A 2-category View for Double Categories with Shared Structure

Roberto Bruni [a], José Meseguer [b], and Ugo Montanari [a]

[a] *Dipartimento di Informatica, Università di Pisa, I-56125 Pisa, Italia,* {bruni,ugo}@di.unipi.it

[b] *Computer Science Laboratory, SRI International, Menlo Park, CA 94025, USA,* meseguer@csl.sri.com

Abstract

2-categories and double categories are respectively the natural semantic ground for *rewriting logic* (RL) and *tile logic* (TL). Since 2-categories can be regarded as a special case of double categories, then RL can be easily embedded into TL, where also rewriting synchronization is considered. Since RL is the semantic basis of several existing languages, it is useful to map TL *back* into RL to have an executable framework for tile specifications. We extend the results of a previous work of two of the authors, focusing on tile systems where the algebraic structures for *configurations* and *observations* rely on some common auxiliary structure (e.g., for pairing, projecting, etc.). The new model theory required to relate the categorical models of the two logics is an *extended* version of the theory of 2-categories, and is defined using *partial membership equational logic*. More concretely, this semantic mapping yields a rewriting logic implementation of tile logic, where a meta-layer is required to control the rewritings, so that only tile proofs are computed.

Key words: Tile logic, rewriting logic, double categories, 2-categories, partial membership equational logic.

1 Introduction

Rewriting logic (RL) [20] is an elegant and expressive semantic framework for the specification of languages and systems, and is also a logical framework in which many other logics can be represented [19]. In it term rewriting is fully exploited as a basic computational paradigm: terms are regarded as states of a concurrent system whose state transitions are given by the rewriting rules of the specification. Moreover, a rewriting system specification is viewed as a logical theory such that each computation corresponds to a *sequent* entailed

Preprint submitted to Elsevier Preprint 7 May 1999

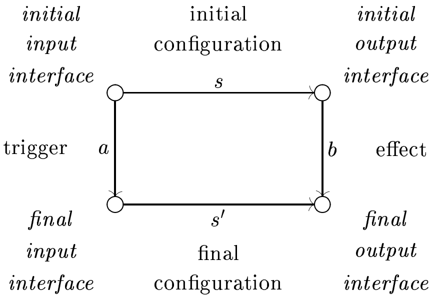

Fig. 1: A tile.

by the theory. The *entailment* relation is specified by simple inference rules, accordingly to the term algebra under consideration. For example, rewriting rules (possibly containing free variables) can be freely contextualized and instantiated, and sequentially composed. Derivations are then provided with an algebraic structure by decorating the sequents with proof terms, and decorated sequents are axiomatized in such a way that deduction in the logic becomes equivalent to concurrent computing. Since system states are defined as algebraic data types satisfying certain equational properties, rewriting logic can be parametrized w.r.t. the underlying equational logic used to axiomatize the states (e.g., order sorted, partial membership, hidden).

The *tile model* [15] is a formalism for modular descriptions of interactive systems, extending the classical SOS approach [24] and RL to a framework where the rules, called *tiles*, have a more general format. Tiles define the behaviour of certain *open* (e.g., partially specified) system configurations, which may interact through their interfaces. Then, the behaviour of a complex system is defined as a coordinated evolution of its sub-configurations. A tile has the form illustrated in Fig. 1, also written $s \xrightarrow[b]{a} s'$, stating that the *initial configuration* s of the system can evolve to the *final configuration* s' producing an observable *effect* b, provided that the subcomponents of s evolve to the subcomponents of s', producing the *trigger* a. The vertices of the tile are called *interfaces* (e.g., each configuration has both an *input* and an *output* interface). Tiles can be composed horizontally (by synchronizing an effect with a trigger), vertically (computational evolutions of a fixed component), and in parallel (concurrent steps) to generate larger tiles. By analogy with RL, the tile model also comes equipped with a purely logical presentation, where tiles become sequents subject to certain inference rules. In this sense, *tile logic* (TL) is a "logic of concurrent systems with synchronization mechanisms."

It is evident that TL extends (unconditional) RL, where both triggers and effects are just identities and rewritings can be applied freely (i.e., without interactions). On the other hand, several language implementation efforts (CafeObj [12], ELAN [1], Maude [9]) not only have RL as their semantic basis, but also support either executable specifications or parallel programming in

RL. Thus, a conservative mapping of TL into RL would facilitate the development of tile specifications. We investigate this issue, extending the results in [22] to configurations and observations relying on common *auxiliary structure*. The ordinary representation of terms over a signature often overrides the auxiliary structure needed to move from the signature to the associated term algebra. We call it auxiliary because it is common to *all* term algebras, i.e., it is independent from the signature under consideration. Our presentation promotes instead this aspect, taking advantage of the work of Pfender [23] on S-monoidal categories (where "S" stands for *substitution*) and of Lawvere [18]. The idea is to view a signature as a graph whose nodes are elements of the free monoid over the set of sorts, and then to freely enrich the graph with the additional structure and operations required. In particular, to recover the cartesian structure of the term algebra three auxiliary operators must be provided: *symmetries* (γ), *duplicators* (∇), and *dischargers* (!) (see Appendix A). Since several applications that we studied concern just symmetric theories rather than cartesian theories, we separately discuss the case where only symmetries are introduced. In particular, we focus on *process tile logic* (PTL) and *term tile logic* (TTL), two classes introduced in [3].

Configurations and observations in PTL are defined in terms of directed acyclic hyper-graphs, where each node has at most one entering/exiting arc (the terminology "process" is taken from *net theory*, due to the analogous characterization of *concatenable processes* of PT Petri nets [11]). Here configurations may model states of a large variety of distributed systems (at a certain level of abstraction), and observations may model causal dependencies between the evolutions of concurrent agents. *Flat* versions (i.e., where sequents are not decorated) of PTL have been used to define suitable models of computation for mobile calculi, and causal and located concurrent systems [14]. The auxiliary tiles of PTL express consistent permutations of interfaces. TTL represents the obvious extension of term rewriting logic. The auxiliary tiles of TTL express consistent permutations of interfaces, copying, and projections.

A rewriting theory $\mathcal{R}$ yields a cartesian *2-category* $\mathcal{L}_{\mathcal{R}}$, which does for $\mathcal{R}$ what a Lawvere theory does for a signature [20]. In [15] it is pointed out that if also rewriting synchronization is considered, then *double categories* offer a natural model. Then, in the same way as the term algebra is freely generated by a signature, and the initial model of rewriting logic is freely generated from the rules of the rewriting system, a tile system freely generates a double category, which gives its natural operational characterization in the spirit of initial model semantics. The natural semantics of process and term tile logics can be given in terms of *symmetric strict monoidal double categories* and *cartesian double categories* ("with consistently chosen products"), respectively. As far as we know, these notions were introduced in [3], because all previous attempts (based on internal constructions) for analogous definitions led to asymmetric models, where the auxiliary structure is fully exploited in one dimension only.

In particular, we developed an alternative approach, based on the idea of *hypertransformations* [13] for many-fold categories, defining the notion of *double transformations* (called *generalized transformations* in [3]), and instantiating such definition to the specific cases of symmetries, duplicators, and dischargers. This approach motivates a formal characterization of auxiliary tiles: *Auxiliary tiles for process and term tile logic are suitable double transformations.*

The comparison between tile logic and rewriting logic takes place by embedding their categorical models in a recently developed, more general framework, called *partial membership equational logic* (**PMEqtl**) [21]. Using **PMEqtl**, it is possible to define an extended version of 2-categories where also double categories can be conveniently embedded. **PMEqtl** offers a natural framework for categorical structures, firstly because arrows composition is a partial operation, and secondly because membership predicates over a poset of sorts allow, e.g., the objects to be modelled as a subset of the arrows, and the arrows as a subset of the cells. Moreover, the *tensor product construction of theories* illustrated in [22] can be easily formulated in **PMEqtl**, and yields a concise definition of, e.g., monoidal double categories.

From the application viewpoint, the mapping from tile logic into rewriting logic becomes effective, provided that the rewriting engine is able to distinguish whether the *proof* of the computation is in the tile logic or not. For this purpose we can exploit the *reflective* capabilities of the Maude language [9] (`http://maude.csl.sri.com`), by defining suitable *internal strategies* [8], which help the user analyze and control computations. Such strategies becomes very simple for a large class of tile systems, called *uniform*, as illustrated, e.g., in [5,6].

Structure of the paper. In Section 2, we present process and term tile logic and their categorical models. In Section 3, we compare TL (with shared structure) and RL using **PMEqtl** techniques, and state the main correspondence results of the paper. In Appendix A, we recall some basics on symmetric and algebraic theories, and on the associated auxiliary structure.

An early version of most material presented in this paper appeared in the Technical Report [3] and in the PhD thesis of one of the authors [2], together with the application of our results to several case studies. Several complementary parts of our research have been published separately: In particular, the part concerning the definition of a kernel of internal strategies to support the implementation of process and term tile systems in Maude appeared in [4], the case studies of finite and located CCS (respectively concerning TTL and PTL) have been discussed in [5], the case study of full CCS (in TTL) is treated in [6], and the presentation of symmetric monoidal and cartesian double categories as models for PTL and TTL has been accepted for publication [7].

2 Background

In tile logic, configurations and observations form two monoidal categories sharing the same class of objects. Category theory offers a convenient characterization of both configurations and observations also in terms of *algebraic theories* [18,17], described in Appendix A. The free algebraic theory associated to a (one-sorted) signature Σ is called the Lawvere theory for Σ. A Lawvere theory is just an alternative presentation of a signature, because the additional term structure is generated in a completely free way. We refer to Appendix A for the definition of *hypersignatures* Σ, and of the associated *symmetric* and *algebraic theories*, respectively denoted by $\mathbf{S}(\Sigma)$ and $\mathbf{A}(\Sigma)$.

The use of a *wire and box* notation gives a visual understanding of the rôle played by auxiliary structure: variables are represented by wires and the operators of the signature are denoted by labelled boxes. For example, let us consider the tile system defined in [14] for located CCS, specifically the tiles for action prefix execution, and synchronization pictured in Figure 2. Without getting too involved in the formal definition of the calculus and of the whole tile system, we just recall that the configuration signature (the system is one-sorted) contains unary action prefix operators μ (one for each action) and a constant *nil*, whereas the observation hypersignature contains unary operations (one for each action) and a special operator that takes two arguments and returns two results. The idea is that sequential agents are represented by composing action prefixes and *nil* (and sums), and that their output interface is the location they are assigned to. Two sequential agents can be composed in parallel, and it is also possible to make them share the working location. On the observation side we capture the interactions between sequential agents. The meaning of the rules in Figure 2 is that:

(1) A generic action prefix configuration (the box μ, whatever is attached on its left, since the trigger of the rule is an identity) can generate an effect consisting of the box together with a link to the location where it took place (i.e. y_1). Then, the location x_1 becomes active and sequential processes located in it are enabled. A *nil* pointer to the old location y_1 is maintained, because other sequential processes might share that location.
(2) Two complementary actions and executed on two locations y_1 and y_2 can be synchronized generating the effect as a result, together with the links to the locations where the actions have been performed. This allows one to keep causality information about the locations where actions took place.

