Zero-Safe Nets: Comparing the Collective and Individual Token Approaches*

Roberto BRUNI and Ugo MONTANARI
Dipartimento di Informatica
Università di Pisa, Italy

*Research supported by Office of Naval Research Contracts N00014-95-C-0225 and N00014-96-C-0114, 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 Architechture" sponsored by NEDO (New Energy and Industrial Technology Development Organization). Also research supported in part by U.S. 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*. Research carried out in part while the second author was on leave at Computer Science Laboratory, SRI International, Menlo Park, USA, and visiting scholar at Stanford University.

Proposed running head:

Zero-Safe Nets

Corresponding author:

Roberto Bruni
Dipartimento di Informatica
Università di Pisa
Corso Italia, 40
I-56125 Pisa, Italy
Phone: +39 50 887268
Fax: +39 50 887226
bruni@di.unipi.it

Abstract

The main feature of *zero-safe nets* is a primitive notion of *transition synchronization*. To this aim, besides ordinary places, called *stable* places, zero-safe nets are equipped with *zero* places, which in an observable marking cannot contain any token. This yields the notion of *transaction*: a basic atomic computation, which may use zero tokens as triggers, but defines an evolution between observable markings only. The abstract counterpart of a generic zero-safe net B consists of an ordinary P/T net whose places are the stable places of B, and whose transitions represent the transactions of B. The two nets offer both the refined and the abstract model of the same system, where the former can be much smaller than the latter, because of the transition synchronization mechanism.

Depending on the chosen approach – *collective* vs *individual token philosophy* – two notions of transaction may be defined, each leading to different operational and abstract models. Their comparison is fully discussed on the basis of a *multicasting system* example.

In the second part of the paper, we make use of category theory to analyze and motivate our framework. More precisely, the two operational semantics of zero-safe nets are characterized as adjunctions, and the derivation of abstract P/T nets as coreflections.

List of Figures

Introduction

Petri nets (Reisig 1985) are unanimously considered as one of the most attractive models of concurrency. As a matter of fact, this model offers a basic concurrent framework that has often been used as a semantic foundation on which to interpret many concurrent languages (Winskel 1982; Van Glabbeek and Vaandrager 1987; Olderog 1987; Degano *et al.* 1988; Gorrieri and Montanari 1995; Best *et al.* 1992). However, though net transitions allow for *token* synchronization, the basic net model does not offer any synchronization mechanism among transitions, while this feature is essential to write modular and expressive programs. In fact, all the above translations involve complex constructions for the net defining the synchronized composition of two programs.

Zero-safe nets (also *ZS nets*), extend Petri nets along this direction, coming equipped with a simple but very general notion of transition synchronization as a built-in feature. ZS nets are based on the notion of *zero* places. Tokens which are produced in a zero place force the firing of transitions which are able to consume them. In addition to zero places, a distinguished set of *stable* places is also present. Stable markings (consisting only of stable tokens) define the observable states of the system, whilst non-stable markings (those involving zero tokens) are the hidden states of the refined model, in the sense that they are not observable at the abstract level. Therefore, any synchronized evolution of a ZS net starts at some observable marking, then evolves through hidden states and eventually terminates into a new observable state.

A *refined* ZS net and an *abstract* P/T net are supposed to model the same given system. The former specifies how every transition of the latter is actually achieved as a different coordinated collection of firings (called *transaction*) and the latter offers the synchronized view, which corresponds to the abstraction from the hidden mechanism that controls the firings through zero tokens. This should favour a uniform approach to concurrent languages translation. For instance, in the case of process algebras in the CCS (Milner 1989) style, the parallel composition of two nets modelling communicating processes involves the combinatorial analysis of all the admissible synchronizations, whereas if zero places were used to model communication channels, then the parallel composition of two nets should just merge the common channels.

We remark that, in our setting, the term transaction denotes a certain activity of the system that might be composed by many, possibly concurrent, coordinated subactivities. Moreover, we require that, at the right level of abstraction, where transactions are considered as atomic activities of the system, all the intermediate states (except the initial and the final states) visited by each transaction must be unobservable. Since the concurrent semantics of an operational model is usually defined by considering as equivalent all the computations where the same concurrent events are executed in different orders, it follows that we should identify those transactions which are equivalent from a concurrent viewpoint, in such a way that the actual order of execution of concurrent transitions in the refined net is invisible in the abstract net. Having this in mind, a real dichotomy runs on the distinction between *collective* and *individual token philosophies* noticed, e.g., in Van Glabbeek and Plotkin (1995).

The simplest approach relies on the collective token philosophy, CTph for short, according to which net semantics should not distinguish among different instances of the idealised resources (i.e., the 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. This school of thought identifies all those firing sequences, which are obtained by repeatedly permuting pairs of (adjacent) concurrently enabled firings. We call *abstract stable transactions* the resulting equivalence classes of basic evolutions of the refined net. As a major drawback, this approach disregards that operationally equivalent resources may have different origins and histories, and may, therefore, carry different *causality* information (e.g., selecting one instance of a resource rather than the other, may be as different as being or not being causally dependent on some previous event). Therefore, causal dependencies on zero tokens are lost. It follows that the class of computations captured by abstract nets may turn out to be too abstract for many applications.

An alternative approach is based on the *individual token* philosophy, ITph for short. According to the ITph, causal dependencies are a central aspect in the dynamic evolution of a net. As a consequence, only the transactions which refer to isomorphic Goltz-Reisig processes (Goltz and Reisig 1983) are identified, and we call *connected transactions* the induced equivalence classes. In this case, the actual order of execution of concurrent transitions in the refined net is invisible in the abstract net, but all the causal dependencies are preserved.

In many cases, the distinction between the CTph and the ITph arises more from an academic debate than from some real issue. On the contrary, in the case of ZS nets, we want to emphasize that the distinction between the two philosophies is motivated also from a pragmatic perspective. In particular, depending on the chosen approach, two different notions of transaction may be defined, each leading to different operational and abstract models.

We will use the ZS net MS pictured in Fig. 1 as our running example throughout this paper. Notice that we extend the standard graphical representation for nets (i.e., boxes stand for transitions, circles for places, dots for tokens, and oriented arcs for the flow relation) by using smaller circles to represent zero places. The ZS net MS will be useful to illustrate some applications of the definitions. In addition, it will serve as a basis to better illustrate and understand the differences between the CTph and the ITph approaches.

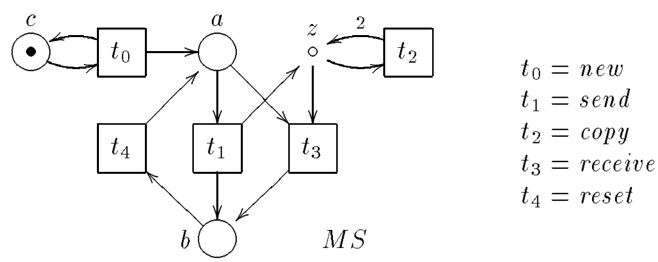

Figure 1: The ZS net MS representing a multicasting system.

The net MS is intended to represent a *multicasting system*. As in a broadcasting system, an agent can simultaneously send the same message to an unlimited number of receivers, but here the receivers are not necessarily all the remaining agents, and thus, several one-to-many communications can take place concurrently. Each token in place a represents a different *active* agent (i.e., an agent which is ready to communicate), while the tokens in place b are *inactive* agents. The zero place z models a buffer in which each token is a message. The transition *new* permits to create an unlimited number of agents. Each firing of *send* opens a one-to-many communication: a message is put in the buffer z and the agent which started the communication is suspended until the end of the current transaction. Each time the transition *copy* fires, a new copy of the same message is created. To complete a transaction, as many firings of transition *receive* are needed, as the number of copies created by *copy*, plus one. Each firing of the transition *receive* synchronizes an active agent with a copy of the message and then suspends the agent. At the end of a session, all the suspended agents are moved into place b. The transition *reset* activates an inactive agent.

We call *copy policy* any strategy for making copies of the messages in the buffer z. For instance, the *sequential one-to-n copying* policy acts as follows: once the message is produced, a firing of *copy* creates two copies of the message in the buffer, one of them is retrieved by the firing of *receive*, the remaining message is duplicated again by a second firing of *copy*, one of the two messages is retrieved by a second agent via a second firing of *receive*, and so on, until the n-th agent consumes the last copy of the message, i.e., the firing sequence starts with *send*, then repeats $n-1$ times *copy* followed by *receive*, and is concluded by the n-th *receive*. As another example, the *parallel one-to-n copying* policy begins with *send*, then creates all the nedeed copies of the message by firing *copy* with the maximal parallelism allowed (e.g., for $n = 8$, first *copy* fires, then two instances of *copy* fire concurrently, producing four copies of the message, then four instances of *copy* fire concurrently), and at last, the n agents retrieve the n copies concurrently (by firing n instances of *receive*).

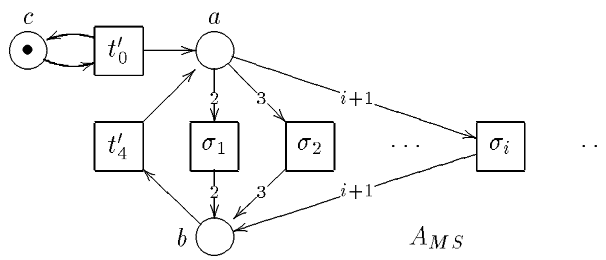

Figure 2: The abstract net for the multicasting system under the CTph.

As suggested before, the ZS net MS may be used as an example to better illustrate the difference between the CTph and the ITph. In the first case, transactions are distinguished only if they differ for the number of involved agents, whereas in the second case also different copy policies involving the same number of agents can be distinguished.

In Fig. 2 we see the infinite abstract P/T net A_{MS} for the refined ZS net MS, according to the CTph (see Def. 2.7). As it will be explained in Section 3.2.2, the *abstract* net A_{MS} comes equipped with a *refinement morphism* ϵ^C_{MS} to the refined net MS. The refinement morphism maps each place of A_{MS} into the homonymous stable place of MS and defines a bijection between the transitions of A_{MS} and (the equivalence classes under the CTph of) the transactions of MS. Therefore, the transition σ_n of A_{MS} represents a one-to-n transmission, i.e., it represents any transaction which corresponds to a one-to-n transmission. For instance, provided that we have reached a marking with at least five active agents, the sequential copying one-to-four transmission sc_4 and the parallel copying one-to-four transmission bc_4 defined by

$$\begin{aligned} sc_4 &= \textit{send-copy-receive-copy-receive-copy-receive-receive} \\ bc_4 &= \textit{send-copy-}\{\textit{copy,copy}\}\textit{-}\{\textit{receive,receive,receive,receive}\} \end{aligned}$$

(where we have used a self-explanatory informal notation to denote sequences of firings and concurrent firings), are in the same equivalence class. In fact it is possible to repeatedly apply some diamond transformation to transaction sc_4 to derive the transaction

$$\textit{send-copy-copy-copy-receive-receive-receive-receive},$$

as well as any other linearization of bc_4.

On the contrary, if we adopt the ITph, then different copy policies may be distinguished. The infinite *causal abstract* P/T net corresponding to the refined ZS net MS under the ITph (see Def. 2.20) is displayed in Fig. 3.

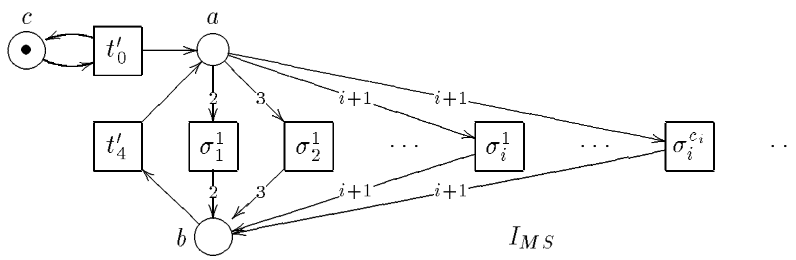

Figure 3: The *causal* abstract net for the multicasting system under the ITph.

As it will be explained in Section 3.3.3, the net I_{MS} comes equipped with a *causal refinement morphism* ϵ^I_{MS} to the refined net MS. Such morphism maps each places of I_{MS} into the homonymous stable place of MS, and defines a bijection between the transitions of I_{MS} and (the equivalence classes under the ITph of) the transaction of MS. We assume that the generic transition σ^k_n corresponds to the one-to-n transmission that follows the k-th codified copy policy (for each one-to-n transmission there are c_n different copy policies). As an example transitions σ^1_4 and σ^2_4 denote the equivalence classes of sc_4 and bc_4 that are no longer equivalent, and

Zero places can be used to coordinate and synchronize in a single transaction any number of transitions of the refined net. Thus it may well happen that the refined net is finite while the abstract net is infinite. This is the case, for instance, of our running example, which models a multicasting system where a message can be delivered to an unlimited number of receivers. Notice also that the abstract and the refined net both rely on the same basic token-pushing mechanism which is used to express their behaviours. This similarity is the key of the constructions described in the second part of the paper.

Though the multicasting system representation offers a good example about the expressivity of ZS nets (at least in our opinion), another interesting application presented in this paper concerns the compositional representation of a simple process algebra equipped with action prefix, parallel composition and restriction operators. In particular, we map each agent in a ZS net (with a sort of *interface*) such that its abstract net is able to (bi)simulate the behaviour of the agent. Moreover, the hiding of local names to the external world is handled very well, because agents that are equivalent up to α-conversion yield isomorphic nets.

The paper is organized as follows: after briefly recalling in Section 1 some basic definitions of net theory, we present ZS nets and their operational and abstract semantics throughout Section 2, where the simpler CTph approach is analyzed first.

The operational semantics for ZS nets in the CTph is defined in Section 2.1.1, while Section 2.1.2 concerns the related abstract semantics. In this context, the basic evolutions of ZS nets are equivalence classes of ordinary firing sequences induced by diamond transformation, and are called *abstract stable transactions*. To illustrate the ITph versions, in Section 2.2.1 we introduce the notion of *causal firing sequence*, which allows for a concise representation of concatenable processes, and has a suggestive implementation on a machine whose states are collections of token stacks. Then, we define the evolution of a ZS net in terms of the equivalence classes of causal firing sequences induced by isomorphic underlying processes. As for the CTph based approach, in Section 2.2.2, ordinary P/T nets are defined as the abstract counterparts of ZS nets. The translation of a simple CCS-like language concludes the first part of the paper.

The second part, consisting of Section 3, has a more mathematical flavour. In a rather precise sense, we give evidence that the definitions and the constructions presented in this paper are the best possible choices. This is achieved by employing some elementary category theory, which is informally summarized at the beginning of the section. Similar constructions apply to the CTph and ITph approaches. Both the operational semantics of ZS nets and the derivation of the abstract P/T nets are characterized as two universal constructions, following the *Petri nets are monoids* approach (Meseguer and Montanari 1990). More precisely, the former can be characterized as an *adjunction* and the latter as a *coreflection*. The universal properties of the two constructions witness that they are the "best" feasible choices, thus giving evidence of the naturality of the definitions presented in the first part of the paper. Moreover, since left adjoints preserve colimits, our semantics preserve several important net compositions (that can be expressed via colimit constructions), e.g., the union of two nets where a subset of places become shared.

Related approaches to the refinement and the abstraction of nets are discussed after the conclusions.

Preliminary versions of the CTph and ITph semantics of ZS nets has been separately presented in Bruni and Montanari (1997) and in Bruni and Montanari (1998), respectively, with some proofs either omitted or just sketched. Besides more detailed proofs, this paper contains in addition a comparison between the two approaches and an example concerning the translation of a simple CCS-like language.

1 Preliminaries

Definition 1.1 [Net] A *net* N is a triple $N = (S_N, T_N; F_N)$ where S_N is the (nonempty) set of *places* $a, a', \ldots$, T_N is the set of *transitions* $t, t', \ldots$ (with $S_N \cap T_N = \emptyset$), and $F_N \subseteq (S_N \times T_N) \cup (T_N \times S_N)$ is called the *flow relation*. $\square$

We will denote $S_N \cup T_N$ by N whenever no confusion arises, and subscripts will be omitted if they are obvious from the context. For $x \in N$, the set ${}^\bullet x = \{y \in N \mid yFx\}$ is called the *pre-set* of x, and the set $x^\bullet = \{y \in N \mid xFy\}$ is called the *post-set* of x. We only consider nets such that for any transition t, ${}^\bullet t \neq \emptyset$. Moreover, let ${}^\circ N = \{x \in N \mid {}^\bullet x = \emptyset\}$ and $N^\circ = \{x \in N \mid x^\bullet = \emptyset\}$ the sets of *initial* and *final elements* of N respectively. A place a is said to be *isolated* iff ${}^\bullet a \cup a^\bullet = \emptyset$.

Definition 1.2 [P/T net] A *marked place/transition net* is a 5-tuple $N = (S, T; F, W, u_{\text{in}})$ such that $(S, T; F)$ is a net, function $W : F \longrightarrow \mathbb{N}$ assigns a positive *weight* to each arc and the multiset $u_{\text{in}} : S \longrightarrow \mathbb{N}$ is the *initial marking* of N. □

We find it convenient to interpret the relation F as a function $F : ((S \times T) \cup (T \times S)) \longrightarrow \{0, 1\}$, assuming $xFy \iff F(x, y) \neq 0$. Then, for nets with weighted arrows we may safely replace $\{0, 1\}$ by $\mathbb{N}$. Thus, relation F becomes a *multiset* relation $F : (S \times T) \cup (T \times S) \longrightarrow \mathbb{N}$ and W is discharged.

