Simulations as a genuinely categorical concept

Jürgen Koslowski

Department of Theoretical Computer Science
Technical University Braunschweig

CT 2006, White Point, Nova Scotia, 2006-06-30

(last updated 2006-07-11)

http://www.iti.cs.tu-bs.de/~koslowj/RESEARCH

01. 3-fold motivation: structure-preserving relations?

Why is it that in concrete categories (over set) almost always "structure-preserving" functions are employed as morphisms?

Structure-preserving relations occur rather seldom, even if the structure is given by relations. Some notable exceptions:

- various partial homomorphisms between partial algebras;
- in CS relations are employed, whenever determinacy and/or termination may be in question; often in an ad hoc fashion;
- order-ideals between pre-ordered sets; this is a particular instance of the notion of profunctor;
- simulations between labeled transition systems (LTSs) in CS; usually these are regarded as a mere stepping stone towards bisimulation equivalence and seldom viewed as morphisms in their own right.

Could simulations be the "right" notion of morphism in "relational algebra" and possibly beyond?

01. 3-fold motivation: structure-preserving relations?

Why is it that in concrete categories (over set) almost always "structure-preserving" functions are employed as morphisms? Structure-preserving relations occur rather seldom, even if the structure is given by relations. Some notable exceptions:

- various partial homomorphisms between partial algebras;
- in CS relations are employed, whenever determinacy and/or termination may be in question; often in an ad hoc fashion;
- order-ideals between pre-ordered sets; this is a particular instance of the notion of profunctor;
- simulations between labeled transition systems (LTSs) in CS; usually these are regarded as a mere stepping stone towards bisimulation equivalence and seldom viewed as morphisms in their own right.

Could simulations be the "right" notion of morphism in "relational algebra" and possibly beyond?

01. 3-fold motivation: structure-preserving relations?

Why is it that in concrete categories (over set) almost always "structure-preserving" functions are employed as morphisms? Structure-preserving relations occur rather seldom, even if the structure is given by relations. Some notable exceptions:

- various partial homomorphisms between partial algebras;
- in CS relations are employed, whenever determinacy and/or termination may be in question; often in an ad hoc fashion;
- order-ideals between pre-ordered sets; this is a particular instance of the notion of profunctor;
- simulations between labeled transition systems (LTSs) in CS; usually these are regarded as a mere stepping stone towards bisimulation equivalence and seldom viewed as morphisms in their own right.

Could simulations be the "right" notion of morphism in "relational algebra" and possibly beyond?

01. 3-fold motivation: structure-preserving relations?

Why is it that in concrete categories (over set) almost always "structure-preserving" functions are employed as morphisms? Structure-preserving relations occur rather seldom, even if the structure is given by relations. Some notable exceptions:

- various partial homomorphisms between partial algebras;
- in CS relations are employed, whenever determinacy and/or termination may be in question; often in an ad hoc fashion;
- order-ideals between pre-ordered sets; this is a particular instance of the notion of profunctor;
- simulations between labeled transition systems (LTSs) in CS; usually these are regarded as a mere stepping stone towards bisimulation equivalence and seldom viewed as morphisms in their own right.

Could simulations be the "right" notion of morphism in "relational algebra" and possibly beyond?

01. 3-fold motivation: structure-preserving relations?

Why is it that in concrete categories (over set) almost always "structure-preserving" functions are employed as morphisms? Structure-preserving relations occur rather seldom, even if the structure is given by relations. Some notable exceptions:

- various partial homomorphisms between partial algebras;
- in CS relations are employed, whenever determinacy and/or termination may be in question; often in an ad hoc fashion;
- order-ideals between pre-ordered sets; this is a particular instance of the notion of profunctor;
- simulations between labeled transition systems (LTSs) in CS; usually these are regarded as a mere stepping stone towards bisimulation equivalence and seldom viewed as morphisms in their own right.

Could simulations be the "right" notion of morphism in "relational algebra" and possibly beyond?

01. 3-fold motivation: structure-preserving relations?

Why is it that in concrete categories (over set) almost always "structure-preserving" functions are employed as morphisms? Structure-preserving relations occur rather seldom, even if the structure is given by relations. Some notable exceptions:

- various partial homomorphisms between partial algebras;
- in CS relations are employed, whenever determinacy and/or termination may be in question; often in an ad hoc fashion;
- order-ideals between pre-ordered sets; this is a particular instance of the notion of profunctor;
- simulations between labeled transition systems (LTSs) in CS; usually these are regarded as a mere stepping stone towards bisimulation equivalence and seldom viewed as morphisms in their own right.

Could simulations be the "right" notion of morphism in "relational algebra" and possibly beyond?

01. 3-fold motivation: structure-preserving relations?

Why is it that in concrete categories (over set) almost always "structure-preserving" functions are employed as morphisms? Structure-preserving relations occur rather seldom, even if the structure is given by relations. Some notable exceptions:

- various partial homomorphisms between partial algebras;
- in CS relations are employed, whenever determinacy and/or termination may be in question; often in an ad hoc fashion;
- order-ideals between pre-ordered sets; this is a particular instance of the notion of profunctor;
- simulations between labeled transition systems (LTSs) in CS; usually these are regarded as a mere stepping stone towards bisimulation equivalence and seldom viewed as morphisms in their own right.

Could simulations be the "right" notion of morphism in "relational algebra" and possibly beyond?

01. 3-fold motivation: structure-preserving relations?

Why is it that in concrete categories (over set) almost always "structure-preserving" functions are employed as morphisms? Structure-preserving relations occur rather seldom, even if the structure is given by relations. Some notable exceptions:

- various partial homomorphisms between partial algebras;
- in CS relations are employed, whenever determinacy and/or termination may be in question; often in an ad hoc fashion;
- order-ideals between pre-ordered sets; this is a particular instance of the notion of profunctor;
- simulations between labeled transition systems (LTSs) in CS; usually these are regarded as a mere stepping stone towards bisimulation equivalence and seldom viewed as morphisms in their own right.

Could simulations be the "right" notion of morphism in "relational algebra" and possibly beyond?

01. 3-fold motivation: structure-preserving relations?

Why is it that in concrete categories (over set) almost always "structure-preserving" functions are employed as morphisms? Structure-preserving relations occur rather seldom, even if the structure is given by relations. Some notable exceptions:

- various partial homomorphisms between partial algebras;
- in CS relations are employed, whenever determinacy and/or termination may be in question; often in an ad hoc fashion;
- order-ideals between pre-ordered sets; this is a particular instance of the notion of profunctor;
- simulations between labeled transition systems (LTSs) in CS; usually these are regarded as a mere stepping stone towards bisimulation equivalence and seldom viewed as morphisms in their own right.

Could simulations be the "right" notion of morphism in "relational algebra" and possibly beyond?

01. 3-fold motivation: structure-preserving relations?

Why is it that in concrete categories (over set) almost always "structure-preserving" functions are employed as morphisms? Structure-preserving relations occur rather seldom, even if the structure is given by relations. Some notable exceptions:

- various partial homomorphisms between partial algebras;
- in CS relations are employed, whenever determinacy and/or termination may be in question; often in an ad hoc fashion;
- order-ideals between pre-ordered sets; this is a particular instance of the notion of profunctor;
- simulations between labeled transition systems (LTSs) in CS; usually these are regarded as a mere stepping stone towards bisimulation equivalence and seldom viewed as morphisms in their own right.

Could simulations be the "right" notion of morphism in "relational algebra" and possibly beyond?

02. 3-fold motivation: what are simulations anyway?

When trying to understand (bi)simulations, you will find

- that Park's 1981 rather operational approach (with "silent transitions" intended to break synchronization) is favoured in CS over Yoeli and Ginzburg's conceptual notion of $\leq$ 1965;
- that coalgebra, initiated by Aczel and Mendler [AM89], until recently was focussed almost entirely on bisimulations;
- that the synthetic theory of (bi)simulations via open maps, as pioneered by Joyal, Nielsen and Winskel [JNW94], or via Cockett and Spooner's covering morphisms [CS97], downplays the 2-dimensional heritage of the notion (just as coalgebra);
- other sources of inspiration, like an intriguing remark by Dusko Pavlović [AP97], Lindsay Errington's thesis [Err99], and a 2002 talk by Krzysztof Worytkiewicz in Ottawa [Wor03].

Sorting out the various angles from which to describe LTSs leads to graph comprehension as our main tool to address simulations.

02. 3-fold motivation: what are simulations anyway?

When trying to understand (bi)simulations, you will find

- that Park's 1981 rather operational approach (with "silent transitions" intended to break synchronization) is favoured in CS over Yoeli and Ginzburg's conceptual notion of $\leq$ 1965;
- that coalgebra, initiated by Aczel and Mendler [AM89], until recently was focussed almost entirely on bisimulations;
- that the synthetic theory of (bi)simulations via open maps, as pioneered by Joyal, Nielsen and Winskel [JNW94], or via Cockett and Spooner's covering morphisms [CS97], downplays the 2-dimensional heritage of the notion (just as coalgebra);
- other sources of inspiration, like an intriguing remark by Dusko Pavlović [AP97], Lindsay Errington's thesis [Err99], and a 2002 talk by Krzysztof Worytkiewicz in Ottawa [Wor03].

Sorting out the various angles from which to describe LTSs leads to graph comprehension as our main tool to address simulations.

02. 3-fold motivation: what are simulations anyway?

When trying to understand (bi)simulations, you will find

- that Park's 1981 rather operational approach (with "silent transitions" intended to break synchronization) is favoured in CS over Yoeli and Ginzburg's conceptual notion of $\leq$ 1965;
- that coalgebra, initiated by Aczel and Mendler [AM89], until recently was focussed almost entirely on bisimulations;
- that the synthetic theory of (bi)simulations via open maps, as pioneered by Joyal, Nielsen and Winskel [JNW94], or via Cockett and Spooner's covering morphisms [CS97], downplays the 2-dimensional heritage of the notion (just as coalgebra);
- other sources of inspiration, like an intriguing remark by Dusko Pavlović [AP97], Lindsay Errington's thesis [Err99], and a 2002 talk by Krzysztof Worytkiewicz in Ottawa [Wor03].

Sorting out the various angles from which to describe LTSs leads to graph comprehension as our main tool to address simulations.

02. 3-fold motivation: what are simulations anyway?

When trying to understand (bi)simulations, you will find

- that Park's 1981 rather operational approach (with "silent transitions" intended to break synchronization) is favoured in CS over Yoeli and Ginzburg's conceptual notion of $\leq$ 1965;
- that coalgebra, initiated by Aczel and Mendler [AM89], until recently was focussed almost entirely on bisimulations;
- that the synthetic theory of (bi)simulations via open maps, as pioneered by Joyal, Nielsen and Winskel [JNW94], or via Cockett and Spooner's covering morphisms [CS97], downplays the 2-dimensional heritage of the notion (just as coalgebra);
- other sources of inspiration, like an intriguing remark by Dusko Pavlović [AP97], Lindsay Errington's thesis [Err99], and a 2002 talk by Krzysztof Worytkiewicz in Ottawa [Wor03].

Sorting out the various angles from which to describe LTSs leads to graph comprehension as our main tool to address simulations.

02. 3-fold motivation: what are simulations anyway?

