BRICS

Basic Research in Computer Science

BRICS RS-98-17 Bruni et al.: Petri Nets and the Collective Token Philosophy

A Comparison of Petri Net Semantics under the Collective Token Philosophy

Roberto Bruni
José Meseguer
Ugo Montanari
Vladimiro Sassone

BRICS Report Series **RS-98-17**

ISSN 0909-0878 **September 1998**

See back inner page for a list of recent BRICS Report Series publications. Copies may be obtained by contacting:

BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK–8000 Aarhus C
Denmark
Telephone: +45 8942 3360
Telefax: +45 8942 3255
Internet: BRICS@brics.dk

BRICS publications are in general accessible through the World Wide Web and anonymous FTP through these URLs:

`http://www.brics.dk`
`ftp://ftp.brics.dk`
This document in subdirectory `RS/98/17/`

A Comparison of Petri Net Semantics under the Collective Token Philosophy

Roberto Bruni José Meseguer Ugo Montanari
Vladimiro Sassone

Abstract

In recent years, several semantics for *place/transition Petri nets* have been proposed that adopt the *collective token philosophy*. We investigate distinctions and similarities between three such models, namely *configuration structures*, *concurrent transition systems*, and *(strictly) symmetric (strict) monoidal categories*. We use the notion of adjunction to express each connection. We also present a purely logical description of the collective token interpretation of net behaviours in terms of theories and theory morphisms in *partial membership equational logic*.

Introduction

Petri nets, introduced by Petri in [17] (see also [18]), are one of the most widely used and representative *models for concurrency*, because of the simple formal description of the net model, and of its natural characterisation of *concurrent* and *distributed systems*. The extensive use of Petri nets has given rise to different schools of thought concerning the semantical interpretation of nets, with each view justified either by the theoretical characterisation of different properties of the modelled systems, or by the architecture of possible implementations.

A real dichotomy runs on the distinction between *collective* and *individual token philosophies* noticed, e.g., in [6]. According to the collective token philosophy, net semantics should not distinguish among different instances of the idealised resources (the so-called 'tokens') that rule the basics of net behaviour. The rationale for this being, of course, that any such instance is *operationally* equivalent to all the others. As obvious as this is, it disregards that operationally equivalent resources may have different origins and histories, and may, therefore, carry different *causality* information. Selecting one

The first and fourth author thank the support by **BRICS** — **B**asic **R**esearch **i**n **C**omputer **S**cience, *Centre of the Danish National Research Foundation*.

The first three authors have been partly supported by Office of Naval Research Contracts N00014-95-C-0225 and N00014-96-C-0114, by National Science Foundation Grant CCR-9633363, and by the Information Technology Promotion Agency, Japan, as part of the Industrial Science and Technology Frontier Program 'New Models for Software Architecture' sponsored by NEDO (New Energy and Industrial Technology Development Organization). Also supported in part by US Army contract DABT63-96-C-0096 (DARPA); CNR Integrated Project *Metodi e Strumenti per la Progettazione e la Verifica di Sistemi Eterogenei Connessi mediante Reti di Comunicazione*; and Esprit Working Groups *CONFER2* and *COORDINA*.

instance of a resource rather than the other, may be as different as being or not being causally dependent on some previous event. And this may well be an information one is not ready to discard, which is the point of view of the individual token philosophy.

In this paper, however, we focus on the collective token interpretation as the first step of a wider programme aimed at investigating the two approaches and their mutual relationships in terms of the behavioural, algebraic, and logical structures that can give adequate semantics account of each of them.

Starting with the classical 'token-game' semantics, many behavioural models for Petri nets have been proposed that follow the collective token philosophy. In fact, too many to be systematically reviewed here. Among all these, however, a relatively recent proposal of van Glabbeek and Plotkin is that of *configuration structures* [6]. Clearly inspired by the domains of configurations of *event structures* [22], these are simply collections of (multi)sets that, at the same time, represent the legitimate system states and the system dynamics, i.e., the transitions between such states. One of the themes of this paper is to compare configuration structure with the algebraic model based on *monoidal categories* [11], which also adopts the collective token philosophy and which provides a precise algebraic reinterpretation [5] of yet another model, namely the *commutative processes* of Best and Devillers [1]. In particular, we shall observe that configuration structures are *too abstract* a model, i.e., that they make undesirable identifications of nets, and conclude that monoidal categories provide a superior model of net behaviour.

To illustrate better the differences between the two semantic frameworks above, we adopt *concurrent transition systems* as a bridge-model. These are a much simplified, deterministic version of *higher dimensional transition systems* [3] that we select as the simplest one able to convey our ideas. Concurrent transition systems resemble configuration structures, but are more expressive. They also draw on earlier very significant models, such as *distributed transition systems* [9], *step* and *PN transition systems* [16], and *local event structures* [8]. Moreover, the equivalence of the behavioural semantics of concurrent transition systems and the algebraic semantics of monoidal categories can be stated very concisely. As we explain also in this paper, the algebraic semantics is itself amenable to a purely logical description in terms of theories in *partial membership equational logic* [10].

The main result of this research is a new precise characterisation of the relationships between all these behavioural, algebraic, and logical models within the collective token philosophy. We show that Best-Devillers commutative processes, the algebraic monoidal category model, and the concurrent transition system behavioural model all coincide in the precise sense of being related by equivalences of categories. And we also show how the behavioural model afforded by configuration structures is too abstract, but is precisely related to all the above models by a natural transformation that characterises the identification of inequivalent nets and behaviours caused by configuration structures.

The structure of the paper is as follows. In Section 1 we recall the basic definitions about PT Petri nets, remarking the distinction between the collective and individual token philosophies, and we introduce the frameworks under comparison, i.e., configuration structures, concurrent transition systems, and monoidal categories (also in their membership equational logic), discussing for each of them the corresponding models that they associate to a Petri net. Section 2 and Section 3 compare concurrent transition systems with, respectively, monoidal categories and configuration structures. Finally, the concluding section describes related work on the individual token philosophy.

1 Background

1.1 Petri Nets and the Collective Token Philosophy

Place/transition nets, the most widespread flavour of Petri nets, are graphs with distributed states described by (finite) distributions of resources ('tokens') in 'places'. These are usually called *markings* and represented as multisets $u\colon S \to \mathbb{N}$, where $u(a)$ indicates the number of tokens that place a carries in u. We shall use $\mu(S)$ to indicate the set of *finite* multisets on S, i.e., multiset that yield a zero on all but finitely many $a \in S$. Multiset union makes $\mu(S)$ a free commutative monoid on S.

DEFINITION 1. A *place/transition* (*PT* for short) *Petri net* N is a tuple $(\partial_0, \partial_1, S, T)$, where S is a set of *places*, T is a set of *transitions*, $\partial_0, \partial_1\colon T \to \mu(S)$ are functions assigning, respectively, source and target to each transition.

Informally, $\partial_0(t)$ prescribes the minimum amount of resources needed to enable t, whilst $\partial_1(t)$ describe the resources that the occurrence of t contributes to the global state. This is made explicit in the following definition, where we shall indicate multiset inclusion, union, and difference by, respectively, $\subseteq$, $+$, and $-$.

