Equivalences and Logics for Reversible Processes

International Training School on Reversible Computation
Toruń, Poland

Iain Phillips, Imperial College London
Joint work with Irek Ulidowski, University of Leicester

30 August 2017

Thanks to Irek Ulidowski

Overview

Overview

Introduction

Semantics via Equivalences

The traditional approach to semantics was denotational:

- map a program to its denotation in some mathematical domain

Good for sequential programs.

Not so clear what this should be for concurrent processes, particularly ones which need not terminate.

Milner pioneered the use of equivalences, particularly bisimulation.

Instead of asking the meaning of a process, we ask whether two processes P and Q have the same observable behaviour.

- Are P and Q equivalent?

Very flexible approach.

Can compare within a single calculus or between different versions of calculi.

Labelled and Unlabelled Transitions

Unlabelled reductions

E.g. ρCCS

$$k : (a.P + P') \,|\, k' : (a.Q + Q') \twoheadrightarrow \nu h, h' \; h : P \,|\, h' : Q \,|\, [a, P', Q', k, k', h, h']$$

Labelled transitions

E.g. CCSk

$$\alpha.P \xrightarrow{\alpha[m]} \alpha[m].P$$

I. Lanese. Reversibility for concurrent interacting systems.

International Training School on Reversible Computation, 2017

Reduction and Barbed Congruence

Reduction Semantics

A brief look at reversible higher-order pi-calculus $\rho\pi$ as an example.

Convention for this section

- forward reduction $\twoheadrightarrow$
- reverse reduction $\rightsquigarrow$
- forward or reverse $\rightarrow = \twoheadrightarrow \cup \rightsquigarrow$

I. Lanese, C.A. Mezzina, and J.-B. Stefani. Reversibility in the higher-order π-calculus.

***Theoretical Computer Science*, 625:25–84, 2016**

Equivalence vs Congruence

An equivalence $P \sim Q$ is not necessarily a congruence.

To be a congruence we need that if $P \sim Q$ then for any context $\mathcal{C}[\bullet]$ we have $\mathcal{C}[P] \sim \mathcal{C}[Q]$.

- can replace P by Q considered as a module within a wider system

An equivalence $\sim$ can always be turned into a congruence by defining

$$P \sim_c Q \text{ iff for all contexts } \mathcal{C}[\bullet] \text{ we have } \mathcal{C}[P] \sim \mathcal{C}[Q]$$

Then

- $\sim_c$ is a congruence
- $\sim_c$ is the largest congruence included in $\sim$

Structural Congruence

Recall that $P \equiv Q$ means that P can be rearranged to get Q.

- $P + Q \equiv Q + P$
- $P|\mathbf{0} \equiv P$
- $\nu z\ (P|Q) \equiv P|\nu z\ Q$ if z not free in P
- etc.

We define $\equiv$ to be the smallest congruence satisfying laws such as the above.

In other words $P \equiv Q$ if we can deduce it from the above laws using equational reasoning.

Structural congruence is used to bring elements together so that they form a redex ready for a reduction step.

Structural congruence is fully reversible.

However no computation steps are involved.

Barbed Congruence

For reduction-based operational semantics.

Take reversible higher-order pi-calculus $\rho\pi$ as an example.

Identify the barbs (basic observations):

$$M \downarrow a \text{ iff } M \equiv \nu \vec{u}\ (k : a\langle P\rangle \mid N) \text{ with } a \notin \vec{u}$$

$\mathcal{R}$ is a weak barbed simulation if whenever $\mathcal{R}(M, N)$ then

- If $M \downarrow a$ then $N \rightarrow^* \downarrow a$
- If $M \rightarrow M'$ then $N \rightarrow^* N'$ with $\mathcal{R}(M', N')$

N simulates M

$\mathcal{R}$ is a weak barbed bisimulation if both $\mathcal{R}$ and $\mathcal{R}^{-1}$ are weak barbed simulations.

Barbed Congruence

Weak barbed bisimilarity:

- $M \approx N$ iff there is weak barbed bisimulation $\mathcal{R}$ such that $\mathcal{R}(M, N)$

Now take the associated congruence (wrt parallel and restriction):

- $\approx_c$ is the largest congruence contained in $\approx$

'Weak' refers to the fact that one reduction can be matched by many (or none). Reasonable if reductions cannot be controlled or counted.

It turns out $\approx_c$ is weak in another sense.

Barbed Congruence

Recall

Loop Lemma

If $M \twoheadrightarrow N$ then $N \rightsquigarrow M$, and if $M \rightsquigarrow N$ then $N \twoheadrightarrow M$.

