TOSCA'01: Workshop finale progetto TOSCA - Udine - Italy - 19–21 November 2001

Comparing Higher-Order Encodings in LF and TL

Part II: TL and TL encodings, comparison with LF, conclusion

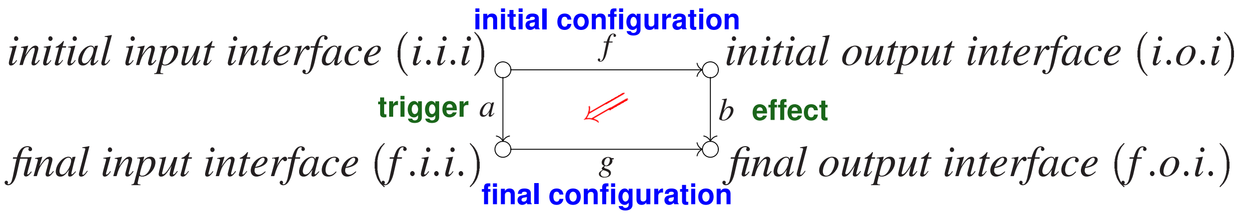
Roberto Bruni Univ. Pisa
joint work with
Furio Honsell Univ. Udine
Marina Lenisa Univ. Udine
Marino Miculan Univ. Udine

Research supported by MURST project *TOSCA*: Tipi, Ordine Superiore e Concorrenza

What Marina left to say

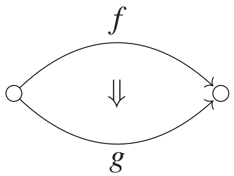
- tile logic (**TL**)
- higher-order tiles (**HOTL**) and cartian closed double categories (**CCDC**'s)
- a snapshot of late **π-calculus** encoding
- a snapshot of lazy simply typed **λ-calculus** encoding
- **comparison** between LF and TL (based on the two encodings)
- a few **conclusions** and **future work**

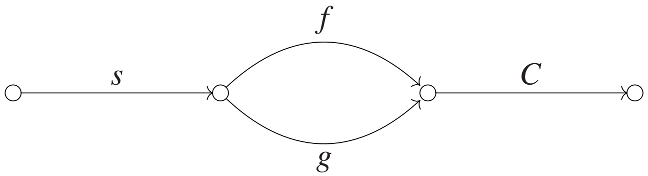
aim: modular description of the dynamics of concurrent systems (in **space** and in **time**)



RL

$f(x) \Rightarrow g(x)$

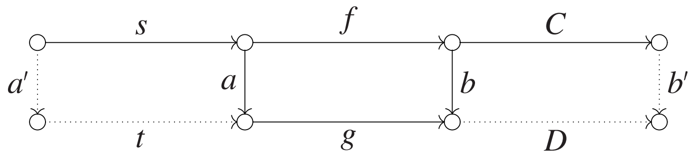




$\forall s, C[\cdot]$, then $C[f(s)] \Rightarrow C[g(s)]$

TL

$$\frac{x \xrightarrow{a} y}{f(x) \xrightarrow{b} g(y)} \qquad f \xrightarrow[b]{a} g$$



$$\forall s, C[\cdot], \text{ then } \frac{s \xrightarrow[a]{a'} t \ \wedge\ C \xrightarrow[b']{b} D}{C[f(s)] \xrightarrow[b']{a'} D[g(t)]}$$

sequents can be composed horizontally, vertically and in parallel

features: open configurations, visual presentation, categorical models

Theory of Configurations Fix the signature of **configurations**identifying the syntactic constructors and freely adjoin the **auxiliary structure** needed in the category of configurations (e.g., symmetries, cartesian operations);

Theory of Observations Fix the signature of **observations** (e.g. ordinary LTS's labels are usually seen as unary operators) and freely adjoin the **auxiliary structure** needed in the category of observations;

Theory of Tiles Define the basic operational rules for the dynamics of system components, and fix the **auxiliary tiles** to be freely adjoined.

issues: tile bisimilarity, tile traces, congruence formats, graphical proofs, applications

