Normal Forms for Partitions and Relations*

Roberto Bruni,[1] Fabio Gadducci[2] and Ugo Montanari[1]

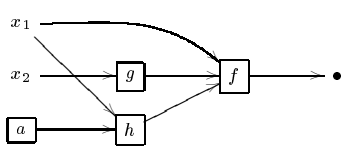
[1] Univ. of Pisa, Dipartimento di Informatica, Corso Italia 40, I-56125 Pisa, Italia.
[2] TU Berlin, Fach. 13 Informatik, Franklinstr. 28/29, D-10587 Berlin, Germany.
bruni@di.unipi.it, gfabio@cs.tu-berlin.de, ugo@di.unipi.it

Abstract. Recently there has been a growing interest towards algebraic structures that are able to express formalisms different from the standard, tree-like presentation of terms. Many of these approaches reveal a specific interest towards their application in the "distributed and concurrent systems" field, but an exhaustive comparison between them is difficult because their presentations can be quite dissimilar. This work is a first step towards a unified view, which is able to recast all those formalisms into a more general one, where they can be easily compared. We introduce a general schema for describing a characteristic normal form for many algebraic formalisms, and show that those normal forms can be thought of as arrows of suitable concrete monoidal categories.

1 Introduction

Since models of computation based on the notion of free and bound *names* are widespread, the notion of *name sharing* is essential for several applications ranging from logic programming, λ-calculus, functional programming and process algebra with restriction (or name hiding mechanisms) to mobile processes (where local names may be communicated to the external world, thus becoming global names). We can think of names as links to communication channels, or to objects, or to locations, or to remote shared resources, or also to some *cause* in the event history of the system. In general, names can be freely α-converted, because the only important information they offer is *sharing*.

An informal "wire and box notation" gives an intuitive understanding of the name sharing mechanism, and of the role played by *auxiliary* term-structure: Wires represents variables and the operators of the signature are denoted by boxes labeled with the corresponding operation symbols. For instance, the term $f(x_1, g(x_2), h(x_1, a))$ over the signature $\Sigma = \{a : 0 \longrightarrow 1, g : 1 \longrightarrow 1, h : 2 \longrightarrow 1, f : 3 \longrightarrow 1\}$ and variables x_1, x_2 admits the graphical representation



* Research partly supported by the EC TMR Network *GETGRATS* through the Technical University of Berlin and the University of Pisa; and by Esprit Working Groups *CONFER2* and *COORDINA*, through the University of Pisa.

Notice that wire duplications (e.g., of x_1) and wires swapping (e.g., of x_2 and a copy of x_1) are auxiliary, in the sense that they belong to any wire and box model, independently from the underlying signature. The properties of the auxiliary structure are far from trivial and could lead to somehow misleading system representations, when their interpretation is not well formalized. For example, let us consider the wire and box diagrams c_1 and c_2 in the picture below:

In a *value-oriented* interpretation, both c_1 and c_2 yield the same term $h(a, a)$. Instead, in a *reference-oriented* interpretation, c_1 and c_2 define different situations: in the former the two arguments of the h operator are uncorrelated, while in the latter they point to the same *shared* location.

Term graphs [32] are a reference-oriented generalization of the ordinary (value-oriented) notion of term, where the sharing of sub-terms can be specified also for closed (i.e., without variables) terms.[1] The distinction is made precise by the axiomatization of *algebraic theories*: Terms and term graphs differ for two axioms, representing, in a categorical setting, the *naturality* of transformations for copying and discharging arguments [8]. Many other mathematical structures have been proposed, for expressing formalisms different from the ordinary tree-like presentation of terms. They range from the *flownomial calculus* of Stefanescu [6, 34], to the *bicategories of processes* of Walters [18, 19], to the *pre-monoidal categories* of Power and Robinson [28], to the *action structures* of Milner [24], to the *interaction categories* of Abramsky [1], to the *sharing graphs* of Hasegawa [16] and to the *gs-monoidal categories* of Corradini and Gadducci [7, 8], just to mention a few (see also [9, 11, 15, 29]). All these structures can be seen as enrichments of symmetric monoidal categories, which give the basis for the description of a distributed environment in terms of a wire and box diagram.

We propose a schema for describing normal forms for this kind of structures, generalizing the one in [12] (and that bears some similarity to the *equational term graph* of [2]), thus obtaining a universal framework where each structure finds its unique standard representation. We describe distributed spaces as sets of *assignments* over sets of variables, distinguishing between four different kinds of assignment, each representing a basic functionality of the space, namely *input* and *output interfaces*, *basic modules*, and *connections*. Changing the constraints on the admissible connections is the key to move between formalisms. We call Σ-spaces the distributed spaces over a signature Σ, and show that the classes of Σ-spaces we are interested in always form a symmetric monoidal category. We then establish a sort of triangular correspondence between various formalisms proposed in the literature (usually in a set-theoretical way), different classes of Σ-spaces and suitable enriched symmetric monoidal categories.

[1] Terms may share variables, but shared sub-terms of a closed term can be freely copied, always yielding an equivalent term.

The structure of the paper is as follows: In Section 2 we give a categorical account of the various formalisms presented in the literature that we want to embed in our concrete normal-form representation. All of them are suitably enriched symmetric monoidal categories. In section 3 we formally define Σ-spaces, and equip them with two operations of parallel and sequential composition, yielding a monoidal category structure. In Section 4 we show how it is possible to have a normal form representation for net processes, relations, labeled partial orders, partitions, and contextual nets using suitable classes of Σ-spaces. We want to remark that all the classes under consideration are characterized by simple restrictions on the admissible links of Σ-spaces. In Section 5 we draw some conclusion and sketch a promising future research aimed at the *integration* of the mathematical structures considered in this paper via an implementation of their uniform normal form representations offered by Σ-spaces.

2 A categorical view for different formalisms

We recall here a few categorical definitions, which allow to recast the usual notion of term over a signature in a more general setting. Moreover, the progressive enrichment of a basic theory with different auxiliary mathematical constructors generates a great variety of different model classes, where the usual notions of relation, partial order, partition, and many other can be represented.

Definition 1 (Signatures). *A* many-sorted hyper-signature Σ *over a set* S_Σ *of sorts is a family* $\{\Sigma_{\omega,\omega'}\}_{\omega,\omega' \in S_\Sigma^*}$ *of sets of operators. If* S_Σ *is a singleton, we denote the hyper-signature* Σ *by the family* $\{\Sigma_{n,m}\}_{n,m \in \mathbb{N}}$.

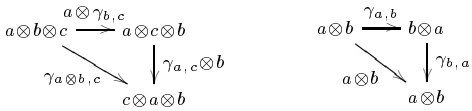
We usually omit the prefix "hyper". When it is clear from the context that a set S is the underlying set of sorts of a signature Σ, we drop the subscript $_\Sigma$.