We deduce that if $M \rightarrow^* N$ then $M \approx_c N$.

In fact if M has set of barbs S then $M \approx \prod_{a \in S} k_a : a\langle \mathbf{0} \rangle$.

Remark

Similar results for other reversible calculi.

So $\approx_c$ is not very discriminating.

Roll π

Weak barbed congruence has been used to demonstrate equivalence between

- high-level semantics of rollback operator roll γ
- low-level implementation (more steps)

Note that we have the same processes but different operational semantics.

I. Lanese, C.A. Mezzina, A. Schmitt, and J.-B. Stefani. Controlling reversibility in higher-order pi.

In *Proceedings of the 22nd International Conference on Concurrency Theory, CONCUR 2011*, volume 6901 of *LNCS*, pages 297–311. Springer-Verlag, 2011

Summary

Equivalences based on arbitrary reductions $\rightarrow$ (either forward or reverse) may equate too many processes.

An obvious alternative is to separate out the forward and reverse reductions.

We shall look at such equivalences shortly.

Labelled Transition Systems

LTSs

Definition

A labelled transition relation is a structure $(\mathrm{Proc}, \mathrm{Act}, \rightarrow)$, where Proc is the set of processes (states), Act is the set of action labels and $\rightarrow \subseteq \mathrm{Proc} \times \mathrm{Act} \times \mathrm{Proc}$ is a transition relation.

A labelled transition system is a structure $(\mathrm{Proc}, \mathrm{Act}, \rightarrow, I)$ where $I \in \mathrm{Proc}$ is the initial state.

A process graph is an LTS where every state is (forwards) reachable from the initial state.

Convention from now on

- forward transition $\xrightarrow{a}$
- reverse transition $\overset{a}{\rightsquigarrow}$

Strong Bisimulation

$\mathcal{R}$ is a strong bisimulation if whenever $\mathcal{R}(P, Q)$ then

- If $P \xrightarrow{\alpha} P'$ then $Q \xrightarrow{\alpha} Q'$ with $\mathcal{R}(P', Q')$
- If $Q \xrightarrow{\alpha} Q'$ then $P \xrightarrow{\alpha} P'$ with $\mathcal{R}(P', Q')$

Strong bisimilarity: $P \sim Q$ iff $\mathcal{R}(P, Q)$ for some strong bisimulation $\mathcal{R}$.

Can be used to relate

- two different LTSs (initial states must be related);
- or two different states of the same LTR.

Example (Interleaving)

$a \,|\, b \sim a.b + b.a$

Strong bisimulation is

$$(a \,|\, b,\ a, b + b.a) \quad (\mathbf{0} \,|\, b,\ b) \quad (a \,|\, \mathbf{0},\ a) \quad (\mathbf{0} \,|\, \mathbf{0},\ \mathbf{0})$$

CCSk

Recall

$$\alpha.P \overset{\alpha[m]}{\rightarrow} \alpha[m].P \qquad \frac{X \overset{\beta[n]}{\rightarrow} X'}{\alpha[m].X \overset{\beta[n]}{\rightarrow} \alpha[m].X'}\ m \neq n$$

$$\frac{X \overset{\alpha[m]}{\rightarrow} X'}{X + Q \overset{\alpha[m]}{\rightarrow} X' + Q}$$

$$\frac{X \overset{\alpha[m]}{\rightarrow} X' \quad \mathrm{fresh}(m, Y)}{X \mid Y \overset{\alpha[m]}{\rightarrow} X' \mid Y} \qquad \frac{X \overset{a[m]}{\rightarrow} X' \quad Y \overset{\overline{a}[m]}{\rightarrow} Y'}{X \mid Y \overset{\tau[m]}{\rightarrow} X' \mid Y'}$$

$$\frac{X \overset{\alpha[m]}{\rightarrow} X'}{\nu a\, X \overset{\alpha[m]}{\rightarrow} \nu a\, X'}\ \alpha \neq a, \overline{a}$$

Reverse transitions $\rightsquigarrow$

$$\alpha[m].P \overset{\alpha[m]}{\rightsquigarrow} \alpha.P \qquad \text{etc.}$$

True Concurrency via Reversibility

In a reversible calculus such as CCSk we can tell $a \mid b$ apart from $a.b + b.a$.

