as the above argument shows, process concatenation has to do with *gluing tokens*, i.e., instances of places, rather than *gluing places*.

Therefore, to deal with process concatenation one must disambiguate the *identity* of each token in a process. This is exactly the idea of *concatenable processes*, which are simply Goltz-Reisig [**26**] processes in which the minimal and maximal places carrying the same label are linearly ordered. This yields immediately an operation of concatenation, since the ambiguity about the identity of tokens is resolved using the additional information given by the orderings. Moreover, the existence of concatenation leads easily to the definition of the category of concatenable processes of N.

It turns out that such a category is a *symmetric monoidal category* whose tensor product is the parallel composition of processes [**18**]. We shall now recall this result, whose relevance is that it describes net behaviours as *algebras* in a remarkably abstract and smooth way, showing how to describe the concatenable processes of N as a symmetric monoidal category $P(N)$ defined axiomatically by means of *universal* constructions. Namely, $P(N)$ is the *free symmetric strict monoidal* category on the net N modulo two simple additional axioms [**77**].

5.2. Monoidal categories and concatenable processes. The notion of *monoidal category* dates back to [**3**] (see [**42**] for an easy thorough introduction and [**21**] for advanced topics). In this paper we shall be concerned only with a particular kind of symmetric monoidal categories, namely those which are *strict monoidal* and whose objects form a *free commutative monoid*.

A *symmetric strict monoidal category* (ssmc for short) is a structure $(\mathsf{C}, \otimes, e, \gamma)$, where C is a category, e is an object of C, called the *unit* object, $\otimes \colon \mathsf{C} \times \mathsf{C} \to \mathsf{C}$ is a functor, called the *tensor product*, subject to the following equations

$$\otimes \circ \langle \otimes \times 1_{\mathsf{C}} \rangle = \otimes \circ \langle 1_{\mathsf{C}} \times \otimes \rangle, \tag{1}$$

$$\otimes \circ \langle \underline{e}, 1_{\mathsf{C}} \rangle = 1_{\mathsf{C}}, \tag{2}$$

$$\otimes \circ \langle 1_{\mathsf{C}}, \underline{e} \rangle = 1_{\mathsf{C}}, \tag{3}$$

where $\underline{e} \colon \mathsf{C} \to \mathsf{C}$ is the constant functor which associate e and id_e respectively to each object and each morphism of C, $\langle \_, \_ \rangle$ is the pairing of functors induced by the cartesian product, and $\gamma \colon x_1 \otimes x_2 \xrightarrow{\sim} x_2 \otimes x_1$ is a natural isomorphism, called the *symmetry* of C, subject to the following Kelly-MacLane *coherence* axioms [**41, 39**]:

$$(\gamma_{x,z} \otimes id_y) \circ (id_x \otimes \gamma_{y,z}) = \gamma_{x \otimes y, z}, \tag{4}$$

$$\gamma_{y,x} \circ \gamma_{x,y} = id_{x \otimes y}. \tag{5}$$

Clearly, equation (1) states that the tensor is associative on both objects and arrows, while (2) and (3) state that e and id_e are, respectively, the unit object and the unit arrow

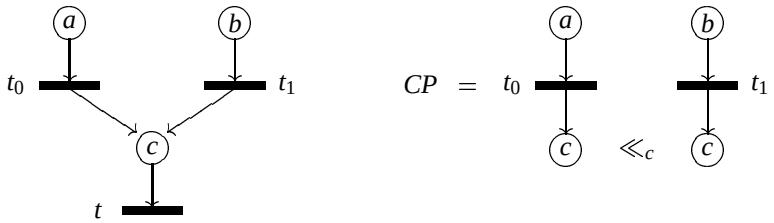


FIGURE 9. A net and one of its two concatenable processes $CP\colon a+b \to 2c$

for $\otimes$. Concerning the coherence axioms, axiom (5) says that $\gamma_{y,x}$ is the inverse of $\gamma_{x,y}$, while (4), the *real key* of symmetric monoidal categories, links the symmetry at composed objects to the symmetry at the components.

As an example easy enough to ponder the above equations, one might think of C as the category of sets, with $\otimes$ being the cartesian product, e the singleton set $\{\bullet\}$, and $\gamma_{X,Y}\colon X\times Y \to Y\times X$ as the map $(x,y)\mapsto(y,x)$.

A *symmetry* s in a symmetric monoidal category C is any arrow obtained as composition and tensor of *identities* and *components* of γ. We use Sym_{C} to denote the subcategory of the symmetries of C.

A *symmetric strict monoidal functor* from $(\mathsf{C},\otimes,e,\gamma)$ to $(\mathsf{D},\otimes',e',\gamma')$, is a functor $\mathsf{F}\colon \mathsf{C}\to\mathsf{D}$ which preserves the monoidal structure, i.e., such that

$$\mathsf{F}(e) = e', \tag{6}$$

$$\mathsf{F}(x\otimes y) = \mathsf{F}(x)\otimes'\mathsf{F}(y), \tag{7}$$

$$\mathsf{F}(\gamma_{x,y}) = \gamma'_{\mathsf{F}x,\mathsf{F}y}. \tag{8}$$

Let SSMC be the category of ssmc's and symmetric strict monoidal functors and let $\mathsf{SSMC}^{\oplus}$ be the full subcategory consisting of the monoidal categories whose objects form *free* commutative monoids.

In this section, we consider only PT nets with *finite markings*, but release all remaining restrictions. Let $S^{\oplus}$ denote the submonoid of $\mu(S)$ consisting of the *finite* multisets of S, i.e., the functions $S\to\omega$ which yield nonzero values at most on finitely many arguments. Then, define Petri to be the subcategory of PTNets consisting of those N whose transitions have source and target in $S_N^{\oplus}$, and of those $f\colon N_0\to N_1$ whose place components map $S_{N_0}^{\oplus}$ to $S_{N_1}^{\oplus}$.

Recall that $S^{\oplus}$ can be characterised as the *free commutative monoid* on S. In particular this means that the place component f_p of a morphism $f\colon N_0\to N_1$ in Petri is determined by assigning a multiset $f_p(a)\in S_{N_1}^{\oplus}$ for each place $a\in S_{N_0}$, and then freely extending it to the entire $S_{N_0}^{\oplus}$.

DEFINITION. A *process net* is a finite, acyclic net Θ such that for all $t\in T_\Theta$, $pre_\Theta(t)$ and $post_\Theta(t)$ are sets (as opposed to multisets), and for all $t_0\neq t_1\in T_\Theta$,

$$pre_\Theta(t_0)\cap pre_\Theta(t_1)=\varnothing \qquad \text{and} \qquad post_\Theta(t_0)\cap post_\Theta(t_1)=\varnothing.$$

Given $N\in\mathsf{Petri}$, a *process* of N is a morphism $\pi\colon\Theta\to N$, where Θ is a process net and π is a net morphism which maps places to places (as opposed to morphisms which map places to markings).

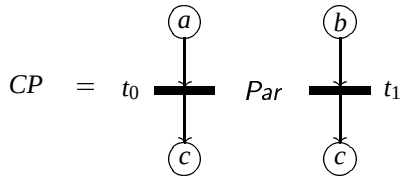


FIGURE 10. *CP* of Figure 9 as the parallel composition of two simpler processes

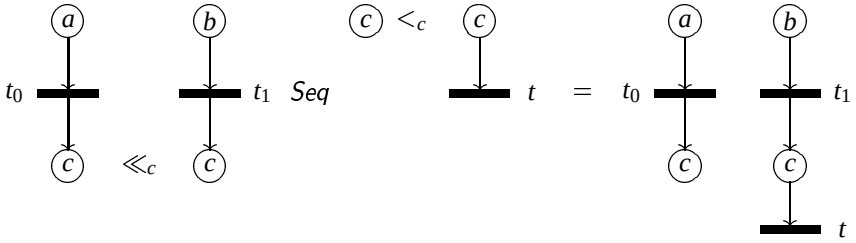


FIGURE 11. Sequential composition (concatenation) of concatenable processes

DEFINITION. A *concatenable process* of N is a triple

$$(\pi\colon \Theta \to N, \{<_a\}_{a\in S_N}, \{\ll_a\}_{a\in S_N}),$$

where π is a process, and $<_a$ and $\ll_a$ are linear orderings of, respectively, the set of minimal and the set of maximal places of Θ contained in $\pi_p^{-1}(a)$ (cf. Figure 9).

In order to abstract from the details concerning the underlying process nets, concatenable processes are considered up to isomorphisms. Formally, two concatenable processes, say with underlying processes $\pi_0\colon \Theta_0 \to N$ and $\pi_1\colon \Theta_1 \to N$, are identified if there exists an isomorphism $\varphi\colon \Theta_0 \to \Theta_1$ which preserves all the orderings and such that $\pi_1 \circ \varphi = \pi_0$.

