IFIP TCS2000 - Sendai - Japan - 17–19 August 2000

Open Ended Systems, Dynamic Bisimulation and Tile Logic

Roberto Bruni Dip. Informatica, Univ. Pisa

joint work with

Ugo Montanari Dip. Informatica, Univ. Pisa

Vladimiro Sassone Dip. Matematica e Informatica, Univ. Catania

Research supported

by CNR Integrated Project *Progettazione e Verifica di Sistemi Eterogenei*;

by Esprit Working Groups *CONFER2* and *COORDINA*;

by TMR Network *GETGRATS*;

and by MURST project *TOSCA*.

Plan of the talk

- Introduction & Motivation
  - Bisimilarity congruences
  - Dynamic bisimilarity
  - Open ended systems
  - Tile logic
- Notation
  - Categories of substitutions
- **Bisimulation and SOS formats**
  - **Bisimilarity as a congruence**
  - **Dynamic bisimilarity**
- **Tile Model**
  - **Tile formats (monoidal, algebraic, term)**
  - **Tile bisimulation**
  - **Tile decomposition & Basic source**
- **Dynamic Tile Bisimulation**
  - **Ground tile bisimulation**
  - **Ground decomposition & Basic source**
  - **Dynamic bisimilarity as ground tile bisimilarity**
- Concluding Remarks

Intro & Motivation

:-) LTS (operational and abstract) semantics and SOS

- → **states as terms over Σ:** $[t_1 \xrightarrow{a} t_2]$
- → **bisimulation and bisimilarity:** $[t_1 \sim t_2]$ **and** $[t_1 \simeq t_2]$
- → **compositionality of abstract sem.:** $[t_1 \simeq t_2 \Rightarrow C[t_1] \simeq C[t_2]]$
- → **SOS spec. formats (e.g., De Simone, GSOS, tyft)**

;-) But *structural axioms* $\mathbb{E}$ are often convenient

- $\equiv_{\mathbb{E}}$ **e.g., CHAM, UML graphs, π-calculus, ambients, ...**

:-(Hence formats may 'fail' to apply

- $\xrightarrow{C[_]}$ **context closure of $\simeq$: not a bisimulation!**
- $\xrightarrow{C[_]}$ **context embedding while playing bisimulation**
- $\xrightarrow{C[_]}$ ***dynamic bisimulation*** **(MS92):** $[t \xrightarrow{C[_]} C[t]]$
- $\xrightarrow{C[_]}$ **application to reconfig. and open-ended systems**

Drawback: **meta-construction flavour and infinitary in principle**

Exploit *Tile Model* **(GM,BMM)**

- □ **abstract, uniform and finitary approach**

Categories of Substitutions

Signature Σ (one sorted) and **Variables** in X

Term algebras $\mathbb{T}_\Sigma(X)$ and $\mathbb{T}_\Sigma = \mathbb{T}_\Sigma(\varnothing)$

Substitutions $\sigma: X \to \mathbb{T}_\Sigma(X)$ (extend to $\sigma: \mathbb{T}_\Sigma(X) \to \mathbb{T}_\Sigma(X)$)

Canonical variables: $\underline{n} = \{x_1, x_2, \ldots, x_n\}$

Finite substitution $\sigma = [t_1/x_1, \ldots, t_n/x_n]$

σ' can be applied elementwise to σ yielding $[\sigma'(t_1)/x_1, \ldots, \sigma'(t_n)/x_n]$

Context $t = C[x_1, ., x_n]$ (using at most $x_1, \ldots, x_n$)

$C[t_1, \ldots, t_n]$ given by applying $[t_1/x_1, \ldots, t_n/x_n]$ to $C[x_1, ., x_n]$

