Concurrency, Causality and Reversibility: part 1

Irek Ulidowski

University of Leicester

April 2014

Joint work with Iain Phillips (Imperial College London)

Overview

Introduction

Usually computation goes forward only $P \rightarrow Q$
But sometimes reversible computation $P \leftrightarrow Q$ can be helpful:

Landauer (1961): irreversibility generates heat; logical (ir)reversibility.

Danos and Krivine (2003-2005): Reversible CCS; biological systems.

Glück and Yokoyama (2007): reversible programming language.

Phillips, Ulidowski, Yuen (2006-now): reversibility in concurrency.

Lanese, Krivine, Stefani: reversible π and higher-order π.

Laneve and Cardelli: reversible asynchronous process calculus.

Reversibility

Reversibility is very common in physics and biochemistry.
In nature reversibility underpins many mechanisms for achieving progress or change.

- e.g. building polymers, signal passing, catalysis

In artificial systems reversibility has a growing number of applications:

- saving energy
- debugging
- recovery from failure
  e.g. long-running transactions with compensations

Reversing CCS

RCCS: Danos and Krivine introduce separate memories to keep track of past behaviour and the discarded alternatives.

CCSK (CCS with keys): Phillips and Ulidowski (2006-7):

- Past behaviour is recorded in the syntax of terms and not on external devices: convert standard operators to static versions
- Predicates are used to control execution of converted terms
- One-time keys allow to distinguish individual action occurrences
- Symmetry between forward and reverse SOS rules

Reversing dynamic operators

Action prefixing: standard SOS rule

$$\frac{}{a.X \xrightarrow{a} X}$$

We use predicates and past actions $\underline{a}$ in the forward SOS rules:

$$\frac{\mathrm{std}(X)}{a.X \xrightarrow{a} \underline{a}.X} \qquad \frac{X \xrightarrow{b} X'}{\underline{a}.X \xrightarrow{b} \underline{a}.X'} \text{ for all } b$$

where $\mathrm{std}(X)$ means that X contains no occurrences of past actions.

Symmetry gives the reverse rules:

$$\frac{\mathrm{std}(X)}{\underline{a}.X \overset{a}{\rightsquigarrow} a.X} \qquad \frac{X \overset{b}{\rightsquigarrow} X'}{\underline{a}.X \overset{b}{\rightsquigarrow} \underline{a}.X'} \text{ for all } b$$

CCS choice: $\dfrac{X \xrightarrow{a}_{S} X'}{X + Y \xrightarrow{a}_{S} X'} \quad \dfrac{Y \xrightarrow{a}_{S} Y'}{X + Y \xrightarrow{a}_{S} Y'}$ for all actions a.

Again use predicates:

$$\frac{X \xrightarrow{a} X' \quad \mathrm{std}(Y)}{X + Y \xrightarrow{a} X' + Y} \qquad \frac{Y \xrightarrow{a} Y' \quad \mathrm{std}(X)}{X + Y \xrightarrow{a} X + Y'}$$

and symmetry for reverse rules:

$$\frac{X \overset{a}{\rightsquigarrow} X' \quad \mathrm{std}(Y)}{X + Y \overset{a}{\rightsquigarrow} X' + Y} \qquad \frac{Y \overset{a}{\rightsquigarrow} Y' \quad \mathrm{std}(X)}{X + Y \overset{a}{\rightsquigarrow} X + Y'}$$

Prefixing and choice are now static.

Occurrences of actions and Keys

The approach so far works for many standard operators.
However sometimes it is necessary to have more control over event occurrences:

- semantic anomalies related to auto-concurrency
- ambiguities connected with reversing CCS communication

We use one-time communication keys.

Keys and Auto-concurrency

The approach so far equates $a.a$ and $a \,|\, a$.

Our solution: Dynamically choose a fresh key $(m, n, \ldots)$ when performing action. Then we can distinguish $a \,|\, a$ from $a.a$:

$$a \,|\, a \xrightarrow{a[m]} a[m] \,|\, a \xrightarrow{a[n]} a[m] \,|\, a[n] \overset{a[m]}{\rightsquigarrow} a \,|\, a[n] \qquad (m \neq n)$$

By contrast

$$a.a \xrightarrow{a[m]} a[m].a \xrightarrow{a[n]} a[m].a[n] \overset{a[m]}{\not\rightsquigarrow} \qquad (m \neq n)$$

Communication and keys

Without keys we may have strange behaviour:

$$a \,|\, \overline{a} \xrightarrow{\tau} \underline{a} \,|\, \underline{\overline{a}} \overset{a}{\rightsquigarrow} a \,|\, \underline{\overline{a}}$$

Desirable to link occurrences of a and $\overline{a}$ together.

