The Chemistry of Lévy Labels

Luca Cardelli Cosimo Laneve

GGJJ colloque

programme

- define a reversible process algebra that is DNA inspired
- study the interplay between reversibility and causal dependency
  - use Lévy labels for tagging redexes (Ll are encoded through DNA domains)
  - define causal equivalent derivations (permutation equivalence)
- provide a fine-grain implementation of asynchronous RCCS and its memories

background: DNA circuits (cf. Cardelli/Phillips)

- subsequences on a DNA strand are called domains

  domains are independent of each other

  - they cannot hybridize from any other domain except their complement

- there are very few short domains with reversible hybridizations

- and long domains with irreversible hybridizations

background: DNA circuits (cf. Cardelli/Phillips)

- subsequences on a DNA strand are called domains

  domains are independent of each other

  - they cannot hybridize from any other domain except their complement

- there are very few short domains with reversible hybridizations

- and long domains with irreversible hybridizations

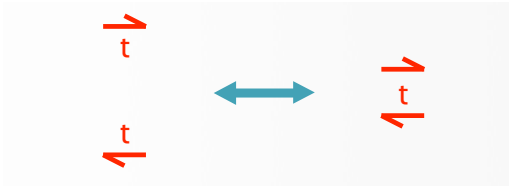
background: DNA circuits (cf. Cardelli/Phillips)

- subsequences on a DNA strand are called domains

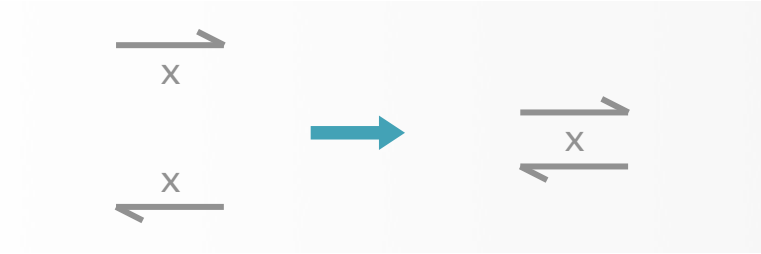
  domains are independent of each other

  - they cannot hybridize from any other domain except their complement

- there are very few short domains with reversible hybridizations

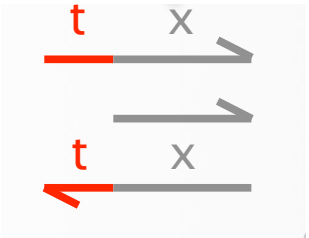


- and long domains with irreversible hybridizations



reversibility in DNA circuits

short-domains mediated strand displacements

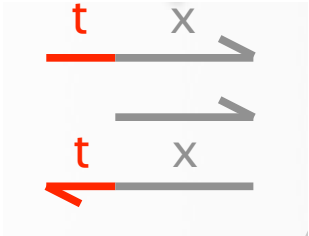


are irreversible

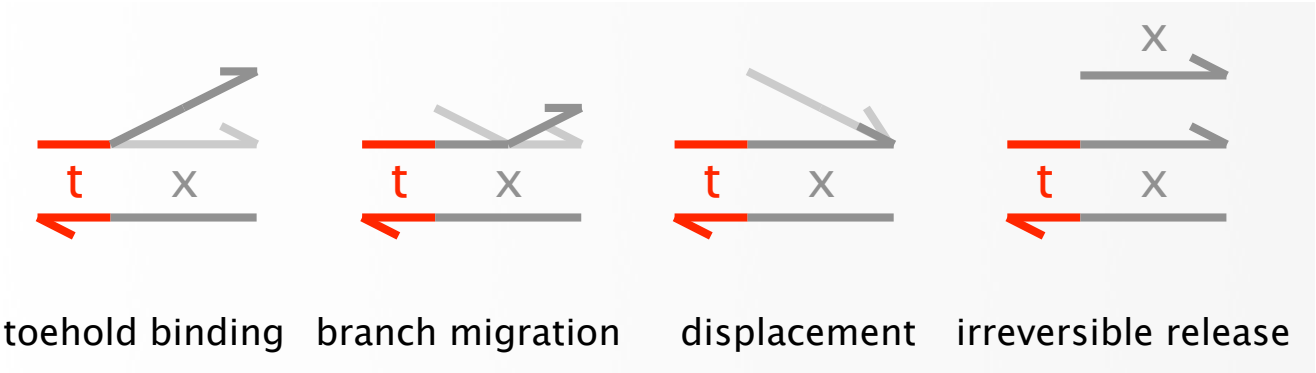
however clever designs may make them reversible

reversibility in DNA circuits

short-domains mediated strand displacements



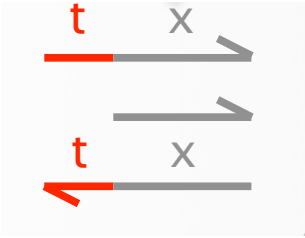
are irreversible



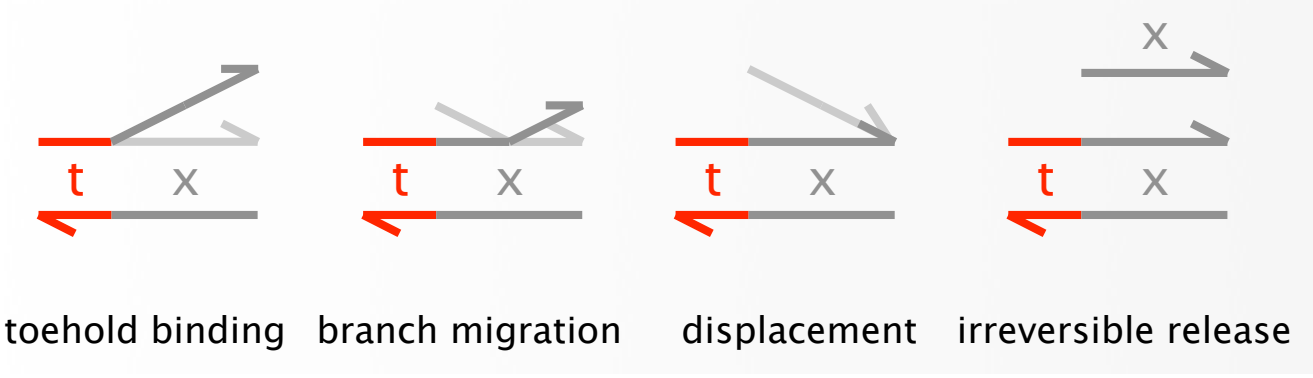
however clever designs may make them reversible

reversibility in DNA circuits

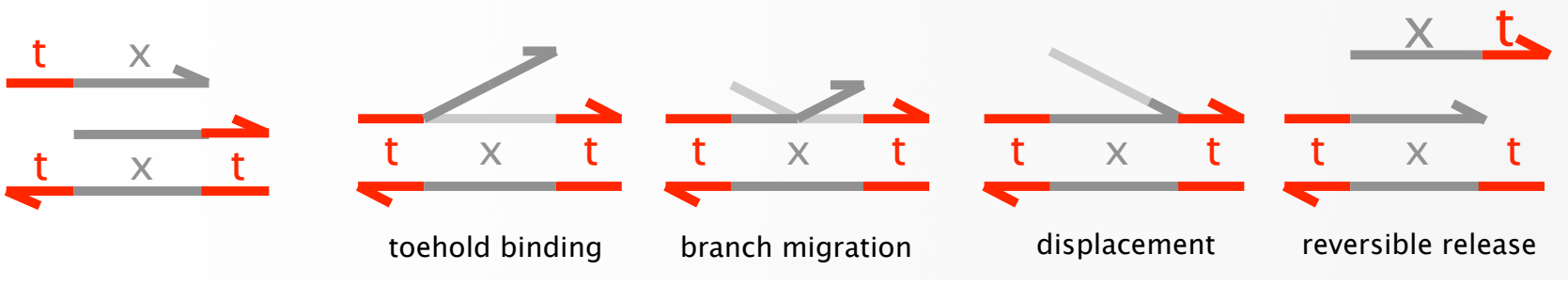
short-domains mediated strand displacements