DEFINITION 2. Let u and v be markings and X a finite multiset of transitions of a net N. We say that u evolves to v under the *step* X, in symbols $u\ [X\rangle\ v$, if the transitions in X are concurrently enabled at u i.e., $\sum_{t\in T_N} X(t)\cdot\partial_0(t) \subseteq u$, and

$$v = u + \textstyle\sum_{t\in T_N} X(t)\cdot(\partial_1(t) - \partial_0(t)).$$

A *step sequence* from u_0 to u_n is a sequence $u_0\ [X_1\rangle\ u_1 \ldots u_{n-1}\ [X_n\rangle\ u_n$.

PT nets are often considered together with a state: a *marked* PT net N is a PT net $(\partial_0, \partial_1, S, T)$ together with an *initial marking* $u_0 \in \mu(S)$. In order to equip PT nets with a natural notion of morphism, since that $\mu(S)$ is a monoid under $+$ with unit $\varnothing$, we consider maps of transition systems that preserve the additional structure.

DEFINITION 3. A *morphism* of nets from $N = (\partial_0, \partial_1, S, T)$ to $N' = (\partial_0', \partial_1', S', T')$ is a pair $\langle f_t, f_p\rangle$ where $f_t\colon T \to T'$ is function, $f_p\colon \mu(S) \to \mu(S')$ is homomorphism of monoids such that $\partial_i' \circ f_t = f_p \circ \partial_i$, for $i = 0, 1$. A morphism of marked nets is a morphism of nets such that $f_p(u_0) = u_0'$.
We shall use $\mathbf{Petri}$ (respectively $\mathbf{Petri}_*$) to indicate the category of (marked) PT nets and their morphisms with the obvious componentwise composition of arrows.

To compare the effects of the collective and of the individual token philosophy on observing causal relations between fired transitions, let us consider the example in Figure 1 that we adapt from [6]. (As usual, boxes stand for transitions, circles for places, dots for tokens, and oriented arcs represent ∂_0 and ∂_1.)

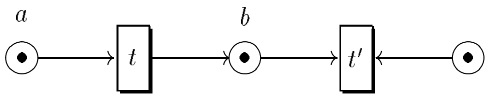

FIGURE 1

Observe that the firing of t produces a second token in place b. According to the individual token philosophy, it makes a difference whether t' consumes the token b originated from the firing of t, or the one coming from the initial marking. In the first case the occurrence of t' causally depends on that of t, and in the second the two firings are independent. In the collective token philosophy, instead, the two firings are always considered to be concurrent, because the firing of t does not change the enabling condition of t'.

1.2 Configuration Structures

In the same paper where they introduce the distinction between collective token and individual token philosophy, van Glabbeek and Plotkin propose *configuration structures* to represent the behaviour of nets according to the collective token philosophy. These are structures inspired by event structures [22] whose dynamics is uniquely determined by an explicitly-given set of possible configurations of the system. However, the structures they end up associating to nets are not exactly configuration structures. They enrich them in two ways: firstly, by considering *multi*sets instead of sets of occurrences, and secondly, by using an explicit transition relation between configurations. While the first point can be handled easily, as we do below, the second one seems to compromise the basic ideas underlying the framework and to show that configuration structures do not offer a faithful representation of the behaviour of nets under the collective token philosophy.

DEFINITION 4. A *configuration structure* is given by a set E and a collection C of finite multisets over the set E. The elements of E are called *events*, and the elements of C *configurations*.

The idea is that an event is an occurrence of an action the system may perform, and that a configuration X represents a state of the system, which is determined by the collection X of occurred events. The set C of admissible configurations yields a relation representing how the system can evolve from one state to another.

DEFINITION 5. Let (E, C) be a configuration structure. For X, Y in C we write $X \longrightarrow Y$ if

(1) $X \subset Y$,

(2) $Y - X$ is finite,

(3) for any multiset Z such that $X \subset Z \subset Y$, we have $Z \in C$.

The relation $\longrightarrow$ is called the *step transition relation*.

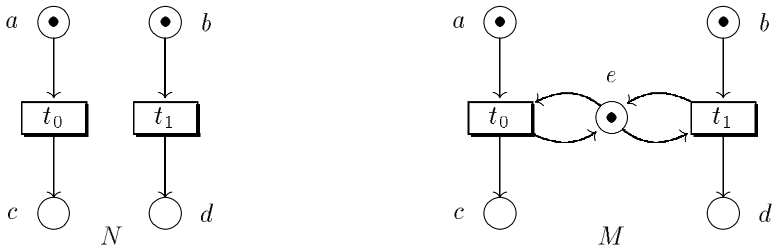

FIGURE 2. The nets N and M of our running example.

Intuitively, $X \longrightarrow Y$ means that the system can evolve from state X to state Y by performing the events in $Y - X$ *concurrently*. To stress this we shall occasionally write $X \xrightarrow{L} Y$, with $L = Y - X$. Observe that the last condition states that the events in $Y - X$ can be performed concurrently if and only if they can be performed in any order. In our opinion, this requirement embodies an *interleaving*-oriented view, as it reduces concurrency to nondeterminism. As we explain below, we view this as the main weakness of configuration structures.

In the following definition we slightly refine the notion of net configuration proposed in [6], as this may improperly include multisets of transitions that cannot be fired from the initial marking.

DEFINITION 6 (From PT Nets to Config. Structures [6])**.** Let $N = (\partial_0, \partial_1, S, T, u_0)$, be a *marked* PT net. A finite multiset X of transitions is called *fireable* if there exists a partition $X_1, ..., X_n$ of X such that $u_0\ [X_1\rangle\ u_1 ... u_{n-1}\ [X_n\rangle\ u_n$ is a step sequence. A *configuration* of N is a fireable multiset X of transitions. The configuration structure associated to N is $cs(N) = (T, C_N)$, where C_N is the set of configurations of N.

It follows that for each configuration X the function $u_X : S \to \mathbb{Z}$ given by

$$u_X = u_0 + \sum_{t \in T} X(t) \cdot (\partial_0(t) - \partial_1(t))$$

is a (reachable) marking, i.e., $0 \leq u_X(a)$ for all $a \in S$. Moreover, if X is a configuration and $u_X\ [U\rangle\ v$, then $X + U$ is also a configuration and $v = u_{X+U}$.

Generally speaking, if N is a *pure net*, i.e., a net with no *self-loops*, $cs(N)$ can be considered a reasonable semantics for N. Otherwise, as observed also in [6], it is not a good idea to reduce N to $cs(N)$. Consider for example, the marked nets N and M of Figure 2. They have very different behaviours, indeed: in N the actions t_0 and t_1 are concurrent, whereas in M they are mutually exclusive. However, since in M any interleaving of t_0 and t_1 is possible, the diagonal $\varnothing \longrightarrow \{t_0, t_1\}$ sneaks into the structure by definition. As a result, both N and M yield the configuration structure represented in Figure 3, even though $\{t_0, t_1\}$ is *not* an admissible step for M. The limit case is the marked net consisting of a single self-loop: the readers can check for themselves that, according to $cs(\_)$, it can fire arbitrarily large steps.