Our solution: use keys to lock the two partners in a communication

$$a \,|\, \overline{a} \xrightarrow{\tau} a[m] \,|\, \overline{a}[m]$$

The partners agree on the key m. Can only reverse together
Can also proceed separately; can reverse in either order, but not together.

$$a \,|\, \overline{a} \xrightarrow{a[m]} a[m] \,|\, \overline{a} \xrightarrow{\overline{a}[n]} a[m] \,|\, \overline{a}[n] \qquad (m \neq n)$$

SOS rules with keys

Action labels are now of the form $a[m]$.
For each rule r add predicates $\mathsf{fsh}[m](X_i)$ to the premises of r for each static i that does not already appear in the premises of r.

CCS prefixing with keys: past actions $\underline{a}$ are now $a[m]$:

$$\frac{\mathsf{std}(X)}{a.X \xrightarrow{a[m]} a[m].X} \qquad \frac{X \xrightarrow{b[n]} X'}{a[m].X \xrightarrow{b[n]} a[m].X'}\ m \neq n$$

The reformulated CCS parallel:

$$\frac{X \xrightarrow{a[m]} X' \quad \mathsf{fsh}[m](Y)}{X \,|\, Y \xrightarrow{a[m]} X' \,|\, Y} \qquad \frac{Y \xrightarrow{a[m]} Y' \quad \mathsf{fsh}[m](X)}{X \,|\, Y \xrightarrow{a[m]} X \,|\, Y'} \qquad \frac{X \xrightarrow{a[m]} X' \quad Y \xrightarrow{\overline{a}[m]} Y'}{X \,|\, Y \xrightarrow{\tau[m]} X' \,|\, Y'}$$

SOS rules for CCSK

$$\frac{\mathsf{std}(X)}{a.X \xrightarrow{a[m]} a[m].X} \qquad \frac{X \xrightarrow{b[n]} X'}{a[m].X \xrightarrow{b[n]} a[m].X'}\ m \neq n$$

$$\frac{X \xrightarrow{a[m]} X' \quad \mathsf{std}(Y)}{X + Y \xrightarrow{a[m]} X' + Y} \qquad \frac{Y \xrightarrow{a[m]} Y' \quad \mathsf{std}(X)}{X + Y \xrightarrow{a[m]} X + Y'}$$

$$\frac{X \xrightarrow{a[m]} X' \quad \mathsf{fsh}[m](Y)}{X \,|\, Y \xrightarrow{a[m]} X' \,|\, Y} \qquad \frac{Y \xrightarrow{a[m]} Y' \quad \mathsf{fsh}[m](X)}{X \,|\, Y \xrightarrow{a[m]} X \,|\, Y'} \qquad \frac{X \xrightarrow{a[m]} X' \quad Y \xrightarrow{\overline{a}[m]} Y'}{X \,|\, Y \xrightarrow{\tau[m]} X' \,|\, Y'}$$

$$\frac{X \xrightarrow{a[m]} X'}{X \setminus A \xrightarrow{a[m]} X' \setminus A}\ a \notin A \qquad \frac{X \xrightarrow{a[m]} X'}{X[f] \xrightarrow{f(a)[m]} X'[f]}$$

Transition systems for reversible calculi

Prime graphs: Enjoy several natural properties, eg event determinism, reverse diamond and forward diamond properties.

Expressive enough to define true concurrency notions such as

- causality
- concurrency
- conflict

What True Concurrency models of computation could be useful? Configurations ...

Configuration structures

A more general model of processes: stable configuration structures. These are $\mathcal{C} = (C, \ell)$ where

- C is a set of configurations
- ℓ is a labelling function on events

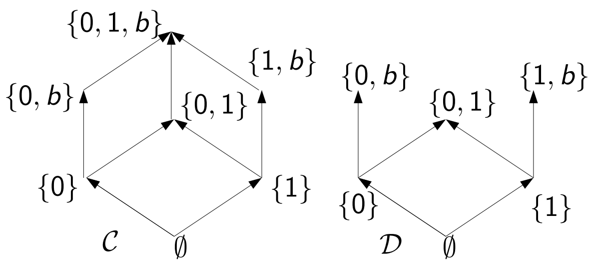
A configuration is a set of events: those that have occurred so far.

Various axioms, including:

- closed under bounded unions:
  if $X, Y, Z \in C$ then $X \cup Y \subseteq Z$ implies $X \cup Y \in C$;
- closed under bounded intersections:
  if $X, Y, Z \in C$ then $X \cup Y \subseteq Z$ implies $X \cap Y \in C$.

Winskel's parallel switch

Bulb b can be lit by connecting switch 0 or switch 1.