By using the wire and box notation for both configurations and observations, it becomes evident that wire rearrangements (e.g., permutations) belong to both dimensions (i.e., they are *shared*), and that consistent rearrangements of wires

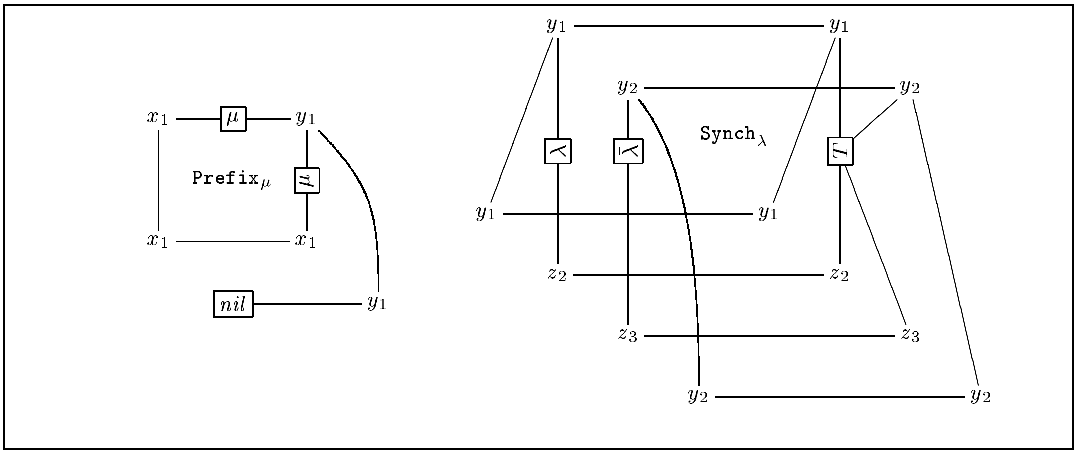

Fig. 2: Wire and box representation for tiles adapted from [14].

$$
\begin{tikzcd}
a \arrow[r, "h"] \arrow[d, "v"'] \arrow[dr, phantom, "A"] & b \arrow[d, "u"] \\
c \arrow[r, "g"'] & d
\end{tikzcd}
\qquad
\begin{tikzcd}
a \arrow[r, "h"] \arrow[d, "v"'] \arrow[dr, phantom, "A"] & b \arrow[r, "f"] \arrow[d, "u"] \arrow[dr, phantom, "B"] & b_1 \arrow[d, "w"] \\
c \arrow[r, "g"'] & d \arrow[r, "k"'] & d_1
\end{tikzcd}
= A * B
\qquad
\begin{tikzcd}
a \arrow[r, "h"] \arrow[d, "v"'] \arrow[dr, phantom, "A"] & b \arrow[d, "u"] \\
c \arrow[r, "g"] \arrow[d, "z"'] \arrow[dr, phantom, "C"] & d \arrow[d, "s"] \\
c_1 \arrow[r, "h'"'] & d_2
\end{tikzcd}
= A \cdot C
$$

Fig. 3: A cell A and horizontal/vertical composition of cells.

on both dimensions do not change the meaning of a rule, but only its interface. (in the sense that, for any auxiliary tile $s \xrightarrow[b]{a} s'$, s followed by b induces the same transformation as a followed by s'). In the example above, permutations of the interface can be necessary to propagate the synchronization through the other operators of the process algebra, or to bring closer the actions to be synchronized, as it is mandatory in the agent $\lambda.nil \mid \alpha.nil \mid \bar{\lambda}.nil$, where the locations of the two synchronizing agents are not contiguous. Thus, we have the following naïve characterization of auxiliary tiles: *Auxiliary tiles are consistent rearrangements of interfaces in both dimensions.* Other examples having a natural axiomatization in terms of the concepts introduced in this paper range from process algebras [15] and mobile calculi [14], to λ-calculus, and logic programming (currently under investigation).

2.1 Symmetric Monoidal and Cartesian Double Categories

A *double category* is a category internal to **Cat**. Due to the specific structure of **Cat**, double categories admit the following presentation [16].

Definition 1 (Double Category) *A* double category $\mathcal{D}$ *consists of a collection* $a, b, c, \ldots$ *of* objects*, a collection* $h, g, f, \ldots$ *of* horizontal arrows*, a collection* $v, u, w, \ldots$ *of* vertical arrows *and a collection* $A, B, C, \ldots$ *of* double cells *(also called* cells*), where:*

(1) *Objects and horizontal arrows form the* horizontal 1-category $\mathcal{H}$, *with identity* id^a *for each object* a, *and composition* $\_ * \_$.
(2) *Objects and vertical arrows form also a category, called the* vertical 1-category $\mathcal{V}$, *with identity* id_a *for each object* a, *and composition* $\_ \cdot \_$ *(abusing the notation we will denote both* id^a *and* id_a *either with the object name* a *or with* id_a*).*
(3) *Cells are assigned* horizontal source *and* target *(which are vertical arrows) and* vertical source *and* target *(which are horizontal arrows). Given a cell* A, *with vertical source* h, *vertical target* g, *horizontal source* v, *and horizontal target* u *(written* $A : h \xrightarrow[u]{v} g$*), then these arrows must form a square as illustrated in Fig. 3. We refer to the* sides *of cells as* north, south, west *and* east, *with the obvious meaning.*
(4) *Cells can be composed both horizontally* ($\_ * \_$) *and vertically* ($\_ \cdot \_$) *as follows: given* $A : h \xrightarrow[u]{v} g$, $B : f \xrightarrow[w]{u} k$, *and* $C : g \xrightarrow[s]{z} h'$, *then* $A * B : (h * f) \xrightarrow[w]{v} (g * k)$, *and* $A \cdot C : h \xrightarrow[u \cdot s]{v \cdot z} h'$ *are cells (see Figure 3). Moreover, given a fourth cell* $D : k \xrightarrow[t]{s} f'$, *the two compositions verify the* exchange law*:* $(A \cdot C) * (B \cdot D) = (A * B) \cdot (C * D)$. *Under these rules, cells form both a horizontal category* $\mathcal{D}^*$ *and a vertical category* $\mathcal{D}^{\cdot}$, *with respective identities* $1_v : a \xrightarrow[v]{v} c$ *and* $1^h : h \xrightarrow[b]{a} h$. *Given* $1^h : h \xrightarrow[b]{a} h$ *and* $1^g : g \xrightarrow[c]{b} g$, *then* $1^h * 1^g = 1^{h*g}$ *(and similarly for vertical composition of horizontal identities). Furthermore, horizontal and vertical identities of identities coincide, i.e.,* $1_{id_a} = 1^{id^a}$ *and are denoted by* 1_a, *or simply* a.

Example 2 (Quartet Category) *Given a category* $\mathcal{C}$, *we denote by* $\Box\mathcal{C}$ *the* category of quartets *of* $\mathcal{C}$*: its objects are the objects of* $\mathcal{C}$, *its horizontal and vertical arrows are the arrows of* $\mathcal{C}$, *and its cells are the commuting square diagrams* $f \xrightarrow[u]{v} g$ *of arrows in* $\mathcal{C}$, *where* $f, v, u, g \in \mathcal{C}$ *and* $f; u = v; g$ *in* $\mathcal{C}$.

Given two double categories $\mathcal{D}$ and $\mathcal{E}$, a *double functor* $F : \mathcal{D} \to \mathcal{E}$ is a 4-tuple of functions mapping objects to objects, horizontal and vertical arrows to horizontal and vertical arrows, and cells to cells, preserving identities and compositions of all kinds. Double functors can be composed in the obvious way, and any double category has associated an identity functor mapping each element into itself. This yields a category **DCat** whose objects are double categories and whose arrows are double functors.

Notice that it is always possible to characterize two suitable *diagonal* subcategories of a double category $\mathcal{D}$. In fact, those cells having identities as both horizontal and vertical targets are the arrows of a category whose composition $\_ \lhd \_$ is defined as $A \lhd B = (A * 1^g) \cdot (1_u * B)$ (for any cells $A : h \xrightarrow[id_b]{v} id_b$, and $B : g \xrightarrow[id_c]{u} id_c$). In a similar way, we can define a diagonal composition $\_ \rhd \_$ for those cells having identities as both horizontal and vertical sources. These two subcategories are respectively denoted by $\mathcal{D}^{\lhd}$ and $\mathcal{D}^{\rhd}$.

Table 1

$$
\begin{array}{ll}
\gamma_{u\otimes w,v} = (1_u \otimes \gamma_{w,v}) * (\gamma_{u,v} \otimes 1_w), & \gamma_{v,u} * \gamma_{u,v} = 1_{v\otimes u}, \\
\rho_{f\otimes g,h} = (1^f \otimes \rho_{g,h}) \cdot (\rho_{f,h} \otimes 1^g), & \rho_{h,g} \cdot \rho_{g,h} = 1^{h\otimes g}, \\
\sigma_{a\otimes b,c} = (1_a \otimes \sigma_{b,c}) \triangleleft (\sigma_{a,c} \otimes 1_b), & \sigma_{a,b} \triangleleft \sigma_{b,a} = 1_{a\otimes b}.
\end{array}
$$

Let $F, G : \mathcal{D} \to \mathcal{E}$ be two double functors. Following the internal construction approach, an internal natural transformation is an arrow in **Cat** which verifies the naturality conditions w.r.t. one composition and which is functorial w.r.t. the other composition. We call natural *cmp*-transformation (with *cmp* $\in \{*, \cdot\}$) a transformation which satisfies the naturality requirement w.r.t. the composition operator *cmp* and which is functorial w.r.t. the other operator.

Definition 3 (Natural Double Transformation) *Let $\mathcal{D}$ and $\mathcal{E}$ be double categories. Given four double functors F_{00}, F_{10}, F_{01}, and F_{11} from $\mathcal{D}$ to $\mathcal{E}$, a* natural double transformation *is a tuple $(\alpha_{0\_}, \alpha_{1\_}, \beta_{\_0}, \beta_{\_1}, \eta)$, which is pictured as the cell*

$$
\begin{tikzcd}
F_{00} \arrow[r, "\alpha_{0\_}"] \arrow[d, "\beta_{\_0}"'] \arrow[dr, phantom, "\eta"] & F_{01} \arrow[d, "\beta_{\_1}"] \\
F_{10} \arrow[r, "\alpha_{1\_}"'] & F_{11}
\end{tikzcd}
$$

, where: (1) for $i = 0, 1$, the symbol $\alpha_{i\_}$ denotes a natural $*$-transformation from F_{i0} to F_{i1} (i.e., $\alpha_{i\_}$ is also a functor from the category $\mathcal{V}$ of vertical arrows of $\mathcal{D}$ to category $\mathcal{E}^{\cdot}$), (2) for $i = 0, 1$, the symbol $\beta_{\_i}$ denotes a natural $\cdot$-transformation from F_{0i} to F_{1i} (i.e., $\beta_{\_i}$ is also a functor from the category $\mathcal{H}$ of horizontal arrows of $\mathcal{D}$ to $\mathcal{E}^*$), and (3) the symbol η defines both a natural transformation from $\alpha_{0\_}$ to $\alpha_{1\_}$ (seen as functors from $\mathcal{V}$ to $\mathcal{E}^{\cdot}$) and from $\beta_{\_0}$ to $\beta_{\_1}$ (seen as functors from $\mathcal{H}$ to $\mathcal{E}^*$).

By analogy with the 1-fold case, the properties of the double transformation can be represented by a commuting *hypercube* (details in [7]).

A *strict monoidal double category* (SMD) is a triple $(\mathcal{D}, \otimes, e)$, where $\mathcal{D}$ is a double category, $\otimes : \mathcal{D} \times \mathcal{D} \to \mathcal{D}$ is a double functor called the *tensor product*, and e is an object of $\mathcal{D}$, such that the tensor product is associative, and that e is the *unit* for $\_ \otimes \_$.