These problems have prompted us to look for a semantic framework that represents net behaviours more faithfully than configuration structures. The key observation is that

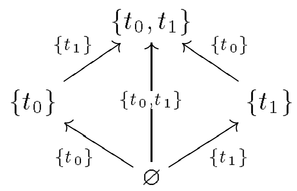

FIGURE 3. The configuration structure $cs(N) = cs(M)$ for the nets N and M.

there is nothing wrong with the assumption that if a step involving many parallel actions can occur in a certain state, then all the possible interleaving sequences of those action can also occur from that state. The problematic bit is assuming the inverse implication, because, as a matter of fact, it reduces concurrency to nondeterminism and makes the set of configurations determine uniquely the transition relation. Our proposed solution is concurrent transition systems.

1.3 Concurrent Transition Systems

The analysis of the previous section suggests seeking a model that enforces the existence of all appropriate interleavings of steps, without allowing this to determine the set of transitions completely. Several such models appear in the literature. Among those that inspired us most, we recall *distributed transition systems* [9], *step transition systems* [16], *PN transition systems* [16], and *higher dimensional transition systems* [3]. Also closely related are the *local event structures* of [8], a model that extends event structures (rather than transition systems) by allowing the firing of sets (but *not* multisets) of events. Drawing on all these, we have here chosen the simplest definition that suits our current aim.

DEFINITION 7. A *concurrent transition system* (CTS for short) is a structure $H = (S, L, \mathit{trans}, s_0)$, where S is a set of *states*, L is a set of *actions*, $s_0 \in S$ is the initial state, and $\mathit{trans} \subseteq S \times (\mu(L) - \{\varnothing\}) \times S$ is a set of *transitions*, such that:

(1) if $(s, U, s_1), (s, U, s_2) \in \mathit{trans}$, then $s_1 = s_2$,

(2) if $(s, U, s') \in \mathit{trans}$ and U_1, U_2 is a partition of U, then there exist $v_1, v_2 \in S$ such that $(s, U_1, v_1), (s, U_2, v_2), (v_1, U_2, s'), (v_2, U_1, s') \in \mathit{trans}$.

Condition (1) above states that the execution of a multiset of labels U in a state s deterministically leads to a different state. The second condition guarantees that all the possible interleavings of the actions in U are possible paths from s to s' if $(s, U, s') \in \mathit{trans}$. Notice that, by (1), the states v_1 and v_2 of (2) are uniquely determined.

We formalise the idea that different paths which are different interleavings of the same concurrent step can be considered equivalent.

DEFINITION 8. A *path* in a CTS is a sequence of contiguous transitions

$$(s, U_1, s_1)(s_1, U_2, s_2) \cdots (s_{n-1}, U_n, s_n).$$

A *run* is a path that originates from the initial state.

DEFINITION 9. Given a CTS H, *adjacency* is the least reflexive, symmetric, binary relation $\leftrightarrow_H$ on the paths of H which is closed under path concatenation and such that

$$(s, U_1, s_1)(s_1, U_2, s_2) \leftrightarrow_H (s, U_1 + U_2, s_2).$$

Then, the *homotopy* relation $\Leftrightarrow_H$ on the paths of H is the transitive closure of $\leftrightarrow_H$. The equivalence classes of runs of H with respect to the homotopy relation are called *computations*.

In order to simplify our exposition, we now refine the notion of concurrent transition system so as to be able to associate to each path between two states the same multiset of actions. As we shall see, such transition systems enjoy interesting properties.

DEFINITION 10. A CTS is *uniform* if all its states are *reachable* from the initial state, and the union of the actions along any two *cofinal* runs yield the same multiset, where cofinal means ending in the same state.

In a uniform CTS $H = (S, L, trans, s_0)$ each state s can be associated with the multiset of actions on any run to s. Precisely, we shall use ς_s to indicate $\sum_{i=1}^{n} U_i$, for $(s_0, U_1, s_1)(s_1, U_2, s_2)...(s_{n-1}, U_n, s)$ a run of H. Observe also that uniform CTS are necessarily acyclic, because any cycle $(s, U_0, s_1)\ldots(s_n, U_n, s)$ would imply the existence of runs to s carrying different actions. In the rest of the paper, we shall consider *only* uniform concurrent transition systems.

Introducing the natural notion of computation-preserving morphism for CTS, we define a category of uniform concurrent transition systems. In the following, for functions $f: A \to B$, we denote by $f^\mu : \mu(A) \to \mu(B)$ the obvious multiset extension of f, i.e.,

$$f^\mu(X)(b) = \sum_{a \in f^{-1}(b)} X(a).$$

DEFINITION 11. For H_1 and H_2 CTS, a *morphism* from H_1 to H_2 consists of a map $f: S_1 \to S_2$ that preserves the initial state and a function $\alpha: L_1 \to L_2$ and such that $(s, U, s') \in trans_1$ implies $(f(s), \alpha^\mu(U), f(s')) \in trans_2$.

We denote by $\mathbf{CTS}$ the category of uniform CTS and their morphisms.

DEFINITION 12 (From PT Nets to CTS)**.** Let $N = (\partial_0, \partial_1, S, T, u_0)$ be a marked PT Petri net. The concurrent transition system associated to N is

$$ct(N) = (M_N, T, trans_N, \varnothing),$$

where M_N is the set of *fireable* multisets of transitions of N, and $(X, U, X') \in trans_N$ if and only if $u_X\ [U\rangle\ u_{X'}$. (Recall that $u_X : S \to \mathbb{Z}$ is by definition a reachable marking.)

Although this construction is formally very close to that proposed for configuration structures, the difference is that CTS do not enforce diagonals to fill the squares: these are introduced if and only if the associated step is actually possible (see Figure 4). We shall give a precise categorical characterisation of the representations of nets in the CTS framework in Section 2. For the time being, we notice the following.

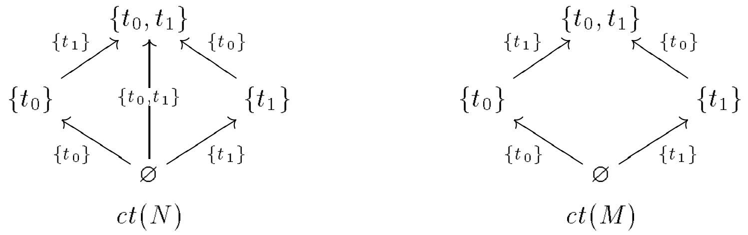

FIGURE 4. The CTS $ct(N)$ and $ct(M)$ for the nets N and M of Figure 2.

PROPOSITION 1. *$ct(N)$ is a functor from* $\mathbf{Petri}_*$ *to* $\mathbf{CTS}$.

Although all cofinal runs of a CTS carry the same multiset of actions, it is not the case that all such runs are homotopic, i.e., they do not necessarily represent the same computation. Enforcing this is the purpose of the next definition.