Concatenable processes allow the operations of *sequential* and *parallel* composition (see Figures 10 and 11, and consult [**18**] for further examples). Let CP_0 and CP_1 be concatenable processes of N, and let $\pi_0\colon \Theta_0 \to N$ and $\pi_1\colon \Theta_1 \to N$ denote their underlying processes.

DEFINITION. The *parallel composition* CP_0 *Par* CP_1 is the concatenable process of N whose underlying process is the disjoint union of π_0 and π_1, i.e., $\pi_0+\pi_1\colon \Theta_0+\Theta_1 \to N$, where $+$ denotes the coproduct in Petri, and whose orderings extend those of CP_0 and CP_1 by making all the places of Θ_0 precede all the places of Θ_1.

DEFINITION. The *sequential composition*, or concatenation, $CP = CP_0$ *Seq* CP_1 is defined if and only if the state reached by CP_0 coincide with the source state of CP_1. In this case, CP is obtained by gluing together π_0 and π_1, identifying injectively each maximal place of Θ_0 with a minimal place of Θ_1 in the *unique* way compatible with the orderings $\ll_a$ on Θ_0 and $<_a$ on Θ_1 for all $a \in S_N$.

Next, we recall the construction of the symmetric strict monoidal category $P(N)$. We start by introducing the *vectors of permutations* (*vperms*) of N, which will provide the symmetry isomorphism of $P(N)$.

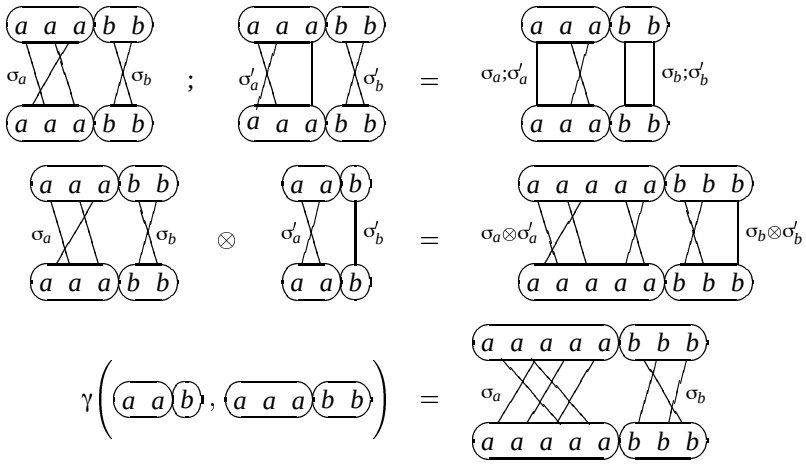


FIGURE 12. The monoidal structure of vperms

For $u \in S^{\oplus}$, a *vperm* $s\colon u \to u$ is a function which assigns to each $a \in S$ a permutation $s(a) \in \Pi(u(a))$. Given $u = n_1 \cdot a_1 + \ldots + n_k \cdot a_k$ in $S_N^{\oplus}$, we shall represent a vperm s on u as a vector of permutations, $\langle \sigma_{a_1}, \ldots, \sigma_{a_k} \rangle$, where $s(a_j) = \sigma_{a_j}$, whence their name. One can define the operations of sequential and parallel composition of vperms, so that they can be organised as the arrows of a ssmc. The details follow (see also Figure 12).

Given the vperms $s = \langle \sigma_{a_1}, \ldots, \sigma_{a_k} \rangle\colon u \to u$ and $s' = \langle \sigma'_{a_1}, \ldots, \sigma'_{a_k} \rangle\colon u \to u$ their *sequential composition* $s;s'\colon u \to u$ is the vperm

$$\langle \sigma_{a_1};\sigma'_{a_1}, \ldots, \sigma_{a_k};\sigma'_{a_k} \rangle,$$

where $\sigma;\sigma'$ is the composition of permutation which we write in the diagrammatic order from left to right.

Given the vperms $s = \langle \sigma_{a_1}, \ldots, \sigma_{a_k} \rangle\colon u \to u$ and $s' = \langle \sigma'_{a_1}, \ldots, \sigma'_{a_k} \rangle\colon v \to v$ (where possibly $\sigma_{a_j} = \varnothing$ for some j), their *parallel composition* $s \otimes s'\colon u + v \to u + v$ is the vperm

$$\langle \sigma_{a_1} \otimes \sigma'_{a_1}, \ldots, \sigma_{a_k} \otimes \sigma'_{a_k} \rangle,$$

where

$$(\sigma \otimes \sigma')(x) = \begin{cases} \sigma(x), & \text{if } 0 < x \leq |\sigma| \\ \sigma'(x - |\sigma|) + |\sigma|, & \text{if } |\sigma| < x \leq |\sigma| + |\sigma'| \end{cases}$$

Let γ be $(1\ 2) \in \Pi(2)$ and consider $u_i = n_1^i \cdot a_1 + \ldots + n_k^i \cdot a_k$, $i = 1,2$, in $S^{\oplus}$. The *interchange vperm* $\gamma(u_1, u_2)$ is the vperm $\langle \sigma_{a_1}, \ldots, \sigma_{a_k} \rangle\colon u_1 + u_2 \to u_1 + u_2$ where

$$\sigma_{a_j}(x) = \begin{cases} x + n_j^2, & \text{if } 0 < x \leq n_j^1 \\ x - n_j^1, & \text{if } n_j^1 < x \leq n_j^1 + n_j^2 \end{cases}$$

It is immediate to verify that $\_;\_$ is associative. Moreover, for each $u \in S^{\oplus}$, the vperm $u = \langle id_{a_1}, \ldots, id_{a_n} \rangle\colon u \to u$, where id_{a_j} is the identity permutation, is an identity for sequential composition. Finally, writing 0 for the empty multiset on S, the (unique) vperm $s\colon 0 \to 0$, is a unit for parallel composition.

Now, for N a net, let Sym_N be the category whose objects are the elements of $S_N^{\oplus}$ and whose arrows are the vperms $s\colon u \to u$ for $u \in S_N^{\oplus}$. It is easy to show that Sym_N is a ssmc with respect to the given composition and tensor product, with identities and unit element as explained above, and with the symmetry natural isomorphism given by the collection $\gamma = \{\gamma(u,v)\}_{u,v \in Sym_N}$ of the interchange vperms. Observe that, although Sym_N is not strictly symmetric, it is so on the objects. More strongly, the objects form a free commutative monoid, i.e., $Sym_N \in \mathsf{SSMC}^{\oplus}$.

We can now define $P(N)$ as the category which includes Sym_N as a subcategory and has as additional arrows those defined by the following rules:

$$\frac{t\colon u \to v \text{ in } T_N}{t\colon u \to v \text{ in } P(N)}$$

$$\frac{\alpha\colon u \to v \text{ and } \beta\colon u' \to v' \text{ in } P(N)}{\alpha \otimes \beta\colon u+u' \to v+v' \text{ in } P(N)} \qquad \frac{\alpha\colon u \to v \text{ and } \beta\colon v \to w \text{ in } P(N)}{\alpha;\beta\colon u \to w \text{ in } P(N)}$$

plus axioms expressing the fact that $P(N)$ is a ssmc with composition $\_;\_$, tensor $\_ \otimes \_$ (extending those already defined on vperms) and symmetry isomorphism γ, and the following axioms involving transitions and vperms

$$\begin{array}{ll} t;s = t & \text{where } t\colon u \to v \text{ in } T_N \text{ and } s\colon v \to v \text{ in } Sym_N, \\ s;t = t & \text{where } t\colon u \to v \text{ in } T_N \text{ and } s\colon u \to u \text{ in } Sym_N. \end{array} \qquad (\Psi)$$

In other words, $P(N)$ is built on the category Sym_N by adding the transitions of N and freely closing with respect to sequential and parallel composition of arrows, so that $P(N)$ is made symmetric strict monoidal and axioms (Ψ) hold.

The relevant fact about $P(N)$ is that its arrows represent exactly the concatenable processes of N, i.e., $P(N)$ represents the noninterleaving behaviour of N, including its algebraic structure. (See [**18**] for proof and details.)