Definition 4 (SsMDCat) *A* symmetric, strict monoidal double category *(*SSMD*) is a tuple $(\mathcal{D}, \otimes, e, \sigma)$ such that $(\mathcal{D}, \otimes, e)$ is a* SMD*, and σ is the natural double isomorphism*

$$
\begin{tikzcd}
\otimes \arrow[r, "\gamma"] \arrow[d, "\rho"'] \arrow[dr, phantom, "\sigma"] & X;\otimes \arrow[d, "1"] \\
X;\otimes \arrow[r, "1"'] & X;\otimes
\end{tikzcd}
$$

, where $X : \mathcal{D} \times \mathcal{D} \to \mathcal{D} \times \mathcal{D}$ is the double functor s.t. for all $A, B \in \mathcal{D}$, $X(A, B) = (B, A)$, and the coherence *axioms in Table 1 hold for γ, ρ and σ (for all vertical arrows v, u, w, horizontal arrows h, g, f, and objects a, b, c). We denote by* **SsMDCat** *the category of* SSMD*'s and monoidal double functors preserving symmetries.*

Table 2

$\nabla_{v\otimes u} = (\nabla_v \otimes \nabla_u) * (1_v \otimes \gamma_{v,u} \otimes 1_u)$,	$\delta_{h\otimes g} = (\delta_h \otimes \delta_g) \cdot (1^h \otimes \rho_{h,g} \otimes 1^g)$,
$!_{v\otimes u} = !_v \otimes !_u$, $\nabla_{id_e} = 1_e = !_{id_e}$,	$\dagger_{h\otimes g} = \dagger_h \otimes \dagger_g$, $\delta_{id_e} = 1_e = \dagger_{id_e}$,
$\nabla_v * (\nabla_v \otimes 1_v) = \nabla_v * (1_v \otimes \nabla_v)$,	$\delta_h \cdot (\delta_h \otimes 1^h) = \delta_h \cdot (1^h \otimes \delta_h)$,
$\nabla_v * \gamma_{v,v} = \nabla_v$, $\nabla_v * (1_v \otimes !_v) = 1_v$,	$\delta_h \cdot \rho_{h,h} = \delta_h$, $\delta_h \cdot (1^h \otimes \dagger_h) = 1^h$.
$\pi_{a\otimes b} = (\pi_a \otimes \pi_b) \triangleleft (1_a \otimes \sigma_{a,b} \otimes 1_b)$,	$\tau_{a\otimes b} = (\tau_a \otimes \tau_b) \triangleright (1_a \otimes \sigma^{-1}_{a,b} \otimes 1_b)$,
$\phi_{a\otimes b} = \phi_a \otimes \phi_b$, $\pi_e = 1_e = \phi_e$,	$\psi_{a\otimes b} = \psi_a \otimes \psi_b$, $\tau_e = 1_e = \psi_e$,
$\pi_a \triangleleft (\pi_a \otimes 1_a) = \pi_a \triangleleft (1_a \otimes \pi_a)$,	$\tau_a \triangleright (\tau_a \otimes 1_a) = \tau_a \triangleright (1_a \otimes \tau_a)$,
$\pi_a \triangleleft \sigma_{a,a} = \pi_a$, $\pi_a \triangleleft (1_a \otimes \phi_a) = 1_a$,	$\tau_a \triangleright \sigma^{-1}_{a,a} = \tau_a$, $\tau_a \triangleright (1_a \otimes \psi_a) = 1_a$.
$\psi_a * \phi_a = !_a$, $\tau_a * \pi_a = \nabla_a$,	$\tau_a \cdot \pi_a = \delta_a$, $\psi_a \cdot \phi_a = \dagger_a$.

The following proposition gives the correspondence between double and ordinary symmetries, i.e., between the two notions of auxiliary tiles that we have introduced.

Proposition 5 *The forgetful functor from* **SsMDCat** *to* **Set**, *mapping a* SSMD *into its set of objects has a left adjoint which maps each set S into the* free SSMD *on S. Moreover, for any S, the free* SSMD *on S is isomorphic to the quartet category of the free symmetric strict monoidal category on S.*

Definition 6 (CDCat) *A* cartesian double category *(*CDC*) (with consistently chosen products) is a tuple $(\mathcal{D}, \otimes, e, \sigma, \pi, \tau, \phi, \psi)$ such that $(\mathcal{D}, \otimes, e, \sigma)$ is a* SSMD, *and the natural double transformations π, τ, ϕ, and ψ are as below*

$$
\begin{tikzcd}
1_{\mathcal{D}} \arrow[r, "\nabla"] \arrow[d, "\delta"'] \arrow[dr, phantom, "\pi"] & \Delta;\otimes \arrow[d, "1"] \\
\Delta;\otimes \arrow[r, "1"'] & \Delta;\otimes
\end{tikzcd}
\qquad
\begin{tikzcd}
1_{\mathcal{D}} \arrow[r, "1"] \arrow[d, "1"'] \arrow[dr, phantom, "\tau"] & 1_{\mathcal{D}} \arrow[d, "\delta"] \\
1_{\mathcal{D}} \arrow[r, "\nabla"'] & \Delta;\otimes
\end{tikzcd}
\qquad
\begin{tikzcd}
1_{\mathcal{D}} \arrow[r, "!"] \arrow[d, "\dagger"'] \arrow[dr, phantom, "\phi"] & e \arrow[d, "1"] \\
e \arrow[r, "1"'] & e
\end{tikzcd}
\qquad
\begin{tikzcd}
1_{\mathcal{D}} \arrow[r, "1"] \arrow[d, "1"'] \arrow[dr, phantom, "\psi"] & 1_{\mathcal{D}} \arrow[d, "\dagger"] \\
1_{\mathcal{D}} \arrow[r, "!"'] & e
\end{tikzcd}
$$

where $\Delta : \mathcal{D} \to \mathcal{D} \times \mathcal{D}$ is the double functor s.t. for all $A \in \mathcal{D}$, $\Delta(A) = (A, A)$. Moreover, the following axioms hold: (1) the coherence axioms for ∇, $!$, δ, and $\dagger$ in the first block of Table 2 (for all vertical arrows v, u, and horizontal arrows h, g), (2) the coherence axioms for π, ϕ, τ, and ψ in the second block of Table 2 (for all objects a, b), (3) the double coherence axioms for π, ϕ, τ, and ψ in the third block of Table 2 (for any object a). We denote by **CDCat** *the category of* CDC*'s and monoidal double functors preserving all symmetries, duplicators and dischargers.*

The coherence axioms in Tables 1 and 2 are the ordinary one-dimensional coherence axioms expressed accordingly to the four compositions (since symmetries have inverses the axioms w.r.t. $\_\triangleright\_$ are redundant). The analogous of Prop. 5 holds for the cartesian case.

Definition 7 *A* process tile system *(*PTS*)* $\mathcal{R}$ *is a 4-tuple* $\langle \Sigma_H, \Sigma_V, N, R\rangle$, *where* Σ_H *and* Σ_V *are hyper-signatures over the same set of sorts* S, N *is a set of rule names and* R *is a function* $R : N \to \mathbf{S}(\Sigma_H) \times \mathbf{S}(\Sigma_V) \times \mathbf{S}(\Sigma_V) \times \mathbf{S}(\Sigma_H)$, *such that for all* $r \in N$, *if* $R(r) = \langle h, u, v, g\rangle$, *then the arrows* h, u, v *and* g *can compose the tile* $r : h \xrightarrow[u]{v} g$.

Rules in $\mathcal{R}$ can be interpreted as labelled sequents in an appropriate *logic of tiles*. In *process tile logic*, the consistent permutation of wires is introduced for free. Starting from a PTS, we derive all the tiles obtained by applications of some *deduction rules*, defining the *free* compositions of the basic and auxiliary tiles in $\mathcal{R}$ according to parallel, horizontal, and vertical compositions. We decorate the sequents with *proof terms*, that can be axiomatized in order to capture equivalent proofs. The resulting equivalence classes make fewer identifications than those induced by the undecorated *flat* version.

Definition 8 (Process Tile Logic) *Let* $\mathcal{R} = \langle \Sigma_H, \Sigma_V, N, R\rangle$ *be a* PTS. *We say that* $\mathcal{R}$ entails *the class* $P_{\mathrm{p}}(\mathcal{R})$ *of* process sequents, *obtained by a finite number of applications of the following inference rules.*

Basic Proof Sequents: Generators and Identities:

$$\frac{r : h \xrightarrow[u]{v} g \in R(N)}{r : h \xrightarrow[u]{v} g \in P_{\mathrm{p}}(\mathcal{R})} \qquad \frac{v : a \to c \in \mathbf{S}(\Sigma_V)}{1_v : id_a \xrightarrow[v]{v} id_c \in P_{\mathrm{p}}(\mathcal{R})} \qquad \frac{h : a \to b \in \mathbf{S}(\Sigma_H)}{1^h :\ h \xrightarrow[id_b]{id_a} h\ \in P_{\mathrm{p}}(\mathcal{R})}$$

Auxiliary Proof Sequents: Symmetries: $\dfrac{a, b \in S^*}{\sigma_{a,b} :\ \gamma_{a,b} \xrightarrow[id_{b\otimes a}]{\gamma_{a,b}} id_{b\otimes a}\ \in P_{\mathrm{p}}(\mathcal{R})}$

Composition Rules: Parallel and Sequential compositions:

$$\frac{\alpha : h \xrightarrow[u]{v} g,\ \ \alpha' : h' \xrightarrow[u']{v'} g' \in P_{\mathrm{p}}(\mathcal{R})}{\alpha \otimes \alpha' :\ h \otimes h' \xrightarrow[u\otimes u']{v\otimes v'} g \otimes g'\ \in P_{\mathrm{p}}(\mathcal{R})}$$

$$\frac{\alpha : h \xrightarrow[u]{v} g,\ \beta : h' \xrightarrow[u']{u} g' \in P_{\mathrm{p}}(\mathcal{R})}{\alpha * \beta : h * h' \xrightarrow[u']{v} g * g' \in P_{\mathrm{p}}(\mathcal{R})} \qquad \frac{\alpha : h \xrightarrow[u]{v} g,\ \beta : g \xrightarrow[u']{v'} g' \in P_{\mathrm{p}}(\mathcal{R})}{\alpha \cdot \beta :\ h \xrightarrow[u\cdot u']{v\cdot v'} g'\ \in P_{\mathrm{p}}(\mathcal{R})}$$

For any sequent $\alpha \in P_{\mathrm{p}}(\mathcal{R})$ *we write* $\mathcal{R} \vdash_{\mathrm{p}} \alpha$. *We also write* $\mathcal{R} \vdash_{\mathrm{fp}} h \xrightarrow[u]{v} g$ *to say that the flat sequent* $h \xrightarrow[u]{v} g$ *is entailed by* $\mathcal{R}$.

Entailment in the logic can be expressed using the algebraic structure on proof sequents. For example, it is easy to verify that the swapping of effects $\gamma_{v,u}$, and of configurations $\rho_{h,g}$ can also be defined. Due to space limitations, we refer the interested reader to [3,2] for the detailed description of the axiomatization of proof sequents (it is reminiscent of the axioms for SSMD's). We say that $\mathcal{R}$ *entails* the class $A_{\mathrm{p}}(\mathcal{R})$ of *abstract process sequents*, whose elements are equivalence classes of $P_{\mathrm{p}}(\mathcal{R})$ modulo such axiomatization. This makes $A_{\mathrm{p}}(\mathcal{R})$ into a SSMD which defines the initial model of PTL in **SsMDCat**.