DEFINITION 13. An *occurrence* concurrent transition system is a concurrent transition system H in which all pairs of *cofinal* transitions $(s_1, U_1, s), (s_2, U_2, s) \in \mathit{trans}_H$ are the final steps of *homotopic* paths.

It can be shown that the previous definition implies the following property.

PROPOSITION 2. *All cofinal paths of an occurrence CTS are homotopic.*

We shall use $\mathbf{oCTS}$ to indicate the full subcategory of $\mathbf{CTS}$ consisting of occurrence CTS. Clearly, a uniform CTS can be unfolded into an occurrence CTS.

DEFINITION 14 (From CTS to Occurrence CTS)**.** Let $H = (S, L, \mathit{trans}, s_0)$ be a concurrent transition system. Its *unfolding* is the occurrence concurrent transition system $\mathcal{O}(H) = (S', L, \mathit{trans}', \epsilon)$, where S' is the collection of computations of H, and

$$\mathit{trans}' = \big\{([\pi]_{\Leftrightarrow}, U, [\pi']_{\Leftrightarrow}) \bigm| \exists s, s' \in S,\ [\pi']_{\Leftrightarrow} \in S',\ \pi' \Leftrightarrow_H \pi(s, U, s')\big\}.$$

PROPOSITION 3. *$\mathcal{O}(\_)$ extends to a right adjoint to the inclusion of* $\mathbf{oCTS}$ *in* $\mathbf{CTS}$.

Proof. For H a concurrent transition system, consider $\varepsilon_H : \mathcal{O}(H) \to H$ that maps each $[\pi]_{\Leftrightarrow} \in S_{\mathcal{O}(H)}$ to its final state $s \in S_H$. It is easy to verify that this forms the counit of the adjunction. □

1.4 Monoidal Categories

Several interesting aspects of Petri net theory can be profitably developed within category theory, see e.g. [21, 11, 2]. Here we focus on the approach initiated in [11] (other relevant references are [5, 13, 19, 15, 20]) which exposes the monoidal structure of Petri nets under the operation of parallel composition. In [11, 5] it is shown that the sets of transitions can be endowed with appropriate algebraic structures in order to capture some basic constructions on nets. In particular, the *commutative processes* by Best and Devillers [1], which represent the natural behavioural model for PT nets under the collective token philosophy, can be characterised adding a functorial *sequential* composition on the *monoid* of steps, thus yielding a strictly symmetric strict monoidal category $\mathcal{T}(N)$.

DEFINITION 15. For N a PT net, let $\mathcal{T}(N)$ be the strictly symmetric strict monoidal category freely generated by N.

Using $\mathbf{CMonCat}$ to denote the category of strictly symmetric strict monoidal categories and strict monoidal functors, $\mathcal{T}(\_)$ is a functor from $\mathbf{Petri}$ to $\mathbf{CMonCat}$. The category $\mathcal{T}(N)$ can be inductively defined by the following inference rules and axioms.

$$\frac{u \in \mu(S_N)}{id_u \colon u \to u \in \mathcal{T}(N)} \qquad \frac{t \in T_N,\ \partial_0(t) = u,\ \partial_1(t) = v}{t \colon u \to v \in \mathcal{T}(N)}$$

$$\frac{\alpha \colon u \to v,\ \beta \colon u' \to v' \in \mathcal{T}(N)}{\alpha \oplus \beta \colon u + u' \to v + v' \in \mathcal{T}(N)} \qquad \frac{\alpha \colon u \to v,\ \beta \colon v \to w \in \mathcal{T}(N)}{\alpha ; \beta \colon u \to w \in \mathcal{T}(N)}$$

where the following equations, stating that $\mathcal{T}(N)$ is a strictly symmetric strict monoidal category, are satisfied by all arrows α, α', β, β', γ, δ and all multisets u and v:

neutral:	$id_\varnothing \oplus \alpha = \alpha,$	
commutativity:	$\alpha \oplus \beta = \beta \oplus \alpha,$	
associativity:	$(\alpha \oplus \beta) \oplus \delta = \alpha \oplus (\beta \oplus \delta),$	$(\alpha;\beta);\gamma = \alpha;(\beta;\gamma),$
identities:	$\alpha; id_u = \alpha = id_v; \alpha,$	$id_u \oplus id_v = id_{u+v},$
functoriality:	$(\alpha;\beta) \oplus (\alpha';\beta') = (\alpha \oplus \alpha');(\beta \oplus \beta').$	

The intuition here is that arrows are step sequences and arrow composition is their concatenation, whereas the monoidal operator $\oplus$ allows for parallel composition. It turns out that this algebraic structure describes precisely the processes à la Best and Devillers.

PROPOSITION 4 (cf. [11])**.** *The presentation of $\mathcal{T}(N)$ given above provides a complete and sound axiomatisation of the algebra of the commutative processes of N.*

By analogy with $\mathbf{Petri}_*$, we take a pointed category $(\mathbf{C}, c_0)$ to be a category $\mathbf{C}$ together with a distinguished object $c_0 \in \mathbf{C}$. Similarly, a pointed functor from $(\mathbf{C}, c_0)$ to $(\mathbf{D}, d_0)$ is a functor $F \colon \mathbf{C} \to \mathbf{D}$ that maps the distinguished object c_0 to the distinguished object d_0. Then, using $\mathbf{CMonCat}_*$ to denote the category of pointed strictly symmetric strict monoidal categories and their pointed functors, the previous construction extends immediately to a functor $\mathcal{T}_*(N) \colon \mathbf{Petri}_* \to \mathbf{CMonCat}_*$, such that for $N = (\partial_0, \partial_1, S, T, u_0)$ a marked PT net, then

$$\mathcal{T}_*(N) = (\mathcal{T}(\partial_0, \partial_1, S, T), u_0).$$

1.5 A Logical Characterisation of the Algebraic Model

The algebraic semantics of PT Petri nets can be expressed very compactly by means of a morphism between theories in *partial membership equational logic* (**PMEqtl**) [10], a logic of partial algebras with subsorts and subsort polymorphism whose *sentences* are

Horn clauses on equations $t = t'$ and membership assertions $t : s$. Such a characterisation can have also practical applications, as there are tools available that support executable specifications in partial algebras. This section and the Appendix provide an informal introduction to the main ideas of **PMEqtl**. The interested reader is referred to [10, 12] for self-contained presentations.

A *theory* in **PMEqtl** is a pair $T = (\Omega, \Gamma)$, where Ω is a signature over a *poset* of sorts and Γ is a set of **PMEqtl**-sentences in the language of Ω. We denote by $\mathbf{PAlg}_\Omega$ the category of partial Ω-algebras, and by $\mathbf{PAlg}_T$ its full subcategory consisting of T-algebras, i.e., those partial Ω-algebras that satisfy all the sentences in Γ.

The features of **PMEqtl** (partiality, poset of sorts, membership assertions) offer a natural framework for the specification of categorical structures. For instance, a notion of *tensor product* for partial algebraic theories is used in [12] to obtain, among other things, a very elegant definition of the theory of monoidal categories that we recall in the Appendix. More precisely, we define the theories `PETRI` of PT nets and `CMONCAT` of strictly symmetric strict monoidal categories, using a self-explanatory Maude-like notation (Maude [4] is a language recently developed at SRI International; it is based on rewriting logic and supports the execution of membership equational logic specifications).