Definition 2 (Graphs). *A* graph G *is a 4-tuple* $(O_G, A_G, \partial_0, \partial_1)$*, where* O_G *is the set of* objects*,* A_G *is the set of* arrows*, and* $\partial_0, \partial_1 : A_G \longrightarrow O_G$ *are functions, called respectively* source *and* target*. We use the standard notation* $f : a \longrightarrow b$ *to denote an arrow* f *with source* a *and target* b*. A graph* G *is* with pairing *if* O_G *is a monoid.*

A many-sorted signature Σ over S is a graph with pairing G_Σ, where its objects are strings on S ($O_{G_\Sigma} = S^*$, string concatenation $_\cdot_$ yields the monoidal operator, and the empty string ϵ is the neutral element), and its edges are the operators of the signature (i.e., $f : \omega \longrightarrow \omega' \in A_{G_\Sigma}$ iff $f \in \Sigma_{\omega,\omega'}$).

A chain of structural enrichments enhances the expressiveness of different classes of models, e.g., the usual algebraic notion of terms over a signature. However, we are interested in weaker theories, where name sharing finds a natural embedding. The first enrichment is common to all the formalisms we will consider, and introduces the sequential and parallel compositions between arrows together with all the arrows necessary for arbitrary permutations of objects (corresponding to the swappings of wires in the wire and box presentation). Then, the models of the resulting *symmetric theory* of Σ are just suitable symmetric monoidal categories [21] (i.e., also equipped with the Σ-structure).

Definition 3 (Symmetric Monoidal Categories). *A* monoidal category *is a triple* $\langle \mathcal{C}, _ \otimes _, e \rangle$*, where* $\mathcal{C}$ *is the underlying category,* $_ \otimes _ : \mathcal{C} \times \mathcal{C} \longrightarrow \mathcal{C}$ *is a functor satisfying the associative law* $(t_1 \otimes t_2) \otimes t_3 = t_1 \otimes (t_2 \otimes t_3)$*, and* e *is an object of* $\mathcal{C}$ *satisfying the identity law* $t \otimes e = t = e \otimes t$*, for all arrows* $t, t_1, t_2, t_3 \in \mathcal{C}$*. A* symmetric monoidal category *is a 4-tuple* $\langle \mathcal{C}, _ \otimes _, e, \gamma \rangle$*, where* $\langle \mathcal{C}, _ \otimes _, e \rangle$ *is a monoidal category, and* $\gamma : _1 \otimes _2 \Rightarrow _2 \otimes _1 : \mathcal{C} \times \mathcal{C} \longrightarrow \mathcal{C}$ *is a natural transformation satisfying the coherence axioms*



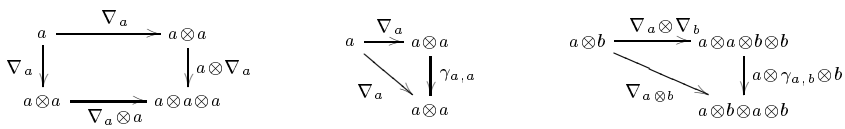
A functor $F : \mathcal{C} \longrightarrow \mathcal{C}'$ *between two (symmetric) monoidal categories is called* monoidal *if* $F(t_1 \otimes t_2) = F(t_1) \otimes' F(t_2)$*, and* $F(e) = e'$*; it is* symmetric *if* $F(\gamma_{a,b}) = \gamma'_{F(a),F(b)}$*. We denote by* **SMCat** *the category of symmetric functors.*

Among the uses of symmetric monoidal categories as a semantics framework, we recall the characterization of *concatenable processes* for Petri nets in [10], and the description of a *basic network algebra* for data-flows in [3, 34].

2.1 Enriching the Monoidal Structure

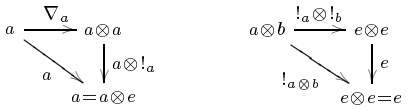
The constructive definition of algebraic theories [20] as enriched monoidal categories dates back to the mid-Seventies [17, 27], even if it has received a new stream of attention in these days. In our opinion, it separates very nicely the auxiliary structure from the Σ-structure (better than the ordinary description involving the meta-operation of substitution). Moreover, the naturality axioms for *duplicator* and *discharger* express a controlled form of data-sharing and data-garbaging. If these axioms are missing, then the corresponding theory, called *gs-monoidal*, is the natural framework for the representation of *term-graphs* rather than terms, as shown in [8].

Definition 4 (Share and GS-Monoidal Categories). *A* share category *is a 5-tuple* $\langle \mathcal{C}, _ \otimes _, e, \gamma, \nabla \rangle$*, where* $\langle \mathcal{C}, _ \otimes _, e, \gamma \rangle$ *is a symmetric monoidal category and* $\nabla : 1 \Rightarrow _1 \otimes _1 : \mathcal{C} \longrightarrow \mathcal{C}$ *is a transformation such that* $\nabla_e = e$*, and satisfying the coherence axioms (the first two diagrams express a sort of "output" associativity and commutativity for the duplicator, and the third diagram tells how the duplicator of a composed object* $a \otimes b$ *can be obtained by composing the duplicators of the subcomponents)*



A share functor $F : \mathcal{C} \longrightarrow \mathcal{C}'$ *between two share categories is a symmetric functor such that* $F(\nabla_a) = \nabla'_{F(a)}$*. We denote by* **ShCat** *the category of share functors.*

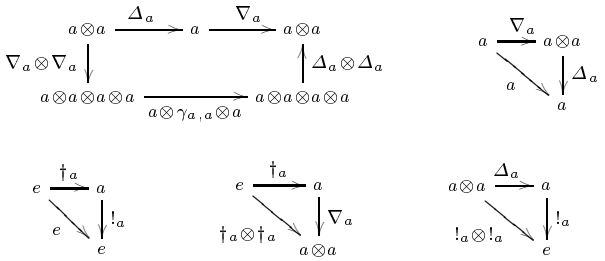
A gs-monoidal category *is a 6-tuple* $\langle \mathcal{C}, _\otimes_, e, \gamma, \nabla, ! \rangle$*, where* $\langle \mathcal{C}, _\otimes_, e, \gamma, \nabla \rangle$ *is a share category, and* $! : 1 \Rightarrow e : \mathcal{C} \longrightarrow \mathcal{C}$ *is a natural transformation such that* $!_e = e$ *and satisfying the coherence axioms (the first diagram says that creating two links to* a *and then discharging one of them just yields the identity on* a*, the second diagram expresses the monoidality of the discharger)*



A gs-monoidal functor $F : \mathcal{C} \longrightarrow \mathcal{C}'$ *is a share functor such that* $F(!_a) = !'_{F(a)}$*. We denote by* **GSCat** *the category of gs-monoidal functors.*

Interesting applications often require the presence of the categorical opposite of duplicators, which may be used to express a sort of *data-matching*. Analogously, co-dischargers are introduced to represent the explicit *creation* of data. Several combinations are then possible, where only some of the operators are considered, and their mixed compositions are differently axiomatized, ranging from the *match-share categories* of [14] to the *dgs-monoidal categories* of [13, 18]. Here we just sketch a survey of the categorical framework, and briefly comment their role in the literature and the main differences between similar models. We start with what we call a *r-monoidal category*: One of the various extensions, albeit with a different name, proposed in [5, 33, 34].