$$a \mid b \xrightarrow{a[m]} a[m] \mid b \xrightarrow{b[n]} a[m] \mid b[n] \overset{a[m]}{\rightsquigarrow} a \mid b[n]$$

whereas

$$a.b + b.a \xrightarrow{a[m]} a[m].b \xrightarrow{b[n]} a[m].b[n] \overset{a[m]}{\not\rightsquigarrow}$$

FR Bisimulation

A natural generalisation of forward-only bisimulation:

$\mathcal{R}$ is an FR bisimulation if whenever $\mathcal{R}(P, Q)$ then

- If $P \xrightarrow{\alpha} P'$ then $Q \xrightarrow{\alpha} Q'$ with $\mathcal{R}(P', Q')$
- If $Q \xrightarrow{\alpha} Q'$ then $P \xrightarrow{\alpha} P'$ with $\mathcal{R}(P', Q')$
- If $P \overset{\alpha}{\rightsquigarrow} P'$ then $Q \overset{\alpha}{\rightsquigarrow} Q'$ with $\mathcal{R}(P', Q')$
- If $Q \overset{\alpha}{\rightsquigarrow} Q'$ then $P \overset{\alpha}{\rightsquigarrow} P'$ with $\mathcal{R}(P', Q')$

FR bisimilarity: $P \sim_{FR} Q$ iff $\mathcal{R}(P, Q)$ for some FR bisimulation $\mathcal{R}$.

Keys and Auto-concurrency

If we adapted CCS to incorporate reverse transitions (as inverses of forward transitions) then we could distinguish $a \mid b$ from $a.b + b.a$.

However we would still equate $a.a$ and $a \mid a$.

But they are different in CCSk:

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

By contrast

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

Auto-concurrency, where two events with the same label are in parallel, is an issue when it comes to the distinguishing power of equivalences.

Can be regarded as eliminated in CCSk by the key mechanism.

Events

Consider an LTR where we have a ‘diamond’

$$P \xrightarrow{a} Q \xrightarrow{b} R \quad P \xrightarrow{b} Q' \xrightarrow{a} R \text{ with } Q \neq Q'$$

Let $\sim$ be the smallest equivalence relation such that $p \xrightarrow{a} q \sim q' \xrightarrow{a} r$, i.e. opposite sides of diamonds are related.

The equivalence classes are the events.

Could have strange scenarios like

$$P \xrightarrow{a} Q \xrightarrow{b} T$$
$$P \xrightarrow{b} S \xrightarrow{a} T$$
$$P \xrightarrow{a} R \xrightarrow{b} T$$

Here all as are the same event.

Can rule this out by requiring

- event determinism (Sassone, Nielsen & Winskel; van Glabbeek): if $P \xrightarrow{a} Q$ and $P \xrightarrow{a} R$, and $(P, a, Q) \sim (P, a, R)$, then $Q = R$

The LTR of CCSk

Consider the LTR of all CCSk processes.

$$a \mid b \xrightarrow{a[m]} a[m] \mid b$$
$$ab + b.a \xrightarrow{a[m]} a[m].b + b.a$$

Some states are irreversible: $X \not\rightsquigarrow$

('standard' terms P)

Can analyse properties, e.g.

- event determinism

Prime LTRs

An LTR is prime if it satisfies:

- **WF** (well-founded) there is no infinite reverse computation;
- **UT** (unique transition) if $P \xrightarrow{a} Q$ and $P \xrightarrow{b} Q$ then $a = b$;
- **ED** (event-deterministic) if $P \xrightarrow{a} Q$ and $P \xrightarrow{a} R$, and $(P, a, Q) \sim (P, a, R)$, then $Q = R$;
- **RD** (reverse diamond) if $Q \xrightarrow{a} P$, $R \xrightarrow{b} P$ and $Q \neq R$, then there is S such that $S \xrightarrow{a} R$, $S \xrightarrow{b} Q$;
- **FD** (forward diamond) if $P \xrightarrow{a} Q \rightarrow^* T$, $P \xrightarrow{b} R \rightarrow^* T$, and $Q \neq R$, then there is S such that $Q \xrightarrow{b} S$, $R \xrightarrow{a} S$ and $S \rightarrow^* T$.

It can be checked that all five properties are independent of each other.

(Phillips & Ulidowski 2007)

Events in Prime LTRs

It can be shown that any path in a prime LTR cannot have repeated events.

So any state in a prime graph is associated with a well-defined set of events.

Can now define true concurrency notions such as

- causality
- concurrency
- conflict

CCSk is Prime

It can also be shown that the labelled transition relation associated with CCSk satisfies all the prime properties.