are irreversible



however clever designs may make them reversible



the algebra of reversible structures

notation:

u, v, w:	ids
$a, \bar{a}, b, \bar{b}$:	names and conames
$\mathtt{A}, \mathtt{B}, \mathtt{C}$:	sequences of names;
$\overline{\mathtt{A}}, \overline{\mathtt{B}}\ \overline{\mathtt{C}}$:	sequences of elements $u : \bar{a}$
$\mathtt{A}^{\perp}, \mathtt{B}^{\perp}, \mathtt{C}^{\perp}$:	sequences of elements $u : a$

gates : g *input part.output.part* + ^

examples : $\hat{}a.a'.v : \bar{b}$ $u : a.\hat{}a'.v : \bar{b}$ $u : a.u' : a'.v : \bar{b}\hat{}$

structures : $\mathtt{S}$ ::=

	$\mathbf{0}$	(null)
|	$u : \bar{a}$	(signal)
|	g	(gate)
|	$\mathtt{S} \mid \mathtt{S}$	(parallel)
|	(new a)S	(new)

the algebra of reversible structures

notation:

u, v, w:	ids
$a, \bar{a}, b, \bar{b}$:	names and conames
$\mathtt{A}, \mathtt{B}, \mathtt{C}$:	sequences of names;
$\overline{\mathtt{A}}, \overline{\mathtt{B}}\ \overline{\mathtt{C}}$:	sequences of elements $u : \bar{a}$
$\mathtt{A}^{\perp}, \mathtt{B}^{\perp}, \mathtt{C}^{\perp}$:	sequences of elements $u : a$

gates : g *input part.output.part* + ˆ

examples : $\hat{}a.a'.v : \bar{b}$ $u : a.\hat{}a'.v : \bar{b}$ $u : a.u' : a'.v : \bar{b}\hat{}$

structures : $\mathtt{S}$ ::=

	$\mathbf{0}$	(null)
|	$u : \bar{a}$	(signal)
|	g	(gate)
|	$\mathtt{S} \mid \mathtt{S}$	(parallel)
|	(new a)S	(new)

the algebra of reversible structures

notation:

u, v, w:	ids
$a, \bar{a}, b, \bar{b}$:	names and conames
$\mathtt{A}, \mathtt{B}, \mathtt{C}$:	sequences of names;
$\overline{\mathtt{A}}, \overline{\mathtt{B}}\ \overline{\mathtt{C}}$:	sequences of elements $u : \bar{a}$
$\mathtt{A}^{\perp}, \mathtt{B}^{\perp}, \mathtt{C}^{\perp}$:	sequences of elements $u : a$

gates : g *input part.output.part* + ˆ

examples : $\hat{}a.a'.v : \bar{b}$ $u : a.\hat{}a'.v : \bar{b}$ $u : a.u' : a'.v : \bar{b}\hat{}$

structures : $\mathtt{S}$ ::=

	$\mathbf{0}$	(null)
|	$u : \bar{a}$	(signal)
|	g	(gate)
|	$\mathtt{S} \mid \mathtt{S}$	(parallel)
|	(new a)S	(new)

the algebra of reversible structures – reductions

input capture: $u : \overline{a} \mid \mathtt{A}^{\perp}.\hat{}a.\mathtt{B}.\overline{\mathtt{C}} \longrightarrow \mathtt{A}^{\perp}.u : a.\hat{}\mathtt{B}.\overline{\mathtt{C}}$

input release: $\mathtt{A}^{\perp}.u : a.\hat{}\mathtt{B}.\overline{\mathtt{C}} \longrightarrow u : \overline{a} \mid \mathtt{A}^{\perp}.\hat{}a.\mathtt{B}.\overline{\mathtt{C}}$

output release: $\mathtt{A}^{\perp}.\overline{\mathtt{B}}.\hat{}u : \overline{a}.\overline{\mathtt{C}} \longrightarrow u : \overline{a} \mid \mathtt{A}^{\perp}.\overline{\mathtt{B}}.u : \overline{a}.\hat{}\overline{\mathtt{C}}$

output capture: $u : \overline{a} \mid \mathtt{A}^{\perp}.\overline{\mathtt{B}}.u : \overline{a}.\hat{}\overline{\mathtt{C}} \longrightarrow \mathtt{A}^{\perp}.\overline{\mathtt{B}}.\hat{}u : \overline{a}.\overline{\mathtt{C}}$

(plus the standard contextual rules about new, $\mid$, and $\equiv$)

example:

$$w : \overline{a} \mid v : a.\hat{}u : \overline{b} \quad \rightarrow \quad w : \overline{a} \mid v : a.u : \overline{b}\hat{} \mid u : \overline{b}$$

$$v : \overline{a} \mid w : \overline{a} \mid \hat{}a.u : \overline{b} \quad \nearrow \quad \searrow$$

$$v : \overline{a} \mid w : a.\hat{}u : \overline{b}$$

the algebra of reversible structures – reductions

input capture: $u : \overline{a} \mid \mathtt{A}^{\perp}.\hat{}a.\mathtt{B}.\overline{\mathtt{C}} \longrightarrow \mathtt{A}^{\perp}.u : a.\hat{}\mathtt{B}.\overline{\mathtt{C}}$

input release: $\mathtt{A}^{\perp}.u : a.\hat{}\mathtt{B}.\overline{\mathtt{C}} \longrightarrow u : \overline{a} \mid \mathtt{A}^{\perp}.\hat{}a.\mathtt{B}.\overline{\mathtt{C}}$

output release: $\mathtt{A}^{\perp}.\overline{\mathtt{B}}.\hat{}u : \overline{a}.\overline{\mathtt{C}} \longrightarrow u : \overline{a} \mid \mathtt{A}^{\perp}.\overline{\mathtt{B}}.u : \overline{a}.\hat{}\overline{\mathtt{C}}$

output capture: $u : \overline{a} \mid \mathtt{A}^{\perp}.\overline{\mathtt{B}}.u : \overline{a}.\hat{}\overline{\mathtt{C}} \longrightarrow \mathtt{A}^{\perp}.\overline{\mathtt{B}}.\hat{}u : \overline{a}.\overline{\mathtt{C}}$

(plus the standard contextual rules about new, $\mid$, and $\equiv$)

example:

$v : \overline{a} \mid w : \overline{a} \mid \hat{}a.u : \overline{b}$

$\nearrow$ $w : \overline{a} \mid v : a.\hat{}u : \overline{b} \longrightarrow w : \overline{a} \mid v : a.u : \overline{b}\hat{} \mid u : \overline{b}$