2.3 Term Tile Logic

In this section, we present a tile format where configurations and observations are just terms. To be consistent with the notation of PTL, we adopt the equivalent presentation of term algebras as algebraic theories.

Definition 9 *A* term tile system *(*TTS*) $\mathcal{R}$ is a 4-tuple $\langle \Sigma_H, \Sigma_V, N, R\rangle$, where Σ_H and Σ_V are signatures over the same set of sorts S, N is a set of rule names and R is a function $R : N \to \mathbf{A}(\Sigma_H) \times \mathbf{A}(\Sigma_V) \times \mathbf{A}(\Sigma_V) \times \mathbf{A}(\Sigma_H)$, such that for all $r \in N$, if $R(r) = \langle h, u, v, g\rangle$, then the arrows h, u, v and g can compose the tile $r : h \xrightarrow[u]{v} g$ (without loss of generality, we can assume that u and g are terms and not term tuples).*

Definition 10 (Term Tile Logic) *Let $\mathcal{R} = \langle \Sigma_H, \Sigma_V, N, R\rangle$ be a* TTS*. We say that $\mathcal{R}$* entails *the class $P_{\mathrm{t}}(\mathcal{R})$ of* term sequents*, obtained by applying the inference rules listed in Def. 8, together with the following rules:*

Auxiliary Proof Sequents: Duplicators and Dischargers:

$$\frac{a \in S^*}{\pi_a : \nabla_a \xrightarrow[id_{a\otimes a}]{\nabla_a} id_{a\otimes a} \in P_{\mathrm{t}}(\mathcal{R})} \qquad \frac{a \in S^*}{\tau_a : id_a \xrightarrow[\nabla_a]{id_a} \nabla_a \in P_{\mathrm{t}}(\mathcal{R})}$$

$$\frac{a \in S^*}{\phi_a : !_a \xrightarrow[id_\lambda]{!_a} id_\lambda \in P_{\mathrm{t}}(\mathcal{R})} \qquad \frac{a \in S^*}{\psi_a : id_a \xrightarrow[!_a]{id_a} !_a \in P_{\mathrm{t}}(\mathcal{R})}$$

For any sequent $\alpha \in P_{\mathrm{t}}(\mathcal{R})$ we write $\mathcal{R} \vdash_{\mathrm{t}} \alpha$. We also write $\mathcal{R} \vdash_{\mathrm{ft}} h \xrightarrow[u]{v} g$ to say that the flat term sequent $h \xrightarrow[u]{v} g$ is entailed by $\mathcal{R}$.

Again, due to lack of space, we refer the interested reader to [3,2] for the axiomatization of term sequents (it is reminiscent of the axioms for CDC's). We say that $\mathcal{R}$ *entails* the class $A_{\mathrm{t}}(\mathcal{R})$ of *abstract term sequents*, whose elements are equivalence classes of $P_{\mathrm{t}}(\mathcal{R})$ modulo such axiomatization. This makes $A_{\mathrm{t}}(\mathcal{R})$ into a cartesian double category, defining a suitable initial model in **CDCat**.

3 Relating Double Categories to Extended 2-categories

Rewriting logic is the semantic basis of several language implementation efforts (CafeOBJ [12], ELAN [1], Maude [9]). In particular, Maude supports executable specifications in rewriting logic and is efficiently implemented. Maude is also equipped with the important feature of meta-control mechanism on computations through user-definable internal execution strategies [10,8]. Since 2-categories are one of the most natural frameworks for interpreting the semantics of a rewriting system, a conservative mapping from double categories to 2-categories can provide a good theoretical basis for the implementation of tile systems. The comparison between TL and RL is carried out by axiomatizing their corresponding categorical models in *partial membership equational logic* [21]. The idea is to stretch tiles into ordinary rewriting rules: $h \xrightarrow[u]{v} g \mapsto a \underset{v;g}{\overset{h;u}{\Downarrow}} d$. However, after the flattening, we should preserve the capacity to distinguish configurations from observations, whereas the auxiliary structure can now be shared. We must be careful about two issues, namely, the possible identification of distinct double cells, and the possible existence of 2-cells having correct horizontal-vertical partition of the source and vertical-horizontal partition of the target, but which do not represent any double cell. From the facts that: (1) each arrow of a 2-category can be viewed as an identity 2-cell, (2) each auxiliary operator is a shared arrow, and (3) there is a *unique* auxiliary tile for each possible bidimensional transformation, (i.e., all possible decompositions of the proof terms of auxiliary tiles yielding the same border are equivalent), we have a third characterization of auxiliary tiles: *Auxiliary tiles are the possible square-shaped decomposition of the identity 2-cells associated to auxiliary constructors.*

After recalling the basics of **PMEqtl**, we extend the 2VH-categories of [22], to include in an appropriate sense the structure of SSMD's and of CDC's (and, more generally, of double categories with shared structure).

3.1 **PMEqtl** *and Tensor Products of Theories*

The features of **PMEqtl** (partiality, poset of sorts, membership assertions) offer a natural framework for the specification of categorical structures. Moreover, in [22] a notion of *tensor product* for partial algebraic theories is explicitly defined using the formulae of **PMEqtl**. As we shall see, the tensor product construction allows a very elegant definition of double categories and of their monoidal version. Due to space limitations, we refer the interested reader to [22] for a self-contained presentation of one-kinded **PMEqtl** (a more detailed exposition for the many-kinded case can be found in [21]). Informally, **PME-**

Table 3

Axioms:	$\mathtt{d}(f) : \mathtt{Object}$	$\mathtt{d}(a) = a$	$\mathtt{c}(f) : \mathtt{Object}$	$\mathtt{c}(a) = a$
Clauses:	$a; f = f \leftarrow \mathtt{d}(f) = a$		$f; a = f \leftarrow \mathtt{c}(f) = a$	
	$f; g : \mathtt{Arrow} \leftarrow \mathtt{c}(f) = \mathtt{d}(g)$		$\mathtt{c}(f) = \mathtt{d}(g) \leftarrow f; g : \mathtt{Arrow}$	
	$\mathtt{d}(f; g) = \mathtt{d}(f) \leftarrow \mathtt{c}(f) = \mathtt{d}(g)$		$\mathtt{c}(f; g) = \mathtt{c}(g) \leftarrow \mathtt{c}(f) = \mathtt{d}(g)$	
	$(f; g); h = f; (g; h) \leftarrow \mathtt{c}(f) = \mathtt{d}(g), \mathtt{c}(g) = \mathtt{d}(h)$			

qtl is a logic of partial algebras with subsorts and subsort polymorphism whose sentences are Horn clauses $\psi \leftarrow \psi_1, ..., \psi_n$ on equations $t = t'$ and on membership assertions $t : s$. Given a partial signature Ω over a poset of sorts, and a set Γ of Ω-sentences, we denote by $\mathbf{PAlg}_\Omega$ and $\mathbf{PAlg}_{\Omega,\Gamma}$ the categories of partial Ω-algebras and its full subcategory of those partial Ω-algebras that satisfy all sentences in Γ.

Example 11 *The theory of categories $T_{\mathtt{CAT}}$ is a theory in* **PMEqtl**. *Its poset of sorts is* $\mathtt{Object} \leq \mathtt{Arrow}$ *(each object a coincides with its identity arrow id_a). There are two unary* domain *and* codomain *operations* $\mathtt{d(\_)}$ *and* $\mathtt{c(\_)}$, *and a binary* composition $\_;\_$. *The sentences of $T_{\mathtt{CAT}}$ are listed in Table 3 (universally quantified over the variables $a : \mathtt{Object}$ and $f, g, h : \mathtt{Arrow}$)*

Proposition 12 (cf. [21]) *Given a theory morphism $H : (\Omega, \Gamma) \to (\Omega', \Gamma')$, its associated forgetful functor $U_H : \mathbf{PAlg}_{\Omega',\Gamma'} \to \mathbf{PAlg}_{\Omega,\Gamma}$ has a left adjoint $F_H : \mathbf{PAlg}_{\Omega,\Gamma} \to \mathbf{PAlg}_{\Omega',\Gamma'}$.*

Definition 13 *A theory morphism $H : (\Omega, \Gamma) \to (\Omega', \Gamma')$ is* conservative *(respectively* complete, persistent*) w.r.t. sort s if, for each algebra $A \in \mathbf{PAlg}_{\Omega,\Gamma}$, the component $(\eta_A)_s : A_s \to (U_H(F_H(A)))_s$ corresponding to s of the unit of the adjunction associated to H is injective (resp. surjective, bijective). The morphism H is* conservative *(resp.* complete, persistent*) if it is conservative (resp. complete, persistent) w.r.t. all sorts $s \in S$.*

Let $T = (\Omega, \Gamma)$ and $T' = (\Omega', \Gamma')$ be theories in **PMEqtl**, with $\Omega = (S, \leq, \Sigma)$ and $\Omega' = (S', \leq', \Sigma')$. Then, their *tensor product theory* $T \otimes T'$ is the theory with signature $\Omega \otimes \Omega'$ having: (1) poset of sorts $(S, \leq) \times (S', \leq')$, and (2) signature $\Sigma \otimes \Sigma'$, with an operator $f^l \in (\Sigma \otimes \Sigma')_n$ for each $f \in \Sigma_n$, and with an operator $g^r \in (\Sigma \otimes \Sigma')_m$ for each $g \in \Sigma'_m$ (here, superscript l and r of operators stand respectively for *left* and *right*). The axioms of $T \otimes T'$ are the *inherited axioms*, the *subalgebra axioms*, and the *homomorphism axioms* defined in [22]. The essential property of $T \otimes T'$ is expressed by Theorem 14.

Theorem 14 *Let T, T' be theories in* **PMEqtl**. *Then we have the following isomorphisms of categories: $\mathbf{PAlg}_{T \otimes T'} \simeq \mathbf{PAlg}_T(\mathbf{PAlg}_{T'}) \simeq \mathbf{PAlg}_{T'}(\mathbf{PAlg}_T)$.*

Table 4

sorts:	`(Object,Object)` to `Object`			`(Arrow,Arrow)` to `Square`		
	`(Arrow,Object)` to `Harrow`			`(Object,Arrow)` to `Varrow`		
operations:	d^l to `w`	c^l to `e`	$\_;^l\_$ to $\_*\_$	d^r to `n`	c^r to `s`	$\_;^r\_$ to $\_\cdot\_$

Table 5

sorts:	`Square` to `Cell`		`Harrow` to `Arrow`		`VArrow` to `Object`	
operations:	`w` to `l`	`e` to `r`	$\_*\_$ to $\_;\_$	`n` to `d`	`s` to `c`	$\_\cdot\_$ to $\_\circ\_$

Example 15 *A* double category *is an object of* $\mathbf{PAlg}_{T_{\mathtt{CAT}}}(\mathbf{PAlg}_{T_{\mathtt{CAT}}}) = \mathbf{DCat}$*, thus* $T_{\mathtt{DCAT}} = T_{\mathtt{CAT}} \otimes T_{\mathtt{CAT}}$*. For simplicity, we rename the sorts and operations of* $T_{\mathtt{DCAT}}$ *as in Table 4. A direct presentation of the theory* $T_{\mathtt{DCAT}}$ *in Maude-like notation can be found in [22].*

3.2 From 2-categories and 2VH-categories to 2EVH-categories

2-categories [16] can be considered as the special case of double categories in which all vertical arrows are objects. The explicit definition of $T_{\mathtt{2CAT}}$ is given by the renaming of $T_{\mathtt{DCAT}}$ in Table 5 that collapses `Varrow` into `Object` (the resulting poset of sorts is `Object` $\leq$ `Arrow` $\leq$ `Cell`).