$\mathcal{C}$ is not stable: inclusive or causation (violates bounded intersection)
$\mathcal{D}$ is stable: exclusive or causation

Ordering and labels

Each configuration X has a causal ordering $\leq_X$ defined by

> *$d \leq_X e$ iff for all configurations Y with $Y \subseteq X$ we have $e \in Y$ implies $d \in Y$.*

Furthermore $d <_X e$ iff $d \leq_X e$ and $d \neq e$.

Very roughly, d is one of the causes of e, and e is one of the effects of d.

We use a labelling function ℓ to give each event a label $(a, b, \ldots)$. Labels are observable; events are not.

Transitions

Definition

We let $X \xrightarrow{a}_{\mathcal{C}} X'$ iff $X, X' \in C$ and $X' \setminus X = \{e\}$ with $\ell(e) = a$.

Allows us to define bisimulations, etc. And, $X \overset{a}{\rightsquigarrow}_{\mathcal{C}} Y$ if $Y \xrightarrow{a}_{\mathcal{C}} X$.

Example (Parallel Switch)

$$\ell(0) = \text{on}, \ell(1) = \text{on}, \ell(b) = \text{bulb}$$

$$\emptyset \xrightarrow{\text{on}}_{\mathcal{D}} \{0\} \xrightarrow{\text{bulb}}_{\mathcal{D}} \{0, b\}$$

Forward-only equivalences

(van Glabbeek & Goltz, 2001)
Various bisimulation-based equivalences, forward-only.

- interleaving bisimulation $\approx_{ib}$

Expand observations:
A step is a multiset of concurrent labelled events ($\{a, a, b\}$).

- step bisimulation $\approx_{sb}$

A pomset is a multiset of partially ordered labelled events ($a < b < a$).

- pomset bisimulation $\approx_{pb}$

Use order isomorphisms between configurations:

- weak history-preserving bisimulation $\approx_{wh}$
- history-preserving bisimulation $\approx_{h}$

Bisimulation

Let $\mathcal{C}, \mathcal{D} \in \mathbb{C}_{stable}$. A relation $R \subseteq C_{\mathcal{C}} \times C_{\mathcal{D}}$ is an *interleaving bisimulation* (ib) between $\mathcal{C}$ and $\mathcal{D}$ if

1. $(\emptyset, \emptyset) \in R$,
2. if $(X, Y) \in R$ then for $a \in \text{Act}$
   - if $X \xrightarrow{a}_{\mathcal{C}} X'$ then $\exists Y'.\ Y \xrightarrow{a}_{\mathcal{D}} Y'$ and $(X', Y') \in R$;
   - if $Y \xrightarrow{a}_{\mathcal{D}} Y'$ then $\exists X'.\ X \xrightarrow{a}_{\mathcal{C}} X'$ and $(X', Y') \in R$.

$\mathcal{C}$ and $\mathcal{D}$ are ib equivalent ($\mathcal{C} \approx_{ib} \mathcal{D}$) iff there is an ib between $\mathcal{C}$ and $\mathcal{D}$.

If we replace actions a by steps or pomsets we obtain $\approx_{sb}$ or $\approx_{pb}$.

If additionally there is an order preserving isomorphism between X, Y we get $\approx_{wh}$ and $\approx_{h}$.

Hierarchy of forward-only equivalences

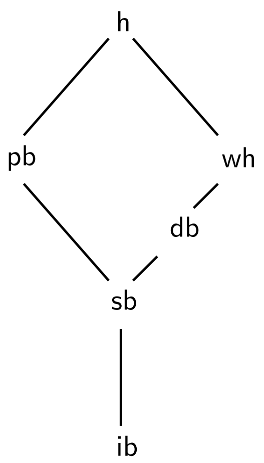
Theorem

$\approx_{wh} \subsetneq \approx_{db} \subsetneq \approx_{sb}$

Example

$$a \,|\, b \approx_{sb} (a \,|\, b) + a.b$$

$$a \,|\, b \not\approx_{db} (a \,|\, b) + a.b$$



Hereditary History-preserving bisimulation

Hereditary history-preserving bisimulation $\approx_{hh}$ (Bednarczyk 1991)

- has reversibility in definition (hereditary property)
- based on order isomorphisms between sets of events - not really observations

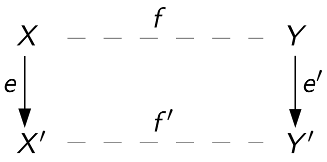
Canonical equivalence on event structures:
characterisation as open map bisimulation with labelled partial orders as the observations (Joyal, Nielsen, Winskel 1996).

We would like to characterise $\approx_{hh}$ using observations. Is it possible?