$\searrow$ $v : \overline{a} \mid w : a.\hat{}u : \overline{b}$

the algebra of reversible structures – reductions

input capture: $u : \overline{a} \mid \mathtt{A}^{\perp}.\hat{}a.\mathtt{B}.\overline{\mathtt{C}} \longrightarrow \mathtt{A}^{\perp}.u : a.\hat{}\mathtt{B}.\overline{\mathtt{C}}$

input release: $\mathtt{A}^{\perp}.u : a.\hat{}\mathtt{B}.\overline{\mathtt{C}} \longrightarrow u : \overline{a} \mid \mathtt{A}^{\perp}.\hat{}a.\mathtt{B}.\overline{\mathtt{C}}$

output release: $\mathtt{A}^{\perp}.\overline{\mathtt{B}}.\hat{}u : \overline{a}.\overline{\mathtt{C}} \longrightarrow u : \overline{a} \mid \mathtt{A}^{\perp}.\overline{\mathtt{B}}.u : \overline{a}.\hat{}\overline{\mathtt{C}}$

output capture: $u : \overline{a} \mid \mathtt{A}^{\perp}.\overline{\mathtt{B}}.u : \overline{a}.\hat{}\overline{\mathtt{C}} \longrightarrow \mathtt{A}^{\perp}.\overline{\mathtt{B}}.\hat{}u : \overline{a}.\overline{\mathtt{C}}$

(plus the standard contextual rules about new, | , and ≡)

example:

$$v : \overline{a} \mid w : \overline{a} \mid \hat{}a.u : \overline{b} \quad \begin{array}{l} \nearrow \\ \searrow \end{array} \quad \begin{array}{l} w : \overline{a} \mid v : a.\hat{}u : \overline{b} \longrightarrow w : \overline{a} \mid v : a.u : \overline{b}\hat{} \mid u : \overline{b} \\ v : \overline{a} \mid w : a.\hat{}u : \overline{b} \end{array}$$

the algebra of reversible structures – reductions

input capture: $u : \overline{a} \mid \mathtt{A}^{\perp}.\hat{}a.\mathtt{B}.\overline{\mathtt{C}} \longrightarrow \mathtt{A}^{\perp}.u : a.\hat{}\mathtt{B}.\overline{\mathtt{C}}$

input release: $\mathtt{A}^{\perp}.u : a.\hat{}\mathtt{B}.\overline{\mathtt{C}} \longrightarrow u : \overline{a} \mid \mathtt{A}^{\perp}.\hat{}a.\mathtt{B}.\overline{\mathtt{C}}$

output release: $\mathtt{A}^{\perp}.\overline{\mathtt{B}}.\hat{}u : \overline{a}.\overline{\mathtt{C}} \longrightarrow u : \overline{a} \mid \mathtt{A}^{\perp}.\overline{\mathtt{B}}.u : \overline{a}.\hat{}\overline{\mathtt{C}}$

output capture: $u : \overline{a} \mid \mathtt{A}^{\perp}.\overline{\mathtt{B}}.u : \overline{a}.\hat{}\overline{\mathtt{C}} \longrightarrow \mathtt{A}^{\perp}.\overline{\mathtt{B}}.\hat{}u : \overline{a}.\overline{\mathtt{C}}$

(plus the standard contextual rules about new, | , and ≡)

example:

$$v : \overline{a} \mid w : \overline{a} \mid \hat{}a.u : \overline{b} \quad \begin{array}{l} \nearrow \\ \searrow \end{array} \quad \begin{array}{l} w : \overline{a} \mid v : a.\hat{}u : \overline{b} \longrightarrow w : \overline{a} \mid v : a.u : \overline{b}\hat{} \mid u : \overline{b} \\ v : \overline{a} \mid w : a.\hat{}u : \overline{b} \end{array}$$

the algebra of reversible structures – reductions

input capture: $u : \overline{a} \mid \mathtt{A}^{\perp}.\hat{}a.\mathtt{B}.\overline{\mathtt{C}} \longrightarrow \mathtt{A}^{\perp}.u : a.\hat{}\mathtt{B}.\overline{\mathtt{C}}$

input release: $\mathtt{A}^{\perp}.u : a.\hat{}\mathtt{B}.\overline{\mathtt{C}} \longrightarrow u : \overline{a} \mid \mathtt{A}^{\perp}.\hat{}a.\mathtt{B}.\overline{\mathtt{C}}$

output release: $\mathtt{A}^{\perp}.\overline{\mathtt{B}}.\hat{}u : \overline{a}.\overline{\mathtt{C}} \longrightarrow u : \overline{a} \mid \mathtt{A}^{\perp}.\overline{\mathtt{B}}.u : \overline{a}.\hat{}\overline{\mathtt{C}}$

output capture: $u : \overline{a} \mid \mathtt{A}^{\perp}.\overline{\mathtt{B}}.u : \overline{a}.\hat{}\overline{\mathtt{C}} \longrightarrow \mathtt{A}^{\perp}.\overline{\mathtt{B}}.\hat{}u : \overline{a}.\overline{\mathtt{C}}$

(plus the standard contextual rules about new, | , and ≡)

example:

$$v : \overline{a} \mid w : \overline{a} \mid \hat{}a.u : \overline{b} \quad \begin{array}{l} \nearrow \\ \searrow \end{array} \quad \begin{array}{l} w : \overline{a} \mid v : a.\hat{}u : \overline{b} \longrightarrow w : \overline{a} \mid v : a.u : \overline{b}\hat{} \mid u : \overline{b} \\ v : \overline{a} \mid w : a.\hat{}u : \overline{b} \end{array}$$

the algebra of reversible structures – reductions

input capture: $u : \overline{a} \mid \mathtt{A}^{\perp}.\hat{}a.\mathtt{B}.\overline{\mathtt{C}} \longrightarrow \mathtt{A}^{\perp}.u : a.\hat{}\mathtt{B}.\overline{\mathtt{C}}$

input release: $\mathtt{A}^{\perp}.u : a.\hat{}\mathtt{B}.\overline{\mathtt{C}} \longrightarrow u : \overline{a} \mid \mathtt{A}^{\perp}.\hat{}a.\mathtt{B}.\overline{\mathtt{C}}$

output release: $\mathtt{A}^{\perp}.\overline{\mathtt{B}}.\hat{}u : \overline{a}.\overline{\mathtt{C}} \longrightarrow u : \overline{a} \mid \mathtt{A}^{\perp}.\overline{\mathtt{B}}.u : \overline{a}.\hat{}\overline{\mathtt{C}}$

output capture: $u : \overline{a} \mid \mathtt{A}^{\perp}.\overline{\mathtt{B}}.u : \overline{a}.\hat{}\overline{\mathtt{C}} \longrightarrow \mathtt{A}^{\perp}.\overline{\mathtt{B}}.\hat{}u : \overline{a}.\overline{\mathtt{C}}$