When trying to understand (bi)simulations, you will find

- that Park's 1981 rather operational approach (with "silent transitions" intended to break synchronization) is favoured in CS over Yoeli and Ginzburg's conceptual notion of $\leq$ 1965;
- that coalgebra, initiated by Aczel and Mendler [AM89], until recently was focussed almost entirely on bisimulations;
- that the synthetic theory of (bi)simulations via open maps, as pioneered by Joyal, Nielsen and Winskel [JNW94], or via Cockett and Spooner's covering morphisms [CS97], downplays the 2-dimensional heritage of the notion (just as coalgebra);
- other sources of inspiration, like an intriguing remark by Dusko Pavlović [AP97], Lindsay Errington's thesis [Err99], and a 2002 talk by Krzysztof Worytkiewicz in Ottawa [Wor03].

Sorting out the various angles from which to describe LTSs leads to graph comprehension as our main tool to address simulations.

02. 3-fold motivation: what are simulations anyway?

When trying to understand (bi)simulations, you will find

- that Park's 1981 rather operational approach (with "silent transitions" intended to break synchronization) is favoured in CS over Yoeli and Ginzburg's conceptual notion of $\leq$ 1965;
- that coalgebra, initiated by Aczel and Mendler [AM89], until recently was focussed almost entirely on bisimulations;
- that the synthetic theory of (bi)simulations via open maps, as pioneered by Joyal, Nielsen and Winskel [JNW94], or via Cockett and Spooner's covering morphisms [CS97], downplays the 2-dimensional heritage of the notion (just as coalgebra);
- other sources of inspiration, like an intriguing remark by Dusko Pavlović [AP97], Lindsay Errington's thesis [Err99], and a 2002 talk by Krzysztof Worytkiewicz in Ottawa [Wor03].

Sorting out the various angles from which to describe LTSs leads to graph comprehension as our main tool to address simulations.

03. 3-fold motivation: functors vs. profunctors

The general theory of modules

- not only explains the connection between profunctors and functors for ordinary categories (and in particular pre-ordered sets), and addresses the compositional shortcomings of (op)lax natural transformations (*cf.*, joint work with Robin Cockett, Robert Seely and Richard Wood) [CKSW03],
- it also works for categories enriched over a bicategory $\mathcal{W}$,
- and moreover for weaker notions than categories, *e.g.*, taxons;

For $\mathcal{W} = \boldsymbol{rel}$ or $\mathcal{W} = \boldsymbol{spn}$ graph comprehension will put us in the same "ball park" and will allow us to treat simulations and (pro)functors on equal footing.

03. 3-fold motivation: functors vs. profunctors

The general theory of modules

- not only explains the connection between profunctors and functors for ordinary categories (and in particular pre-ordered sets), and addresses the compositional shortcomings of (op)lax natural transformations (*cf.*, joint work with Robin Cockett, Robert Seely and Richard Wood) [CKSW03],
- it also works for categories enriched over a bicategory $\mathcal{W}$,
- and moreover for weaker notions than categories, *e.g.*, taxons;

For $\mathcal{W} = \boldsymbol{rel}$ or $\mathcal{W} = \boldsymbol{spn}$ graph comprehension will put us in the same "ball park" and will allow us to treat simulations and (pro)functors on equal footing.

03. 3-fold motivation: functors vs. profunctors

The general theory of modules

- not only explains the connection between profunctors and functors for ordinary categories (and in particular pre-ordered sets), and addresses the compositional shortcomings of (op)lax natural transformations (*cf.*, joint work with Robin Cockett, Robert Seely and Richard Wood) [CKSW03],
- it also works for categories enriched over a bicategory $\mathcal{W}$,
- and moreover for weaker notions than categories, *e.g.*, taxons;

For $\mathcal{W} = \boldsymbol{rel}$ or $\mathcal{W} = \boldsymbol{spn}$ graph comprehension will put us in the same "ball park" and will allow us to treat simulations and (pro)functors on equal footing.

03. 3-fold motivation: functors vs. profunctors

The general theory of modules

- not only explains the connection between profunctors and functors for ordinary categories (and in particular pre-ordered sets), and addresses the compositional shortcomings of (op)lax natural transformations (*cf.*, joint work with Robin Cockett, Robert Seely and Richard Wood) [CKSW03],
- it also works for categories enriched over a bicategory $\mathcal{W}$,
- and moreover for weaker notions than categories, *e.g.*, taxons;

For $\mathcal{W} = \boldsymbol{rel}$ or $\mathcal{W} = \boldsymbol{spn}$ graph comprehension will put us in the same "ball park" and will allow us to treat simulations and (pro)functors on equal footing.

03. 3-fold motivation: functors vs. profunctors

The general theory of modules

- not only explains the connection between profunctors and functors for ordinary categories (and in particular pre-ordered sets), and addresses the compositional shortcomings of (op)lax natural transformations (*cf.*, joint work with Robin Cockett, Robert Seely and Richard Wood) [CKSW03],
- it also works for categories enriched over a bicategory $\mathcal{W}$,
- and moreover for weaker notions than categories, *e.g.*, taxons;

For $\mathcal{W} = \boldsymbol{rel}$ or $\mathcal{W} = \boldsymbol{spn}$ graph comprehension will put us in the same "ball park" and will allow us to treat simulations and (pro)functors on equal footing.

04. Labeled transition systems, constrained

Traditional labeled transition systems (LTSs) over a *label set* X are not allowed to have repeated labels along parallel arrows:

$$\begin{array}{cl}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(faithful graph morphism)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \text{(jointly mono)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \text{(jointly mono)} \\ \hline\hline
 & \\ \hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel} & \text{(graph morphism)}
\end{array}$$

where $\boldsymbol{Q} = (Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0)$ is a graph and $\boldsymbol{X} = (X \overset{!}{\underset{!}{\rightrightarrows}} 1)$ is a single-node graph with arrow-set X.

If $\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel}$ factors through $\boldsymbol{set}$, the LTS is called deterministic. Then the graph morphism $\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X}$ is a discrete opfibration.

04. Labeled transition systems, constrained

Traditional labeled transition systems (LTSs) over a *label set* X are not allowed to have repeated labels along parallel arrows:

$$\begin{array}{cl}
Q \xrightarrow{\langle !,\ell\rangle} X & \text{(faithful graph morphism)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \text{(jointly mono)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \text{(jointly mono)} \\ \hline\hline
 & \\ \hline\hline
X \xrightarrow{L} \mathit{rel} & \text{(graph morphism)}
\end{array}$$

where $Q = (Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0)$ is a graph and $X = (X \overset{!}{\underset{!}{\rightrightarrows}} 1)$ is a single-node graph with arrow-set X.

If $X \xrightarrow{L} \mathit{rel}$ factors through set, the LTS is called deterministic. Then the graph morphism $Q \xrightarrow{\langle !,\ell\rangle} X$ is a discrete opfibration.

04. Labeled transition systems, constrained

Traditional labeled transition systems (LTSs) over a *label set* X are not allowed to have repeated labels along parallel arrows:

$$\begin{array}{cl}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(faithful graph morphism)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \text{(jointly mono)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \text{(jointly mono)} \\ \hline\hline
 & \\ \hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel} & \text{(graph morphism)}
\end{array}$$

where $\boldsymbol{Q} = (Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0)$ is a graph and $\boldsymbol{X} = (X \overset{!}{\underset{!}{\rightrightarrows}} 1)$ is a single-node graph with arrow-set X .

If $\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel}$ factors through $\boldsymbol{set}$, the LTS is called deterministic. Then the graph morphism $\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X}$ is a discrete opfibration.

04. Labeled transition systems, constrained

Traditional labeled transition systems (LTSs) over a *label set* X are not allowed to have repeated labels along parallel arrows:

$$\begin{array}{cl}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(faithful graph morphism)} \\
\hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \text{(jointly mono)} \\
\hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \text{(jointly mono)} \\
\hline\hline
 & \\
\hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel} & \text{(graph morphism)}
\end{array}$$

where $\boldsymbol{Q} = (Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0)$ is a graph and $\boldsymbol{X} = (X \overset{!}{\underset{!}{\rightrightarrows}} 1)$ is a single-node graph with arrow-set X .

If $\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel}$ factors through $\boldsymbol{set}$, the LTS is called deterministic. Then the graph morphism $\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X}$ is a discrete opfibration.

04. Labeled transition systems, constrained

Traditional labeled transition systems (LTSs) over a *label set* X are not allowed to have repeated labels along parallel arrows:

$$\begin{array}{cl}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(faithful graph morphism)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \underset{t}{\overset{s}{\rightrightarrows}} Q_0 & \text{(jointly mono)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \text{(jointly mono)} \\ \hline\hline
Q_0 \times X \xrightarrow{L} Q_0 & \text{(textbook automaton?)} \\ \hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel} & \text{(graph morphism)}
\end{array}$$

where $\boldsymbol{Q} = (Q_1 \underset{t}{\overset{s}{\rightrightarrows}} Q_0)$ is a graph and $\boldsymbol{X} = (X \underset{!}{\overset{!}{\rightrightarrows}} 1)$ is a single-node graph with arrow-set X.

If $\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel}$ factors through $\boldsymbol{set}$, the LTS is called deterministic. Then the graph morphism $\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X}$ is a discrete opfibration.

04. Labeled transition systems, constrained

Traditional labeled transition systems (LTSs) over a *label set* X are not allowed to have repeated labels along parallel arrows:

$$\begin{array}{cl}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(faithful graph morphism)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \text{(jointly mono)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \text{(jointly mono)} \\ \hline\hline
Q_0 \times X \xrightarrow{L} Q_0\mathcal{P} & \text{(textbook automaton!)} \\ \hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel} & \text{(graph morphism)}
\end{array}$$

where $\boldsymbol{Q} = (Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0)$ is a graph and $\boldsymbol{X} = (X \overset{!}{\underset{!}{\rightrightarrows}} 1)$ is a single-node graph with arrow-set X .

If $\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel}$ factors through $\boldsymbol{set}$, the LTS is called deterministic. Then the graph morphism $\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X}$ is a discrete opfibration.

04. Labeled transition systems, constrained

Traditional labeled transition systems (LTSs) over a *label set* X are not allowed to have repeated labels along parallel arrows:

$$\begin{array}{cl}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(faithful graph morphism)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \underset{t}{\overset{s}{\rightrightarrows}} Q_0 & \text{(jointly mono)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \text{(jointly mono)} \\ \hline\hline
Q_0 \xrightarrow{L} X \times Q_0 & \text{(non-obvious relation)} \\ \hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel} & \text{(graph morphism)}
\end{array}$$

where $\boldsymbol{Q} = (Q_1 \underset{t}{\overset{s}{\rightrightarrows}} Q_0)$ is a graph and $\boldsymbol{X} = (X \underset{!}{\overset{!}{\rightrightarrows}} 1)$ is a single-node graph with arrow-set X.

If $\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel}$ factors through $\boldsymbol{set}$, the LTS is called deterministic. Then the graph morphism $\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X}$ is a discrete opfibration.

04. Labeled transition systems, constrained

Traditional labeled transition systems (LTSs) over a *label set* X are not allowed to have repeated labels along parallel arrows:

$$\begin{array}{cc}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(faithful graph morphism)} \\
\hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \text{(jointly mono)} \\
\hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \text{(jointly mono)} \\
\hline\hline
Q_0 \xrightarrow{L} (X \times Q_0)\mathcal{P} & \text{(coalgebra)} \\
\hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel} & \text{(graph morphism)}
\end{array}$$

where $\boldsymbol{Q} = (Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0)$ is a graph and $\boldsymbol{X} = (X \overset{!}{\underset{!}{\rightrightarrows}} 1)$ is a single-node graph with arrow-set X .