History-preserving bisimulations

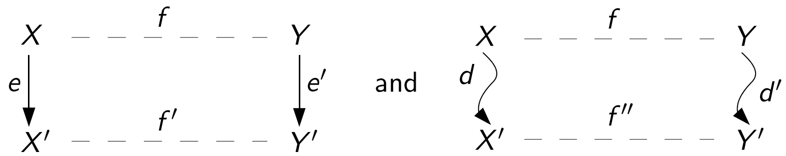
Let $\mathcal{C}, \mathcal{D} \in \mathbb{C}_{stable}$ and let the sets of configurations be $C_{\mathcal{C}}, C_{\mathcal{D}}$.

Consider a relation $\mathcal{R} \subseteq C_{\mathcal{C}} \times C_{\mathcal{D}} \times \mathcal{P}(E_{\mathcal{C}} \times E_{\mathcal{D}})$ such that $\mathcal{R}(\emptyset, \emptyset, \emptyset)$ and if $\mathcal{R}(X, Y, f)$ then



- if f and f' are isomorphisms, then $\mathcal{R}$ is a weak history-preserving (WH) bisimulation.
- if additionally $f' \upharpoonright X = f$, then $\mathcal{R}$ is a history-preserving (H) bisimulation.

- if f, f' and f'' are isomorphisms and $f \upharpoonright X' = f''$, where



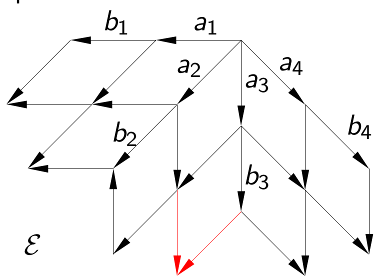
  then $\mathcal{R}$ is a hereditary weak history-preserving (HWH) bisimulation.
- if additionally $f' \upharpoonright X = f$, then $\mathcal{R}$ is a hereditary history-preserving (HH) bisimulation (Bednarczyk 1991).

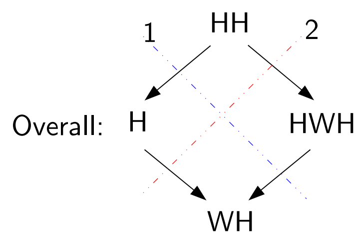
Hierarchy of equivalences

1. The Absorption Law holds only for H and WH

$$(a \,|\, (b + c)) + (a \,|\, b) + ((a + c) \,|\, b) = (a \,|\, (b + c)) + ((a + c) \,|\, b)$$

2. $\mathcal{E}$ below and $\mathcal{F}$ ($\mathcal{E}$ without red transitions) are only WH and HWH equivalent.





Reverse bisimulation

Start with reverse interleaving bisimulation ($\approx_{ri-ib}$).
Match on labels, with reverse as well as forward transitions: if $(X, Y) \in R$

- if $X \xrightarrow{a}_{\mathcal{C}} X'$ then $\exists Y'.\ Y \xrightarrow{a}_{\mathcal{D}} Y'$ and $(X', Y') \in R$;
- if $Y \xrightarrow{a}_{\mathcal{D}} Y'$ then $\exists X'.\ X \xrightarrow{a}_{\mathcal{C}} X'$ and $(X', Y') \in R$.
- if $X \overset{a}{\rightsquigarrow}_{\mathcal{C}} X'$ then $\exists Y'.\ Y \overset{a}{\rightsquigarrow}_{\mathcal{D}} Y'$ and $(X', Y') \in R$;
- if $Y \overset{a}{\rightsquigarrow}_{\mathcal{D}} Y'$ then $\exists X'.\ X \overset{a}{\rightsquigarrow}_{\mathcal{C}} X'$ and $(X', Y') \in R$.

We already saw that $a \mid b \not\approx_{ri-ib} a.b + b.a$.

The Absorption Law

$$(a \mid (b + c)) + (a \mid b) + ((a + c) \mid b) = (a \mid (b + c)) + ((a + c) \mid b)$$

holds for $\approx_h$, but not for $\approx_{ri-ib}$.

Auto-concurrency

Reverse bisimulation is insensitive to auto-concurrency: $a \mid a \approx_{ri-ib} a.a$

Certainly $\approx_{hh} \subsetneq \approx_{ri-ib}$.

Theorem (Bednarczyk 1991—prime event structures)

If no auto-concurrency then $\approx_{ri-ib} = \approx_{hh}$.

We improved this in the SOS 2009 paper to:

Theorem (stable configuration structures)

If no equidepth auto-concurrency then $\approx_{ri-ib} = \approx_{hh}$.

To cope with auto-concurrency, need to enhance observations.