To study the relationships between `PETRI` and `CMONCAT`, the Appendix defines also an intermediate theory `CMON-AUT` of automata whose states form a commutative monoid. Our main result is then that the composition of the obvious inclusion functor of $\mathbf{Petri}$ into $\mathbf{PAlg}_{\mathtt{CMON-AUT}}$ and the free functor $\mathcal{F}_{\mathtt{V}}$ from $\mathbf{PAlg}_{\mathtt{CMON-AUT}}$ to $\mathbf{PAlg}_{\mathtt{CMONCAT}}$ associated to the theory morphism `V` from `CMON-AUT` to `CMONCAT` corresponds exactly to the functor $\mathcal{T}(\_) \colon \mathbf{Petri} \to \mathbf{CMonCat}$.

PROPOSITION 5. *The functor* $\mathcal{T}(\_) \colon \mathbf{Petri} \to \mathbf{CMonCat}$ *is the composition*

$$\mathbf{Petri} \hookrightarrow \mathbf{PAlg}_{\mathtt{CMON-AUT}} \xrightarrow{\mathcal{F}_V} \mathbf{PAlg}_{\mathtt{CMONCAT}}$$

2 Concurrent Transition Systems and Monoidal Categories

In this section we state the faithfulness of the CTS representation of nets, as given in Definition 12, with respect to the collective token philosophy. To accomplish this aim, we show that both the $ct(\_)$ and the $\mathcal{T}(\_)$ constructions yield two equivalent categories of net behaviours.

Regarding the monoidal approach, the obvious choice consists in taking the comma category of $\mathcal{T}(N)$ with respect to the initial marking, thus yielding a category whose objects are the commutative processes of N from its initial marking. An arrow from process p to process q is then the *unique* commutative process r such that $p; r = q$ in $\mathcal{T}(N)$. We denote the resulting category by $(u_0 \downarrow \mathcal{T}(N))$.

An analogous construction can be defined starting from $ct(N)$. The first step is to observe that the paths of a generic CTS under the homotopy relation define a category.

DEFINITION 16. For $H = (S, L, \mathit{trans}, s_0)$ a CTS, we define the *category of computations* of H to be the category $\mathcal{C}(H)$ whose

▷ *objects* are computations $[\pi]_{\Leftrightarrow}$ of H,

▷ *arrows* are the homotopy equivalence classes of paths in H such that

$$[\psi]_{\Leftrightarrow} \colon [\pi]_{\Leftrightarrow} \to [\pi']_{\Leftrightarrow} \quad \text{iff} \quad \pi' \Leftrightarrow_H \pi\psi,$$

▷ *arrow composition* is defined as the homotopy class of path concatenation, i.e.,

$$[\psi]_{\Leftrightarrow} ; [\psi']_{\Leftrightarrow} = [\psi\psi']_{\Leftrightarrow},$$

▷ *identity arrow* at $[\pi]_{\Leftrightarrow}$ is $\epsilon_{[\pi]_{\Leftrightarrow}}$, the homotopy class of the empty path at the final state of π.

This construction extends easily to a functor $\mathcal{C}(\_)$ from $\mathbf{CTS}$ to $\mathbf{Cat}$, the category of (small) categories and functors, yielding a functor $\mathcal{C}(ct(\_))$ from $\mathbf{Petri}_*$ to $\mathbf{Cat}$. Observe also that $\mathcal{C}(\_)$ factors through $\mathcal{O} \colon \mathbf{CTS} \to \mathbf{oCTS}$ via the obvious path construction.

THEOREM 1. *Let N be marked PT net with initial marking u_0. Then, the categories $\mathcal{C}(ct(N))$ and $(u_0 \downarrow \mathcal{T}(N))$ are isomorphic.*

Proof. We sketch the definition of functors

$$\mathsf{F} \colon (u_0 \downarrow \mathcal{T}(N)) \to \mathcal{C}(ct(N)) \quad \text{and} \quad \mathsf{G} \colon \mathcal{C}(ct(N)) \to (u_0 \downarrow \mathcal{T}(N))$$

inverses to each other. The functor F maps an object of the comma category to the homotopy class of any of the object's interleaving (which is well-defined because of the diamond equivalence of [1]). Its action on morphisms is analogous.

On the other hand, for a computation $[\pi]_{\Leftrightarrow}$ in $\mathcal{C}(ct(N))$, starting from the initial marking we can determine uniquely the corresponding arrow on $\mathcal{T}(N)$, and therefore define the action of G on both objects and arrows. □

The categories of computations for the concurrent transition systems associated to nets N and M of Figure 2 are shown in Figure 5, where we use c_0 and c_1 to denote, respectively, the computations $[(\varnothing, \{t_0\}, \{t_0\})]_{\Leftrightarrow}$, and $[(\varnothing, \{t_1\}, \{t_1\})]_{\Leftrightarrow}$ in both of $ct(N)$ and $ct(M)$. Analogously, p_1 and p_0 indicate the homotopy classes of the paths $[(\{t_0\}, \{t_1\}, \{t_0, t_1\})]_{\Leftrightarrow}$ and $[(\{t_1\}, \{t_0\}, \{t_0, t_1\})]_{\Leftrightarrow}$, respectively. However, $c_0; p_1$ and $c_1; p_0$ yield the same result $c = [(\varnothing, \{t_0, t_1\}, \{t_0, t_1\})]_{\Leftrightarrow}$ in $\mathcal{C}(ct(N))$, whereas in $\mathcal{C}(ct(M))$ they denote different objects: $c' = [(\varnothing, \{t_0\}, \{t_0\})(\{t_0\}, \{t_1\}, \{t_0, t_1\})]_{\Leftrightarrow}$ and $c'' = [(\varnothing, \{t_1\}, \{t_1\})(\{t_1\}, \{t_0\}, \{t_0, t_1\})]_{\Leftrightarrow}$.

3 Configuration Structures and Concurrent Transition Systems

In this section we first give a categorical structure to the class of configuration structures, and then show that the obvious injection of configuration structures into CTS yields a reflection.

$$
\begin{tikzcd}
 & c & \\
c_0 \arrow[ur, "p_1"] & & c_1 \arrow[ul, "p_0"'] \\
 & \epsilon \arrow[ul, "c_0"] \arrow[ur, "c_1"'] \arrow[uu, "c"'] &
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
c' & & c'' \\
c_0 \arrow[u, "p_1"] & & c_1 \arrow[u, "p_0"'] \\
 & \epsilon \arrow[ul, "c_0"] \arrow[ur, "c_1"'] &
\end{tikzcd}
$$

$\mathcal{C}(ct(N))$ $\qquad$ $\mathcal{C}(ct(M))$

FIGURE 5. The categories $\mathcal{C}(ct(N))$ and $\mathcal{C}(ct(M))$ for the nets of Figure 2.