The poset of sorts for *2VH-categories* [22] is shown in Fig. 4(a). The idea is that the theory $T_{\mathtt{2CAT}}$ is imported in $T_{\mathtt{2VHCAT}}$ as such, without any renaming. In addition, new sorts `Harrow`, `Varrow` and `Square` are introduced, which correspond to the homonymous sorts of $T_{\mathtt{DCAT}}$. If we are given a 2-category with two distinguished subcategories `Harrow` and `Varrow` of `Arrow` such that they are disjoint except for objects, and such that the horizontal and vertical components can be recovered from their composition, then we can form a double category by considering squares with horizontal and vertical sides, and we can define their horizontal and vertical composition by using the already existing cell composition of the 2-category. The complete description of $T_{\mathtt{2VHCAT}}$ can be found in [22]. We focus instead on a particular clause of the specification: (for all t : `Arrow`) t : `Object` $\leftarrow$ t : `Harrow`, t : `Varrow`. As a consequence, no shared structure between horizontal and vertical dimension can be defined, except for objects. For SSMD's and CDC's this is counterintuitive and leads us to the definition of the theory of *2EVH-categories*.

The poset of sorts that we propose for 2EVH-categories is shown in Figure 4(b). The sort `Mix` includes the auxiliary structure which is thus shared between the sorts `Horizontal` and `Vertical`. The sort `Arrow` contains `Vertical` and `Horizontal`. The sorts `Harrow` and `Hmix` (respectively `Varrow` and `Vmix`)

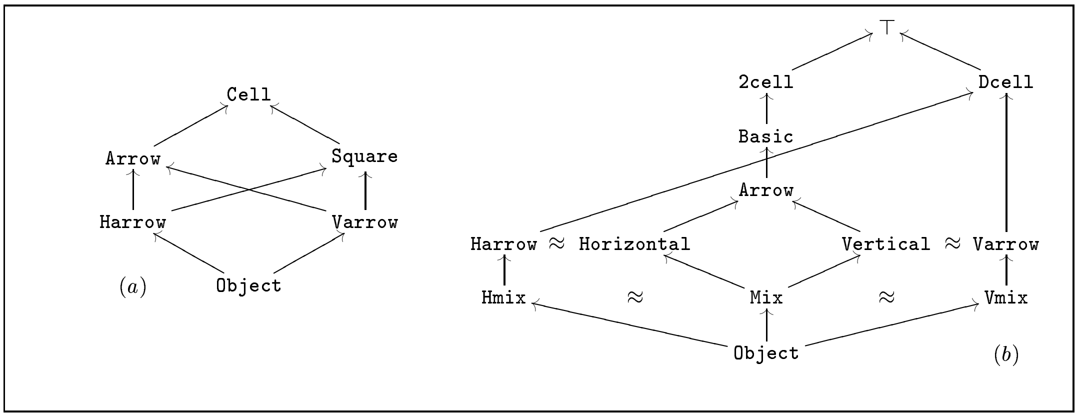

Fig. 4: The posets of sorts for (a) $T_{\texttt{2VHCAT}}$ and (b) $T_{\texttt{2EVHCAT}}$.

are isomorphic copies of sorts `Horizontal` and `Mix` (respectively `Vertical` and `Mix`). Figure 4(b) suggests calling *internal* the sorts `Horizontal`, `Vertical`, and `Mix` and calling *external* their isomorphic copies. The sort `Basic` contains identity cells and possibly some additional basic cells (e.g., of a 2-computad). Since the sorts `Horizontal` and `Vertical` share `Mix` arrows, it follows that more than one double cell can correspond to the same 2-cell. Moreover, there may exist some cells that do not generate any double cell. Thus, it is possible to define a total mapping from `Dcell` onto `2cell`, but in general this mapping is neither injective, nor surjective. However, the isomorphisms between external and internal sorts follows from the definition of the more general mapping.

Basically the theory $T_{\texttt{2EVHCAT}}$ includes the theory $T_{\texttt{2CAT}}$ (by renaming the sort `Cell` to `2cell`). Then, we require that the sequential composition of two `Horizontal` (`Vertical`) arrows is also a `Horizontal` (`Vertical`) arrow, and that the only arrows which are both `Horizontal` and `Vertical` are those of sort `Mix`. This is formalized by the following clauses (for all $t, t' : \texttt{Arrow}$, $h, h' : \texttt{Horizontal}$, $v, v' : \texttt{Vertical}$):

$$\begin{aligned}
t;t' : \texttt{Horizontal} &\leftarrow \texttt{r}(t) = \texttt{l}(t'), t : \texttt{Horizontal}, t' : \texttt{Horizontal} \\
\texttt{r}(h) = \texttt{l}(h') &\leftarrow h;h' : \texttt{Horizontal} \\
t;t' : \texttt{Vertical} &\leftarrow \texttt{r}(t) = \texttt{l}(t'), t : \texttt{Vertical}, t' : \texttt{Vertical} \\
\texttt{r}(v) = \texttt{l}(v') &\leftarrow v;v' : \texttt{Vertical} \\
t : \texttt{Mix} &\leftarrow t : \texttt{Horizontal}, t : \texttt{Vertical}
\end{aligned}$$

Then we introduce an operation `mk`, with arity five, such that if a `Basic` cell s has source $h; u$ and target $v; g$, where h and g are `Horizontal`, and u and v are `Vertical`, then $\texttt{mk}(s, h, u, v, g)$ is a `Dcell`. In this way we can distinguish all the possible double cells which are generated by different decompositions of the same `Basic` 2-cell. Since `Basic` contains all the identity cells, and `Horizontal` and `Vertical` can share more structure than just `Object`, it follows that the decomposition of many cells is not unique. As an example, let $m : a \to b$ be an

Table 6

Ax.	$\mathtt{mkh}(h) = \mathtt{mk}(h, h, \mathtt{r}(h), \mathtt{l}(h), h)$		$\mathtt{mkv}(v) = \mathtt{mk}(v, \mathtt{l}(v), v, v, \mathtt{r}(v))$	
	$\mathtt{mkh}(m) : \mathtt{Hmix}$	$\mathtt{mkh}(h) : \mathtt{Harrow}$	$\mathtt{mkv}(m) : \mathtt{Vmix}$	$\mathtt{mkv}(v) : \mathtt{Varrow}$
	$\mathtt{mk}(a, a, a, a, a) : \mathtt{Object}$		$\mathtt{mk}(a, a, a, a, a) = a$	
	$\mathtt{prj}(hm) : \mathtt{Mix}$	$\mathtt{prj}(vm) : \mathtt{Mix}$	$\mathtt{prj}(ha) : \mathtt{Horizontal}$	
	$\mathtt{prj}(va) : \mathtt{Vertical}$		$\mathtt{prj}(p) : \mathtt{2cell}$	
Cl.	$\mathtt{mk}(s, h, u, v, g) : \mathtt{Dcell} \leftarrow h; u = \mathtt{d}(s), v; g = \mathtt{c}(s)$			
	$h; u = \mathtt{d}(l) \leftarrow \mathtt{mk}(l, h, u, v, g) : \mathtt{Dcell}$		$v; g = \mathtt{c}(l) \leftarrow \mathtt{mk}(l, h, u, v, g) : \mathtt{Dcell}$	
	$\mathtt{prj}(\mathtt{mk}(l, h, u, v, g)) = l \leftarrow \mathtt{mk}(l, h, u, v, g) : \mathtt{Dcell}$			
	$\mathtt{mk}(\mathtt{prj}(p), h, u, v, g) : \mathtt{Dcell} \leftarrow h; u = \mathtt{d}(\mathtt{prj}(p)), v; g = \mathtt{c}(\mathtt{prj}(p))$			
	$h; u = \mathtt{d}(\mathtt{prj}(p)) \leftarrow \mathtt{mk}(\mathtt{prj}(p), h, u, v, g) : \mathtt{Dcell}$			
	$v; g = \mathtt{c}(\mathtt{prj}(p)) \leftarrow \mathtt{mk}(\mathtt{prj}(p), h, u, v, g) : \mathtt{Dcell}$			

arrow having sort Mix, then there are at least four possible ways in which the identity cell on m can generate a Dcell, namely $m_1 = \mathtt{mk}(m, m, b, m, b)$, $m_2 = \mathtt{mk}(m, m, b, a, m)$, $m_3 = \mathtt{mk}(m, a, m, m, b)$ and $m_4 = \mathtt{mk}(m, a, m, a, m)$ pictured as (vertical arrows are dotted):
$$\begin{tikzcd}[row sep=tiny, column sep=small] & b \arrow[dr, dotted, "b"] & \\ a \arrow[ur, "m"] \arrow[dr, dotted, "m"'] \arrow[rr, phantom, "m_1\Downarrow"] & & b \\ & b \arrow[ur, "b"'] & \end{tikzcd} \quad \begin{tikzcd}[row sep=tiny, column sep=small] & b \arrow[dr, dotted, "b"] & \\ a \arrow[ur, "m"] \arrow[dr, dotted, "a"'] \arrow[rr, phantom, "m_2\Downarrow"] & & b \\ & a \arrow[ur, "m"'] & \end{tikzcd} \quad \begin{tikzcd}[row sep=tiny, column sep=small] & a \arrow[dr, dotted, "m"] & \\ a \arrow[ur, "a"] \arrow[dr, dotted, "m"'] \arrow[rr, phantom, "m_3\Downarrow"] & & b \\ & b \arrow[ur, "b"'] & \end{tikzcd} \quad \begin{tikzcd}[row sep=tiny, column sep=small] & a \arrow[dr, dotted, "m"] & \\ a \arrow[ur, "a"] \arrow[dr, dotted, "a"'] \arrow[rr, phantom, "m_4\Downarrow"] & & b \\ & a \arrow[ur, "m"'] & \end{tikzcd}$$

We introduce two additional derived operators, mkh and mkv, which are "specialized" version of the more general mk. In the example above $m_2 = \mathtt{mkh}(m)$ and $m_3 = \mathtt{mkv}(m)$. Moreover, the isomorphisms between internal and external sorts are given by mkh and mkv. Their inverses can be easily defined using a projection operator prj (see Table 6, for all $l : \mathtt{2cell}$, $s : \mathtt{Basic}$, $p : \mathtt{Dcell}$, $h, g : \mathtt{Horizontal}$, $u, v : \mathtt{Vertical}$, $m : \mathtt{Mix}$, $a : \mathtt{Object}$, $ha : \mathtt{Harrow}$, $va : \mathtt{Varrow}$, $hm : \mathtt{Hmix}$, $vm : \mathtt{VMix}$). To conclude the definition of $T_{\mathtt{2EVHCAT}}$, we add a double category structure on the Dcell's. This can be easily done, since all the Dcell are in the image of mk (see Table 7, for all $l', l'' : \mathtt{2cell}$, $h', h'', g', g'', f : \mathtt{Horizontal}$, $v', v'', u', u'', w : \mathtt{Vertical}$). Note that the axiom $t : \mathtt{Object} \leftarrow t : \mathtt{Harrow}, t : \mathtt{Varrow}$ is still valid.