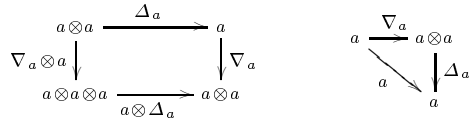
Definition 5 (R-Monoidal Categories). *A* r-monoidal category *is an 8-tuple* $\langle \mathcal{C}, _\otimes_, e, \gamma, \nabla, !, \Delta, \dagger \rangle$ *such that* $\langle \mathcal{C}, _\otimes_, e, \gamma, \nabla, ! \rangle$ *and* $\langle \mathcal{C}^{op}, _\otimes^{op}_, e, \gamma^{op}, \Delta^{op}, \dagger^{op} \rangle$ *are both gs-monoidal categories, and satisfying the additional coherence axioms (expressing the interplay between the gs and co-gs structures)*



A r-monoidal functor $F : \mathcal{C} \longrightarrow \mathcal{C}'$ *is a gs-monoidal functor such that also* F^{op} *is so. We denote by* **RMCat** *the category of r-monoidal functors.*

The axioms we considered naturally embed the properties of relations, and the (partial) algebraic structure of r-monoidal categories yields a useful mathematical tool for their representation [5]. A stronger version of the axiom involving the composite $\Delta;\nabla$ is the basis for a different family of structures. In a certain sense, the stronger axiom establishes that duplicators and co-duplicators embed a sort of transitive and symmetric closure of the relation, i.e., it does not matter how two objects are connected, but just the fact that they are connected.

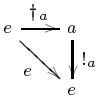
Definition 6 (Match-Share Categories). *A* match-share category *is a 6-tuple* $\langle \mathcal{C}, _ \otimes _, e, \gamma, \nabla, \Delta \rangle$ *such that* $\langle \mathcal{C}, _ \otimes _, e, \gamma, \nabla \rangle$ *and* $\langle \mathcal{C}^{op}, _ \otimes^{op} _, e, \gamma^{op}, \Delta^{op} \rangle$ *are both share categories, and satisfying the additional coherence axioms*



A match-share functor $F : \mathcal{C} \longrightarrow \mathcal{C}'$ *is a share functor such that also* F^{op} *is so. We denote by* **MShCat** *the category of match-share functors.*

Match-share categories have been introduced in [14], and used to embed the algebraic properties of *processes* for *contextual nets* [25]. They are the basis for a class of categories where suitable models of partition-based structures can live.

Definition 7 (Part-Monoidal Categories). *A* part-monoidal category *is an 8-tuple* $\langle \mathcal{C}, _ \otimes _, e, \gamma, \nabla, !, \Delta, \dagger \rangle$ *such that both* $\langle \mathcal{C}, _ \otimes _, e, \gamma, \nabla, ! \rangle$ *and* $\langle \mathcal{C}^{op}, _ \otimes^{op} _, e, \gamma^{op}, \Delta^{op}, \dagger^{op} \rangle$ *are gs-monoidal categories,* $\langle \mathcal{C}, _ \otimes _, e, \gamma, \nabla, \Delta \rangle$ *is a match-share category, and satisfying the additional coherence axiom*



A part-monoidal functor $F : \mathcal{C} \longrightarrow \mathcal{C}'$ *is a gs-monoidal functor such that also* F^{op} *is so. We denote by* **PartMCat** *the category of part-monoidal functors.*

Differently from the previous structures, part-monoidal categories, as far as we know, has never been explicitly analyzed in the literature. The leading idea is that a generic arrow from $a_1 \otimes \cdots \otimes a_n$ to $b_1 \otimes \cdots \otimes b_m$ (where the a_i's and the b_i's are "basic" objects) represents some kind of partition of $\{a_1, ..., a_n, b_1, ..., b_m\}$. For example, the axiom $\Delta_a ; !_a = !_a \otimes !_a$ of r-monoidal categories does not hold for partitions, because in the partition $\Delta_a ; !_a$ both a sources belong to the same class, whereas in partition $!_a \otimes !_a$ they belong to disjoint classes.

3 The general model: Σ-spaces

We introduce now a concrete representation for the several formalisms discussed in the previous sections. It can be thought of as a normal form presentation of the less formal wire and box diagrams illustrated in the introduction. Basically, we split the operative components from the logical connectivity of a diagram. The typical operative components of a distributed system are the *input* and *output interfaces*, and the *basic functional modules* (n-to-m transforming black-boxes which give the building blocks of the wire and box notation). The way these components can interact constitutes the logical part of the system. Suitable *link-channels* can faithfully express this connectivity-related aspect. As a matter of notation, we write $z : s$ to say that the variable z has sort s.

Definition 8 (Σ-Spaces). *Let Σ be a signature. A* Σ-assignment *over a bipartite set $\mathcal{Z} = \mathcal{X} \uplus \mathcal{Y}$ ($\mathcal{X}$ is called the set of* names*, and $\mathcal{Y}$ the set of* results*) of typed variables is any of the following* sentences*, where the x's, the y's and the z's range over $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$ respectively.*

Generator: $x_1...x_n \overset{f}{\rightarrowtail\!\!\!\!\!\rightarrow} y_1...y_m$ *, where $x_i : s_i$ for $i = 1...n$, $y_j : s'_j$ for $j = 1...m$, and $f \in \Sigma_{\omega,\omega'}$ with $\omega = s_1...s_n$, and $\omega' = s'_1...s'_m$,*
Link: $z_1 \rightarrowtail\!\!\!\!\!\rightarrow z_2$ *, where $z_1 : s$ and $z_2 : s$,*
Input: $\Box \rightarrowtail\!\!\!\!\!\rightarrow y_1...y_n$*, where $n \geq 0$,*
Output: $x_1...x_n \rightarrowtail\!\!\!\!\!\rightarrow \Box$*, where $n \geq 0$.*
We say that a variable z is used *(respectively* assigned*) if it appears in the left-hand (respectively right-hand) side of a sentence.*

A distributed space *over Σ is a set G of Σ-*assignments *such that*

1. *G contains exactly one input sentence $\Box \rightarrowtail\!\!\!\!\!\rightarrow \alpha_{in}(G)$, denoted by $in(G)$,*
2. *G contains exactly one output sentence $\alpha_{out}(G) \rightarrowtail\!\!\!\!\!\rightarrow \Box$, denoted by $out(G)$,*
3. *all the variables in $in(G)$, $out(G)$ and $gen(G) = \{\ x_1...x_n \overset{f}{\rightarrowtail\!\!\!\!\!\rightarrow} y_1...y_m\ \in G\}$ are different,*
4. *all the variables in $link(G) = \{\ z_1 \rightarrowtail\!\!\!\!\!\rightarrow z_2\ \in G\}$ occur also in either $gen(G)$, $in(G)$ or $out(G)$.*

Given a distributed space G, its flow relation *F_G is the pre-order induced over variables by (the reflexive and transitive closure of) the set of sentences; its* link-flow relation *LF_G is the pre-order induced over variables by the set of links.*

*We call Σ-*spaces *the equivalence classes of distributed spaces over Σ up to α-conversion. Abusing the notation, we denote a generic Σ-space by the same symbols of the distributed spaces in the equivalence class it represents.*

Σ-spaces yield a monoidal category. Indeed, let G be any Σ-space, and let $st : V^* \longrightarrow S^*$ be the function mapping a list of typed variables into the corresponding list of types, e.g. $st(z_1...z_n) = s_1...s_n$ if $z_i : s_i$ for $i = 1...n$. The objects are the elements of S^* (the free monoid over the set of sorts S), while each G is viewed as an arrow $G : st(\alpha_{in}(G)) \longrightarrow st(\alpha_{out}(G))$.