- steps
- pomsets
- depth

Reverse Step bisimulation

Reverse step bisimulation ($\approx_{rs-sb}$) defined as (forward-only) step bisimulation, with matching on reverse steps as well: if $(X, Y) \in R$

- if $X \overset{A}{\rightsquigarrow}_{\mathcal{C}} X'$ then $\exists Y'.\ Y \overset{A}{\rightsquigarrow}_{\mathcal{D}} Y'$ and $(X', Y') \in R$;
- if $Y \overset{A}{\rightsquigarrow}_{\mathcal{D}} Y'$ then $\exists X'.\ X \overset{A}{\rightsquigarrow}_{\mathcal{C}} X'$ and $(X', Y') \in R$.

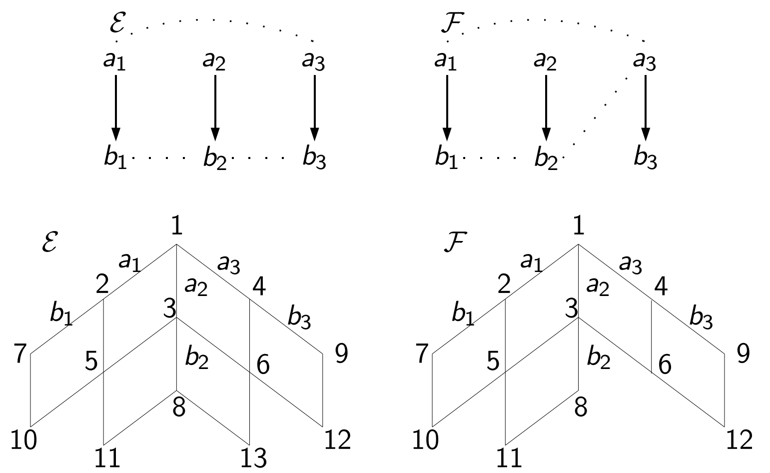
We have $\approx_{hh} \subseteq \approx_{rs-sb} \subsetneq \approx_{ri-ib}$.

Open Question (Bednarczyk 1991)

Does $\approx_{rs-sb} = \approx_{hh}$?

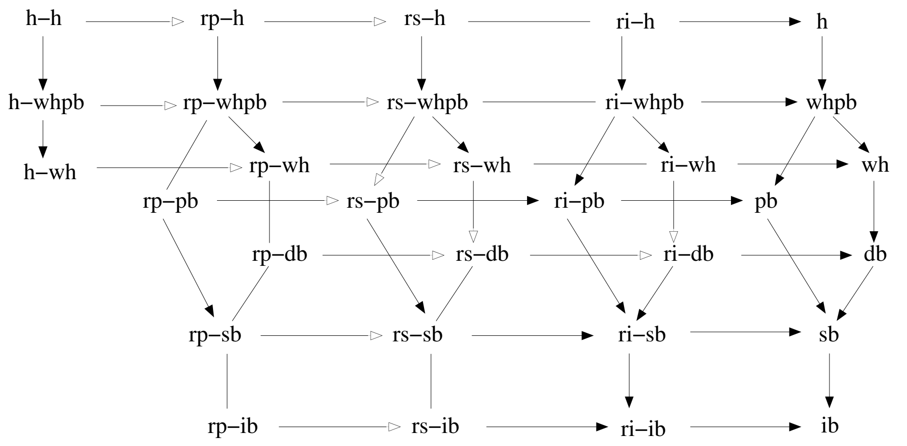
Counterexample

We discovered that $\approx_{rs-sb} \neq \approx_{hh}$.



A Hierarchy of equivalences

We made a detailed study bisimulations rX-Yb that combine reverse observations X with forward observations Y. For example ri-ib, and rp-ib, rp-pb, rp-h, ...



Logics

- We present modal logics which describe how processes can perform both forward and reverse transitions
- These logics correspond to true-concurrency equivalences on stable configuration structures.

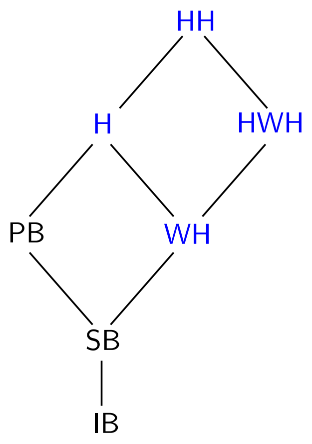
Motivation

Interleaving bisimulation (IB) and Hennessy-Milner logic (HML) equate $a \mid b$ and $a.b + b.a$ but true-concurrency bisimulations distinguish them.

We aim to extend HML so that it can characterise true-concurrency bisimulations.

Reverse modalities are useful: $a|b$ satisfies $\langle a\rangle\rangle\langle b\rangle\rangle\langle\langle a\rangle \mathtt{tt}$, while $a.b + b.a$ does not. They are not sufficient, especially in the presence of auto-concurrency. $\langle a\rangle\rangle\langle a\rangle\rangle\langle\langle a\rangle \mathtt{tt}$ is satisfied by both $a|a$ and $a.a$.

