Reversing Higher-Order Pi

Ivan Lanese [1] **Claudio Antares Mezzina** [2]
Jean-Bernard Stefani [2]

Focus Team, University of Bologna/INRIA, Italy

INRIA Grenoble-Rhône-Alpes, France

01-09-2010

Concur 2010.

Roadmap

1. Motivations
2. The $\rho\pi$ calculus
3. Encoding $\rho\pi$ in $HO\pi^+$
4. Future work

Roadmap

1. Motivations
2. The $\rho\pi$ calculus
3. Encoding $\rho\pi$ in $HO\pi^+$
4. Future work

Motivations

1. Study reversibility as basis for programming languages with comprehensive failure handling, e.g.:
   - checkpointing
   - transactions

2. Extend Danos and Krivine approach to π and higher order languages.

RCCS

Extension of the CCS LTS in order to support reversibility.

Main ideas

1. processes univocally identified by tags
2. tags are stacks carrying information needed to reverse actions
3. LTS semantics, with labels containing tags

RCCS rules

Extract of semantic rules:

$$m \triangleright \alpha.P + Q \xrightarrow{m:\alpha} \langle *, \alpha, Q\rangle \cdot m \triangleright P \quad act$$

$$\langle *, \alpha, Q\rangle \cdot m \triangleright P \xrightarrow{m:\alpha_*} m \triangleright \alpha.P + Q \quad act_*$$

Extract of structural laws:

$$m \triangleright (P \mid Q) \equiv \langle 1\rangle \cdot m : P \mid \langle 2\rangle \cdot m : Q$$

RCCS rules

Extract of semantic rules:

$$m \triangleright \alpha.P + Q \xrightarrow{m:\alpha} \langle *, \alpha, Q \rangle \cdot m \triangleright P \quad act$$

$$\langle *, \alpha, Q \rangle \cdot m \triangleright P \xrightarrow{m:\alpha_*} m \triangleright \alpha.P + Q \quad act_*$$

Extract of structural laws:

$$m \triangleright (P \mid Q) \equiv \langle 1 \rangle \cdot m : P \mid \langle 2 \rangle \cdot m : Q$$

not compatible with standard structural congruence.

RCCS results

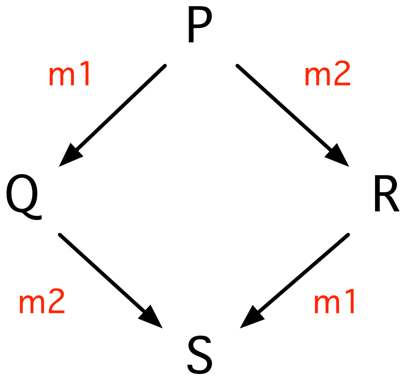
Results

- Rollback is causally consistent.
- Reversible actions as basis for transactions.

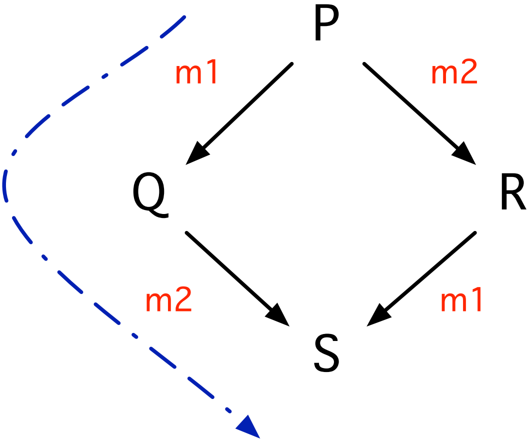
Causally Consistent

States reached during a backward computation could have been reached during the computation history by just performing independent actions in a different order.