The parallel composition of two Σ-space G_1 and G_2 is always defined, yielding as a result the Σ-space $G_1 \otimes G_2$ that can be constructed as follows: Choose two distributed spaces in the classes of G_1 and G_2 such that their underlying sets of variables are disjoint (we can assume without loss of generality that G_1 and G_2 are already variable-disjoint), then take the following distributed space

$$\begin{aligned}
\alpha_{in}(G_1 \otimes G_2) &= \alpha_{in}(G_1) \cdot \alpha_{in}(G_2),\\
\alpha_{out}(G_1 \otimes G_2) &= \alpha_{out}(G_1) \cdot \alpha_{out}(G_2),\\
gen(G_1 \otimes G_2) &= gen(G_1) \cup gen(G_2),\\
link(G_1 \otimes G_2) &= link(G_1) \cup link(G_2),
\end{aligned}$$

where $_ \cdot _$ denotes ordinary string concatenation.

Proposition 1. *Let G_1, G_2 be two Σ-spaces. Then $G_1 \otimes G_2$ is a Σ-space.*

The empty Σ-space $G_\epsilon = \{ \Box \rightarrowtail\!\!\!\!\!\rightarrow \epsilon,\ \epsilon \rightarrowtail\!\!\!\!\!\rightarrow \Box\}$, where ϵ denotes the empty list of variables, is the unit element for parallel composition.

Proposition 2. *Let G be a Σ-space. Then $G \otimes G_\epsilon = G = G_\epsilon \otimes G$.*

The sequential composition of G_1 and G_2 is defined if and only if $st(\alpha_{out}(G_1)) = st(\alpha_{in}(G_2))$. As before, we assume G_1 and G_2 to be variable-disjoint and take the distributed space $G_1; G_2$ defined by

$$\begin{aligned}
\alpha_{in}(G_1; G_2) &= \alpha_{in}(G_1),\\
\alpha_{out}(G_1; G_2) &= \alpha_{out}(G_2),\\
gen(G_1; G_2) &= gen(G_1) \cup gen(G_2),\\
link(G_1; G_2) &= [link(G_1)/\alpha_{out}(G_1)][link(G_2)/\alpha_{in}(G_2)],
\end{aligned}$$

where the composition of links is defined as

$$[L_1/X][L_2/Y] = I_1 \cup I_2 \cup F_{fw} \cup F_{bw} \cup D_1 \cup D_2$$

(but in all the cases we consider, except for *partitions* in Section 4.5 and Section 4.6, the three sets F_{bw}, D_1, and D_2 are empty), and if we denote by $X[i]$ the i-th variable of the list X, the formal definition of each subset of links is

$$\begin{aligned}
I_1 &= \{ z_1 \rightarrowtail\!\!\!\!\!\rightarrow z_1' \ \in L_1\},\\
I_2 &= \{ z_2 \rightarrowtail\!\!\!\!\!\rightarrow z_2' \ \in L_2\},\\
F_{fw} &= \{ z_1 \rightarrowtail\!\!\!\!\!\rightarrow z_2 \ \mid\ z_1 \rightarrowtail\!\!\!\!\!\rightarrow X[i] \ \in L_1,\ Y[i] \rightarrowtail\!\!\!\!\!\rightarrow z_2 \ \in L_2\},\\
F_{bw} &= \{ z_2 \rightarrowtail\!\!\!\!\!\rightarrow z_1 \ \mid\ z_2 \rightarrowtail\!\!\!\!\!\rightarrow Y[i] \ \in L_2,\ X[i] \rightarrowtail\!\!\!\!\!\rightarrow z_1 \ \in L_1\},\\
D_1 &= \{ z_1 \rightarrowtail\!\!\!\!\!\rightarrow z_1' \ \mid\ z_1 \rightarrowtail\!\!\!\!\!\rightarrow X[i]\ ,\ X[j] \rightarrowtail\!\!\!\!\!\rightarrow z_1' \ \in L_1,\ Y[i] \rightarrowtail\!\!\!\!\!\rightarrow Y[j] \ \in L_2\}\\
D_2 &= \{ z_2 \rightarrowtail\!\!\!\!\!\rightarrow z_2' \ \mid\ z_2 \rightarrowtail\!\!\!\!\!\rightarrow Y[i]\ ,\ Y[j] \rightarrowtail\!\!\!\!\!\rightarrow z_2' \ \in L_2,\ X[i] \rightarrowtail\!\!\!\!\!\rightarrow X[j] \ \in L_1\}.
\end{aligned}$$

with $z_1, z_1' \notin X$ and $z_2, z_2' \notin Y$.

Intuitively, variables in $\alpha_{out}(G_1)$ and $\alpha_{in}(G_2)$ are removed from the composition $G_1; G_2$, but their ingoing and outgoing links are propagated to the remaining variables, with respect to the matching $\alpha_{out}(G_1)[i] \leftrightarrow \alpha_{in}(G_2)[i]$, for $i = 1...n$, of the variables in the "merged" interfaces of G_1 and G_2. Therefore, $link(G_1; G_2)$ contains: (1) the *internal* links I_1 of G_1, and I_2 of G_2, i.e. those links not involving removed variables, (2) the *forward* and *backward fusion* links F_{fw} and F_{bw}, and (3) the *forward* and *backward derived* links D_1 and D_2, propagating links involving only removed variables. However, as said before, most of the classes of Σ-spaces that we consider are acyclic, and define an intuitive *information flow* from left ($in(G)$) to right ($out(G)$): in this case, $F_{bw} \cup D_1 \cup D_2 = \emptyset$.

Proposition 3. *Let G_1, G_2 be two Σ-spaces, such that $st(\alpha_{out}(G_1)) = st(\alpha_{in}(G_2))$. Then $G_1; G_2$ is a Σ-space.*

For each $\omega \in S^*$, the Σ-space G_ω such that $st(\alpha_{in}(G)) = st(\alpha_{out}(G))$, $gen(G) = \emptyset$ and $link(G) = \{\ \alpha_{in}(G)[i] \rightarrowtail\!\!\!\!\!\rightarrow \alpha_{out}(G)[i]\ \mid i = 1...|\omega|\}$, behaves as the identity with respect to the sequential composition.