THEOREM 5.1. *For any net N there exists a one-to-one correspondence between the arrows of $P(N)$ and the concatenable processes of N such that, for each $u, v \in S_N^{\oplus}$, the arrows of type $u \to v$ correspond to the processes enabled by u and producing v, and such that sequential and parallel composition (tensor product) of processes (arrows) are respected.*

Vperms play in this correspondence an absolutely fundamental role: Sym_N accounts for the families of orderings $\{<_a\}_{a \in S_N}$ and $\{\ll_a\}_{a \in S_N}$, which are the key to concatenable processes, guaranteeing a correct treatment of sequential composition. In other words, Sym_N is an algebraic representation of the 'threads of causality' in process concatenation.

5.3. Axiomatising concatenable processes. Unfortunately, the concrete definition of vperms weakens considerably the essentially axiomatic character of $P(N)$, as the laws which rule it remain partly concealed in Sym_N. The aim of this section is to provide a fully axiomatic description of the concatenable processes of N obtained by proving that $P(N)$ is a quotient of the free ssmc on N [**77**]. The key to this result will be an axiomatisation of the category of vperms Sym_N. We start by showing that we can associate a free ssmc to each net N.

PROPOSITION 5.2. *The forgetful functor* $\mathcal{U}\colon \mathsf{SSMC}^{\oplus} \to \mathsf{Petri}$ *admits a left adjoint* $\mathcal{F}\colon \mathsf{Petri} \to \mathsf{SSMC}^{\oplus}$.

PROOF. Consider the category $\mathcal{F}(N)$ whose objects are the elements of $S_N^{\oplus}$ and whose arrows are generated by the inference rules

$$\frac{u \in S_N^{\oplus}}{id_u\colon u \to u \text{ in } \mathcal{F}(N)} \qquad \frac{a \text{ and } b \text{ in } S_N}{c_{a,b}\colon a+b \to b+a \text{ in } \mathcal{F}(N)} \qquad \frac{t\colon u \to v \text{ in } T_N}{t\colon u \to v \text{ in } \mathcal{F}(N)}$$

$$\frac{\alpha\colon u \to v \text{ and } \beta\colon u' \to v' \text{ in } \mathcal{F}(N)}{\alpha \otimes \beta\colon u+u' \to v+v' \text{ in } \mathcal{F}(N)} \qquad \frac{\alpha\colon u \to v \text{ and } \beta\colon v \to w \text{ in } \mathcal{F}(N)}{\alpha;\beta\colon u \to w \text{ in } \mathcal{F}(N)}$$

modulo the axioms expressing that $\mathcal{F}(N)$ is a strict monoidal category, namely,

$$\begin{array}{rl} \alpha; id_v = \alpha = id_u; \alpha, & \text{and} \quad (\alpha;\beta);\gamma = \alpha;(\beta;\gamma), \\ (\alpha \otimes \beta) \otimes \gamma = \alpha \otimes (\beta \otimes \gamma), & \text{and} \quad id_0 \otimes \alpha = \alpha = \alpha \otimes id_0, \\ id_u \otimes id_v = id_{u+v}, & \text{and} \quad (\alpha \otimes \alpha');(\beta \otimes \beta') = (\alpha;\beta) \otimes (\alpha';\beta'), \end{array} \tag{9}$$

the latter whenever the right-hand term is defined, and the following axioms

$$c_{a,b};c_{b,a} = id_{a+b}, \tag{10}$$

$$c_{u,u'};(\beta \otimes \alpha) = (\alpha \otimes \beta);c_{v,v'}, \quad \text{for } \alpha\colon u \to v,\ \beta\colon u' \to v', \tag{11}$$

where $c_{u,v}$ for $u, v \in S_N^{\oplus}$ denote *any* term obtained from $c_{a,b}$ for $a, b \in S_N$ by applying recursively the following rules (compare with axiom (4)):

$$\begin{array}{rcl} c_{0,u} & = & c_{0,u} \quad = \quad id_u, \\ c_{a+u,v} & = & (id_a \otimes c_{u,v});(c_{a,v} \otimes id_u), \\ c_{u,v+a} & = & (c_{u,v} \otimes id_a);(id_v \otimes c_{u,a}). \end{array} \tag{12}$$

Observe that equation (11), in particular, equalises all the terms obtained from (12) for fixed u and v. In fact, let $c_{u,v}$ and $c'_{u,v}$ be two such terms and take α and β to be, respectively, the identities of u and v. Now, since $id_u \otimes id_v = id_{u \oplus v} = id_v \otimes id_u$, from (11) we have that $c_{u,v} = c'_{u,v}$ in $\mathcal{F}(N)$. Then, we claim that the collection $\{c_{u,v}\}_{u,v \in S_N^{\oplus}}$ is a symmetry natural isomorphism which makes $\mathcal{F}(N)$ into a ssmc and that, in addition, $\mathcal{F}(N)$ is the free ssmc on N.

In order to show the first claim, observe that the naturality of c is expressed directly from axiom (11). We need to check that for any u and v we have $c_{u,v};c_{v,u} = id_{u \oplus v}$, which follows from (10) by induction on the sum of the sizes of u and v.

As for the second, for C in $\mathsf{SSMC}^{\oplus}$, the net $\mathcal{U}(\mathsf{C})$ is obtained by forgetting the categorical structure of C. The markings and the transitions of $\mathcal{U}(\mathsf{C})$ are, respectively, the objects and the arrows of C with the given sources and targets. Similarly, for F a symmetric strict monoidal functor in $\mathsf{SSMC}^{\oplus}$, $\mathcal{U}(\mathsf{F})$ is the net morphism whose components are the restrictions of F to, respectively, arrows and objects. Consider the net $\mathcal{U}\mathcal{F}(N)$ and the net morphism $\eta\colon N \to \mathcal{U}\mathcal{F}(N)$, where η_p is the identity homomorphism and η_t is the obvious injection of T_N in $T_{\mathcal{U}\mathcal{F}(N)}$. One can then show (cf. [**77**] for the details) that η is universal, i.e., that for any C in $\mathsf{SSMC}^{\oplus}$ and for any net morphism $f\colon N \to \mathcal{U}(\mathsf{C})$, there is a unique symmetric strict monoidal functor $\mathsf{F}\colon \mathcal{F}(N) \to \mathsf{C}$ which

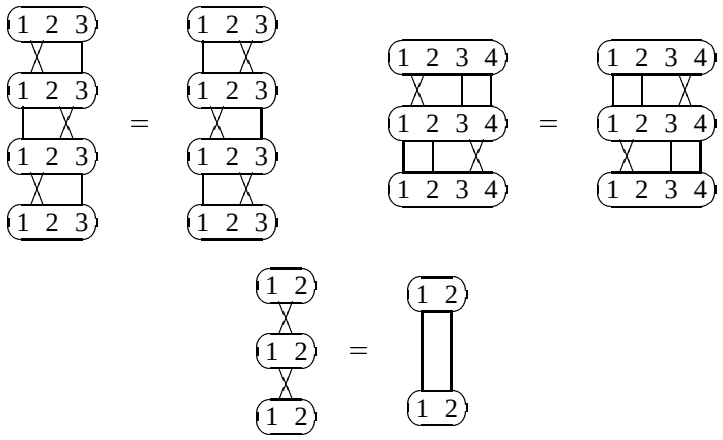
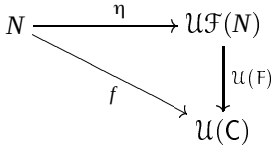


FIGURE 13. Some instances of the axioms of permutations

makes the following diagram commute.



□

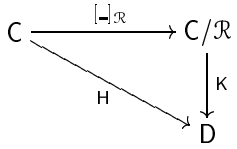
Thus, establishing the adjunction $\mathcal{F} \dashv \mathcal{U} \colon \mathsf{Petri} \rightharpoonup \mathsf{SSMC}^{\oplus}$, we have identified $\mathcal{F}(N)$, the *free ssmc on N*, as a category generated, modulo appropriate equations, from the net N viewed as a graph enriched with formal arrows id_u, which play the role of the identities, and $c_{a,b}$ for $a, b \in S_N$, which generate all the needed symmetries.

Our aim is to relate $\mathcal{F}(N)$ and $P(N)$. As a matter of fact, $\mathcal{F}(N)$ is positively more concrete than $P(N)$ and far from being isomorphic (or equivalent) to it. For example, for $a \neq b$ in S_N, we have $c_{a,b} \neq id_{a \oplus b}$ in $\mathcal{F}(N)$, whilst $\gamma(a,b) = id_{a \oplus b}$ in $P(N)$. Therefore, no symmetric monoidal functor $\mathsf{Q} \colon \mathcal{F}(N) \to P(N)$ can be mono. Also, $\mathcal{F}(N)$ possesses no counterpart of axioms (Ψ). We shall see that these are precisely the differences between $\mathcal{F}(N)$ and $P(N)$. Namely, we shall obtain $P(N)$ as a quotient of $\mathcal{F}(N)$ by enforcing the axioms outlined above. The next proposition, which is the adaptation to ssmc's of the usual notion of quotient algebras, provides the tool we shall use for this purpose.