If $\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel}$ factors through $\boldsymbol{set}$, the LTS is called deterministic. Then the graph morphism $\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X}$ is a discrete opfibration.

04. Labeled transition systems, constrained

Traditional labeled transition systems (LTSs) over a *label set* X are not allowed to have repeated labels along parallel arrows:

$$\begin{array}{cl}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(faithful graph morphism)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \underset{t}{\overset{s}{\rightrightarrows}} Q_0 & \text{(jointly mono)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \text{(jointly mono)} \\ \hline\hline
X \overset{L}{\longrightarrow\!\!\triangleright} Q_0 \times Q_0 & \text{(obvious relation)} \\ \hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel} & \text{(graph morphism)}
\end{array}$$

where $\boldsymbol{Q} = (Q_1 \underset{t}{\overset{s}{\rightrightarrows}} Q_0)$ is a graph and $\boldsymbol{X} = (X \underset{!}{\overset{!}{\rightrightarrows}} 1)$ is a single-node graph with arrow-set X .

If $\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel}$ factors through $\boldsymbol{set}$, the LTS is called deterministic. Then the graph morphism $\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X}$ is a discrete opfibration.

04. Labeled transition systems, constrained

Traditional labeled transition systems (LTSs) over a *label set* X are not allowed to have repeated labels along parallel arrows:

$$\begin{array}{cl}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(faithful graph morphism)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \underset{t}{\overset{s}{\rightrightarrows}} Q_0 & \text{(jointly mono)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \text{(jointly mono)} \\ \hline\hline
X \xrightarrow{L} (Q_0 \times Q_0)\mathcal{P} & \text{(this looks promising!)} \\ \hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel} & \text{(graph morphism)}
\end{array}$$

where $\boldsymbol{Q} = (Q_1 \underset{t}{\overset{s}{\rightrightarrows}} Q_0)$ is a graph and $\boldsymbol{X} = (X \underset{!}{\overset{!}{\rightrightarrows}} 1)$ is a single-node graph with arrow-set X.

If $\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel}$ factors through $\boldsymbol{set}$, the LTS is called deterministic. Then the graph morphism $\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X}$ is a discrete opfibration.

04. Labeled transition systems, constrained

Traditional labeled transition systems (LTSs) over a *label set* X are not allowed to have repeated labels along parallel arrows:

$$\begin{array}{cl}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(faithful graph morphism)} \\
\hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \text{(jointly mono)} \\
\hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \text{(jointly mono)} \\
\hline\hline
X \xrightarrow{L} (Q_0, Q_0)\boldsymbol{rel} & \\
\hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel} & \text{(graph morphism)}
\end{array}$$

where $\boldsymbol{Q} = (Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0)$ is a graph and $\boldsymbol{X} = (X \overset{!}{\underset{!}{\rightrightarrows}} 1)$ is a single-node graph with arrow-set X.

If $\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel}$ factors through $\boldsymbol{set}$, the LTS is called deterministic. Then the graph morphism $\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X}$ is a discrete opfibration.

04. Labeled transition systems, constrained

Traditional labeled transition systems (LTSs) over a *label set* X are not allowed to have repeated labels along parallel arrows:

$$\begin{array}{cl}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(faithful graph morphism)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \text{(jointly mono)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \text{(jointly mono)} \\ \hline\hline
X \xrightarrow{L} (Q_0, Q_0)\boldsymbol{rel} & \\ \hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel} & \text{(graph morphism)}
\end{array}$$

where $\boldsymbol{Q} = (Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0)$ is a graph and $\boldsymbol{X} = (X \overset{!}{\underset{!}{\rightrightarrows}} 1)$ is a single-node graph with arrow-set X.

If $\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel}$ factors through $\boldsymbol{set}$, the LTS is called deterministic. Then the graph morphism $\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X}$ is a discrete opfibration.

04. Labeled transition systems, constrained

Traditional labeled transition systems (LTSs) over a *label set* X are not allowed to have repeated labels along parallel arrows:

$$\begin{array}{cl}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(faithful graph morphism)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \text{(jointly mono)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \text{(jointly mono)} \\ \hline\hline
X \xrightarrow{L} (Q_0, Q_0)\boldsymbol{rel} & \\ \hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel} & \text{(graph morphism)}
\end{array}$$

where $\boldsymbol{Q} = (Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0)$ is a graph and $\boldsymbol{X} = (X \overset{!}{\underset{!}{\rightrightarrows}} 1)$ is a single-node graph with arrow-set X.

If $\boldsymbol{X} \xrightarrow{L} \boldsymbol{rel}$ factors through $\boldsymbol{set}$, the LTS is called deterministic. Then the graph morphism $\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X}$ is a discrete opfibration.

05. Labeled transition systems, unconstrained

Dropping the constraint that parallel arrows must have different labels yields a similar bijective correspondence

$$\begin{array}{c}
Q \xrightarrow{\langle !,\ell\rangle} X \qquad \text{(graph morphism)} \\
\hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 \\
\hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 \\
\hline\hline
X \xrightarrow{L} \langle Q_0, Q_0\rangle\, spn \\
\hline\hline
X \xrightarrow{L} spn \qquad \text{(graph morphism)}
\end{array}$$

Of course, the bijective correspondence between X-controlled processes and X-controlled systems does *not* depend on X having just a single node. In fact, multi-sorted control can be useful for implementing certain features.

05. Labeled transition systems, unconstrained

Dropping the constraint that parallel arrows must have different labels yields a similar bijective correspondence

$$\begin{array}{cl} Q \xrightarrow{\langle !,\ell\rangle} X & \text{(graph morphism)} \\ \hline\hline X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \\ \hline\hline X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \\ \hline\hline X \xrightarrow{L} \langle Q_0, Q_0\rangle spn & \\ \hline\hline X \xrightarrow{L} spn & \text{(graph morphism)} \end{array}$$

Of course, the bijective correspondence between X-controlled processes and X-controlled systems does *not* depend on X having just a single node. In fact, multi-sorted control can be useful for implementing certain features.

05. Labeled transition systems, unconstrained

Dropping the constraint that parallel arrows must have different labels yields a similar bijective correspondence

$$\begin{array}{cl}
Q \xrightarrow{\langle !,\ell\rangle} X & \text{(graph morphism)} \\
\hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \\
\hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \\
\hline\hline
X \xrightarrow{L} \langle Q_0, Q_0\rangle spn & \\
\hline\hline
X \xrightarrow{L} spn & \text{(graph morphism)}
\end{array}$$

Of course, the bijective correspondence between X-controlled processes and X-controlled systems does *not* depend on X having just a single node. In fact, multi-sorted control can be useful for implementing certain features.

05. Labeled transition systems, unconstrained

Dropping the constraint that parallel arrows must have different labels yields a similar bijective correspondence

$$\begin{array}{cl}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(graph morphism)} \\
\hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \\
\hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \\
\hline\hline
X \xrightarrow{L} \langle Q_0, Q_0 \rangle \boldsymbol{spn} & \\
\hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{spn} & \text{(graph morphism)}
\end{array}$$

Of course, the bijective correspondence between $\boldsymbol{X}$-controlled processes and $\boldsymbol{X}$-controlled systems does *not* depend on $\boldsymbol{X}$ having just a single node. In fact, multi-sorted control can be useful for implementing certain features.

05. Labeled transition systems, unconstrained

Dropping the constraint that parallel arrows must have different labels yields a similar bijective correspondence

$$\begin{array}{cl}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(graph morphism)} \\
\hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \\
\hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \\
\hline\hline
X \xrightarrow{L} \langle Q_0, Q_0\rangle \boldsymbol{spn} & \\
\hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{spn} & \text{(graph morphism)}
\end{array}$$

Of course, the bijective correspondence between $\boldsymbol{X}$-controlled processes and $\boldsymbol{X}$-controlled systems does *not* depend on $\boldsymbol{X}$ having just a single node. In fact, multi-sorted control can be useful for implementing certain features.

05. Labeled transition systems, unconstrained

Dropping the constraint that parallel arrows must have different labels yields a similar bijective correspondence

$$\begin{array}{cl}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(graph morphism)} \\
\hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \\
\hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \\
\hline\hline
X \xrightarrow{L} \langle Q_0, Q_0\rangle \boldsymbol{spn} & \\
\hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{spn} & \text{(graph morphism)}
\end{array}$$

Of course, the bijective correspondence between X-controlled processes and X-controlled systems does *not* depend on X having just a single node. In fact, multi-sorted control can be useful for implementing certain features.

05. Labeled transition systems, unconstrained

Dropping the constraint that parallel arrows must have different labels yields a similar bijective correspondence

$$\begin{array}{c}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} \qquad \text{(graph morphism)} \\
\hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 \\
\hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 \\
\hline\hline
X \xrightarrow{L} \langle Q_0, Q_0\rangle \boldsymbol{spn} \\
\hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{spn} \qquad \text{(graph morphism)}
\end{array}$$

Of course, the bijective correspondence between $\boldsymbol{X}$-controlled processes and $\boldsymbol{X}$-controlled systems does *not* depend on $\boldsymbol{X}$ having just a single node. In fact, multi-sorted control can be useful for implementing certain features.

05. Labeled transition systems, unconstrained

Dropping the constraint that parallel arrows must have different labels yields a similar bijective correspondence

$$\begin{array}{cl} \boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} & \text{(graph morphism)} \\ \hline\hline X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 & \\ \hline\hline X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 & \\ \hline\hline X \xrightarrow{L} \langle Q_0, Q_0\rangle \boldsymbol{spn} & \\ \hline\hline \boldsymbol{X} \xrightarrow{L} \boldsymbol{spn} & \text{(graph morphism)} \end{array}$$

Of course, the bijective correspondence between $\boldsymbol{X}$-controlled processes and $\boldsymbol{X}$-controlled systems does *not* depend on $\boldsymbol{X}$ having just a single node. In fact, multi-sorted control can be useful for implementing certain features.

05. Labeled transition systems, unconstrained

Dropping the constraint that parallel arrows must have different labels yields a similar bijective correspondence

$$\begin{array}{c}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} \qquad \text{(graph morphism)} \\ \hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 \\ \hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 \\ \hline\hline
X \xrightarrow{L} \langle Q_0, Q_0\rangle \boldsymbol{spn} \\ \hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{spn} \qquad \text{(graph morphism)}
\end{array}$$

Of course, the bijective correspondence between $\boldsymbol{X}$-controlled processes and $\boldsymbol{X}$-controlled systems does *not* depend on $\boldsymbol{X}$ having just a single node. In fact, multi-sorted control can be useful for implementing certain features.

05. Labeled transition systems, unconstrained

Dropping the constraint that parallel arrows must have different labels yields a similar bijective correspondence

$$\begin{array}{c}
\boldsymbol{Q} \xrightarrow{\langle !,\ell\rangle} \boldsymbol{X} \qquad \text{(graph morphism)} \\
\hline\hline
X \xleftarrow{\ell} Q_1 \overset{s}{\underset{t}{\rightrightarrows}} Q_0 \\
\hline\hline
X \xleftarrow{\ell} Q_1 \xrightarrow{\langle s,t\rangle} Q_0 \times Q_0 \\
\hline\hline
X \xrightarrow{L} \langle Q_0, Q_0\rangle \boldsymbol{spn} \\
\hline\hline
\boldsymbol{X} \xrightarrow{L} \boldsymbol{spn} \qquad \text{(graph morphism)}
\end{array}$$

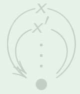
Of course, the bijective correspondence between $\boldsymbol{X}$-controlled processes and $\boldsymbol{X}$-controlled systems does *not* depend on $\boldsymbol{X}$ having just a single node. In fact, multi-sorted control can be useful for implementing certain features.

