A proof theoretic framework for process verification

Cosimo Perini Brogi
IMT School for Advanced Studies Lucca

Joint work with Rocco De Nicola (CNR, Pisa) & Omar Inverso (GSSI, L'Aquila)

PACMAN 2 Workshop
May 14$^{\text{th}}$ $\rightarrow$ 16$^{\text{th}}$, 2025
UniRoma3

Work supported by the project SERICS - PE0000014, financed within PNRR, M4C2 I.1.3, funded by the European Union - NextGenerationEU

General framework

Main goal

Apply proof-theoretic tools and techniques to formal verification of *generally specified* concurrent programs

General framework

Main goal

Apply proof-theoretic tools and techniques to formal verification of *generally specified* concurrent programs

How to reach it

- Exploit the interplay between (non-classical) logics, process algebras/calculi and labelled transition systems
- From the particular to the general, starting with well-established specification systems for process calculi

Stirling's research question

SCHOOL FOR ADVANCED STUDIES LUCCA

Theoretical Computer Science 49 (1987) 311–347 311
North-Holland

MODAL LOGICS FOR COMMUNICATING SYSTEMS

Colin STIRLING

Department of Computer Science, Edinburgh University, Edinburgh EH8 9YL, Scotland, U.K.

Abstract. Simple modal logics for Milner's SCCS and CCS are presented. We offer sound and complete axiomatizations of validity relative to these calculi as models. Also we present compositional proof systems for when a program satisfies a formula. These involve proof rules which are like Gentzen introduction rules except that there are also introduction rules for the program combinators of SCCS and CCS. The compositional rules for restriction (or hiding) and parallel combinators arise out of a simple semantic strategy.

Verification of modular processes *should* be modular

"[*C*]*ompositional, syntax-directed* proof systems" for verifying properties of concurrent systems expressed in the language of modal logics

Simpson's answer

ELSEVIER

Available online at www.sciencedirect.com

SCIENCE DIRECT®

The Journal of Logic and Algebraic Programming 60–61 (2004) 287–322

THE JOURNAL OF LOGIC AND ALGEBRAIC PROGRAMMING

www.elsevier.com/locate/jlap

Sequent calculi for process verification: Hennessy–Milner logic for an arbitrary GSOS☆

Alex Simpson

Laboratory for Foundations of Computer Science, School of Informatics, University of Edinburgh, King's Buildings, Edinburgh EH9 3JZ, UK

Abstract

We argue that, by supporting a mixture of "compositional" and "structural" styles of proof, sequent-based proof systems provide a useful framework for the formal verification of processes. As a worked example, we present a sequent calculus for establishing that processes satisfy assertions in Hennessy–Milner logic. The main novelty lies in the use of the operational semantics to derive introduction rules, on the left and right of sequents, for the operators of the process calculus. This gives a generic proof system applicable to any process algebra with an operational semantics specified in the GSOS format. Using a general algebraic notion of GSOS model, we prove a completeness theorem for the cut-free fragment of the proof system, thereby establishing the admissibility of the cut rule. Under mild (and necessary) conditions on the process algebra, an ω-completeness result, relative to the "intended" model of closed process terms, follows.

Verification of modular processes *can* be modular *and natural*

"[C]ompositional, structural and naturalness aspects of sequent-based proof follow from properties of the basic sequent calculus [. . .] [It is possible] to relate processes (or programs) to their logical properties [. . .] without breaking the fundamental structural properties of sequent calculus."

Our improvement

A more principled approach

Apply *contemporary* proof-theoretic techniques to enhance Simpson's idea and uniformly obtain a new family of modular sequent calculi for logical verification of concurrent processes.
The *motto*, after (Dyckhoff and Negri 2015), is

"Keep left & Geometrize!"