So just analysing the LTR takes us to a true concurrency model.

The CCSk LTR has a further property:

> **NR** *(non-repeating) there are no repeated labels in forward computations.*

CCSk graphs have NR as a consequence of using the keys mechanism.

Prime graphs that satisfy NR are called *non-repeating* prime graphs.

Prime Event Structures

Labelled prime event structure: $\mathcal{E} = (E, <, \sharp, \ell)$ where

- E are events
- $<$ is the causality relation (a partial order)
  (finitely many causes for an event)
- $\sharp \subseteq E \times E$ is the conflict relation
  (conflict heredity: if $e \sharp e' < e''$ then $e \sharp e''$)
- ℓ is the labelling $\ell : E \to \text{Act}$.

Conflict

Can generalise the conflict relation to be non-binary.

Can have $\sharp\{a, b, c\}$ without any proper subset being in conflict.

Configuration Graphs

Configuration graph:

- configurations are finite conflict-free downwards-closed sets of events.
- transitions: $X \xrightarrow{a}_{\mathcal{E}} Y$ if $Y \setminus X = \{e\}$ with $\ell(e) = a$.

The configuration graph of a prime event structure (even with non-binary conflict) is a prime graph.

Conversely a prime graph can be converted into a prime event structure with non-binary conflict.

Theorem

Prime graphs and prime event structures with non-binary conflict are isomorphic as models.

(with help of result of van Glabbeek & Vaandrager)

Non-repeating

Prime event structures that correspond to non-repeating prime graphs enjoy the properties of

- no auto-concurrency (not $a \mid a$)
- and no auto-causation (not $a < a$).

FR Bisimulation

Bednarczyk (1991)

FR bisimulation same as hereditary history-preserving (HH) bisimulation on prime event structures with binary conflict and no auto-concurrency.

HH bisimulation defined later.

Theorem

Let $\mathcal{E}$ and $\mathcal{F}$ be non-repeating prime event structures with non-binary conflict, and let $\mathcal{C}$ and $\mathcal{D}$ be their configuration graphs.

Then, $\mathcal{C} \sim_{HH} \mathcal{D}$ if and only if $\mathcal{C} \sim_{FR} \mathcal{D}$.

We shall see an improvement later.

Summary

- The LTR of CCSk processes is prime and non-repeating
- The graph of a CCSk process corresponds to a prime event structure
- FR bisimulation on CCSk processes correponds to hereditary history-preserving (HH) bisimulation on prime event structures
- HH bisimulation is regarded as the canonical equivalence on labelled event structures.

Equivalences on configuration structures

Configuration structures

A more general model of processes: stable configuration structures.

These are $\mathcal{C} = (C, \ell)$ where

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

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

Various axioms:

- rooted: $\emptyset \in C$;
- connected: $\emptyset \neq X \in C$ implies $\exists e \in X : X \setminus \{e\} \in C$;
- 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.

$$\begin{tikzcd}
 & \{0,1,b\} & \\
\{0,b\} \arrow[ur] & \{0,1\} \arrow[u] & \{1,b\} \arrow[ul] \\
\{0\} \arrow[u] \arrow[ur] & & \{1\} \arrow[u] \arrow[ul] \\
\mathcal{C} & \emptyset \arrow[ul] \arrow[ur] &
\end{tikzcd}
\qquad
\begin{tikzcd}
\{0,b\} & \{0,1\} & \{1,b\} \\
\{0\} \arrow[u] \arrow[ur] & & \{1\} \arrow[u] \arrow[ul] \\
\mathcal{D} & \emptyset \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

$\mathcal{C}$ 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.

Reverse transitions: let $X \overset{a}{\rightsquigarrow}_{\mathcal{C}} Y$ if $Y \xrightarrow{a}_{\mathcal{C}} X$.

Example (Parallel Switch)

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

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

- By connectedness, every configuration of a stable configuration structure is reachable (by forwards transitions).
- With reverse transitions we have an abstract model of a reversible system satisfying the Loop Lemma.

Forward-only equivalences

(van Glabbeek & Goltz, 2001)

Various bisimulation-based equivalences, forward-only.

- interleaving bisimulation $\approx_{ib}$

Can 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 $\approx_{pb}$

Can 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 $a \in \mathsf{Act}$
   - 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 $\mathcal{D}$.

$$a \,|\, b \approx_{ib} a.b + b.a$$

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

$\approx_{db}$ is depth-respecting bisimulation - idea is that one can observe the causal depth of an event.