06. More on graphs

Example

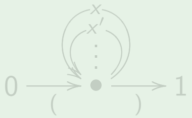
In order to model automata over $\boldsymbol{X}$ with initial and/or final states, we extend the control graph with such states, *e.g.*,

$\boldsymbol{X}$:



becomes

$\boldsymbol{X}_{()}$:



Definition

We denote the (bi)categories of small, respectively, locally small graphs and graph morphisms by $\boldsymbol{grph}$ and by $\boldsymbol{Grph}$. These have non-full sub(bi)categories $\boldsymbol{cat}$ and $\boldsymbol{Cat}$, respectively.

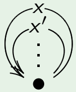
We call a $\boldsymbol{Grph}$-morphism fiber-small, if each object in the codomain has at most a set of pre-images.

06. More on graphs

Example

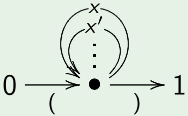
In order to model automata over $\boldsymbol{X}$ with initial and/or final states, we extend the control graph with such states, *e.g.*,

$\boldsymbol{X}$:



becomes

$\boldsymbol{X}_{()}$:



Definition

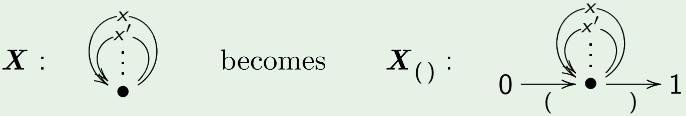
We denote the (bi)categories of small, respectively, locally small graphs and graph morphisms by $\boldsymbol{grph}$ and by $\boldsymbol{Grph}$. These have non-full sub(bi)categories $\boldsymbol{cat}$ and $\boldsymbol{Cat}$, respectively.

We call a $\boldsymbol{Grph}$-morphism fiber-small, if each object in the codomain has at most a set of pre-images.

06. More on graphs

Example

In order to model automata over $\boldsymbol{X}$ with initial and/or final states, we extend the control graph with such states, *e.g.*,



Definition

We denote the (bi)categories of small, respectively, locally small graphs and graph morphisms by $\boldsymbol{grph}$ and by $\boldsymbol{Grph}$. These have non-full sub(bi)categories $\boldsymbol{cat}$ and $\boldsymbol{Cat}$, respectively.

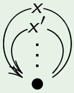
We call a $\boldsymbol{Grph}$-morphism fiber-small, if each object in the codomain has at most a set of pre-images.

06. More on graphs

Example

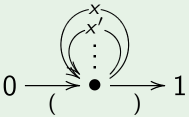
In order to model automata over $\boldsymbol{X}$ with initial and/or final states, we extend the control graph with such states, *e.g.*,

$\boldsymbol{X}$:



becomes

$\boldsymbol{X}_{()}$:



Definition

We denote the (bi)categories of small, respectively, locally small graphs and graph morphisms by $\boldsymbol{grph}$ and by $\boldsymbol{Grph}$. These have non-full sub(bi)categories $\boldsymbol{cat}$ and $\boldsymbol{Cat}$, respectively.

We call a $\boldsymbol{Grph}$-morphism fiber-small, if each object in the codomain has at most a set of pre-images.

07. Graph comprehension

Theorem (generalizing an observation of Pavlović [AP97])

Every (locally) small graph $\boldsymbol{X}$ induces an essentially bijective correspondence between (fiber-)small $\boldsymbol{X}$-controlled processes and $\boldsymbol{X}$-controlled systems $\boldsymbol{X} \longrightarrow \boldsymbol{spn}$.

If $\boldsymbol{X}$ is a (locally) small category, extending a (fiber-)small process $Q \longrightarrow \boldsymbol{X}$ to a functor $Q^{\star} \longrightarrow \boldsymbol{X}$ corresponds to saturating a system $\boldsymbol{X} \xrightarrow{L} \boldsymbol{spn}$ to a lax functor $\boldsymbol{X} \xrightarrow{\hat{L}} \boldsymbol{spn}$.

Proof.

An inverse image construction turns processes into systems, while disjoint unions work in the opposite direction. ☐

It now suffices to settle on morphisms (and possibly 2-cells) for either processes or systems, whatever is more convenient.

07. Graph comprehension

Theorem (generalizing an observation of Pavlović [AP97])

Every (locally) small graph $\boldsymbol{X}$ *induces an essentially bijective correspondence between (fiber-)small* $\boldsymbol{X}$*-controlled processes and* $\boldsymbol{X}$*-controlled systems* $\boldsymbol{X} \longrightarrow \boldsymbol{spn}$.

If $\boldsymbol{X}$ *is a (locally) small category, extending a (fiber-)small process* $\boldsymbol{Q} \longrightarrow \boldsymbol{X}$ *to a functor* $\boldsymbol{Q}^{\star} \longrightarrow \boldsymbol{X}$ *corresponds to saturating a system* $\boldsymbol{X} \xrightarrow{L} \boldsymbol{spn}$ *to a lax functor* $\boldsymbol{X} \xrightarrow{\hat{L}} \boldsymbol{spn}$.

Proof.

An inverse image construction turns processes into systems, while disjoint unions work in the opposite direction. □

It now suffices to settle on morphisms (and possibly 2-cells) for either processes or systems, whatever is more convenient.

07. Graph comprehension

Theorem (generalizing an observation of Pavlović [AP97])

Every (locally) small graph $\boldsymbol{X}$ *induces an essentially bijective correspondence between (fiber-)small* $\boldsymbol{X}$*-controlled processes and* $\boldsymbol{X}$*-controlled systems* $\boldsymbol{X} \longrightarrow \boldsymbol{spn}$.

If $\boldsymbol{X}$ *is a (locally) small category, extending a (fiber-)small process* $\boldsymbol{Q} \longrightarrow \boldsymbol{X}$ *to a functor* $\boldsymbol{Q}^{\star} \longrightarrow \boldsymbol{X}$ *corresponds to saturating a system* $\boldsymbol{X} \xrightarrow{L} \boldsymbol{spn}$ *to a lax functor* $\boldsymbol{X} \xrightarrow{\hat{L}} \boldsymbol{spn}$.

Proof.

An inverse image construction turns processes into systems, while disjoint unions work in the opposite direction. $\square$

It now suffices to settle on morphisms (and possibly 2-cells) for either processes or systems, whatever is more convenient.

07. Graph comprehension

Theorem (generalizing an observation of Pavlović [AP97])

Every (locally) small graph $\boldsymbol{X}$ *induces an essentially bijective correspondence between (fiber-)small* $\boldsymbol{X}$*-controlled processes and* $\boldsymbol{X}$*-controlled systems* $\boldsymbol{X} \longrightarrow \boldsymbol{spn}$.

If $\boldsymbol{X}$ *is a (locally) small category, extending a (fiber-)small process* $\boldsymbol{Q} \longrightarrow \boldsymbol{X}$ *to a functor* $\boldsymbol{Q}^{\star} \longrightarrow \boldsymbol{X}$ *corresponds to saturating a system* $\boldsymbol{X} \xrightarrow{L} \boldsymbol{spn}$ *to a lax functor* $\boldsymbol{X} \xrightarrow{\hat{L}} \boldsymbol{spn}$.

Proof.

An inverse image construction turns processes into systems, while disjoint unions work in the opposite direction. ☐

It now suffices to settle on morphisms (and possibly 2-cells) for either processes or systems, whatever is more convenient.

08. Graph comprehension in context

Remarks

- For a category $\boldsymbol{X}$, Pavlović observed an equivalence between the categories $(\boldsymbol{Cat}/\boldsymbol{X})_{\mathrm{fs}}$ of fiber-small functors into $\boldsymbol{X}$ and commutative triangles as morphisms, and $\lfloor \boldsymbol{X}, \boldsymbol{spn} \rfloor_{\mathrm{folx}}$ of lax functors $\boldsymbol{X} \longrightarrow \boldsymbol{spn}$ with *functional* oplax transformations.
- For discrete $\boldsymbol{X}$, systems trivially factor through $\boldsymbol{set}$: we recover the correspondence $(\boldsymbol{Set}/\boldsymbol{X})_{\mathrm{fs}} \cong [\boldsymbol{X}, \boldsymbol{set}]$ between fiber-small functions into $\boldsymbol{X}$ and $\boldsymbol{X}$-indexed sets.
- If $\boldsymbol{X} = \mathbf{1}$, we recover the correspondences between small graphs and endo-spans on sets, respectively, between small categories and monads in $\boldsymbol{spn}$.
- The Conduché correspondence and the Grothendieck construction can also be obtained by restricting the equivalence above.

08. Graph comprehension in context

Remarks

- For a category X, Pavlović observed an equivalence between the categories $(\boldsymbol{Cat}/X)_{\mathrm{fs}}$ of fiber-small functors into X and commutative triangles as morphisms, and $\lfloor X, \boldsymbol{spn} \rfloor_{\mathrm{folx}}$ of lax functors $X \longrightarrow \boldsymbol{spn}$ with *functional* oplax transformations.
- For discrete X, systems trivially factor through $\boldsymbol{set}$: we recover the correspondence $(\boldsymbol{Set}/X)_{\mathrm{fs}} \cong [X, \boldsymbol{set}]$ between fiber-small functions into X and X-indexed sets.
- If $X = \mathbf{1}$, we recover the correspondences between small graphs and endo-spans on sets, respectively, between small categories and monads in $\boldsymbol{spn}$.
- The Conduché correspondence and the Grothendieck construction can also be obtained by restricting the equivalence above.

08. Graph comprehension in context

Remarks

- For a category $\boldsymbol{X}$, Pavlović observed an equivalence between the categories $(\boldsymbol{Cat}/\boldsymbol{X})_{\mathrm{fs}}$ of fiber-small functors into $\boldsymbol{X}$ and commutative triangles as morphisms, and $\lfloor \boldsymbol{X}, \boldsymbol{spn} \rfloor_{\mathrm{folx}}$ of lax functors $\boldsymbol{X} \longrightarrow \boldsymbol{spn}$ with *functional* oplax transformations.
- For discrete $\boldsymbol{X}$, systems trivially factor through $\boldsymbol{set}$: we recover the correspondence $(\boldsymbol{Set}/\boldsymbol{X})_{\mathrm{fs}} \cong [\boldsymbol{X}, \boldsymbol{set}]$ between fiber-small functions into $\boldsymbol{X}$ and $\boldsymbol{X}$-indexed sets.
- If $\boldsymbol{X} = \mathbf{1}$, we recover the correspondences between small graphs and endo-spans on sets, respectively, between small categories and monads in $\boldsymbol{spn}$.
- The Conduché correspondence and the Grothendieck construction can also be obtained by restricting the equivalence above.