Our improvement

A more principled approach

Apply *contemporary* proof-theoretic techniques to enhance Simpson's idea and uniformly obtain a new family of modular sequent calculi for logical verification of concurrent processes.
The *motto*, after (Dyckhoff and Negri 2015), is

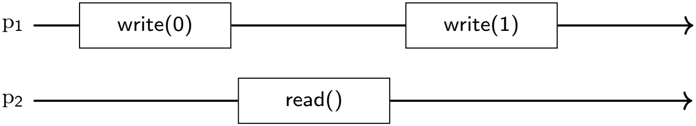
"Keep left & Geometrize!"

Results

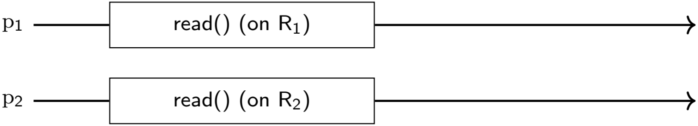
- Constructive cut-elimination from calculi for Hennessy-Milner logic and GSOS processes
- Substantial simplification of Simpson's proofs for structural and semantic completeness of this kind of calculi

Outline

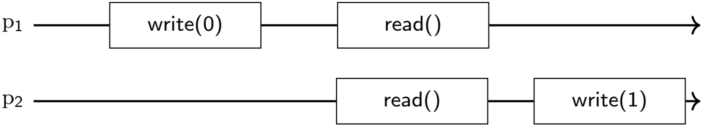
Concurrent processes



(a) *Concurrent* but *not parallel*



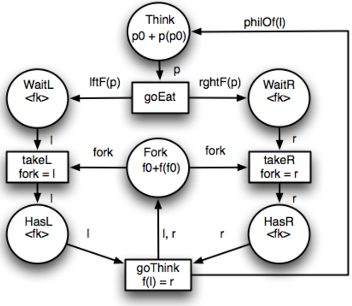
(b) *Parallel* but *not concurrent*



(c) *Concurrent* and *parallel*

Concurrent processes

Example from informatics




```
ADT Philosopher
Sort philo
Use fork
Ops
p0: → philo
p: philo → philo
lftF: philo → fork
rghtF: philo → fork
philoOf : fork → philo
Axioms
p(p(po))=p0
lftF(p0)=f0
lftF(p(ph))=f(lftF(ph))
rghtF(p0)=f(f0)
rghtF(p(ph))=f(rghtF(ph))
philOf(f0)=p0
philOf(f(fk))=p(philOf(fk))

With ph : philo, fk : fork
```

```
ADT Fork
Sort fork
Ops
f0: → fork
f: fork → fork
Axioms
f(f(f0))=f0

With fk : fork
```

Dining philosophers problem (WikiMedia, CC-BY-SA-3.0)

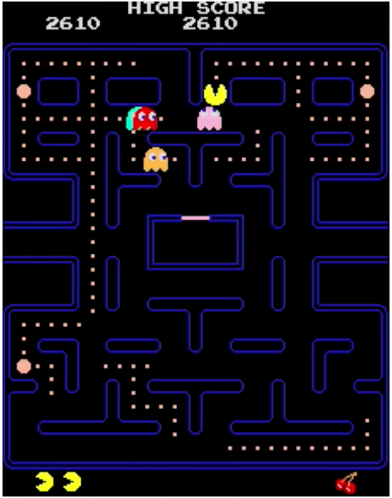
Concurrent processes

Example from biology

Starlings flocks, murmuration (WikiMedia, CC-BY-SA-2.0)

Concurrent processes

Example from. . .



In-game screenshot, (WikiMedia, CC-BY-3.0)

Process calculi

Key insight

Hoare's and Milner's proposal

Process calculi provide a syntactic characterisation of concurrent programs that is based on process operators building new process behaviours from simpler ones

To describe processes, focus on interactions!

SCHOOL FOR ADVANCED STUDIES LUCCA

Definition (LTS)

A *labelled transition system* $\mathcal{T} := \langle T, \mathcal{A}_\tau, \rightarrow \rangle$ consists of

- a set of states T,
- a set of actions $\mathcal{A}_\tau$ (including a "silent" τ), and
- a mapping $\rightarrow$ from actions to pair of states.