(plus the standard contextual rules about **new**, $\mid$, and $\equiv$)

example:

$$w : \overline{a} \mid v : a.\hat{}u : \overline{b} \quad \rightarrow \quad w : \overline{a} \mid v : a.u : \overline{b}\hat{} \mid u : \overline{b}$$

$$v : \overline{a} \mid w : \overline{a} \mid \hat{}a.u : \overline{b} \quad \nearrow \quad \searrow \quad v : \overline{a} \mid w : a.\hat{}u : \overline{b}$$

the algebra of reversible structures – reductions

input capture: $u : \overline{a} \mid \mathtt{A}^{\perp}.\hat{}a.\mathtt{B}.\overline{\mathtt{C}} \longrightarrow \mathtt{A}^{\perp}.u : a.\hat{}\mathtt{B}.\overline{\mathtt{C}}$

input release: $\mathtt{A}^{\perp}.u : a.\hat{}\mathtt{B}.\overline{\mathtt{C}} \longrightarrow u : \overline{a} \mid \mathtt{A}^{\perp}.\hat{}a.\mathtt{B}.\overline{\mathtt{C}}$

output release: $\mathtt{A}^{\perp}.\overline{\mathtt{B}}.\hat{}u : \overline{a}.\overline{\mathtt{C}} \longrightarrow u : \overline{a} \mid \mathtt{A}^{\perp}.\overline{\mathtt{B}}.u : \overline{a}.\hat{}\overline{\mathtt{C}}$

output capture: $u : \overline{a} \mid \mathtt{A}^{\perp}.\overline{\mathtt{B}}.u : \overline{a}.\hat{}\overline{\mathtt{C}} \longrightarrow \mathtt{A}^{\perp}.\overline{\mathtt{B}}.\hat{}u : \overline{a}.\overline{\mathtt{C}}$

(plus the standard contextual rules about **new**, $\mid$, and $\equiv$)

example:

$$v : \overline{a} \mid w : \overline{a} \mid \hat{}a.u : \overline{b} \;\nearrow\; w : \overline{a} \mid v : a.\hat{}u : \overline{b} \;\to\; w : \overline{a} \mid v : a.u : \overline{b}\hat{} \mid u : \overline{b}$$

$$v : \overline{a} \mid w : \overline{a} \mid \hat{}a.u : \overline{b} \;\searrow\; v : \overline{a} \mid w : a.\hat{}u : \overline{b}$$

weak coherence

a structure $\mathtt{S}$ *is weak coherent whenever in its normal form* $(\mathtt{new}\ \widetilde{a})\mathtt{S}'$*,*

- *ids are uniquely associated to names and co-names*
  (if $u : \alpha$ *and* $u : \alpha'$ *occur in* $\mathtt{S}'$ *then either* $\alpha = \alpha'$ *or* $\alpha = \overline{\alpha'}$*)*

proposition: weak coherent reversible strand algebra may be implemented into three domains DNA circuits

- the correspondence is consistent and complete wrt reductions

permutation equivalence: prelude

– derivations may have a lot of forward and backward reductions that continuously do and undo stuff:

$$v : \bar{a} \mid w : \bar{a} \mid \hat{}a.u : \bar{b} \longrightarrow w : \bar{a} \mid v : a.\hat{}u : \bar{b} \longrightarrow v : \bar{a} \mid w : \bar{a} \mid \hat{}a.u : \bar{b}$$

is computationally equivalent to the empty one

– the two derivations

$$\begin{aligned} v : \bar{a} \mid w : \bar{a} \mid \hat{}a.u : \bar{b} \mid \hat{}a.z : \bar{c} &\longrightarrow w : \bar{a} \mid v : a.\hat{}u : \bar{b} \mid \hat{}a.z : \bar{c} \\ &\longrightarrow v : a.\hat{}u : \bar{b} \mid w : a.\hat{}z : \bar{c} \\ &\longrightarrow v : \bar{a} \mid \hat{}a.u : \bar{b} \mid w : a.\hat{}z : \bar{c} \end{aligned}$$

and

$$\begin{aligned} v : \bar{a} \mid w : \bar{a} \mid \hat{}a.u : \bar{b} \mid \hat{}a.z : \bar{c} &\longrightarrow v : \bar{a} \mid \hat{}a.u : \bar{b} \mid w : a.\hat{}z : \bar{c} \\ &\longrightarrow v : a.\hat{}u : \bar{b} \mid w : a.\hat{}z : \bar{c} \\ &\longrightarrow v : \bar{a} \mid \hat{}a.u : \bar{b} \mid w : a.\hat{}z : \bar{c} \end{aligned}$$

are computationally equivalent as well

permutation equivalence: prelude

– derivations may have a lot of forward and backward reductions that continuously do and undo stuff:

$$v : \bar{a} \mid w : \bar{a} \mid \hat{}a.u : \bar{b} \longrightarrow w : \bar{a} \mid v : a.\hat{}u : \bar{b} \longrightarrow v : \bar{a} \mid w : \bar{a} \mid \hat{}a.u : \bar{b}$$

is computationally equivalent to the empty one

– the two derivations

$$\begin{aligned} v : \bar{a} \mid w : \bar{a} \mid \hat{}a.u : \bar{b} \mid \hat{}a.z : \bar{c} &\longrightarrow w : \bar{a} \mid v : a.\hat{}u : \bar{b} \mid \hat{}a.z : \bar{c} \\ &\longrightarrow v : a.\hat{}u : \bar{b} \mid w : a.\hat{}z : \bar{c} \\ &\longrightarrow v : \bar{a} \mid \hat{}a.u : \bar{b} \mid w : a.\hat{}z : \bar{c} \end{aligned}$$

and

$$\begin{aligned} v : \bar{a} \mid w : \bar{a} \mid \hat{}a.u : \bar{b} \mid \hat{}a.z : \bar{c} &\longrightarrow v : \bar{a} \mid \hat{}a.u : \bar{b} \mid w : a.\hat{}z : \bar{c} \\ &\longrightarrow v : a.\hat{}u : \bar{b} \mid w : a.\hat{}z : \bar{c} \\ &\longrightarrow v : \bar{a} \mid \hat{}a.u : \bar{b} \mid w : a.\hat{}z : \bar{c} \end{aligned}$$

are computationally equivalent as well

permutation equivalence: Lévy labels

define an equivalence identifying the above derivations

Lévy: use labels to address reactants

- the label of a signal $u : \bar{a}$ is u
- the label of a gate is
  the sequence of ids in the input part (if any)
  + the sequence of names without ids in the input part
  + the sequence of ids in the output part

permutation equivalence: Lévy labels

define an equivalence identifying the above derivations

Lévy: use labels to address reactants

- the label of a signal $u : \bar{a}$ is u
- the label of a gate is
  the sequence of ids in the input part (if any)
  + the sequence of names without ids in the input part
  + the sequence of ids in the output part

permutation equivalence: Lévy labels

define an equivalence identifying the above derivations

Lévy: use labels to address reactants

- the label of a signal $u : \bar{a}$ is u
- the label of a gate is
  the sequence of ids in the input part (if any)
  + the sequence of names without ids in the input part
  + the sequence of ids in the output part

permutation equivalence

$[\mu]^+$ = the *converse label of* μ in weak coherent structures

examples:

$$\begin{aligned} [u \mid v\hat{}a \bullet w]^+ &= \widetilde{v}u\hat{} \bullet \widetilde{w} \\ [vu\hat{} \bullet w]^+ &= u \mid v\hat{}a \bullet w \\ [vu \bullet \hat{}w]^+ &= u \mid vu \bullet w\hat{} \\ [u \mid vu \bullet w\hat{}]^+ &= vu \bullet \hat{}w \end{aligned}$$

permutation equivalence $\sim$ is the least equivalence relation between computations closed under composition and such that:

$$\begin{aligned} \mu;[\mu]^+ &\sim \varepsilon \\ \mu;\nu &\sim \nu;\mu \qquad \text{if } \mu \text{ and } \nu \text{ are coinitial and } \mu \cap \nu = \emptyset \end{aligned}$$

permutation equivalence

$[\mu]^+$ = the *converse label of* μ in weak coherent structures

examples:

$$\begin{aligned} [u \mid v\hat{}a \bullet w]^+ &= \widetilde{v}u\hat{} \bullet \widetilde{w} \\ [vu\hat{} \bullet w]^+ &= u \mid v\hat{}a \bullet w \\ [vu \bullet \hat{}w]^+ &= u \mid vu \bullet w\hat{} \\ [u \mid vu \bullet w\hat{}]^+ &= vu \bullet \hat{}w \end{aligned}$$

permutation equivalence $\sim$ is the least equivalence relation between computations closed under composition and such that:

$$\begin{aligned} \mu;[\mu]^+ &\sim \varepsilon \\ \mu;\nu &\sim \nu;\mu \qquad \textit{if} \quad \mu \textit{ and } \nu \textit{ are coinitial and } \quad \mu \cap \nu = \emptyset \end{aligned}$$

permutation equivalence: example

$$u : \bar{a} \mid \hat{}a.v : \bar{b} \mid u : a.\hat{}v : \bar{b}$$

$$u \mid \hat{}a \bullet v \swarrow \qquad\qquad \searrow u \bullet \hat{}v$$

$$u : a.\hat{}v : \bar{b} \mid u : a.\hat{}v : \bar{b} \qquad\qquad u : \bar{a} \mid \hat{}a.v : \bar{b} \mid u : a.v : \bar{b}\hat{} \mid v : \bar{b}$$

$$u \bullet \hat{}v \searrow \qquad\qquad \swarrow u \mid \hat{}a \bullet v$$

$$u : a.\hat{}v : \bar{b} \mid u : a.v : \bar{b}\hat{} \mid v : \bar{b}$$

$$\downarrow u\hat{} \bullet v$$

$$u : \bar{a} \mid \hat{}a.v : \bar{b} \mid u : a.v : \bar{b}\hat{} \mid v : \bar{b}$$

permutation equivalence: example

$$u : \bar{a} \mid \hat{}a.v : \bar{b} \mid u : a.\hat{}v : \bar{b}$$

$$u \mid \hat{}a \bullet v \swarrow \qquad\qquad \searrow u \bullet \hat{}v$$

$$u : a.\hat{}v : \bar{b} \mid u : a.\hat{}v : \bar{b} \qquad\qquad u : \bar{a} \mid \hat{}a.v : \bar{b} \mid u : a.v : \bar{b}\hat{} \mid v : \bar{b}$$

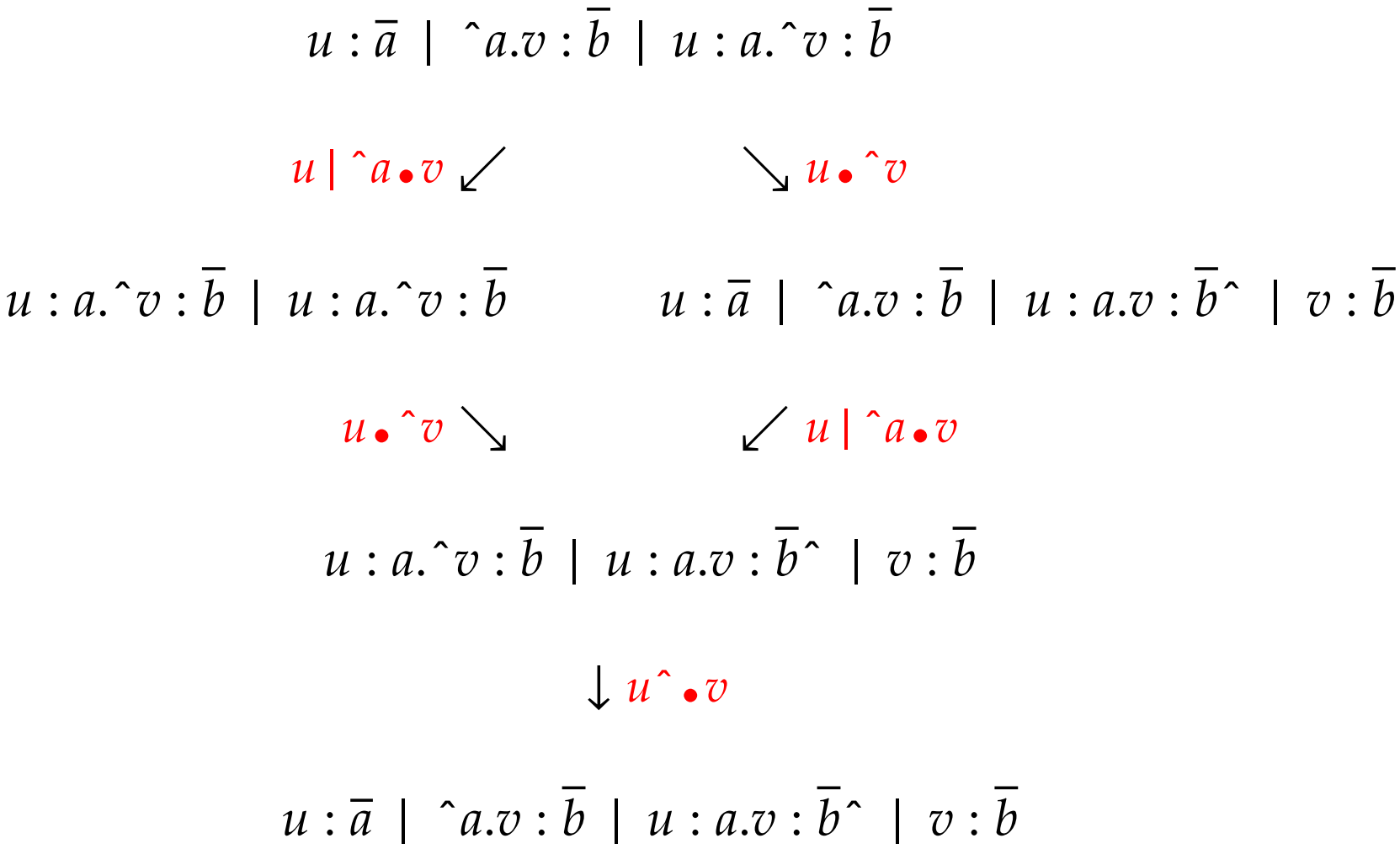
$$u \bullet \hat{}v \searrow \qquad\qquad \swarrow u \mid \hat{}a \bullet v$$

$$u : a.\hat{}v : \bar{b} \mid u : a.v : \bar{b}\hat{} \mid v : \bar{b}$$

$$\downarrow u\hat{} \bullet v$$

$$u : \bar{a} \mid \hat{}a.v : \bar{b} \mid u : a.v : \bar{b}\hat{} \mid v : \bar{b}$$

permutation equivalence: example



permutation equivalence: example

$$u : \bar{a} \mid \hat{}a.v : \bar{b} \mid u : a.\hat{}v : \bar{b}$$

$$u \mid \hat{}a \bullet v \swarrow \qquad \searrow u \bullet \hat{}v$$

$$u : a.\hat{}v : \bar{b} \mid u : a.\hat{}v : \bar{b} \qquad u : \bar{a} \mid \hat{}a.v : \bar{b} \mid u : a.v : \bar{b}\hat{} \mid v : \bar{b}$$