08. Graph comprehension in context

Remarks

- For a category X, Pavlović observed an equivalence between the categories $(Cat/X)_{\mathrm{fs}}$ of fiber-small functors into X and commutative triangles as morphisms, and $\lfloor X, spn \rfloor_{\mathrm{folx}}$ of lax functors $X \longrightarrow spn$ with *functional* oplax transformations.
- For discrete X, systems trivially factor through set: we recover the correspondence $(Set/X)_{\mathrm{fs}} \cong [X, set]$ between fiber-small functions into X and X-indexed sets.
- If $X = \mathbf{1}$, we recover the correspondences between small graphs and endo-spans on sets, respectively, between small categories and monads in spn.
- The Conduché correspondence and the Grothendieck construction can also be obtained by restricting the equivalence above.

09. Which functorial processes do we want?

For processes $Q \xrightarrow{\ell} X$ one is usually interested in arrows of the free category $Q^\star$, and hence in uniformly extending ℓ functorially.

(0) Allowing all graphs as control forces us to form $Q^\star \xrightarrow{\ell^\star} X^\star$, *i.e.*, only free categories arise as controls of functorial processes, which then, in particular, reflect identities.

(1) But a meaningful interpretation of “silent transitions” in Q would seem to require identities in X, hence X should be a category. Restricting the controls to categories from the outset, allows extensions of the form $Q^\star \xrightarrow{\ell^-} X$. This keeps all categories available as controls for functorial processes and fits in well with the saturation of the corresponding systems.

(2) In [BF00] Bunge and Fiore proposed UniqueFactorizationLifting-functors to obtain more functorial processes with the good properties of (0). Since these also reflect identities, they would seem to be incompatible with the notion of silent transition, though.

09. Which functorial processes do we want?

For processes $\boldsymbol{Q} \xrightarrow{\ell} \boldsymbol{X}$ one is usually interested in arrows of the free category $\boldsymbol{Q}^{\star}$, and hence in uniformly extending ℓ functorially.

(0) Allowing all graphs as control forces us to form $\boldsymbol{Q}^{\star} \xrightarrow{\ell^{\star}} \boldsymbol{X}^{\star}$, *i.e.*, only free categories arise as controls of functorial processes, which then, in particular, reflect identities.

(1) But a meaningful interpretation of "silent transitions" in $\boldsymbol{Q}$ would seem to require identities in $\boldsymbol{X}$, hence $\boldsymbol{X}$ should be a category. Restricting the controls to categories from the outset, allows extensions of the form $\boldsymbol{Q}^{\star} \xrightarrow{\ell^{-}} \boldsymbol{X}$. This keeps all categories available as controls for functorial processes and fits in well with the saturation of the corresponding systems.

(2) In [BF00] Bunge and Fiore proposed Unique F actorization L ifting-functors to obtain more functorial processes with the good properties of (0). Since these also reflect identities, they would seem to be incompatible with the notion of silent transition, though.

09. Which functorial processes do we want?

For processes $Q \xrightarrow{\ell} X$ one is usually interested in arrows of the free category $Q^\star$, and hence in uniformly extending ℓ functorially.

(0) Allowing all graphs as control forces us to form $Q^\star \xrightarrow{\ell^\star} X^\star$, *i.e.*, only free categories arise as controls of functorial processes, which then, in particular, reflect identities.

(1) But a meaningful interpretation of “silent transitions” in Q would seem to require identities in X , hence X should be a category. Restricting the controls to categories from the outset, allows extensions of the form $Q^\star \xrightarrow{\ell^-} X$. This keeps all categories available as controls for functorial processes and fits in well with the saturation of the corresponding systems.

(2) In [BF00] Bunge and Fiore proposed UniqueFactorizationLifting-functors to obtain more functorial processes with the good properties of (0). Since these also reflect identities, they would seem to be incompatible with the notion of silent transition, though.

09. Which functorial processes do we want?

For processes $\boldsymbol{Q} \xrightarrow{\ell} \boldsymbol{X}$ one is usually interested in arrows of the free category $\boldsymbol{Q}^{\star}$, and hence in uniformly extending ℓ functorially.

(0) Allowing all graphs as control forces us to form $\boldsymbol{Q}^{\star} \xrightarrow{\ell^{\star}} \boldsymbol{X}^{\star}$, *i.e.*, only free categories arise as controls of functorial processes, which then, in particular, reflect identities.

(1) But a meaningful interpretation of "silent transitions" in $\boldsymbol{Q}$ would seem to require identities in $\boldsymbol{X}$, hence $\boldsymbol{X}$ should be a category. Restricting the controls to categories from the outset, allows extensions of the form $\boldsymbol{Q}^{\star} \xrightarrow{\ell^{-}} \boldsymbol{X}$. This keeps all categories available as controls for functorial processes and fits in well with the saturation of the corresponding systems.

(2) In [BF00] Bunge and Fiore proposed $\text{U}_{\text{nique}}\text{F}_{\text{actorization}}\text{L}_{\text{ifting}}$-functors to obtain more functorial processes with the good properties of (0). Since these also reflect identities, they would seem to be incompatible with the notion of silent transition, though.

09. Which functorial processes do we want?

For processes $\boldsymbol{Q} \xrightarrow{\ell} \boldsymbol{X}$ one is usually interested in arrows of the free category $\boldsymbol{Q}^{\star}$, and hence in uniformly extending ℓ functorially.

(0) Allowing all graphs as control forces us to form $\boldsymbol{Q}^{\star} \xrightarrow{\ell^{\star}} \boldsymbol{X}^{\star}$, *i.e.*, only free categories arise as controls of functorial processes, which then, in particular, reflect identities.

(1) But a meaningful interpretation of "silent transitions" in $\boldsymbol{Q}$ would seem to require identities in $\boldsymbol{X}$, hence $\boldsymbol{X}$ should be a category. Restricting the controls to categories from the outset, allows extensions of the form $\boldsymbol{Q}^{\star} \xrightarrow{\ell^{-}} \boldsymbol{X}$. This keeps all categories available as controls for functorial processes and fits in well with the saturation of the corresponding systems.

(2) In [BF00] Bunge and Fiore proposed $\text{U}_{\text{nique}}\text{F}_{\text{actorization}}\text{L}_{\text{ifting}}$-functors to obtain more functorial processes with the good properties of (0). Since these also reflect identities, they would seem to be incompatible with the notion of silent transition, though.

09. Which functorial processes do we want?

For processes $\boldsymbol{Q} \xrightarrow{\ell} \boldsymbol{X}$ one is usually interested in arrows of the free category $\boldsymbol{Q}^\star$, and hence in uniformly extending ℓ functorially.

(0) Allowing all graphs as control forces us to form $\boldsymbol{Q}^\star \xrightarrow{\ell^\star} \boldsymbol{X}^\star$, *i.e.*, only free categories arise as controls of functorial processes, which then, in particular, reflect identities.

(1) But a meaningful interpretation of "silent transitions" in $\boldsymbol{Q}$ would seem to require identities in $\boldsymbol{X}$, hence $\boldsymbol{X}$ should be a category. Restricting the controls to categories from the outset, allows extensions of the form $\boldsymbol{Q}^\star \xrightarrow{\ell^-} \boldsymbol{X}$. This keeps all categories available as controls for functorial processes and fits in well with the saturation of the corresponding systems.

(2) In [BF00] Bunge and Fiore proposed U$_{\text{nique}}$F$_{\text{actorization}}$L$_{\text{ifting}}$-functors to obtain more functorial processes with the good properties of (0). Since these also reflect identities, they would seem to be incompatible with the notion of silent transition, though.

10. Process- vs. system-view

- Commutative triangles in the process-view, *i.e.*, graph morphisms over $\boldsymbol{X}$, do not produce simulations.
- But for lax functors into any bicategory $\mathcal{W}$, the notion of (op)lax natural transformation is already well-established.

Let's try to weaken this for graph morphisms into $\mathcal{W}$:

Definition

For graph morphisms into a bicategory $\boldsymbol{X} \overset{M}{\underset{L}{\rightrightarrows}} \mathcal{W}$, a lax, respectively, oplax transform $M \overset{\tau}{\Longrightarrow} L$ maps $\boldsymbol{X}$-objects x to 1-cells $xM \xrightarrow{x\tau} xL$ in $\mathcal{W}$ and $\boldsymbol{X}$-arrows $x \xrightarrow{a} y$ to 2-cells in $\mathcal{W}$

$$\begin{tikzcd} xM \arrow[r, "x\tau"] \arrow[d, "aM"'] \arrow[dr, phantom, "\overset{}{\underset{a\tau}{\Swarrow}}"] & xL \arrow[d, "aL"] \\ yM \arrow[r, "y\tau"'] & yL \end{tikzcd} \quad \text{respectively} \quad \begin{tikzcd} xM \arrow[r, "x\tau"] \arrow[d, "aM"'] \arrow[dr, phantom, "\overset{a\tau}{\Nearrow}"] & xL \arrow[d, "aL"] \\ yM \arrow[r, "y\tau"'] & yL \end{tikzcd}$$

10. Process- vs. system-view

- Commutative triangles in the process-view, *i.e.*, graph morphisms over $\boldsymbol{X}$, do not produce simulations.
- But for lax functors into any bicategory $\mathcal{W}$, the notion of (op)lax natural transformation is already well-established.

Let's try to weaken this for graph morphisms into $\mathcal{W}$:

Definition

For graph morphisms into a bicategory $\boldsymbol{X} \overset{M}{\underset{L}{\rightrightarrows}} \mathcal{W}$, a lax, respectively, oplax transform $M \xRightarrow{\tau} L$ maps $\boldsymbol{X}$-objects x to 1-cells $xM \xrightarrow{x\tau} xL$ in $\mathcal{W}$ and $\boldsymbol{X}$-arrows $x \xrightarrow{a} y$ to 2-cells in $\mathcal{W}$

$$\begin{tikzcd}
xM \arrow[r, "x\tau"] \arrow[d, "aM"'] \arrow[dr, phantom, "\swarrow_{a\tau}"] & xL \arrow[d, "aL"] & \text{respectively} & xM \arrow[r, "x\tau"] \arrow[d, "aM"'] \arrow[dr, phantom, "\overset{a\tau}{\nearrow}"] & xL \arrow[d, "aL"] \\
yM \arrow[r, "y\tau"'] & yL & & yM \arrow[r, "y\tau"'] & yL
\end{tikzcd}$$

10. Process- vs. system-view

- Commutative triangles in the process-view, *i.e.*, graph morphisms over $\boldsymbol{X}$, do not produce simulations.
- But for lax functors into any bicategory $\mathcal{W}$, the notion of (op)lax natural transformation is already well-established.

Let's try to weaken this for graph morphisms into $\mathcal{W}$:

Definition

For graph morphisms into a bicategory $\boldsymbol{X} \overset{M}{\underset{L}{\rightrightarrows}} \mathcal{W}$, a lax, respectively, oplax transform $M \xRightarrow{\tau} L$ maps $\boldsymbol{X}$-objects x to 1-cells $xM \xrightarrow{x\tau} xL$ in $\mathcal{W}$ and $\boldsymbol{X}$-arrows $x \xrightarrow{a} y$ to 2-cells in $\mathcal{W}$

$$\begin{tikzcd} xM \arrow[r, "x\tau"] \arrow[d, "aM"'] & xL \arrow[d, "aL"] \arrow[dl, Rightarrow, shorten=4mm, "a\tau"'] \\ yM \arrow[r, "y\tau"'] & yL \end{tikzcd} \quad \text{respectively} \quad \begin{tikzcd} xM \arrow[r, "x\tau"] \arrow[d, "aM"'] & xL \arrow[d, "aL"] \\ yM \arrow[r, "y\tau"'] \arrow[ur, Rightarrow, shorten=4mm, "a\tau"] & yL \end{tikzcd}$$

10. Process- vs. system-view

- Commutative triangles in the process-view, *i.e.*, graph morphisms over $\boldsymbol{X}$, do not produce simulations.
- But for lax functors into any bicategory $\mathcal{W}$, the notion of (op)lax natural transformation is already well-established.