Process calculi

GSOS

$$\frac{\{x_i \xrightarrow{\mu_{ij}} y_{ij} \mid 1 \leq i \leq n, 1 \leq j \leq m_i\} \qquad \cup \qquad \{x_i \stackrel{\nu_{ik}}{\not\rightarrow} \mid 1 \leq i \leq n, 1 \leq k \leq \ell_i\}}{f(\vec{x}) \xrightarrow{\pi} p(\vec{x}, \vec{y})}$$

where:

- f is an operator on states;
- the x_i's and the y_{ij}'s ($1 \leq i \leq n$ and $1 \leq j \leq m_i$) are all distinct state variables;
- n, m_i and ℓ_i are natural numbers;
- $p(\vec{x}, \vec{y})$ is a state term with variables including at most the x_i's and y_{ij}'s; and
- the μ_{ij}'s, ν_{ik}'s and π are actions from $\mathcal{A}_\tau$.

Behavioural equivalence

Via structural semantics

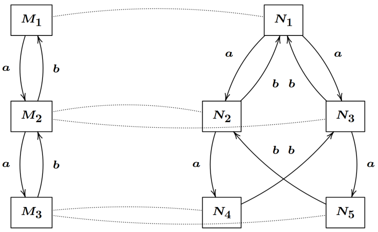
Using the structural semantics, labelled transition systems are rigorously associated to concurrent processes:
Just consider $\mathcal{T} := \langle T, \mathcal{A}_\tau, \rightarrow \rangle$, where $\rightarrow$ applied to μ is the least relation on T generated by the rules of the structural semantics.

Behavioural equivalence

Via structural semantics

Using the structural semantics, labelled transition systems are rigorously associated to concurrent processes:
Just consider $\mathcal{T} := \langle T, \mathcal{A}_\tau, \rightarrow \rangle$, where $\rightarrow$ applied to μ is the least relation on T generated by the rules of the structural semantics.

When are two processes observationally identifiable?



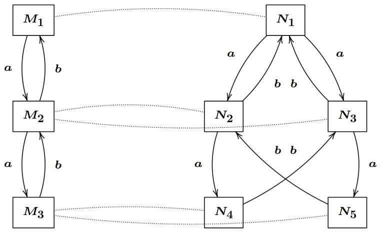
Two states are bisimilar (and we write $p \simeq q$) when there exists a bisimulation R such that pRq.

Behavioural equivalence

Via structural semantics

Using the structural semantics, labelled transition systems are rigorously associated to concurrent processes:
Just consider $\mathcal{T} := \langle T, \mathcal{A}_\tau, \rightarrow\rangle$, where $\rightarrow$ applied to μ is the least relation on T generated by the rules of the structural semantics.

When are two processes observationally identifiable?



Two states are bisimilar (and we write $p \simeq q$) when there exists a bisimulation R such that pRq.

Quotienting LTS over $\simeq$ provides a semantics of processes that focuses of (some) external behaviour of processes, abstracting from their specific identity.

Definition (Hennessy-Milner logic)

Formulas of HM are defined by the following grammar:

$$A \in \mathtt{Frm}_{\mathsf{HM}} ::= \top \mid \neg A \mid A \wedge B \mid \langle \mu \rangle A,$$

where $\mu \in \mathcal{A}_\tau$, $\neg$ and $\wedge$ denote classical negation and conjunction, resp.

Now, given the LTS $\langle T, \mathcal{A}_\tau, \rightarrow \rangle$, we can define a standard notion of local forcing as follows:

- $p \Vdash \top$ for any $p \in T$;
- $p \Vdash \neg A$ iff $p \nVdash A$;
- $p \Vdash A \wedge B$ iff $p \Vdash A$ and $p \Vdash B$;
- $p \Vdash \langle \mu \rangle A$ iff there exists a $q \in T$ such that $p \xrightarrow{\mu} q$ and $q \Vdash A$.