DEFINITION 17. For (E_1, C_1) and (E_2, C_2) configuration structures, a *cs-morphism* from (E_1, C_1) to (E_2, C_2) is a function $g \colon E_1 \to E_2$ such that for each configuration $X \in C_1$, then $g^\mu(X) \in C_2$. We denote by $\mathbf{CSCat}$ the category of configuration structures and cs-morphisms.

The obvious injection functor $\mathcal{I}(\_)$ from $\mathbf{CSCat}$ to $\mathbf{CTS}$ maps a configuration structure $CS = (E, C)$ into the concurrent transition system

$$\mathcal{I}(CS) = (C, E, trans_{CS}, s_0),$$

where $trans_{CS} = \{(X, L, Y) \mid X \xrightarrow{L} Y\}$, and maps a cs-morphism $g \colon E_1 \to E_2$ to the morphism (g', g), where $g' \colon C_1 \to C_2$ is the obvious extension g^μ of g to multisets, with domain restricted to C_1.

THEOREM 2. *The functor $\mathcal{I}(\_) \colon \mathbf{CSCat} \to \mathbf{CTS}$ is the right adjoint of a functor $\mathcal{R}(\_) \colon \mathbf{CTS} \to \mathbf{CSCat}$. Moreover, since the counit of the adjunction is the identity, $\mathcal{I}(\_)$ and $\mathcal{R}(\_)$ define a full reflection.*

Proof. We sketch the proof, giving the precise definition of the reflection functor. The reflection functor $\mathcal{R}(\_)$ maps a uniform CTS $H = (S, L, trans, s_0)$ into the configuration structure $\mathcal{R}(H) = (L, C_S)$ such that $C_S = \{\varsigma_s \mid s \in S\}$ (recall that ς_s is the multiset union of the actions of any run leading to s). $\square$

We denote the component at H of the unit of the adjunction by $\rho_H \colon H \to \mathcal{I}(\mathcal{R}(H))$.

THEOREM 3 (Configuration Structures via CTS)**.** *Let N be a marked PT net. Then* $cs(N) = \mathcal{R}(ct(N))$.

Proof. The events of $cs(N)$, the actions of $ct(N)$ and, thus, the events of $\mathcal{R}(ct(N))$ are the transitions of N. The states S of the uniform CTS $ct(N)$ are exactly the configurations of $cs(N)$, and for each $s \in S$, we have $\varsigma_s = s$. This suffices, since a configuration structure is entirely determined by its set of configurations. $\square$

These results support our claim that configuration structures do not offer a faithful representation of net behaviours. In fact, $\mathcal{R}(\_)$ clearly collapses the structure excessively, as the natural transformation associated to the reflection map ρ can identify non homotopic runs (e.g., c' and c'' of Figure 5).

Concluding Remarks and Future Work

We have investigated the expressiveness of some 'collective-token' semantics for PT nets. In particular, to remedy the weakness of configuration structures, we have introduced *concurrent transition systems* — a version of higher dimensional transition system [3] more suited to the collective token philosophy, as they do not assign individual identities to multiple action occurrences in a multiset — and have shown that they can provide a faithful description of net behaviours.

$$
\begin{tikzcd}
\mathbf{Petri}_* \arrow[rr, "\mathcal{T}_*(\_)"] \arrow[rrrr, bend left, "\mathcal{CP}(\_)", "\simeq"'] \arrow[drr, "ct(\_)"] \arrow[ddr, "cs(\_)"'] \arrow[dr, phantom, "="] & & \mathbf{CMonCat}_* \arrow[rr, "u_0\downarrow\_"] \arrow[d, phantom, "\simeq"] & & \mathbf{Cat} \\
& & \mathbf{CTS} \arrow[urr, "\mathcal{C}(\_)"'] \arrow[dl, "\mathcal{R}(\_)"'] \arrow[dr, equal] \arrow[r, phantom, "="] & & \\
& \mathbf{CSCat} \arrow[rr, hook, "\mathcal{I}(\_)"'] \arrow[r, phantom, "\overset{\rho}{\Leftarrow}"] & & \mathbf{CTS} \arrow[uur, "\mathcal{C}(\_)"'] &
\end{tikzcd}
$$

FIGURE 6

The diagram of functors, equivalences and natural transformations in Figure 6 summarises the relationships between all these models. In the diagram, commutation on the nose (resp. natural equivalence) is represented by $=$ (resp. $\simeq$), and ρ denotes the unit of the reflection into the subcategory of configuration structures. The functor $\mathcal{CP}(\_)$ gives the category of Best-Devillers commutative processes. The functor $ct(\_)$ corresponds to the construction of the CTS for a given net, as defined in Section 1.3. The functor $\mathcal{C}(\_)$ yields the construction of the category of computations (i.e., homotopy equivalence classes of paths beginning in the initial state) of a CTS. The equivalence $\simeq$ between $\mathcal{C}(ct(\_))$ and $(u_{\mathrm{in}} \downarrow \mathcal{T}(\_))$ is shown in Section 2, providing the faithfulness of the construction. The functor $cs(\_)$ represents the abstraction from nets to configuration structure, defined in Section 1.2. Unfortunately, $\mathbf{CSCat}$ is a reflective subcategory of $\mathbf{CTS}$, as shown in Section 3 via the adjunction $\mathcal{R}(\_) \dashv \mathcal{I}(\_)$. The reflection functor $\mathcal{R}(\_)$ identifies too many things, so that the natural transformation associated to the reflection map ρ can identify non homotopic runs. Our running example shows that causality informations can get lost when using configuration structures, because homotopic paths are mapped into the same equivalence class.

The conceptual framework of this paper is summarised in Table 1, which makes explicit our research programme on the *behavioural*, *algebraic* and *logical* aspects of the two computational interpretations of PT nets, namely the *collective token* and the *individual token* philosophies, from the viewpoints of the structures suited to each of them and their mutual relationships.

The first row of Table 1 has been treated in this paper. As for the individual token interpretation, obvious candidates for suitable behavioural structures are *event structures*, *concatenable pomsets* and, especially, various kinds of *concatenable processes* [5, 20].

	Structures		
Computation Model	**Behavioural**	**Algebraic**	**Logical**
Nets and Collective Token Philosophy	Conf. structures, CTS, Commutative processes	$\mathcal{T}(N)$	$\mathtt{CAT}{\otimes}\mathtt{CMON}$
Nets and Individual Token Philosophy	Conc. Pomsets, Event Struct., Processes	$\mathcal{P}(N), \mathcal{Q}(N)$ $\mathcal{Z}(N)$?	$\mathtt{CAT} \otimes \mathtt{MON}$ $+ \mathtt{SYM}$

TABLE 1

From the logical viewpoint, it is not difficult to formulate a theory $\mathtt{SYM}$ of permutations and symmetries (cf. [19]) bridging the gap from strictly symmetric categories to categories symmetric only up to coherent isomorphism. On the other hand, the investigation of suitable algebraic models is still open, as our current best candidates, the symmetric strict monoidal categories $\mathcal{P}(N)$ of *concatenable processes* [5] and $\mathcal{Q}(N)$ of *strongly concatenable processes* [20], are both somehow unsatisfactory: $\mathcal{P}(\_)$ is a *non*-functorial construction, a drawback that inhibits many of the applications we have in mind, whilst $\mathcal{Q}(\_)$ solves the problem at the price of complicating the construction and relying on a non commutative monoid of objects.