Proposition 4. *Let G be a Σ-space. Then $G; G_{st(\alpha_{out}(G))} = G_{st(\alpha_{in}(G))}; G = G$.*

Also the other properties of monoidal categories (e.g., $G_\omega \otimes G_{\omega'} = G_{\omega \cdot \omega'}$, and $(G_1; G_1') \otimes (G_2; G_2') = (G_1 \otimes G_2); (G_1' \otimes G_2')$, giving the functoriality of the tensor product $_ \otimes _$) are trivially satisfied by the definitions.

4 A Unified View For Nets, Relations and Partitions

In this section we show that all the models presented in Section 2 can be characterized as suitable Σ-spaces, simply imposing different requirements over the set of links. Moreover, such properties are preserved by the parallel and sequential composition as defined in Section 3, so that those spaces can be considered as concrete representations of their categorical counterparts.

4.1 Symmetric Monoidal

We begin by providing the class of Σ-spaces which characterize symmetric monoidal categories: its elements must satisfy a tight requirement over links.

Definition 9 (Symmetric Σ-space). *A Σ-space G is called* symmetric *if and only if each variable is used and assigned exactly once in G, and the flow relation induced by the assignment in G is acyclic.*

Or, in other words, the flow relation is actually a *partial order*.

Proposition 5. *If G is a symmetric Σ-space, then*

- *the only kind of link sentences allowed consists of sentences of the form $y \rightarrowtail\!\!\!\!\!\rightarrow x$ where y is a result and x is a name,*
- *no result can be discharged in G,*
- *no name can be created in G,*
- *parallel and sequential composition of symmetric Σ-spaces yield symmetric Σ-spaces.*

Note that the restrictions always imply that $F_{bw} = D_1 = D_2 = \emptyset$, whenever two symmetric spaces are sequentially composed. To show that each symmetric Σ-space G defines a concrete symmetric monoidal category, we have to show what the symmetries are. For each pair of string $\omega, \omega' \in S^*$, with $|\omega| = n$, and $|\omega|' = m$, we define the symmetric Σ-space $G_{\omega,\omega'}$ as follows: Let Y and X be two lists of (pairwise disjoint) names such that $|Y| = |X| = n + m$, $Y[i] : (\omega \cdot \omega')[i]$, and $X[i] : (\omega' \cdot \omega)[i]$ for $i = 1...n + m$, then

$$G_{\omega,\omega'} = \{\ \Box \rightarrowtail\!\!\!\!\!\rightarrow\ Y,\ X \rightarrowtail\!\!\!\!\!\rightarrow\ \Box\} \cup \{\ Y[i] \rightarrowtail\!\!\!\!\!\rightarrow X[i+m]\ \mid i = 1...n\} \cup$$
$$\cup\ \{\ Y[i] \rightarrowtail\!\!\!\!\!\rightarrow X[i-n]\ \mid i = n+1...n+m\}.$$

Theorem 1. *Symmetric Σ-spaces are the arrows of a concrete symmetric monoidal category, which is (isomorphic to) the one freely generated by Σ.*

Proof sketch. From the last property of Proposition 5, we have just to show that the family of Σ-spaces $\{G_{\omega,\omega'}\}_{\omega,\omega' \in S^*}$ is a natural isomorphism from $_1 \otimes _2$ to $_2 \otimes _1$, and verifies the coherence axioms of Def. 3. This can be easily done by exploiting the definition of $G_{\omega,\omega'}$ and applying the definition of parallel and sequential composition. The initiality result relies on previous characterization results for symmetric monoidal categories as suitable *Petri processes* [31, 30], to which our spaces are equivalent. □

4.2 GS-Monoidal

As illustrated in Section 2, gs-monoidal categories are symmetric monoidal categories enriched with suitable transformations for copying and discharging information, which lack the naturality axiom. In our setting, this enrichment reflects into a relaxation of the previous constraints over symmetric Σ-spaces.

Definition 10 (GS-Monoidal Σ-space). *A Σ-space G is called* gs-monoidal *if and only if each variable is assigned exactly once in G, and the flow relation induced by the assignment in G is acyclic.*

Proposition 6. *If G is a gs-monoidal Σ-space, then*

- *the only kind of link sentences allowed consists of sentences of the form $y \rightarrowtail\!\!\!\!\!\rightarrow x$ where y is a result and x is a name,*
- *no name can be created in G,*
- *parallel and sequential composition of gs-monoidal Σ-spaces yield gs-monoidal Σ-spaces.*

The characterization of duplicators and dischargers is intuitive. For each string $\omega \in S^*$, with $|\omega| = n$, we define the gs-monoidal Σ-spaces G^2_ω, and G^0_ω as follows: Let Y and X be list of names such that $|Y| = n$, $|X| = 2n$, $Y[i] : \omega[i]$, $X[i] : \omega[i]$, and $X[i+n] : \omega[i]$, for $i = 1...n$. Then

$$
\begin{aligned}
G^0_\omega &= \{\ \Box \rightarrowtail\!\!\!\!\!\rightarrow\ Y,\ \epsilon \rightarrowtail\!\!\!\!\!\rightarrow\ \Box\},\\
G^2_\omega &= \{\ \Box \rightarrowtail\!\!\!\!\!\rightarrow\ Y,\ X \rightarrowtail\!\!\!\!\!\rightarrow\ \Box\} \cup \{\ Y[i] \rightarrowtail\!\!\!\!\!\rightarrow X[i]\ \mid i = 1...n\} \cup\\
&\quad \cup \{\ Y[i] \rightarrowtail\!\!\!\!\!\rightarrow X[i+n]\ \mid i = 1...n\},
\end{aligned}
$$

where we recall that ϵ denotes the empty list of variables.

Theorem 2. *GS-Monoidal Σ-spaces are the arrows of a concrete gs-monoidal category, which is (isomorphic to) the one freely generated by Σ.*

Proof sketch. From the last property of Proposition 6, we have just to show that the families of Σ-spaces $\{G^2_\omega\}_{\omega\in S^*}$, and $\{G^0_\omega\}_{\omega\in S^*}$ are transformation from $_1$ to $_1 \otimes _1$, and to G_ϵ, respectively, verifying the coherence axioms of Def. 4. This can be easily done exploiting the definition of G^2_ω, and G^0_ω. As an important remark, it is trivial to verify that the naturality axioms are not satisfied by $\{G^2_\omega\}_{\omega\in S^*}$, and $\{G^0_\omega\}_{\omega\in S^*}$. To show the initiality of our model we rely on the results of [8], since gs-monoidal Σ-spaces offer a concrete mathematical structure corresponding to a normalized representation for *gs-graphs*. □

The main result of [8] states that the free gs-monoidal category over a (ordinary) signature is isomorphic to the class of (ranked) *term graphs* labeled over it. Such a property is exploited in [7] to give an inductive, algebraic account of term graph *rewriting*. It is in this setting that we recover the intuitive interpretation of copying and discharging as suitable operations over graphical structures.