Monoids. For the monoidal extension of the framework, we introduce a theory $T_{\mathtt{MON}}$ of (strict) monoids that has only one sort Monoid a constant 1 and a binary total operator $\otimes$ (which is associative and has 1 as neutral element). Then, the theory of strict monoidal categories is $T_{\mathtt{MONCAT}} = T_{\mathtt{MON}} \otimes T_{\mathtt{CAT}}$, where the sorts and the operators are renamed as in Table 8. The monoidal theories of 2-categories, 2EVH-categories, and double categories are defined by very similar tensor product constructions between $T_{\mathtt{MON}}$ and $T_{\mathtt{2CAT}}$, $T_{\mathtt{2EVHCAT}}$, $T_{\mathtt{DCAT}}$.

Table 7

Ax.	$\mathtt{n}(p) : \mathtt{Harrow}$	$\mathtt{s}(p) : \mathtt{Harrow}$	$\mathtt{w}(p) : \mathtt{Varrow}$	$\mathtt{e}(p) : \mathtt{Varrow}$
	$\mathtt{n}(va) : \mathtt{Object}$	$\mathtt{s}(va) : \mathtt{Object}$	$\mathtt{d}(\mathtt{prj}(p)) = \mathtt{prj}(\mathtt{n}(p)); \mathtt{prj}(\mathtt{e}(p))$	
	$\mathtt{w}(ha) : \mathtt{Object}$	$\mathtt{e}(ha) : \mathtt{Object}$	$\mathtt{c}(\mathtt{prj}(p)) = \mathtt{prj}(\mathtt{w}(p)); \mathtt{prj}(\mathtt{s}(p))$	
Cl.	$\mathtt{n}(\mathtt{mk}(l,h,u,v,g)) = \mathtt{mkh}(h) \leftarrow \mathtt{mk}(l,h,u,v,g) : \mathtt{Dcell}$			
	$\mathtt{s}(\mathtt{mk}(l,h,u,v,g)) = \mathtt{mkh}(g) \leftarrow \mathtt{mk}(l,h,u,v,g) : \mathtt{Dcell}$			
	$\mathtt{w}(\mathtt{mk}(l,h,u,v,g)) = \mathtt{mkv}(v) \leftarrow \mathtt{mk}(l,h,u,v,g) : \mathtt{Dcell}$			
	$\mathtt{e}(\mathtt{mk}(l,h,u,v,g)) = \mathtt{mkv}(u) \leftarrow \mathtt{mk}(l,h,u,v,g) : \mathtt{Dcell}$			
	$\mathtt{mk}(\mathtt{prj}(p),h,u,v,g) = p \leftarrow$ $h = \mathtt{prj}(\mathtt{n}(p)), u = \mathtt{prj}(\mathtt{e}(p)), v = \mathtt{prj}(\mathtt{w}(p)), g = \mathtt{prj}(\mathtt{s}(p))$			
	$\mathtt{mk}(l',h',u,v,g') * \mathtt{mk}(l'',h'',w,u,g'') : \mathtt{Dcell} \leftarrow$ $\mathtt{mk}(l',h',u,v,g') : \mathtt{Dcell}, \mathtt{mk}(l'',h'',w,u,g'') : \mathtt{Dcell}$			
	$\mathtt{mk}(l',h,u',v',g) \cdot \mathtt{mk}(l'',g,u'',v'',f) : \mathtt{Dcell} \leftarrow$ $\mathtt{mk}(l',h,u',v',g) : \mathtt{Dcell}, \mathtt{mk}(l'',g,u'',v'',f) : \mathtt{Dcell}$			
	$\mathtt{mkh}(h') * \mathtt{mkh}(h'') : \mathtt{Harrow} \leftarrow \mathtt{r}(h') = \mathtt{l}(h'')$			
	$\mathtt{mkv}(v') \cdot \mathtt{mkv}(v'') : \mathtt{Varrow} \leftarrow \mathtt{r}(v') = \mathtt{l}(v'')$			
	$\mathtt{mk}(l',h',u,v,g') * \mathtt{mk}(l'',h'',w,u,g'') =$ $\mathtt{mk}((h';l'') \circ (l';g''), (h';h''), w, v, (g';g'')) \leftarrow$ $\mathtt{mk}(l',h',u,v,g') : \mathtt{Dcell}, \mathtt{mk}(l'',h'',w,u,g'') : \mathtt{Dcell}$			
	$\mathtt{mk}(l',h,u',v',g) \cdot \mathtt{mk}(l'',g,u'',v'',f) =$ $\mathtt{mk}((l';u'') \circ (v';l''), h, (u';u''), (v',v''), f) \leftarrow$ $\mathtt{mk}(l',h,u',v',g) : \mathtt{Dcell}, \mathtt{mk}(l'',g,u'',v'',f) : \mathtt{Dcell}$			

Table 8

sorts:	$(\mathtt{Monoid}, \mathtt{Object})$ to $\mathtt{Object}$		$(\mathtt{Monoid}, \mathtt{Arrow})$ to $\mathtt{Arrow}$		
operations:	$\mathtt{1}^l$ to $\mathtt{1}$	$\_ \otimes^l \_$ to $\_ \otimes \_$	$\mathtt{d}^r$ to $\mathtt{d}$	$\mathtt{c}^r$ to $\mathtt{c}$	$\_ ;^r \_$ to $\_ ; \_$

Table 9

Ax.	$\mathtt{sym}(a,b) : \mathtt{Arrow}$	$\mathtt{d}(\mathtt{sym}(a,b)) = a \otimes b$
	$\mathtt{sym}(a,b); \mathtt{sym}(b,a) = a \otimes b$	$\mathtt{c}(\mathtt{sym}(a,b)) = b \otimes a$
	$\mathtt{sym}(a, b \otimes c) = (a \otimes \mathtt{sym}(b,c)); (\mathtt{sym}(a,c) \otimes b)$	
Cl.	$(t \otimes t'); \mathtt{sym}(b,b') = \mathtt{sym}(a,a'); (t' \otimes t) \leftarrow \mathtt{d}(t) = a, \mathtt{c}(t) = b, \mathtt{d}(t') = a', \mathtt{c}(t') = b'$	

Table 10

Axioms:	$\mathtt{sym}(a,b) : \mathtt{Mix}$
	$\sigma(a,b) = \mathtt{mk}(\mathtt{sym}(a,b), \mathtt{sym}(a,b), (b \otimes a), \mathtt{sym}(a,b), (b \otimes a))$
Clauses:	$\rho(ha, ha') = \mathtt{mk}((\mathtt{prj}(ha) \otimes \mathtt{prj}(ha')); \mathtt{sym}(b,b'), (\mathtt{prj}(ha) \otimes \mathtt{prj}(ha')), \mathtt{sym}(b,b'), \mathtt{sym}(a,a'), (\mathtt{prj}(ha') \otimes \mathtt{prj}(ha))) \leftarrow \mathtt{w}(ha) = a, \mathtt{e}(ha) = b, \mathtt{w}(ha') = a', \mathtt{e}(ha') = b'$
	$\gamma(va, va') = \mathtt{mk}((\mathtt{prj}(va) \otimes \mathtt{prj}(va')); \mathtt{sym}(b,b'), \mathtt{sym}(a,a'), (\mathtt{prj}(va') \otimes \mathtt{prj}(va)), (\mathtt{prj}(va) \otimes \mathtt{prj}(va')), \mathtt{sym}(b,b')) \leftarrow \mathtt{n}(va) = a, \mathtt{s}(va) = b, \mathtt{n}(va') = a', \mathtt{s}(va') = b'$

Symmetries. To define the theory $T_{\mathtt{SYMCAT}}$ of symmetric strict monoidal categories we first include $T_{\mathtt{MONCAT}}$ and then, for each pair a, b of `Objects` we introduce an arrow $\mathtt{sym}(a,b)$, which plays the rôle of the symmetry for a and b. The naturality axiom schema is defined for each pair of arrows t and t', and the coherence axioms identify all the equivalent symmetries. The corresponding axioms and clauses are listed in Table 9 (for all a, a', b, b', c : `Object` and t, t' : `Arrow`). A similar construction applies to 2-categories. The main difference between $T_{\mathtt{SYMCAT}}$ and $T_{\mathtt{SYM2CAT}}$ (apart from the renaming of some operators) is given by the naturality axiom, which applies to generic arrows in $T_{\mathtt{SYMCAT}}$ and to generic cells in $T_{\mathtt{SYM2CAT}}$, i.e., for all l, l' : `Cell`:

$$(l \otimes l'); \mathtt{sym}(b,b') = \mathtt{sym}(a,a'); (l' \otimes l) \leftarrow \mathtt{l}(l) = a, \mathtt{r}(l) = b, \mathtt{l}(l') = a', \mathtt{r}(l') = b'$$

The definition of the theory $T_{\mathtt{SYMDCAT}}$ of SSMD's requires all the axioms for double symmetries (i.e., properties of natural double transformations and coherence), plus some additional axioms induced by the membership logic; it is straightforward but too long to be written here (see [2]). Instead 2EVH-categories give us the opportunity to represent symmetries as arrows of sort `Mix`, and then to define the components γ, ρ and σ of the double symmetries via the `mk` operation. Moreover, we can import the axiomatization of symmetries from $T_{\mathtt{SYM2CAT}}$. Therefore we need only the additional axioms and clauses in Table 10 (for all a, a', b, b' : `Object`, ha, ha' : `Harrow`, va, va' : `Varrow`).

Theorem 16 *The obvious signature morphism* `SE` *from* $T_{\mathtt{SYMDCAT}}$ *to* $T_{\mathtt{SYM2EVHCAT}}$ *mapping sort* `Square` *to* `Dcell` *and for the rest relating homonymous structure is a theory morphism. Moreover* `SE` *is: (1) persistent w.r.t. sorts* `Object`*,* `Harrow`*, and* `Varrow`*, and (2) complete.*

Cartesian theories. The theory $T_{\mathtt{CARTCAT}}$ of cartesian categories includes the axioms and clauses of $T_{\mathtt{SYMCAT}}$ and moreover requires duplicators $\mathtt{dup}(a)$ and dischargers $\mathtt{dis}(a)$ for all a : `Object`. These must also satisfy coherence

Table 11

Ax.	$\mathtt{dup}(a) : \mathtt{Arrow}$	$\mathtt{d}(\mathtt{dup}(a)) = a$	$\mathtt{c}(\mathtt{dup}(a)) = a \otimes a$
	$\mathtt{dup}(1) = 1$	$\mathtt{dup}(a \otimes b) = (\mathtt{dup}(a) \otimes \mathtt{dup}(b)); (a \otimes \mathtt{sym}(a,b) \otimes b))$	
	$\mathtt{dis}(1) = 1$	$\mathtt{dis}(a \otimes b) = \mathtt{dis}(a) \otimes \mathtt{dis}(b)$	$\mathtt{dup}(a); \mathtt{sym}(a,a) = \mathtt{dup}(a)$
	$\mathtt{dis}(a) : \mathtt{Arrow}$	$\mathtt{dup}(a); (\mathtt{dup}(a) \otimes a) = \mathtt{dup}(a); (a \otimes \mathtt{dup}(a))$	
	$\mathtt{d}(\mathtt{dis}(a)) = a$	$\mathtt{c}(\mathtt{dis}(a)) = 1$	$\mathtt{dup}(a); (a \otimes \mathtt{dis}(a)) = a$
Cl.	$t; \mathtt{dup}(b) = \mathtt{dup}(a); (t \otimes t) \leftarrow \mathtt{d}(t) = a, \mathtt{c}(t) = b$		
	$t; \mathtt{dis}(b) = \mathtt{dis}(a) \leftarrow \mathtt{d}(t) = a, \mathtt{c}(t) = b$		

and naturality as expressed by the axioms and clauses in Table 11 (for all $a : \mathtt{Object}$ and $t : \mathtt{Arrow}$). Then, the theory of cartesian 2-categories can be defined by just adding the two naturality axioms on cells (for `dup` and `dis`). The theory $T_{\mathtt{CARTDCAT}}$ of CDC's requires all the axioms of Def. 6 (see [2]). However, as for $T_{\mathtt{SYMDCAT}}$, it can be conservatively derived from the corresponding theory $T_{\mathtt{CART2EVHCAT}}$ of cartesian 2EVH-categories. The latter can be easily defined pursuing the analogy with the definition of $T_{\mathtt{SYM2EVHCAT}}$ (see Table 10).

