Open ended systems, dynamic bisimulation and tile logic

Roberto Bruni[1], Ugo Montanari[1] and Vladimiro Sassone[2]

[1] Dipartimento di Informatica, Università di Pisa, Italia.
[2] Dipartimento di Matematica e Informatica, Università di Catania, Italia.
bruni@di.unipi.it, ugo@di.unipi.it, vs@dmi.unict.it

Abstract The SOS formats ensuring that bisimilarity is a congruence often fail in the presence of structural axioms on the algebra of states. Dynamic bisimulation, introduced to characterize the coarsest congruence for CCS which is also a (weak) bisimulation, reconciles the bisimilarity as congruence property with such axioms and with the specification of open ended systems, where states can be reconfigured at run-time, at the cost of an infinitary operation at the meta-level. We show that the compositional framework offered by tile logic is suitable to deal with structural axioms and open ended systems specifications, allowing for a finitary presentation of context closure.
Keywords: Bisimulation, SOS formats, dynamic bisimulation, tile logic.

Introduction

The semantics of dynamic systems can be conveniently expressed via labelled transition systems (LTS) whose states are terms over a certain algebra and whose labels describe some abstract behavioral information. Provided such information models the possible interactions between various components, the framework yields a compositional semantics. Plotkin's *structured operational semantics* (SOS) [23] is one of the most successful such frameworks, where the transitions a system can perform are defined by recursion on the structure of its states.

Several notions of equivalence on the state space of LTSs have been considered in the literature that take into account particular aspects. For example, if one is interested only in the action sequences performed by the system, then one should observe *traces*, whereas in a truly concurrent approach one would rather observe *partial orderings* of actions. State equivalences can then be defined on the basis of the chosen observables. In this paper we consider *bisimulation* equivalences [18,22] (with *bisimilarity* meaning the maximal bisimulation), where the entire branching structure of the transition system is accounted for: two states are equivalent if whatever transition one can perform, the other can simulate it via a transition with the same observation, still ending in equivalent states.

Research supported by CNR Integrated Project *Progettazione e Verifica di Sistemi Eterogenei Connessi mediante Reti*; by Esprit Working Groups *CONFER2* and *COORDINA*; by TMR Network *GETGRATS*; and by MURST project *TOSCA*.

One of the main advantages of a compositional semantics is that each subcomponent of a state can be safely replaced by any equivalent subcomponent without affecting the overall behavior. This triggered many efforts devoted to the study of SOS *formats* whose syntactic constraints guarantee that *bisimilarity is a congruence*. Among the most popular such formats, we mention the simple *De Simone* format [10], the more general *positive* GSOS format [3], and the 'liberal' family of `tyft` formats [13,1] (`tyxt`, `zyft`, *promoted* `tyft`,...). Yet, in many interesting cases the use of these formats is not straightforward. For instance, although it is often convenient to have some structural axioms on states, these cannot be handled by ordinary formats. Fall in this case several semantic descriptions based on the *chemical abstract machine* [2], where operators are often assumed to be associative, commutative and with unit, as e.g., the parallel composition of CCS. Further examples are given by systems modeled using UML graphs, that must be taken up to suitable isomorphisms and therefore require the LTS to be defined on suitable structural equivalence classes. Also the (finite) π-calculus [19] is defined by rules in good format when substitution is explicit, but agents are subject to substitution axioms.

It is therefore often necessary to resort to the largest congruence included in the bisimilarity by an operation of closure 'for all contexts', which results in a congruence which is no longer a bisimulation. *Dynamic bisimulation* [20], instead, performs such a context closure during the bisimulation 'game'. Dynamic bisimilarity was shown to capture the coarsest equivalence for CCS agents among weak bisimulations that are also congruences, being completely axiomatized by the axioms of strong observational equivalence plus two of the three Milner's τ-laws. The basic idea is to allow at every step of bisimulation not only the execution of an action, but also the embedding of the two states under comparison within the same, but otherwise arbitrary, context. It is worth remarking that such dynamical contextual embedding has a natural interpretation in terms of dynamic reconfiguration of the system, and hence can find many application in practice for modeling *open ended systems*. Its main drawback is the lack of a convenient representation at the level of the SOS rules; it rather has the spirit of a 'meta' construction, involving a *universal* quantification on contexts. An interesting approach – in the style of Sewell's work on defining LTS from reduction systems [24] – would be to enrich the transition system with all (unary) contexts as labels, and all transitions $p \xrightarrow{C[\_]} C[p]$ for any state p and context $C[\_]$. This solution, however, would still be at the meta-level and infinitary in principle.

In this paper we propose to recast dynamic bisimulation inside the *tile model*, where it can be finitely modeled. The tile model [12] is a formalism for modular descriptions of the dynamic evolution of concurrent systems. It relies on a form of rewrite rules with side effects, called *basic tiles*, which are reminiscent of both SOS rules and *context systems* [14], collecting intuitions coming from *structured transition systems* [9] and *rewriting logic* [16]. In particular, by analogy with rewriting logic, the tile model can be defined employing a logical presentation, called *tile logic*, where tiles are (decorated) sequents subject to inference rules.

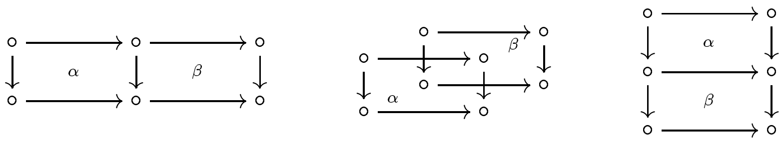

Figure 1. Horizontal, parallel and vertical tile compositions.

Tile logic. Tile logic extends rewriting logic (in the unconditional case) by taking into account state changes with side effects and rewriting synchronization. Basically, a set of rules describes the behaviour of certain (partially specified, in the sense that can contain variables) *configurations*, which may interact through their *interfaces*. Then, the behaviour of a system as a whole consists of a coordinated evolution of its sub-systems. The name 'tile' is due to the graphical representation of such rules, which have the form

$$\begin{tikzcd} \textit{initial input interface} \circ \arrow[r, "s"] \arrow[d, "a"'] \arrow[dr, phantom, "\alpha"] & \circ \textit{ initial output interface} \arrow[d, "b"] \\ \textit{final input interface} \circ \arrow[r, "s'"'] & \circ \textit{ final output interface} \end{tikzcd}$$

also written $\alpha : s \xrightarrow[b]{a} s'$, stating that the *initial configuration* s of the system evolves to the *final configuration* s' via the tile α, producing the *effect* b, which can be observed by the rest of the system. However, such a step is allowed only if the subcomponents of s (i.e., the arguments of the *context* s) evolve to the subcomponents of s', producing the effect a, which acts as the *trigger* of α. Triggers and effects are called *observations* and tile vertices are called *interfaces*. The arrows s, a, b and s' give the *border* of α.

Tiles are a natural model for reactive systems that are *compositional in space* – the behaviour of structured states can be defined as a coordination of the activities of the subcomponents – and *compositional in time*, as they offer a powerful framework for modelling the composition of computations. Indeed, tiles can be composed horizontally, vertically, and in parallel to generate larger steps. The operation of parallel composition corresponds to building concurrent steps, where two (or more) disjoint configurations can concurrently evolve. Of course, the border of a concurrent step is the parallel composition of the borders of each component of the step. Horizontal composition yields rewriting synchronization: the effect of the first tile provides the trigger for the second tile, and the resulting tile expresses the synchronized behaviour of both. Vertical composition models the execution of a sequence of steps starting from an initial configuration. It corresponds to sequential composition of computations. The three compositions are illustrated in Figure 1.