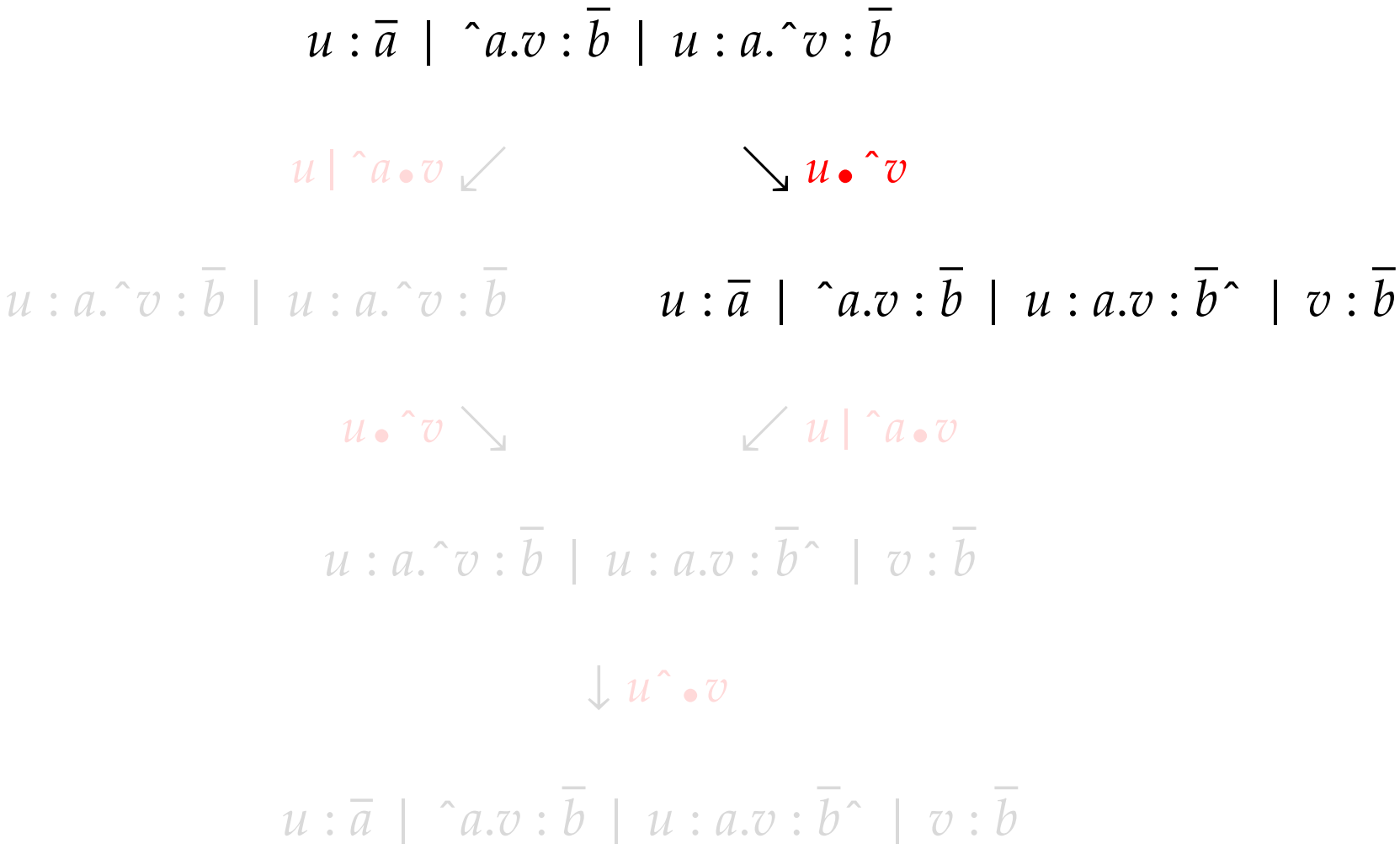
$$u \bullet \hat{}v \searrow \qquad \swarrow u \mid \hat{}a \bullet v$$

$$u : a.\hat{}v : \bar{b} \mid u : a.v : \bar{b}\hat{} \mid v : \bar{b}$$

$$\downarrow u\hat{} \bullet v$$

$$u : \bar{a} \mid \hat{}a.v : \bar{b} \mid u : a.v : \bar{b}\hat{} \mid v : \bar{b}$$

permutation equivalence: example



permutation equivalence: anomalies

$u : \bar{a} \mid u : \bar{a} \mid \hat{}a.v : \bar{b} \mid w : c.u : \bar{a}\hat{}$

$$u \mid \hat{}a \bullet v \longrightarrow \quad u : \bar{a} \mid u : a.\hat{}v : \bar{b} \mid w : c.u : \bar{a}$$

$$u \mid w \bullet u\hat{} \longrightarrow \quad u : a.\hat{}v : \bar{b} \mid w : c.\hat{}u : \bar{a}$$

$$u\hat{} \bullet v \longrightarrow \quad u : \bar{a} \mid \hat{}a.v : \bar{b} \mid w : c.\hat{}u : \bar{a}$$

cannot be simplified!

p.e. may not swap causally independent reductions

because labels are not expressive enough

- a similar anomaly is present in Petri nets (*cf.* Montanari)

rationale: in mass action systems, molecules have concentrations and two occurrences of a same molecule cannot be separated

permutation equivalence: anomalies

$$u : \bar{a} \mid u : \bar{a} \mid \hat{}a.v : \bar{b} \mid w : c.u : \bar{a}\hat{} \qquad \xrightarrow{u \mid \hat{}a \bullet v} \qquad u : \bar{a} \mid u : a.\hat{}v : \bar{b} \mid w : c.u : \bar{a}$$

$$\xrightarrow{u \mid w \bullet u\hat{}} \qquad u : a.\hat{}v : \bar{b} \mid w : c.\hat{}u : \bar{a}$$

$$\xrightarrow{u\hat{} \bullet v} \qquad u : \bar{a} \mid \hat{}a.v : \bar{b} \mid w : c.\hat{}u : \bar{a}$$

cannot be simplified!

p.e. may not swap causally independent reductions

because labels are not expressive enough

- a similar anomaly is present in Petri nets (*cf.* Montanari)

rationale: in mass action systems, molecules have concentrations and two occurrences of a same molecule cannot be separated

permutation equivalence: anomalies

$$u : \bar{a} \mid u : \bar{a} \mid \hat{}a.v : \bar{b} \mid w : c.u : \bar{a}\hat{} \quad \xrightarrow{u \mid \hat{}a \bullet v} \quad u : \bar{a} \mid u : a.\hat{}v : \bar{b} \mid w : c.u : \bar{a}$$

$$\xrightarrow{u \mid w \bullet u\hat{}} \quad u : a.\hat{}v : \bar{b} \mid w : c.\hat{}u : \bar{a}$$

$$\xrightarrow{u\hat{} \bullet v} \quad u : \bar{a} \mid \hat{}a.v : \bar{b} \mid w : c.\hat{}u : \bar{a}$$

cannot be simplified!