Theorem 17 *The obvious signature morphism* `CE` *from* $T_{\mathtt{CARTDCAT}}$ *to* $T_{\mathtt{CART2EVHCAT}}$ *mapping sort* `Square` *to* `Dcell` *and for the rest relating the homonymous sorts and operators is a theory morphism. Moreover* `CE` *is: (1) persistent w.r.t. sorts* `Object`*,* `Harrow`*,* `Varrow` *and (2) complete.*

3.2.1 Computads.

The notion of *computad* [25] provides a finitary structure, able to generate a double category via a free construction. Therefore, in some sense, a computad for tile logic provides a formalization of *tile theories*. Informally, a *computad* is a triple $\langle H, V, T\rangle$, where H and V are categories with the same set of objects O, and T is a set of tiles, each of which has assigned two pairs of arrows, in H and V, as vertical and horizontal source and target, respectively. A computad is *cartesian* (*symmetric*) if both H and V are cartesian categories (symmetric strict monoidal categories). We explain the relevant part of the theory for the cartesian case only ($T_{\mathtt{CARTCTD}}$). Two disjoint cartesian structures are imported to model horizontal and vertical arrows, sharing the objects. Also the two monoidal operators are different, except when applied to objects, but they will be identified when generating the associated double category. The results presented in [22] for the monoidal case, can then be extended to the symmetric and cartesian cases. In particular we have the theory morphisms $\mathtt{SD} : T_{\mathtt{SYMCTD}} \to T_{\mathtt{SYMDCAT}}$ and $\mathtt{CD} : T_{\mathtt{CARTCTD}} \to T_{\mathtt{CARTDCAT}}$ that can be composed with morphisms `SE` and `CE`, respectively to obtain $\mathtt{SVH} : T_{\mathtt{SYMCTD}} \to T_{\mathtt{SYM2EVHCAT}}$ and $\mathtt{CVH} : T_{\mathtt{CARTCTD}} \to T_{\mathtt{CART2EVHCAT}}$.

Proposition 18 *The forgetful functor* $U_{\mathtt{CD}} : \mathbf{PAlg}_{T_{\mathtt{CARTDCAT}}} \to \mathbf{PAlg}_{T_{\mathtt{CARTCTD}}}$ *has a left adjoint* $F_{\mathtt{CD}}$*, and analogously* $F_{\mathtt{SD}} \dashv U_{\mathtt{SD}} : \mathbf{PAlg}_{T_{\mathtt{SYMDCAT}}} \to \mathbf{PAlg}_{T_{\mathtt{SYMCTD}}}$*. Similarly, the forgetful functor* $U_{\mathtt{CVH}} : \mathbf{PAlg}_{T_{\mathtt{CART2EVHCAT}}} \to \mathbf{PAlg}_{T_{\mathtt{CARTCTD}}}$ *has a left adjoint* $F_{\mathtt{CVH}}$*, and also* $F_{\mathtt{SVH}} \dashv U_{\mathtt{SVH}} : \mathbf{PAlg}_{T_{\mathtt{SYM2EVHCAT}}} \to \mathbf{PAlg}_{T_{\mathtt{SYMCTD}}}$*. Furthermore,* $F_{\mathtt{CVH}} = F_{\mathtt{CD}}; F_{\mathtt{CE}}$ *and* $F_{\mathtt{SVH}} = F_{\mathtt{SD}}; F_{\mathtt{SE}}$*.*

Tile Systems and Computads. We can now formalize the correspondence between TTL and CDC's (analogous result hold between PTL and SSMD's). We first show how to translate a generic TTS into a computad. As an important result the free cartesian double category arising from the computad entails the same sequents as the TTL associated to the TTS. Considering flat sequents, an analogous result holds for the free 2EVH-category generated from the same computad (this is not necessarily true if proof terms are considered).

Definition 19 *Let* $\mathcal{R} = \langle \Sigma_H, \Sigma_V, N, R \rangle$ *be a* TTS*. The associated cartesian computad* $Ctd(\mathcal{R})$ *is the triple* $\langle T_{\Sigma_H}(X), T_{\Sigma_V}(X), T_R \rangle$*, where the set of tiles* T_R *is such that*

$$\begin{tikzcd} a \arrow[r, "h"] \arrow[d, "v"'] \arrow[dr, phantom, "r"] & b \arrow[d, "u"] \\ c \arrow[r, "g"'] & d \end{tikzcd} \in T_R \quad \Longleftrightarrow \quad r : h \xrightarrow[u]{v} g \in \mathcal{R}.$$

Definition 20 *Given a* TTS $\mathcal{R}$*, the* cartesian tile logic *of* $\mathcal{R}$ *is the* CDC $\mathcal{L}_{\mathcal{D}}(\mathcal{R}) = F_{\mathtt{CD}}(Ctd(\mathcal{R}))$ *freely generated from the computad* $Ctd(\mathcal{R})$ *by the functor* $F_{\mathtt{CD}}$ *described in Prop. 18. For* $\alpha \in F_{\mathtt{CD}}(Ctd(\mathcal{R}))_{\mathtt{Square}}$ *we write* $\mathcal{R} \vdash_{\mathrm{c}} \alpha$ *(the notation* $\vdash_{\mathrm{fc}}$ *is used for flat sequents).*

Definition 21 *Given a* TTS $\mathcal{R} = \langle \Sigma_H, \Sigma_V, N, R \rangle$ *the* extended logic *of* $\mathcal{R}$ *is the cartesian 2EVH-category* $F_{\mathtt{CVH}}(Ctd(\mathcal{R})) = F_{\mathtt{CE}}(F_{\mathtt{CD}}(Ctd(\mathcal{R})))$ *freely generated from* $Ctd(\mathcal{R})$ *by the functor* $F_{\mathtt{CVH}}$ *of Prop. 18. For* $\beta \in F_{\mathtt{CVH}}(Ctd(\mathcal{R}))_{Dcell}$ *we write* $\mathcal{R} \vdash_{\mathrm{e}} \beta$ *(the notation* $\vdash_{\mathrm{fe}}$ *is used for flat sequents).*

Theorem 22 *Given a* TTS $\mathcal{R}$*, then* $\mathcal{R} \vdash_{\mathrm{t}} \alpha \Longleftrightarrow \mathcal{R} \vdash_{\mathrm{c}} \alpha$*.*

Corollary 23 *Given a* TTS $\mathcal{R}$*, then* $\mathcal{R} \vdash_{\mathrm{fc}} h \xrightarrow[u]{v} g \Longleftrightarrow \mathcal{R} \vdash_{\mathrm{fe}} h \xrightarrow[u]{v} g$*.*

Uniform Systems. The relevance of the results presented in this section, is that we can use a rewriting logic framework to entail the same flat sequents which are entailed in TTL. Since there are several available languages designed for dealing with rewriting logic specifications, we can actually build tools which deal with TTL as well (completely analogous results hold for PTL). The constraint imposed on the extended logic, namely that the rewrite proof must be a `Dcell`, can be enforced by the meta-level of the rewriting system using suitable internal strategies. Moreover if the TTS is *uniform* then the internal strategy becomes very simple, since it suffices to check if a given state can be decomposed as a vertical arrow v followed by a horizontal arrow g [4].

$$
\begin{tikzcd}
 & \mathbf{PAlg}_{T_{\mathtt{SYM2EVHCAT}}} \arrow[dl, shift left, "U_{\mathtt{SVH}}"] \arrow[d, shift left, "U_{\mathtt{SE}}"] \\
\mathbf{PAlg}_{T_{\mathtt{SYMCTD}}} \arrow[ur, shift left, "F_{\mathtt{SVH}}"] \arrow[ur, phantom, "\perp"] \arrow[r, shift left, "F_{\mathtt{SD}}"] \arrow[r, phantom, "\perp"] & \mathbf{PAlg}_{T_{\mathtt{SYMDCAT}}} \arrow[l, shift left, "U_{\mathtt{SD}}"] \arrow[u, shift left, "F_{\mathtt{SE}}"] \arrow[u, phantom, "\dashv"]
\end{tikzcd}
\qquad
\begin{tikzcd}
 & \mathbf{PAlg}_{T_{\mathtt{CART2EVHCAT}}} \arrow[dl, shift left, "U_{\mathtt{CVH}}"] \arrow[d, shift left, "U_{\mathtt{CE}}"] \\
\mathbf{PAlg}_{T_{\mathtt{CARTCTD}}} \arrow[ur, shift left, "F_{\mathtt{CVH}}"] \arrow[ur, phantom, "\perp"] \arrow[r, shift left, "F_{\mathtt{CD}}"] \arrow[r, phantom, "\perp"] & \mathbf{PAlg}_{T_{\mathtt{CARTDCAT}}} \arrow[l, shift left, "U_{\mathtt{CD}}"] \arrow[u, shift left, "F_{\mathtt{CE}}"] \arrow[u, phantom, "\dashv"]
\end{tikzcd}
$$

Fig. 5: Computads, double categories and 2EVH-categories.

Definition 24 (Uniform Systems) *A* CDC *(resp. a* SSMD*)* $\mathcal{D}$ *is* uniform *if the 2EVH-category* $F_{\mathtt{CE}}(\mathcal{D})$ *(resp.* $F_{\mathtt{SE}}(\mathcal{D})$*) satisfies the membership clause:*

$$\mathtt{mk}(A, h, u, v, g) : \mathtt{Dcell} \leftarrow \mathtt{d}(A) = h; u, \mathtt{c}(A) = v; g,$$

for all $A : \mathtt{2cell}$*,* $h, g : \mathtt{Horizontal}$*,* $v, u : \mathtt{Vertical}$*. A* TTS *(resp.* PTS*)* $\mathcal{R}$ *is* uniform *if its associated double category is uniform.*

If the TTS is not uniform, then also derivation proofs must be taken into account. Consequently, the computations would be affected becoming very slow. However, at present we do not have any meaningful example of non-uniform systems.

4 Conclusion

Our work is a natural continuation of the line of research initiated by two of the authors in [22]. The aim is to investigate the relationship between two very general models of computation, rewriting logic and tile logic: RL extends the algebraic semantics approach to concurrent systems with state changes and is the basis of several existing languages; on the other hand, TL offers a natural framework for interactive systems.

To compare TL and RL, we have employed **PMEqtl** techniques. As suggested in the conclusion of [22], we have developed that part of the theory related to the case of symmetric monoidal, and cartesian shared structures, corresponding to TTL and PTL. The results (summarized in Fig. 5) show that: (a) rewriting logic derivation of Dcell's within 2EVH-categories coincides with tile logic derivation within double categories, (b) in general we need a meta-level to control the correctness of the result, and (c) the last constraint can be elegantly solved for uniform systems.