More complex modalities (both forward and reverse) capture some equivalences up to history-preserving bisimulation (H) but not beyond.



Our solution

Keep track of the identities of events as they execute.

When we perform an event we declare an identifier $(x, y, \ldots)$ for that event, allowing us to refer to it again when reversing it. Now we can write $\langle x : a\rangle\rangle\langle y : a\rangle\rangle\langle\langle x\rangle \mathtt{tt}$ to say that we reverse the first a, and this is satisfied by $a|a$, but not by $a.a$.

$\Longrightarrow$ can characterise H and HH.

Also, we add declarations $(x : a)\phi$. We can now express $\langle\langle a\rangle\phi$ by the formula $(x : a)\langle\langle x\rangle\phi$ (where x does not occur (free) in ϕ).

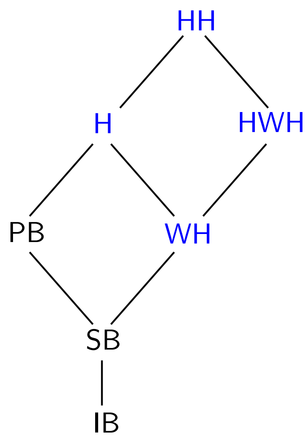
$\Longrightarrow$ can characterise WH and HWH.

Related work

Many papers on logics with reverse modalities. Only backtracking allowed. The satisfaction relations defined over computations (runs).

Nielsen & Clausen 1994: reverse event index modality, reversing allowed. Characterisation of HH stated.

Baldan & Crafa 2010: event identifiers, complex forward-only modalities, no reversing. Characterisation of SB, PB, H and HH.



Hennessy-Milner Logic

Action labels $a, b, \ldots$

$$\varphi ::= \mathrm{tt} \mid \mathrm{ff} \mid \neg\varphi \mid \phi_1 \wedge \varphi_2 \mid \phi_1 \vee \varphi_2 \mid \langle a\rangle\rangle\varphi \mid [a]]\,\varphi$$

We write diamond and box in a non-standard way, to emphasise that they are forward modalities.

Remark

ff, $\vee$, $[a]]$ can be derived. Alternatively, $\neg$ can be omitted.

Satisfaction relation: $P \models \langle a\rangle\rangle\varphi$ iff $\exists Q.\ P \xrightarrow{a} Q \models \varphi$

Theorem (Hennessy & Milner 1985)

HML characterises bisimulation (for image-finite labelled transition systems)

Forward-Reverse Logic

Let us add reverse modalities to get FRL:

$$\varphi ::= \mathrm{tt} \mid \mathrm{ff} \mid \neg\varphi \mid \phi_1 \wedge \varphi_2 \mid \phi_1 \vee \varphi_2 \mid \langle a\rangle\rangle\varphi \mid [a]]\,\varphi \mid \langle\langle a\rangle\varphi \mid [[a]\,\varphi$$

We can now make true-concurrency distinctions:

$$a\,|\,b \models \langle a\rangle\rangle\langle b\rangle\rangle\langle\langle a\rangle\mathrm{tt} \qquad a.b + b.a \not\models \langle a\rangle\rangle\langle b\rangle\rangle\langle\langle a\rangle\mathrm{tt}$$

Theorem

FRL characterises ri-ib bisimulation (for stable configuration structures).

We already saw that $a\,|\,b \not\approx_{ri-ib} a.b + b.a$. A formula is $\langle a\rangle\rangle\langle b\rangle\rangle\langle\langle a\rangle\mathrm{tt}$.

The Absorption Law

$$(a\,|\,(b+c)) + (a\,|\,b) + ((a+c)\,|\,b) = (a\,|\,(b+c)) + ((a+c)\,|\,b)$$

holds for $\approx_h$, but not for $\approx_{ri-ib}$. A formula is

$$\langle a\rangle\rangle(\,[c]]\,\mathrm{ff} \wedge \langle b\rangle\rangle\langle\langle a\rangle\ \ [c]]\,\mathrm{ff})$$