Linear terms, contexts, substitutions

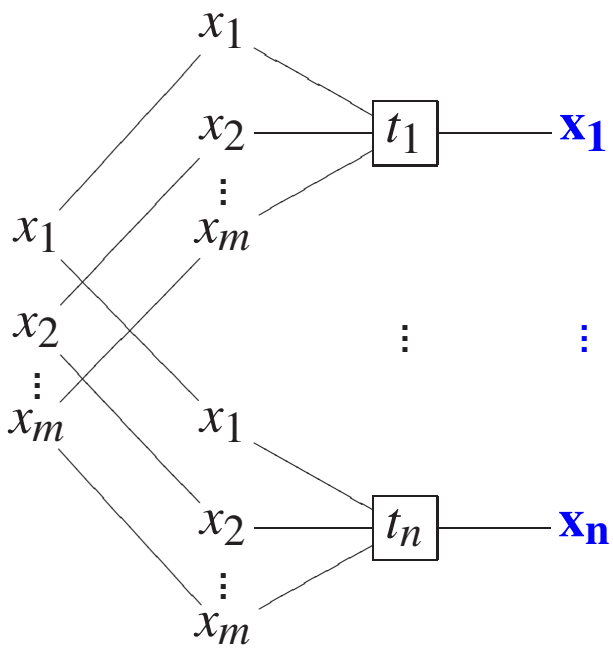
Categories of Substitutions (II)

Context as substitution from n terms to 1: $[C[x_1,\ldots,x_n]/x_1]$.

More generally: For $t_i \in \mathbb{T}_\Sigma(\underline{m})$, $\sigma\colon \underline{m} \to \underline{n}$.

given m terms as inputs, return n terms as output

$[t_1/x_1,\ldots,t_n/x_n]$ written just as $\langle t_1,\ldots,t_n\rangle$



Sequential composition is substitution application.

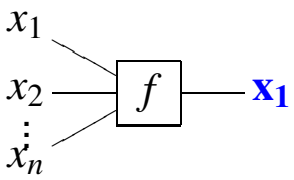
Brackets can be omitted **ex. a, b const. and f binary**

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

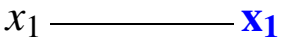
Categories of Substitutions (III)

Algebraic Theories **(Law)**: **finite subs on Σ generated by**

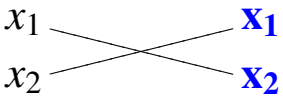
Th. $\langle \mathbf{f} \rangle : \underline{n} \to \underline{1}$ **for** $f \in \Sigma$ **with arity** n



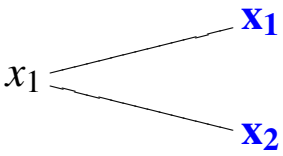
Th. $\langle \mathbf{x_1} \rangle : \underline{1} \to \underline{1}$



Th. $\langle \mathbf{x_2}, \mathbf{x_1} \rangle : \underline{2} \to \underline{2}$

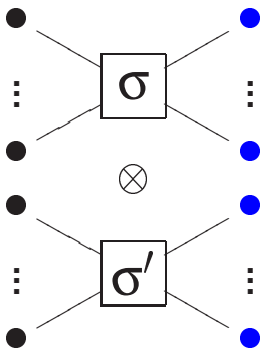


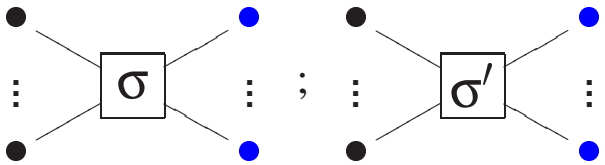
Th. $\langle \mathbf{x_1}, \mathbf{x_1} \rangle : \underline{1} \to \underline{2}$



Th. $\langle \rangle : \underline{1} \to \underline{0}$

composing with tensor $_ \otimes _$ **and application** $_ ; _$





Bisimulation and SOS

Bisimulation:

$t_1 \sim t_2$ and $t_1 \xrightarrow{a} s_1$ imply $t_2 \xrightarrow{a} s_2$ with $s_1 \sim s_2$

(and vice versa)

Bisimilarity $\simeq$ is the largest bisimulation

Is $\simeq$ a congruence? $t_i \simeq s_i \stackrel{?}{\Rightarrow} f(t_1,\ldots,t_n) \simeq f(s_1,\ldots,s_n)$

De Simone, positive GSOS, `tyft/tyxt` formats

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

Conclusions $C[x_1,\ldots,x_n] \xrightarrow{a} t$ never allowed!