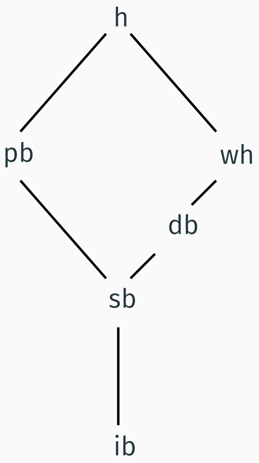
Theorem

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

Example

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

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


$$\begin{tikzcd}
 & h \arrow[dl, dash] \arrow[dr, dash] & \\
pb \arrow[ddr, dash] & & wh \arrow[d, dash] \\
 & & db \arrow[dl, dash] \\
 & sb \arrow[d, dash] & \\
 & ib &
\end{tikzcd}$$

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.

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

$$\begin{tikzcd}
X \arrow[rrr, dash, dashed, "f"] \arrow[d, "e"'] & & & Y \arrow[d, "e'"] \\
X' \arrow[rrr, dash, dashed, "f'"] & & & Y'
\end{tikzcd}$$

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

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

$$\begin{tikzcd}
X \arrow[r, dash, dashed, "f"] \arrow[d, "e"'] & Y \arrow[d, "e'"] \\
X' \arrow[r, dash, dashed, "f'"] & Y'
\end{tikzcd}
\quad \text{and} \quad
\begin{tikzcd}
X \arrow[r, dash, dashed, "f"] \arrow[d, squiggly, "d"'] & Y \arrow[d, squiggly, "d'"] \\
X' \arrow[r, dash, dashed, "f''"] & Y'
\end{tikzcd}$$

  then $\mathcal{R}$ is a hereditary weak history-preserving (HWH) bisimulation.

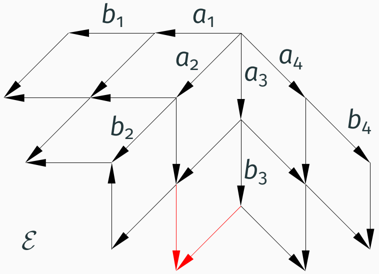
- if additionally $f' \restriction X = f$, then $\mathcal{R}$ is a hereditary history-preserving (HH) bisimulation (Bednarczyk 1991).

Hierarchy of equivalences

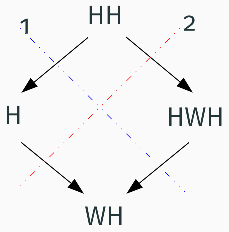
1. The Absorption Law holds only for H and WH

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

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



Overall:



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

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

The Absorption Law

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

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

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 $(X', Y') \in R$.

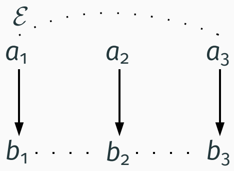
We have $\approx_{hh} \subseteq \approx_{rs-sb} \subsetneq \approx_{ri-ib}$.

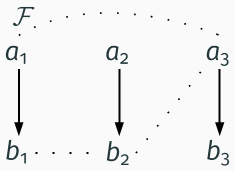
Open Question (Bednarczyk 1991)

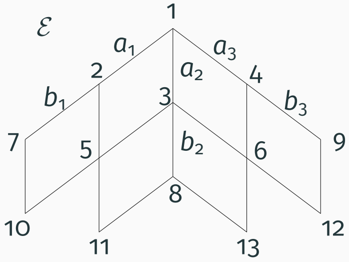
Does $\approx_{rs-sb} = \approx_{hh}$?