We are currently searching for a better categorical construction, say $\mathcal{Z}(N)$, based on a suitable notion of *pre-net* that may subsume and underly the theory of PT nets and allow us to complete our programme.

Also, the complete analysis and comparison of bisimulation related issues in the various models considered in the paper (as in [6] for configuration structures) deserve further work that we leave for a future paper.

References

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

[2] C. BROWN AND D. GURR (1990), A Categorical Linear Framework for Petri Nets, in *Proceedings of the 5th Symposium on Logics in Computer Science*, 208–218, IEEE Press.

[3] G.L. CATTANI AND V. SASSONE (1996), Higher Dimensional Transition Systems, in *Proceedings of the 11th Symposium on Logics in Computer Science*, 55–62, IEEE Press.

[4] M. CLAVEL, S. EKER, P. LINCOLN, AND J. MESEGUER (1996), Principles of Maude, in *Proceedings First Intl. Workshop on Rewriting Logic and its Applications*, J. Meseguer (Ed.), *Electronic Notes in Theoretical Computer Science* **4**, `http://www.elsevier.nl/locate/tcs`, Elsevier.

[5] P. DEGANO, J. MESEGUER, AND U. MONTANARI (1996), Axiomatizing the Algebra of Net Computations and Processes. *Acta Informatica* **33**(7), 641–667, Springer-Verlag.

[6] R.J. VAN GLABBEEK AND G.D. PLOTKIN (1995), Configuration Structures, in *Proceedings of the 10th Symposium on Logics in Computer Science*, 199–209, IEEE Press.

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

[8] P.W. HOOGERS, H.C.M. KLEIJN, AND P.S. THIAGARAJAN (1996), An Event Structure Semantics for General Petri Nets. *Theoretical Computer Science* **153**(1-2), 129–170, Elsevier.

[9] K. LODAYA, R. RAMANUJAM, AND P.S. THIAGARAJAN (1989), A Logic for Distributed Transition Systems, 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**, 508–522, Springer-Verlag.

[10] J. MESEGUER (1998), Membership Equational Logic as a Logical Framework for Equational Specification, in *Proceedings of the 12th WADT Workshop on Algebraic Development Techniques*, F. Parisi-Presicce (Ed.), *Lecture Notes in Computer Science* **1376**, 18–61, Springer-Verlag.

[11] J. MESEGUER AND U. MONTANARI (1990), Petri Nets are Monoids. *Information and Computation* **88**(2), 105–155, Academic Press.

[12] J. MESEGUER AND U. MONTANARI (1998), Mapping Tile Logic into Rewriting Logic. in *Proceedings of the 12th WADT Workshop on Algebraic Development Techniques*, F. Parisi-Presicce (Ed.), *Lecture Notes in Computer Science* **1376**, 62–91, Springer-Verlag.

[13] J. MESEGUER, U. MONTANARI, AND V. SASSONE (1996), Process versus Unfolding Semantics for Place/Transition Petri Nets. *Theoretical Computer Science* **153**(1-2), 171–210, Elsevier.

[14] 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.

[15] J. MESEGUER, U. MONTANARI, AND V. SASSONE (1997), Representation Theorems for Petri Nets, in *Foundations of Computer Science*, C. Freska *et al.* (Eds.), *Lecture Notes in Computer Science* **1337**, 239–249, Springer-Verlag.

[16] M. MUKUND (1992), Petri Nets and Step Transition Systems. *International Journal of Foundations of Computer Science*, **3**(4), 443–478, World Scientific.

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

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

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

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

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

[22] 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.

Appendix. Recovering the Algebraic Semantics of Nets via Theory Morphisms

In order to define the theory of strictly symmetric strict monoidal categories, we first recall the definition of the theory of categories from [12].

The poset of sorts of the **PMEqtl**-theory of categories is `Object` $\le$ `Arrow`. There are two unary operations `d(_)` and `c(_)`, for *domain* and *codomain*, and a binary composition operation `_;_` defined if and only if the codomain of the first argument is equal to the domain of the second argument. Functions with explicitly given domain and codomain are always *total*.

```
fth CAT is
 sorts Object Arrow.
 subsort Object < Arrow.
 ops d(_) c(_) : Arrow -> Object.
 op  _;_.
 var a : Object.
 vars f g h : Arrow.
 eq  d(a) = a.
 eq  c(a) = a.
 ceq a;f = f if d(f) == a.
 ceq f;a = f if c(f) == a.
 cmb f;g : Arrow iff c(f) == d(g).
 ceq d(f;g) = d(f) if c(f) == d(g).
 ceq c(f;g) = c(g) if c(f) == d(g).
 ceq (f;g);h = f;(g;h) if c(f) == d(g) and c(g) == d(h).
endfth
```

The extension of the theory `CAT` to the theory of monoidal categories is almost effortless thanks to the tensor product construction of theories, which is informally defined as follows.

Let $T = (\Omega, \Gamma)$ and $T' = (\Omega', \Gamma')$ be theories in partial membership equational logic, with $\Omega = (S, \le, \Sigma)$ and $\Omega' = (S', \le', \Sigma')$. Their *tensor product* $T \otimes T'$ is the theory with signature $\Omega \otimes \Omega'$ having: poset of sorts $(S, \le) \times (S', \le')$, and signature $\Sigma \otimes \Sigma'$, with operators $f_l \in (\Sigma \otimes \Sigma')_n$ and $g_r \in (\Sigma \otimes \Sigma')_m$ for each $f \in \Sigma_n$ and $g \in \Sigma'_m$ (indices l and r stand respectively for `left` and `right` and witness whether the operator is inherited from the left or from the right component). The axioms of $T \otimes T'$ are the determined from those of T and T' as explained in [12].

The essential property of the tensor product of theories is expressed in the following theorem, where $\mathbf{PAlg}_T(\mathbf{C})$ indicates the category of T-algebras taken over the base category $\mathbf{C}$ rather than over $\mathbf{Set}$, the category of small sets and function.