Given a set of basic tiles, the associated tile logic is obtained by adding some canonical 'auxiliary' tiles and then closing by composition – horizontally, vertically, and in parallel – both auxiliary and basic tiles. As an example, auxiliary tiles may be introduced that accommodate isomorphic transformations of interfaces, yielding consistent rearrangements of configurations and observations.

Tile logic deals with algebraic structures on configurations that can be different from the ordinary, tree-like presentation of terms, as e.g., term graphs, partitions, preorders, relations, net processes. In fact, all these structures give rise to monoidal categories and, therefore, possess the two basic operations needed by tile configurations – the monoidal tensor product gives the parallel composition and the sequential composition of the category represents the application of a context to its arguments. In this paper we shall use (monoidal) categories in a very elementary way, i.e., just for representing terms and substitution diagrammatically as abstract arrows (from the arguments to the result). In particular we shall consider neither the categorical models of tile logic (expressed by suitable *monoidal double categories* [12,6]), nor the axiomatized proof terms decorating tile sequents. Varying the algebraic structure of configurations and observations tiles can model many different aspects of dynamic systems, ranging from synchronization of net transitions [7], to causal dependencies for located calculi and finitely branching approaches for name-passing calculi [11], to actor systems [21]. In addition, tile logic allows one to reason about terms with variables as Larsen and Xinxin's context systems [14], while SOS formats work for ground terms only.

Several formats for tiles have been defined where the structure of configurations is given by the term algebra over a signature. Namely, (1) the *monoidal tile format* [17], which has monoidal structures of both configurations and observations; (2) the *algebraic tile format* [12], which has a cartesian structure of configurations but only a monoidal structure of observations; and (3) the *term tile format* [6,4], which has cartesian structures of both configurations and observations. Although none of them ensures that tile bisimilarity is a congruence, by restricting these formats one can easily recover either the De Simone, or the positive GSOS, or the `zyft` format. Here, we shall focus only on the monoidal and term tile formats, showing that it is always possible to manage dynamic bisimulation via ordinary tile bisimulation by extending the vertical signature with a finite number of operators determined from the signature of configurations. In particular, for each operator f of the horizontal signature we shall add the observation $\tilde{f}$ and the tile

$$\begin{tikzcd} \cdot \arrow[r, "id"] \arrow[d, "id"'] & \cdot \arrow[d, "\tilde{f}"] \\ \cdot \arrow[r, "f"'] & \cdot \end{tikzcd}$$

where id denotes identity in the appropriate category. Such a tile can then be applied to any configuration, embedding it in context f and producing effect $\tilde{f}$, which can now be observed in the ordinary bisimulation game. As we shall see, the congruence proof for bisimilarity in the enriched systems can be carried out as an abstract tile pasting. Moreover, such bisimilarity coincides with the dynamic bisimilarity on the original system.

The idea of allowing contexts as observations has been at the basis of the *promoted* `tyft/tyxt` format [1], designed for dealing with higher order languages. We think that such an extension can be carried out in a natural way in the abstract framework provided by tile logic.

Structure of the paper. In Section 1 we fix the notation, recall the most diffused rule formats for transition system specifications, and motivate dynamic bisimulation. In Section 2 we summarize the tile formats which we focus on. Section 3 introduces syntactical constraints on basic tiles that guarantee the 'bisimilarity as a congruence' property. Section 4 presents our main results: a finitary presentation of dynamic bisimulation via monoidal and term tile systems.

1 Bisimulation and SOS Formats

The notion of bisimulation dates back to the pioneering work of David Park and Robin Milner [18,22] on process algebras and provides the standard framework to express behavioural equivalences of complex dynamical systems.

Definition 1. *A* labeled transition system *(*LTS*) is a triple $L = (S, \Lambda, \rightarrow)$, where S is a set of* states*, Λ is a set of* labels*, and $\rightarrow$ is a ternary relation*

$$\rightarrow \subseteq S \times \Lambda \times S$$

We let $s, t, s', t' \ldots$ range over S and $a, b, c, \ldots$ range over Λ. For $\langle s, a, s' \rangle \in \rightarrow$ we use the notation $s \xrightarrow{a} s'$.

Definition 2. *For $L = (S, \Lambda, \rightarrow)$ a* LTS*, a* bisimulation *on L is a symmetric, reflexive relation $\sim \subseteq S \times S$ such that if $s \sim t$, then for any transition $s \xrightarrow{a} s'$ there exists a transition $t \xrightarrow{a} t'$ with $s' \sim t'$.*

We denote by $\simeq$ the largest bisimulation and call it *bisimilarity*, and we say that two states s and t are *bisimilar* whenever $s \simeq t$ or, equivalently, whenever there exists a bisimulation $\sim$ such that $s \sim t$.

In this paper we shall consider LTS whose states are terms over a given signature Σ. Although our results are easily extended to many-sorted signatures, for simplicity we focus on the one-sorted case. A *one-sorted signature* is a set Σ of *operators* together with an arity function $ar_\Sigma \colon \Sigma \rightarrow \mathbb{N}$ assigning to each operator the number of arguments it takes. The subset of Σ consisting of the operators of arity n is denoted by Σ_n. Operators in Σ_0 are called *constants*. We denote by $\mathbb{T}_\Sigma(X)$ the term algebra over Σ and variables in X (with X and Σ disjoint). We use $\mathbb{T}_\Sigma$ for $\mathbb{T}_\Sigma(\varnothing)$, the *term algebra* over Σ. For $t \in \mathbb{T}_\Sigma(X)$, we write $var(t)$ for the set of variables that appear in t. Term t is said *closed* or also *ground* if $var(t) = \varnothing$. We use a for a constant $a() \in \mathbb{T}_\Sigma(X)$.

A *substitution* is a mapping $\sigma \colon X \rightarrow \mathbb{T}_\Sigma(X)$. It is closed if each variable is mapped into a closed term. Substitutions extend to mappings from terms to terms as usual: $\sigma(t)$ is the term obtained by concurrently replacing all occurrences of variables x in t by $\sigma(x)$. The substitution mapping x_i to t_i for $i \in [1, n]$ is denoted by $[t_1/x_1, \ldots, t_n/x_n]$. Substitution σ' can be applied elementwise to substitution $\sigma = [t_1/x_1, \ldots, t_n/x_n]$ yielding the composed substitution

$$\sigma; \sigma' = \sigma'([t_1/x_1, \ldots, t_n/x_n]) = [\sigma'(t_1)/x_1, \ldots, \sigma'(t_n)/x_n].$$

A *context* $t = C[x_1, ., x_n]$ denotes a term in which at most the distinct variables $x_1, ..., x_n$ appear. The term $C[t_1, ..., t_n]$ is then obtained by applying the substitution $[t_1/x_1, ..., t_n/x_n]$ to $C[x_1, ., x_n]$. A context can therefore be regarded as a function from n terms to 1. Notice that the x_i may as well not appear in $C[x_1, ..., x_n]$. For example, the context $x_2[x_1, x_2, x_3]$ is a substitution from three arguments to one, which is the projection on the second argument.