A *marking* $u : S \longrightarrow \mathbb{N}$ is a finite multiset of places. It can be written either as $u = \{n_1 a_1, ..., n_k a_k\}$ where the natural number $n_i > 0$ (if $n_i = 0$ then the corresponding term $n_i a_i$ is safely omitted) dictates the number of occurrences (*tokens*) of the place a_i in u, i.e., $n_i = u(a_i)$, or as a formal sum $u = \bigoplus_{a_i \in S} n_i a_i$ denoting an element of the commutative monoid $S^\oplus$ freely generated from the set of places S (the order of summands is immaterial and the monoidal composition is defined by taking $(\bigoplus_i n_i a_i) \oplus (\bigoplus_i m_i a_i) = (\bigoplus_i (n_i + m_i) a_i)$ and 0 as the neutral element). For any transition $t \in T$, let $pre(t)$ and $post(t)$ be the multisets over S such that $pre(t)(a) = F(a, t)$ and $post(t)(a) = F(t, a)\ \forall a \in S$.

The evolution of a net (i.e., its interleaving behaviour) is usually described in terms of firing sequences. Given a P/T net N, let u and u' be markings of N. Then a transition $t \in T_N$ is *enabled* at u if $pre(t)(a) \leq u(a)$, $\forall a \in S_N$. Moreover, we say that u evolves to u' under the *firing* of t, written $u[t\rangle u'$, if and only if t is enabled at u and $u'(a) = u(a) - pre(t)(a) + post(t)(a)$, $\forall a \in S$. A *firing sequence* from u_0 to u_n is a sequence of markings and firings such that $u_0[t_1\rangle u_1 ... u_{n-1}[t_n\rangle u_n$. Given a marking u of N the set $[u\rangle$ of its *reachable markings* is the smallest set of markings such that $u \in [u\rangle$, and moreover, $\forall u' \in [u\rangle$ such that $u'[t\rangle u''$ for some transition t, then $u'' \in [u\rangle$.

Besides firings and firing sequences, *steps* and *steps sequences* are also usually introduced. A step allows for the simultaneous execution of several concurrently enabled transitions, i.e., the execution of a multiset of transitions. Formally, a multiset $X : T \longrightarrow \mathbb{N}$ is enabled at u if $\sum_{t \in T} X(t) \cdot pre(t)(a) \leq u(a)$, $\forall a \in S_N$. We say that u evolves to u' under the *step* X, written $u[X\rangle u'$, if and only if X is enabled at u and $u'(a) = u(a) + \sum_{t \in T} X(t) \cdot (post(t)(a) - pre(t)(a))$, $\forall a \in S$.

We conclude this introductory section by recalling the notion of *safety*, which plays an important role in Net Theory. A place is *n-safe* if it contains at most n tokens in any reachable marking. A net is n-safe if all its places are n-safe. A net is *safe* if a bound can be given for the number of tokens in each place (for all reachable markings), i.e., $\exists n \in \mathbb{N}$ such that $\forall u \in [u_{\text{in}}\rangle$, $\forall a \in S$, $u(a) \leq n$.

2 Zero Safe Computations

We augment P/T nets with special places called zero places. Their role is to coordinate the atomic execution of complex collections of transitions. From an abstract viewpoint, all the coordinated transitions will appear as being synchronized. However no new interaction mechanism is needed, and the coordination of the transitions participating in a step is handled by the ordinary token-pushing rules of nets. Notice that, whereas in the standard terminology, a n-safe net is a net whose places are all n safe, in the case of ZS nets only a subset of places (the zero places) are required to satisfy a 0-safe condition w.r.t. reachable markings.

Definition 2.1 [ZS net] A *zero-safe net* (also *ZS net* for short) is a 5-tuple $B = (S_B, T_B; F_B, u_B; Z_B)$ where $N_B = (S_B, T_B; F_B, u_B)$ is the *underlying* P/T net and the set $Z_B \subseteq S_B$ is the set of *zero places*. The places in $S_B \setminus Z_B$ are called *stable places*. A *stable marking* is a multiset of stable places. $\square$

Stable markings describe *observable* states of the system, whereas the presence of one or more zero tokens in a given marking marks it as internal and hence unobservable. We call *stable tokens* and *zero tokens* the tokens that respectively belong to stable places and to zero places.

2.1 Collective Token Approach

2.1.1 Operational Semantics

A *stable step* of a ZS net B may involve the execution of several transitions of the underlying P/T net N_B (it is actually a firing sequence of N_B). At the beginning, the state must contain enough stable (i.e., nonzero) tokens to enable all these transitions independently. As the computation progresses, the firings can only consume the stable tokens that were also available at the beginning of the computation and the zero tokens that have been produced by some fired transition. The stable tokens that are produced by some firings cannot be consumed in the same transaction. However, no token can be left on zero places at the end of the computation (nor can belong to the starting configuration). A stable step whose intermediate markings are all not stable and which consumes all the available stable tokens is called *stable transaction*.

In a certain sense, each stable step can be thought of as a collection of transactions plus a collection of idle resources; this means that once the basic transactions are known, then all the *correct* behaviours of the system may be derived. We ask the reader to keep in mind this observation because it constitutes the basis for our approach. *Stable step sequences* are sequences of stable steps.

Definition 2.2 [Stable step, transaction and step sequence] Let B be a ZS net and let $s = u_0[t_1\rangle u_1 \ldots u_{n-1}[t_n\rangle u_n$ be a firing sequence of the underlying net N_B of B. Sequence s is a *stable step* of B if:

- $\forall a \in S_B \setminus Z_B,\ \sum_{i=1}^{n} pre(t_i)(a) \leq u_0(a)$ (concurrent enabling);
- u_0 and u_n are stable markings of B (stable fairness).

We write $u_0\{\![s\rangle u_n$ to denote the stable step s, and $O(s)$ and $D(s)$ to denote the markings u_0 and u_n respectively. A stable step s is a *stable transaction* of B if in addition:

- markings $u_1, \ldots, u_{n-1}$ are not stable (atomicity);
- $\forall a \in S_B \setminus Z_B,\ \sum_{i=1}^{n} pre(t_i)(a) = u_0(a)$ (perfect enabling).

A *stable step sequence* is a sequence $u_0\{\![s_1\rangle u_1 \ldots u_{n-1}\{\![s_n\rangle u_n$. We also say that u_n is *reachable from* u_0 and we write $u_n \in \{\![u_0\rangle$. Sometimes we will denote the set $\{\![u_B\rangle$ of *reachable (stable) markings* of B by $\{\![B\rangle$. $\square$

In a stable transaction, each transition represents a micro-step carrying on the atomic evolution through invisible states. Stable tokens produced during the transaction become operative in the system only at the end of the transaction (i.e., after the firing of the *commit* transition t_n).

Example 2.3 Consider the ZS net MS of Fig. 1.

The firing sequence

$$\{2a\}[t_1\rangle\{a, b, z\}[t_4\rangle\{2a, z\}[t_3\rangle\{a, b\}$$

is not a stable step since the concurrent enabling condition is not satisfied $(pre(t_1)(b) + pre(t_4)(b) + pre(t_3)(b) = 1 \not\leq 0)$.

The firing sequence

$$\{4a\}[t_1\rangle\{3a, b, z\}[t_2\rangle\{3a, b, 2z\}[t_3\rangle\{2a, 2b, z\}[t_3\rangle\{a, 3b\}$$

is a stable step but not a stable transaction since the perfect enabling condition is not satisfied $(pre(t_1)(a) + pre(t_2)(a) + 2pre(t_3)(a) = 3 < 4)$.

The firing sequence

$$s' = \{2a, b\}[t_1\rangle\{a, 2b, z\}[t_3\rangle\{3b\}[t_4\rangle\{a, 2b\}$$

is a stable step but not a stable transaction since the atomicity constraint is not satisfied (the inner marking $\{3b\}$ is stable).

The firing sequence

$$s'' = \{2a, b\}[t_1\rangle\{a, 2b, z\}[t_4\rangle\{2a, b, z\}[t_3\rangle\{a, 2b\}$$

is a stable transaction (as opposed to the first sequence of this example that starts from the marking $2a$ mimicking s'', but it is not a stable transaction). □

The concurrent semantics of an operational model is usually defined by considering as equivalent all the computations where the same concurrent events are executed in different orders. In the case of P/T nets, the simplest approach relies on the collective token philosophy, which forces to identify all the firing sequences obtained by repeatedly permuting pairs of firings which are concurrently (i.e., independently) enabled. Thus, Def. 2.4 allows for a more satisfactory notion of stable step (transaction) in a concurrent setting.

An alternative approach, following the individual token philosophy, is presented in the Section 2.2.

Definition 2.4 [Diamond transformation and abstract sequence] Given a P/T net N, let

$$s = u_0[t_1\rangle u_1 \cdots u_{i-1}[t_i\rangle u_i[t_{i+1}\rangle u_{i+1} \cdots u_{n-1}[t_n\rangle u_n$$

be a firing sequence of N, where t_i and t_{i+1} are concurrently enabled by u_{i-1}, i.e., $pre(t_i)(a) + pre(t_{i+1})(a) \leq u_{i-1}(a)$ for any place a. Let s' be the firing sequence obtained by permuting the firing ordering of t_i and t_{i+1}, i.e.:

$$s' = u_0[t_1\rangle u_1 \cdots u_{i-1}[t_{i+1}\rangle u'_i[t_i\rangle u_{i+1} \cdots u_{n-1}[t_n\rangle u_n.$$

The sequence s' is a *diamond transformed* of s. The reflexive and transitive closure of the relation induced by diamond transformations gives the natural equivalence in the CTph interpretation. Notice that all the equivalent sequences have the same first and last markings u_0 and u_n. Equivalence classes are called *abstract sequences* and are denoted by σ. The abstract sequence of s is written $[\![s]\!]$. We also write $pre([\![s]\!]) = O(s)$ and $post([\![s]\!]) = D(s)$ to denote the origins and the destinations of $[\![s]\!]$, respectively. □

Definition 2.5 [Abstract stable step and transaction] Given a ZS net B, an *abstract stable step* is an abstract sequence $[\![s]\!]$ of the underlying net N_B, where s is a stable step. An *abstract stable transaction* is an abstract sequence of N_B which contains only stable transactions of B. We denote by Σ_B the set of all abstract stable transactions of B. □

As a matter of fact, the equivalence induced by diamond transformations preserves stable steps (because the diamond transformation preserves the properties of concurrent enabling and of stable fairness that are required by Def. 2.2) but not stable transactions. Generally speaking, the problem is that two stable transaction that are concurrently enabled could be interleaved in such a way that the resulting sequence also is a stable transaction. Of course, such transaction cannot be considered as a representative of an atomic activity of the system, because it can be expressed in terms of two sub-activities. Thus, it is not enough to require s to be a stable transaction to make sure that $[\![s]\!]$ is an abstract stable transaction, and we need a stronger constraint, namely that all the sequences in the equivalence class are stable transactions.

Example 2.6 As a counterexample showing that stable transactions are not preserved by our equivalence, consider the net MS defined in Fig. 1 and the stable steps s' and s'' defined in Ex. 2.3. It is easy to verify that $[\![s']\!] = [\![s'']\!]$, since s'' is obtained from s' by a diamond transformation. However s'' is a stable transaction whereas s' is not. Thus $[\![s'']\!]$ is not a stable transaction.

Conversely the firing sequence

$$s = \{4a\}[t_1\rangle\{3a, b, z\}[t_2\rangle\{\ldots\}[t_2\rangle\{\ldots\}[t_3\rangle\{\ldots\}[t_3\rangle\{a, 3b, z\}[t_3\rangle\{4b\}$$

defines an abstract stable transaction $[\![s]\!]$. In fact the stable transaction

$$\overline{s} = \{4a\}[t_1\rangle\{3a, b, z\}[t_2\rangle\{\ldots\}[t_3\rangle\{\ldots\}[t_2\rangle\{\ldots\}[t_3\rangle\{a, 3b, z\}[t_3\rangle\{4b\}$$

is the unique diamond transformed of s (and vice versa). □

2.1.2 Abstract Semantics

According to the CTph, since the basic execution steps of a system modelled via ZS nets consist of abstract stable transactions, it is natural to define a high-level description of such a model as a net whose transitions are abstract stable transactions.

Definition 2.7 [Abstract net] Let $B = (S_B, T_B; F_B, u_B; Z_B)$ be a ZS net. The net $A_B = (S_B \backslash Z_B, \Sigma_B; F, u_B)$, with $F(a, \sigma) = pre(\sigma)(a)$ and $F(\sigma, a) = post(\sigma)(a)$, is the *abstract net* of B (we recall that $pre(\sigma)$ and $post(\sigma)$ yield the first and last marking of any stable transaction in the equivalence class σ, and that Σ_B is the set of all the abstract stable transactions of B). □

Example 2.8 Let MS be the ZS net of our running example and let $\{c\}$ be its initial marking. Consider the following firing sequences of the underlying net N_{MS} of MS:

$$
\begin{aligned}
s_{new} &= \{c\}[t_0\rangle\{a,c\}, \\
s_{res} &= \{b\}[t_4\rangle\{a\}, \\
s_1 &= \{2a\}[t_1\rangle\{a,b,z\}[t_3\rangle\{2b\}, \\
&\dots \\
s_i &= \{(i+1)a\}[t_1\rangle\{ia,b,z\}[t_2\rangle\cdots[t_2\rangle\{ia,b,iz\}[t_3\rangle\cdots[t_3\rangle\{(i+1)b\}, \\
&\dots
\end{aligned}
$$

where s_i has $i-1$ firings of t_2 and i firings of t_3.

Thus $\Sigma_{MS} = \{t'_0, t'_4, \sigma_1, \dots, \sigma_i, \dots\}$ with $t'_0 = [\![s_{new}]\!]$, $t'_4 = [\![s_{res}]\!]$ and $\sigma_i = [\![s_i]\!]$, for $i \geq 1$. The (infinite) abstract net A_{MS} of MS is (partially!) pictured in Fig. 2. This abstract net consists of three places and infinitely many transitions: one for creating a new active process, one for reactivating a process after a synchronization, and one for each possible multicasting communication involving i receivers. □

2.2 Individual Token Approach

In this section, the basic activities of ZS nets are defined accordingly to the ITph, instead of the CTph. This choice has a great impact on the resulting notion of transaction, yielding dramatically different abstract models of ZS nets. To better understand the differences between the two philosophies, we propose the following example.

Example 2.9 Let MS be the net in Fig. 1, and suppose that the current marking is $\{a, b\}$. If t_4 fires then a new token is produced into place a. A firing of t_1 consumes a token from place a. In the ITph approach, it makes a difference if t_1 gets the token produced by t_4 or the one already present in a (in the former case the firing of t_1 causally depends on that of t_4 while in the latter case the firings of t_1 and of t_4 are concurrent activities). In the CTph approach the two firings are always concurrent, since the initial marking enables both t_1 and t_4, i.e., the execution of t_4 does not modify the enabling condition of t_1. Thus t_1 and t_4 may fire in any order always originating equivalent computations. □

2.2.1 Operational Semantics

In the ITph, a marking can be thought of as an indexed (over the places of the net) collection of ordered sequences of tokens. Moreover, the firing of a transition specifies which tokens (of each ordered sequence) are consumed and also the correspondence between each token in the reached marking with either some produced token or an idle token of the original marking. Using multisets instead of ordered sequences would make it impossible to recognize which token was produced by which firing, as it happens for the CTph.

The Stacks Based Approach. The approach we propose is very similar to the one adopted in Ristori (1994): we choose a canonical interpretation of the tokens that are to be consumed and produced in a firing and we introduce *permutation firings* with the task of rearranging the orderings of the indexed sequences of tokens. Therefore a marking is represented as a collection of stacks, one for each place, on which transitions can only access through a firing in order to extract and to insert tokens. Therefore, the extraction and the insertion of tokens in each place follow the LIFO policy. However, also permutation firings are allowed to modify the token positions in the state (i.e., the collection of stacks) of the system. Informally, permutation firings permit to choose which tokens to consume next. In what follows, we will denote the token stack associated to a certain place a by the term a-stack.