PROPOSITION 5.3. *For* C *a ssmc, let* $\mathcal{R}$ *be a function which assigns to each pair of objects a and b of* C *a binary relation* $\mathcal{R}_{a,b}$ *on the homset* $\mathsf{C}(a,b)$*. Then, there exist a ssmc* $\mathsf{C}/\mathcal{R}$ *and a symmetric strict monoidal functor* $[\_]_{\mathcal{R}} \colon \mathsf{C} \to \mathsf{C}/\mathcal{R}$ *such that*

i) If $f \; \mathcal{R}_{a,b} \; f'$ *then* $[f]_{\mathcal{R}} = [f']_{\mathcal{R}}$*;*

ii) *For each symmetric strict monoidal* $\mathsf{H}\colon \mathsf{C} \to \mathsf{D}$ *such that* $\mathsf{H}(f) = \mathsf{H}(f')$ *whenever* $f \mathcal{R}_{a,b} f'$*, there exists a unique* $\mathsf{K}\colon \mathsf{C}/\mathcal{R} \to \mathsf{D}$*, which is necessarily symmetric strict monoidal, such that the following diagram commutes.*



PROOF. Take $\mathsf{C}/\mathcal{R}$ to be the category whose objects are those of C, whose homset $\mathsf{C}/\mathcal{R}(a,b)$ is $\mathsf{C}(a,b)/\mathcal{R}_{a,b}$, with composition of arrows given by $[g]_{\mathcal{R}} \circ [f]_{\mathcal{R}} = [g \circ f]_{\mathcal{R}}$, and define $[f]_{\mathcal{R}} \otimes [g]_{\mathcal{R}} = [f \otimes g]_{\mathcal{R}}$.

Say that $\mathcal{R}$ is a *congruence* if $\mathcal{R}_{a,b}$ is an equivalence for each a and b and if $\mathcal{R}$ respects composition, i.e., whenever $f \mathcal{R}_{a,b} f'$ then, for all $h\colon a' \to a$ and $k\colon b \to b'$, we have $(k \circ f \circ h) \mathcal{R}_{a',b'} (k \circ f' \circ h)$. Call $\mathcal{R}$ a $\otimes$*-congruence* if it is a congruence and it similarly respects tensor. It is easy to check that, if $\mathcal{R}$ is a $\otimes$-congruence, then the above definition makes the quotient category $\mathsf{C}/\mathcal{R}$ into a ssmc with symmetry isomorphisms $[\gamma_{u,v}]_{\mathcal{R}}$ and unit object e.

Observe now that, given $\mathcal{R}$ as in the hypothesis, it is always possible to find the least $\otimes$-congruence $\mathcal{R}'$ which includes (componentwise) $\mathcal{R}$. Then, take $\mathsf{C}/\mathcal{R}$ to be $\mathsf{C}/\mathcal{R}'$ and $[\_]_{\mathcal{R}}$ to be the obvious projection of C into $\mathsf{C}/\mathcal{R}$, which is clearly a symmetric strict monoidal functor. □

In order to show that $P(N)$ is a monoidal quotient of $\mathcal{T}(N)$, we need a more abstract understanding of the structure of the vperms of $P(N)$. To this aim, we shall make use of the following lemma, originally proved in [**52**].

LEMMA 5.4. *The symmetric group* $\Pi(n)$ *is (isomorphic to) the group G freely generated from the set* $\{\tau_i \mid 1 \leq i < n\}$*, modulo the equations (see also Figure 13)*

$$\begin{aligned} \tau_i \tau_{i+1} \tau_i &= \tau_{i+1} \tau_i \tau_{i+1}, \\ \tau_i \tau_j &= \tau_j \tau_i \quad \text{if } |i-j| \geq 1, \\ \tau_i \tau_i &= e, \end{aligned} \tag{13}$$

where e is the unit element of G.

The previous lemma is easily adapted to vperms by translating axioms (13) as follows.

LEMMA 5.5. *The arrows of* Sym_N *are freely generated by composition and tensor from the vperms* $\gamma(a,a)\colon 2 \cdot a \to 2 \cdot a$*, for* $a \in S_N$*, modulo the axioms (9) of strict monoidal categories and the following additional axioms*

$$\begin{aligned} \big((id_a \otimes \gamma(a,a)) ; (\gamma(a,a) \otimes id_a)\big)^3 &= id_{3 \cdot a}, \\ \gamma(a,a)^2 &= id_{2 \cdot a}, \\ (id_b \otimes \gamma(a,a)) ; (\gamma(a,a) \otimes id_b) &= id_{2 \cdot a + b}, \quad \text{if } a \neq b \in S_N, \end{aligned} \tag{14}$$

where f^n *indicates the composition of* f *with itself n times.*

PROOF. See [**77**]. □

We are now ready to give the promised characterisation of $P(N)$.

PROPOSITION 5.6. *$P(N)$ is the monoidal quotient of the free ssmc on N modulo the axioms*

$$c_{a,b} = id_{a+b}, \quad \textit{if } a,b \in S_N \textit{ and } a \neq b, \tag{15}$$

$$s;t;s' = t, \quad \textit{if } t \in T_N \textit{ and } s,s' \textit{ are symmetries.} \tag{16}$$

PROOF. We prove that $P(N)$ is isomorphic to $\mathcal{F}(N)/\mathcal{R}$, where $\mathcal{R}$ is the congruence for $\otimes$ and $\_;\_$ generated from equations (15) and (16).

Since $P(N)$ belongs to $\mathsf{SSMC}^{\oplus}$, it follows from Proposition 5.2 that, corresponding to the net *inclusion* morphism $N \to \mathcal{U}P(N)$, there is a *unique* symmetric strict monoidal functor $\mathsf{Q}\colon \mathcal{F}(N) \to P(N)$ which is the identity on the places and on the transitions of N. In particular, Q is such that

$$\mathsf{Q}(c_{a,b}) = \gamma(a,b), \quad \text{for } a,b \in S_N.$$

For $a \neq b \in S_N$, since $\gamma(a,b) = id_{a \oplus b}$, we have that $\mathsf{Q}(c_{a,b}) = \mathsf{Q}(id_{a \oplus b})$. Moreover, since symmetric monoidal functors map symmetries to symmetries, and since (16) holds in $P(N)$, we have that $\mathsf{Q}(s;t;s') = \mathsf{Q}(s);t;\mathsf{Q}(s') = t = \mathsf{Q}(t)$ for s and s' in $Sym_{\mathcal{F}(N)}$ and $t \in T_N$. In other words, Q equalises the pairs $\langle c_{a,b}, id_{a \oplus b} \rangle$ with $a \neq b \in S_N$ and the pairs $\langle s;t;s',t \rangle$ with s and s' symmetries and $t \in T_N$. Then, in force of Proposition 5.3 applied to Q, there is a (unique) symmetric strict monoidal functor $\mathsf{H}\colon \mathcal{F}(N)/\mathcal{R} \to P(N)$ which is the identity on the objects and is such that

$$\mathsf{H}([t]_{\mathcal{R}}) = t, \quad \text{for } t \in T_N.$$

One can then prove that H is an isomorphism by producing its inverse $P(N) \to \mathcal{F}(N)/\mathcal{R}$ as the functor G which acts identically on the objects and is defined on the arrows by

$$\begin{aligned} \mathsf{G}(t) &= [t]_{\mathcal{R}}, && \text{if } t \in T_N, \\ \mathsf{G}(\gamma(a,a)) &= [c_{a,a}]_{\mathcal{R}}, && \text{if } a \in S_N, \end{aligned}$$

extended to identities, composition and tensor as usual: $\mathsf{G}(id_u) = [id_u]_{\mathcal{R}}$, $\mathsf{G}(\alpha;\beta) = \mathsf{G}(\alpha);\mathsf{G}(\beta)$, and $\mathsf{G}(\alpha \otimes \beta) = \mathsf{G}(\alpha) \otimes \mathsf{G}(\beta)$. Notice that it follows from the definition of $P(N)$ and from Lemma 5.5 that the equations above define G uniquely. □

The merit of this result is to describe the algebraic structure of $P(N)$, and thus of the concatenable processes of N, in terms of *universal* constructions, namely the construction on the free ssmc on Petri and a quotient construction on $\mathsf{SSMC}^{\oplus}$, providing in this way a completely abstract view of $P(N)$. It may be worth noticing in this context that (15) is actually a problematic axiom: because of its negative premise, viz., $a \neq b$, it invalidates the freeness of $\mathcal{F}(N)$ on Petri. Even worse, $\mathcal{F}(\_)/\mathcal{R}$ and $P(\_)$ fail to be functors from Petri to SSMC. On the other hand, axiom (15) plays very relevant a role in capturing algebraically the essence of concatenable process, and it cannot be dispensed with easily. A detailed study of this issue and a possible solution is provided in [**78, 76**]. In particular, in *loc. cit.*, a functorial and universal construction for net computations is devised, based on a refinement of the notion of concatenable processes called *strongly concatenable processes*.