Moving to Higher-Order Tile Logic

monoidal structure: minimal tile model

symmetric monoidal structure: (consistent) permutations in the two dimensions

cartesianity: (consistent) pairing and projections

cartesian closed structure: higher-order configurations and observations

Example a *chemical solution* (à la CHAM) where functions and arguments float together

$$\begin{tikzcd} \Gamma \arrow[r, "\mathbf{P}:\tau^{\sigma},\mathbf{M}:\sigma,\mathbf{N}:\sigma"] \arrow[d, "\mathbf{id}_{\Gamma}"'] \arrow[dr, phantom, "\Leftarrow"] & f:\tau^{\sigma},x:\sigma,y:\sigma \arrow[d, "\mathbf{f\ x,y}"] \\ \Gamma \arrow[r, "\mathbf{PM}:\tau,\mathbf{N}:\sigma"'] & z:\tau,w:\sigma \end{tikzcd}$$

we expect consistency of horizontal/vertical applications

Roberto Bruni (bruni@di.unipi.it)

exploit the correspondence between ccc and simply typed λ-calculus to define a **double λ-notation** on the basis of **cartesian closed double categories**
note: **only object (interface) abstraction is possible**

double λ-signature: $\Sigma = \langle \overset{\textit{type const.}}{\mathbb{B}}, \overset{\textit{horiz. term const.}}{\mathbb{H}}, \overset{\textit{vert. term const.}}{\mathbb{V}}, \overset{\textit{tile term const.}}{\mathbb{C}} \rangle$

$h \in \mathbb{H}$ and $v \in \mathbb{V}$ (with $\varnothing \vdash h : \sigma$ and $\varnothing \vdash v : \tau$) can have h.o. types σ and τ;
$\alpha \in \mathbb{C}$ has a *closed contour type* (defining the tile border)

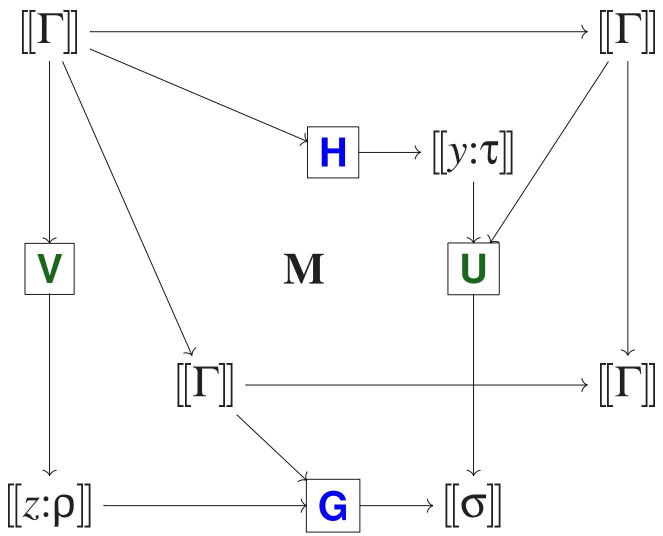
a generic contour type $\mathbf{t}$ has the form $\mathbf{H} : \tau \xrightarrow[\mathbf{U}:\sigma^\tau]{\mathbf{V}:\rho} \mathbf{G} : \sigma^\rho$; $\mathbf{t}$ is *well-typed* under Γ if H, G, V and U are well-typed under Γ, and it is *closed* if H, G, V and U are closed.

$M ::= \alpha \mid h \mid v \mid x \mid \star \mid \langle M, M \rangle^\xi \mid Proj_i^\xi M \mid \lambda^\xi x : \sigma . M \mid M @^\xi M \mid M \xi M$ (for $\xi \in \{*, \cdot, \triangleleft, \triangleright\}$)

type sentences have the form: $\Gamma \vdash M \square\ \mathbf{H} : \tau \xrightarrow[\mathbf{U}:\sigma^\tau]{\mathbf{V}:\rho} \mathbf{G} : \sigma^\rho$

Γ is a global environment for M (names in Γ can be used in H, V, U and G)

$$\Gamma \vdash M \,\square\, \mathbf{H} : \tau \xrightarrow[\lambda y.U:\sigma^{\tau}]{\mathbf{V}:\rho} \lambda z.G : \sigma^{\rho}$$



this makes evident that Γ is a global environment

Encoding the Pi-Calculus in TL

reverse observations for handling names

final configuration

final input interface — *final output interface*

trigger — **effect**

$$\begin{tikzcd} \circ \arrow[r, "H"] \arrow[d, "V"'] \arrow[dr, phantom, "\Rightarrow"] & \circ \arrow[d, "U"] \\ \circ \arrow[r, "G"'] & \circ \end{tikzcd}$$

initial input interface — *initial output interface*

initial configuration

fix a compact notation: $\mathbf{H} : \tau \xrightarrow[\lambda y:\tau.U:\sigma^{\tau}]{\mathbf{V}:\rho} \lambda z:\rho.G : \sigma^{\rho}$ is written $\mathbf{G}[\mathbf{V}/z] \Rightarrow \mathbf{U}[\mathbf{H}/y] : \sigma$