What about structural axioms?

Example

Example 1 $\Sigma = \{nil, \alpha._-, \bar{\alpha}._-, _ \mid _\}$

sync. axiom plus rules that propagates τ through $_ \mid _$ (asynch.)

$$\frac{}{\lambda.x_1 \mid \bar{\lambda}.x_2 \xrightarrow{\tau} x_1 \mid x_2} \qquad \frac{x_1 \xrightarrow{\tau} x_1'}{x_1 \mid x_2 \xrightarrow{\tau} x_1' \mid x_2} \qquad \frac{x_2 \xrightarrow{\tau} x_2'}{x_1 \mid x_2 \xrightarrow{\tau} x_1 \mid x_2'}$$

$\alpha.nil \simeq \bar{\alpha}.nil$, *but consider the context* $x_1 \mid \bar{\alpha}.nil$*:*

$\alpha.\mathbf{nil} \mid \bar{\alpha}.\mathbf{nil} \xrightarrow{\tau} \mathbf{nil} \mid \mathbf{nil}$ $\qquad$ $\bar{\alpha}.\mathbf{nil} \mid \bar{\alpha}.\mathbf{nil} \not\rightarrow$

Hence $\alpha.nil \mid \bar{\alpha}.nil \not\simeq \bar{\alpha}.nil \mid \bar{\alpha}.nil$

Example 2 *In the running example* $\alpha.nil \not\simeq_{\mathrm{d}} \bar{\alpha}.nil$

To see this, take $\mathbf{C[x_1] = x_1 \mid \bar{\alpha}.nil}$

Dynamic Bisimulation

With *dynamic bisimulation*, all moves can put the states under comparison in the same context (note the $C[_]$)

$t_1 \sim_d t_2$ and $C[t_1] \xrightarrow{a} s_1$ imply $C[t_2] \xrightarrow{a} s_2$ with $s_1 \sim_d s_2$
(and vice versa)

Dynamic bisimilarity $\simeq_d$ is the largest dyn. bisimulation

$\simeq_d$ is the coarsest congr. that is also a (ordinary) bisim.

We introduce an alternative, original presentation:

Definition 1 *Given* LTS $L = (\mathbb{T}_\Sigma, \Lambda, \rightarrow)$
dynamic extension $\widehat{L} = (\mathbb{T}_\Sigma, \Lambda', \rightarrow \cup \rightrightarrows)$, *where:*
$s \xrightarrow{C[_]} C[s]$ *for all* $s \in \mathbb{T}_\Sigma$ *and all unary contexts* $C[_]$

Proposition 1 $s \simeq_d t$ *in* L *iff* $s \simeq t$ *in* $\widehat{L}$

Main Features of Tile Logic

TL is a logic for modular descriptions of the dynamic evolution of concurrent systems

- Suited to deal with **synchronization issues**
- General notion of **configuration** and of **observation**
- Defines compositional models of computation in **space** and in **time**
- Allows to reason about **contexts**
- Admits suggestive **graphical presentation**
- Can model **heterogeneous systems**
- Yields an algebra of **(open) computation proofs**
- provides a **categorical framework**

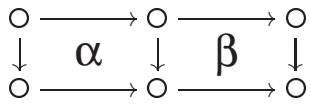
TL has been applied to
process algebras (GM,BMM,BFMM)
message passing systems (BMM)
coordination languages (MR)
causal and located systems (FM)
π-calculus (FM,BM)
actor systems (MT)