Resuming our work, we give an alternative form of axiom (16).

COROLLARY 5.7. *Axiom (16) in Proposition 5.6 can be replaced by the axioms*

$$
\begin{array}{rcll}
t;(id_u \otimes c_{a,a} \otimes id_v) & = & t, & \text{if } t \in T_N \text{ and } a \in S_N, \\
(id_u \otimes c_{a,a} \otimes id_v);t & = & t, & \text{if } t \in T_N \text{ and } a \in S_N.
\end{array} \tag{17}
$$

PROOF. Since $(id_u \otimes \gamma_{a,a} \otimes id_v)$ and all the identities are symmetries, axiom (16) implies the present ones. It is easy to see that, on the other hand, the axioms above, together with axiom (15), imply (16).

Let $s\colon u \to u$ by a symmetry of $\mathcal{T}(N)$ and suppose $s \neq id_u$. By repeated applications of (12), together with the functoriality of $\otimes$, we obtain the following equality:

$$s = (id_{u_1} \otimes c_{a_1,b_1} \otimes id_{v_1}); \ldots ; (id_{u_h} \otimes c_{a_h,b_h} \otimes id_{v_h})$$

for some $h \in \omega$. Moreover, by exploiting axiom (15), we can drop every term in which $a_i \neq b_i$. Thus we have

$$s = (id_{u_1} \otimes c_{a_1,a_1} \otimes id_{v_1}); \ldots ; (id_{u_k} \otimes c_{a_k,a_k} \otimes id_{v_k})$$

for some $k \leq h$. Then, by this equation and by repeated applications of axioms (17), one can prove $s;t;s' = t$. □

Finally, the next corollary sums up the purely algebraic characterisation of the category of concatenable processes that we illustrated here. In particular, it identifies in algebraic terms the essential components of concatenable processes and the laws which rule their sequential and parallel composition.

COROLLARY 5.8. *The category $P(N)$ of concatenable processes of N is the category whose objects are the elements of $S_N^{\oplus}$ and whose arrows are generated by the inference rules*

$$\frac{u \in S_N^{\oplus}}{id_u\colon u \to u \text{ in } P(N)} \qquad \frac{a \text{ in } S_N}{c_{a,a}\colon a+a \to a+a \text{ in } P(N)} \qquad \frac{t\colon u \to v \text{ in } T_N}{t\colon u \to v \text{ in } P(N)}$$

$$\frac{\alpha\colon u \to v \text{ and } \beta\colon u' \to v' \text{ in } P(N)}{\alpha \otimes \beta\colon u+u' \to v+v' \text{ in } P(N)} \qquad \frac{\alpha\colon u \to v \text{ and } \beta\colon v \to w \text{ in } P(N)}{\alpha;\beta\colon u \to w \text{ in } P(N)}$$

modulo the axioms expressing that $P(N)$ is a strict monoidal category, namely,

$$
\begin{array}{rll}
\alpha;id_v = \alpha = id_u;\alpha, & \text{and} & (\alpha;\beta);\gamma = \alpha;(\beta;\gamma), \\
(\alpha \otimes \beta) \otimes \gamma = \alpha \otimes (\beta \otimes \gamma), & \text{and} & id_0 \otimes \alpha = \alpha = \alpha \otimes id_0, \\
id_u \otimes id_v = id_{u \oplus v}, & \text{and} & (\alpha \otimes \alpha');(\beta \otimes \beta') = (\alpha;\beta) \otimes (\alpha';\beta'),
\end{array}
$$

the latter whenever the right-hand term is defined, and the following axioms

$$
\begin{array}{rcll}
c_{a,a};c_{a,a} & = & id_{a+a}, & \\
t;(id_u \otimes c_{a,a} \otimes id_v) & = & t, & \text{if } t \in T_N, \\
(id_u \otimes c_{a,a} \otimes id_v);t & = & t, & \text{if } t \in T_N, \\
c_{u,u'};(\beta \otimes \alpha) & = & (\alpha \otimes \beta);c_{v,v'}, & \text{for } \alpha\colon u \to v,\ \beta\colon u' \to v',
\end{array}
$$

where $c_{u,v}$, for $u,v \in S_N^{\oplus}$, is obtained from $c_{a,a}$ by applying recursively the rules:

$$\begin{aligned} c_{a,b} &= id_{a+b}, \quad \textit{if } a=0 \textit{ or } b=0 \textit{ or } (a,b \in S_N \textit{ and } a \neq b), \\ c_{a+u,v} &= (id_a \otimes c_{u,v}); (c_{a,v} \otimes id_u), \\ c_{u,v+a} &= (c_{u,v} \otimes id_a); (id_v \otimes c_{u,a}). \end{aligned}$$

PROOF. Observe that the terms and the axioms above are obtained normalising those of $\mathcal{T}(N)$ with respect to $c_{a,b} = id_{a+b}$, for $a \neq b \in S_N$, and then adding axioms (15) and (17). The claim then follows immediately from Proposition 5.2, Proposition 5.6, and Corollary 5.7. □

6. Conclusions and Related Work

We have presented some examples of the use of category theory in understanding the behaviour and structure of Petri nets and their relationships to other models for concurrency. However, there are many further examples of applications of categorical ideas in concurrency.

The results on categorical relationships between models is a small part of a general picture, as illustrated in [**80**], in which a number of important constructions from concurrency theory emerges as parts of (coreflective) 'unfoldings' and 'sequentialisations', and (reflective) 'determinizations'. Also, several results in the literature [**74, 79, 62**] concern trace structures and other models for concurrency — including the pomsets of Pratt [**69**] and the partial words of Grabowsky [**28**].

We have illustrated how to introduce bisimilarity for Petri nets following a general pattern which automatically guarantees consistency with bisimilarity on a number of related models. From the work on open maps, it was suggested in [**35**] to study *presheaves* as models derived directly from path categories. Intuitively, a presheaf represents a system by 'gluing' together (an abstract set of) computation paths, and the advantage of this approach is that a number of categorical concepts provide a *uniform* notion of model of computation, as an alternative to the often ad hoc, concrete constructions adopted in the literature. Formally, given a path category P, the category $\widehat{\mathsf{P}}$ of presheaves over P consists of functors $\mathsf{P}^{op} \to \mathsf{Set}$ (where Set is the category of sets with functions) as objects, and natural transformations between them as morphisms. A presheaf $\mathsf{F}\colon \mathsf{P}^{op} \to \mathsf{Set}$ can be thought of as specifying for each path object P, the set $\mathsf{F}(P)$ of paths from P. It acts on a morphism $m\colon P \to Q$ in P to give a function $\mathsf{F}(m)\colon \mathsf{F}(Q) \to \mathsf{F}(P)$ saying how Q-paths restrict to P-paths.

Presheaves may thus be looked upon as labelled transition systems, where states are (abstract representations of) sets of possible runs of the paths in P, labels are path extensions, and the transitions describe how runs extend each other. Based on this view of presheaves, [**98**] provided logical and game-theoretic characterisations of open maps and their bisimulations on presheaves, which in turn may be specialised to concrete models like Petri nets via uniform representations as presheaves. Also, i n recent work by G. Winskel and others, these presheaf models have been used successfully in dealing with higher-order models in concurrency [**96, 15**].

We then illustrated how to use categorical tools 'in the small', focusing our study at level of single nets. Building on [**45, 18, 77**], we described the concatenable processes

of a Petri net N in terms of *universal* constructions, providing in such a way an abstract, fully axiomatic presentation of their algebraic structure. In particular, Corollary 5.8 provides a *term algebra* and an *equational theory* of the concatenable processes of N. Technically, relying on the characterisation of the concatenable processes of N as the arrows of the symmetric strict monoidal category $P(N)$, the result was illustrated by showing (cf. Proposition 5.6) that $P(N)$ is the quotient of the free symmetric strict monoidal category on N modulo two simple axioms. The proof of this fact makes an essential use of the axiomatisation of Sym_N, the category of *symmetries* of $P(N)$, provided by Lemma 5.5.