p.e. may not swap causally independent reductions

because labels are not expressive enough

- a similar anomaly is present in Petri nets (*cf.* Montanari)

rationale: in mass action systems, molecules have concentrations and two occurrences of a same molecule cannot be separated

permutation equivalence: anomalies

$$u : \bar{a} \mid u : \bar{a} \mid \hat{}a.v : \bar{b} \mid w : c.u : \bar{a}\hat{} \qquad \begin{array}{l} \xrightarrow{u \mid \hat{}a \bullet v} \\ \xrightarrow{u \mid w \bullet u\hat{}} \\ \xrightarrow{u\hat{} \bullet v} \end{array} \qquad \begin{array}{l} u : \bar{a} \mid u : a.\hat{}v : \bar{b} \mid w : c.u : \bar{a} \\ u : a.\hat{}v : \bar{b} \mid w : c.\hat{}u : \bar{a} \\ u : \bar{a} \mid \hat{}a.v : \bar{b} \mid w : c.\hat{}u : \bar{a} \end{array}$$

cannot be simplified!

p.e. may not swap causally independent reductions

because labels are not expressive enough

- a similar anomaly is present in Petri nets (*cf.* Montanari)

rationale: in mass action systems, molecules have concentrations and two occurrences of a same molecule cannot be separated

permutation equivalence: anomalies

$$u : \bar{a} \mid u : \bar{a} \mid \hat{}a.v : \bar{b} \mid w : c.u : \bar{a}\hat{} \qquad \begin{array}{ll} \xrightarrow{u \mid \hat{}a \bullet v} & u : \bar{a} \mid u : a.\hat{}v : \bar{b} \mid w : c.u : \bar{a} \\ \xrightarrow{u \mid w \bullet u\hat{}} & u : a.\hat{}v : \bar{b} \mid w : c.\hat{}u : \bar{a} \\ \xrightarrow{u\hat{} \bullet v} & u : \bar{a} \mid \hat{}a.v : \bar{b} \mid w : c.\hat{}u : \bar{a} \end{array}$$

cannot be simplified!

p.e. may not swap causally independent reductions

because labels are not expressive enough

- a similar anomaly is present in Petri nets (*cf.* Montanari)

rationale: in mass action systems, molecules have concentrations and two occurrences of a same molecule cannot be separated

permutation equivalence: anomalies

$$u : \bar{a} \mid u : \bar{a} \mid \hat{}a.v : \bar{b} \mid w : c.u : \bar{a}\hat{} \qquad \begin{array}{ll} \xrightarrow{u \mid \hat{}a \bullet v} & u : \bar{a} \mid u : a.\hat{}v : \bar{b} \mid w : c.u : \bar{a} \\ \xrightarrow{u \mid w \bullet u\hat{}} & u : a.\hat{}v : \bar{b} \mid w : c.\hat{}u : \bar{a} \\ \xrightarrow{u\hat{} \bullet v} & u : \bar{a} \mid \hat{}a.v : \bar{b} \mid w : c.\hat{}u : \bar{a} \end{array}$$

cannot be simplified!

p.e. may not swap causally independent reductions

because labels are not expressive enough

- a similar anomaly is present in Petri nets (*cf.* Montanari)

rationale: in mass action systems, molecules have concentrations and two occurrences of a same molecule cannot be separated

the standardization theorem

let $\mu_1 ; \cdots ; \mu_n$ be a computation of a weak coherent structure *such that* μ_n *is the converse of* μ_1

- there is a shorter computation that is permutation equivalent to $\mu_1 ; \cdots ; \mu_n$

the evolution of a gate in a derivation without converse labels (normal) is unidirectional

the standardization theorem

let $\mu_1 ; \cdots ; \mu_n$ be a computation of a weak coherent structure *such that* μ_n *is the converse of* μ_1

- there is a shorter computation that is permutation equivalent to $\mu_1 ; \cdots ; \mu_n$

the evolution of a gate in a derivation without converse labels (normal) is unidirectional

coherence

a solution must contain exactly one molecule of every species

a weak-coherent structure is *coherent* whenever

- different gates have types with no id in common – *the type of a gate is the sequence of ids in the output part*
- ids occur at most twice: one occurrence is positive and the other is negative

$$u_1 : a_1 . \cdots u_m : a_m . \quad \underbrace{v_1 : \overline{b_1} . \cdots . v_k : \overline{b_k}}_{\text{negative occurrences}} \quad .\hat{} v_{k+1} : \overline{b_{k+1}} . \cdots . v_n : \overline{b_n}$$

coherence

a solution must contain exactly one molecule of every species

a weak-coherent structure is *coherent* whenever

- different gates have types with no id in common – *the type of a gate is the sequence of ids in the output part*
- ids occur at most twice: one occurrence is positive and the other is negative

$$u_1 : a_1 . \cdots u_m : a_m . \quad \underbrace{v_1 : \overline{b_1} . \cdots . v_k : \overline{b_k}}_{\text{negative occurrences}} \quad .\hat{} \, v_{k+1} : \overline{b_{k+1}} . \cdots . v_n : \overline{b_n}$$

coherence and concurrency

coherence allows one to encode *in a causally consistent way* reversible process calculi

a first sign of this property:

theorem: *two coinitial computations of a coherent structure are permutation equivalent if and only if they are cofinal*

coherence and RCCS

coherent structures encode asynchronous Reversible CCS

example: the RCCS process $a.P + \bar{a}$ transitions are

$$a.P + \bar{a} \quad \begin{array}{l} \xleftrightarrow{m,a} \langle m, a, \bar{a} \rangle \triangleright P \\ \xleftrightarrow{m,\bar{a}} \langle m, \bar{a}, a.P \rangle \triangleright \mathbf{0} \end{array}$$

$a.P + \bar{a}$ is encoded as $[\![a.P + \bar{a}]\!]_c =$

$$(\mathtt{new}\ c', u, v)((\hat{}c.a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid \hat{}c.v : \bar{a})$$

$u' : \bar{c}$ ↙↗ ↘↖ $u' : \bar{c}$

$$(\mathtt{new}\ c', u, v)((\hat{}c.a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid u' : c.\hat{}v : \bar{a})$$

$$(\mathtt{new}\ c', u, v)((u' : c.\hat{}a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid \hat{}c.v : \bar{a})$$

remark: RCCS memories are (fine-grain) implemented by inactive processes

coherence and RCCS

coherent structures encode asynchronous Reversible CCS

example: the RCCS process $a.P + \bar{a}$ transitions are

$$a.P + \bar{a} \quad \begin{array}{l} \overset{m,a}{\longleftrightarrow} \langle m, a, \bar{a} \rangle \triangleright P \\ \overset{m,\bar{a}}{\longleftrightarrow} \langle m, \bar{a}, a.P \rangle \triangleright \mathbf{0} \end{array}$$

$a.P + \bar{a}$ is encoded as $[\![a.P + \bar{a}]\!]_c =$

$$(\mathtt{new}\ c', u, v)((\hat{}c.a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid \hat{}c.v : \bar{a})$$

$u' : \bar{c}$ ↙↗ ↘↖ $u' : \bar{c}$

$$(\mathtt{new}\ c', u, v)((\hat{}c.a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid u' : c.\hat{}v : \bar{a})$$

$$(\mathtt{new}\ c', u, v)((u' : c.\hat{}a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid \hat{}c.v : \bar{a})$$

remark: RCCS memories are (fine-grain) implemented by inactive processes

coherence and RCCS

coherent structures encode asynchronous Reversible CCS

example: the RCCS process $a.P + \bar{a}$ transitions are

$$a.P + \bar{a} \quad \begin{array}{l} \overset{m,a}{\longleftrightarrow} \langle m, a, \bar{a} \rangle \triangleright P \\ \overset{m,\bar{a}}{\longleftrightarrow} \langle m, \bar{a}, a.P \rangle \triangleright \mathbf{0} \end{array}$$