A term is *linear* if each variable occurs at most once in it. Similarly, $\sigma = [t_1/x_1, ..., t_n/x_n]$ is *linear* if each t_i is linear and $var(t_i) \cap var(t_j) = \varnothing$ for $i \neq j$.

Substitutions and their composition $\_;\_$ form a (cartesian) category $\mathbf{Subs}_\Sigma$, with linear substitutions forming a monoidal subcategory. An alternative presentation of $\mathbf{Subs}_\Sigma$ can be obtained resorting to *algebraic theories*. An algebraic theory [15] is a cartesian category having 'underlined' natural numbers as objects. The free algebraic theory associated to a signature Σ is denoted by $\mathbf{Th}[\Sigma]$: the arrows from $\underline{m}$ to $\underline{n}$ are in a one-to-one correspondence with n-tuples of terms of the free Σ-algebra with (at most) m variables, and composition of arrows is term substitution. In particular, $\mathbf{Th}[\Sigma]$ is isomorphic to $\mathbf{Subs}_\Sigma$, and the arrows from $\underline{0}$ to $\underline{1}$ are in bijective correspondence with the closed terms over Σ. As a matter of notation, we assume a standard naming of the $\underline{m}$ input variables, namely $x_1, ..., x_m$. When composing two arrows $\vec{s}: \underline{m} \to \underline{k}$ and $\vec{t}: \underline{k} \to \underline{n}$, the resulting term $\vec{s}; \vec{t}$ is obtained by replacing each occurrence of x_i in $\vec{t}$ by the i-th term of the tuple $\vec{s}$, for $i \in [1, k]$. For example, constants a, b in Σ are arrows from $\underline{0}$ to $\underline{1}$, a binary operator $f(x_1, x_2)$ defines an arrow from $\underline{2}$ to $\underline{1}$, and the composition $\langle a, b\rangle; \langle f(x_1, x_2), x_1\rangle; f(x_2, x_1)$, where the angle brackets denote term tupling, yields the term $f(a, f(a, b))$, which is an arrow from $\underline{0}$ to $\underline{1}$. In fact,

$$\langle a, b\rangle; \langle f(x_1, x_2), x_1\rangle; f(x_2, x_1) = \langle f(a, b), a\rangle; f(x_2, x_1) = f(a, f(a, b))$$

Monoidal theories stay to algebraic theories as linear substitutions stay to generic substitutions. More precisely, in monoidal theories variables can be neither duplicated (as e.g. in $f(x_1, x_1)$) nor projected. Even though terms are formally annotated with the variables on which they are built, when no confusion can arise, we avoid such annotations, and also the use of angle brackets.

LTS defined over closed terms of a given signature Σ and label alphabet Λ can be conveniently specified as collections of inductive proof rules, called *transition system specifications*. A *transition rule* α has the form

$$\alpha : \frac{s_1 \xrightarrow{a_1} t_1 \; ... \; s_n \xrightarrow{a_n} t_n}{s \xrightarrow{a} t}$$

where the s_i, t_i, s and t range over $\mathbb{T}_\Sigma(X)$ and the a_i, a range over Λ. Transitions in the upper part of the rule are called *premises*, the one in the lower part *conclusion*. The rule α is closed if it does not contain variables. A *transition system specification* (TSS) is a set of transition rules.

A proof of a closed *transition rule* $H/s \xrightarrow{a} t$ with H a set of (closed) premises, is a well-founded, upwardly branching tree whose nodes are labeled by closed transitions, the root is labeled by $s \xrightarrow{a} t$ and if H_r is the set of labels for the nodes above a node r with label $s_r \xrightarrow{a_r} t_r$ then either $H_r/s_r \xrightarrow{a_r} t_r$ is a closed rule that can be obtained as an instance of a rule in the TSS, or $s_r \xrightarrow{a_r} t_r \in H$ and $H_r = \varnothing$. A proof of a closed *transition* $s \xrightarrow{a} t$ is a proof of $\varnothing/s \xrightarrow{a} t$. The LTS *associated* to a TSS consists of the set of provable closed transitions.

Among the formats that guarantee the fundamental property of 'bisimilarity as congruence' on the associated LTS we shall in the following recall the *De Simone*, the GSOS and the `tyft` formats. We remind that an equivalence relation $\mathcal{R}$ over $\mathbb{T}_\Sigma$ is a *congruence* if it respects the algebraic structure of states given by Σ, i.e. if for all $f \in \Sigma$

$$s_i \mathcal{R} t_i, \text{ for } i \in [1, ar(f)], \quad \text{implies} \quad f(s_1, ..., s_{ar(f)}) \mathcal{R} f(t_1, ..., t_{ar(f)}).$$

Definition 3. *Let Σ be a signature and Λ an alphabet of observations. A transition rule is in* De Simone format *if it has the form*

$$\frac{\{x_i \xrightarrow{a_i} y_i \mid i \in I\}}{f(x_1, ..., x_n) \xrightarrow{a} t}$$

where the a_i and a are labels in Λ, f is a n-ary operator, $I \subseteq \{1, ..n\}$ and the variables x_i and y_i are all distinct and form the set V. Moreover, the target $t \in \mathbb{T}_\Sigma(V)$ does not contain x_i for $i \in I$ and is linear. A TSS *is in De Simone format if all its rule are in such form.*

Observe that the source of the conclusion of a rule in De Simone format consists of a *single function symbol.* This fact is crucial in the proof that bisimilarity is a congruence for TSS specified in De Simone format.

The *positive* GSOS *format* [3] extends De Simone rules in several ways: (1) multiple testings of the same argument (the x_i) are allowed in the premises; (2) tested arguments can appear in the target t of the conclusion; (3) the target t of the conclusion can be a non-linear context (variables can be used more than once). The `tyft` *format* [13] further generalizes the GSOS, by allowing generic terms t_i as sources of the transitions in the premises. Notice however that the main restriction about the source of the conclusion still persists: allowing conclusions of the form $C[x_1, ..., x_n] \xrightarrow{a} t$ could compromise the 'bisimilarity as congruence' property as the following example illustrates.

Example 1. Consider a process algebra over the signature $\Sigma = \{nil, \alpha.\_, \bar{\alpha}.\_, \_ \mid \_\}$ with α ranging over a suitable set of channels A, where *nil* is the empty agent, $\alpha.\_$ and $\bar{\alpha}.\_$ are two complementary unary operator for action prefix (on the channel α) and $\_ \mid \_$ is a binary parallel composition operator. If we consider the TTS consisting of the axiom $\lambda.x_1 \mid \bar{\lambda}.x_2 \xrightarrow{\tau} x_1 \mid x_2$ plus the usual rules that propagate the τ through the $\_ \mid \_$ operator (asynchronously), then it is obvious that $\alpha.nil \simeq \bar{\alpha}.nil$. However, if put in the context $x_1 \mid \bar{\alpha}.nil$, then $\alpha.nil \mid \bar{\alpha}.nil \xrightarrow{\tau} nil \mid nil$, while $\bar{\alpha}.nil \mid \bar{\alpha}.nil$ cannot move. Hence $\alpha.nil \mid \bar{\alpha}.nil \not\simeq \bar{\alpha}.nil \mid \bar{\alpha}.nil$.

Dynamic bisimulation has been defined to reproduce the effect of run-time reconfiguration in open ended systems. It extends the ordinary bisimulation-game by allowing moves that put the states under comparison in the same context.