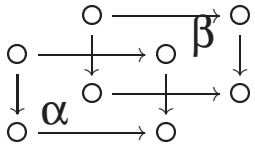
Tile Model

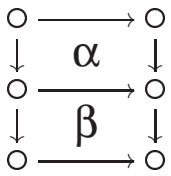
A ***tiles system*** is a tuple $\mathcal{R} = (\mathcal{H}, \mathcal{V}, N, R)$ where:
$\mathcal{H}$ monoidal category of ***configurations***
$\mathcal{V}$ monoidal category of ***observations***
$\mathcal{H}$ and $\mathcal{V}$ have the same objects (called ***interfaces***)
N is the set of *tile names*
$R: N \to \mathcal{H} \times \mathcal{V} \times \mathcal{V} \times \mathcal{H}$

if $R(\alpha) = \langle s, a, b, t \rangle$, written α: $s \overset{a}{\underset{b}{\to}} t$, then $\begin{array}{ccc} x & \xrightarrow{s} & y \\ {\scriptstyle a}\downarrow & \alpha & \downarrow{\scriptstyle b} \\ z & \xrightarrow[t]{} & w \end{array}$

Tiles can be composed







Identities

$\begin{array}{ccc} x & \xrightarrow{s} & y \\ {\scriptstyle id_x}\downarrow & id_s & \downarrow{\scriptstyle id_x} \\ x & \xrightarrow[s]{} & y \end{array}$ $\qquad$ $\begin{array}{ccc} x & \xrightarrow{id_x} & x \\ {\scriptstyle a}\downarrow & id_a & \downarrow{\scriptstyle a} \\ z & \xrightarrow[id_z]{} & z \end{array}$

(possibly auxiliary tiles)

Tile Logic and Tile Formats

Given $\mathcal{R}$, the associated *tile logic* is obtained by adding some canonical 'auxiliary' tiles and then closing by composition both auxiliary and basic tiles.

aux. tiles can e.g. model isomorphic transf. of interfaces

$$\mathcal{R} \vdash s \frac{a}{b} t$$

Starting from signatures (generating $\mathcal{H}$ and $\mathcal{V}$)

$$\frac{\{x_i \xrightarrow{a_i} y_i\}}{C[x_1,\ldots,x_n] \xrightarrow{a} D[y_1,\ldots,y_n]}$$

$$\begin{array}{ccc} \underline{n} & \xrightarrow{C[x_1,\ldots,x_n]} & \underline{1} \\ {\scriptstyle a_1 \otimes \cdots \otimes a_n}\downarrow & & \downarrow{\scriptstyle a} \\ \underline{n} & \xrightarrow[D[x_1,\ldots,x_n]]{} & \underline{1} \end{array}$$

note the canonical renaming of the y's

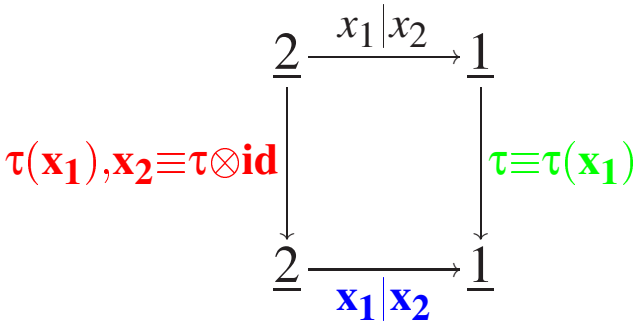
Algebraic format **(GM97)**: horizontal cartesianity

Monoidal format **(MM97)**: no auxiliary tiles

Term format **(BMM98)**: horizontal and vertical cartesianity

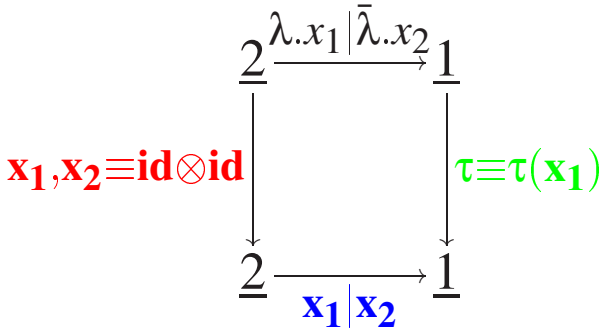
Example

$$\frac{x_1 \xrightarrow{\tau} x_1'}{x_1 \mid x_2 \xrightarrow{\tau} x_1' \mid x_2}$$