Interfaces	**Configurations**	**Observations**	**ProcessMapping**
n	$\mathbf{0} : p$	$\tau : p \to p$	$[\![0]\!] = \mathbf{0}$
p	$\tau_p : p \to p$	$out : \langle n,n \rangle \to p \to p$	$[\![\tau.P]\!] = \tau_p\ [\![P]\!]$
	$in_p : n \to (n \to p) \to p$	$in : n \to (n \to p) \to p$	$[\![x(y).P]\!] = in_p\ x\ \lambda y : n.[\![P]\!]$
	$out_p : \langle n,n \rangle \to p \to p$	$bout : n \to (n \to p) \to p$	$[\![\bar{x}y.P]\!] = out_p\ \langle x,y \rangle\ [\![P]\!]$
	$\mid\ : \langle p,p \rangle \to p$		$[\![P_1 \mid P_2]\!] = [\![P_1]\!] \mid [\![P_2]\!]$
	$\nu : (n \to p) \to p$		$[\![(\nu x)P]\!] = \nu\ \lambda x : n.[\![P]\!]$
	$if : \langle n,n \rangle \to p \to p$		$[\![[x=y]P]\!] = if\ \langle x,y \rangle [\![P]\!]$

Tile term constants

$$q : p \vdash \mathtt{Tau}@^{\triangleleft} q \,\Box\, \tau_p\ [q] \ \Rightarrow\ \tau\ [q]\ : p$$

$$x : n, r : n \to (p \times n) \vdash \mathtt{Open}@^{\triangleleft} x @^{\triangleleft} r \,\Box\, \nu[\lambda y : n.out\ \langle x, Proj_2(ry) \rangle Proj_1(ry)] \ \Rightarrow\ bout\ x[\lambda y : n.Proj_1(ry)]\ : p$$

Encoding the Pi-Calculus in TL: an Example

$$\frac{P \xrightarrow{\bar{x}y} P'}{(\nu y)P \xrightarrow{\bar{x}(w)} P'\{w/y\}} \; y \neq x, w \notin fn((\nu y)P') \qquad \text{(OPEN)}$$

$$\begin{tikzcd}
\Gamma, y : n \arrow[r, "\langle [\![P']\!], y\rangle"] \arrow[d] \arrow[dr, phantom, "\mathbf{M}"] & q : p, y' : n \arrow[d, "out\langle x, y'\rangle q"] \\
\varepsilon \arrow[r, "[\![P]\!]"'] & : p
\end{tikzcd}
\qquad
\Gamma, y : n \vdash \mathtt{M} \,\square\, [\![P]\!][\,] \;\Rightarrow\; out\ \langle x, [y]_1\rangle[[\![P]\!]]_2 \ : p$$

$$\begin{tikzcd}
\Gamma \arrow[r, "\lambda y : n.\langle [\![P']\!], y\rangle"] \arrow[d] \arrow[dr, phantom, "\lambda^{*}\mathbf{y}.\mathbf{M}"] & s : (p \times n)^n \arrow[d, "\lambda y' : n.out\langle x, Proj_2(sy')\rangle Proj_1(sy)"] \\
\varepsilon \arrow[r, "\lambda y : n.[\![P]\!]"'] & : p^n
\end{tikzcd}
\qquad
\begin{tikzcd}
s : (p \times n)^n \arrow[r, "\lambda w : n.Proj_1(sw)"] \arrow[d] \arrow[dr, phantom, "\mathtt{Open@}^{\triangleleft}x"] & q_2 : p^n \arrow[d, "bout\ x\ q_2"] \\
q_1 : p^n \arrow[r, "\nu\ q_1"'] & : p
\end{tikzcd}$$

$$\begin{tikzcd}
\Gamma \arrow[r, "\lambda w : n.[\![P'\{w/y\}]\!]"] \arrow[d] \arrow[dr, phantom, "(\lambda^{*}\mathbf{y}.\mathbf{M}) * (\mathtt{Open@}^{\triangleleft}\mathbf{x})"] & q : p^n \arrow[d, "bout\ x\ q"] \\
\varepsilon \arrow[r, "[\![(\nu y)P]\!]"'] & : p
\end{tikzcd}$$

Proposition If $P \xrightarrow{\mu} Q$ then $fn(P) \vdash M \Box S \xrightarrow[\llbracket\mu\rrbracket]{id} \llbracket P \rrbracket$, in $\mathcal{R}_{late}$

where $S = \llbracket Q \rrbracket$ if μ is free,

while $S = \lambda y : n.\llbracket Q \rrbracket$ if μ is bound with object y.