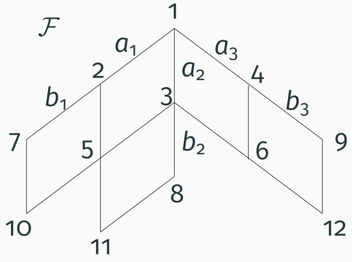
Counterexample

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









A Hierarchy of Equivalences

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

A Hierarchy of Equivalences

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

$$\begin{tikzcd}
\mathrm{h-h} \arrow[rr, -{Triangle[open]}] \arrow[d] & & \mathrm{rp-h} \arrow[rrr, -{Triangle[open]}] \arrow[d] & & & \mathrm{rs-h} \arrow[rrr, dash] \arrow[d] & & & \mathrm{ri-h} \arrow[rrr] \arrow[d] & & & \mathrm{h} \arrow[d] & \\
\mathrm{h-whpb} \arrow[rr, -{Triangle[open]}] \arrow[d] & & \mathrm{rp-whpb} \arrow[rrr, -{Triangle[open]}] \arrow[dr] \arrow[ddl, dash] & & & \mathrm{rs-whpb} \arrow[rrr, dash] \arrow[dr] \arrow[ddl, -{Triangle[open]}] & & & \mathrm{ri-whpb} \arrow[rrr] \arrow[dr] \arrow[ddl] & & & \mathrm{whpb} \arrow[dr] \arrow[ddl] & \\
\mathrm{h-wh} \arrow[rrr, -{Triangle[open]}] & & & \mathrm{rp-wh} \arrow[rrr, -{Triangle[open]}] \arrow[dd, dash] & & & \mathrm{rs-wh} \arrow[rrr, dash] \arrow[dd, -{Triangle[open]}] & & & \mathrm{ri-wh} \arrow[rrr] \arrow[dd, -{Triangle[open]}] & & & \mathrm{wh} \arrow[dd] \\
& \mathrm{rp-pb} \arrow[rrr, -{Triangle[open]}] \arrow[ddr] & & & \mathrm{rs-pb} \arrow[rrr] \arrow[ddr] & & & \mathrm{ri-pb} \arrow[rrr] \arrow[ddr] & & & \mathrm{pb} \arrow[ddr] & & \\
& & & \mathrm{rp-db} \arrow[rrr, -{Triangle[open]}] \arrow[dl, dash] & & & \mathrm{rs-db} \arrow[rrr, -{Triangle[open]}] \arrow[dl, dash] & & & \mathrm{ri-db} \arrow[rrr] \arrow[dl] & & & \mathrm{db} \arrow[dl] \\
& & \mathrm{rp-sb} \arrow[rrr, -{Triangle[open]}] \arrow[d, dash] & & & \mathrm{rs-sb} \arrow[rrr] \arrow[d, dash] & & & \mathrm{ri-sb} \arrow[rrr] \arrow[d] & & & \mathrm{sb} \arrow[d] & \\
& & \mathrm{rp-ib} \arrow[rrr, -{Triangle[open]}] & & & \mathrm{rs-ib} \arrow[rrr] & & & \mathrm{ri-ib} \arrow[rrr] & & & \mathrm{ib} &
\end{tikzcd}$$

Open-headed arrows: open whether inclusion is proper.

Summary

- Rich hierarchy of equivalences on stable configuration structures.
- Observations can be (labels of) single events, steps (of concurrent events), pomsets.
- Can combine different amounts of observational power in forward or reverse directions.

Logics for Reversibility

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.

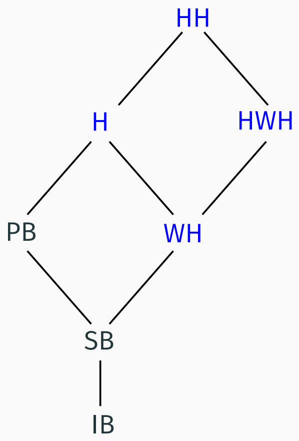
Motivation

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 $\langle a\rangle\!\rangle\langle b\rangle\!\rangle\langle\!\langle a\rangle$tt, while $a.b + b.a$ does not. They are not sufficient, especially in the presence of auto-concurrency. $\langle a\rangle\!\rangle\langle a\rangle\!\rangle\langle\!\langle a\rangle$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 $\langle x : a\rangle\rangle\langle y : a\rangle\rangle\langle\langle x\rangle \mathrm{tt}$ 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).

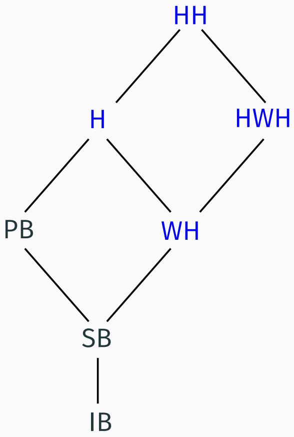
$\Longrightarrow$ can characterise WH and HWH.

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 $\exists Q.\ P \xrightarrow{a} Q \models \varphi$

$P \models [a]]\ \varphi$ iff $\forall Q.\ P \xrightarrow{a} Q$ implies $Q \models \varphi$

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:

$$a \mid b \models \langle a \rangle\!\rangle \langle b \rangle\!\rangle \langle\!\langle a \rangle \mathrm{tt} \qquad a.b + b.a \not\models \langle a \rangle\!\rangle \langle b \rangle\!\rangle \langle\!\langle a \rangle \mathrm{tt}$$

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

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

Event Identifier Logic

Assume an infinite set of identifiers x which can be bound to any events.