(Basic Source)

$$\frac{}{\lambda.x_1 \mid \bar{\lambda}.x_2 \xrightarrow{\tau} x_1 \mid x_2}$$



(Not Basic Source)

Tile Bisimulation and Tile Decomposition

Tiles allow for a very general notion of bisimulation (note that t_1 and t_2 can be open terms)

$\mathbf{t_1} \sim_{\mathrm{t}} \mathbf{t_2}$ and $\mathcal{R} \vdash t_1 \xrightarrow[b]{a} s_1$ imply $\mathcal{R} \vdash t_2 \xrightarrow[b]{a} s_2$ with $\mathbf{s_1} \sim_{\mathrm{t}} \mathbf{s_2}$ (and vice versa)

Tile bisimilarity $\simeq_{\mathrm{t}}$ is the largest tile bisimulation

Definition 2 *A tile system $\mathcal{R}$ enjoys the* **decomposition property** *if for all $\mathcal{R} \vdash s \xrightarrow[b]{a} t$*

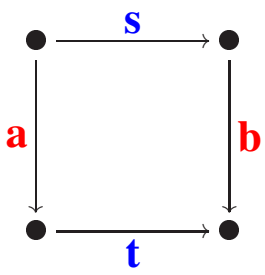
1. *if $s = s_1; s_2$ then*
   $\mathcal{R} \vdash s_1 \xrightarrow[c]{a} t_1$, $\mathcal{R} \vdash s_2 \xrightarrow[b]{c} t_2$ *with* $t = t_1; t_2$

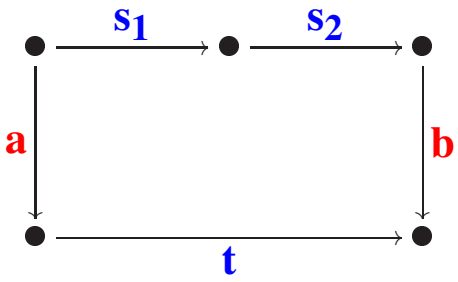
2. *if $s = s_1 \otimes s_2$ then*
   $\mathcal{R} \vdash s_1 \xrightarrow[b_1]{a_1} t_1$, $\mathcal{R} \vdash s_2 \xrightarrow[b_2]{a_2} t_2$,
   with $a = a_1 \otimes a_2$, $b = b_1 \otimes b_2$ *and* $t = t_1 \otimes t_2$

Proposition 2 *If $\mathcal{R}$ enjoys the decomposition property, then tile bisimilarity is a congruence*

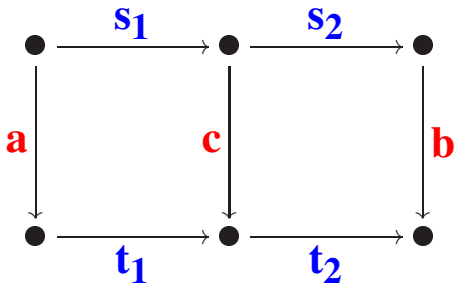
For basic source and tile bisimilarity congr. see (CONCUR 2000)

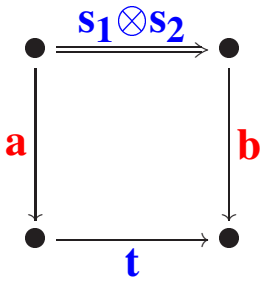
Decomposition Illustrated



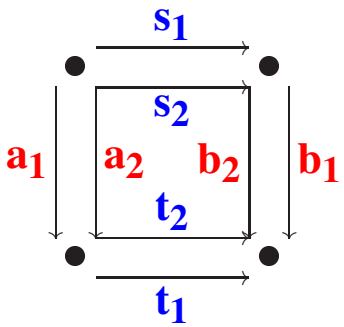


implies





implies



Ground Tile Bisimulation

Being interested in closed terms (*complete systems*), we introduce ***ground tile bisimulation*** (note the trivial trigger $id_{\underline{0}}$, with t_1 and t_2 ground)