Definition 2.10 [Causal firing] Let N be a P/T net, and $s = u[t\rangle u'$ be a firing of N for some marking u and transition t. We interpret firing s as a *causal firing* by assuming that s consumes the first $pre(t)(a)$ tokens of the a-stack of u and produces the first $post(t)(a)$ tokens of the a-stack of u', for each place a. □

Definition 2.11 [Permutation firing] Let N be a P/T net. Given a marking $u = \{n_a a\}_{a\in S_N}$ of N, a *symmetry* p on u is a vector of *permutations* $p = \langle \pi_a \rangle_{a\in S_N}$ with $\pi_a \in \Pi(n_a)$, $\forall a \in S_N$, i.e., each π_a is a permutation of n_a elements. We denote by $\Pi(u)$ the set of all symmetries on u. Each symmetry p on u induces a *permutation firing* $s = u[p\rangle u$ on the net. □

Definition 2.12 [Causal firing sequence] Given a P/T net N, a *causal firing sequence* is a finite sequence $\omega = s_1 \cdots s_n$ of causal and permutation firings such that $s_i = u_{i-1}[X_i\rangle u_i$ with $X_i \in T_N \cup \Pi(u_{i-1})$ for $i = 1, \ldots, n$. We say that ω *starts at* u_0 (written $O(\omega) = u_0$) and *ends in* u_n (written $D(\omega) = u_n$). □

Example 2.13 Let N_{MS} be the underlying net of the ZS net MS defined in the introduction (i.e., in N_{MS} we do not distinguish between stable and zero places). A causal firing sequence for N_{MS} is

$$\omega = \{b, c\}[t_0\rangle\{a, b, c\}[t_4\rangle\{2a, c\}[t_1\rangle\{a, b, c, z\}[t_3\rangle\{2b, c\}.$$

At the beginning of the sequence ω, the stacks of places a and z are empty while the stacks of places b and c contain one token each. After the firing of t_0, the token in the c-stack is replaced by a new one and a token is also inserted in the a-stack. The firing of t_4 consumes the token in the b-stack and puts a new token on top of the a-stack. The firing of t_1 consumes the token on top of the a-stack (the one produced by the firing of t_4) and produces a token in the b-stack and a token in the z-stack. The firing of t_3 consumes the token in the z-stack and also the token produced by the firing of t_0 in the a-stack; then, it inserts a token on top of the b-stack. Since the sequence ω does not involve any symmetry, it follows that the latest tokens produced are the first to be consumed next (for any place). To represent the sequence in which t_1 depends on t_0 (and t_3 depends on t_4) we have two possibilities. The first one is to execute t_0 after t_4 (they are concurrently enabled), thus obtaining the sequence $\omega' = \{b,c\}[t_4\rangle\{a,c\}[t_0\rangle\{2a,c\}[t_1\rangle\{a,b,c,z\}[t_3\rangle\{2b,c\}$. The second possibility is to reorganize the a-stack just before executing t_1. This can be done via the symmetry $p = \langle\pi_a,\pi_b,\pi_c,\pi_z\rangle$, where π_b and π_z are the empty permutations in $\Pi(0)$, $\pi_a = \{1 \to 2, 2 \to 1\} \in \Pi(2)$, and π_c is the unique identity permutation in $\Pi(1)$. Indeed we have the sequence $\omega'' = \{b,c\}[t_0\rangle\{a,b,c\}[t_4\rangle\{2a,c\}[p\rangle\{2a,c\}[t_1\rangle\{a,b,c,z\}[t_3\rangle\{2b,c\}$. □

Review of Concatenable Processes. Causal firing sequences establish a correspondence among the tokens produced and consumed via firings. This is due to the implicit orders which are imposed on the markings and is strictly related to a *process* view of computations. *Concatenable processes* (Degano *et al.* 1989; Degano *et al.* 1996; Sassone 1996) are obtained from processes by imposing a total ordering on the origins that are instances of the same place and, similarly, on the destinations.

A net K is a *causal net* if $\forall a \in S_K,\ |{}^\bullet a| \leq 1 \wedge |a^\bullet| \leq 1$ and F_K^* is acyclic (F^* denotes the reflexive and transitive closure of relation F), i.e., $\forall x,y \in K,\ xF_K^*y \wedge yF_K^*x \implies x = y$. A (Goltz-Reisig) *process* for a P/T net N is a mapping $P : K \longrightarrow N$, from a causal net K to N, such that $P(S_K) \subseteq S_N$, $P(T_K) \subseteq T_N$, ${}^\circ K \subseteq S_K$, and $\forall t \in T_K$, $\forall a \in S_N$, $F_N(a, P(t)) = \left|P^{-1}(a) \cap {}^\bullet t\right| \wedge F_N(P(t), a) = \left|P^{-1}(a) \cap t^\bullet\right|$. As usual we denote the set of *origins* (i.e., minimal or initial places) and *destinations* (i.e., final or maximal places) by $O(K) = {}^\circ K$ and $D(K) = K^\circ \cap S_K$, resp. Two processes P and P' of N are *isomorphic* and thus identified if there exists an isomorphism $\psi : K_P \longrightarrow K_{P'}$ such that $P' \circ \psi = P$.

Given a set S with a labelling function $l : S \longrightarrow S'$, a *label-indexed ordering function* for l is a family $\beta = \{\beta_a\}_{a \in S'}$ of bijections, where $\beta_a : l^{-1}(a) \longrightarrow \{1, \ldots, \left|l^{-1}(a)\right|\}$. A *concatenable process* for a P/T net N is a triple $C = (P, {}^\circ\ell, \ell^\circ)$ where $P : K \longrightarrow N$ is a process for N and ${}^\circ\ell$, ℓ° are label-indexed ordering functions for the labelling function P (restricted to $O(K)$ and $D(K)$, resp.). Two concatenable processes C and C' are isomorphic if P_C and $P_{C'}$ are isomorphic via a morphism that preserves all the orderings.

A partial binary operation $\_;\_$ (associative up to iso and with identities) of concatenation of concatenable processes (whence their names) can be easily defined: we take as source (target) the image through P of the initial (maximal) places of K_P; then the composition of $C = (P, {}^\circ\ell, \ell^\circ)$ and $C' = (P', {}^\circ\ell', \ell'^\circ)$ is realized by merging, when it is possible, the maximal places of K_P with the initial places of $K_{P'}$ according to their labelling and ordering functions so to match those places one-to-one.

Concatenable processes admit also a monoidal *parallel* composition $\_\otimes\_$, which can be represented by putting two processes side by side. Due to space limitation, we refer the interested reader to Degano *et al.* (1996) for the formal definitions.

From Causal Sequences to Processes. It may be easily noticed that each causal firing sequence uniquely determines a concatenable process. Informally the construction associates an elementary (concatenable) process to each causal and permutation firing.

From causal firings to processes. Let N be a P/T net and $s = u[t\rangle u'$ be a causal firing, with $u = \{n_a a\}_{a \in S_N}$, $pre(t) = \{h_a a\}_{a \in S_N}$, $post(t) = \{k_a a\}_{a \in S_N}$. The associated concatenable process $pr(s) = (P : K \longrightarrow N, {}^{\circ}\ell, \ell^{\circ})$ is defined as follows:

- $T_K = \{\tilde{t}\},\ P(\tilde{t}) = t$;
- $S_K = \{\tilde{a}_i \mid a \in S_N,\ 1 \leq i \leq n_a + k_a\},\ P(\tilde{a}_i) = a$;
- ${}^{\bullet}\tilde{t} = \{\tilde{a}_i \mid a \in S_N,\ 1 \leq i \leq h_a\}$, $\tilde{t}^{\bullet} = \{\tilde{a}_i \mid a \in S_N,\ h_a + 1 \leq i \leq h_a + k_a\}$. Therefore
  - $O(K) = \{\tilde{a}_i \in S_K \mid i \leq h_a \vee i \geq h_a + k_a + 1\}$ and
  - $D(K) = \{\tilde{a}_i \in S_K \mid i \geq h_a + 1\}$;
- $\forall \tilde{a}_i \in O(K),\ {}^{\circ}\ell_a(\tilde{a}_i) = \begin{cases} i & \text{if } 1 \leq i \leq h_a \\ i - k_a & \text{if } h_a + 1 + k_a \leq i \leq n_a + k_a \end{cases}$;
- $\forall \tilde{a}_i \in D(K),\ \ell^{\circ}_a(\tilde{a}_i) = i - h_a$.

A brief explanation is necessary. The causal net K contains a unique transition $\tilde{t}$ (which is mapped in the transition t of N), and a place for each consumed, produced, and idle token of s. For each place $a \in S_N$ we need exactly $n_a + k_a$ different places in S_K. We denote the generic i-th place associated to place a by $\tilde{a}_i$. The set $\{\tilde{a}_i \mid a \in S_N,\ 1 \leq i \leq h_a\}$ represents the tokens which are consumed by the causal firing of t, i.e., the first h_a tokens of each a-stack in the starting state. The set $\{\tilde{a}_i \mid a \in S_N,\ h_a + 1 \leq i \leq h_a + k_a\}$ represents the tokens which are produced by the causal firing of t, i.e., the first k_a tokens of each a-stack in the ending state. The set $\{\tilde{a}_i \mid a \in S_N,\ h_a + k_a + 1 \leq i \leq n_a + k_a\}$ contains the remaining idle tokens, i.e., the last $n_a - h_a$ token of each a-stack of both u and u'. Functions ${}^{\circ}\ell$ and ℓ° are defined accordingly with these assumptions.

From permutation firings to processes. Let N be a P/T net and $s = u[p\rangle u'$ be a permutation firing, with $u = \{n_a a\}_{a \in S_N}$ and $p = \langle \pi_a \rangle_{a \in S_N}$. The associated concatenable process $pr(s) = (P : K \longrightarrow N, {}^{\circ}\ell, \ell^{\circ})$ is defined as follows:

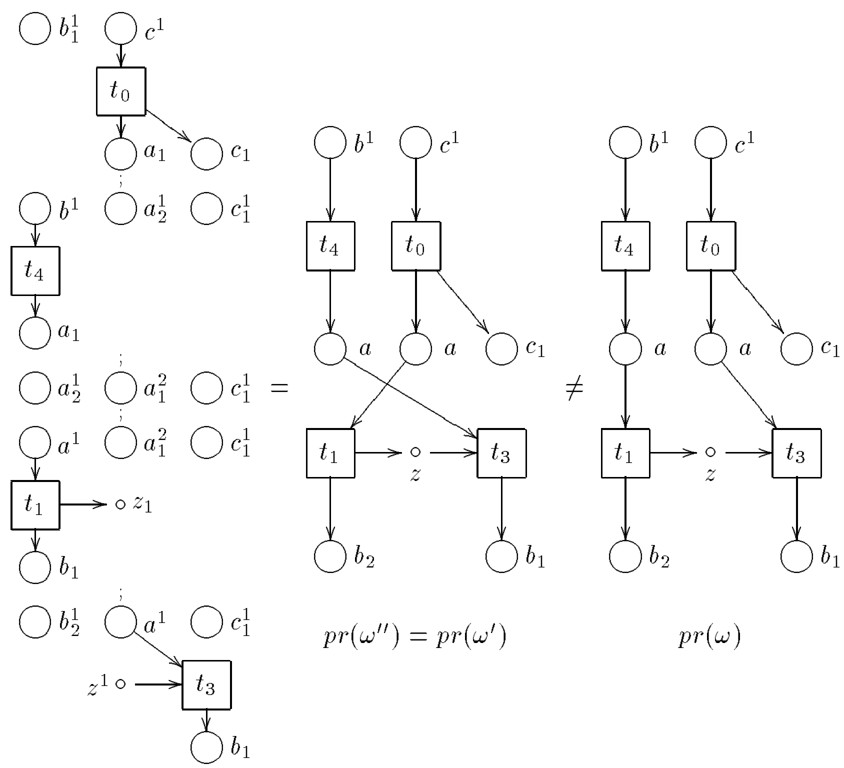

Figure 4: The concatenable processes derived from sequences ω'', ω', and ω of Ex. 2.13.

- $T_K = \emptyset$ (it follows that $O(K) = D(K) = S_K$);
- $S_K = \{\tilde{a}_i \mid a \in S_N,\ 1 \leq i \leq n_a\},\ P(\tilde{a}_i) = a$;
- $\forall \tilde{a}_i \in O(K),\ \ {}^{\circ}\ell_a(\tilde{a}_i) = i$;
- $\forall \tilde{a}_i \in D(K),\ \ell_a^{\circ}(\tilde{a}_i) = \pi_a(i)$.

In this case the set of transitions is empty and all the tokens stay idle. The generic place $\tilde{a}_i$ of k denotes the instance of place a which corresponds to the i-th token (from the top) of the a-stack of the starting state. The re-organization induced by the permutation firing is realized differentiating the functions ${}^{\circ}\ell$ and ℓ°.

The concatenable process associated to a (finite) causal firing sequence is given by the concatenation of the concatenable processes associated to each step of the given sequence. In what follows we denote by $pr(\omega)$ the concatenable process associated to the causal firing sequence ω (up to iso).

Example 2.14 The concatenable processes derived from the sequences ω, ω' and ω'' of Ex. 2.13 are presented in Fig. 4 (we use the standard notation that labels the places and transitions of the causal net K with their images in N; a superscript for any initial place and a subscript for any final place denotes the value of ${}^{\circ}\ell$ and ℓ°, resp.), the construction of $pr(\omega'')$ being explained in details. $\square$

Terminology. Let us introduce some properties of processes that will be used extensively. A process is *active* if it includes at least one transition, *inactive* otherwise. An active process is *decomposable into parallel activities* if it is the parallel composition of two (or more) active processes. If such a decomposition does not exist, then the process is called *connected*. A connected process may involve idle places, but it does not admit disjoint activities. The resources which are first produced and then consumed (i.e., the inner places) are called *evolution places*. More formally, a concatenable process $C = (P : K \longrightarrow N, {}^{\circ}\ell, \ell^{\circ})$ is *connected* if and only if the set of transitions of K is non-empty and moreover, for each pair (t, t') of transitions of K there exists an undirected path (through the arcs of the flow relation) connecting t and t'. Process C is *full* if it does not contain idle (i.e., isolated) places (i.e., $\forall a \in S_K,\ |{}^{\bullet}a| + |a^{\bullet}| \geq 1$). Finally, the set of *evolution places* of process C is the set $E_C = \{P(a) \mid a \in K,\ |{}^{\bullet}a| = |a^{\bullet}| = 1\}$.

Connected Transactions. A causal firing sequence is essentially a linearization of a (concatenable) process and more than one sequence may correspond to the same (up to iso) concatenable process. Indeed, sequences differing in the order in which concurrent firings are executed or for the way equivalent symmetries are performed are identified. For example, repeatedly swapping two tokens (of the same place) for an odd number of times is equivalent to apply the swap just once. We will consider the equivalence over sequences induced by isomorphic processes. Moreover, we will notice that, for the kind of sequences under consideration, the label-indexed ordering functions of origins and destination are no longer important, so we define the equivalence on the underlying Goltz-Reisig processes, rather than on the concatenable processes.

Definition 2.15 [Equivalent causal firings] Let N be a net and ω, ω' be two causal firing sequences. We say that ω and ω' are *causally equivalent*, written $\omega \approx \omega'$ iff $pr(\omega) = (P : K \longrightarrow N, {}^{\circ}\ell, \ell^{\circ})$ and $pr(\omega') = (P' : K' \longrightarrow N, {}^{\circ}\ell', \ell'^{\circ})$ with process P isomorphic to P'. The equivalence class of ω is denoted by $[\![\omega]\!]_{\approx}$. We use ξ to range over equivalence classes. Since relation $\approx$ respects the initial and final marking, we extend the notation letting $O(\xi) = O(\omega)$ and $D(\xi) = D(\omega)$, for $\xi = [\![\omega]\!]_{\approx}$. $\square$

In the ITph based approach, state changes are given in terms of *connected steps*, which may involve the concurrent execution and synchronization of several transitions.

Definition 2.16 [Connected step and transaction] Given a ZS net B, let $\omega = s_1 \cdots s_n$ be a causal firing sequence of the underlying P/T net N_B. The equivalence class $\xi = [\![\omega]\!]_{\approx}$ is a *connected step* of B, written $O(\xi)[\![\xi\rangle D(\xi)$, if:

- $O(\omega)$ and $D(\omega)$ are stable markings (stable fairness);
- $E_{pr(\omega)} \subseteq Z_B$ (atomicity).

A *connected step sequence* is a sequence $u_0[\![\xi_1\rangle u_1 \ldots u_{n-1}[\![\xi_n\rangle u_n$ and we say that u_n is reachable from u_0. Furthermore, the connected step ξ is a *connected transaction* of B if:

- $pr(\omega)$ is connected;
- $pr(\omega)$ is full.

We denote by Ξ_B (ranged by δ) the set of connected transactions of B. □

As for the CTph approach, a connected step may be applied if the starting state contains enough stable tokens to enable all the transitions independently, and no token may be left on zero places at the end of the step (nor may be found on them at the beginning of the step). This means that all the zero tokens which are produced are also consumed in the same step, and defines the synchronization mechanism. A *connected transaction* is a connected step such that no intermediate marking is stable, and which consumes all the available stable tokens of the starting state. Connected steps differ from stable steps in that they allow for a finer causal relationship among firings. Let us point out that all conditions in Def. 2.16 impose constraints only over the Goltz-Reisig process associated with $pr(\omega)$, thus all the conditions are preserved by the equivalence $\approx$.

Proposition 2.17 *Conditions "stable fairness", "atomicity", "connectedness", and "fullness" (see Def. 2.16) are preserved by the equivalence* $\approx$.

We claim that, in the ITph context, connected transactions are a good definition for the basic behaviours of the systems. Our assertion is supported by the fact that connected transactions denote *atomic computations that cannot be extended further*. Atomicity follows immediately from the connectedness of the associated processes (e.g., a transaction cannot be decomposed in two disjoint activities, because they must interact by definition of connectedness). The second argument deserves a more precise explanation. An atomic behaviour can be extended if there exists a broader atomic behaviour of which the former is a sub-part. From our viewpoint, the only interaction allowed in a transaction of a ZS net is given by the flow of tokens through zero places. Since connected steps and connected transactions start and also end in stable markings, it is impossible to hook them in a wider atomic computation by means of zero tokens. This is very clear for connected transactions, because they consume all the needed resources. This is not the case of connected steps, since it could be possible for some resource to stay idle during the whole sequence of moves. However this kind of resources are stable and not connected to the rest of the step, thus, any other activity involving them is intrinsically concurrent w.r.t. the step under consideration. It follows that any wider behaviour extending a connected step is not atomic (e.g., it can be expressed in terms of concurrent components).

Example 2.18 Let us consider the ZS net MS of Fig. 1.

The equivalence class of the causal firing sequence

$$\{a\}[t_1\rangle\{b,z\}[t_4\rangle\{a,z\}[t_3\rangle\{b\}$$

is not a connected step since the "atomicity" requirements is not fullfilled. More precisely, this sequence defines a communication between an agent and itself, which is forbidden.

The equivalence class of the causal firing sequence

$$\{4a\}[t_1\rangle\{3a,b,z\}[t_2\rangle\{3a,b,2z\}[t_3\rangle\{2a,2b,z\}[t_3\rangle\{a,3b\}$$

is a connected step but not a connected transaction since the associated process is not full.

The equivalence class of the causal firing sequence

$$\{2a,c\}[t_1\rangle\{a,b,c,z\}[t_3\rangle\{2b,c\}[t_0\rangle\{a,2b,c\}$$

is a connected step but not a connected transaction since the associated process is not connected.