$a.P + \bar{a}$ is encoded as $[\![a.P + \bar{a}]\!]_c =$

$$(\mathtt{new}\ c', u, v)((\hat{}c.a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid \hat{}c.v : \bar{a})$$

$u' : \bar{c}$ ↙ ↗ $u' : \bar{c}$

$(\mathtt{new}\ c', u, v)((\hat{}c.a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid u' : c.\hat{}v : \bar{a})$

$(\mathtt{new}\ c', u, v)((u' : c.\hat{}a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid \hat{}c.v : \bar{a})$

remark: RCCS memories are (fine-grain) implemented by inactive processes

coherence and RCCS

coherent structures encode asynchronous Reversible CCS

example: the RCCS process $a.P + \bar{a}$ transitions are

$$a.P + \bar{a} \quad \begin{array}{l} \overset{m,a}{\longleftrightarrow} \langle m, a, \bar{a} \rangle \triangleright P \\ \overset{m,\bar{a}}{\longleftrightarrow} \langle m, \bar{a}, a.P \rangle \triangleright \mathbf{0} \end{array}$$

$a.P + \bar{a}$ is encoded as $[\![a.P + \bar{a}]\!]_c =$

$$(\mathtt{new}\ c', u, v)((\hat{}c.a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid \hat{}c.v : \bar{a})$$

$u' : \bar{c}$ ↙↗ ↘↖ $u' : \bar{c}$

$(\mathtt{new}\ c', u, v)((\hat{}c.a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid u' : c.\hat{}v : \bar{a})$

$(\mathtt{new}\ c', u, v)((u' : c.\hat{}a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid \hat{}c.v : \bar{a})$

remark: RCCS memories are (fine-grain) implemented by inactive processes

coherence and RCCS

coherent structures encode asynchronous Reversible CCS

example: the RCCS process $a.P + \bar{a}$ transitions are

$$a.P + \bar{a} \quad \begin{array}{l} \overset{m,a}{\longleftrightarrow} \langle m, a, \bar{a} \rangle \triangleright P \\ \overset{m,\bar{a}}{\longleftrightarrow} \langle m, \bar{a}, a.P \rangle \triangleright \mathbf{0} \end{array}$$

$a.P + \bar{a}$ is encoded as $[\![a.P + \bar{a}]\!]_c =$

$$(\mathtt{new}\ c', u, v)((\hat{}c.a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid \hat{}c.v : \bar{a})$$

$u' : \bar{c}$ ↙↗ ↘↖ $u' : \bar{c}$

$$(\mathtt{new}\ c', u, v)((\hat{}c.a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid u' : c.\hat{}v : \bar{a})$$

$$(\mathtt{new}\ c', u, v)((u' : c.\hat{}a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid \hat{}c.v : \bar{a})$$

remark: RCCS memories are (fine-grain) implemented by inactive processes

coherence and RCCS

coherent structures encode asynchronous Reversible CCS

example: the RCCS process $a.P + \bar{a}$ transitions are

$$a.P + \bar{a} \quad \begin{array}{l} \overset{m,a}{\longleftrightarrow} \langle m, a, \bar{a} \rangle \triangleright P \\ \overset{m,\bar{a}}{\longleftrightarrow} \langle m, \bar{a}, a.P \rangle \triangleright \mathbf{0} \end{array}$$

$a.P + \bar{a}$ is encoded as $[\![a.P + \bar{a}]\!]_c =$

$$(\mathtt{new}\ c', u, v)((\hat{}c.a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid \hat{}c.v : \bar{a})$$

$u' : \bar{c}$ ↙↗ ↘↖ $u' : \bar{c}$

$$(\mathtt{new}\ c', u, v)((\hat{}c.a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid u' : c.\hat{}v : \bar{a})$$

$$(\mathtt{new}\ c', u, v)((u' : c.\hat{}a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid \hat{}c.v : \bar{a})$$

remark: RCCS memories are (fine-grain) implemented by inactive processes

coherence and RCCS

coherent structures encode asynchronous Reversible CCS

example: the RCCS process $a.P + \bar{a}$ transitions are

$$a.P + \bar{a} \quad \begin{array}{l} \overset{m,a}{\longleftrightarrow} \langle m, a, \bar{a} \rangle \triangleright P \\ \overset{m,\bar{a}}{\longleftrightarrow} \langle m, \bar{a}, a.P \rangle \triangleright \mathbf{0} \end{array}$$

$a.P + \bar{a}$ is encoded as $[\![a.P + \bar{a}]\!]_c =$

$$(\mathtt{new}\ c', u, v)((\hat{}c.a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid \hat{}c.v : \bar{a})$$

$u' : \bar{c}$ ↙↗ ↘↖ $u' : \bar{c}$

$$(\mathtt{new}\ c', u, v)((\hat{}c.a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid u' : c.\hat{}v : \bar{a})$$

$$(\mathtt{new}\ c', u, v)((u' : c.\hat{}a.u : \overline{c'} \mid [\![P]\!]_{c'}) \mid \hat{}c.v : \bar{a})$$

remark: RCCS memories are (fine-grain) implemented by inactive processes

conclusions

possible research directions

- coherence is very hard to achieve in nature
  - + biology prompts a thorough study of reversible concurrent calculi where processes have multiplicities and the causal dependencies between copies may be exchanged
- reversible structures may be extended with irreversible combinators (that may be implemented in DNA)
  - + this makes possible to model standard irreversible operators of programming languages in DNA
- studying biological relevant problems in reversible structures may be simpler
  - + we already studied reachability; other issues are absence of molecules/processes, persistence of materials, $\cdots$

conclusions

possible research directions

- coherence is very hard to achieve in nature
  - + biology prompts a thorough study of reversible concurrent calculi where processes have multiplicities and the causal dependencies between copies may be exchanged
- reversible structures may be extended with irreversible combinators (that may be implemented in DNA)
  - + this makes possible to model standard irreversible operators of programming languages in DNA
- studying biological relevant problems in reversible structures may be simpler
  - + we already studied reachability; other issues are absence of molecules/processes, persistence of materials, $\cdots$

conclusions

possible research directions

- coherence is very hard to achieve in nature
  - + biology prompts a thorough study of reversible concurrent calculi where processes have multiplicities and the causal dependencies between copies may be exchanged
- reversible structures may be extended with irreversible combinators (that may be implemented in DNA)
  - + this makes possible to model standard irreversible operators of programming languages in DNA
- studying biological relevant problems in reversible structures may be simpler
  - + we already studied reachability; other issues are absence of molecules/processes, persistence of materials, $\cdots$

conclusions

possible research directions

- coherence is very hard to achieve in nature
  - + biology prompts a thorough study of reversible concurrent calculi where processes have multiplicities and the causal dependencies between copies may be exchanged
- reversible structures may be extended with irreversible combinators (that may be implemented in DNA)
  - + this makes possible to model standard irreversible operators of programming languages in DNA
- studying biological relevant problems in reversible structures may be simpler
  - + we already studied reachability; other issues are absence of molecules/processes, persistence of materials, $\cdots$