Definition 4. *Given a* LTS $L = (\mathbb{T}_\Sigma, \Lambda, \rightarrow)$, *a* dynamic bisimulation *on L is a symmetric, reflexive relation $\sim_{\mathrm{d}} \subseteq \mathbb{T}_\Sigma \times \mathbb{T}_\Sigma$ such that if $s \sim_{\mathrm{d}} t$ then for any unary context $C[\_]$ (including the identity) and transition $C[s] \xrightarrow{a} s'$ there exists a transition $C[t] \xrightarrow{a} t'$ with $s' \sim_{\mathrm{d}} t'$.*

Two states s and t are *dynamic bisimilar*, written $s \simeq_{\mathrm{d}} t$, if there exists a dynamic bisimulation $\sim_{\mathrm{d}}$ such that $s \sim_{\mathrm{d}} t$. Thus, w.r.t. Example 1, we have, e.g., $\alpha.nil \not\simeq_{\mathrm{d}} \bar{\alpha}.nil$. Dynamic bisimilarity is the *coarsest congruence* which is also a *bisimulation*. Note that context moves cannot be 'observed': they are part of the game but not of the LTS. Even if not remarked in [20], dynamic bisimulation can however be recasted in ordinary bisimulation over an extended system. Observe that the dynamic extension is an *infinitary* construction on the LTS and is not expressed at the level of the TSS, i.e., it remains at the '*meta-level*'.

Definition 5. *Given a* LTS $L = (\mathbb{T}_\Sigma, \Lambda, \rightarrow)$, *its* dynamic extension $\widehat{L}$ *is the* LTS $(\mathbb{T}_\Sigma, \Lambda, \rightarrow \cup \rightrightarrows)$, *where* $s \overset{C[\_]}{\rightrightarrows} C[s]$ *for all $s \in \mathbb{T}_\Sigma$ and unary contexts $C[\_]$.*

Proposition 1. *$s \simeq_{\mathrm{d}} t$ in L iff $s \simeq t$ in $\widehat{L}$.*

The proof of Proposition 1 relies on the fact that if s makes a move $C[\_]$, then t can always simulate such move in a unique way.

2 Tile Formats

A *tiles system* is a tuple $\mathcal{R} = (\mathcal{H}, \mathcal{V}, N, R)$ where $\mathcal{H}$ and $\mathcal{V}$ are monoidal categories with the same set of objects $O_\mathcal{H} = O_\mathcal{V}$, N is the set of rule names and $R\colon N \rightarrow \mathcal{H} \times \mathcal{V} \times \mathcal{V} \times \mathcal{H}$ is a function such that for all $\alpha \in N$, if $R(\alpha) = \langle s, a, b, t\rangle$, then we have $s\colon x \rightarrow y$, $a\colon x \rightarrow z$, $b\colon y \rightarrow w$, and $t\colon z \rightarrow w$ for suitable objects x, y, z and w. We shall write such rule either as the sequent $\alpha\colon s \xrightarrow[b]{a} t$, or as the tile

$$\begin{tikzcd} x \arrow[r, "s"] \arrow[d, "a"'] \arrow[dr, phantom, "\alpha"] & y \arrow[d, "b"] \\ z \arrow[r, "t"'] & w \end{tikzcd}$$

thus making explicit the source and target of each arrow. The category $\mathcal{H}$ is called *horizontal* and its arrows *configurations*. The category $\mathcal{V}$ is called *vertical* and its arrows *observations*. The objects of $\mathcal{H}$ and $\mathcal{V}$ are called *interfaces*.

Starting from the basic tiles $R(\alpha)$ of the system, more complex tiles can be constructed via horizontal, vertical and parallel composition. Moreover, the horizontal and vertical identities are always added to the system and composed with the basic tiles. All this is illustrated in Figure 2. Depending on the chosen tile format, $\mathcal{H}$ and $\mathcal{V}$ must satisfy certain constraints and suitable *auxiliary tiles* are added and composed with basic tiles and identities in all the possible ways. The set of resulting tiles (called *flat sequents*) define the *flat tile logic* associated to $\mathcal{R}$. We say that $s \xrightarrow[b]{a} t$ is *entailed* by the logic, written $\mathcal{R} \vdash s \xrightarrow[b]{a} t$, if the sequent $s \xrightarrow[b]{a} t$ can be expressed as the composition of basic and auxiliary tiles.

$$\frac{s \xrightarrow[b]{a} t \quad h \xrightarrow[c]{b} f}{s; h \xrightarrow[c]{a} t; f} \qquad \frac{s \xrightarrow[b]{a} t \quad t \xrightarrow[d]{c} h}{s \xrightarrow[b;d]{a;c} h} \qquad \frac{s \xrightarrow[b]{a} t \quad h \xrightarrow[d]{c} f}{s \otimes h \xrightarrow[b\otimes d]{a\otimes c} t \otimes f}$$

$$\frac{t\colon x \to y \in \mathcal{H}}{t \xrightarrow[y]{x} t} \qquad \frac{a\colon x \to z \in \mathcal{V}}{x \xrightarrow[a]{a} z}$$

Figure 2.

Definition 6. *Let $\mathcal{R} = (\mathcal{H}, \mathcal{V}, N, R)$ be a tile system. A symmetric relation $\sim_{\mathrm{t}}$ on configurations is called* tile bisimulation *if whenever $s \sim_{\mathrm{t}} t$ and $\mathcal{R} \vdash s \xrightarrow[b]{a} s'$, then there exists t' such that $\mathcal{R} \vdash t \xrightarrow[b]{a} t'$ and $s' \sim_{\mathrm{t}} t'$.*

The maximal tile bisimulation is denoted by $\simeq_{\mathrm{t}}$, and two configurations s and t are said to be *tile bisimilar* if $s \simeq_{\mathrm{t}} t$.

We are particularly interested in considering tiles systems where the monoidal categories of configurations and observations are freely generated from suitable horizontal and vertical signatures, respectively, i.e., they are categories of *substitutions*, as discussed in the previous section.

The tile format proposed in the original presentation of tiles [12] is the so-called *algebraic tile format* that recollected the perspective of ordinary TSS: configurations are terms over a certain signature, and observations are the arrows of the monoidal category freely generated by certain labels (regarded as unary operators). Auxiliary tiles lift the horizontal cartesian structure to the horizontal composition of tiles. In the algebraic tile format basic tiles have the form:

$$\begin{tikzcd} \underline{n} \arrow[r, "s"] \arrow[d, "a_1\otimes\cdots\otimes a_n"'] & \underline{1} \arrow[d, "a"] \\ \underline{n} \arrow[r, "t"'] & \underline{1} \end{tikzcd}$$

where the a_i and a can be either labels (viewed as arrows from $\underline{1}$ to $\underline{1}$) or identities and $s, t \in \mathbb{T}_{\Sigma}(\{x_1, ..., x_n\})$. The idea is that each interface represents an ordered sequence of variables; therefore each variable is completely identified by its position in the tuple, and a standard naming $x_1, ..., x_n$ of the variables can be assumed for all interfaces. A typical auxiliary tile for the algebraic format is

$$\begin{tikzcd} \underline{1} \arrow[r, "\langle x_1, x_1\rangle"] \arrow[d, "a"'] & \underline{2} \arrow[d, "a\otimes a"] \\ \underline{1} \arrow[r, "\langle x_1, x_1\rangle"'] & \underline{2} \end{tikzcd}$$

that duplicates the observation a (trigger of the tile) propagating it to two instances of the unique variable in the initial interface. We refer to [12] for more details. The algebraic tile format corresponds to SOS rules of the form

$$\frac{\{x_i \xrightarrow{a_i} y_i \mid i \in I\}}{C[x_1, ..., x_n] \xrightarrow{a} D[y_1, ..., y_n]}$$

where $I \subseteq \{1, ..., n\}$, $C[x_1, ..., x_n]$ and $D[y_1, ..., y_n]$ are contexts (corresponding to s and t in the tile), and all the y_i and x_i are different if $i \in I$, but $y_i = x_i$ otherwise. The correspondence follows since for all closed terms s and t and for any label a, $\mathcal{R} \vdash s \xrightarrow[a]{id_0} t$ if and only if the LTS associated to the SOS specification includes the transition $t \xrightarrow{a} s$.