Also the *open graphs* of [26] form a free gs-monoidal category: The one generated by the one-sorted signature Σ such that $\Sigma_{h,k} = \emptyset$ if $k \neq 0$, and $\Sigma_{h,0} = L_h$ for $h \in \mathbb{N}$, where $L = \{L_h\}_{h\in\mathbb{N}}$ is the set of labels for the edges of the graph. Therefore, they find a normalized presentation in terms of Σ-spaces.

4.3 Relations

Due to the presence of both co-duplicators and co-dischargers in the relational model, we have no restriction on the number of ingoing and outgoing links in the corresponding version of Σ-spaces. The weaker constraint considered here just involves the global structure of the link sentences.

Definition 11. *A Σ-space G is called* relational *if and only if its link sentences have the form $y \rightarrowtail\!\!\!\!\!\rightarrow x$ where y is a result and x is a name, and the flow relation induced by the assignments in a relational Σ-space G is acyclic.*

For example, in a relational Σ-space, each variable can be assigned and used as many times as necessary.

Proposition 7. *The parallel and sequential composition of relational Σ-spaces yield relational Σ-spaces.*

For each string $\omega \in S^*$, with $|\omega| = n$, we define the relational Σ-spaces $\overline{G}^2_\omega$, and $\overline{G}^0_\omega$ as follows: Let Y and X be list of names such that $|Y| = 2n$, $|X| = n$, $Y[i] : \omega[i]$, $Y[i+n] : \omega[i]$, and $X[i] : \omega[i]$, for $i = 1...n$. Then

$$\begin{aligned}
\overline{G}^0_\omega &= \{\ \square \rightarrowtail\!\!\!\!\!\rightarrow\ \epsilon,\ X \rightarrowtail\!\!\!\!\!\rightarrow\ \square\},\\
\overline{G}^2_\omega &= \{\ \square \rightarrowtail\!\!\!\!\!\rightarrow\ Y,\ X \rightarrowtail\!\!\!\!\!\rightarrow\ \square\} \cup \{\ Y[i] \rightarrowtail\!\!\!\!\!\rightarrow X[i]\ \mid i = 1...n\} \cup\\
&\quad \cup\ \{\ Y[i+n] \rightarrowtail\!\!\!\!\!\rightarrow X[i]\ \mid i = 1...n\},
\end{aligned}$$

Families $\{\overline{G}^2_\omega\}_{\omega\in S^*}$, and $\{\overline{G}^0_\omega\}_{\omega\in S^*}$ are the co-duplicators and the co-dischargers.

Theorem 3. *Relational Σ-spaces are the arrows of a concrete r-monoidal category, which is (isomorphic to) the one freely generated by Σ.*

Therefore, relational Σ-spaces define an initial relational model for Σ.

Theorem 4. *Given any two strings $\omega, \omega' \in S^*$, the class of relational Σ-spaces G from ω to ω' such that $gen(G) = \emptyset$ is isomorphic to the class of possible type-preserving relations between the components of ω and those of ω'.*

The proof relies on Theorem 3 and on the results of [5, 34].

4.4 Partial Orders

If Σ is a one-sorted signature that contains only unary operators, then we may use relational Σ-spaces for the representation of labeled partial orders.

Definition 12. *A relational Σ-space G is called po_Σ-space if $S_\Sigma = \{s\}$, and $\Sigma = \Sigma_{s,s}$.*

Let $(P, \sqsubseteq, \ell, A)$, where $\ell : P \longrightarrow A$, be a generic partial order labeled over the set A, and let us consider the Σ-spaces over the signature $\Sigma_A = \{a : 1 \longrightarrow 1 \mid a \in A\}$. In a similar fashion to the proposal of [14], the basic ingredients are the Σ_A-spaces in the family $\{G_a\}_{a \in A}$, where

$$G_a = \{\ \Box \rightarrowtail\!\!\!\!\!\rightarrow\ y_1,\ x \overset{a}{\rightarrowtail\!\!\!\!\!\rightarrow} y\ ,\ x_1 \rightarrowtail\!\!\!\!\!\rightarrow\ \Box,\ y_1 \rightarrowtail\!\!\!\!\!\rightarrow x\ ,\ y \rightarrowtail\!\!\!\!\!\rightarrow x_1\ ,\ y_1 \rightarrowtail\!\!\!\!\!\rightarrow x_1\ \}.$$

Intuitively, each Σ-space G_a represents the label a in parallel with an identity. The presence of the links $y_1 \rightarrowtail\!\!\!\!\!\rightarrow x$ and $y_1 \rightarrowtail\!\!\!\!\!\rightarrow x_1$, acting as a duplicator of the input position, together with the presence of the links $y \rightarrowtail\!\!\!\!\!\rightarrow x_1$ and $y_1 \rightarrowtail\!\!\!\!\!\rightarrow x_1$, acting as a match in the output position, creates a sort of implicit transitive closure of identities whenever the sequential composition is applied. As a result, the ordering relation of the partial order, say between e and e', is represented by propagating a copy of the identity of e in parallel with e'. The Σ_A-space associated to P is obtained by composing the G_a's following the intuitive correspondence with the labeled elements of P.

Example 1. The Σ_A-space $G^0; G_a; G^2; (G_b \otimes G_c); \overline{G}^2; G_b; \overline{G}^0$ corresponds to the partial order in Fig. 1. The explicit construction for a sub-term is given in Fig. 2.

Proposition 8. *Let G be a po_{Σ_A}-space. Then the class of arrows in the hom-set $G[\epsilon, \epsilon]$ are in one-to-one correspondence with the partial orders labeled on A.*

We believe that our characterization of partial orders as "closed elements" of an axiomatically defined algebra of relations is new.

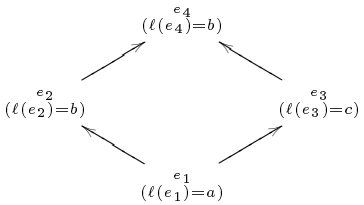


Fig. 1. A partial order $P = \{e_i\}_{i=1\ldots4}$ with labels over the set $A = \{a, b, c\}$.

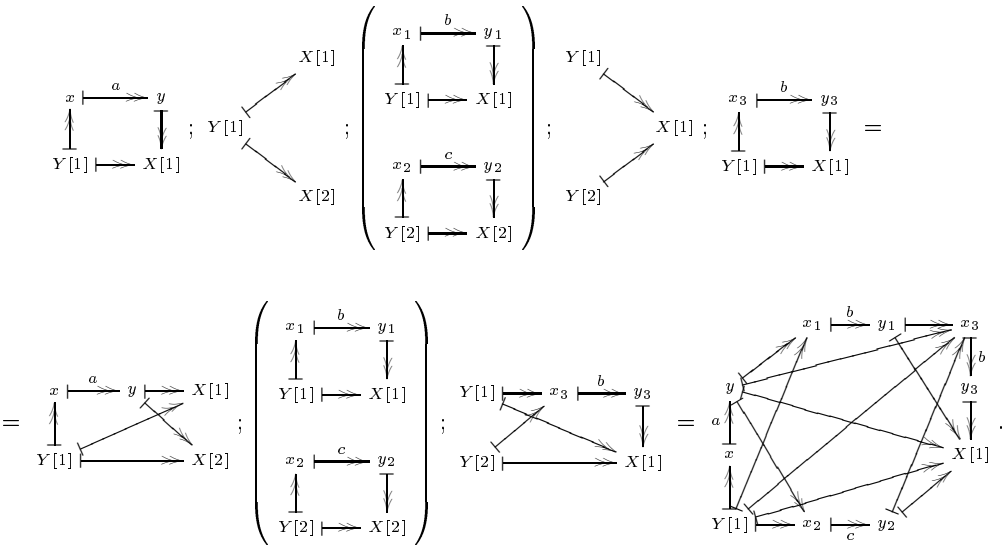


Fig. 2. Step-by-step illustration of $G_a; G^2; (G_b \otimes G_c); \overline{G}^2; G_b$. For simplicity we adopt a self-explanatory vector notation for the input and output variables.