Let's try to weaken this for graph morphisms into $\mathcal{W}$:

Definition

For graph morphisms into a bicategory $\boldsymbol{X} \overset{M}{\underset{L}{\rightrightarrows}} \mathcal{W}$, a lax, respectively, oplax transform $M \overset{\tau}{\Longrightarrow} L$ maps $\boldsymbol{X}$-objects x to 1-cells $xM \xrightarrow{x\tau} xL$ in $\mathcal{W}$ and $\boldsymbol{X}$-arrows $x \xrightarrow{a} y$ to 2-cells in $\mathcal{W}$

$$\begin{tikzcd}
xM \arrow[r, "x\tau"] \arrow[d, "aM"'] \arrow[dr, phantom, "\Swarrow a\tau"] & xL \arrow[d, "aL"] \\
yM \arrow[r, "y\tau"'] & yL
\end{tikzcd}
\quad \text{respectively} \quad
\begin{tikzcd}
xM \arrow[r, "x\tau"] \arrow[d, "aM"'] \arrow[dr, phantom, "a\tau \Nearrow"] & xL \arrow[d, "aL"] \\
yM \arrow[r, "y\tau"'] & yL
\end{tikzcd}$$

11. System-view: weak homomorphisms

As early as 1963 Abraham Ginzburg and Michael Yoeli proposed a definition for ordinary one-sorted LTSs over $\boldsymbol{rel}$ [GY63], which then appeared in a joint paper [GY65], and in Ginzburg's book *Algebraic Automata Theory* [Gin68] (referenced by Milner [Mil71] and Park [Par81]):

Definition (Ginzburg/Yoeli, 1963)

For LTSs $X \xrightarrow{L} \langle Q, Q\rangle\boldsymbol{rel}$ and $X \xrightarrow{M} \langle R, R\rangle\boldsymbol{rel}$ a relation $Q \xrightarrow{S} R$ is called a weak homomorphism from L to M, provided

$$\begin{tikzcd} Q & R \arrow[l, "S^{\mathrm{op}}"'] \arrow[d, "aM"] \\ Q \arrow[from=u, "aL"'] \arrow[ur, phantom, "\rotatebox{45}{$\supseteq$}"] & R \arrow[l, "S^{\mathrm{op}}"] \end{tikzcd} \quad \text{for all } a \in X$$

Weak homomorphimsms $L \Longrightarrow M$ are just lax transforms $M \Longrightarrow L$.

11. System-view: weak homomorphisms

As early as 1963 Abraham Ginzburg and Michael Yoeli proposed a definition for ordinary one-sorted LTSs over $\boldsymbol{rel}$ [GY63], which then appeared in a joint paper [GY65], and in Ginzburg's book *Algebraic Automata Theory* [Gin68] (referenced by Milner [Mil71] and Park [Par81]):

Definition (Ginzburg/Yoeli, 1963)

For LTSs $X \xrightarrow{L} \langle Q, Q\rangle\boldsymbol{rel}$ and $X \xrightarrow{M} \langle R, R\rangle\boldsymbol{rel}$ a relation $Q \xrightarrow{S} R$ is called a weak homomorphism from L to M, provided

$$\begin{tikzcd} Q \arrow[d, "aL"'] & R \arrow[l, "S^{\mathrm{op}}"'] \arrow[d, "aM"] \arrow[dl, phantom, "\subseteq"] \\ Q & R \arrow[l, "S^{\mathrm{op}}"] \end{tikzcd} \quad \text{for all } a \in X$$

Weak homomorphimsms $L \Longrightarrow M$ are just lax transforms $M \Longrightarrow L$.

11. System-view: weak homomorphisms

As early as 1963 Abraham Ginzburg and Michael Yoeli proposed a definition for ordinary one-sorted LTSs over $\boldsymbol{rel}$ [GY63], which then appeared in a joint paper [GY65], and in Ginzburg's book *Algebraic Automata Theory* [Gin68] (referenced by Milner [Mil71] and Park [Par81]):

Definition (Ginzburg/Yoeli, 1963)

For LTSs $X \xrightarrow{L} \langle Q, Q\rangle\boldsymbol{rel}$ and $X \xrightarrow{M} \langle R, R\rangle\boldsymbol{rel}$ a relation $Q \xrightarrow{S} R$ is called a weak homomorphism from L to M, provided

$$\begin{tikzcd}
Q \arrow[d, "aL"'] \arrow[dr, phantom, "\supseteq"] & R \arrow[l, "S^{\mathrm{op}}"'] \arrow[d, "aM"] \\
Q & R \arrow[l, "S^{\mathrm{op}}"]
\end{tikzcd}$$

for all $a \in X$

Weak homomorphimsms $L \Longrightarrow M$ are just lax transforms $M \Longrightarrow L$.

12. Weak homomorphisms vs. simulations

Milner and Park were more interested in process algebra than in automata theory, and Milner coined the more suggestive names "simulation" and "bisimulation" (instead of Park's "mimicry").

Park introduced simulations in a more operational form that still prevails in most CS accounts of the subject. The less than catchy "weak homomorphisms" were largely forgotten, but rediscovered at various times by categorically-minded researchers...

Among many other things, "weak homomorphism" refers to a subalgebra of a binary cartesian product, *cf.*, Lambek [Lam58]. Of course, LTSs $X \xrightarrow{L} \langle Q, Q\rangle\boldsymbol{rel}$ and $X \xrightarrow{M} \langle R, R\rangle\boldsymbol{rel}$ as relational algebras also have a product wrt. function-based homomorphisms:

$$\begin{tikzcd}
X \arrow[r, dotted, "V"] \arrow[d, "X\Delta"'] & \langle Q \times R, Q \times R\rangle\boldsymbol{rel} \\
X \times X \arrow[r, "L \times M"'] & \langle Q, Q\rangle\boldsymbol{rel} \times \langle R, R\rangle\boldsymbol{rel} \arrow[u, tail]
\end{tikzcd}$$

12. Weak homomorphisms vs. simulations

Milner and Park were more interested in process algebra than in automata theory, and Milner coined the more suggestive names "simulation" and "bisimulation" (instead of Park's "mimicry").

Park introduced simulations in a more operational form that still prevails in most CS accounts of the subject. The less than catchy "weak homomorphisms" were largely forgotten, but rediscovered at various times by categorically-minded researchers...

Among many other things, "weak homomorphism" refers to a subalgebra of a binary cartesian product, *cf.*, Lambek [Lam58]. Of course, LTSs $X \xrightarrow{L} \langle Q, Q\rangle\boldsymbol{rel}$ and $X \xrightarrow{M} \langle R, R\rangle\boldsymbol{rel}$ as relational algebras also have a product wrt. function-based homomorphisms:

$$\begin{tikzcd}
X \arrow[r, dotted, "V"] \arrow[d, "X\Delta"'] & \langle Q \times R, Q \times R\rangle\boldsymbol{rel} \\
X \times X \arrow[r, "L \times M"'] & \langle Q, Q\rangle\boldsymbol{rel} \times \langle R, R\rangle\boldsymbol{rel} \arrow[u, tail]
\end{tikzcd}$$

12. Weak homomorphisms vs. simulations

Milner and Park were more interested in process algebra than in automata theory, and Milner coined the more suggestive names "simulation" and "bisimulation" (instead of Park's "mimicry").

Park introduced simulations in a more operational form that still prevails in most CS accounts of the subject. The less than catchy "weak homomorphisms" were largely forgotten, but rediscovered at various times by categorically-minded researchers...

Among many other things, "weak homomorphism" refers to a subalgebra of a binary cartesian product, *cf.*, Lambek [Lam58]. Of course, LTSs $X \xrightarrow{L} \langle Q, Q\rangle\mathbf{rel}$ and $X \xrightarrow{M} \langle R, R\rangle\mathbf{rel}$ as relational algebras also have a product wrt. function-based homomorphisms:

$$\begin{tikzcd}
X \arrow[r, dotted, "V"] \arrow[d, "X\Delta"'] & \langle Q \times R, Q \times R\rangle\mathbf{rel} \\
X \times X \arrow[r, "L \times M"'] & \langle Q, Q\rangle\mathbf{rel} \times \langle R, R\rangle\mathbf{rel} \arrow[u, tail]
\end{tikzcd}$$

12. Weak homomorphisms vs. simulations

Milner and Park were more interested in process algebra than in automata theory, and Milner coined the more suggestive names "simulation" and "bisimulation" (instead of Park's "mimicry").

Park introduced simulations in a more operational form that still prevails in most CS accounts of the subject. The less than catchy "weak homomorphisms" were largely forgotten, but rediscovered at various times by categorically-minded researchers...

Among many other things, "weak homomorphism" refers to a subalgebra of a binary cartesian product, *cf.*, Lambek [Lam58]. Of course, LTSs $X \xrightarrow{L} \langle Q, Q\rangle\boldsymbol{rel}$ and $X \xrightarrow{M} \langle R, R\rangle\boldsymbol{rel}$ as relational algebras also have a product wrt. function-based homomorphisms:

$$\begin{tikzcd}
X \arrow[r, dotted, "V"] \arrow[d, "X\Delta"'] & \langle Q\times R, Q\times R\rangle\boldsymbol{rel} \\
X\times X \arrow[r, "L\times M"'] & \langle Q, Q\rangle\boldsymbol{rel}\times\langle R, R\rangle\boldsymbol{rel} \arrow[u, tail]
\end{tikzcd}$$

12. Weak homomorphisms vs. simulations

Milner and Park were more interested in process algebra than in automata theory, and Milner coined the more suggestive names “simulation” and “bisimulation” (instead of Park’s “mimicry”).

Park introduced simulations in a more operational form that still prevails in most CS accounts of the subject. The less than catchy “weak homomorphisms” were largely forgotten, but rediscovered at various times by categorically-minded researchers...