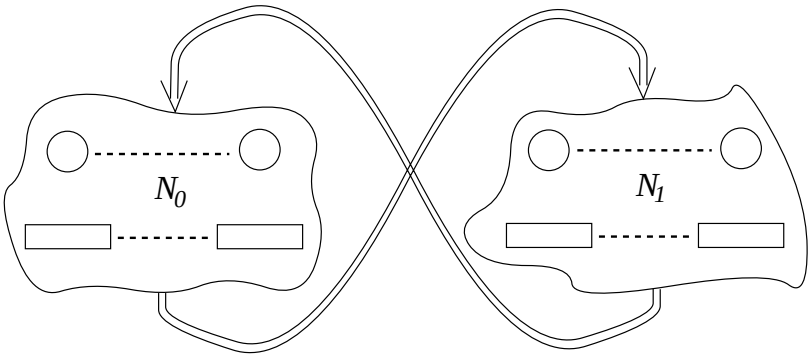
It is worth noticing that lifting these result to the totality of nets is rather problematic, as the negative premise of axiom (15) — essential from the computational viewpoint — breaks the freeness of $\mathcal{F}(N)$ on Petri and makes $P(\_)$ fail to be functor from Petri to SSMC. The interested reader is referred to [**78**] for a detailed study of the problem, and for a suggested solution based on a refined notion of so-called *strongly* concatenable processes.

An aspect of Petri nets we did not touch in this paper is their use as a semantic basis to interpret concurrent languages (see for example [**90, 64, 25, 17**]), an application that clearly calls for a '*process algebra-like*' description of nets and, possibly, for a suitable abstract characterisation of it. And in fact, the literature is rich of examples of process algebras over nets, as, e.g., [**54, 27, 8, 51, 58**] (see also the early [**19, 13, 99, 38, 70**] on compositionality issues). Observe that category theory can clearly play an interesting role in this, as we are called to consider the totality of nets, as in Part 1, focusing this time — as in Part 2 — on algebraic and compositional aspects. We conclude this paper explaining the basic ideas underlying the algebra of nets presented in [**58**], and how it may be related to the categorical approach we have taken here.

The approach is entirely based on a notion of *interface* for Petri nets. Informally, an interface for a net N is a selection of places and transitions of N which specifies what parts of N are *public*, i.e., accessible to the environment, and what parts are *private* to N. The private places and transitions cannot be accessed and, therefore, they cannot be used for connecting N with other nets. Net interfaces are built out of two components: an '*input*' interface, consisting of places, and an '*output*' interface, consisting of transitions. Intuitively, the input interface provides the buffers in which the tokens arriving from the environment are gathered, whilst the output interface sends tokens out to the environment.

Drawing on the experience of developments in concurrency theory, one aims at defining a minimal set of net combinators expressive enough to form a rudimentary calculus of nets. This should certainly include operations allowing (forms of) *communication* and *parallel composition* (and, to make easier the description of (large) modular systems, also operations as *relabelling* and *hiding*). However, in order to avoid a chaotic 'structural' calculus where everything is permitted, it is obvious that some restrictions on the allowed connections via places and transitions must be imposed. Interfaces readily suggest a reasonable discipline of interaction: connections between nets should go from outputs to inputs, involving only public components. This formalises the well-motivated and solid intuition that the only allowed interactions are achieved by *sending* and *receiving* along interfaces, to be thought of as communication channels.

The main way of combining nets provided in [**58**] therefore consists of connecting the outputs of one net to the inputs of another net and, possibly, vice versa, as schematically shown by the following picture.



Following the principle of considering as simple operations as possible, this is realised by two more basic combinators: $par(\_,\_)$, which puts its two arguments side by side, and $add(\_)$, which augments its argument by a new arc from an interface-transition to an interface-place. The operation above is then obtained by repeatedly applying $add(\_)$ to $par(N_0, N_1)$.

Observe that $add(\_)$ in isolation provides an interesting form of *recursion* consisting of *feeding back* outputs to inputs, and represents a bridge to structures of recent common interest in category theory and in computer science: the *traced monoidal categories* [**36, 40**], i.e., monoidal categories equipped with an *feedback* operation completely analogous to the one discussed above. Algebraic structures based on a central operation of *iteration*, or *feedback* — inspired by flowcharts and program schemata — have appeared rather early in computer science, see, e.g., [**22, 2, 85, 86, 51**] and [**10**], that offers for a thorough exposition of so-called '*iteration theory*' and more references. The advent of traced monoidal categories, though, has recently revived interest in using such abstract structures in semantics of computation, as e.g., in [**1, 37, 31, 32**]. Obviously, the calculus of [**58**] fits nets into this framework very nicely, although some of the details still need to be clarified.

Acknowledgements

We would like to thank many colleagues who have been part of developing the material behind this chapter. In particular, we acknowledge our close collaboration with A. Joyal, J. Meseguer, U. Montanari, L. Priese, G. Rozenberg, P.S. Thiagarajan, and G. Winskel on many parts of this work.

References

[1] S. ABRAMSKY (1996), Retracing Some Paths in Process Algebra, in *Proceedings of CONCUR 96*, U. Montanari and V. Sassone (Eds.), *Lecture Notes in Computer Science* **1119**, 1–17, Springer-Verlag.

[2] E.S. BAINBRIDGE (1976), Feedback and Generalized Logic. *Information and Control* **31**, 75–96.

[3] J. BÉNABOU (1963), Categories with Multiplication. *Comptes Rendus Académie Science Paris* **256**, 1887–1890.

[4] M.A. BEDNARCZYK (1987), *Categories of Asynchronous Systems*. PhD thesis, University of Sussex, Report **1/88**, School of Cognitive and Computing Sciences, University of Sussex.

[5] D. BENSON AND O. BEN-SHACHAR (1988), Bisimulation of Automata. *Information and Computation* **79**, 60–83, Academic Press.

[6] G. BERRY (1979), *Modèles completement adéquats et stables des λ-calculs typées*. Thèse de Doctorat d'Etat, Université Paris VII.

[7] E. BEST AND R. DEVILLERS (1987), Sequential and Concurrent Behaviour in Petri Net Theory. *Theoretical Computer Science* **55**, 87–136, Elsevier.

[8] E. BEST, R. DEVILLERS, AND J. HALL (1992), The Petri Box Calculus: a New Causal Algebra with Multilabel Communication, in *Advances in Petri Nets 92*, G. Rozenberg (Ed.), *Lecture Notes in Computer Science* **609**, 21–69, Springer-Verlag.

[9] E. BEST AND C. FERNANDEZ (1988), *Non-sequential Processes: A Petri Net View*. EATCS Monographs on Theoretical Computer Science **13**, Springer-Verlag.

[10] S.L. BLOOM AND Z. ÉSIK (1991), *Iteration Theories: the Equational Logic of Iterative Processes*. EATCS Monographs on Theoretical Computer Science **30**, Springer-Verlag.

[11] C. BROWN AND D. GURR (1992), Temporal Logic and Categories of Petri Nets, in *Proceedings of ICALP 93*, A. Lingas *et al.* (Eds.), *Lecture Notes in Computer Science* **700**, 570–581, Springer-Verlag.

[12] C. BROWN, D. GURR, AND V. DE PAIVA (1991), *A Linear Specification Language for Petri Nets*. Technical Report DAIMI PB-**363**, Computer Science Dept., University of Aarhus.

[13] J. BRUNO AND S.M. ALTMAN (1971), A Theory of Asynchronous Control Networks. *IEEE Transactions on Computers*, Vol. C-**20**, 629–638, IEEE Press.

[14] I. CASTELLANI (1985), Bisimulation and Abstraction Homomorphisms, in *Proceedings of CAAP 85*, H. Ehrig *et al.* (Eds.), *Lecture Notes in Computer Science* **185**, 223–238, Springer-Verlag.

[15] G.L. CATTANI, M. FIORE, AND G. WINSKEL (1998), A Theory of Recursive Domains with Applications to Concurrency, in *Proceedings of the 13th LICS Symposium*, IEEE Press. To appear.

[16] P. DEGANO, R. DE NICOLA, AND U. MONTANARI (1987). Observational Equivalences for Concurrency Models, in *Formal Description of Programming Concepts – III, IFIP*, M. Wirsing (Ed.), 105–132, Elsevier.

[17] P. DEGANO, R. DE NICOLA, AND U. MONTANARI (1988), A Distributed Operational Semantics for CCS based on Condition/Event Systems. *Acta Informatica* **26**, 59–91, Springer-Verlag.

[18] P. DEGANO, J. MESEGUER, AND U. MONTANARI (1989), Axiomatizing Net Computations and Processes, in *Proceedings of the 4th LICS Symposium*, 175–185, IEEE Press.