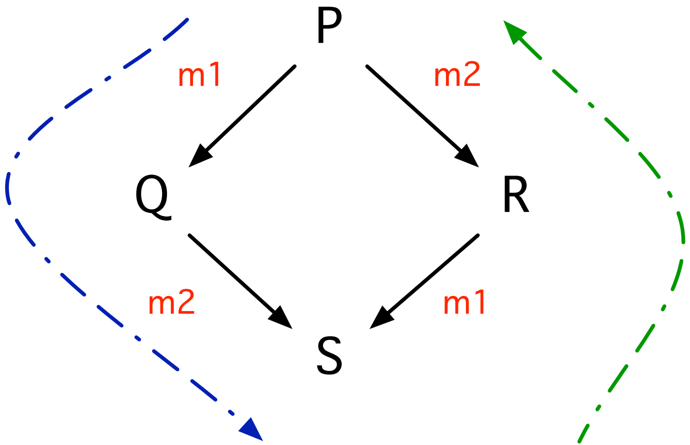
Causal Consistency: a simple example



Causal Consistency: a simple example



Causal Consistency: a simple example



Other related works

Phillips et al. devised an alternative way to reverse calculi in (subset of) *path* format:

- making all operators static
- recording past behavior and discarded alternatives in the syntax

Our work compared to RCCS

1. Extend reversible actions to $HO\pi$.
2. Preserve the standard structural laws of $HO\pi$.
3. Study the expressive power of reversible actions:
   - can reversibility be encoded in HOπ?

Roadmap

Key ideas of $\rho\pi$

- Processes identified by tags
  - we rely on name restriction to ensure the uniqueness of tags.
- Information necessary to roll back is stored in dedicated processes, called *memories*.

In contrast to Danos & Krivine, we use a **reduction semantics** approach.

Syntax

processes $$P, Q ::= \mathbf{0} \mid X \mid \nu a.P \mid (P \mid Q) \mid a\langle P\rangle \mid a(X) \triangleright P$$

Syntax

processes $P, Q ::= \mathbf{0} \mid X \mid \nu a.P \mid (P \mid Q) \mid a\langle P\rangle \mid a(X) \triangleright P$

configurations
$$M, N ::= \mathbf{0} \mid \kappa : P \mid [\mu; k] \mid \nu u.M \mid (M \mid N)$$
$$\kappa ::= k \mid \langle h, \tilde{h}\rangle \cdot k$$
$$\mu ::= ((\kappa_1 : a\langle P\rangle) \mid (\kappa_2 : a(X) \triangleright Q))$$

Syntax

$$\text{processes} \quad P, Q ::= \mathbf{0} \mid X \mid \nu a.P \mid (P \mid Q) \mid a\langle P\rangle \mid a(X) \triangleright P$$

$$\begin{aligned}\text{configurations} \quad & M, N ::= \mathbf{0} \mid \kappa : P \mid [\mu; k] \mid \nu u.M \mid (M \mid N) \\ & \kappa ::= k \mid \langle h, \tilde{h}\rangle \cdot k \\ & \mu ::= ((\kappa_1 : a\langle P\rangle) \mid (\kappa_2 : a(X) \triangleright Q))\end{aligned}$$

Processes of the form $[\mu; k]$ are memories:

- μ is the **past** configuration;
- k is the **tag** of the continuation process.

Structural laws

To the standard π structural laws we add the following one:

$$k : \prod_{i=1}^{n} \tau_i \equiv \nu \tilde{h} \prod_{i=1}^{n} (\langle h_i, \tilde{h} \rangle \cdot k : \tau_i) \quad \tilde{h} = \{h_1, \ldots, h_n\}$$

in which τ_i are *threads*, i.e. **messages** or **input processes**.