The equivalence class of the causal firing sequence

$$\{5a\}[t_1\rangle\{\cdots\}[t_2\rangle\{\cdots\}[t_2\rangle\{\cdots\}[t_2\rangle\{\cdots\}[t_3\rangle\{\cdots\}[t_3\rangle\{\cdots\}[t_3\rangle\{\cdots\}[t_3\rangle\{5b\}$$

(where the obvious intermediate markings have been omitted), is a connected transaction. □

2.2.2 Abstract Semantics

Also in the ITph based approach it comes natural to define an abstract view of the systems modelled via zero safe nets. As for the CTph, since transactions rewrite multisets of stable tokens, it is natural to choose a net as a candidate for the abstraction. Furthermore, since the ordering of tokens in the pre-set (post-set) of a transition is useless we should abstract from it. This is already done because we consider equivalence classes of causal firing sequences which denote different Goltz-Reisig processes.

When restricted to connected steps, this equivalence intuitively corresponds to limit the symmetries of permutation firings to be vectors of permutations over the zero places only, with the assumption that the stable tokens which are produced in a transaction are not reused during the same transaction. The last statement was also the basis for the CTph approach.

Example 2.19 Consider the ZS net MS defined in the introduction. Let $s = \{2a\}[p\rangle\{2a\}$ where $p \in \Pi(2a)$ is the symmetry which swaps the two tokens in a, $s' = \{2b\}[p'\rangle\{2b\}$ where $p' \in \Pi(2b)$ is the symmetry which swaps the two tokens in b and $\omega = \{2a\}[t_1\rangle\{a, b, z\}[t_3\rangle\{2b\}$. Clearly the causal sequences ω, $s\omega$, and $\omega s'$ define the same connected transaction $\xi = [\![\omega]\!]_{\approx}$, but $pr(s\omega) \neq pr(\omega) \neq pr(\omega s')$. If we represent the connected transaction ξ as a transition t of a net, then its preset (as well as its postset) is an unordered multiset. This means that when t fires it is impossible to distinguish among the tokens in a and in b that it produces and consumes. It follows that it makes no sense to have many different transitions to represent behaviours that we cannot reproduce at the abstract level. Thus we are forced to identify $pr(s\omega)$, $pr(\omega)$, $pr(\omega s')$, and also $pr(s\omega s')$. $\square$

Definition 2.20 [Causal abstract net] Let $B = (S_B, T_B; F_B, u_B; Z_B)$ be a ZS net. The net $I_B = (S_B \setminus Z_B, \Xi_B; F, u_B)$, with $F(a, \delta) = pre(\delta)(a)$ and $F(\delta, a) = post(\delta)(a)$, is the *causal abstract net* of B (we recall that $pre(\delta)$ and $post(\delta)$ denote the multisets $O(\delta)$ and $D(\delta)$, respectively, and that Ξ_B is the set of all the connected transactions of B). $\square$

We conclude this section by examining the causal abstract net of the multicasting system defined by the ZS net MS in the introduction.

Example 2.21 Let MS be the zero safe net defined in the introduction. Its causal abstract net I_{MS} is (partially!) depicted in Fig. 3. We now comment on the possible connected transactions. Transition t'_0 is the basic activity which creates a new communicating process and it corresponds to $[\![\omega_0]\!]_{\approx}$ with $\omega_0 = \{c\}[t_0\rangle\{a, c\}$. Similarly t'_4 is the equivalence class of the the firing of t_4 in the marking $\{b\}$. Each σ_i^k describes a different one-to-i communication. The index k identifies the copy policy (see the Introduction) under consideration. Essentially, a generic one-to-i communication can be described as follows: a firing of t_1 initiates the communication, then the system executes as many firings of t_2 as the number of copies of the message needed (i.e., $i-1$ since a message is already present in the buffer), and finally i firings of t_3 synchronize the remaining i messages with i active processes. In the ITph we distinguish among tokens in a same marking, which were created by different firings. In this way we have a one-to-one correspondence among copy policies and the *complete* binary trees with exactly i leaves. We recall that a binary tree is *complete* if any internal node has exactly two children (we do not distinguish between *left* and *right* children). For any i, the total number of copy policies can be defined recursively.

- For $i = 1$ there is only one tree whose root is the unique leaf.
- If $i = 2h + 1$ for some positive integer h, it follows that one of the subtrees rooted in the children of the root is a complete binary tree with $j \leq h$ leaves, while the subtree rooted in the other child is a complete binary tree with $i - j$ leaves; for any j, we know that there are $c_j \cdot c_{i-j}$ possible trees made in this way, thus $c_i = \sum_{j=1}^{h} c_j \cdot c_{i-j}$.
- If $i = 2h$ for some positive integer h, we adopt an analogous reasoning to deduce that for any $j < h$ there are $c_j \cdot c_{i-j}$ possible complete binary trees such that exactly j leaves belongs to one of the subtrees rooted in the children of the root. However we have to pay attention to the case $j = h$. In fact, if the two subtrees have the same number h of leaves then there are $\frac{c_h \cdot (c_h+1)}{2}$ possible combinations for choosing them. It follows that $c_i = \frac{c_h \cdot (c_h+1)}{2} + \sum_{j=1}^{h-1} c_j \cdot c_{n-j}$.

Since there are no transitions in MS requiring 2 or more zero token, there are no other transactions (it is impossible to synchronize separate communications). $\square$

2.3 Concurrent Language Translation

Before presenting the categorical justifications in Section 3, we conclude the more descriptive part of the paper, offering some general hints for the translation of concurrent languages, equipped with a CCS-like communication mechanism, into ZS nets. The idea is to represent each channel by a pair of zero places, one for input and one for output, and to model each input (output) action on a channel with a transition, which produces a token on the input (output) zero place associated to the channel. A synchronization transition, also associated to the channel, is fired by a token in the input and a token in the output zero place. If the channel is restricted, this setting forces synchronization of input/output actions. If the channel is not restricted, two additional transitions are associated to the channel, simulating the effect of the environment.

More precisely, for every channel name a we define a ZS net Z_a consisting of two zero places $a!$ and $a?$, and three transitions in_a, syn_a, and out_a as depicted in Fig. 5.

We denote by $Z(\{a_1, ..., a_n\})$ the ZS net obtained as the disjoint union of $Z_{a_1}, \ldots, Z_{a_n}$, where $\{a_1, ..., a_n\}$ is any set of channel names.

Definition 2.22 [Interfaced net] A $\{a_1, ..., a_n\}$-*interfaced net* is a triple

$$\langle B, \{a_1, ..., a_n\}, P\rangle,$$

where B is a (marked) ZS net – in our translation the initial marking will always be a set – and P is an injective mapping from $Z(\{a_1, ..., a_n\})$ to B,

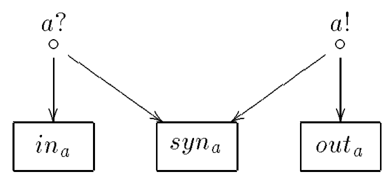

Figure 5: The ZS net Z_a.

which preserves the ZS net structure. The set of names $\{a_1, ..., a_n\}$ is also called the *interface* of the net. $\square$

Then, each agent p is modelled by an $fv(p)$-interfaced net $[\![p]\!]$, where the set $fv(p)$ is the set of the free (i.e., non-restricted) channel names in p. We consider a simple process algebra equipped with the operations of inaction nil, input and output action prefix $a.\_$ and $\bar{a}.\_$, parallel composition $\_|\_$, and restriction $\_\backslash a$. Other operations, as nondeterministic sum and recursion, could also be added, with limitations similar to those described in the literature for the existing approaches. The definition is given by initiality, and thus it is enough to define the corresponding operations on interfaced nets.

Inaction. The inactive net nil is a $\emptyset$-interfaced net $\langle B, \emptyset, \emptyset\rangle$, where B consists of a single place, and has one token in it as the initial marking.

Action prefix. The interfaced net $a.\langle B, A \cup \{a\}, P\rangle$ is given by adding a new stable place b and a new transition t to B. The initial marking consists of a token in b. Transition t takes a token in b, and produces the initial marking of B plus a token in zero place $P(a?)$. If the name a is not contained in the interface of the given net, then also a copy of Z_a has to be added, and the injective mapping P is extended in the obvious way. A similar construction is defined for an output action prefix $\bar{a}.p$ (we substitute $a!$ for $a?$ in the post-set of the new transition t).

Parallel composition. We have $\langle B_1, A_1, P_1\rangle | \langle B_2, A_2, P_2\rangle = \langle B, A_1 \cup A_2, P\rangle$, with B given by the union of B_1 and B_2 where only $P_1(Z(A_1 \cap A_2))$ and $P_2(Z(A_1 \cap A_2))$ are identified, and with the mapping P given by the union of P_1 and P_2. The initial marking of B is the union of the initial markings of B_1 and B_2.

Restriction. We have $\langle B, A, P\rangle\backslash a = \langle B', A', P'\rangle$, with $A' = A \setminus \{a\}$, $B' = B \setminus \{P(in_a), P(out_a)\}$, and P' is P restricted to $Z(A \setminus \{a\})$.

Now, it should be clear that the image of $Z(fv(p))$ plays the role of the interface, since it is the only part of the net $[\![p]\!]$ that is modified by the construction defined above: it can be increased (as in the case of action prefix), it can be merged with another interface (as in the case of parallel composition) and it can also be restricted (as in the case of the restriction operator).

Since each transition of the interfaced nets associated to the agents consumes and produces at most one token in each zero place, it follows that the corresponding abstract nets obtained under the CTph and the ITph coincide.

Proposition 2.23 *For each agent p, with $[\![p]\!] = \langle B, A, P\rangle$, we have $A_B = I_B$.*

To relate CCS-like processes and interfaced nets behaviours, we add a labelling function ϕ from the transitions of abstract nets to the set of CCS actions. We first need the following proposition.

Proposition 2.24 *Let p be an agent, then each (connected) transaction ξ of $[\![p]\!]$ contains at most one firing of transitions in $P(Z(fv(p)))$.*

Definition 2.25 Let p be an agent. For each (connected) transaction ξ of $[\![p]\!]$, we define

$$\phi(\xi) = \begin{cases} a_i & \text{if } a_i \in fv(p) \text{ and } P(in_{a_i}) \text{ is fired in } \xi \\ \bar{a}_i & \text{if } a_i \in fv(p) \text{ and } P(out_{a_i}) \text{ is fired in } \xi \\ \tau & \text{otherwise} \end{cases}$$

□

The previous definition is correct, because, by Prop. 2.24, if ξ contains the firing of $P(in_{a_i})$ for some action $a_i \in fv(p)$, then it cannot contain neither the firing of $P(in_{a_j})$, nor the firing of $P(out_{a_k})$, for $a_j, a_k \in fv(p)$, $a_j \neq a_i$.

Example 2.26 The interfaced net for the agent $(a_1.a_2.nil|\bar{a}_2.nil)\backslash a_2$ is presented in Fig. 6, together with its (labelled) abstract net. □

It is out of the scope of this presentation to investigate the relationships between the net semantics we propose for CCS-like agents and those already known in the literature, and we prefer to leave this topic for future works. We just remark that our translation is very linear, and that it provides a reasonable concurrent semantics for the simple class of agents considered here. This last assertion is supported by the following results (we assume the reader familiar with the ordinary operational semantics of CCS, see e.g. Milner (1989)).

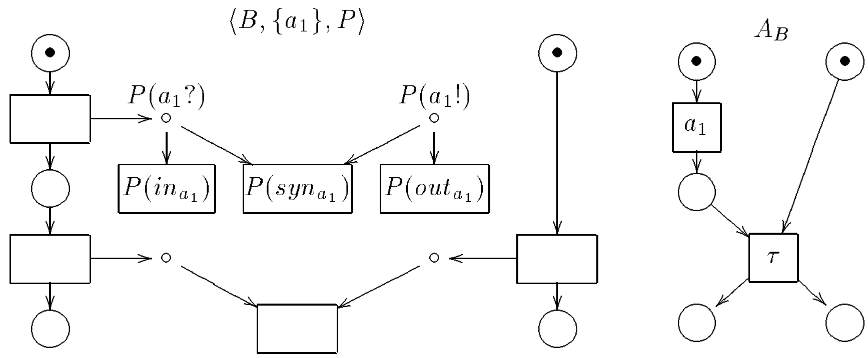

Figure 6: An interfaced net (left) and its abstract net (right).

Definition 2.27 Let p be an agent, let N be a net whose transitions are labelled by ϕ over the set of CCS actions, and let u be a marking of N. We say that p is *bisimilar* to u in N if there exists a relation $\mathcal{B}$ between processes and markings of N such that $p\mathcal{B}u$, and:

1. for each transition $p \xrightarrow{a} p'$ there exists a firing $u[t\rangle u'$ of N such that $\phi(t) = a$ and $p'\mathcal{B}u'$;
2. for each firing $u[t\rangle u'$ of N with $\phi(t) = a$ there exists a transition $p \xrightarrow{a} p'$ such that $p'\mathcal{B}u'$ in N.

□

Proposition 2.28 *Let p be an agent, and $[\![p]\!] = \langle B, A, P\rangle$, then p is bisimilar to the initial marking of A_B.*

Proof (Sketch). After observing that net A_B is acyclic, we proceed by induction on the proof of the transition and on the structure of A_B as obtained by the translation of the process p into $[\![p]\!]$. □

The restriction operator allows hiding local names w.r.t. some external observer. Therefore, a restricted name has only a local scope, and agents that differ only for the local names (i.e., agents that can be obtained one from the other by α-conversion) are usually considered equivalent. We use the symbol $\_ \equiv_\alpha \_$ to denote such equivalence. Prop. 2.30 shows that our translation deals very well with α-conversion.

Definition 2.29 Two A-interfaced nets $\langle B, A, P\rangle$ and $\langle B', A, P'\rangle$ are *isomorphic* if there exists a ZS net isomorphism ψ from B to B' that respect interfaces, i.e., such that $\psi(P(x)) = P'(x)$ for each element $x \in Z(A)$ (either place or transition). □

Proposition 2.30 *If p and q are agents such that $p \equiv_\alpha q$, then $[\![p]\!]$ and $[\![q]\!]$ are isomorphic.*

$$\begin{tikzcd} u & \mathcal{G}(u) \arrow[l, "\epsilon_u"'] \\ a \arrow[u, "\forall f"] \arrow[ur, dotted, "\exists! f'"'] & \end{tikzcd}$$

Figure 7: Coreflection diagram.

3 Universal Constructions

This section has a more abstract and mathematical flavour. Its aim is to give evidence that the definitions and the constructions presented in the previous section are natural. The tool we use is some elementary category theory. In particular, three concepts are useful here. The first notion is the *category of models* itself, where the objects are models (in our case ZS nets) and arrows represent some notion of simulation. The choice of arrows is very informative, since they complement and in a sense redefine – in what isomorphic objects are often identified – the meaning of models.

The second notion is *adjunction*, which is useful to characterize "natural" constructions. The typical scenario includes two categories C_1 and C_2 - where C_2 has more structure than C_1 - and a (usually obvious) forgetful functor $\mathcal{U} : C_2 \longrightarrow C_1$ which deletes the extra structure. It might happen that $\mathcal{U}$ has a left adjoint $\mathcal{F} : C_1 \longrightarrow C_2$. If this is the case, $\mathcal{F}$ represents the "best" construction for adding the extra structure. In fact the left adjoint is unique (up to isomorphism) and satisfies a key universal property.

The third notion is *coreflection*, which is a special kind of adjunction. Here the scenario includes a category C and a subcategory C' of it. Category C represents the "operational" models, while C' defines certain "abstract" models. In addition there is a functor $\mathcal{G} : C \longrightarrow C'$ whose left adjoint is the inclusion functor from C' to C. For every object u of C there is a unique arrow $\epsilon_u : \mathcal{G}(u) \longrightarrow u$ with the universal property that, given any abstract object a of C', for every arrow $f : a \longrightarrow u$ there is a unique arrow $f' : a \longrightarrow \mathcal{G}(u)$ with $f = f'; \epsilon_u$, as the commuting diagram in Fig. refcomdiag illustrates.

This situation is ideal from a semantic point of view. In fact $\mathcal{G}(u)$ can be understood as an abstraction of model u (e.g., its behaviour), with the additional advantage of being at the same time a model itself. The universal property above means that if we observe models from an abstract point of view (i.e., via morphisms originating from objects in C'), then there is an isomorphism (via left composition with ϵ_u) between observations of u and observations of its abstract counterpart $\mathcal{G}(u)$. Thus in a sense, model u seen from C' is the same as $\mathcal{G}(u)$. Again, if a coreflection exists between C and C' with the inclusion as left adjoint, then it is unique up to isomorphism.

3.1 Review of the Monoidal Structure of Petri Nets

Petri net theory can be profitably developed within category theory. Among the existing approaches we mention (Winskel 1987; Meseguer and Montanari 1988; Brown and Gurr 1990). We follow the approach initiated by Meseguer and Montanari (1988) (other references are (Meseguer and Montanari 1990; Degano *et al.* 1989; Degano *et al.* 1996; Meseguer *et al.* 1996; Meseguer *et al.* 1998)), which focuses on the monoidal structure of Petri nets, where the monoidal operation means parallel composition.

The basic observation is that a (place/transition) *Petri net* is just a graph $(S^{\oplus}, T, \partial_0, \partial_1)$ where the set of nodes is the free commutative monoid $S^{\oplus}$ over the set of *places* S (functions $\partial_0, \partial_1 : T \longrightarrow V$ are called *source* and *target*, resp., and we write $t : u \longrightarrow v$, with obvious meaning, to shorten the notation). A *Petri net morphism* is a graph morphism $h = (f : T \longrightarrow T', g : S^{\oplus} \longrightarrow S'^{\oplus})$ (i.e., $g(\partial_i(u)) = \partial'_i(f(u))$ for $i = 0, 1$) where g is a monoid homomorphism (this defines the category **Petri**).