The algebraic tile format is not uniform in the two dimensions, since $\mathcal{H}$ is cartesian, whereas $\mathcal{V}$ is only monoidal (non symmetric). Since our idea is to observe contexts by replicating (part of) the horizontal structure in the vertical dimension, we prefer (1) to renounce to the cartesian structure altogether, resorting to the simpler *monoidal tile format* [17] where only *linear* contexts are allowed, or (2) consider the more general *term tile format* [4], where also $\mathcal{V}$ is cartesian. Notice that monoidal theories suffice for expressing all closed terms, even though, as explained in [5], the term tile format is more expressive.

Since in all tile formats the categories of configurations and observations are *freely generated* by the horizontal signature Σ and by (the signature associated to) the set of labels Λ, monoidal/algebraic/term tile systems are usually represented as tuples of the form $\mathcal{R} = (\Sigma, \Lambda, N, R)$.

According to the term tile format each basic tile has the form:

$$\begin{tikzcd} \underline{n} \arrow[r, "\vec{h}"] \arrow[d, "\vec{v}"'] & \underline{m} \arrow[d, "u"] \\ \underline{k} \arrow[r, "g"'] & \underline{1} \end{tikzcd}$$

with $\vec{h} \in \mathbb{T}_{\Sigma^{\mathrm{H}}}(X_n)^m$, $g \in \mathbb{T}_{\Sigma^{\mathrm{H}}}(X_k)$, $\vec{v} \in \mathbb{T}_{\Sigma^{\mathrm{V}}}(X_n)^k$, and $u \in \mathbb{T}_{\Sigma^{\mathrm{V}}}(X_m)$, where $X_i = \{x_1, ..., x_i\}$ is a chosen set of variables, Σ^{H} is the horizontal signature of configurations and Σ^{V} is the vertical signature of observations. Of course, if Λ contains only elementary actions, regarded as unary operators, then $m = 1$. We present tiles more concisely as logic sequents $n \triangleleft\ \vec{h} \xrightarrow[u]{\vec{v}} g$, where the number of variables in the 'upper-left' corner of the tile is made explicit (the values m and k can be recovered from the lengths of $\vec{h}$ and $\vec{v}$). Again, a standard naming for the variables in the interfaces is assumed. For example, if the variable x_i appears in the effect u of the above rule, then the effect u depends on the ith component h_i of the initial configuration. Analogously for the remaining connections. As already remarked, the same variable x_i denotes the ith element of different interfaces when used in each of the four border-arrows of the tile (as a matter of fact, only the occurrences of x_i in $\vec{h}$ and in $\vec{v}$ denote the same element of the initial input interface $\underline{n}$).

Auxiliary tiles for term tile systems consist of all term tiles $n \triangleleft\ h \xrightarrow[u]{v} g$ such that h, g, u and v are terms over the empty signature – and therefore also terms of $\mathbb{T}_{\Sigma^{\mathrm{V}}}(X)$ and $\mathbb{T}_{\Sigma^{\mathrm{H}}}(X)$ – such that $h; u = v; g$, i.e., all tiles that perform consistent rearrangements of a generic interface in the two dimensions. A typical auxiliary term tile is $1 \triangleleft\ x_1 \xrightarrow[x_1, x_1]{x_1} x_1, x_1$ that consistently duplicates the unary interface. Observe that term tile format *extends* the positive GSOS format.

An interesting question concerns suitable restrictions of the monoidal and term tile formats such that tile bisimilarity yields a congruence. Two main properties have been investigated in the literature for obtaining tile bisimilarity congruences: the *basic source* and the *tile decomposition*. Tile decomposition has a completely abstract formulation that applies to all tile systems.

Definition 7. *A tile system $\mathcal{R} = (\mathcal{H}, \mathcal{V}, N, R)$ enjoys the* decomposition property *if for all arrows $s: x \to y \in \mathcal{H}$ and for all sequents $s \xrightarrow[b]{a} t$ entailed by $\mathcal{R}$*

- *if $s = s_1; s_2$ then there exists $c \in \mathcal{V}$ and $t_1, t_2 \in \mathcal{H}$ such that $\mathcal{R} \vdash s_1 \xrightarrow[c]{a} t_1$, $\mathcal{R} \vdash s_2 \xrightarrow[b]{c} t_2$ and $t = t_1; t_2$;*
- *if $s = s_1 \otimes s_2$ then there exists $a_1, a_2, b_1, b_2 \in \mathcal{V}$ and $t_1, t_2 \in \mathcal{H}$ such that $\mathcal{R} \vdash s_1 \xrightarrow[b_1]{a_1} t_1$, $\mathcal{R} \vdash s_2 \xrightarrow[b_2]{a_2} t_2$, $a = a_1 \otimes a_2$, $b = b_1 \otimes b_2$ and $t = t_1 \otimes t_2$;*

Proposition 2. *If $\mathcal{R}$ enjoys the decomposition property, then tile bisimilarity is a congruence (w.r.t. the operations of the horizontal category, i.e., sequential and parallel composition).*

Even though we did not address it explicitly, all the definitions we have given for tiles apply also to the case of term algebras modulo structural axioms (e.g., associativity and commutativity of parallel composition in CCS) and all our results can be immediately extended.

3 Basic Source

The results in this section mildly extend those of [12]. Since in this paper we consider equivalences on closed terms, we refine the notion of tile bisimulation to *ground tile bisimulation*.

Definition 8. *Let $\mathcal{R} = (\Sigma^{\mathrm{H}}, \Sigma^{\mathrm{V}}, N, R)$ be a monoidal (resp. term) tile system. A symmetric relation $\sim_{\mathrm{g}}$ on closed configurations (i.e., elements of $\mathbb{T}_{\Sigma^{\mathrm{H}}}$) is called* ground tile bisimulation *if whenever $s \sim_{\mathrm{g}} t$ and $\mathcal{R} \vdash s \xrightarrow[a]{id_0} s'$, then there exists t' such that $\mathcal{R} \vdash t \xrightarrow[a]{id_0} t'$ and $s' \sim_{\mathrm{g}} t'$.*

Ground tile bisimulation is the exact counterpart of ordinary bisimulation for LTS. It differs from tile bisimulation in that is not defined on contexts. (Since ground terms need no trigger, ground bisimulation tests only the effects they can produce.) The maximal ground tile bisimulation is denoted by $\simeq_{\mathrm{g}}$, and two closed configurations s and t are said to be *ground tile bisimilar* if $s \simeq_{\mathrm{g}} t$. In fact we are interested in the LTS associated to tile systems.