$\mathbf{t_1} \sim_{\mathrm{g}} \mathbf{t_2}$ and $\mathcal{R} \vdash t_1 \xrightarrow{\underline{id_0}}{a} s_1$ imply $\mathcal{R} \vdash t_2 \xrightarrow{\underline{id_0}}{a} s_2$ with $\mathbf{s_1} \sim_{\mathrm{g}} \mathbf{s_2}$
(and vice versa)

Not defined on contexts, but on ground terms!

Ground tile bisimilarity $\simeq_{\mathrm{g}}$ is the largest ground tile bisim.

What is the analogous of decomposition property?

Ground Decomposition

Definition 3 *A term (resp. monoidal) tile system $\mathcal{R}$ enjoys the* **ground decomposition property** *if*
for all $\mathcal{R} \vdash s \xrightarrow[b]{id_0} t$

1. *if* $s = C[s_1]$
   (C (resp. linear) unary context), then
   $\mathcal{R} \vdash s_1 \xrightarrow[c]{id_0} t_1$, $\mathcal{R} \vdash C[x_1] \xrightarrow[b]{c} D[x_1]$ *with* $t = D[t_1]$

Theorem 1 *If a term (resp. monoidal) tile system $\mathcal{R}$ enjoys the ground decomposition property, then ground tile bisimilarity is a congruence*

The ground decomposition property can be enforced by syntactical constraints on basic tiles.

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

But in the presence of structural axioms this may fail!

Dynamic Tile Bisimulation

How to obtain the coarsest congruence in any situation?

Dynamic Extension $\widehat{\mathcal{R}}$:
Given $\mathcal{R}$ we add (for all $f \in \Sigma$):
the **auxiliary observation** $\tilde{f}$ (of arity n) and
the **auxiliary tiles**:

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

For t generic context, $\tilde{\mathbf{t}}$ denotes the associated observation
(e.g., if $\mathbf{t} = \mathbf{f}(\mathbf{f}(\mathbf{a}, \mathbf{x_1}), \mathbf{b})$ then $\tilde{\mathbf{t}} = \tilde{\mathbf{f}}(\tilde{\mathbf{f}}(\tilde{\mathbf{a}}, \mathbf{x_1}), \tilde{\mathbf{b}})$)

Lemma 1 *Given a term (resp. monoidal) tile system $\mathcal{R}$, for each (resp. linear) context $t : \underline{n} \to \underline{1}$ we have:*
$\widehat{\mathcal{R}} \vdash x_1, \ldots, x_n \xrightarrow[\tilde{t}]{x_1,\ldots,x_n} t$ *and* $\widehat{\mathcal{R}} \vdash t \xrightarrow[x_1]{\tilde{t}} x_1$

Congruence Results

Theorem 2 *Given a term (resp. monoidal) tile system $\mathcal{R}$, the ground tile bisimilarity defined on $\widehat{\mathcal{R}}$ is a congruence.*

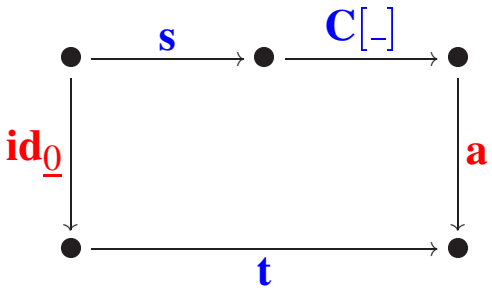
Definition 4 *For $\mathcal{R}$ a term (resp. monoidal) tile system, the* LTS *associated to $\mathcal{R}$ is $L_{\mathcal{R}}$ where*

$s \xrightarrow{a} t$ *iff* $\mathcal{R} \vdash s \overset{id_0}{\underset{a}{\Rightarrow}} t$