It has been shown (Meseguer and Montanari 1990; Degano *et al.* 1989) that it is possible to enrich the algebraic structure of transitions in order to capture some basic constructions on nets. As an example, the forgetful functor from **CMonRPetri** (Meseguer and Montanari 1990) to **Petri** has a left adjoint which associates to each Petri net N its *marking graph* $\mathcal{C}[N]$, which corresponds to the ordinary operational semantics of N (i.e., its arrows are the step sequences of N). The objects of **CMonRPetri** are called *reflexive Petri commutative monoids* (i.e., Petri nets together with a function $id : S^{\oplus} \longrightarrow T$, where T is a commutative monoid $(T, \otimes, 0)$ and ∂_0, ∂_1 and id are monoid homomorphisms), and its arrows are Petri net morphisms preserving identities and the monoidal structures.

3.2 Collective Token Approach

In this section we describe two constructions involving ZS nets. The first construction starts from a category **dZPetri** (where ZS nets are considered as programs) and exhibits an adjuction from it to a category **HCatZPetri** consisting of some kind of machines, equipped with operations and transitions between states. It is proved that this adjunction corresponds to the token-pushing semantics of ZS nets defined in the Section 2.1, in the sense that the transitions of the machine $\mathcal{Z}[B]$ corresponding to a ZS net B are exactly the abstract stable steps of B. The second construction starts from a different category **ZSN** of ZS nets (which however is strictly related to **HCatZPetri**), having the ordinary category **Petri** of P/T nets as a subcategory, and yields a coreflection corresponding exactly to the construction of the abstract net in Def. 2.7.

3.2.1 Operational Semantics as Adjunction

Analogously to P/T nets, ZS nets can be seen as graphs whose set of nodes is the free commutative monoid over the set of places $S = L \cup Z$ (partitioned into the sets L and Z, respectively of stable and zero places), and we have the following category of ZS nets.

Definition 3.1 [Category **dZPetri**] A *ZS net morphism* is a Petri net morphism $(f, g) : N \longrightarrow N'$ where g is a monoid homomorphism which preserves partitioning of places (i.e., if $a \in Z$ then $g(a) \in Z'^{\oplus}$ and if $a \in S \setminus Z$ then $g(a) \in (S' \setminus Z')^{\oplus}$) and satisfies the additional condition of mapping zero places into pairwise disjoint (non-empty) zero markings (i.e., $\forall z, z' \in Z_B$ with $z \neq z'$, if $g_Z(z) = n_1 a_1 \oplus \cdots \oplus n_k a_k$ and $g_Z(z') = m_1 b_1 \oplus \cdots \oplus m_l b_l$ then we have that $a_i \neq b_j$, for $i = 1, \ldots, k$ and $j = 1, \ldots, l$), which is called the *disjoint image* property. We call *disjoint* any morphism of this kind. This defines the category **dZPetri**. $\square$

Remark 3.2 Since $S^{\oplus}$ is a free commutative monoid we can equivalently represent the set of nodes of a ZS net as $L^{\oplus} \times Z^{\oplus}$ (i.e., sources and targets are pairs whose components are elements of the free commutative monoids over stable and zero places respectively). Thus ZS net morphisms become triples of the form $h = (f, g_L, g_Z)$ where both g_L and g_Z are monoid homomorphisms on the monoids of stable and zero places respectively.

Example 3.3 The graph corresponding to the ZS net MS defined in Fig. 1 has the following set of arcs: $T_{MS} = \{t_0 : (c, 0) \longrightarrow (a \oplus c, 0), t_1 : (a, 0) \longrightarrow (b, z), t_2 : (0, z) \longrightarrow (0, 2z), t_3 : (a, z) \longrightarrow (b, 0), t_4 : (b, 0) \longrightarrow (a, 0)\}$. $\square$

We now introduce the category **HCatZPetri** which is the analogous to **CMonRPetri** for ZS nets. The models of **HCatZPetri** (which we call *ZS graphs*) are more complex than those of **CMonRPetri** since they must be equipped with an operation of composition of arrows to allow for the construction of transactions. Thus **HCatZPetri** is in a sense intermediate between **CMonRPetri** and the category **CatPetri** introduced in Meseguer and Montanari (1990).

Definition 3.4 [Category **HCatZPetri**] A *ZS graph* $H = ((L \cup Z)^{\oplus}, (T, \otimes, 0, id, ;), \partial_0, \partial_1)$ is both a ZS net and a reflexive Petri commutative monoid. In addition, it is equipped with a partial function $\_;\_$ called *horizontal composition*:

$$\frac{\alpha : (u, x) \longrightarrow (v, y),\ \beta : (u', y) \longrightarrow (v', z)}{\alpha; \beta : (u \oplus u', x) \longrightarrow (v \oplus v', z)}. \tag{1}$$

Horizontal composition is associative and has identities

$$id_{(0,x)} : (0, x) \longrightarrow (0, x)$$

for any $x \in Z^{\oplus}$; in addition, the commutative monoidal operator $\_ \otimes \_$ is functorial w.r.t. horizontal composition, i.e., the axiom

$$(\alpha \otimes \beta); (\alpha' \otimes \beta') = (\alpha; \alpha') \otimes (\beta; \beta')$$

holds whenever the right member is defined.

Given two ZS graphs H and H', a *ZS graph morphism* $h = (f, g_L, g_Z) : H \longrightarrow H'$ is both a ZS net morphism and a reflexive Petri monoid morphism such that $f(\alpha; \beta) = f(\alpha); f(\beta)$. ZS graphs and horizontal morphisms (together with the obvious composition and identities) constitute the category **HCatZPetri**. □

Horizontal composition is the key of our approach. It acts as a sequential composition on zero places and as a parallel composition on stable places. This is exactly what we need to model stable steps, because two successive firings in a stable step are allowed iff the stable tokens which are needed are already present in the initial marking.

Proposition 3.5 *If $\alpha : (u, 0) \longrightarrow (v, 0)$ and $\alpha' : (u', 0) \longrightarrow (v', 0)$ are two transitions of a ZS graph then $\alpha; \alpha' = \alpha \otimes \alpha'$.*

Proof. It can be easily verified that:

$$\alpha; \alpha' = (\alpha \otimes id_{(0,0)}); (id_{(0,0)} \otimes \alpha') = (\alpha; id_{(0,0)}) \otimes (id_{(0,0)}; \alpha') = \alpha \otimes \alpha'.$$

By commutativity of $\_ \otimes \_$, it follows that $\alpha; \alpha' = \alpha'; \alpha$. □

Next results show that **HCatZPetri** has **CMonRPetri** as a subcategory.

Proposition 3.6 *The full subcategory of* **HCatZPetri** *whose objects are all and only Petri nets (i.e., $Z = \emptyset$) is isomorphic to* **CMonRPetri**.

Proof. Let H be a ZS graph such that Z_H is empty. Then $\forall \alpha \in T_H$, there exist $u, v \in L_H^{\oplus}$ such that $\alpha : (u, 0) \longrightarrow (v, 0)$. Thus $\forall \alpha, \beta \in H$ it follows that $\alpha; \beta = \alpha \otimes \beta$, i.e., the horizontal composition adds no structure. □

The following theorem defines the algebraic semantics of ZS nets by means of a universal property.

Theorem 3.7 *Let $\mathcal{U}$:* **HCatZPetri** $\longrightarrow$ **dZPetri** *be the functor which forgets about the additional structure on transitions, i.e.,*

$$\mathcal{U}[((L \cup Z)^{\oplus}, (T, \otimes, 0, id, ;), \partial_0, \partial_1)] = (L^{\oplus} \times Z^{\oplus}, T, \partial_0, \partial_1).$$

The functor $\mathcal{U}$ has a left adjoint $\mathcal{Z}$: **dZPetri** $\longrightarrow$ **HCatZPetri**.

Proof (Sketch). Let functor $\mathcal{Z} : \mathbf{dZPetri} \longrightarrow \mathbf{HCatZPetri}$ map a ZS net B into the ZS graph which is defined by the following inference rules and axioms:

$$\frac{(u,x) \in L_B^{\oplus} \times Z_B^{\oplus}}{id_{(u,x)} : (u,x) \longrightarrow (u,x) \in \mathcal{Z}[B]} \qquad \textbf{(identities)}$$

$$\frac{t : (u,x) \longrightarrow (v,y) \in T_B}{t : (u,x) \longrightarrow (v,y) \in \mathcal{Z}[B]} \qquad \textbf{(transitions)}$$

$$\frac{\alpha : (u,x) \longrightarrow (v,y),\ \beta : (u',x') \longrightarrow (v',y') \in \mathcal{Z}[B]}{\alpha \otimes \beta : (u \oplus u', x \oplus x') \longrightarrow (v \oplus v', y \oplus y') \in \mathcal{Z}[B]} \qquad \textbf{(par. comp.)}$$

$$\frac{\alpha : (u,x) \longrightarrow (v,y),\ \beta : (u',y) \longrightarrow (v',z) \in \mathcal{Z}[B]}{\alpha;\beta : (u \oplus u', x) \longrightarrow (v \oplus v', z) \in \mathcal{Z}[B]} \qquad \textbf{(seq. comp.)}$$

Where transitions form a commutative monoid, i.e.,

$$\alpha \otimes \beta = \beta \otimes \alpha, \quad (\alpha \otimes \beta) \otimes \delta = \alpha \otimes (\beta \otimes \delta), \text{ and } \quad id_{(0,0)} \otimes \alpha = \alpha,$$

for any $\alpha, \beta, \delta \in \mathcal{Z}[B]$; the horizontal composition $\_;\_$ is associative and has identities, i.e.,

$$(\alpha;\beta);\delta = \alpha;(\beta;\delta), \text{ and } \quad \alpha; id_{(0,y)} = \alpha = id_{(0,x)};\alpha$$

whenever such compositions are defined; finally the monoidal operator $\_\otimes\_$ is functorial, i.e.,

$$id_{(u,x)} \otimes id_{(v,y)} = id_{(u\oplus v, x \oplus y)}, \text{ and } \quad (\alpha \otimes \alpha');(\beta \otimes \beta') = (\alpha;\beta) \otimes (\alpha';\beta')$$

(the latter holds whenever the rightmost member of the equation is defined). It is easy to verify that mapping $\mathcal{Z}$ extends to a functor which is a right adjoint to functor $\mathcal{U}$. □

The following theorem shows that the algebraic semantics of ZS nets is an extension of the ordinary semantics of P/T nets.

Theorem 3.8 *When restricted to P/T nets, functor $\mathcal{Z}$ coincides with $\mathcal{C}$.*

Proof. The result immediately follows from Prop. 3.5, because if $Z_B = \emptyset$ then $\forall t : (u,x) \longrightarrow (v,y) \in T_B$ is $x = y = 0$. □

Example 3.9 Let MS be the ZS net of our running example whose set of arcs is defined in Ex. 3.3. For instance the arrow $t_1; t_3 \in \mathcal{Z}[MS]$ has source $(2a, 0)$ and target $(2b, 0)$. Instead, notice that the arrow $(t_1 \otimes id_{(a,0)});(id_{(b,0)} \otimes t_3)$ goes from $(3a \oplus b, 0)$ to $(a \oplus 3b, 0)$.

As another example, the following expressions are identified, in $\mathcal{Z}[MS]$, i.e., they all denote the same arrow:

$$\begin{array}{ccc} & t_1; t_2; (t_2 \otimes id_{(0,z)}); (t_3 \otimes id_{(0,2z)}); (t_3 \otimes id_{(0,z)}); t_3 & = \\ = & t_1; t_2; (t_2 \otimes id_{(0,z)}); (t_3 \otimes t_3 \otimes t_3) & = \\ = & t_1; t_2; (t_2 \otimes id_{(0,z)}); (id_{(0,2z)} \otimes t_3); (t_3 \otimes t_3) & = \\ = & t_1; t_2; (t_2 \otimes t_3); (t_3 \otimes t_3). & \end{array}$$

□

Definition 3.10 [Prime transition] A transition $\alpha : (u, 0) \longrightarrow (v, 0)$ of a ZS graph H is *prime* iff α cannot be expressed as the monoidal composition of non-trivial arrows (i.e., $\nexists \beta, \gamma \in H, \beta \neq id_{(0,0)} \neq \gamma$ such that $\alpha = \beta \otimes \gamma$).□

Theorem 3.11 *Given a ZS net B, there is a one-to-one correspondence between arrows $\alpha : (u, 0) \longrightarrow (v, 0) \in \mathcal{Z}[B]$ and abstract stable steps of B. Moreover, if such an arrow is prime then the corresponding abstract stable step is an abstract stable transaction.*

Proof. Given (the equivalence class of) a generic stable step

$$s = u \oplus u_0[t_1\rangle u \oplus u_1 \oplus x_1[t_2\rangle \cdots u \oplus u_{n-1} \oplus x_{n-1}[t_n\rangle u \oplus u_n$$

where each multiset x_i contains (all) the zero tokens at the i-th stage of the step, u is the multiset of the stable tokens which are idle in s and $t_i : (w_i, y_i) \longrightarrow (v_i, z_i)$ for $i = 1, \ldots, n$. Then, the corresponding arrow is

$$\alpha_s = (t_1 \otimes id_{(u,x'_1)}); (t_2 \otimes id_{(0,x'_2)}); \cdots; (t_n \otimes id_{(0,x'_n)})$$

with $x'_1 = 0$ and $x'_i \oplus y_i = x_{i-1}$, for $i = 2, \ldots, n$, where y_i is the multiset of zero places in the source of t_i (see above). The correctness of our definition follows immediately, simply noticing that each diamond transformed s' of s is mapped into an arrow $\alpha_{s'}$ which can be proved equal to α_s thanks to the functoriality axiom. In fact, since s' is a diamond transformed of s, then $\exists k$ such that t_k and t_{k+1} are concurrently enabled, i.e., $y_{k+1} \leq x'_k$. Thus (for generic stable markings v and v'):

$$\begin{array}{ccc} & (t_k \otimes id_{(v,x'_k)}); (t_{k+1} \otimes id_{(v',x'_{k+1})}) & = \\ = & t_k \otimes t_{k+1} \otimes id_{(v \oplus v', x')} & = \\ = & (t_{k+1} \otimes id_{(v',x' \oplus y_k)}); (t_k \otimes id_{(v,x' \oplus z_{k+1})}) & \end{array}$$

where $x' \oplus y_{k+1} = x'_k$. Then, it can be easily checked that the axioms given in the proof of Theorem 3.7 identify equivalent steps only.

For the converse correspondence, let

$$(t_1 \otimes id_{(u_1,x_1)}); (t_2 \otimes id_{(u_2,x_2)}); \cdots ; (t_n \otimes id_{(u_n,x_n)})$$

be any (arbitrarily chosen) *linearization* of a given α, with $t_j : (w_j, y_j) \longrightarrow (v_j, z_j)$. Then, the sequence

$$s = u'_0[t_1\rangle u'_1 \cdots u'_{k-1}[t_k\rangle u'_k[t_{k+1}\rangle u'_{k+1} \cdots u'_{n-1}[t_n\rangle u'_n$$

with $u'_0 = u$ is a stable step.

Now, suppose that α is prime and that $[\![s]\!]$ is not a stable transaction. Then, $\exists s' \in [\![s]\!]$ such that $s' = u[t_{i_1}\rangle p_1 ... p_{n-1}[t_{i_n}\rangle v$ with $p_k \in L_B^{\oplus}$ for a certain index k ($\sum_{j=1}^{n} pre(t_{i_j})(a) = u(a)$ for any stable place a, because α is prime). Then, $\alpha = \beta;\gamma$ with $\beta : (q, 0) \longrightarrow (q', 0)$ and $\gamma : (r, 0) \longrightarrow (r', 0)$ for some (non-trivial) arrows β and γ with $u = q \oplus r$, $v = q' \oplus r'$ and $q' \oplus r = p_k$. This is contradictory, since $\alpha = \beta;\gamma = \beta \otimes \gamma$ while α is prime by hypothesis. $\square$

Th. 3.11 shows that the operational and algebraic semantics of ZS nets coincide, in the sense that we have a precise algebraic characterization of the basic moves allowed by the operational model.

Example 3.12 In our running example the prime arrows of $\mathcal{Z}[MS]$ are

$$\begin{array}{rcl}
\tau_0 & = & t_0, \\
\tau_4 & = & t_4, \\
\alpha_1 & = & t_1; t_3, \\
\alpha_2 & = & t_1; t_2; (t_3 \otimes id_{(0,z)}); t_3, \\
& \vdots & \\
\alpha_i & = & t_1; \beta_i; \delta_i, \\
& \vdots & \\
\text{with } \beta_i & = & t_2; (t_2 \otimes id_{(0,z)}); \cdots ; (t_2 \otimes id_{(0,(i-2)z)}) \\
\delta_i & = & (t_3 \otimes id_{(0,(i-1)z)}); \cdots ; (t_3 \otimes id_{(0,z)}); t_3
\end{array}$$

The correspondence with the abstract stable transactions of MS, which are given in Ex. 2.8, is the intuitive one. As a further example, some more compact notations to define arrows α_i are either $\delta_i = t_3 \otimes \cdots \otimes t_3$ where t_3 is repeated exactly i times or $\alpha_i = t_1; t_2; (t_2 \otimes t_3); \cdots ; (t_2 \otimes t_3); (t_3 \otimes t_3)$ where expression $(t_2 \otimes t_3)$ appears exactly $i - 2$ times. $\square$

3.2.2 Abstract Semantics as Coreflection

We now present the universal construction yielding the abstract semantics of our nets. To this purpose we define a category **ZSN** of ZS nets where the morphisms may map a transition into a transaction. This construction is reminiscent of the construction of **ImplPetri** in Meseguer and Montanari (1990).