[19] J.B. DENNIS (1970), Modular, Asynchronous Control Structures for a High Performance Processor, in *Record of the Project MAC Conference on Concurrent Systems and Parallel Computation*, ACM Press.

[20] A. EHRENFEUCHT AND G. ROZENBERG (1987), On the Structure of Dependence Graphs, in *Concurrency and Nets*, K. Voss *et al.* (Eds.), 141–170, Springer-Verlag.

[21] S. EILENBERG, AND G.M. KELLY (1966), Closed Categories, in *Proceedings of the Conference on Categorical Algebra*, S. Eilenberg *et al.* (Eds.), 421–562, Springer-Verlag.

[22] C. ELGOT (1975), Monadic Computation and Iterative Algebraic Theories, in *Logic Colloquium '73*, H.E. Rose and J.C. Shepherdson (Eds.), 175–230, North-Holland.

[23] G. FERRARI, U. MONTANARI, AND M. MOWBRAY (1997), Structured Transition Systems with Parametric Observations: Observational Congruences and Minimal Realizations. *Mathematical Structures in Computer Science* **7**, 241–282, Cambridge University Press.

[24] R.J. VAN GLABBEEK AND U. GOLTZ (1989), Equivalence Notions for Concurrent Systems and Refinement of Actions, in *Proceedings of MFCS 89*, A. Kreczmar and G. Mirkowska (Eds.), *Lecture Notes in Computer Science* **379**, 237–248, Springer-Verlag.

[25] R.J. VAN GLABBEEK AND F. VAANDRAGER (1987), Petri Net Models for Algebraic Theories of Concurrency, in *Proceedings of PARLE*, J.W. de Bakker *et al.* (Eds.), *Lecture Notes in Computer Science* **259**, 224–242, Springer-Verlag.

[26] U. GOLTZ AND W. REISIG (1983), The Non-Sequential Behaviour of Petri Nets. *Information and Computation* **57**, 125–147, Academic Press.

[27] R. GORRIERI AND U. MONTANARI (1990), Scone: A Simple Calculus of Nets, in *Proceedings of CONCUR 90*, J.C.M. Baeten and J.W. Klop (Eds.), *Lecture Notes in Computer Science* **458**, 2–31, Springer-Verlag.

[28] J. GRABOWSKY (1981), On partial languages. *Annales Societatis Mathematicae Polonae* **IV.2**, 428–498, North-Holland.
[29] M. HENNESSY (1988), *Algebraic Theory of Processes*. MIT Press.
[30] C.A.R. HOARE (1985), *Communicating Sequential Processes*. Prentice Hall.
[31] M. HASEGAWA (1997), Recursion from Cyclic Sharing: Traced Monoidal Categories, in *Proceedings of TLCA 97*, Ph. de Groote and J.R. Hindley (Eds.), *Lecture Notes in Computer Science* **1210**, 196–213, Springer-Verlag.
[32] T.T. HILDEBRANDT, P. PANANGADEN, AND G. WINSKEL (1998), Relational Semantics of Non-Deterministic Dataflow, in *Proceedings of CONCUR 98*, D. Sangiorgi and R. de Simone (Eds.), *Lecture Notes in Computer Science*, Springer-Verlag. To appear.
[33] P.W. HOOGERS, H.C.M. KLEIJN, AND P.S. THIAGARAJAN (1992), A Trace Semantics for Petri Nets, in *Proceedings of ICALP 92*, W. Kuich (Ed.), *Lecture Notes in Computer Science* **623**, 595–604, Springer-Verlag.
[34] T. HUNE AND M. NIELSEN (1998), Timed Bisimulation and Open Maps, in *Proceedings of MFCS 98*, J. Gruska *et al.* (Eds.), *Lecture Notes in Computer Science*, Springer-Verlag. To appear.
[35] A. JOYAL, M. NIELSEN, AND G. WINSKEL (1996), Bisimulation from Open Maps. *Information and Computation* **127**, 164–185, Academic Press.
[36] A. JOYAL, R. STREET, AND D. VERITY (1996), Traced Monoidal Categories. *Mathematical Proceedings of the Cambridge Philosophical Society* **119**, 447–468, Cambridge University Press.
[37] P. KATIS, N. SABADINI, AND R. WALTERS (1997), Bicategories of Processes. *Journal of Pure and Applied Algebra* **115**, 141–178, North-Holland.
[38] R.M. KELLER (1974), Towards a Theory of Universal Speed-Independent Modules. *IEEE Transactions on Computers* vol. **C-23**, 21–33, IEEE Press.
[39] G.M. KELLY (1964), On MacLane's Conditions for Coherence of Natural Associativities, Commutativities, etc. *Journal of Algebra* **1**, 397–402, Acamedic Press.
[40] G.M. KELLY AND M. LAPLAZA (1980), Coherence for Compact Closed Categories. *Journal of Pure and Applied Algebra* **19**, 193–213, North-Holland.
[41] S. MACLANE (1963), Natural Associativity and Commutativity. *Rice University Studies* **49**, pp. 28–46, Rice University Press.
[42] S. MACLANE (1971), *Categories for the Working Mathematician*. Graduate Texts in Mathematics, Springer-Verlag.
[43] A. MAZURKIEWICZ (1977), *Concurrent Program Schemes and their Interpretations*. Technical Report DAIMI PB-**78**, Computer Science Dept., University of Aarhus.
[44] A. MAZURKIEWICZ (1987), Trace theory, in *Petri Nets, Applications and Relationship to other Models of Concurrency*, W. Brauer *et al.* (Eds.), *Lecture Notes in Computer Science* **255**, 279–324, Springer-Verlag.
[45] J. MESEGUER AND U. MONTANARI (1990), Petri Nets are Monoids. *Information and Computation* **88**, 105- 155, Academic Press.
[46] J. MESEGUER, U. MONTANARI, AND V. SASSONE (1992), On the Semantics of Petri Nets, in *Proceedings of CONCUR 92*, R. Cleaveland (Ed.), *Lecture Notes in Computer Science* **630**, 286–301, Springer-Verlag.
[47] J. MESEGUER, U. MONTANARI, AND V. SASSONE (1996), Process versus Unfolding Semantics for Place/Transition Petri Nets. *Theoretical Computer Science* **153**, 171–210, Elsevier.
[48] J. MESEGUER, U. MONTANARI, AND V. SASSONE (1997), On the Semantics of Place/Transition Petri Nets. *Mathematical Structures in Computer Science* **7**, 359–397, Cambridge University Press.
[49] R. MILNER (1980), *A Calculus of Communicating Systems*. Lecture Notes in Computer Science **92**, Springer-Verlag.
[50] R. MILNER (1989), *Communication and Concurrency*. Prentice-Hall.
[51] R. MILNER (1993), Action Calculi or Syntactic Action Structures, in *Proceedings of MFCS 93*, A. Borzyszkowski, S. Sokolowski (Eds.), *Lecture Notes in Computer Science* **711**, 105–121, Springer-Verlag.
[52] E.H. MOORE (1897), Concerning the abstract group of order *k*! isomorphic with the symmetric substitution group on *k* letters. *Proceedings of the London Mathematical Society* **28**, 357–366, Claredon Press.

[53] M. MUKUND (1992), Petri Nets and Step Transition Systems. *International Journal of Foundations of Computer Science* **3**, 443- 478, World Scientific.

[54] M. MUKUND AND M. NIELSEN (1992), Locations and Asynchronous Transition Systems, in *Proceedings of FST & TCS 92*, R. Shyamasundar (Ed.), *Lecture Notes in Computer Science* **652**, 328–341, Springer-Verlag.

[55] M. NIELSEN AND A. CHENG (1995), Observing Behaviour Categorically, in *Proceedings of FST & TCS 95*, P.S. Thiagarajan (Ed.), *Lecture Notes in Computer Science* **1026**, 263–278, Springer-Verlag.

[56] M. NIELSEN AND C. CLAUSEN (1995), Games and Logics for a Noninterleaving Bisimulation. *Nordic Journal of Computing* **2**, 222–250, Publishing Association NJC.

[57] M. NIELSEN, G. PLOTKIN, AND G. WINSKEL (1981), Petri Nets, Event Structures and Domains, Part 1. *Theoretical Computer Science* **13**, 85–108, Elsevier.

[58] M. NIELSEN, L. PRIESE, AND V. SASSONE (1995), Characterizing Behavioural Congruences for Petri Nets, in *Proceedings of CONCUR 95*, I. Lee and S. Smolka (Eds.), *Lecture Notes in Computer Science* **962**, 175–189, Springer-Verlag.

[59] M. NIELSEN, G. ROZENBERG, AND P.S. THIAGARAJAN (1990), Behavioural Notions for Elementary Net Systems. *Distributed Computing* **4**, 45–57, Springer-Verlag.