Among many other things, “weak homomorphism” refers to a subalgebra of a binary cartesian product, *cf.*, Lambek [Lam58]. Of course, LTSs $X \xrightarrow{L} \langle Q, Q\rangle \boldsymbol{rel}$ and $X \xrightarrow{M} \langle R, R\rangle \boldsymbol{rel}$ as relational algebras also have a product wrt. function-based homomorphisms:

$$\begin{tikzcd}
X \arrow[r, dotted, "V"] \arrow[d, "X\Delta"'] & \langle Q \times R, Q \times R\rangle \boldsymbol{rel} \\
X \times X \arrow[r, "L\times M"'] & \langle Q, Q\rangle \boldsymbol{rel} \times \langle R, R\rangle \boldsymbol{rel} \arrow[u, tail]
\end{tikzcd}$$

13. Taking simulations apart (also works over spn)

Weak homomorphisms/simulations, are precisely those “weak sub-structures”, where the existence of outgoing transitions with label $a \in X$ is equivalent to the existence of such transitions in the first component. Then the Australian Mate Calculus becomes applicable:

$$\begin{tikzcd}
Q \arrow[d, "aL"'] & R \arrow[l, "S^{\mathrm{op}}"'] \arrow[d, "aM"] \arrow[dl, phantom, "\subseteq"] \\
Q & R \arrow[l, "S^{\mathrm{op}}"]
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[d, "aL"'] & S \arrow[l, "Q\pi"'] \arrow[d, "aU"] \arrow[dl, phantom, "="] & R \arrow[l, "R\pi^{\mathrm{op}}"'] \arrow[d, "aM"] \arrow[dl, phantom, "\subseteq"] \\
Q & S \arrow[l, "Q\pi"] & R \arrow[l, "R^{\mathrm{op}}\pi"]
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[d, "aL"'] & S \arrow[l, "Q\pi"'] \arrow[r, "R\pi"] \arrow[d, "aU"] \arrow[dl, phantom, "="] \arrow[dr, phantom, "\subseteq"] & R \arrow[d, "aM"] \\
Q & S \arrow[l, "Q\pi"] \arrow[r, "R\pi"'] & R
\end{tikzcd}$$

Sub-structures, or bisimulations, have an equality in the rightmost square. Oplax transforms, or “op-simulations” take care of incoming transitions:

$$\begin{tikzcd}
Q \arrow[r, "S"] \arrow[d, "aL"'] \arrow[dr, phantom, "\supseteq"] & R \arrow[d, "aM"] \\
Q \arrow[r, "S"'] & R
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[r, "Q\pi^{\mathrm{op}}"] \arrow[d, "aL"'] \arrow[dr, phantom, "="] & S \arrow[r, "R\pi"] \arrow[d, "aV"] \arrow[dr, phantom, "\supseteq"] & R \arrow[d, "aM"] \\
Q \arrow[r, "Q\pi^{\mathrm{op}}"'] & S \arrow[r, "R\pi"'] & R
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[r, "Q\pi^{\mathrm{op}}"] \arrow[d, "aL"'] \arrow[dr, phantom, "="] & S \arrow[d, "aV"] \arrow[dr, phantom, "\supseteq"] & R \arrow[l, "R\pi^{\mathrm{op}}"'] \arrow[d, "aM"] \\
Q \arrow[r, "Q\pi^{\mathrm{op}}"'] & S & R \arrow[l, "R\pi^{\mathrm{op}}"]
\end{tikzcd}$$

13. Taking simulations apart (also works over $\boldsymbol{spn}$)

Weak homomorphisms/simulations, are precisely those "weak sub-structures", where the existence of outgoing transitions with label $a \in X$ is equivalent to the existence of such transitions in the first component. Then the Australian Mate Calculus becomes applicable:

$$\begin{tikzcd}
Q \arrow[dd, "aL"description, -{Triangle[open]}] \arrow[ddr, phantom, "\circlearrowright"] & R \arrow[l, "S^{\mathrm{op}}"', -{Triangle[open]}] \arrow[dd, "aM"description, -{Triangle[open]}] & & Q \arrow[dd, "aL"description, -{Triangle[open]}] \arrow[ddr, phantom, "="] & S \arrow[l, "Q\pi"'] \arrow[dd, "aU"description, -{Triangle[open]}] \arrow[ddr, phantom, "\circlearrowright"] & R \arrow[l, "R\pi^{\mathrm{op}}"', -{Triangle[open]}] \arrow[dd, "aM"description, -{Triangle[open]}] & & Q \arrow[dd, "aL"description, -{Triangle[open]}] \arrow[ddr, phantom, "="] & S \arrow[l, "Q\pi"'] \arrow[r, "R\pi"] \arrow[dd, "aU"description, -{Triangle[open]}] \arrow[ddr, phantom, "\circlearrowleft"] & R \arrow[dd, "aM"description, -{Triangle[open]}] \\
& & \text{iff} & & & & \text{iff} & & & \\
Q & R \arrow[l, "S^{\mathrm{op}}", -{Triangle[open]}] & & Q & S \arrow[l, "Q\pi"] & R \arrow[l, "R^{\mathrm{op}}\pi", -{Triangle[open]}] & & Q & S \arrow[l, "Q\pi"] \arrow[r, "R\pi"'] & R
\end{tikzcd}$$

Sub-structures, or bisimulations, have an equality in the rightmost square. Oplax transforms, or "op-simulations" take care of incoming transitions:

$$\begin{tikzcd}
Q \arrow[r, "S", -{Triangle[open]}] \arrow[dd, "aL"description, -{Triangle[open]}] \arrow[ddr, phantom, "\circlearrowleft"] & R \arrow[dd, "aM"description, -{Triangle[open]}] & & Q \arrow[r, "Q\pi^{\mathrm{op}}", -{Triangle[open]}] \arrow[dd, "aL"description, -{Triangle[open]}] \arrow[ddr, phantom, "="] & S \arrow[r, "R\pi"] \arrow[dd, "aV"description, -{Triangle[open]}] \arrow[ddr, phantom, "\circlearrowleft"] & R \arrow[dd, "aM"description, -{Triangle[open]}] & & Q \arrow[r, "Q\pi^{\mathrm{op}}", -{Triangle[open]}] \arrow[dd, "aL"description, -{Triangle[open]}] \arrow[ddr, phantom, "="] & S \arrow[dd, "aV"description, -{Triangle[open]}] \arrow[ddr, phantom, "\circlearrowright"] & R \arrow[l, "R\pi^{\mathrm{op}}"', -{Triangle[open]}] \arrow[dd, "aM"description, -{Triangle[open]}] \\
& & \text{iff} & & & & \text{iff} & & & \\
Q \arrow[r, "S"', -{Triangle[open]}] & R & & Q \arrow[r, "Q\pi^{\mathrm{op}}"', -{Triangle[open]}] & S \arrow[r, "R\pi"'] & R & & Q \arrow[r, "Q\pi^{\mathrm{op}}"', -{Triangle[open]}] & S & R \arrow[l, "R\pi^{\mathrm{op}}", -{Triangle[open]}]
\end{tikzcd}$$

13. Taking simulations apart (also works over spn)

Weak homomorphisms/simulations, are precisely those “weak sub-structures”, where the existence of outgoing transitions with label $a \in X$ is equivalent to the existence of such transitions in the first component. Then the Australian Mate Calculus becomes applicable:

$$\begin{tikzcd}
Q & R \arrow[l, "S^{\mathrm{op}}"'] \\
Q \arrow[from=1-1, "aL"'] & R \arrow[l, "S^{\mathrm{op}}"] \arrow[from=1-2, "aM"] \arrow[from=1-1, to=2-2, phantom, "\circlearrowleft"]
\end{tikzcd}$$

iff

$$\begin{tikzcd}
Q & S \arrow[l, "Q\pi"'] & R \arrow[l, "R\pi^{\mathrm{op}}"'] \\
Q \arrow[from=1-1, "aL"'] & S \arrow[l, "Q\pi"] \arrow[from=1-2, "aU"] \arrow[from=1-1, to=2-2, phantom, "="] & R \arrow[l, "R^{\mathrm{op}}\pi"] \arrow[from=1-3, "aM"] \arrow[from=1-2, to=2-3, phantom, "\circlearrowleft"]
\end{tikzcd}$$

iff

$$\begin{tikzcd}
Q & S \arrow[l, "Q\pi"'] \arrow[r, "R\pi"] & R \\
Q \arrow[from=1-1, "aL"'] & S \arrow[l, "Q\pi"] \arrow[r, "R\pi"'] \arrow[from=1-2, "aU"] \arrow[from=1-1, to=2-2, phantom, "="] & R \arrow[from=1-3, "aM"] \arrow[from=1-2, to=2-3, phantom, "\circlearrowright"]
\end{tikzcd}$$

Sub-structures, or bisimulations, have an equality in the rightmost square. Oplax transforms, or “op-simulations” take care of incoming transitions:

$$\begin{tikzcd}
Q \arrow[r, "S"] & R \\
Q \arrow[from=1-1, "aL"'] \arrow[r, "S"'] & R \arrow[from=1-2, "aM"] \arrow[from=1-1, to=2-2, phantom, "\circlearrowright"]
\end{tikzcd}$$

iff

$$\begin{tikzcd}
Q \arrow[r, "Q\pi^{\mathrm{op}}"] & S \arrow[r, "R\pi"] & R \\
Q \arrow[from=1-1, "aL"'] \arrow[r, "Q\pi^{\mathrm{op}}"'] & S \arrow[r, "R\pi"'] \arrow[from=1-2, "aV"] \arrow[from=1-1, to=2-2, phantom, "="] & R \arrow[from=1-3, "aM"] \arrow[from=1-2, to=2-3, phantom, "\circlearrowright"]
\end{tikzcd}$$

iff

$$\begin{tikzcd}
Q \arrow[r, "Q\pi^{\mathrm{op}}"] & S & R \arrow[l, "R\pi^{\mathrm{op}}"'] \\
Q \arrow[from=1-1, "aL"'] \arrow[r, "Q\pi^{\mathrm{op}}"'] & S \arrow[from=1-2, "aV"] \arrow[from=1-1, to=2-2, phantom, "="] & R \arrow[l, "R\pi^{\mathrm{op}}"] \arrow[from=1-3, "aM"] \arrow[from=1-2, to=2-3, phantom, "\circlearrowleft"]
\end{tikzcd}$$

13. Taking simulations apart (also works over spn)

Weak homomorphisms/simulations, are precisely those “weak sub-structures”, where the existence of outgoing transitions with label $a \in X$ is equivalent to the existence of such transitions in the first component. Then the Australian Mate Calculus becomes applicable:

$$\begin{tikzcd}
Q \arrow[d, "aL"', "|" marking, -{Triangle[open]}] \arrow[dr, phantom, "\circlearrowright"] & R \arrow[l, "S^{\mathrm{op}}"', -{Triangle[open]}] \arrow[d, "aM", "|" marking, -{Triangle[open]}] \\
Q & R \arrow[l, "S^{\mathrm{op}}", -{Triangle[open]}]
\end{tikzcd}$$

iff

$$\begin{tikzcd}
Q \arrow[d, "aL"', "|" marking, -{Triangle[open]}] \arrow[dr, phantom, "\diagdown\!\diagdown"] & S \arrow[l, "Q\pi"'] \arrow[d, "aU" description, "|" marking, -{Triangle[open]}] \arrow[dr, phantom, "\circlearrowright"] & R \arrow[l, "R\pi^{\mathrm{op}}"', -{Triangle[open]}] \arrow[d, "aM", "|" marking, -{Triangle[open]}] \\
Q & S \arrow[l, "Q\pi"] & R \arrow[l, "R^{\mathrm{op}}\pi", -{Triangle[open]}]
\end{tikzcd}$$

iff

$$\begin{tikzcd}
Q \arrow[d, "aL"', "|" marking, -{Triangle[open]}] \arrow[dr, phantom, "\diagdown\!\diagdown"] & S \arrow[l, "Q\pi"'] \arrow[r, "R\pi"] \arrow[d, "aU" description, "|" marking, -{Triangle[open]}] \arrow[dr, phantom, "\circlearrowleft"] & R \arrow[d, "aM", "|" marking, -{Triangle[open]}] \\
Q & S \arrow[l, "Q\pi"] \arrow[r, "R\pi"'] & R
\end{tikzcd}$$