Definition 3.13 [Abstract transition] An *abstract transition* of a given ZS net B is either a prime arrow of $\mathcal{Z}[B]$ or a transition of B. □

Definition 3.14 [Refinement morphism] Given two ZS nets B and B', a *refinement morphism* $h : B \longrightarrow B'$ is a (disjoint) ZS net morphism $(f, g_L, g_Z) : B \longrightarrow \mathcal{Z}[B']$ such that function f maps transitions into abstract transitions □

Lemma 3.15 *Given a refinement morphism $h : B \longrightarrow B'$, let $\widetilde{h}$ be its unique extension in* **HCatZPetri**. *Then, $\widetilde{h}$ preserves prime arrows.*

Proof. We want to show that if α is prime in $\mathcal{Z}[B]$, then also $\widetilde{h}(\alpha)$ is prime in $\mathcal{Z}[B']$. Now let $u[t_1\rangle u_1 \cdots u_{n-1}[t_n\rangle u_n$ be a firing sequence corresponding to (a linearization of) α.

- If $n = 1$ then $\widetilde{h}(\alpha) = \widetilde{h}(t_1) = h(t_1)$ which is prime.
- If $n > 1$ then we proceed by contradiction. Suppose that $\widetilde{h}(\alpha)$ is not prime; this implies that $\exists \beta_1, \beta_2 \in \mathcal{Z}[B']$ with $\widetilde{h}(\alpha) = \beta_1 \otimes \beta_2$. Since α is prime, then each t_i involves at least a zero place. Hence, by Def. 3.13, each $h(t_i)$ is a transition. This induces a corresponding linearization for $\widetilde{h}(\alpha)$, given by $h(u)[h(t_1)\rangle h(u_1) \cdots h(u_{n-1})[h(t_n)\rangle h(u_n)$. Then, let $v[s_1\rangle v_1 \cdots v_{k-1}[s_k\rangle v_k$ and $w[s_{k+1}\rangle w_1 \cdots w_{l-1}[s_n\rangle w_l$, with $k + l = n$, be some firing sequences corresponding to β_1 and β_2 respectively. Now, suppose that a diamond transformation at position i can be applied to $\widetilde{h}(\alpha)$, i.e., $h(t_i)$ and $h(t_{i+1})$ are both enabled at $h(u_{i-1})$. The disjoint image property of h allows to infer that also t_i and t_{i+1} are both enabled at u_{i-1}, so a diamond transformation can also be applied to α at position i. Iteratively applying to α the *specular* diamond transformations needed to reach the sequence $h(u)[s_1\rangle u''_1 \cdots u''_{n-1}[s_n\rangle u''_n$ starting from $\widetilde{h}(\alpha)$, we obtain the sequence $u[t_{i_1}\rangle u'_1 \cdots u'_{n-1}[t_{i_n}\rangle u'_n$ with $u'_n = u_n$ and where $h(u'_j) = u''_j$ and $h(t_{i_j}) = s_j$ for $j = 1, \ldots, n$. It is easy to show that the sequences $v'[t_{i_1}\rangle v'_1 \cdots v'_{k-1}[t_{i_k}\rangle v'_k$ and $w'[t_{i_{k+1}}\rangle w'_1 \cdots w'_{l-1}[t_{i_n}\rangle w'_l$ (with $v' \oplus w' = u$, $h(v') = v$, $h(v'_j) = v_j$ for $j = 1, \ldots, k$, $h(w') = w$ and $h(w'_j) = w_j$ for $j = 1, \ldots, l$) define two arrows α_1 and α_2 such that $\alpha = \alpha_1 \otimes \alpha_2$, thus contradicting the hypothesis that α is prime.

□

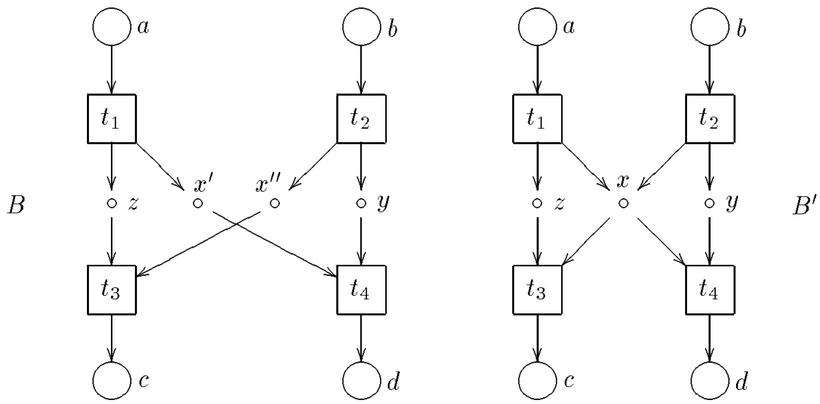

Figure 8: Two ZS nets that are used to illustrate the importance of the disjoint image property (see Remark 3.16).

Remark 3.16 The disjoint image property on zero places required for the morphisms in **dZPetri** is necessary for Lemma 3.15 to hold. As an example, consider ZS nets B and B' in Fig. 8. Then morphism $h : B \longrightarrow B'$, merging both zero places x' and x'' into x and leaving unchanged the rest, maps the prime arrow $\alpha = (t_1 \otimes t_2); (t_3 \otimes t_4) \in \mathcal{Z}[B]$ into the arrow

$$h(\alpha) = (t_1 \otimes t_2); (t_3 \otimes t_4) = (t_1; t_3) \otimes (t_2; t_4)$$

which is not prime. In fact, the morphism h does not respect the disjoint image property.

Definition 3.17 [Category **ZSN**] The category **ZSN** has ZS nets as objects and refinement morphisms as arrows. The composition between two refinement morphisms $h : B \longrightarrow B'$ and $h' : B' \longrightarrow B''$ is defined as the ZS net morphism $\widetilde{h'} \circ h : B \longrightarrow \mathcal{Z}[B'']$, where $\widetilde{h'}$ is the unique extension of h' to a morphism in **HCatZPetri**. Notice that Lemma 3.15 guarantees that $\widetilde{h'} \circ h$ is a refinement morphism. $\square$

Theorem 3.18 *Category* **Petri** *is embedded into* **ZSN** fully *and* faithfully *as a coreflective subcategory. Furthermore the functor $\mathcal{A}[\_]$, which is the right adjoint of the coreflection, maps every ZS net B into its abstract net A_B (see Def. 2.7), i.e., $\mathcal{A}[B] = A_B$.*

Proof (Sketch). We start by defining the functor $\mathcal{D}[\_] : \textbf{Petri} \longrightarrow \textbf{ZSN}$. Let $\mathcal{D}[(S^{\oplus}, T, \partial_0, \partial_1)] = (S^{\oplus} \times \{0\}, T, (\partial_0, 0), (\partial_1, 0))$, i.e., $\mathcal{D}[N]$ is the ZS net generated by N whose nodes are renamed as pairs having the second component equal to 0. The abstract stable transactions (i.e., prime arrows, by Theorem 3.11) of $\mathcal{D}[N]$ are all and only its transitions. Thus, a refinement morphism $h : \mathcal{D}[N] \longrightarrow \mathcal{D}[N']$ maps transitions into transitions. We extend $\mathcal{D}[\_]$ to a functor by defining $\mathcal{D}[(f, g)] = (f, g, 0)$. Next we want to prove that $\mathcal{D}[\_] \dashv \mathcal{A}[\_] : \mathbf{ZSN} \longrightarrow \mathbf{Petri}$ where $\mathcal{A}[\_]$ maps each ZS net B into its *abstract net* A_B. Consider a refinement morphism $h = (f, g_L, g_Z) : B \longrightarrow B'$. Let $\widetilde{h}$ be the unique extension of h in **HCatZPetri**. Morphism $\widetilde{h}$ preserves prime arrows (by Lemma 3.15). Then mapping $\mathcal{A}[\_]$ extends to a functor by defining $\mathcal{A}[h] = (f', g_L)$ with $f'(\sigma) = \widetilde{h}(\sigma)\ \forall \sigma \in \Sigma_B$. It follows that the unit component η_N^C of the adjunction is the identity and the counit component ϵ_B^C maps transitions of the abstract net into appropriate abstract transactions. $\square$

The previous theorem is very important, because the universal properties of the coreflection witnesses that the abstract counterpart A_B of a ZS net B (see Def. 2.7) is the P/T net that better approximates the behaviour of B (in the CTph).

3.3 Individual Token Approach

The aim of this section is to propose an algebraic characterization of the definitions and the constructions presented in Section 2.2. In doing this, we will follow the outline of the previous section using analogous concepts of category theory. Once again, the *category of models* (objects are models and arrows represent some notion of simulation) is the category **dZPetri** presented in Def. 3.1. Then we consider a construction that exhibits an *adjunction* from **dZPetri** to a category **ZSCGraph**. It is proved that this adjunction is striclty related to the operational semantics of ZS nets as defined in the Section 2.2. Our second construction starts from a category **ZSC** of ZS nets and more complex morphisms, having the ordinary category **Petri** of P/T nets as a subcategory, and yields a *coreflection* corresponding exactly to the construction of the causal abstract net in Def. 2.20.

3.3.1 Review of the Axiomatization of Concatenable Processes

The algebraic structure of process is well exploited in (Degano *et al.* 1989; Degano *et al.* 1996; Sassone 1996). There it is shown how to associate a free symmetric strict monoidal category (see Appendix A) $\mathcal{F}[N]$ to each net N in such a way that, under two suitable axioms, it characterizes the concatenable processes of N. This is due to the existence of a left adjoint functor $\mathcal{F} : \mathbf{Petri} \longrightarrow \mathbf{SSMC}^{\oplus}$ to the forgetful functor $\mathcal{U} : \mathbf{SSMC}^{\oplus} \longrightarrow \mathbf{Petri}$. This axiomatization would be useful to shorten the notation in the sketched proof of Th. 3.30. Given a net N the category $\mathcal{F}[N]$ has the elements of $S_N^{\oplus}$ as objects, while its arrows are generated by the following inference rules:

$$\frac{u \in S_N^{\oplus}}{id_u : u \longrightarrow u \in \mathcal{F}[N]} \qquad \textbf{(identities)}$$

$$\frac{t : u \longrightarrow v \in T_N}{t : u \longrightarrow v \in \mathcal{F}[N]} \qquad \textbf{(transitions)}$$

$$\frac{a, b \in S_N}{c_{a,b} : a \oplus b \longrightarrow b \oplus a \in \mathcal{F}[N]} \qquad \textbf{(symmetries)}$$

$$\frac{\alpha : u \longrightarrow v,\ \beta : u' \longrightarrow v' \in \mathcal{F}[N]}{\alpha \otimes \beta : u \oplus u' \longrightarrow v \oplus v' \in \mathcal{F}[N]} \qquad \textbf{(parallel composition)}$$

$$\frac{\alpha : u \longrightarrow v,\ \beta : v \longrightarrow w \in \mathcal{F}[N]}{\alpha;\beta : u \longrightarrow w \in \mathcal{F}[N]} \qquad \textbf{(sequential composition)}$$

modulo the axioms expressing that $\mathcal{F}[N]$ is a strict monoidal category, and the axioms stating that the collection $\{c_{u,v}\}_{u,v \in S_N^{\oplus}}$ plays the role of the symmetry natural isomorphism which makes $\mathcal{F}[N]$ into a ssmc (the terms $c_{u,v}$ for $u, v \in S_N^{\oplus}$ denote the arrows obtained composing $c_{a,b}$ for $a, b \in S_N$ by recursive rules analogous to axioms (4) given in Th. 3.24).

Theorem 3.19 (Sassone 1996) *Given a net N, the concatenable processes of N are isomorphic to the arrows of the category $\mathcal{P}[N]$ which is the monoidal quotient of the free ssmc on N ($\mathcal{F}[N]$) modulo the axioms*

$$c_{a,b} = id_{a \oplus b} \quad \textit{if} \quad a \neq b \in S_N, \tag{2}$$

$$s; t; s' = t \quad \textit{if} \quad t \in T_N, \textit{ and } s, s' \textit{ are symmetries.} \tag{3}$$

The previous construction provides an algebraic view of net computations which is strictly related to a process understanding of the causal behaviour of a net, but which is not functorial. The main problem is that there exist reasonable morphisms of nets which cannot be extended to a monoidal functor. We report below a convincing example illustrated in Meseguer *et al.* (1998).

Example 3.20 Consider the nets N and N' pictured in Fig. 9 and the net morphism $f : N \longrightarrow N'$ such that $f(t_i) = t'_i$, $f(a_i) = a'$, $f(b) = b'$ and $f(c) = c'$ for $i = 0, 1$. Morphism f cannot be extended to a functor $\mathcal{P}[f] : \mathcal{P}[N] \longrightarrow \mathcal{P}[N']$. In fact, if such an extension F existed, then $F(t_0 \otimes t_1) = F(t_0) \otimes F(t_1) = t'_0 \otimes t'_1$ by the monoidality of F, and since $t_0 \otimes t_1 = t_1 \otimes t_0$ in $\mathcal{P}[N]$, it would follow that $t'_0 \otimes t'_1 = t'_1 \otimes t'_0$ which is impossible, as the two expressions denote different processes in $\mathcal{P}[N']$. $\square$

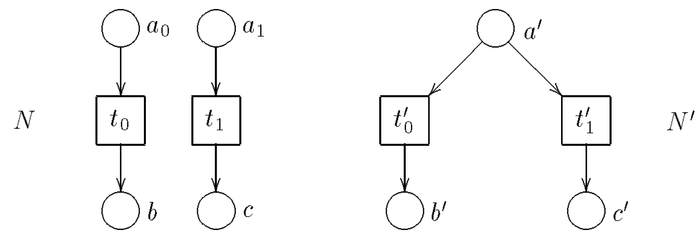

Figure 9: Two nets for which there exists a trivial morphism that cannot be extended to a monoidal functor among the corresponding categories of concatenable processes.

We show that this kind of morphisms are reasonably avoided in the category **dZPetri**. Moreover, this choice is justified by the necessity to preserve atomic behaviours through morphisms.

3.3.2 Operational Semantics as Adjunction

We present here the universal construction yielding the operational semantics of zero safe nets in the ITph interpretation. Disjoint morphisms play a very important role here. If we restrict to consider disjoint morphisms only, then we avoid the awkward situation arising from Ex. 3.20. Moreover, if we identified two different zero places via a (non-disjoint) morphism then the behaviour of the abstract model might dramatically change. Since we use zero places to specify a synchronization mechanism, it is important to ensure that this mechanism is always preserved.

The next definition introduces a category of more structured models (called *ZS causal graphs*) which is reminiscent of the constructions both of marking graphs and free ssmc's.

Definition 3.21 [Category **ZSCGraph**] A *ZS causal graph*

$$E = ((L \cup Z)^{\oplus}, (T, \otimes, 0, id, *), \partial_0, \partial_1)$$

is both a ZS net and a reflexive Petri monoid. In addition, it comes equipped with a partial function $\_ * \_$ called *horizontal composition*:

$$\frac{\alpha : (u, x) \longrightarrow (v, y),\ \ \beta : (u', y) \longrightarrow (v', y')}{\alpha * \beta : (u \oplus u', x) \longrightarrow (v \oplus v', y')}$$

and a family of *horizontal swappings* $\{e_{x,y} : (0, x \oplus y) \longrightarrow (0, y \oplus x)\}_{x,y \in Z^{\oplus}}$. Horizontal composition is associative and has identities $id_{(0,x)}$ for any $x \in Z^{\oplus}$. The monoidal operator $\_ \otimes \_$ is functorial w.r.t. horizontal composition. The (horizontal) *naturality* axiom

$$e_{x,x'} * (\beta \otimes \alpha) = (\alpha \otimes \beta) * e_{y,y'}$$

holds for any $\alpha : (u, x) \longrightarrow (v, y)$ and $\beta : (u', x') \longrightarrow (v', y')$. Moreover, the following *coherence* axioms are satisfied for any $x, y, y' \in Z^{\oplus}$:

$$e_{x,y} * e_{y,x} = id_{(0,x\oplus y)} \qquad e_{x,y\oplus y'} = (e_{x,y} \otimes id_{(0,y')}) * (id_{(0,y)} \otimes e_{x,y'}).$$

A morphism h between two ZS causal graphs E and E' is a monoidal disjoint morphism which in addition respects horizontal composition and swappings (i.e., $h(\alpha * \beta) = h(\alpha) *' h(\beta)$ and $h(e_{x,y}) = e'_{h(x),h(y)}$). This defines the category **ZSCGraph**. □

Horizontal composition is the key feature of our approach. It behaves like sequential composition on zero places and like the ordinary parallel composition on stable places. This is necessary to avoid the construction of steps which reuse stable tokens. Swappings are used to specify the causality relation among produced and consumed zero tokens.

Proposition 3.22 *If $\alpha : (u, 0) \longrightarrow (v, 0)$ and $\alpha' : (u', 0) \longrightarrow (v', 0)$ are two transition of a ZS causal graph then $\alpha \otimes \alpha' = \alpha' \otimes \alpha$ and $\alpha * \alpha' = \alpha \otimes \alpha'$.*

Proof. It can be easily verified that:

$$\begin{aligned} \alpha \otimes \alpha' &= (\alpha \otimes id_{(0,0)}) * (id_{(0,0)} \otimes \alpha') = \\ &= \alpha * \alpha' = \\ &= (id_{(0,0)} \otimes \alpha) * (\alpha' \otimes id_{(0,0)}) = \\ &= \alpha' \otimes \alpha. \end{aligned}$$

□

Proposition 3.23 *The full subcategory of* **ZSCGraph** *whose objects are Petri nets (i.e., $Z = \emptyset$) is isomorphic to* **CMonRPetri***.*

Proof. Let E be a ZS causal graph such that Z_E is empty. Then $\forall \alpha \in T_E$, there exist $u, v \in L_E^{\oplus}$ such that $\alpha : (u, 0) \longrightarrow (v, 0)$. Thus $\_ * \_$ is no longer partial and it can be applied to any pair of arrows: $\forall \alpha, \beta \in E$ it follows that $\alpha * \beta = \alpha \otimes \beta$ (by Prop. 3.22), i.e., the horizontal composition adds no structure and the parallel composition is commutative. Eventually, observe that the collection of swappings is empty. □

Next theorem defines the algebraic semantics of ZS nets by means of a universal property.