Theorem (Hennessy and Milner 1985)

Let's say that a state p of an LTS $\mathcal{T}$ is finitely branching if the set of states that are reachable in $\mathcal{T}$ from p is finite.
Then, given two finitely branching states p, q

$$p \simeq q \text{ iff, for any } A \in \mathtt{Frm}_{HM},\ p \Vdash A \text{ iff } q \Vdash A.^{a}$$

[a] The finite branching condition can be discarded if infinite conjunctions are allowed in the basic language.

Principled design

Basic language

Our proof system $\mathrm{G3HML}_{\mathrm{GSOS}}$ is based on the explicit internalisation of GSOS process algebras in standard sequent calculi.

Principled design

Basic language

Our proof system $\mathrm{G3HML}_{\mathrm{GSOS}}$ is based on the explicit internalisation of GSOS process algebras in standard sequent calculi.
We work with labelled formulas with shape:

$$\underbrace{\overbrace{p \xrightarrow{\mu} q}^{\text{Positive transitional}} \quad \mid \quad \overbrace{p \not\xrightarrow{\mu}}^{\text{Negative transitional}} \quad \mid \quad \overbrace{p \equiv q}^{\text{Congruence}}}_{\text{Structural atoms}} \quad \mid \quad \underbrace{p : A}_{\text{Forcing atoms}}$$

Principled design

Basic language

Our proof system G3HML$_{\text{GSOS}}$ is based on the explicit internalisation of GSOS process algebras in standard sequent calculi.
We work with labelled formulas with shape:

$$\underbrace{\overbrace{p \xrightarrow{\mu} q}^{\text{Positive transitional}} \;\Big|\; \overbrace{p \not\xrightarrow{\mu}}^{\text{Negative transitional}} \;\Big|\; \overbrace{p \equiv q}^{\text{Congruence}}}_{\text{Structural atoms}} \;\Big|\; \underbrace{p : A}_{\text{Forcing atoms}}$$

Sequents of G3HML$_{\text{GSOS}}$ are expressions $\Gamma \Rightarrow \Delta$, where Γ, Δ are finite multisets of labelled formulas, and structural atoms may occur only in Γ.

Principled design

Logical rules

$$\dfrac{}{\Gamma \Rightarrow \Delta, p : \top}\ R\top$$

$$\dfrac{\Gamma \Rightarrow \Delta, p : A}{p : \neg A, \Gamma \Rightarrow \Delta}\ L\neg \qquad \dfrac{p : A, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, p : \neg A}\ R\neg$$

$$\dfrac{p : A, p : B, \Gamma \Rightarrow \Delta}{p : A \wedge B, \Gamma \Rightarrow \Delta}\ L\wedge \qquad \dfrac{\Gamma \Rightarrow \Delta, p : A \qquad \Gamma \Rightarrow \Delta, p : B}{\Gamma \Rightarrow \Delta, p : A \wedge B}\ R\wedge$$

$$\dfrac{p \xrightarrow{\mu} y, y : A, \Gamma \Rightarrow \Delta}{p : \langle \mu \rangle A, \Gamma \Rightarrow \Delta}\ L\Diamond_{(!y)} \qquad \dfrac{p \xrightarrow{\mu} q, \Gamma \Rightarrow \Delta, p : \langle \mu \rangle A, q : A}{p \xrightarrow{\mu} q, \Gamma \Rightarrow \Delta, p : \langle \mu \rangle A}\ R\Diamond$$

Principled design

Compositional rules

$$\frac{\{x_i \xrightarrow{\mu_{ij}} y_{ij} \mid 1 \leq i \leq n, 1 \leq j \leq m_i\} \quad \cup \quad \{x_i \stackrel{\nu_{ik}}{\not\rightarrow} \mid 1 \leq i \leq n, 1 \leq k \leq \ell_i\}}{f(\vec{x}) \xrightarrow{\pi} p(\vec{x}, \vec{y})}$$