This rule is compatible with **abelian monoid** laws for parallel and $\mathbf{0}$, and it provides unique tags for identifying threads

Reduction semantics

$$\text{Fw} \quad \frac{M = [\kappa_1 : a\langle P\rangle \mid \kappa_2 : a(X) \triangleright Q \; ; \; k]}{\kappa_1 : a\langle P\rangle \mid \kappa_2 : a(X) \triangleright Q \twoheadrightarrow \nu k.(k : Q\{^P/_X\} \mid M)}$$

$$\text{Bw} \qquad k : P \mid [M \; ; \; k] \rightsquigarrow M$$

$\rightarrow = \twoheadrightarrow \cup \rightsquigarrow$ is closed under structural congruence and contexts

Causal consistency

We adapted the RCCS proof strategy about causal consistency to $\rho\pi$:

- $M \twoheadrightarrow N$ iff $N \rightsquigarrow M$
- two traces that are co-initial and co-final are **causally** equivalent.

Roadmap

Encoding: $HO\pi^+$

We encoded $\rho\pi$ into $HO\pi^+$, a variant of $HO\pi$ with:

- binary **join patterns**;
- **sub-addressing** $(a(X, \backslash h) \triangleright P)$;
- **abstractions** over names $((l)P)$ and process variables $((X)P)$;
- **applications** $(P V)$.

The encoding $(\!|\_|\!)$ is a **function** $\mathcal{C}_{\rho\pi} \to \mathcal{P}_{HO\pi^+}$

Encoding: Ideas

- Each process comes with a process in charge of aborting it.
- A process tag is encoded as a special signaling channel.
- The encoding of a process is an abstraction over a signaling channel.

A configuration and its translation are weakly barbed bisimilar

$$M \overset{\cdot}{\approx} (\!| M |\!)$$

Encoding of a message

$$(\!|a\langle P\rangle|\!) = (l)(\texttt{Msg}\ a\ (\!|P|\!)\ l)$$
$$\texttt{Msg} = (a\ X\ l)a\langle X, l\rangle \mid (\texttt{KillM}\ a\ X\ l)$$
$$\texttt{KillM} = (a\ X\ l)(a(X, \backslash l) \triangleright l\langle (h)\texttt{Msg}\ a\ X\ h\rangle \mid \texttt{Rew}\ l)$$
$$\texttt{Rew} = (l)l(Z) \triangleright Zl$$

- a message is encoded in a message, carrying a pair, and a killer process in parallel;
- `KillM` consumes the message and signals on the abstracted channel that the abort is done;
- `Rew` is used to **abort** the decision of rollbacking.

Encoding of an input process

$$(\!|a(X) \triangleright P|\!) = (l)(\mathtt{Trig}_{(\!|P|\!)}\ a\ l)$$

$$\mathtt{Trig}_{(\!|P|\!)} = (a\ l)\nu\mathtt{t}.\mathtt{t} \mid (a(X,h) \mid \mathtt{t} \triangleright \nu k.(\!|P|\!)\, k \mid (\mathtt{Mem}_{(\!|P|\!)}\ a\ X\ h\ k\ l)) \mid$$
$$(\mathtt{KillT}_{(\!|P|\!)}\ \mathtt{t}\ l\ a)$$

$$\mathtt{KillT}_{(\!|P|\!)} = (\mathtt{t}\ l\ a)(\mathtt{t} \triangleright l\langle (h)\mathtt{Trig}_{(\!|P|\!)}\, a\, h\rangle \mid \mathtt{Rew}\, l)$$

$$\mathtt{Mem}_{(\!|P|\!)} = (a\ X\ h\ k\ l)k(Z) \triangleright (\mathtt{Msg}\ a\ X\ h) \mid (\mathtt{Trig}_{(\!|P|\!)}\ a\ l)$$

- `KillT` aborts the input process by acquiring the lock `t`;
- the memory process awaits the aborting of the spawned process.

Roadmap

1. Motivations
2. The $\rho\pi$ calculus
3. Encoding $\rho\pi$ in $HO\pi^{+}$
4. Future work

Future work

- Further look to the expressiveness of reversible actions:
  - full abstraction;
  - absence of divergence;
  - minimizing garbage.
- Use our approach to obtain Phillips-like results for more general SOS format.
- Semantics and bisimulation for reversible calculi.