Theorem 3.24 *The obvious forgetful functor*

$$\mathcal{U} : \mathbf{ZSCGraph} \longrightarrow \mathbf{dZPetri}$$

has a left adjoint $\mathcal{CG} : \mathbf{dZPetri} \longrightarrow \mathbf{ZSCGraph}$ *which maps a ZS net* B *into the ZS causal graph* $\mathcal{CG}[B]$ *whose arrows are generated by the following inference rules:*

$$\frac{(u,x) \in L_B^{\oplus} \times Z_B^{\oplus}}{id_{(u,x)} : (u,x) \longrightarrow (u,x) \in \mathcal{CG}[B]} \qquad \textbf{(identities)}$$

$$\frac{t : (u,x) \longrightarrow (v,y) \in T_B}{t : (u,x) \longrightarrow (v,y) \in \mathcal{CG}[B]} \qquad \textbf{(transitions)}$$

$$\frac{z, z' \in Z_B}{d_{z,z'} : (0, z \oplus z') \longrightarrow (0, z' \oplus z) \in \mathcal{CG}[B]} \qquad \textbf{(symmetries)}$$

$$\frac{\alpha : (u,x) \longrightarrow (v,y),\ \ \beta : (u',x') \longrightarrow (v',y') \in \mathcal{CG}[B]}{\alpha \otimes \beta : (u \oplus u', x \oplus x') \longrightarrow (v \oplus v', y \oplus y') \in \mathcal{CG}[B]} \qquad \textbf{(par. comp.)}$$

$$\frac{\alpha : (u,x) \longrightarrow (v,y),\ \ \beta : (u',y) \longrightarrow (v',z) \in \mathcal{CG}[B]}{\alpha * \beta : (u \oplus u', x) \longrightarrow (v \oplus v', z) \in \mathcal{CG}[B]} \qquad \textbf{(seq. comp.)}$$

modulo the axioms expressing that the arrows form a (strict) monoid:

$$(\alpha \otimes \beta) \otimes \delta = \alpha \otimes (\beta \otimes \delta), \text{ and } id_{(0,0)} \otimes \alpha = \alpha = \alpha \otimes id_{(0,0)}$$

(for any arrows $\alpha, \beta, \delta \in \mathcal{CG}[\mathcal{B}]$*); the axioms expressing that horizontal composition* $\_ * \_$ *is associative and has identities:*

$$(\alpha * \beta) * \delta = \alpha * (\beta * \delta), \text{ and } \alpha * id_{(0,y)} = \alpha = id_{(0,x)} * \alpha$$

(whenever such compositions are well-defined); the functoriality axioms

$$(\alpha \otimes \alpha') * (\beta \otimes \beta') = (\alpha * \beta) \otimes (\alpha' * \beta'), \text{ and } id_{(u,x)} \otimes id_{(v,y)} = id_{(u \oplus v, x \oplus y)}$$

(the former whenever the rightmost member of the equation is defined); finally, the axioms expressing that the collection of swappings $d_{x,y}$ *plays the role of the "horizontal" natural isomorphism:*

$$d_{x,x'} * (\beta \otimes \alpha) = (\alpha \otimes \beta) * d_{y,y'}, \text{ and } d_{z,z'} * d_{z',z} = id_{z \oplus z'}$$

for any arrows $\alpha : (u,x) \longrightarrow (v,y), \beta : (u',x') \longrightarrow (v',y') \in \mathcal{CG}[B]$*, and for any* $z, z' \in Z_B$*, where* $d_{x,y}$ *for* $x, y \in Z_B^{\oplus}$ *denotes any term obtained from the basic symmetries by applying recursively the rules:*

$$\begin{aligned} d_{0,x} &= id_{(0,x)} = d_{x,0}, \\ d_{z \oplus x, y} &= (id_{(0,z)} \otimes d_{x,y}) * (d_{z,y} \otimes id_{(0,x)}), \text{ and} \\ d_{x, y \oplus z} &= (d_{x,y} \otimes id_{(0,z)}) * (id_{(0,y)} \otimes d_{x,z}). \end{aligned} \qquad (4)$$

$$\begin{tikzcd} B \arrow[r, "\eta_B"] \arrow[dr, "h"'] & \mathcal{U}[\mathcal{CG}[B]] \arrow[d, "\mathcal{U}[k]"] \\ & \mathcal{U}[E] \end{tikzcd}$$

Figure 10: Universality of η_B in **dZPetri**.

Proof. It follows immediately from the definition that $\mathcal{CG}[\mathcal{B}]$ is a ZS causal graph. We need to show that it is the *free* ZS causal graph on B. Let $\eta_B : B \longrightarrow \mathcal{U}[\mathcal{CG}[\mathcal{B}]]$ the disjoint ZS net morphism which is the identity on places and the obvious injection on transitions. We show that η_B is universal, i.e., for any ZS causal graph E and for any disjoint ZS net morphism $h = (f, g_L, g_Z) : B \longrightarrow \mathcal{U}[E]$, there exists a unique ZS causal graph morphism $k : \mathcal{CG}[\mathcal{B}] \longrightarrow E$ such that the diagram in Fig. 10 commutes in **dZPetri**. For such diagram to commute, morphisms k and h must agree on the generators of $\mathcal{CG}[\mathcal{B}]$ and the extension of k to tensor and horizontal composition is uniquely determined by its definition on the generator (namely, $k((u,x)) = (g_L(u), g_Z(x))$, $k(t) = f(t)$, $k(id_{(u,x)}) = id_{(g_L(u), g_Z(x))}$, $k(d_{z,z'}) = e_{g_Z(z), g_Z(z')}$, $k(\alpha * \beta) = k(\alpha) * k(\beta)$ and $k(\alpha \otimes \beta) = k(\alpha) \otimes k(\beta)$). To conclude the proof we have just to show that k is well-defined. This can be done by verifying that k preserves the axioms which generate $\mathcal{CG}[B]$. $\square$

The following theorem shows that the algebraic semantics of ZS nets is an extension of the ordinary semantics of P/T nets.

Theorem 3.25 *When restricted to P/T nets, functor $\mathcal{CG}$ coincides with $\mathcal{C}$.*

Proof. From Prop. 3.22, because if $Z_B = \emptyset$ then $\forall t : (u, x) \longrightarrow (v, y) \in T_B$ is $x = y = 0$. $\square$

The ZS causal graph $\mathcal{CG}[B]$ is still too concrete w.r.t. the operational semantics of ZS nets defined in the previous section. More precisely we need two more axioms (which are the transposition of axioms (2) and (3) of Th. 3.19).

Definition 3.26 Given a ZS net B, let us denote by $\mathcal{CG}[B]/\Psi$ the quotient of the free ZS causal graph $\mathcal{CG}[B]$ generated by B in **ZSCGraph** modulo the axioms

$$d_{z,z'} = id_{(0, z \oplus z')} \quad \text{if} \quad z \neq z' \in Z_B, \tag{5}$$

$$d * t * d' = t \quad \text{if} \quad t \in T_B, \text{ and } d, d' \text{ are swappings.} \tag{6}$$

$\square$

$$
\begin{tikzcd}
\mathcal{CG}[B] \arrow[r, "Q_\Psi"] \arrow[dr, "k"'] & \mathcal{CG}[B]/\Psi \arrow[d, dotted, "k_\Psi"] \\
& E
\end{tikzcd}
$$

Figure 11: Quotient diagram in **ZSCGraph**.

$$
\begin{tikzcd}
\mathcal{CG}[B] \arrow[rr, "Q_\Psi"] \arrow[dr, "\mathcal{CG}[h]"'] & & \mathcal{CG}[B]/\Psi \arrow[d, dotted, "k_\Psi"] \\
& \mathcal{CG}[B'] \arrow[r, "Q'_\Psi"'] & \mathcal{CG}[B']/\Psi
\end{tikzcd}
$$

Figure 12: The unique extension of morphism $h : B \longrightarrow B'$ in **ZSCGraph**.

The quotient $\mathcal{CG}[B]/\Psi$ is such that for any ZS causal graph morphism $k : \mathcal{CG}[B] \longrightarrow E$ which respects axioms (5) and (6) (e.g., $k(d_{z,z'}) = id_{(0,k(z)\oplus k(z'))}$ for any $z \neq z' \in Z_B$ and $k(d * t * d') = k(t)$ for any transition t and swappings d, d' in $\mathcal{CG}[B]$), there exists a unique arrow k_Ψ such that the diagram in Fig. 11 commutes in **ZSCGraph**, where Q_Ψ is the obvious ZS causal graph morphism associated to the (least) congruence generated by the imposed axiomatization.

Proposition 3.27 *For any disjoint morphism $h : B \longrightarrow B'$ in* **dZPetri** *there is a unique extension $\hat{h} : \mathcal{CG}[B]/\Psi \longrightarrow \mathcal{CG}[B']/\Psi$ of h in* **ZSCGraph**.

Proof. Let h as in the hypothesis and take

$$k = Q'_\Psi \circ \mathcal{CG}[h] : \mathcal{CG}[B] \longrightarrow \mathcal{CG}[B']/\Psi.$$

It can be easily verified that k respects axioms (5) and (6) (because h is disjoint and maps transitions to transitions), thus k_Ψ is uniquely determined. Then, take $\hat{h} = k_\Psi = (Q'_\Psi \circ \mathcal{CG}[h])_\Psi$ (see Fig. 12). $\square$

Example 3.28 Let MS be the ZS net of our running example whose set of arcs is defined in Ex. 3.3. For instance the arrow $t_1 * t_3 \in \mathcal{CG}[MS]/\Psi$ has source $(2a, 0)$ and target $(2b, 0)$. Instead, notice that the arrow $(t_1 \otimes id_{(a,0)}) * (id_{(b,0)} \otimes t_3)$ goes from $(3a \oplus b, 0)$ to $(a \oplus 3b, 0)$.

As another example, the following expressions are all identified, i.e., they all denote the same arrow in $\mathcal{CG}[MS]/\Psi$:

$$\begin{array}{rcl} & t_1 * t_2 * (t_2 \otimes t_3) * (t_3 \otimes t_3) & = \\ = & t_1 * t_2 * (t_2 \otimes id_{(0,z)}) * (t_3 \otimes t_3 \otimes t_3) & = \\ = & t_1 * t_2 * d_{z,z} * (t_2 \otimes id_{(0,z)}) * (t_3 \otimes t_3 \otimes t_3) & = \\ = & t_1 * t_2 * (id_{(0,z)} \otimes t_2) * (t_3 \otimes t_3 \otimes t_3) & = \\ = & t_1 * t_2 * (t_3 \otimes t_2) * (t_3 \otimes t_3). & \end{array}$$

□

To give the expected correspondence among algebraic and operational semantics we rephrase in the current setting the definition of prime arrows given for the category **HCatZPetri**. Since the two notion are essentially the same, we hope that the reader will not be confused by the overloading of the terminology.

Definition 3.29 [Prime arrow II] An arrow $\alpha : (u,0) \longrightarrow (v,0)$ of a ZS causal graph E is *prime* iff α cannot be expressed as the monoidal composition of non-trivial arrows. □

Theorem 3.30 *Given a ZS net B, there is a one-to-one correspondence between arrows $\alpha : (u,0) \longrightarrow (v,0) \in \mathcal{CG}[B]/\Psi$ and the connected steps of B. Moreover, if such an arrow is prime (and is not an identity) then the corresponding connected step is a connected transaction.*

Proof (Sketch). For any arrow β of $\mathcal{CG}[B]/\Psi$ we define inductively on the structure of β a concatenable process $C(\beta)$ of N_B as follows:

- $C(t) = t$,
- $C(id_{(u,x)}) = id_u \otimes id_x$,
- $C(d_{z,z}) = c_{z,z}$,
- $C(\beta' \otimes \beta'') = C(\beta') \otimes C(\beta'')$, and
- $C(\beta' * \beta'') = (C(\beta') \otimes u'); (v \otimes C(\beta''))$, where $\beta' : (u,x) \longrightarrow (v,y)$ and $\beta'' : (u',x') \longrightarrow (v',y')$.

It can be verified that any different expression denoting β yields the same result. Moreover, if $\alpha : (u,0) \longrightarrow (v,0) \in \mathcal{CG}[B]/\Psi$ then the process obtained from $C(\alpha)$ by forgetting the label-indexed ordering functions of origins and destinations denotes a connected step of B.

Conversely, let $\xi = [\![\omega]\!]_\approx$ (for some causal firing sequence ω) be a connected step. The concatenable process $pr(\omega)$ of N_B can be denoted algebraically as the sequential and parallel composition of transitions, identities and symmetries. Moreover, we can take an equivalent process C

$$
\begin{tikzcd}
B \arrow[d, "h"'] \\
\mathcal{U}[\mathcal{CG}[B']/\Psi]
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
\mathcal{CG}[B] \arrow[r, "Q_\Psi"] \arrow[d, "\mathcal{CG}[h]"'] & \mathcal{CG}[B]/\Psi \arrow[d, dotted, "\hat{h}"] \\
\mathcal{CG}[\mathcal{U}[\mathcal{CG}[B']/\Psi]] \arrow[r, "\epsilon_{\mathcal{CG}[B']/\Psi}"'] & \mathcal{CG}[B']/\Psi
\end{tikzcd}
$$

Figure 13: The unique extension of morphism $h : B \longrightarrow \mathcal{U}[\mathcal{CG}[B']/\Psi]$ in **ZSCGraph**.

without stable symmetries, i.e., C can be expressed as $\alpha_1; \ldots; \alpha_n$ where $\alpha_i = \beta_i \otimes u_i \otimes x_i$ with $u_i \in L_B^{\oplus}$, $x_i \in Z_B^{\oplus}$ and $\beta_i \in T_B \cup \{c_{kz,z}\}_{z \in Z_B, k \in \mathbb{N}}$. Then take $\alpha' = \alpha'_1 * \ldots * \alpha'_n$ where $\alpha'_i = \beta'_i \otimes x_i$ and $\beta'_i = \beta_i$ if $\beta_i \in T_B$ and $\beta'_i = d_{kz,z}$ if $\beta_i = c_{kz,z}$ for some zero place z and integer k. Eventually, $\alpha = \alpha' \otimes u'$ where u' is the multiset of idle tokens. $\square$

Example 3.31 In our running example, some prime arrows of $\mathcal{CG}[MS]$ are t_0, $t_1 * t_3$, and $t_1 * t_2 * (t_2 \otimes t_2) * (t_3 \otimes t_2 \otimes t_3 \otimes t_3) * (t_3 \otimes t_3)$. As a counterexample, the arrow $(t_1 \otimes t_1) * d_{z,z} * (t_2 \otimes t_3) * (t_3 \otimes t_3)$ is not prime. $\square$

3.3.3 Abstract Semantics as Coreflection

We conclude this section presenting the universal construction of the abstract semantics of ZS nets in the ITph. We make use of a category **ZSC** whose objects are ZS nets and whose morphisms allow for the refinement of a transition into an abstract connected transaction.

Definition 3.32 Given ZS net B, a *causal abstract transition* of a $\mathcal{CG}[B]/\Psi$ is either a prime arrow of $\mathcal{CG}[B]/\Psi$ or a transition of B. $\square$

Definition 3.33 Given two ZS net B and B', a *causal refinement morphism* $h : B \longrightarrow B'$ is a disjoint ZS net morphism $h = (f, g_L, g_Z)$ from B to (the image through the forgetful functor of) $\mathcal{CG}[B']/\Psi$ such that function f maps transitions into causal abstract transitions. $\square$

Since morphism h is disjoint, a transition can be refined into a transaction if and only if both its preset and its postset are stable. Transition involving zero places can only be mapped to transitions.