Principled design

Compositional rules

$$\frac{\{x_i \xrightarrow{\mu_{ij}} y_{ij} \mid 1 \leq i \leq n, 1 \leq j \leq m_i\} \qquad \cup \qquad \{x_i \overset{\nu_{ik}}{\not\to} \mid 1 \leq i \leq n, 1 \leq k \leq \ell_i\}}{f(\vec{x}) \xrightarrow{\pi} p(\vec{x}, \vec{y})}$$

Geometrize!

$$(\circ)\, \forall \vec{x}, \vec{y} : \left[\left(\bigwedge_{1\leq i\leq n, 1\leq j\leq m_i} (x_i \xrightarrow{\mu_{ij}} y_{ij}) \;\&\; \bigwedge_{1\leq i\leq n, 1\leq k\leq \ell_i} (x_i \overset{\nu_{ik}}{\not\to}\ )\right) \supset (f(\vec{x}) \xrightarrow{\pi} p(\vec{x}, \vec{y}))\right]$$

$$(\circ\circ)\, \forall \vec{r}, z : \left[(f(\vec{r}) \xrightarrow{\pi} z) \supset \left(\exists \vec{y} : p(\vec{r}, \vec{y}) \equiv z \;\&\; \bigwedge_{1\leq i\leq n, 1\leq j\leq m_i} (r_i \xrightarrow{\mu_{ij}} y_{ij}) \;\&\; \bigwedge_{1\leq i\leq n, 1\leq k\leq \ell_i} (r_i \overset{\nu_{ik}}{\not\to}\ )\right)\right]$$

Principled design

Compositional rules

$$\frac{\{x_i \xrightarrow{\mu_{ij}} y_{ij} \mid 1 \le i \le n, 1 \le j \le m_i\} \qquad \cup \qquad \{x_i \overset{\nu_{ik}}{\not\to} \mid 1 \le i \le n, 1 \le k \le \ell_i\}}{f(\vec{x}) \xrightarrow{\pi} p(\vec{x}, \vec{y})}$$

Geometrize!

$$(\circ)\, \forall \vec{x}, \vec{y} : \left[\left(\bigwedge_{1 \le i \le n, 1 \le j \le m_i} (x_i \xrightarrow{\mu_{ij}} y_{ij}) \;\&\; \bigwedge_{1 \le i \le n, 1 \le k \le \ell_i} (x_i \overset{\nu_{ik}}{\not\to}\,)\right) \supset (f(\vec{x}) \xrightarrow{\pi} p(\vec{x}, \vec{y}))\right]$$

$$(\circ\circ)\, \forall \vec{r}, z : \left[(f(\vec{r}) \xrightarrow{\pi} z) \supset \left(\exists \vec{y} : p(\vec{r}, \vec{y}) \equiv z \;\&\; \bigwedge_{1 \le i \le n, 1 \le j \le m_i} (r_i \xrightarrow{\mu_{ij}} y_{ij}) \;\&\; \bigwedge_{1 \le i \le n, 1 \le k \le \ell_i} (r_i \overset{\nu_{ik}}{\not\to}\,)\right)\right]$$

Keep left!

$$\frac{f(\vec{x}) \xrightarrow{\pi} p(\vec{x}, \vec{y}),\ \{x_i \xrightarrow{\mu_{ij}} y_{ij}\}_{1 \le i \le n, 1 \le j \le m_i},\ \{x_i \overset{\nu_{ik}}{\not\to}\,\}_{1 \le i \le n, 1 \le k \le \ell_i}, \Gamma \Rightarrow \Delta}{\{x_i \xrightarrow{\mu_{ij}} y_{ij}\}_{1 \le i \le n, 1 \le j \le m_i},\ \{x_i \overset{\nu_{ik}}{\not\to}\,\}_{1 \le i \le n, 1 \le k \le \ell_i}, \Gamma \Rightarrow \Delta}\; f\circ \qquad \frac{}{p \overset{\mu}{\not\to}\,, p \xrightarrow{\mu} q, \Gamma \Rightarrow \Delta}\; \overset{\mu}{\not\to} Def$$