Definition 9. *For $\mathcal{R} = (\Sigma^{\mathrm{H}}, \Sigma^{\mathrm{V}}, N, R)$ a monoidal/term tile system, the* LTS *associated to $\mathcal{R}$ is $L_{\mathcal{R}} = (\mathbb{T}_{\Sigma^{\mathrm{H}}}, \mathbb{T}_{\Sigma^{\mathrm{V}}}(\{x_1\}), \to)$ where $s \xrightarrow{a} t$ iff $\mathcal{R} \vdash s \xrightarrow[a]{id_0} t$.*

The decomposition property can be refined and related to the 'tile bisimilarity as congruence' property.

Definition 10. *A term (resp. monoidal) tile system $\mathcal{R}$ enjoys the* ground decomposition property *if for any $s \in \mathbb{T}_{\Sigma^{\mathrm{H}}}$ and any sequent $\mathcal{R} \vdash C[s_1] \xrightarrow[b]{id_0} t$ with C a unary (resp. linear unary) context and s_1 a ground term such that $s = C[s_1]$, then there exists an observation c, a ground term t_1 and a (resp. linear) context D such that $\mathcal{R} \vdash s_1 \xrightarrow[c]{id_0} t_1$ and $\mathcal{R} \vdash C[x_1] \xrightarrow[b]{c} D[x_1]$, with $t = D[t_1]$.*

Theorem 1. *Let $\mathcal{R} = (\Sigma^{\mathrm{H}}, \Sigma^{\mathrm{V}}, N, R)$ be a term tile system. The ground decomposition property implies that ground tile bisimilarity on $\mathcal{R}$ is a congruence.*

Proof. Standard. Define the congruence $\hat{\simeq}_{\mathrm{g}}$ as the minimal relation such that if $s \simeq_{\mathrm{g}} t$ and $C[x_1]$ is a unary context then $C[s] \hat{\simeq}_{\mathrm{g}} C[t]$. Obviously $\simeq_{\mathrm{g}} \subseteq \hat{\simeq}_{\mathrm{g}}$. We then show that $\hat{\simeq}_{\mathrm{g}}$ is a ground tile bisimulation and, therefore, coincides with ground tile bisimilarity. In fact, let $C[s] \simeq_{\mathrm{g}} C[t]$ for $s, t \in \mathbb{T}_{\Sigma^{\mathrm{H}}}$ with $s \simeq_{\mathrm{g}} t$ and $C[x_1]$ a unary context. By ground decomposition we have that if $\mathcal{R} \vdash C[s] \xrightarrow[a]{id_0} s'$, there exist $s_1 \in \mathbb{T}_{\Sigma^{\mathrm{H}}}$, an observation b and a unary context $D[x_1]$ such that $\mathcal{R} \vdash s \xrightarrow[b]{id_0} s_1$, $\mathcal{R} \vdash C[x_1] \xrightarrow[a]{b} D[x_1]$ and $s' = D[s_1]$. Since $s \simeq_{\mathrm{g}} t$ then $\mathcal{R} \vdash t \xrightarrow[b]{id_0} t_1$ for some $t_1 \in \mathbb{T}_{\Sigma^{\mathrm{H}}}$ with $s_1 \simeq_{\mathrm{g}} t_1$. By horizontal composition of tiles we then have $\mathcal{R} \vdash C[t] \xrightarrow[a]{id_0} D[t_1]$. By definition of $\hat{\simeq}_{\mathrm{g}}$ we have that $D[s_1] \hat{\simeq}_{\mathrm{g}} D[t_1]$. $\square$

Theorem 2. *Given a monoidal tile system $\mathcal{R} = (\Sigma, \Lambda, N, R)$, the ground decomposition property implies that ground tile bisimilarity is a congruence.*

Proof. Similar to the proof of Theorem 1, but requires $\hat{\simeq}_{\mathrm{g}}$ to be the minimal relation such that if $s \simeq_{\mathrm{g}} t$ and $C[x_1]$ is a *linear* (rather than generic) unary context then $C[s] \hat{\simeq}_{\mathrm{g}} C[t]$. Observe that the congruence property then holds for generic contexts. In fact, if $D[\_]$ is not linear, let $D'[x_1, x_2, ..., x_n]$ be the linear context obtained from $D[\_]$ by replacing each occurrence of the hole '_' by a different variable x_i. Then given any two closed terms s and t we have

$$D[s] = D'[\underbrace{s, s, ..., s}_{n}] \hat{\simeq}_{\mathrm{g}} D'[t, \underbrace{s, ..., s}_{n-1}] \hat{\simeq}_{\mathrm{g}} D'[t, t, \underbrace{s, ..., s}_{n-2}] \hat{\simeq}_{\mathrm{g}} \cdots \hat{\simeq}_{\mathrm{g}} D'[\underbrace{t, t, ..., t}_{n}] = D[t]$$

since all contexts $D'[\_, s, ..., s]$, $D'[t, \_, s, ..., s]$, $\ldots$, $D'[t, ..., t, \_]$ are linear. $\square$

The ground decomposition property can be enforced by syntactical constraints on basic tiles. A tile system verifies the *basic source property* if the initial configuration of each basic tile consists of a *single operator*, rather than a generic context.

Proposition 3. *If a monoidal (resp. term) tile system $\mathcal{R}$ enjoys the basic source property, then the ground tile bisimilarity on $\mathcal{R}$ is a congruence.*

The proof of Proposition 3 consists of two steps: we first prove that basic source implies *ground tile decomposition* and then we conclude by exploiting Theorems 1 and 2. Although Theorems 1 and 2 hold also when structural axioms are imposed on configurations, this is not necessarily the case for Proposition 3, as the first part of the argument above may fail. We remark that for the monoidal tile format the proof coincides with that for the De Simone format, since the two formats are essentially the same.

4 Dynamic Tile Bisimulation

When the basic source property is not satisfied we are likely to have bisimilarities that are not congruences. This consideration applies to all the most diffused formats, unless the specification contains enough rules to distinguish context dependent redexes. For example, Corradini and Heckel in joint work with the second author [8] suggested one may deal with this situation via a closure operation on the TSS rules. And Sewell in [24] when passing from reduction systems to transition systems performs such a closure by adding a transition labeled with C for each state s and context C which s can react with in order to perform a reduction. However, such closures are expressed at a meta-level, as they are not handled by adding rules to the original specification. Therefore, from a different perspective, dynamic bisimilarity is more satisfactory, since it allows for a concise definition of the congruence one is looking for.

Tiles have the expressive power to reconcile finitary system specifications and dynamic bisimulation within the same perspective, i.e., by adding suitable auxiliary tiles. Moreover, the closure w.r.t. all contexts can be expressed simply by adding just as many basic tiles as the operators in the signature. Hence, if the signature is finite, so are the additional auxiliary tiles needed.