Event Identifier Logic (EIL) is:

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

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 free identifiers $\mathrm{fi}(\phi) \subseteq \mathrm{dom}(\rho)$ and $\mathrm{rge}(\rho \restriction \mathrm{fi}(\phi)) \subseteq X$.

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

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

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 $\mathrm{rge}(\rho \restriction y) = \{e'_a\} \not\subseteq \{e_a\}$.

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

Examples

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

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

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 $\mathcal{D} \models \phi$:

$$\begin{tikzcd}
& \text{HH} \arrow[dl] \arrow[dr] & \\
\text{H} \arrow[dr] & & \text{HWH} \arrow[dl] \\
& \text{WH} &
\end{tikzcd}$$

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 $\mathrm{EIL}_{\mathrm{ro}}$:

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

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

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

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

Theorem

$\mathrm{EIL}_{\mathrm{h}}$ *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 $\mathrm{EIL_{ro}}$:

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

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.

Adding Recursion

To express more interesting properties, such as:

can do a causal chain of bs to reach a state where can do a

we can add recursion, following Baldan & Crafa (2014).

- Recursion variables $X(\vec{x})$ with free identifiers $\vec{x}$.
- Least fixed point operator $\mu X(\vec{x}).\phi$.
- Occurrences of X within ϕ must be positive to guarantee least fixed point exists.
- Can see $\mu X(\vec{x})\phi$ as an infinite disjunction of the finite unfoldings.
- Adding recursion to the various sublogics does not change the induced equivalences.

Examples

We translate some examples from Baldan & Crafa (2014):

Example

Infinite causal chain of events labelled a: $e_0 < e_1 < e_2 < \cdots$.

$$\langle x : a \rangle\!\rangle \mu X(x).\, \langle y : a \rangle\!\rangle (\neg \langle\!\langle x \rangle \mathrm{tt} \wedge X(y))$$

Think of $X(x)$ as the state of the computation just after performing x.

Example

Can do a causal chain of zero or more bs to reach a state where can do a.

Examples

We translate some examples from Baldan & Crafa (2014):

Example

Infinite causal chain of events labelled a: $e_0 < e_1 < e_2 < \cdots$.

$$\langle x : a \rangle\!\rangle \mu X(x). \langle y : a \rangle\!\rangle (\neg \langle\!\langle x \rangle \mathrm{tt} \land X(y))$$

Think of $X(x)$ as the state of the computation just after performing x.

Example

Can do a causal chain of zero or more bs to reach a state where can do a.

$$\langle y : a \rangle\!\rangle \mathrm{tt} \lor \langle x : b \rangle\!\rangle \mu X(x). [\langle z : a \rangle\!\rangle \lor \langle y : b \rangle\!\rangle (\neg \langle\!\langle x \rangle \mathrm{tt} \land X(y))]$$

Examples

Example

Can perform a series of $\{a, b\}$ steps (not necessarily causally related) to reach a state where can perform c.

Examples

Example

Can perform a series of $\{a, b\}$ steps (not necessarily causally related) to reach a state where can perform c. step:

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

Answer:

$$\mu X. \left[\langle z : c\rangle\rangle \mathrm{tt} \vee \langle x : a\rangle\rangle \langle y : b\rangle\rangle (\langle\langle x\rangle \mathrm{tt} \wedge X)\right]$$

No parameter needed in X.

Characteristic Formulas

Characteristic Formulas

Can we reduce checking whether $\mathcal{C}$ and $\mathcal{D}$ satisfy the same formulas in EIL to

- does $\mathcal{D}$ satisfy $\chi_{\mathcal{C}}$?

Here $\chi_{\mathcal{C}}$ is the characteristic formula of $\mathcal{C}$

- completely expresses the behaviour of $\mathcal{C}$, at least as far as the particular logic is concerned

May or may not exist.

If exists, checking whether two structures are equivalent is changed from the problem of potentially having to check infinitely many formulas into a single model-checking problem $\mathcal{D} \models \chi_{\mathcal{C}}$.

HH

Consider just finite configuration structures.

Consider characteristic formulas wrt HH equivalence.

Not obvious that such formulas can be finite since we can go forward and backwards indefinitely.