[60] M. NIELSEN, G. ROZENBERG, AND P.S. THIAGARAJAN (1992), Elementary Transition Systems. *Theoretical Computer Science* **96**, 3–33, Elsevier.

[61] M. NIELSEN, G. ROZENBERG, AND P.S. THIAGARAJAN (1995), Transition Systems, Event Structures and Unfoldings. *Information and Computation* **118**, 191–207, Academic Press.

[62] M. NIELSEN AND G. WINSKEL (1995), Trace Structures and other Models for Concurrency, in *The Book of Traces*, V. Diekert and G. Rozenberg (Eds.), 271–306, World Scientific.

[63] M. Nielsen and G. Winskel (1996), Petri Nets and Bisimulation. *Theoretical Computer Science* **153**, 211–244, Elsevier.

[64] E.R. OLDEROG (1987), A Petri Net Semantics for CCSP, in *Advances in Petri Nets 86*, W. Brauer *et al.* (Eds.), *Lecture Notes in Computer Science* **255**, 196–223, Springer-Verlag.

[65] E.R. OLDEROG (1991), *Nets, Terms and Formulas*. Cambridge Tracts in Theoretical Computer Science, Cambridge University Press.

[66] C.A. PETRI (1962), *Kommunikation mit Automaten*. PhD thesis, Institut für Instrumentelle Mathematik, Bonn.

[67] C.A. PETRI (1973), Concepts of Net Theory, in *Proceedings of MFCS 73*, 137–146, Mathematics Institute of the Slovak Academy of Science.

[68] C.A. PETRI (1977), *Non-Sequential Processes*. Interner Bericht ISF–**77**–**5**, Gesellschaft für Mathematik und Datenverarbeitung, Bonn.

[69] V. PRATT (1986), Modelling Concurrency with Partial Orders. *International Journal of Parallel Programming* **15**, 33–71, Plenum.

[70] L. PRIESE (1983), Automata and Concurrency. *Theoretical Computer Science* **25**, 221–265, Elsevier.

[71] A. RABINOVITCH AND B. TRAKHTENBROT (1988), Behaviour structures and nets. *Fundamenta Informatica* **11**, 357–404, North-Holland.

[72] W. REISIG (1985), *Petri Nets (an Introduction)*. EATCS Monographs on Theoretical Computer Science **4**, Springer-Verlag.

[73] G. ROZENBERG (1987), Behaviour of Elementary Net Systems, in *Advances in Petri Nets 86*, W. Brauer *et al.* (Eds.), *Lecture Notes in Computer Science* **254**, 60–94, Springer-Verlag.

[74] B. ROZOY AND P.S. THIAGARAJAN (1991), Event Structures and Trace Monoids. *Theoretical Computer Science* **91**, 285–313, Elsevier.

[75] V. SASSONE (1995), Axiomatizing Petri Net Concatenable Processes, in *Proceedings of FCT 95*, H. Reichel (Ed.), *Lecture Notes in Computer Science* **962**, 414–423, Springer-Verlag

[76] V. SASSONE (1995), On the Category of Petri Net Computations, in *Proceedings of TAPSOFT 95*, P. Mosses *et al.* (Eds,), *Lecture Notes in Computer Science* **915**, 334–348, Springer-Verlag.

[77] V. SASSONE (1996), An Axiomatization of the Algebra of Petri Net Concatenable Processes. *Theoretical Computer Science* **170**, 277–296, Elsevier.

[78] V. SASSONE (1998), An Axiomatization of the Category of Petri Net Computations. *Mathematical Structures in Computer Science* **8**, 117–151, Cambridge University Press.

[79] V. SASSONE, M. NIELSEN, AND G. WINSKEL (1993), Deterministic Behavioural Models for Concurrency, in *Proceedings of MFCS 93* A. Borzyszkowski, S. Sokolowski (Eds.), *Lecture Notes in Computer Science* **711**, 682–692, Springer-Verlag.

[80] V. SASSONE, M. NIELSEN, AND G. WINSKEL (1996), Models for Concurrency: Towards a Classification. *Theoretical Computer Science* **170**, 297–348, Elsevier.

[81] D. SCOTT (1970), Outline of a Mathematical Theory of Computation, in *Proceedings of 4th Annual Princeton Conference on Information Science and Systems*, 169–176, Princeton University Press.

[82] K. SEGERBERG (1968), Decidability of S4.1. *Theoria* **34**,7–20, Gordon and Breach.

[83] M.W. SHIELDS (1985), Concurrent machines. *Computer Journal* **28**, 449–465, Cambridge University Press.

[84] E.W. STARK (1989), Concurrent Transition Systems. *Theoretical Computer Science* **64**, 221–269, Elsevier.

[85] G. ŞTEFANĔSCU (1987), On Flowchart Theories: Part I. The Deterministic Case. *Journal of Computer and System Sciences* **35**, 163–191, Academic Press.

[86] G. ŞTEFANĔSCU (1987), On Flowchart Theories: Part I. The Nondeterministic Case. *Theoretical Computer Science* **52**, 307–340, Elsevier.

[87] P.S. THIAGARAJAN (1987), Elementary Net Systems, in *Advances in Petri Nets 86*, W. Brauer *et al.* (Eds.), *Lecture Notes in Computer Science* **254**, 26–59, Springer-Verlag.

[88] J. VAN BENTHAM (1984), Correspondence Theory, in *Handbook of Philosophical Logic* vol. **2**, P. Gabbay and F. Guenthner (Eds.), 167–247, Reidel.

[89] W. VOGLER (1991), Deciding History Preserving Bisimilarity. in *Proceedings of ICALP 91*, J. Leach Albert *et al.* (Eds.), *Lecture Notes in Computer Science* **510**, 495–505, Springer-Verlag.

[90] G. WINSKEL (1982), Event Structure Semantics for CCS and related languages, in *Proceedings of the 9th ICALP*, M. Nielsen and E.M. Schmidt (Eds.), *Lecture Notes in Computer Science* **140**, 561–576, Springer-Verlag.

[91] G. WINSKEL (1984), A New Definition of Morphism on Petri Nets, in *Proceedings of STACS 84*, M. Fontet and K. Mehlhorn (Eds.), *Lecture Notes in Computer Science* **166**, 140–150, Springer-Verlag.

[92] G. WINSKEL (1984), Categories of Model for Concurrency, in *Seminar on Concurrency*, S. Brookes *et al.* (Eds.), *Lecture Notes in Computer Science* **197**, 246–267, Springer-Verlag.

[93] G. WINSKEL (1987), Petri Nets, Algebras, Morphisms and Compositionality. *Information and Computation* **72**, 197- 238, Academic Press.

[94] G. WINSKEL (1986), Event Structures, in *Advances in Petri Nets 86*, W. Brauer *et al.* (Eds.), *Lecture Notes in Computer Science* **255**, 325–392, Springer-Verlag.

[95] G. WINSKEL (1988), An Introduction to Event Structures, in *Linear time, branching time, and partial order in logics and models for concurrency*, J.W. de Bakker *et al.* (Eds.), *Lecture Notes in Computer Science* **354**, 365–397, Springer-Verlag.

[96] G. WINSKEL (1996), A Presheaf Semantics of Value-Passing Processes, in *Proceedings of CONCUR 96*, U. Montanari and V. Sassone (Eds.), *Lecture Notes in Computer Science* **1119**, 98–114, Springer-Verlag.

[97] G. WINSKEL AND M. NIELSEN (1995), Models for Concurrency, in *Handbook of Logic in Computer Science* vol. **4**, S. Abramsky *et al.* (Eds), 1–148, Oxford University Press.

[98] G. WINSKEL AND M. NIELSEN (1997), Presheaves as Transition Systems, in *Proceedings of POMIV 96*, D. Peled *et al.* (Eds.), *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **29**, 129–140, American Mathematical Society Press.

[99] M. YOELI (1973), *Petri Nets and Asynchronous Control Networks*. Research Report CS–**73**–**07**, Department of Applied Mathematics and Computer Science, University of Waterloo.

BRICS, **B**ASIC **R**ESEARCH **I**N **C**OMPUTER **S**CIENCE, DEPT. OF COMPUTER SCIENCE, UNIVERSITY OF AARHUS, NY MUNKEGADE BLD. 540, DK-8000 ÅRHUS C, DENMARK

E-mail address: mnielsen@brics.dk

QUEEN MARY AND WESTFIELD COLLEGE, UNIVERSITY OF LONDON, MILE END ROAD, LONDON E1 4NS, UK

E-mail address: vs@dcs.qmw.ac.uk