Step-Reverse Logic, Pomset-Reverse Logic

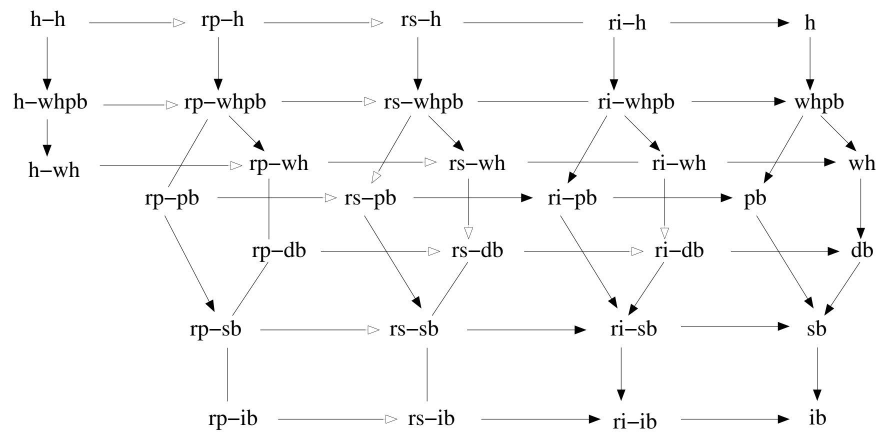
Step modalities: $\langle A\rangle\rangle\varphi$ and $\langle\langle A\rangle\varphi$. This gives us the logic SRL.
Pomset modalities $\langle p\rangle\rangle\varphi$ and $\langle\langle p\rangle\varphi$. This gives us the logic PRL.
Clearly SRL generalises FRL, and PRL generalises SRL.

$$a\,|\,a \models \langle\{a,a\}\rangle\rangle \mathrm{tt} \qquad a.a \not\models \langle\{a,a\}\rangle\rangle \mathrm{tt}$$

$$a\,|\,a \not\models \langle a < a\rangle\rangle \mathrm{tt} \qquad a.a \models \langle a < a\rangle\rangle \mathrm{tt}$$

Theorem

SRL characterises rs-sb bisimulation and PRL characterises rp-pb bisimulation (for stable configuration structures).



Event identifiers

If we want to capture HH bisimulation, we need to have control over which events are reversed. In the formula below we need to know whether the first or second event labelled with a is being reversed.

$$\langle a\rangle\rangle\langle a\rangle\rangle\langle\langle a\rangle \mathrm{tt}$$

However we do not want to talk about events directly, since that would not be abstract enough. So we use event identifiers:

$$\langle x : a\rangle\rangle\langle y : a\rangle\rangle\langle\langle x\rangle \mathrm{tt}$$

Here the event being reversed is the first a rather than the second a.

Example

$$a\,|\,a \models \langle x : a\rangle\rangle\langle y : a\rangle\rangle\langle\langle x\rangle \mathrm{tt} \qquad a.a \not\models \langle x : a\rangle\rangle\langle y : a\rangle\rangle\langle\langle x\rangle \mathrm{tt}$$

Event Identifier Logic

Assume an infinite set of identifiers x which can be bound to any events. Event Identifier Logic (EIL) is:

$$\phi ::= \mathrm{tt} \mid \neg\phi \mid \phi \wedge \phi' \mid \langle x : a\rangle\rangle\phi \mid (x : a)\phi \mid \langle\langle x\rangle\phi$$

We need to treat forward and backwards modalities differently:

- Going forward, x is bound to a new event that has not yet occurred.
- Reversing, x is interpreted as the event to which x is already bound.
- Once x is reversed, there is no further access to the binding for x (achieved via notion of permissible environment).

Thus, for example, x is bound in $\langle x : a\rangle\rangle\phi$.

Satisfaction

An environment ρ is a partial mapping from identifiers to events. We say that ρ is a permissible environment for ϕ and a configuration X if $\mathrm{fi}(\phi) \subseteq \mathrm{dom}(\rho)$ and $\mathrm{rge}(\rho \restriction \mathrm{fi}(\phi)) \subseteq X$.

$$X, \rho \models \langle x : a \rangle\rangle \phi \iff \exists e, Y.\ X \xrightarrow{e} Y \text{ with } \ell(e) = a \text{ and } Y, \rho[x \mapsto e] \models \phi$$

$$X, \rho \models \langle\langle x \rangle \phi \iff \exists e, Y.\ X \overset{e}{\rightsquigarrow} Y \text{ with } \rho(x) = e \text{ and } Y, \rho \models \phi \quad (\rho \text{ is permissible for } \phi \text{ and } Y)$$

Example

Consider $e_a < e'_a$. Is $\langle x : a \rangle\rangle \langle y : a \rangle\rangle \langle\langle y \rangle \langle\langle y \rangle \mathrm{tt}$ satisfied?
After performing e_a, e'_a and reversing e'_a we have
$\{e_a\}, [x \mapsto e_a,\ y \mapsto e'_a] \not\models \langle\langle y \rangle \mathrm{tt}$ since $\mathrm{rge}(\rho \restriction y) = \{e'_a\} \not\subseteq \{e_a\}$.

$$X, \rho \models (x : a)\phi \iff \exists e.\ \ell(e) = a \text{ and } \rho[x \mapsto e] \models \phi$$