$$\frac{\left\{p_h(\vec{r}, \vec{y}) \equiv z,\ \{r_i \xrightarrow{\mu_{ij}} y_{ij}\}_{1 \le i \le n, 1 \le j \le m_i},\ \{x_i \overset{\nu_{ik}}{\not\to}\,\}_{1 \le i \le n, 1 \le k \le \ell_i}, f(\vec{r}) \xrightarrow{\pi} z, \Gamma \Rightarrow \Delta\right\}_{1 \le h \le N}}{f(\vec{r}) \xrightarrow{\pi} z, \Gamma \Rightarrow \Delta}\; f\circ\circ_{(!\vec{y})}$$

$+$ Rules for $\equiv$-repl.

Worked-out examples

Choice, top-down

$$\textsc{Sum}_1 \; \frac{p \xrightarrow{\mu} p'}{p + q \xrightarrow{\mu} p'} \quad \rightsquigarrow \quad (p \xrightarrow{\mu} p') \to (p + q \xrightarrow{\mu} p')$$

$$\rightsquigarrow \quad \frac{p + q \xrightarrow{\mu} p', p \xrightarrow{\mu} p', \Gamma \Rightarrow \Delta}{p \xrightarrow{\mu} p', \Gamma \Rightarrow \Delta} \; \textsc{Sum}_{\circ 1}$$

Worked-out examples

Choice, bottom-up

$$\textsc{Sum}_1\ \frac{p \xrightarrow{\mu} p'}{p+q \xrightarrow{\mu} p'} + \textsc{Sum}_2\ \frac{q \xrightarrow{\mu} q'}{p+q \xrightarrow{\mu} q'} \qquad \rightsquigarrow$$

$$(p+q \xrightarrow{\mu} x) \to ((x \equiv p') \wedge (p \xrightarrow{\mu} x)) \vee ((x \equiv q') \wedge (q \xrightarrow{\mu} x)) \qquad \rightsquigarrow$$

$$\frac{x \equiv p', p \xrightarrow{\mu} x, p+q \xrightarrow{\mu} x, \Gamma \Rightarrow \Delta \qquad x \equiv q', q \xrightarrow{\mu} x, p+q \xrightarrow{\mu} x, \Gamma \Rightarrow \Delta}{p+q \xrightarrow{\mu} x, \Gamma \Rightarrow \Delta}\ \textsc{Sum}_{\circ\circ}$$

Main results

SCHOOL FOR ADVANCED STUDIES LUCCA

Theorem

G3HML$_{GSOS}$ *satisfies the following properties:*

Soundness: *If the sequent* $\Gamma \Rightarrow \Delta$ *is derivable, then* $\Gamma \vDash \Delta$

Completeness: *If the sequent* $\Gamma \Rightarrow \Delta$ *is not derivable, then it is possible to extract from the failed proof search an LTS-countermodel to* $\Gamma \Rightarrow \Delta$

Structural completeness:

- *Generalised initial sequents are derivable*
- *Substitution rule for states over variables are height-preserving admissible*
- *Weakening rules are height preserving admissible*
- *All the rules are height-preserving invertible*
- *Contraction rules are height-preserving admissible*

Cut elimination: *The cut rule can be effectively eliminated*

Put in perspective

- Substantial refinement of Simpson's original labelled sequent calculi for GSOS
- *More principled* formalisation and approach to verification of GSOS processes
- Our cut elimination algorithm as basic result for automation of verification tasks

Put in perspective

- Substantial refinement of Simpson's original labelled sequent calculi for GSOS
- *More principled* formalisation and approach to verification of GSOS processes
- Our cut elimination algorithm as basic result for automation of verification tasks