However for each $\mathcal{C}$ we get a family of formulas $\chi^{\mathrm{hh}}_{\mathcal{C},s}$ with size parameter s satisfying

$$\mathcal{C} \approx_{\mathsf{hh}} \mathcal{D} \textit{ iff } \mathcal{D} \models \chi^{\mathrm{hh}}_{\mathcal{C},s}$$

where s depends on $\mathcal{D}$.

Thus we do not have a single characteristic formula for $\mathcal{C}$, but we can deal uniformly with all $\mathcal{D}$ up to a certain size.

Almost as good as having a single characteristic formula for $\mathcal{C}$, since we can generate a formula of the appropriate size once we have settled on $\mathcal{D}$.

Example

Can certainly have a single formula in individual cases.

Example

Consider the configuration structure $\mathcal{C}_a$ represented by the CCS process a. Configurations are $\emptyset$ and $\{e\}$ with $\ell(e) = a$.

The single formula

$$\phi_a \overset{\text{df}}{=} \langle x : a \rangle\!\rangle \text{tt} \wedge (\,[x : a]]\bigwedge_{b \in \text{Act}} [y : b]]\,\text{ff}) \wedge \bigwedge_{b \in \text{Act}, b \neq a} [y : b]]\,\text{ff}$$

characterises $\mathcal{C}_a$ for HH equivalence.

Single formulas possible in more complicated examples, but open question whether always possible.

Remark

Single characteristic formulas are possible for H and WH equivalence.

Summary

- Can extend Hennessy-Milner logic with reverse modalities and event identifiers. Also recursion.
- The full logic characterises Hereditary History-preserving equivalence.
- Equivalences in the hierarchy from the previous section are characterised by sublogics.
- Characteristic formulas are more efficient for equivalence checking.

Conclusions

Conclusions

- Equivalences such as weak barbed congruence which do not distinguish between forward and reverse transitions are very undiscriminating on fully reversible calculi.
- 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 Hennessy-Milner Logic with reverse modalities and event identifiers. Simple logical characterisations of wh, h and hh bisimulations.
- The 'canonical' equivalence for reversible calculi is not yet clear.

References I

P. Baldan and S. Crafa.
A logic for true concurrency.
In *Proceedings of 21st International Conference on Concurrency Theory, CONCUR 2010*, volume 6269 of *LNCS*, pages 147–161. Springer-Verlag, 2010.

Paolo Baldan and Silvia Crafa.
A logic for true concurrency.
J. ACM, 61(4):24:1–24:36, 2014.

M.A. Bednarczyk.
Hereditary history preserving bisimulations or what is the power of the future perfect in program logics.
Technical report, Institute of Computer Science, Polish Academy of Sciences, Gdańsk, 1991.

References II

R.J. van Glabbeek and U. Goltz.
Refinement of actions and equivalence notions for concurrent systems.
Acta Informatica, 37(4/5):229–327, 2001.

I. Lanese.
Reversibility for concurrent interacting systems.
International Training School on Reversible Computation, 2017.

I. Lanese, C.A. Mezzina, A. Schmitt, and J.-B. Stefani.
Controlling reversibility in higher-order pi.
In *Proceedings of the 22nd International Conference on Concurrency Theory, CONCUR 2011*, volume 6901 of *LNCS*, pages 297–311. Springer-Verlag, 2011.

References III

I. Lanese, C.A. Mezzina, and J.-B. Stefani.
Reversibility in the higher-order π-calculus.
Theoretical Computer Science, 625:25–84, 2016.

I.C.C. Phillips and I. Ulidowski.
Reversing algebraic process calculi.
In *Proceedings of 9th International Conference on Foundations of Software Science and Computation Structures, FOSSACS 2006*, volume 3921 of *LNCS*, pages 246–260. Springer-Verlag, 2006.

I.C.C. Phillips and I. Ulidowski.
Reversibility and models for concurrency.
In M.C.B. Hennessy and R. van Glabbeek, editors, *Proceedings of Fourth International Workshop on Structural Operational Semantics (SOS 2007)*, volume 192(1) of *Electronic Notes in*

References IV

Theoretical Computer Science, pages 93–108. Elsevier, Amsterdam, 2007.

I.C.C. Phillips and I. Ulidowski.
Reversing algebraic process calculi.
Journal of Logic and Algebraic Programming, 73(1-2):70–96, 2007.

I.C.C. Phillips and I. Ulidowski.
A hierarchy of reverse bisimulations on stable configuration structures.
Mathematical Structures in Computer Science, 22:333–372, 2012.

I.C.C. Phillips and I. Ulidowski.
Event identifier logic.
Mathematical Structures in Computer Science, 24:e240204, 2014.