- the proof is by induction on the proof of $P \xrightarrow{\mu} Q$;
- define a denotational mapping from proof terms of late LTS to well-typed tile terms;
- $\mathcal{R}_{late}$ defines a *richer* LTS.

Conjecture Tile bisimilarity is finer than late bisimilarity.

processes with different sets of free names are not tile bisimilar

Interfaces	Configurations	Observations	Tile term constants
ι	$c\colon \star \to \iota$	$\hat{c}\colon \star \to \iota$	$\vdash \mathtt{Dyn}_c \,\square\, \hat{c}[\star] \;\Rightarrow\; c[\star] \;:\iota$
κ	$0\colon \star \to \kappa$	$\downarrow_c\colon \iota \to \kappa$	$\vdash \mathtt{Const}_c \,\square\, \downarrow_c [c] \;\Rightarrow\; 0[\star] \;:\kappa$

- observing configurations;
- since $\star$ is terminal we cannot take $\downarrow_c\colon \iota \to \star$ (otherwise $\downarrow_c = \downarrow_{c'}$ for all c, c');
- thus type constant κ and horizontal term constant 0 are used;
- this time tile *are not* reversed:

$$
\begin{tikzcd}
\circ \arrow[r, "\star"] \arrow[d, "\star"'] \arrow[dr, phantom, "\mathtt{Dyn}_c"] & \circ \arrow[d, "\hat{c}"] \\
\circ \arrow[r, "c"'] & \circ
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
\circ \arrow[r, "c"] \arrow[d, "\star"'] \arrow[dr, phantom, "\mathtt{Const}_c"] & \circ \arrow[d, "\downarrow_c"] \\
\circ \arrow[r, "0"'] & \circ
\end{tikzcd}
$$

Proposition For any closed configuration M, we have that $\mathcal{R}_\lambda \vdash \star \xrightarrow[\hat{M}]{\star} M$,
where $\hat{M}$ is the term obtained by the syntactic replacement of all c by $\hat{c}$.

Corollary For any closed configurations $M \in \Lambda_\emptyset^{\sigma\to\tau}$ and $N \in \Lambda_\emptyset^{\sigma}$, we have $\mathcal{R}_\lambda \vdash M \xrightarrow[\star]{@\hat{N}} MN$,
where $@ \triangleq \lambda x : \tau.\lambda f : \sigma \to \tau. fx$.

Proposition **Tile bisimilarity coincides with applicative bisimilarity.**

- τ moves of the applicative LTS are mapped to vertical identities;
- the two kinds of labeled steps have their counterparts in TL;
- the type system allows for correct applications only.

Comparison Between Encodings

the choice of π and λ is instrumental in comparing TL and LF (not for the novelty of the encoding technique)

	LF	TL
π	**encoding driven by syntax**	**encoding driven by strong typing**
	labels are separate entities $\overset{\tau}{\longmapsto}$, $\overset{out(x,y)}{\longmapsto}$, $\overset{in(x)}{\longmapsto\!\!\!\!\rightarrow}$, $\overset{bout(x)}{\longmapsto\!\!\!\!\rightarrow}$	**labels are h.o. vertical operators** τ, $out\ xy$, $in\ x$, $bout\ x$
	only curryed operators	**cartesian pairing**
	transition proved by inhabitance	**semantic domain for proofs**
	late bis. needs stronger type theory	**tile bisimilarity or ad hoc def.**
	handle name abstraction, creation, passing, application	
λ	**transition proved by inhabitance**	**dynamic application is built-in**
	can handle other evaluation strategies **can handle extrinsic type systems**	**up-to α, β, η** **untyped λ needs some extension**
	handle name abstraction, application	

Overall Comparison

LF	TL
more expressive power	**only object abstraction**
tools (automatization)	**visual formalism**
flat view	**space/time distinction**
functor categories	**categorical view**
delegate to the metalanguage many aspects of object systems (e.g. name binding, name passing, name creation, name freshness, capture-avoiding substitutions, α-conversion	
several case studies presented in the literature	
HOA**S** (**S**=Syntax)	HOA**S** (**S**=Semantics)

Future Work

- extend the comparison via a thorough exploration of other semantics for the two object languages considered here (e.g. call-by-value and untyped λ-calculus, early π-calculus, barbed equivalence);

- consider other object systems (e.g. ambient, spi-calculus, polyadic π-calculus);

- syntactical encoding of HOTL in LF (feasible);

- semantical encoding of LF in HOTL (difficult, because of dependent types)