◇ Future extensions with more expressive logics

◇ Modular extensions for more general process formats

◇ Implementation of terminating proof-search strategies (for decidable settings)

◇ Development of tools for proof-based process verification and analysis

Put in perspective

IMT SCHOOL FOR ADVANCED STUDIES LUCCA

- Substantial refinement of Simpson's original labelled sequent calculi for GSOS
- *More principled* formalisation and approach to verification of GSOS processes
- Our cut elimination algorithm as basic result for automation of verification tasks

- ◇ Future extensions with more expressive logics
- ◇ Modular extensions for more general process formats
- ◇ Implementation of terminating proof-search strategies (for decidable settings)
- ◇ Development of tools for proof-based process verification and analysis

Many thanks for listening!

cosimo.perinibrogi@imtlucca.it

Process calculi

CCS

No rules for $\mathbf{0}$

$$\textsc{Act}\ \frac{}{\mu.p \xrightarrow{\mu} p}$$

$$\textsc{Def}\ \frac{p \xrightarrow{\mu} q \qquad p \triangleq k}{k \xrightarrow{\mu} q}$$

Modulo restrictions
on the structure of p and q!

$$\textsc{Ren}\ \frac{p \xrightarrow{\mu} q}{p[f] \xrightarrow{f(\mu)} q[f]}$$

$$\textsc{Res}\ \frac{p \xrightarrow{\mu} q}{p_{\setminus L} \xrightarrow{\mu} q_{\setminus L}}\ \mu,\overline{\mu} \notin L$$

$$\textsc{Sum}_1\ \frac{p \xrightarrow{\mu} p'}{p + q \xrightarrow{\mu} p'}$$

$$\textsc{Sum}_2\ \frac{q \xrightarrow{\mu} q'}{p + q \xrightarrow{\mu} q'}$$

$$\textsc{Com}_1\ \frac{p \xrightarrow{\mu} p'}{p|q \xrightarrow{\mu} p'|q}$$

$$\textsc{Com}_2\ \frac{q \xrightarrow{\mu} q'}{p|q \xrightarrow{\mu} p|q'}$$

$$\textsc{Com}_3\ \frac{p \xrightarrow{\lambda} p' \qquad q \xrightarrow{\overline{\lambda}} q'}{p|q \xrightarrow{\tau} p'|q'}$$

Worked-out examples

Communication, top-down

$$\textsc{Com}_1 \; \frac{p \xrightarrow{\mu} p'}{p|q \xrightarrow{\mu} p'|q} \quad \rightsquigarrow \quad (p \xrightarrow{\mu} p') \to (p|q \xrightarrow{\mu} p'|q)$$

$$\rightsquigarrow \quad \frac{p|q \xrightarrow{\mu} p'|q, p \xrightarrow{\mu} p', \Gamma \Rightarrow \Delta}{p \xrightarrow{\mu} p', \Gamma \Rightarrow \Delta} \; \textsc{Com}_{\circ 1}$$

Worked-out examples

Communication, top-down

IMT SCHOOL FOR ADVANCED STUDIES LUCCA

$$\textsc{Com}_3\ \frac{p \xrightarrow{\lambda} p' \qquad q \xrightarrow{\overline{\lambda}} q'}{p|q \xrightarrow{\tau} p'|q'} \quad \rightsquigarrow \quad (p \xrightarrow{\lambda} p') \wedge (q \xrightarrow{\overline{\lambda}} q') \to (p|q \xrightarrow{\tau} p'|q')$$

$$\rightsquigarrow \quad \frac{p|q \xrightarrow{\tau} p'|q', p \xrightarrow{\lambda} p', q \xrightarrow{\overline{\lambda}} q', \Gamma \Rightarrow \Delta}{p \xrightarrow{\lambda} p', q \xrightarrow{\overline{\lambda}} q', \Gamma \Rightarrow \Delta}\ \textsc{Com}_{\circ 3}$$