Examples

If we are not careful with the handling of identifier bindings, we can make the logic too strong. Consider

$$\phi \overset{\mathrm{df}}{=} \langle x : a \rangle\rangle \langle y : b \rangle\rangle \langle\langle y \rangle \langle\langle x \rangle \langle z : a \rangle\rangle \neg \langle y : b \rangle\rangle \mathrm{tt}$$

If the three ys are all bound to the same event, we have

$$a.b + a.b \models \phi \qquad a.b \not\models \phi$$

This makes the logic more discriminating than HH bisimulation.
However, if we regard $\langle y : b \rangle\rangle$ as binding a fresh event, there is no problem, as

$$a.b + a.b \not\models \phi \qquad a.b \not\models \phi$$

More examples

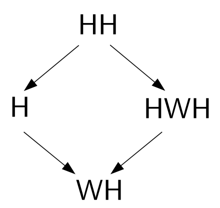
1. $\langle x : a \rangle\rangle \langle y : a \rangle\rangle \langle\langle x \rangle \mathrm{tt}$ is satisfied by $a \,|\, a$ but not by $a.a$.
2. $[x : a]]\ [y : a]]\ \langle\langle x \rangle \mathrm{tt}$ is satisfied only by $a \,|\, a$ but not by $(a \,|\, a) + a.a$.

EIL characterises HH bisimulation

For closed ϕ we define $\mathcal{C} \models \phi$ iff $\emptyset \models_{\mathcal{C}} \phi$.

Let $\mathcal{C}, \mathcal{D}$ be stable configuration structures.

Then $\mathcal{C}$ and $\mathcal{D}$ are HH eqt iff for all $\phi \in \mathrm{EIL}$ we have $\mathcal{C} \models \phi$ iff $\mathcal{D} \models \phi$:



Theorem

EIL *characterises HH bisimulation (for stable configuration structures).*

The proof would still work with the logic without declarations $(x : a)\phi$.

Next, we look for sublogics of EIL.

A logic for History Preserving bisimulation

Reverse-only logic $\mathrm{EIL_{ro}}$:

$$\phi ::= \mathrm{tt} \mid \neg\phi \mid \phi \wedge \phi' \mid (x : a)\phi \mid \langle\langle x \rangle \phi$$

This logic is preserved between isomorphic configurations, and characterises configurations up to isomorphism.

$\mathrm{EIL_h}$ (no forward after reverse logic) is given as follows, where ϕ_r is a formula of $\mathrm{EIL_{ro}}$:

$$\phi ::= \mathrm{tt} \mid \neg\phi \mid \phi \wedge \phi' \mid \langle x : a \rangle\rangle \phi \mid (x : a)\phi \mid \phi_r$$

Theorem

$\mathrm{EIL_h}$ *characterises H bisimulation (for stable configuration structures).*

A logic for Weak History Preserving bisimulation

We get from $\mathrm{EIL_h}$ to $\mathrm{EIL_{wh}}$ by simply requiring that all formulas of $\mathrm{EIL_{wh}}$ are *closed*, where ϕ_{rc} is a *closed* formula of $\mathrm{EIL_{ro}}$:

$$\phi ::= \mathrm{tt} \mid \neg\phi \mid \phi \wedge \phi' \mid \langle a \rangle\rangle \phi \mid \phi_{rc}$$

We write $\langle a \rangle\rangle \phi$ rather than $\langle x : a \rangle\rangle \phi$ since ϕ is closed and in particular x does not occur free in ϕ.

Also we omit declarations $(x : a)\phi$ since they have no effect when ϕ is closed.

Of course declarations can occur in ϕ_{rc}.

Theorem

$\mathrm{EIL_{wh}}$ *characterises WH bisimulation (for stable configuration structures).*

Similarly, for HWH.

Conclusions

- We have reversed CCS using keys, predicates and fixed term structure. Recently the π calculus was reversed with the help of keys.
- A wealth of reverse bisimulations. We have strengthened Bednarczyk's result as much as possible:
  in the absence of equidepth auto-concurrency, ri-ib is as strong as hh.
- Logics: extensions of HML and the new Event Identifier Logic. Simple logical characterisations of wh, h and hh bisimulations.

References

- Iain Phillips and Irek Ulidowski, Reversing Algebraic Process Calculi. In FOSSACS 2006. Springer, LNCS 3921.
- Iain Phillips and Irek Ulidowski, Reversing Algebraic Process Calculi. The Journal of Logic and Algebraic Programming, 73, pp 70-96, 2007.
- Iain Phillips and Irek Ulidowski, A Hierarchy of Reverse Bisimulations on Stable Configuration Structures. Mathematical Structures in Computer Science, vol 22, pp 333-372, 2012.
- Iain Phillips and Irek Ulidowski, An Event Identifier Logic. Mathematical Structures in Computer Science, vol 24. 2014