4.5 Partitions

The most complex class of Σ-spaces we consider gives a framework for partitions.

Definition 13. *A Σ-space G is called* partition space *if and only if the link-flow relation is an equivalence relation.*

Proposition 9. *The parallel and sequential composition of partition spaces yield partition spaces as a result.*

Due to the peculiar nature of partition space, symmetries, (co)duplicators, and (co)dischargers still exist, but the constraints of partition spaces force some of them to have a richer structure than the one previously considered. In fact, they can be obtained freely adding link sentences, in order to obtain the minimal equivalence relation.

For each pair of string $\omega, \omega' \in S^*$, with $|\omega| = n$, and $|\omega|' = m$, the partition space $G_{\omega,\omega'}$ (which plays the role of the symmetry) is explicitly described as follows: Let Y and X be list of (pairwise disjoint) names such that $|Y| = |X| = n + m$, $Y[i] : (\omega \cdot \omega')[i]$, and $X[i] : (\omega' \cdot \omega)[i]$ for $i = 1...n + m$. Then

$$\begin{aligned} G_{\omega,\omega'} = &\{ \Box \rightarrowtail Y,\ X \rightarrowtail \Box \} \cup \{ Y[i] \rightarrowtail X[i-n] \ \mid i = n+1...n+m \} \cup \\ &\cup \{ Y[i] \rightarrowtail X[i+m] \ \mid i = 1...n \} \cup \{ X[i+m] \rightarrowtail Y[i] \ \mid i = 1...n \} \cup \\ &\cup \{ X[i-n] \rightarrowtail Y[i] \ \mid i = n+1...n+m \} \end{aligned}$$

where we skipped all the reflexive links, for the sake of readability. For each string $\omega \in S^*$, with $|\omega| = n$, the partition spaces G^2_ω, and G^0_ω are as follows: Let Y and X be list of names such that $|Y| = n$, $|X| = 2n$, $Y[i] : \omega[i]$, $X[i] : \omega[i]$, and $X[i+n] : \omega[i]$, for $i = 1...n$. Then

$$\begin{aligned} G^0_\omega &= \{ \Box \rightarrowtail Y,\ \epsilon \rightarrowtail \Box \}, \\ G^2_\omega &= \{ \Box \rightarrowtail Y,\ X \rightarrowtail \Box \} \cup \{ Y[i] \rightarrowtail X[i] \ \mid i = 1...n \} \cup \\ &\quad \cup \{ Y[i] \rightarrowtail X[i+n] \ \mid i = 1...n \} \cup \{ X[i] \rightarrowtail X[i+n] \ \mid i = 1...n \} \cup \\ &\quad \cup \{ X[i] \rightarrowtail Y[i] \ \mid i = 1...n \} \cup \{ X[i+n] \rightarrowtail Y[i] \ \mid i = 1...n \} \cup \\ &\quad \cup \{ X[i+n] \rightarrowtail X[i] \ \mid i = 1...n \}. \end{aligned}$$

And similarly for their co-version.

Theorem 5. *Partition spaces are the arrows of a concrete part-monoidal category which is isomorphic to the one freely generated from Σ.*

Next correspondence theorem justifies the name part-monoidal.

Theorem 6. *Let G be a relational Σ-space with no generators. Then for each sort s and each $n, m \in \mathbb{N}$ the class of arrows in the hom-set $G[s^n, s^m]$ are in one-to-one correspondence with the partitions over $\{1 \ldots n+m\}$.*

We believe that our algebraic characterization of partitions is new.

4.6 Contextual Nets

As a last case we consider a subclass of partition Σ-spaces, which embeds processes of contextual nets.

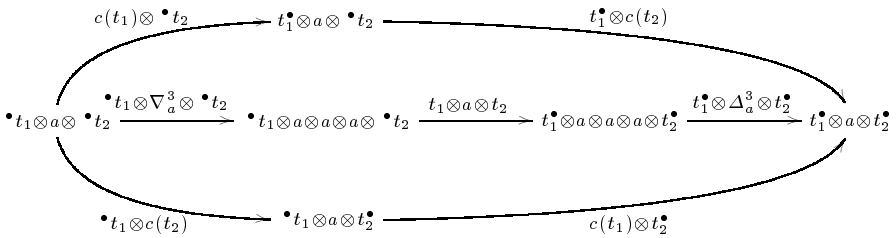
Definition 14. *A partition Σ-space G is called* contextual *if and only if each variable is used and assigned at least once in G.*

Proposition 10. *If G is a contextual Σ-space, then*

- *no result can be discharged in G,*
- *no name can be created in G,*
- *parallel and sequential composition of contextual Σ-spaces yield contextual Σ-spaces.*

Theorem 7. *Contextual Σ-spaces are the arrows of a concrete match-share category, which is (isomorphic to) the one freely generated from Σ.*

Contextual nets extend ordinary Petri nets with *read arcs*, following the paradigm of read-write access to shared resources, where readers are allowed to progress in parallel. The axioms of match-share categories faithfully embed the compositional properties of contextual net processes [14]. In fact, the basic process associated to a transition t (with pre-set ${}^\bullet t$, context $\widehat{t}$ and post-set $t^\bullet$) corresponds to the arrow $c(t) = ({}^\bullet t \otimes \nabla_{\widehat{t}}); (t \otimes \widehat{t}); (t^\bullet \otimes \Delta_{\widehat{t}})$, where $t : {}^\bullet t \otimes \widehat{t} \longrightarrow t^\bullet \otimes \widehat{t}$. Intuitively this means that first a copy of the resources in the context $\widehat{t}$ is created, then the firing of t consumes both its pre-set and a copy of the context, producing its post-set and a *fresh* copy of the context, which is then matched (and thus identified) with the initial copy left. For example, consider two transitions t_1 and t_2 such that $\widehat{t_1} = a = \widehat{t_2}$, then the following diagram commutes, and represents the concurrent access of t_1 and t_2 to the shared context a.



It follows that creating a copy of the context, firing t_1, matching the contexts, creating another copy of the context, firing t_2 and matching the contexts (upper path in the diagram) is equivalent to creating two copies of the context, concurrently firing t_1 and t_2 and matching the three copies of the context left, namely the one produced by t_1, the one produced by t_2 and the initial one (central path), and this is also equivalent to creating a copy of the context, firing t_2, matching the contexts, creating another copy of the context, firing t_1 and matching the contexts (lower path in the diagram).