Worked-out examples

Communication, bottom-up

$$\textsc{Com}_1 + \textsc{Com}_2 + \textsc{Com}_3 \quad \rightsquigarrow \quad (p|q \xrightarrow{\mu} z) \to (\exists x, p \xrightarrow{\mu} x \wedge z \equiv x|q) \vee (\exists y, q \xrightarrow{\mu} y \wedge z \equiv p|y)$$

$$\& \quad \begin{array}{lll} (p|q \xrightarrow{\tau} z) & \to & (\exists x, p \xrightarrow{\tau} x \wedge z \equiv x|q) \vee (\exists y, q \xrightarrow{\tau} y \wedge z \equiv p|y) \\ & & \vee(\exists x \exists y, p \xrightarrow{\lambda} x \wedge q \xrightarrow{\overline{\lambda}} y \wedge z \equiv x|y) \end{array}$$

$$\rightsquigarrow \quad \frac{x|q \equiv z, p \xrightarrow{\mu} x, p|q \xrightarrow{\mu} z, \Gamma \Rightarrow \Delta \qquad p|y \equiv z, q \xrightarrow{\mu} y, p|q \xrightarrow{\mu} z, \Gamma \Rightarrow \Delta}{p|q \xrightarrow{\mu} z, \Gamma \Rightarrow \Delta}\ \textsc{Com}_{\circ\circ 1(!x,!y)}$$

$$\& \quad \frac{x|q \equiv z, p \xrightarrow{\tau} x, p|q \xrightarrow{\tau} z, \Gamma \Rightarrow \Delta \qquad p|y \equiv z, q \xrightarrow{\tau} y, p|q \xrightarrow{\tau} z, \Gamma \Rightarrow \Delta \qquad x|y \equiv z, p \xrightarrow{\lambda} x, q \xrightarrow{\overline{\lambda}} y, p|q \xrightarrow{\tau} z, \Gamma \Rightarrow \Delta}{p|q \xrightarrow{\tau} z, \Gamma \Rightarrow \Delta}\ \textsc{Com}_{\circ\circ 2(!x,!y)}$$

Terminating proof-search

GSOS operators

Working example

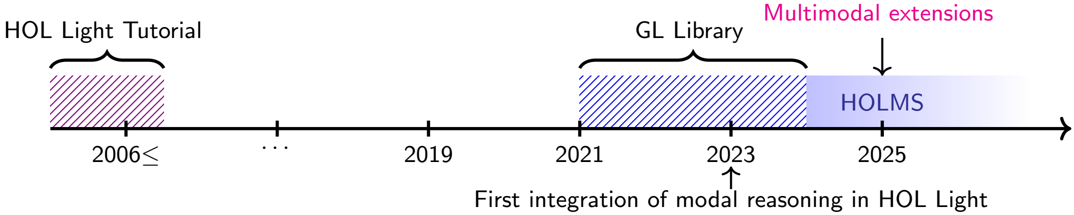
$$\frac{x \xrightarrow{\mu} x' \qquad x \overset{\nu}{\not\to} \quad (\text{for all } \nu > \mu)}{\theta(x) \xrightarrow{\mu} \theta(x')}$$

Non-working example

$$\frac{}{f(x) \xrightarrow{\mu} x} \qquad \frac{x \xrightarrow{\mu} y}{f(x) \xrightarrow{\mu} f(x')} \qquad \frac{}{c \xrightarrow{\mu} 0} \qquad \frac{}{c \xrightarrow{\mu} f(c)}$$

HOLMS Library

Mechanised modal reasoning in HOL Light



Underlying methodology

Axiomatic calculus $\Longleftrightarrow$ Frame class characterising modal schemas

Decision procedure $\leftarrow\cdots$ Labelled sequent calculus

.... shallow embedding === formalised adequacy theorems

Parametric Ad-hoc Polimorphic