Definition 11. *Given a term (resp. monoidal) tile system $\mathcal{R} = (\Sigma^{\mathrm{H}}, \Sigma^{\mathrm{V}}, N, R)$ its* dynamic extension *$\widehat{\mathcal{R}}$ is obtained by adding for all n and for any operator $f \in \Sigma^{\mathrm{H}}_n$ the auxiliary operator $\tilde{f}$ to Σ^{V}_n and the following auxiliary tiles.*

$$
\begin{array}{ccc}
\underline{n} & \xrightarrow{x_1,.,x_n} & \underline{n} \\
{\scriptstyle x_1,.,x_n}\big\downarrow & & \big\downarrow{\scriptstyle \tilde{f}(x_1,.,x_n)} \\
\underline{n} & \xrightarrow[f(x_1,.,x_n)]{} & \underline{1}
\end{array}
\qquad\qquad
\begin{array}{ccc}
\underline{n} & \xrightarrow{f(x_1,.,x_n)} & \underline{1} \\
{\scriptstyle \tilde{f}(x_1,.,x_n)}\big\downarrow & & \big\downarrow{\scriptstyle x_1} \\
\underline{1} & \xrightarrow[x_1]{} & \underline{1}
\end{array}
$$

For t a generic horizontal context, we let $\tilde{t}$ denote the corresponding vertical context on the extended signature, which is obtained by replacing each operator f that appears in t by its vertical counterpart $\tilde{f}$, leaving variables unchanged. For the proof of the main theorem we need the following technical lemmas that require some acquaintance with the term tile format. A corresponding lemma can be proved for the monoidal tile format, by considering linear contexts only.

Lemma 1. *Given a term tile system $\mathcal{R} = (\Sigma^{\mathrm{H}}, \Sigma^{\mathrm{V}}, N, R)$, for each context $t{:}\,\underline{n} \to \underline{1}$ we have $\widehat{\mathcal{R}} \vdash n \triangleleft\ x_1, ..., x_n \xrightarrow[\tilde{t}]{x_1,...,x_n} t$.*

Proof. The proof proceeds by induction on the (maximum) depth m of the tree-like representation of t. The base case $m = 0$ is trivial. If $m > 1$ then $t = f(t_1, ..., t_k)$, where f is a k-ary operator of Σ^{H} and $t_1, ..., t_k$ are context with (maximum) depth strictly less than m. Then we can apply the inductive hypothesis to conclude that $n \triangleleft\ x_1, ..., x_n \xrightarrow[\tilde{t_i}]{x_1,...,x_n} t_i$ for $i \in [1, k]$. Composing in parallel such sequents we get $k \cdot n \triangleleft\ x_1, ..., x_{k\cdot n} \xrightarrow[\tilde{s_1},...,\tilde{s_k}]{x_1,...,x_{k\cdot n}} s_1, ..., s_k,$ where

$$\begin{tikzcd}
\cdot \arrow[r, "s"] \arrow[d, "id"'] & \cdot \arrow[r, "id"] \arrow[d, dotted, "id"] & \cdot \arrow[r, "C"] \arrow[d, dotted, "C"] & \cdot \arrow[d, "id"] \\
\cdot \arrow[r, dotted, "s"] \arrow[d, "id"'] & \cdot \arrow[r, dotted, "C"] \arrow[d, phantom, "\alpha"] & \cdot \arrow[r, dotted, "id"] \arrow[d, dotted, "a"] & \cdot \arrow[d, "a"] \\
\cdot \arrow[rr, "t"'] & & \cdot \arrow[r, "id"'] & \cdot
\end{tikzcd}$$

Figure 3.

$s_i = t_i[x_{n\cdot(i-1)+1}/x_1, \ldots, x_{n\cdot(i-1)+n}/x_n]$, i.e., the s_i are the t_i with the variables suitably renamed according to the initial input interface. Then we can horizontally compose with the auxiliary tile of term tile systems that makes k copies of n inputs $n \triangleleft\ x_1, ., x_n, ..., x_1, ., x_n \xrightarrow[x_1,...,x_{k\cdot n}]{x_1,.,x_n,...,x_1,.,x_n} x_1, ., x_{k\cdot n}$, obtaining the sequent $\alpha = n \triangleleft\ x_1, ..., x_n \xrightarrow[\tilde{t_1},...,\tilde{t_k}]{x_1,...,x_n} t_1, ..., t_k$. Finally we can compose it with the auxiliary sequent of the extended system $\beta = k \triangleleft\ x_1, ..., x_k \xrightarrow[\tilde{f}(x_1,...,x_k)]{x_1,...,x_k} f(x_1, ..., x_k)$ as in

$$\begin{tikzcd}
\cdot \arrow[r] \arrow[d] & \cdot \arrow[r] \arrow[d] & \cdot \arrow[d] \\
\cdot \arrow[r] \arrow[d] \arrow[u, phantom, ""] & \cdot \arrow[r] \arrow[d] & \cdot \arrow[d] \\
\cdot \arrow[r] & \cdot \arrow[r] & \cdot
\arrow[from=1-1, to=2-2, phantom, "\alpha"]
\arrow[from=1-2, to=2-3, phantom, "\gamma"]
\arrow[from=2-1, to=3-2, phantom, "\delta"]
\arrow[from=2-2, to=3-3, phantom, "\beta"]
\end{tikzcd}$$

where γ is the horizontal identity for the effect of α and δ is the vertical identity for the final configuration of α. The composition yields the sequent $n \triangleleft\ x_1, ..., x_n \xrightarrow[\tilde{f}(\tilde{t_1},...,\tilde{t_k})]{x_1,...,x_n} f(t_1, ..., t_k) \ = n \triangleleft\ x_1, ..., x_n \xrightarrow[\tilde{t}]{x_1,...,x_n} t$ and concludes the proof. $\square$

A similar argument shows the following lemma.

Lemma 2. *Given a term tile system $\mathcal{R} = (\Sigma^{\mathrm{H}}, \Sigma^{\mathrm{V}}, N, R)$, for each context $t: \underline{n} \to \underline{1}$ we have $\widehat{\mathcal{R}} \vdash n \triangleleft\ t \xrightarrow[x_1]{\tilde{t}} x_1$.*

Theorem 3. *Let $\mathcal{R} = (\Sigma^{\mathrm{H}}, \Sigma^{\mathrm{V}}, N, R)$ be a term tile system. The ground tile bisimilarity defined on $\widehat{\mathcal{R}}$ defines a congruence for $\mathcal{R}$.*

Proof. First notice that the auxiliary tiles do not influence the definition of ground tile bisimilarity, which deals only with null triggers. Then, we prove that $\widehat{\mathcal{R}}$ enjoys the ground decomposition property from which we get the expected 'bisimilarity as a congruence' property. In fact, given a generic sequent $\alpha: C[s] \xrightarrow[a]{id_0} t$ entailed by $\widehat{\mathcal{R}}$, we can always construct two tiles with source s and $C[x_1]$ respectively that decompose α, as illustrated in Figure 3. $\square$

Theorem 4. *Ground tile bisimilarity on $\widehat{\mathcal{R}}$, denoted by $\simeq_{\hat{\mathrm{g}}}$, coincides with the dynamic bisimilarity on $L_{\mathcal{R}}$.*