5 Conclusion

We have investigated many of the numerous algebraic structures related to the "concurrent and distributed systems" representation, which have been proposed in the recent literature. Since all of them admit a presentation in terms of enriched symmetric monoidal categories, we have looked for a unifying concrete framework based on the same concepts. This has led us to a precise characterization of four fundamental aspects of an abstract distributed system: the *input* and *output interfaces*, the *basic functionalities* provided by the system (i.e., the constructors) and the *links* for the connection of subsystems.

As a main result, many of the algebraic approaches found in the literature are precisely characterized in our framework by considering different restrictions on the link sentences. We plan to show how these representation results allow for a uniform translation of these theories into a suitable specification formalism already available for partial algebras [22], as proved for a class of these structures in [23, 4], the main point being the availability of tools supporting executability.

References

1. S. Abramsky, S. Gay, and R. Nagarajan. Interaction categories and the foundations of typed concurrent programming. In *Deductive Program Design*, Nato ASI Series, pages 403–442. Springer Verlag, 1994.
2. Z.M. Ariola and J.W. Klop. Equational term graph rewriting. *Fundamenta Informaticae*, 26:207–240, 1996.
3. J.A. Bergstra and G. Stefanescu. Network algebra for synchronous and asynchronous dataflow. Technical Report Logic Group Preprint Series n.122, Department of Philosophy, University of Utrecht, 1994.
4. R. Bruni, J. Meseguer, and U. Montanari. Process and term tile logic. Technical Report SRI-CSL-98-06, SRI International, 1998.
5. V.A. Cazanescu and G. Stefanescu. Classes of finite relations as initial abstract data types I. *Discrete Mathematics*, 90:233–265, 1991.
6. V.E. Cazanescu and G. Stefanescu. Towards a new algebraic foundation of flowchart scheme theory. *Fundamenta Informaticae*, 13:171–210, 1990.
7. A. Corradini and F. Gadducci. A 2-categorical presentation of term graph rewriting. In *Category Theory and Computer Science*, volume 1290 of *LNCS*. Springer Verlag, 1997.
8. A. Corradini and F. Gadducci. An algebraic presentation of term graphs, via gs-monoidal categories. *Applied Categorical Structures*, 1998. To appear.
9. A. Corradini and F. Gadducci. Functorial semantics for multi-algebras. Presented at WADT'98, 1998.
10. P. Degano, J. Meseguer, and U. Montanari. Axiomatizing the algebra of net computations and processes. *Acta Informatica*, 33:641–647, 1996.
11. G. Ferrari and U. Montanari. A tile-based coordination view of the asynchronous π-calculus. In *Mathematical Foundations of Computer Science*, volume 1295 of *LNCS*, pages 52–70. Springer Verlag, 1997.
12. G. Ferrari and U. Montanari. Tiles for concurrent and located calculi. In C. Palamidessi and J. Parrow, editors, *Expressiveness in Concurrency*, volume 7 of *Electronic Notes in Computer Science*. Elsevier Sciences, 1997.

13. F. Gadducci and R. Heckel. An inductive view of graph transformation. In F. Parisi-Presicce, editor, *Recent Trends in Algebraic Development Techniques*, volume 1376 of *LNCS*, pages 219–233. Springer Verlag, 1998.
14. F. Gadducci and U. Montanari. Axioms for contextual net processes. In *Automata, Languages and Programming*, volume 1443 of *LNCS*, pages 296–308. Springer Verlag, 1998.
15. F. Gadducci and U. Montanari. The tile model. In G. Plotkin, C. Stirling, and M. Tofte, editors, *Proof, Language and Interaction: Essays in Honour of Robin Milner*. MIT Press, 1999. to appear.
16. M. Hasegawa. Recursion from cyclic sharing: Traced monoidal categories and models of cyclic lambda-calculus. In *Typed Lambda Calculi and Applications*, volume 1210 of *LNCS*, pages 196–213. Springer Verlag, 1997.
17. H.-J. Hoenke. On partial recursive definitions and programs. In M. Karpinski, editor, *Fundamentals of Computations Theory*, volume 56 of *LNCS*, pages 260–274. Springer Verlag, 1977.
18. P. Katis, N. Sabadini, and R.F.C. Walters. Bicategories of processes. *Journal of Pure and Applied Algebra*, 115:141–178, 1997.
19. P. Katis, N. Sabadini, and R.F.C. Walters. SPAN(Graph): A categorical algebra of transition systems. In M. Johnson, editor, *Algebraic Methodology and Software Technology*, volume 1349 of *LNCS*, pages 307–321. Springer Verlag, 1997.
20. F.W. Lawvere. Functorial semantics of algebraic theories. *Proc. National Academy of Science*, 50:869–872, 1963.
21. S. Mac Lane. *Categories for the working mathematician*. Springer Verlag, 1971.
22. J. Meseguer. Membership algebra as a logical framework for equational specification. In F. Parisi-Presicce, editor, *Recent Trends in Algebraic Development Techniques*, volume 1376 of *LNCS*. Springer Verlag, 1998.
23. J. Meseguer and U. Montanari. Mapping tile logic into rewriting logic. In F. Parisi-Presicce, editor, *Recent Trends in Algebraic Development Techniques*, volume 1376 of *LNCS*. Springer Verlag, 1998.
24. R. Milner. Calculi for interaction. *Acta Informatica*, 33:707–737, 1996.
25. U. Montanari and F. Rossi. Contextual nets. *Acta Informatica*, 32:545–596, 1995.
26. U. Montanari and F. Rossi. Graph rewriting, constraint solving and tiles for coordinating distributed systems. *Applied Categorical Structures*, 1998. To appear.
27. M. Pfender. Universal algebra in s-monoidal categories. Technical Report 95–22, University of Munich - Department of Mathematics, 1974.
28. J. Power and E. Robinson. Premonoidal categories and notions of computation. *Mathematical Structures in Computer Science*, 7:453–468, 1998.
29. E. Robinson and G. Rosolini. Categories of partial maps. *Information and Computation*, 79:95–130, 1988.
30. V. Sassone. *On the Semantics of Petri Nets: Processes, Unfolding and Infinite Computations*. PhD thesis, University of Pisa - Department of Computer Science, 1994.
31. V. Sassone. An axiomatization of the algebra of Petri net concatenable processes. *Theoret. Comput. Sci.*, 170:277–296, 1996.
32. M.R. Sleep, M.J. Plasmeijer, and M.C. van Eekelen, editors. *Term Graph Rewriting: Theory and Practice*. Wiley, London, 1993.
33. G. Stefanescu. On flowchart theories: Part II. The nondeterministic case. *Theoret. Comput. Sci.*, 52:307–340, 1987.
34. G. Stefanescu. Algebra of flownomials. Technical Report SFB-Bericht 342/16/94 A, Technical University of München, Institut für Informatik, 1994.