THEOREM 4. *Let T, T' be theories in partial membership equational logic. Then, we have the following isomorphisms of categories:*

$$\mathbf{PAlg}_T(\mathbf{PAlg}_{T'}) \simeq \mathbf{PAlg}_{T \otimes T'} \simeq \mathbf{PAlg}_{T'}(\mathbf{PAlg}_T).$$

To define the theory of monoidal categories, we introduce a theory CMON of commutative monoids and apply the tensor product construction. Here we exploit the possibility given by Maude of declaring the associativity, commutativity and unit element as attributes of the monoidal operator.

```
fth CMON is
 sort Monoid.
 op 0 : -> Monoid.
 op _⊕_ : Monoid Monoid -> Monoid [assoc comm id: 0].
endfth
```

The theory of strictly symmetric strict monoidal categories is then defined as follows. Notice also the use of left and right corresponding to the indices l and r discussed above.

```
fth CMONCAT is CMON ⊗ CAT renamed by (
 sort (Monoid,Object) to Object.
 sort (Monoid,Arrow) to Arrow.
 op  0 left to 0.
 op _⊕_ left to _⊕_.
 op _;_ right to _;_.
 op d(_) right to d(_).
 op c(_) right to c(_).).
endfth
```

In order to define a theory in **PMEqtl** that represents PT Petri nets and their morphisms, we first introduce a theory whose models are automata whose states form a commutative monoid.

```
fth CMON-AUT is
 sorts State Transition.
 op 0 : -> State.
 op _⊗_ : State State -> State [assoc comm id: 0].
 ops origin(_) destination(_) : Transition -> State.
endfth
```

PROPOSITION 6. *The category* $\mathbf{Petri}$ *is a full subcategory of* $\mathbf{PAlg}_{\mathtt{CMON-AUT}}$.

Proof. It is immediate to check that each PT net is just a model of CMON-AUT whose states are the object of the commutative monoid freely generated by the set of places. □

Exploiting the modularity features of Maude, we can characterise $\mathbf{Petri}$ as a subcategory of $\mathbf{PAlg}_{\mathtt{CMON-AUT}}$. We import a functional module MSET[E :: TRIV] of multisets, parametrised on a functional theory of TRIV of elements, whose models are sets corresponding to the places of the net.

```
fth TRIV is sort Element.
endfth

fmod MSET[E :: TRIV] is
 sort MSet.
 subsort Element < MSet.
 op ∅ : -> MSet.
 op _+_ : MSet MSet -> MSet [assoc comm id: ∅].
endfm

fth PETRI[S :: TRIV] is
 protecting MSET[S] renamed by (sort MSet to Marking.).
 sort Transition.
 ops pre(_) post(_) : Transition -> Marking.
endfth
```

A theory morphism H from T to T', also called a *view* in Maude, is a mapping of the operators and sorts of T into T', preserving domain, codomain and subsorting, and such that the translation of the axioms of T are entailed by those of T'. It originates a forgetful functor $\mathcal{U}_H : \mathbf{PAlg}_{T'} \to \mathbf{PAlg}_T$ that — for T and T' theories without freeness constraints, such as those required in `PETRI[S]` — admits a left adjoint $\mathcal{F}_H : \mathbf{PAlg}_T \to \mathbf{PAlg}_{T'}$ whose effect is to lift H to a free model construction in $\mathbf{PAlg}_{T'}$. The inclusion functor from $\mathbf{Petri}$ to $\mathbf{PAlg}_{\mathtt{CMON-AUT}}$ is induced as the forgetful functor of a theory morphism `I` specified as a `view` in Maude as follows.

```
view I from CMON-AUT to PETRI[S :: TRIV] is
 sort Marking to MSet.
 op origin(_) to pre(_).
 op destination(_) to post(_).
 op 0 to ∅.
 op _⊗_ to _+_.
endview
```

Finally, the algebraic semantics of PT nets under the collective token philosophy, i.e., the construction $\mathcal{T}(\_)$, can be easily recovered via a simple theory morphism specified in Maude-like notation as

```
view V from CMON-AUT to CMONCAT is
 sort State to Object.
 sort Transition to Arrow.
 op origin(_) to d(_).
 op destination(_) to c(_).
endview
```

As stated in Proposition 5, the construction $\mathcal{T}(\_) : \mathbf{Petri} \to \mathbf{CMonCat}$ is then the following functor composition.

$$\mathbf{Petri} \hookrightarrow \mathbf{PAlg}_{\mathtt{CMON-AUT}} \xrightarrow{\mathcal{F}_V} \mathbf{PAlg}_{\mathtt{CMONCAT}}$$

DIPARTIMENTO DI INFORMATICA, UNIVERSITÀ DI PISA, CORSO ITALIA 40,
I-56125 PISA, ITALIA
bruni@di.unipi.it

COMPUTER SCIENCE LABORATORY, SRI INTERNATIONAL,
MENLO PARK, CA, USA
meseguer@csl.sri.com

DIPARTIMENTO DI INFORMATICA, UNIVERSITÀ DI PISA, CORSO ITALIA 40,
I-56125 PISA, ITALIA
ugo@di.unipi.it

QUEEN MARY AND WESTFIELD COLLEGE, UNIVERSITY OF LONDON,
LONDON E1 4NS, UNITED KINGDOM,
vs@dcs.qmw.ac.uk
AND
BRICS, COMPUTER SCIENCE DEPT., UNIVERSITY OF AARHUS, NY MUNKEGADE 540
8000 AARHUS C, DENMARK
vs@brics.dk

Recent BRICS Report Series Publications

RS-98-17 **Roberto Bruni, José Meseguer, Ugo Montanari, and Vladimiro Sassone.** ***A Comparison of Petri Net Semantics under the Collective Token Philosophy*****. September 1998. 20 pp. To appear in** ***4th Asian Computing Science Conference*****, ASIAN '98 Proceedings, LNCS, 1998.**

RS-98-16 **Stephen Alstrup, Thore Husfeldt, and Theis Rauhe.** ***Marked Ancestor Problems*****. September 1998.**

RS-98-15 **Jung-taek Kim, Kwangkeun Yi, and Olivier Danvy.** ***Assessing the Overhead of ML Exceptions by Selective CPS Transformation*****. September 1998. 31 pp. To appear in the proceedings of the** ***1998 ACM SIGPLAN Workshop on ML*****, Baltimore, Maryland, September 26, 1998.**

RS-98-14 **Sandeep Sen.** ***The Hardness of Speeding-up Knapsack*****. August 1998. 6 pp.**

RS-98-13 **Olivier Danvy and Morten Rhiger.** ***Compiling Actions by Partial Evaluation, Revisited*****. June 1998. 25 pp.**

RS-98-12 **Olivier Danvy.** ***Functional Unparsing*****. May 1998. 7 pp. This report supersedes the earlier report BRICS RS-98-5. Extended version of an article to appear in** ***Journal of Functional Programming*****.**

RS-98-11 **Gudmund Skovbjerg Frandsen, Johan P. Hansen, and Peter Bro Miltersen.** ***Lower Bounds for Dynamic Algebraic Problems*****. May 1998. 30 pp.**

RS-98-10 **Jakob Pagter and Theis Rauhe.** ***Optimal Time-Space Trade-Offs for Sorting*****. May 1998. 12 pp.**

RS-98-9 **Zhe Yang.** ***Encoding Types in ML-like Languages (Preliminary Version)*****. April 1998. 32 pp.**

RS-98-8 **P. S. Thiagarajan and Jesper G. Henriksen.** ***Distributed Versions of Linear Time Temporal Logic: A Trace Perspective*****. April 1998. 49 pp. To appear in** ***3rd Advanced Course on Petri Nets*****, ACPN '96 Proceedings, LNCS, 1998.**

RS-98-7 **Stephen Alstrup, Thore Husfeldt, and Theis Rauhe.** ***Marked Ancestor Problems (Preliminary Version)*****. April 1998. 36 pp.**