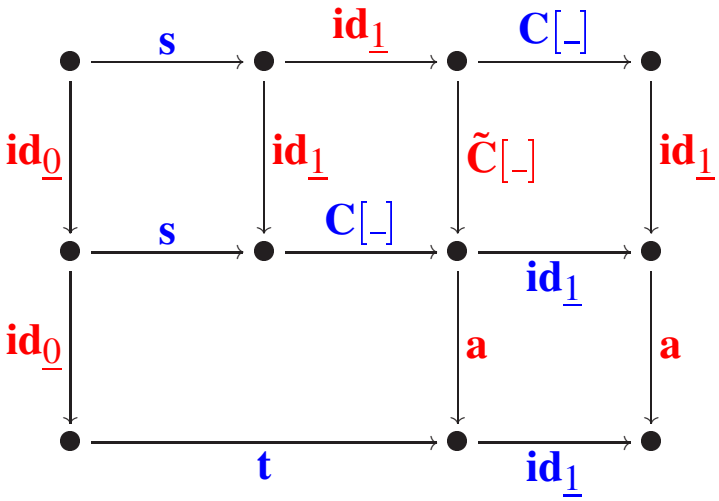
Theorem 3 *Ground tile bisimilarity on $\widehat{\mathcal{R}}$, denoted by $\simeq_{\hat{\mathrm{g}}}$, coincides with the dynamic bisimilarity on $L_{\mathcal{R}}$.*

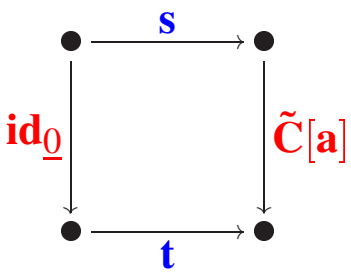
This reconciles dynamic extension and dynamic bisimilarity

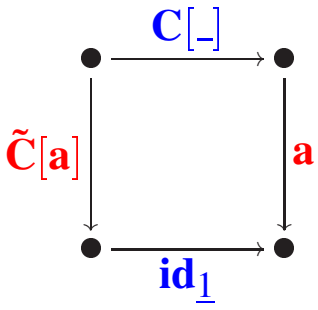
Congruence Proof



remind that $s; C[_] = C[s]$







Example

Example 3 *In our example let* $_ \mid _$ *be* ***associative*** *(but neither commutative nor with unit)*

$$\alpha.nil \mid \bar{\alpha}.nil \simeq_g \beta.nil \mid \bar{\beta}.nil \not\simeq_{\hat{g}} \alpha.nil \mid \bar{\alpha}.nil$$

consider the context $\mathbf{C}[\mathbf{x_1}] = \bar{\alpha}.\mathbf{nil} \mid \mathbf{x_1} \mid \alpha.\mathbf{nil}$***, then*** $\mathbf{C}[\alpha.\mathbf{nil} \mid \bar{\alpha}.\mathbf{nil}]$ ***can move twice, while*** $\mathbf{C}[\beta.\mathbf{nil} \mid \bar{\beta}.\mathbf{nil}]$ ***only once***

Instead, for $t_\lambda = nil \mid \lambda.nil \mid \bar{\lambda}.nil \mid nil$

$$t_\alpha \simeq_g t_\beta \simeq_{\hat{g}} t_\alpha$$

the nil ***processes block the interference of the environment***

Concluding Remarks

Tile logic as an **abstract, compositional framework** suitable to deal in a uniform way with

- ⋆ structural axioms
- ⋆ dynamic bisimulation
- ⋆ open ended systems
- ⋆ open terms (CONCUR 2000)

Exploiting the analogy between configurations and observations, dynamic bisimulation can be handled with:

- ⋆ finitary enrichment of the specification
- ⋆ pictorial congruence proof

Related works by Karen Bernstein, Robin Milner, Peter Sewell, James Leifer, Andrea Corradini, Reiko Heckel, Fabio Gadducci on **observing contexts**