Proof. We must show that $L_{\widehat{\mathcal{R}}} = \widehat{L}_{\mathcal{R}}$. The inclusion $L_{\widehat{\mathcal{R}}} \supseteq \widehat{L}_{\mathcal{R}}$ follows directly from the technical lemmas above, whilst the inclusion $L_{\widehat{\mathcal{R}}} \subseteq \widehat{L}_{\mathcal{R}}$ is more involved. The key point is showing that if an auxiliary 'context' tile gives rise to a new transitions, its label $\tilde{f}$ appears manifestly, i.e., the use of auxiliary tiles cannot be 'hidden' inside the proof to originate unexpected reactions. To show this, let $\mathcal{R} \vdash s \xrightarrow[a]{id_0} t$ and suppose that the proof of $s \xrightarrow[a]{id_0} t$ contains the auxiliary tile $\alpha = \vec{x} \xrightarrow[\tilde{f}(\vec{x})]{\vec{x}} f(\vec{x})$, for some operator f. We proceed by induction on the number k of such auxiliary tiles and then by case analysis. If $k = 0$ then $s \xrightarrow{a} t \in L_{\mathcal{R}} \subseteq \widehat{L}_{\mathcal{R}}$. If $k > 1$ then we take one such auxiliary tile α in the proof and examine the following three cases: (1) the effect of α is propagated to the final effect $a = A[\tilde{f}(\vec{a})]$ and thus can be observed; (2) the tile α is horizontally composed with $\beta = f(\vec{x}) \xrightarrow[x_1]{\tilde{f}(\vec{x})} x_1$ and thus does not appear in a; (3) the effect $\tilde{f}$ is vertically composed with other effects that override it. In case (1), α corresponds to a context move in $\widehat{L}_{\mathcal{R}}$. In case (2), the composition of α with β yields the vertical identity on f which is of course entailed in $\mathcal{R}$. Finally, in case (3), the effect $\tilde{f}$ can only be overridden because of structural axioms on observations, and in particular those involving projections. But then it can be shown that if projections are used that throw $\tilde{f}$ away, then also the result of applying the context f in the intermediate state is thrown away in the proof. Therefore, we can always reduce to a proof with $k - 1$ auxiliary tiles for adding contexts and conclude the proof by inductive hypothesis. $\square$

For monoidal tile systems the proof simplifies considerably, since case (3) cannot occur. We remark that if observations are subject to structural axioms (e.g., $b; a = a$ for all observations b), then such axioms *must not* be extended to the $\tilde{f}$, otherwise the case (3) of the proof could be compromised.

Example 2. Let us take again the process algebra of Example 1, with $\_ \mid \_$ associative (but neither commutative nor with unit). Then $\alpha.nil \mid \bar{\alpha}.nil \simeq_{\mathrm{g}} \beta.nil \mid \bar{\beta}.nil \not\simeq_{\hat{\mathrm{g}}} \alpha.nil \mid \bar{\alpha}.nil$. While, e.g., $t_\alpha \simeq_{\mathrm{g}} t_\beta \simeq_{\hat{\mathrm{g}}} t_\alpha$ for $t_\lambda = nil \mid \lambda.nil \mid \bar{\lambda}.nil \mid nil$.

Concluding Remarks

We have proposed tile logic as a compositional framework suitable to deal with open ended systems, dynamic bisimulation and structural axioms on states. Such characteristics follows naturally from the abstract 'geometrical' concepts which tile configurations and observations are based on. In particular, the winning feature, is the possibility of exploiting the analogy between horizontal and vertical arrows, making observations out of contexts. Moreover, dynamic bisimulation is handled via a finitary enrichment of the specification and the congruence proof has a simple pictorial representation that exploits ground decomposition.

Following the lines suggested in this paper, one could limit run-time reconfiguration either to a sub-class of contexts or to a sub-class of configurations. Although we have not discussed the issue here, tile logic can deal with trace semantics as well.

References

1. K. Bernstein. A congruence theorem for structured operational semantics of higher-order languages. In *Proc. 13th LICS*, IEEE Press, 1998.
2. G. Berry and G. Boudol. The chemical abstract machine. *Theoret. Comput. Sci.*, 96(1):217–248, 1992.
3. B. Bloom, S. Istrail, and A.R. Meyer. Bisimulation can't be traced. *Journal of the ACM*, 42(1):232–268, 1995.
4. R. Bruni, J. Meseguer, and U. Montanari. Process and term tile logic. Technical Report SRI-CSL-98-06. SRI International, 1998.
5. R. Bruni, J. Meseguer, and U. Montanari. Executable tile specifications for process calculi. In *Proc. FASE'99*, vol. 1577 of *LNCS*, pages 60–76, Springer, 1999.
6. R. Bruni, J. Meseguer, and U. Montanari. Symmetric monoidal and cartesian double categories as a semantic framework for tile logic. *Mathematical Structures in Computer Science*, 2000. To appear.
7. R. Bruni and U. Montanari. Zero-safe nets: Comparing the collective and individual token approaches. *Information and Computation*, 156:46–89, 2000.
8. A. Corradini, R. Heckel, and U. Montanari. From SOS specifications to structured coalgebras: how to make bisimulation a congruence. In *Proc. CMCS'99*, vol. 19 of *Elect. Notes in Th. Comput. Sci.*, Elsevier Science, 1999.
9. A. Corradini and U. Montanari. An algebraic semantics for structured transition systems and its application to logic programs. *Th. Comput. Sci.*, 103:51–106, 1992.
10. R. De Simone. Higher level synchronizing devices in MEIJE–SCCS. *Theoret. Comput. Sci.*, 37:245–267, 1985.
11. G.L. Ferrari and U. Montanari. Tile formats for located and mobile systems. *Information and Computation*, 156:173–235, 2000.
12. F. Gadducci and U. Montanari. The tile model. In *Proof, Language and Interaction: Essays in Honour of Robin Milner*, MIT Press, 2000. To appear.
13. J.F. Groote and F. Vaandrager. Structured operational semantics and bisimulation as a congruence. *Information and Computation*, 100:202–260, 1992.
14. K.G. Larsen and L. Xinxin. Compositionality through an operational semantics of contexts. In *Proc. ICALP'90*, vol. 443 of *LNCS*, pages 526–539, Springer, 1990.
15. F.W. Lawvere. Functorial semantics of algebraic theories. *Proc. National Academy of Science*, 50:869–872, 1963.
16. J. Meseguer. Conditional rewriting logic as a unified model of concurrency. *Theoret. Comput. Sci.*, 96:73–155, 1992.
17. J. Meseguer and U. Montanari. Mapping tile logic into rewriting logic. In *Proc. WADT'97*, vol. 1376 of *Lect. Notes in Comput. Sci.*, pages 62–91, Springer, 1998.
18. R. Milner. *A Calculus of Communicating Systems*, vol. 92 of *LNCS* Springer, 1980.
19. R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes (parts I and II). *Information and Computation*, 100:1–77, 1992.
20. U. Montanari and V. Sassone. Dynamic congruence vs. progressing bisimulation for CCS. *Fundamenta Informaticae*, 16:171–196, 1992.
21. U. Montanari and C. Talcott. Can actors and π-agents live together? In *Proc. HOOTS'97*, vol. 10 of *Elect. Notes in Th. Comput. Sci.*, Elsevier Science, 1998.
22. D. Park. Concurrency and automata on infinite sequences. In *Proc. 5th G-I Conference*, vol. 104 of *Lect. Notes in Comput. Sci.*, pages 167–183, Springer, 1981.
23. G. Plotkin. A structural approach to operational semantics. Technical Report DAIMI FN-19, Aarhus University, Computer Science Department, 1981.
24. P. Sewell. From rewrite rules to bisimulation congruences. In *Proc. CONCUR'98*, vol. 1466 of *Lect. Notes in Comput. Sci.*, pages 269–284, Springer, 1998.