During our research, we have experimented with Maude, a logic language equipped with reflective features, which gives the possibility of meta-programming internal execution strategies for rewriting. In particular, we have implemented several uniform tile specifications of CCS-like process algebras [4–6].

References

[1] P. Borovanský, C. Kirchner, H. Kirchner, P.-E. Moreau, and M. Vittek, ELAN: A logical framework based on computational systems, in: J. Meseguer, Ed., *Proc. WRLA'96*, *ENTCS* **4** (1996).

[2] R. Bruni, Tile Logic for Synchronized Rewriting of Concurrent Systems, PhD Thesis **TD-99-01**, Computer Science Deptartment, University of Pisa (1999).

[3] R. Bruni, J. Meseguer, and U. Montanari, Process and term tile logic, Technical Report SRI-CSL-98-06, SRI International (1998). Also Technical Report TR-98-09, Department of Computer Science, University of Pisa (1998).

[4] R. Bruni, J. Meseguer, and U. Montanari, Internal strategies in a rewriting implementation of tile systems, in *Proc. WRLA'98*, *ENTCS* **15** (1998).

[5] R. Bruni, J. Meseguer, and U. Montanari, Implementing tile systems: Some examples from process calculi, in *Proc. ICTCS'98*, World Scientific (1998).

[6] R. Bruni, J. Meseguer, and U. Montanari, Executable Tile Specifications for Process Calculi, in *Proc. of FASE'99*, *LNCS* **1577**, 60–76 (1999).

[7] R. Bruni, J. Meseguer, and U. Montanari, Symmetric and Cartesian Double Categories as a Semantic Framework for Tile Logic, *MSCS*, to appear.

[8] M. Clavel, *Reflection in General Logics and in Rewriting Logic with Applications to the Maude Language*, PhD Thesis. Universidad de Navarra (1998).

[9] M.G. Clavel, F. Duran, S. Eker, P. Lincoln, and J. Meseguer, *An Introduction to Maude (Beta Version)*, SRI International (1998).

[10] M. Clavel and J. Meseguer, Internal strategies in a reflective logic, in *Proc. CADE-14 Workshop on Strategies in Automated Deduction*, 1–12 (1997).

[11] P. Degano, J. Meseguer, and U. Montanari, Axiomatizing the algebra of net computations and processes, *Acta Informatica* **33**(7), 641–667 (October 1996).

[12] R. Diaconescu and K. Futatsugi, CafeOBJ *Report*, *AMAST Series in Computing* **6**, World Scientific (1998).

[13] C. Ehresmann, Catégories structurées: I and II, *Ann. Éc. Norm. Sup.* **80**, 349–426, Paris (1963); III, *Topo. et Géo. Diff.* **5**, Paris (1963).

[14] G.L. Ferrari and U. Montanari, Tile formats for located and mobile systems, *Information and Computation*, to appear.

[15] F. Gadducci and U. Montanari, The tile model, in: G. Plotkin, C. Stirling, and M. Tofte, Eds., *Proof, Language and Interaction: Essays in Honour of Robin Milner*, MIT Press, to appear.

[16] G.M. Kelly and R.H. Street, Review of the elements of 2-categories, *Lecture Notes in Mathematics* **420**, 75–103, (1974).

[17] A. Kock and G.E. Reyes, Doctrines in categorical logic, in: John Barwise, Ed., *Handbook of Mathematical Logic*, North Holland, 283–313 (1977).

[18] F.W. Lawvere, Functorial semantics of algebraic theories, *Proceedings National Academy of Science of the United States of America* **50**(1), 869–872 (1963).

[19] N. Martí-Oliet and J. Meseguer, General logics and logical frameworks, in: D. Gabbay, Ed., *What is a logical system?*, 355–392, Oxford Univ. Press (1994).

[20] J. Meseguer, Conditional rewriting logic as a unified model of concurrency, *Theoretical Computer Science* **96**, 73–155 (1992).

[21] J. Meseguer, Membership equational logic as a logical framework for equational specification, in *Proc. 12th WADT'97*, *LNCS* **1376**, 18–61 (1998).

[22] J. Meseguer and U. Montanari, Mapping tile logic into rewriting logic, in *Proc. 12th WADT'97*, *LNCS* **1376**, 62–91 (1998).

[23] M. Pfender, Universal algebra in S-monoidal categories, Algebra-berichte **20**, Department of Mathematics, University of Munich (1974).

[24] G. Plotkin, A structural approach to operational semantics, Technical Report DAIMI FN-19, Computer Science Department, Aarhus University (1981).

[25] R. Street, Categorical structures, in: M. Hazewinkel, Ed., *Handbook of Algebra*, 529–577, Elsevier (1996).

A Symmetric and Algebraic Theories

We recall that a *many-sorted hypersignature* Σ over a set S of sorts is a family $\{\Sigma_{a,b}\}_{a,b\in S^*}$ of sets of operators, where each operator $f \in \Sigma_{a,b}$ takes $|a|$ arguments typed accordingly to a, and returns a tuple of $|b|$ values, typed accordingly to b. A *many-sorted signature* is a hypersignature where $\Sigma_{a,b} \neq \emptyset \Rightarrow b \in S$. If S is a singleton set, then Σ is called *one-sorted*.

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 $\partial_0, \partial_1 : A_G \to O_G$ are functions, called respectively *source* and *target*. We use the standard notation $f : a \to b$ to denote an arrow f with source a and target b. A graph G is *reflexive* if there exists an *identity*-forming function $id : O_G \to A_G$ such that $\partial_0(id(a)) = a = \partial_1(id(a))$, for all $a \in O_G$; it is called *with pairing* if O_G is a monoid; and it is called *monoidal* if it is reflexive, both O_G and A_G are monoids, and the functions ∂_0,∂_1, and id are monoid homomorphisms.

It is immediate that a many-sorted hypersignature Σ over S can be seen as a graph with pairing G_Σ such that its objects are strings on S (i.e., $O_{G_\Sigma} = S^*$, the monoidal operator is string concatenation, and the empty string λ is the unit), and its arcs are the operators in Σ (i.e., $f : a \to b \in A_{G_\Sigma}$ iff $f \in \Sigma_{a,b}$).

$$(generators)\ \frac{f \in \Sigma_{a,b}}{f : a \to b \in \mathbf{S}(\Sigma)} \qquad (pairing)\ \frac{t : a \to b,\ t' : a' \to b' \in \mathbf{S}(\Sigma)}{t \otimes t' : a \otimes a' \to b \otimes b' \in \mathbf{S}(\Sigma)}$$

$$(identities)\ \frac{a \in S^*}{id_a : a \to a \in \mathbf{S}(\Sigma)} \qquad (composition)\ \frac{t : a \to b,\ t' : b \to c \in \mathbf{S}(\Sigma)}{t; t' : a \to c \in \mathbf{S}(\Sigma)}$$

$$(symmetries)\ \frac{a, b \in S^*}{\gamma_{a,b} : a \otimes b \to b \otimes a \in \mathbf{S}(\Sigma)}$$

Fig. A.1: Inference rules for symmetric theories.

$$(duplicators)\ \frac{a \in S^*}{\nabla_a : a \to a \otimes a \in \mathbf{A}(\Sigma)} \qquad (dischargers)\ \frac{a \in S^*}{!_a : a \to \lambda \in \mathbf{A}(\Sigma)}$$

Fig. A.2: Additional inference rules for algebraic theories.

Table A.1

$\nabla_{a\otimes b} = (\nabla_a \otimes \nabla_b); (id_a \otimes \gamma_{a,b} \otimes id_b)$	$\nabla_\lambda = id_\lambda = !_\lambda$	$!_{a\otimes b} = !_a \otimes !_b$
$\nabla_a; (1_a \otimes \nabla_a) = \nabla_a; (\nabla_a \otimes 1_a)$	$\nabla_a; \gamma_{a,a} = \nabla_a$	$\nabla_a; (1_a \otimes !_a) = id_a$

Definition 25 (Symmetric Theory) *Given a hyper-signature Σ, its* symmetric theory $\mathbf{S}(\Sigma)$ *is the monoidal*[1] *graph with objects the elements of the free monoid on S, and arrows those generated by the inference rules in Fig. A.1. Moreover, the operator $\_;\_$ is associative, and the arrows in $\mathbf{S}(\Sigma)$ satisfy*

- *the* identity axioms $id_a; t = t = t; id_b$ *for all* $t : a \to b$,
- *the* functoriality axiom $(s \otimes t); (s' \otimes t') = (s; s') \otimes (t; t')$, *when both* $s; s'$ *and* $t; t'$ *are defined,*
- *the* naturality axiom $(t \otimes t'); \gamma_{b,b'} = \gamma_{a,a'}; (t' \otimes t)$ *for all* $t : a \to b$ *and* $t' : a' \to b'$, *and*
- *the* coherence axioms $\gamma_{a\otimes b,c} = (id_a \otimes \gamma_{b,c}); (\gamma_{a,c} \otimes id_b)$ *and* $\gamma_{a,b}; \gamma_{b,a} = id_{a\otimes b}$ *for all* $a, b, c \in S^*$.

Symmetries represent all the possible permutations of the elements in the strings on S that are the "interfaces" of the modelled system. Hence their usage allows connecting the generators in all possible ways, always getting an acyclic graph where each node has at most one incoming/outgoing hyperarc. Notice that complex symmetries are defined in terms of smaller ones; indeed all symmetries are generated from the basic arrows $\gamma_{s,s'}$ for all $s, s' \in S$.

A further enrichment adds duplicators and dischargers and provides a cartesian structure. In this case it is no longer necessary to start with hypersignatures, since the same arrows can be expressed in terms of suitable signatures.

[1] Monoidality implies that id_λ is the neutral element of the monoid of arrows, and that the *monoidality* axiom $id_{a\otimes b} = id_a \otimes id_b$ holds for all $a, b \in S^*$.

Definition 26 (Algebraic Theory) *Given a signature* Σ*, the associated* algebraic theory $\mathbf{A}(\Sigma)$ *is the monoidal graph generated by the same inference rules and axioms given for* $\mathbf{S}(\Sigma)$*, together with the inference rules in Figure A.2. Moreover, the arrows in* $\mathbf{A}(\Sigma)$ *verify the additional*

- naturality axioms $t; \nabla_b = \nabla_a; (t \otimes t)$ *and* $t; !_b = !_a$ *for all* $t : a \to b$*,*
- coherence axioms *in Table A.1 (for all* $a, b \in S^*$*).*

A classical result states the correspondence of $\mathbf{A}(\Sigma)$ with the Lawvere–Benabou algebraic theory over Σ and, by looking at hom functors, with finitely generated free Σ-algebras. We believe that the constructive definition of algebraic theories separates very nicely the auxiliary structure from the Σ-structure, better than the ordinary description involving the meta-operation of substitution, which is here represented by sequential composition.

Proposition 27 *Let* Σ *be a signature. Then, for all* $a, b \in S^*$*, there exists a one-to-one correspondence between the set* $\mathbf{A}(\Sigma)[a, b]$ *and the* $|b|$*-tuples of elements of* $T_\Sigma(X)$ *(typed accordingly to b) over an ordered set* $X = \{x_1, ..., x_{|a|}\}$ *of variables, typed accordingly to a (i.e., such that* $x_i : a_i$ *for all* $i \in \{1, .., |a|\}$*).*