Lemma 3.34 *Each causal refinement morphism $h : B \longrightarrow B'$ uniquely extends to a morphisms $\hat{h} : \mathcal{CG}[B]/\Psi \longrightarrow \mathcal{CG}[B']/\Psi$ in* **ZSCGraph**. *In addition, such extension preserves prime arrows.*

Proof. Let $\epsilon_E : \mathcal{CG}[\mathcal{U}[E]] \longrightarrow E$ denote the *counit* of the adjunction between **dZPetri** and **ZSCGraph** defined in Th.3.24, then assume $\hat{h} = k_\Psi$ for $k = \epsilon_{\mathcal{CG}[B']/\Psi} \circ \mathcal{CG}[h]$ (see Fig. 13). $\square$

Definition 3.35 [Category **ZSC**] The category **ZSC** has ZS nets as objects and causal refinement morphisms as arrows, their composition being defined through the extension in **ZSCGraph** given by Lemma 3.34. □

Theorem 3.36 *Category* **Petri** *is embedded in* **ZSC** *fully and faithfully as a coreflective subcategory. Furthermore, the functor $\mathcal{I}[\_]$, which is the right adjoint of the coreflection, maps every ZS net B into its causal abstract net I_B (see Def. 2.20).*

Proof (Sketch). The connected transactions (i.e., prime arrows, by Theorem 3.30) of a P/T net are all and only its transitions. Thus a causal refinement morphism $h : N \longrightarrow N'$ maps transitions into transitions. Next we want to prove that the obvious inclusion functor from **Petri** to **ZSC** has a right adjoint $\mathcal{I}[\_] : \mathbf{ZSC} \longrightarrow \mathbf{Petri}$ such that $\mathcal{I}[\_]$ maps each ZS net B into its causal abstract net I_B. We verify that $\mathcal{I}[\_]$ extends to a functor. Consider a causal refinement morphism $h = (f, g_L, g_Z) : B \longrightarrow B'$. Let $\hat{h} : \mathcal{CG}[B]/\Psi \longrightarrow \mathcal{CG}[B']/\Psi$ be the unique extension of h in **ZSCGraph**. Morphism $\hat{h}$ preserves prime arrows (by Lemma 3.34). Then we define $\mathcal{I}[h] = (f', g)$ with $f'(\xi) = \hat{h}(\xi)$ for any $\xi \in \Xi_B$ and $g(a) = g_L(a)$ for any $a \in L_B$. It follows that the unit component η^I_N of the adjunction is the identity and the counit component ϵ^I_B maps each transition of the abstract net into the appropriate connected transaction. □

Conclusions

In this paper we have presented both the CTph and ITph semantics for ZS nets, considering suitable equivalence relations over (enriched) firing sequences, which satisfy some additional *atomicity* constraint. Correspondingly, for the CTph approach, we have based our categorical models on monoidal graphs equipped with an operation of horizontal composition. For the ITph approach, we also have introduced a collection of special transitions, called *swappings*, to represent the permutations of tokens which are all present at the same time into the same place, following the style of Degano *et al.* (1989).

We find out that, whatever the philosophy adopted is, we need analogous restrictions for the arrows of categories **ZSC** and **ZSN**, which are used to define the abstract semantics. Moreover, the multicasting example should somewhat concretize the philosophical distinction between *collective* and *individual* token semantics.

We have also shown that CCS-like *programs* may be easily interpreted by representing the communication channels as zero places, in the style of the multicasting example. Thus, ZS nets could offer a suitable framework for an uniform approach to concurrent language translations.

Finally we want to mention a connection between ZS nets and the tile model (Gadducci 1996; Gadducci and Montanari 1997; Bruni *et al.*

1998). Tiles are rewrite rules, similar to SOS inference rules, equipped with three operations of composition: horizontal, vertical and parallel. Horizontal composition builds tiles corresponding to synchronized steps, vertical composition to sequentialized steps and parallel composition to concurrent steps. Tiles can be exactly interpreted as double cells in suitable monoidal double categories, and provide an expressive and clean metalanguage to define a variety of models of computation.

ZS nets represent the simple case where basic tiles are net transitions. The idea is to represent the commutative monoids of stable and zero places as two suitable monidal categories with only one object called 0, which is the same in both representations and it is also the neutral element of the tensor product. Notice that sequential composition in the two categories is equivalent to parallel composition. In fact for any $f, g : 0 \longrightarrow 0$

$$f; g = (f \otimes 0); (0 \otimes g) = f; 0 \otimes 0; g = f \otimes g.$$

Then, a generic ZS net can be thought of as the computad of a (symmetric) strict monoidal double category (Bruni *et al.* 1998; Gadducci and Montanari 1997), with only one object 0, and whose horizontal and vertical arrows consist of the elements of the free commutative monoid of stable and zero places respectively. In this setting, the horizontal composition of tiles corresponds to the horizontal composition of arrows in the category of ZS (causal) graphs, and the vertical composition of (prime) tiles yields step sequences (see Def. 2.2).

Related Works

Several notion of net abstraction/refinement have been proposed in the literature. An extensive comparison of the different approaches can be found in (Brauer *et al.* 1991; Van Glabbeek and Goltz 1998). Most of the times they find application in a top-down design procedure of a concurrent system, where one starts with a very abstract model of the system and then replaces some components by more detailed structures, until the desired level of abstraction is reached. Typically such refinement processes take several steps to be completed, where at each step a single transition (let say t) of the actual net N is refined into a suitable subnet M, yielding the net $N[t \rightarrow M]$ (the net M is usually equipped with initial and final transitions that, in $N[t \rightarrow M]$, become connected to the pre- and post-set of t by some standard mechanism, e.g. suitable place mergings). We can call *structural* such kind of refinement.

Obviously, a minimal requirement in structural refinement is that the behaviour of the refined net is in some sense consistent with the one of the original net. In general some constraints must be assumed on the "daughter" net M to accomplish the expected behavioural consistency, as e.g. in (Vogler 1987; Valette 1979; Suzuki and Murata 1983). In Vogler

(1990), the refinement of places (as opposed to the refinement of transition) is considered, in the context of a general synchronization operator on nets which composes nets by merging transitions.

A different and appealing idea is elaborated in (Kiehn 1989; Kiehn 1990), where transitions can be interpreted as procedure calls. According to this view, the firing of a transition creates an instance of its corresponding subnet, and such calls can also be recursive.

Our approach is more *behaviour* oriented, in the style of (Winskel 1987; Meseguer and Montanari 1990; Sassone 1996). In particular, for the first time we apply the idea sketched in Meseguer and Montanari (1990) and then hinted at in successive works, of refining transition with *behaviours* of a more concrete net, using a completely algebraic approach. Categorically speaking, since the behaviour of a net can be specified by a free construction, we take a suitable subcategory of the Kliesli category associated to such construction (the choice of the subcategory being justified by the atomicity requirements imposed on transactions).

This is clearly different from the structural approach described above, in that we refine all the transitions of the net N by computations of another (zero safe) net. In particular, the refinements of two transitions t_1 and t_2 of N might use the same kind of resources (e.g., place z of our running example), whereas in the structural approaches, the nets M_1 and M_2 that refine t_1 and t_2 are disjoint (except for places already shared by the pre- and post-sets of t_1 and t_2). Moreover, the coreflections of Th. 3.18 and 3.36 yield the notion of abstract net associated to the refined net, that is missing in all structural approaches.

Before concluding, we only want to mention Desel (1991), where abstraction of nets, according to feasible equivalence relations that define the coarsening, is taken as a primitive notion (as opposed to refinement), and the works on *Petri boxes*, see e.g. (Best *et al.* 1992; Best *et al.* 1993).

Acknowledgements

We would like to thank the anonymous referees for their helpful comments.

References

Best, E., Devillers, R., and Esparza, J. (1993), General Refinement and Recursion for the Box Calculus, *in* "Proceedings STACS'93", *LNCS* **665**, 130–140, Springer-Verlag.

Best, E., Devillers, R., and Hall, J. (1992), The Box Calculus: A New Causal Algebra with Multi-label Communication, *in* "Advances in Petri Nets'92", *LNCS* **609**, 21–69, Springer-Verlag.

Brauer, W., Gold, R., and Vogler, W. (1991), A Survey of Behaviour and Equivalence Preserving Refinements of Petri Nets, *in* "Advances in Petri Nets 1990", (G. Rozenberg, Editor), *LNCS* **483**, 1–46, Springer-Verlag.

Brown, C., and Gurr, D. (1990), A Categorical Linear Framework for Petri Nets, *in* "Proceedings of the 5th LICS Symposium", pp. 208–218, IEEE.

Bruni, R., Meseguer, J., and Montanari, U. (1998), "Process and Term Tile Logic", Technical Report SRI-CSL-98-06, Computer Science Laboratory, SRI International, Menlo Park, California, USA. Also Technical Report TR-98-09, Department of Computer Science, University of Pisa, Italy.

Bruni, R., and Montanari, U. (1997), Zero-Safe Nets, or Transition Synchronization Made Simple, *in* "Proceedings of EXPRESS'97", (C. Palamidessi, J. Parrow, Editors), *ENTCS* **7**.

Bruni, R., and Montanari, U. (1998), Zero-Safe Nets: The Individual Token Approach. *in* "Proceedings 12th Workshop on Algebraic Development Techniques", (F. Parisi-Presicce, Editor), *LNCS* **1376**, 122–140.

Degano, P., Meseguer, J., and Montanari, U. (1989), Axiomatizing Net Computations and Processes, *in* "Proceedings 4th LICS Symposium", pp. 175-185, IEEE.

Degano, P., Meseguer, J., and Montanari, U. (1996), Axiomatizing the Algebra of Net Computations and Processes, *Acta Informatica* **33**(7), 641–667.

Degano, P., De Nicola, R., and Montanari, U. (1988), A Distributed Operational Semantics for CCS based on Condition/Event Systems, *Acta Informatica* **26**, 59–91.

Desel, J. (1991), On Abstraction of Nets, *in* "Advances in Petri Nets 1991", *LNCS* **524**, 78–92, Springer-Verlag.

Gadducci, F. (1996), On the Algebraic Approach to Concurrent Term Rewriting, PhD thesis TD 02/96, Department of Computer Science, University of Pisa.

Gadducci, F., and Montanari, U. (1997), The Tile Model, *in* "Proof, Language and Interaction: Essays in Honour of Robin Milner", (G. Plotkin, C. Stirling, and M. Tofte, Editors), MIT Press, to appear. Also Technical Report TR-96-27, Department of Computer Science, University of Pisa (1996).

Van Glabbeek, R.J., and Goltz, U. (1998), Refinement of Actions and Equivalence Notions for Concurrent Systems, Hildesheimer Informatik Bericht 6/98, Institut fuer Informatik, Universitaet Hildesheim.

Van Glabbeek, R.J., and Plotkin, G.D. (1995), Configuration Structures, *in* "Proc. of the 10th LICS Symposium", (D. Kozen, Editor), pp. 199–209, IEEE.

Van Glabbeek, R.J., and Vaandrager, F. (1987), Petri Net Models for Algebraic Theories of Concurrency, *in* "Proceedings of PARLE", *LNCS* **259**, 224–242. Springer-Verlag.

Goltz, U., and Reisig, W. (1983), The Non-Sequential Behaviour of Petri Nets, *Information and Computation* **57**, 125–147.

Gorrieri, R., and Montanari, U. (1995), On the Implementation of Concurrent Calculi into Net Calculi: Two Case Studies, *Theoretical Computer Science* **141**(1-2), 195–252.

Kiehn, A. (1989), A Structuring Mechanism for Petri Nets, PhD Thesis, Technical Report TUM-I8902, Institut für Informatik, Technische Universität München.

Kiehn, A. (1990), Petri Net Systems and their Closure Properties, *in* "Advances in Petri Nets 1989", (G. Rozenberg, Editor), *LNCS* **424**, 306–328, Springer.

MacLane, S. (1971), "Categories for the Working Mathematician", Springer.

Meseguer, J., and Montanari, U. (1988), Petri Nets are Monoids: A New Algebraic Foundations for Net Theory, *in* "Proceedings of the 3rd LICS Symposium", pp. 155–164, IEEE.

Meseguer, J., and Montanari, U. (1990), Petri Nets are Monoids, *Information and Computation* **88**(2), 105–155.

Meseguer, J., Montanari, U., and Sassone, V. (1996), Process versus Unfolding Semantics for Place/Transition Petri Nets, *Theoretical Computer Science* **153**(1-2), 171-210.

Meseguer, J., Montanari, U., and Sassone, V. (1998), Representation Theorems for Petri Nets, Festschrift in honor of Prof. Wilfried Brauer to appear.

Milner, R. (1989), "Communication and Concurrency", Prentice Hall.

Olderog, E.R. (1987), Operational Petri Net Semantics for CCSP, *in* "Advances in Petri Nets'87", (G. Rozenberg, Editor), *LNCS* **266**, 196–223, Springer-Verlag.

Reisig, W. (1985), "Petri Nets", Springer-Verlag.

Ristori, G. (1994), Modelling Systems with Shared Resources via Petri Nets. PhD thesis TD 05/94, Department of Computer Science, University of Pisa.

Sassone, V. (1996), An Axiomatization of the Algebra of Petri Net Concatenable Processes, *Theoretical Computer Science* **170**(1-2), 277–296.

Suzuki, I., and Murata, T. (1983), A Method for Stepwise Refinement and Abstraction of Petri Nets, *Journal of Computer and System Sciences* **27**, 51–76.

Valette, R. (1979), Analysis of Petri Nets by Stepwise Refinement, *Journal of Computer and System Sciences* **18**, 35–46.

Vogler, W. (1987), Behaviour Preserving Refinements of Petri Nets, *in* "Proceedings 12th International Workshop on Graph-Theoretic Concepts in Computer Science", (G. Tinhofer, G. Schmidt, Editors), *LNCS* **246**, 82–93.

Vogler, W. (1990), Failures Semantics of Petri Nets and the Refinement of Places and Transitions, Technical Report TUM-I9003, Institut für Informatik, Technische Universität München.

Winskel, G. (1982), Event Structure Semantics of CCS and Related Languages, *in* "Proceedings of ICALP'82", *LNCS* **140**, 561–567, Springer-Verlag.

Winskel, G. (1987), Petri Nets, Algebras, Morphisms and Compositionality, *Information and Computation* **72**, 197–238.

A Monoidal Categories

Definition A.1 [Symmetric, strict monoidal category] A *symmetric, strict monoidal category* (MacLane 1971), *ssmc* in the following, is a 4-tuple $(\mathcal{C}, \otimes, e, \gamma)$ where:

$$\begin{tikzcd}
\mathcal{C}\times\mathcal{C}\times\mathcal{C} \arrow[r, "\otimes\times 1"] \arrow[d, "1\times\otimes"'] & \mathcal{C}\times\mathcal{C} \arrow[d, "\otimes"] \\
\mathcal{C}\times\mathcal{C} \arrow[r, "\otimes"'] & \mathcal{C}
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathcal{C} \arrow[r, "\langle 1,e\rangle"] \arrow[dr, "1"'] & \mathcal{C}\times\mathcal{C} \arrow[d, "\otimes"] & \mathcal{C} \arrow[l, "\langle e,1\rangle"'] \arrow[dl, "1"] \\
& \mathcal{C} &
\end{tikzcd}$$

Figure 14: Diagrams for associativity and neutral element of the tensor product.

- $\mathcal{C}$ is the underlying category (with composition $\_;\_$ and identity id_x for each object x);
- $\otimes : \mathcal{C}\times\mathcal{C} \longrightarrow \mathcal{C}$ is a functor called the *tensor product*;
- e is an object of $\mathcal{C}$ called the *unit object*;
- the diagrams in Fig. refssmcfig commute, where $\langle\_,\_\rangle$ is the pairing of functors induced by the cartesian product of categories, $1 : \mathcal{C} \longrightarrow \mathcal{C}$ is the identity functor on $\mathcal{C}$ and $e : \mathcal{C} \longrightarrow \mathcal{C}$ (with a slight abuse of notation) is the constant functor which associates e and id_e respectively to each object and each morphism of $\mathcal{C}$; clearly, these equations state that the tensor product $\_\otimes\_$ is associative on both objects and arrows and that e (respectively id_e) is the unit object (resp. the unit arrow) for $\_\otimes\_$;
- $\gamma : \__1 \otimes \__2 \Rightarrow \__2 \otimes \__1$ is a natural isomorphism called *symmetry* satisfying the Kelly-MacLane *coherence axioms* expressed by the following equations:

$$\begin{aligned}
\gamma_{x\otimes y,z} &= (id_x \otimes \gamma_{y,z});(\gamma_{x,z}\otimes id_y) &\quad (7)\\
\gamma_{x,y};\gamma_{y,x} &= id_{x\otimes y} &\quad (8)
\end{aligned}$$

the first axiom is the key to reduce the symmetry at composed object to the symmetry at the components, while the second axiom simply says that the inverse of γ is $\gamma\Delta$ where Δ is the bifunctor which swaps its arguments, i.e. if $(\mathcal{C}\times\mathcal{C},\pi_1,\pi_2)$ is the product then $\Delta : \mathcal{C}\times\mathcal{C} \longrightarrow \mathcal{C}\times\mathcal{C}$ is such that $\Delta = \langle\pi_2,\pi_1\rangle$ $(\gamma : \otimes \Rightarrow \otimes\Delta : \mathcal{C}\times\mathcal{C}\longrightarrow\mathcal{C})$.

$\square$

Remark A.2 Translating the more general notion of symmetric monoidal category into the special case of symmetric *strict* monoidal category it could seem that also the axiom $\gamma_{e,x} = id_x$ should be stated. However it is immediate to show that the others are sufficient to guarantee this constraint. In fact, since $e \otimes e = e$, it follows that $\gamma_{e,x} = \gamma_{e\otimes e,x} = (id_e \otimes \gamma_{e,x});(\gamma_{e,x} \otimes id_e) = \gamma_{e,x};\gamma_{e,x}$. Thus, composing with $\gamma_{x,e}$, we obtain $id_{e\otimes x} = \gamma_{e,x}$. Finally, recalling that $e \otimes x = x$ we can conclude that $\gamma_{e,x} = id_x$.

Definition A.3 [Categories **SSMC** and **SSMC**$^{\oplus}$] Let $\mathcal{C}$ and $\mathcal{C}'$ be two symmetric, strict monoidal categories. A *symmetric strict monoidal functor* from $\mathcal{C}$ to $\mathcal{C}'$ is a functor $F_{\otimes} : \mathcal{C} \longrightarrow \mathcal{C}'$ which preserves the monoidal structure, i.e. such that:

$$
\begin{aligned}
F(e) &= e' && (9)\\
F(\_{1} \otimes \_{2}) &= F(\_{1}) \otimes' F(\_{2}) && (10)\\
F(\gamma_{x,y}) &= \gamma'_{Fx,Fy} && (11)
\end{aligned}
$$

Let **SSMC** be the category of symmetric strict monoidal categories (as objects) and symmetric strict monoidal functors (as arrows). We denote by **SSMC**$^{\oplus}$ be the full subcategory consisting of the monoidal categories whose objects form *free* commutative monoids. $\square$