Sub-structures, or bisimulations, have an equality in the rightmost square. Oplax transforms, or “op-simulations” take care of incoming transitions:

$$\begin{tikzcd}
Q \arrow[r, "S", -{Triangle[open]}] \arrow[d, "aL"', "|" marking, -{Triangle[open]}] \arrow[dr, phantom, "\circlearrowleft"] & R \arrow[d, "aM", "|" marking, -{Triangle[open]}] \\
Q \arrow[r, "S"', -{Triangle[open]}] & R
\end{tikzcd}$$

iff

$$\begin{tikzcd}
Q \arrow[r, "Q\pi^{\mathrm{op}}", -{Triangle[open]}] \arrow[d, "aL"', "|" marking, -{Triangle[open]}] \arrow[dr, phantom, "\diagup\!\diagup"] & S \arrow[r, "R\pi"] \arrow[d, "aV" description, "|" marking, -{Triangle[open]}] \arrow[dr, phantom, "\circlearrowleft"] & R \arrow[d, "aM", "|" marking, -{Triangle[open]}] \\
Q \arrow[r, "Q\pi^{\mathrm{op}}"', -{Triangle[open]}] & S \arrow[r, "R\pi"'] & R
\end{tikzcd}$$

iff

$$\begin{tikzcd}
Q \arrow[r, "Q\pi^{\mathrm{op}}", -{Triangle[open]}] \arrow[d, "aL"', "|" marking, -{Triangle[open]}] \arrow[dr, phantom, "\diagup\!\diagup"] & S \arrow[d, "aV" description, "|" marking, -{Triangle[open]}] \arrow[dr, phantom, "\circlearrowright"] & R \arrow[l, "R\pi^{\mathrm{op}}"', -{Triangle[open]}] \arrow[d, "aM", "|" marking, -{Triangle[open]}] \\
Q \arrow[r, "Q\pi^{\mathrm{op}}"', -{Triangle[open]}] & S & R \arrow[l, "R\pi^{\mathrm{op}}", -{Triangle[open]}]
\end{tikzcd}$$

13. Taking simulations apart (also works over spn)

Weak homomorphisms/simulations, are precisely those “weak sub-structures”, where the existence of outgoing transitions with label $a \in X$ is equivalent to the existence of such transitions in the first component. Then the Australian Mate Calculus becomes applicable:

$$\begin{tikzcd}
Q \arrow[d, "aL"'] & R \arrow[l, "S^{\mathrm{op}}"'] \arrow[d, "aM"] \arrow[dl, phantom, "\circlearrowleft"] \\
Q & R \arrow[l, "S^{\mathrm{op}}"]
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[d, "aL"'] & S \arrow[l, "Q\pi"'] \arrow[d, "aU"] \arrow[dl, phantom, "="] & R \arrow[l, "R\pi^{\mathrm{op}}"'] \arrow[d, "aM"] \arrow[dl, phantom, "\circlearrowleft"] \\
Q & S \arrow[l, "Q\pi"] & R \arrow[l, "R^{\mathrm{op}}\pi"]
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[d, "aL"'] & S \arrow[l, "Q\pi"'] \arrow[r, "R\pi"] \arrow[d, "aU"] \arrow[dl, phantom, "="] & R \arrow[d, "aM"] \arrow[dl, phantom, "\circlearrowright"] \\
Q & S \arrow[l, "Q\pi"] \arrow[r, "R\pi"'] & R
\end{tikzcd}$$

Sub-structures, or bisimulations, have an equality in the rightmost square. Oplax transforms, or “op-simulations” take care of incoming transitions:

$$\begin{tikzcd}
Q \arrow[r, "S"] \arrow[d, "aL"'] \arrow[dr, phantom, "\circlearrowright"] & R \arrow[d, "aM"] \\
Q \arrow[r, "S"'] & R
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[r, "Q\pi^{\mathrm{op}}"] \arrow[d, "aL"'] \arrow[dr, phantom, "="] & S \arrow[r, "R\pi"] \arrow[d, "aV"] \arrow[dr, phantom, "\circlearrowright"] & R \arrow[d, "aM"] \\
Q \arrow[r, "Q\pi^{\mathrm{op}}"'] & S \arrow[r, "R\pi"'] & R
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[r, "Q\pi^{\mathrm{op}}"] \arrow[d, "aL"'] \arrow[dr, phantom, "="] & S \arrow[d, "aV"] \arrow[dr, phantom, "\circlearrowleft"] & R \arrow[l, "R\pi^{\mathrm{op}}"'] \arrow[d, "aM"] \\
Q \arrow[r, "Q\pi^{\mathrm{op}}"'] & S & R \arrow[l, "R\pi^{\mathrm{op}}"]
\end{tikzcd}$$

13. Taking simulations apart (also works over spn)

Weak homomorphisms/simulations, are precisely those “weak sub-structures”, where the existence of outgoing transitions with label $a \in X$ is equivalent to the existence of such transitions in the first component. Then the Australian Mate Calculus becomes applicable:

$$\begin{tikzcd} Q \arrow[d, "aL"'] & R \arrow[l, -{Triangle[open]}, "S^{\mathrm{op}}"'] \arrow[d, "aM"] \arrow[dl, phantom, "\supseteq"] \\ Q & R \arrow[l, -{Triangle[open]}, "S^{\mathrm{op}}"] \end{tikzcd}$$

iff

$$\begin{tikzcd} Q \arrow[d, "aL"'] & S \arrow[l, "Q\pi"'] \arrow[d, "aU"] \arrow[dl, phantom, "="] & R \arrow[l, -{Triangle[open]}, "R\pi^{\mathrm{op}}"'] \arrow[d, "aM"] \arrow[dl, phantom, "\supseteq"] \\ Q & S \arrow[l, "Q\pi"] & R \arrow[l, -{Triangle[open]}, "R^{\mathrm{op}}\pi"] \end{tikzcd}$$

iff

$$\begin{tikzcd} Q \arrow[d, "aL"'] & S \arrow[l, "Q\pi"'] \arrow[d, "aU"] \arrow[dl, phantom, "="] \arrow[r, "R\pi"] \arrow[dr, phantom, "\subseteq"] & R \arrow[d, "aM"] \\ Q & S \arrow[l, "Q\pi"] \arrow[r, "R\pi"'] & R \end{tikzcd}$$

Sub-structures, or bisimulations, have an equality in the rightmost square. Oplax transforms, or “op-simulations” take care of incoming transitions:

$$\begin{tikzcd} Q \arrow[r, -{Triangle[open]}, "S"] \arrow[d, "aL"'] \arrow[dr, phantom, "\subseteq"] & R \arrow[d, "aM"] \\ Q \arrow[r, -{Triangle[open]}, "S"'] & R \end{tikzcd}$$

iff

$$\begin{tikzcd} Q \arrow[r, -{Triangle[open]}, "Q\pi^{\mathrm{op}}"] \arrow[d, "aL"'] \arrow[dr, phantom, "="] & S \arrow[r, "R\pi"] \arrow[d, "aV"] \arrow[dr, phantom, "\subseteq"] & R \arrow[d, "aM"] \\ Q \arrow[r, -{Triangle[open]}, "Q\pi^{\mathrm{op}}"'] & S \arrow[r, "R\pi"'] & R \end{tikzcd}$$

iff

$$\begin{tikzcd} Q \arrow[r, -{Triangle[open]}, "Q\pi^{\mathrm{op}}"] \arrow[d, "aL"'] \arrow[dr, phantom, "="] & S \arrow[d, "aV"] \arrow[dr, phantom, "\supseteq"] & R \arrow[l, -{Triangle[open]}, "R\pi^{\mathrm{op}}"'] \arrow[d, "aM"] \\ Q \arrow[r, -{Triangle[open]}, "Q\pi^{\mathrm{op}}"'] & S & R \arrow[l, -{Triangle[open]}, "R\pi^{\mathrm{op}}"] \end{tikzcd}$$

13. Taking simulations apart (also works over spn)

Weak homomorphisms/simulations, are precisely those “weak sub-structures”, where the existence of outgoing transitions with label $a \in X$ is equivalent to the existence of such transitions in the first component. Then the Australian Mate Calculus becomes applicable:

$$\begin{tikzcd}
Q \arrow[d, "aL"'] & R \arrow[l, "S^{\mathrm{op}}"'] \arrow[d, "aM"] \arrow[dl, phantom, "\Swarrow"] & \text{iff} & Q \arrow[d, "aL"'] & S \arrow[l, "Q\pi"'] \arrow[d, "aU"] \arrow[dl, phantom, "="] & R \arrow[l, "R\pi^{\mathrm{op}}"'] \arrow[d, "aM"] \arrow[dl, phantom, "\Swarrow"] & \text{iff} & Q \arrow[d, "aL"'] & S \arrow[l, "Q\pi"'] \arrow[r, "R\pi"] \arrow[d, "aU"] \arrow[dl, phantom, "="] \arrow[dr, phantom, "\Searrow"] & R \arrow[d, "aM"] \\
Q & R \arrow[l, "S^{\mathrm{op}}"] & & Q & S \arrow[l, "Q\pi"] & R \arrow[l, "R^{\mathrm{op}}\pi"] & & Q & S \arrow[l, "Q\pi"] \arrow[r, "R\pi"'] & R
\end{tikzcd}$$

Sub-structures, or bisimulations, have an equality in the rightmost square.

Oplax transforms, or “op-simulations” take care of incoming transitions:

$$\begin{tikzcd}
Q \arrow[r, "S"] \arrow[d, "aL"'] \arrow[dr, phantom, "\Searrow"] & R \arrow[d, "aM"] & \text{iff} & Q \arrow[r, "Q\pi^{\mathrm{op}}"] \arrow[d, "aL"'] \arrow[dr, phantom, "="] & S \arrow[r, "R\pi"] \arrow[d, "aV"] \arrow[dr, phantom, "\Searrow"] & R \arrow[d, "aM"] & \text{iff} & Q \arrow[r, "Q\pi^{\mathrm{op}}"] \arrow[d, "aL"'] \arrow[dr, phantom, "="] & S \arrow[d, "aV"] \arrow[dr, phantom, "\Swarrow"] & R \arrow[l, "R\pi^{\mathrm{op}}"'] \arrow[d, "aM"] \\
Q \arrow[r, "S"'] & R & & Q \arrow[r, "Q\pi^{\mathrm{op}}"'] & S \arrow[r, "R\pi"'] & R & & Q \arrow[r, "Q\pi^{\mathrm{op}}"'] & S & R \arrow[l, "R\pi^{\mathrm{op}}"]
\end{tikzcd}$$

13. Taking simulations apart (also works over spn)

Weak homomorphisms/simulations, are precisely those “weak sub-structures”, where the existence of outgoing transitions with label $a \in X$ is equivalent to the existence of such transitions in the first component. Then the Australian Mate Calculus becomes applicable:

$$\begin{tikzcd}
Q \arrow[d, "|" marking, -{Triangle[open]}, "aL"'] & R \arrow[l, -{Triangle[open]}, "S^{\mathrm{op}}"'] \arrow[d, "|" marking, -{Triangle[open]}, "aM"] \arrow[dl, phantom, "\circlearrowleft"] & \text{iff} & Q \arrow[d, "|" marking, -{Triangle[open]}, "aL"'] & S \arrow[l, "Q\pi"'] \arrow[d, "|" marking, -{Triangle[open]}, "aU"] \arrow[dl, phantom, "="] & R \arrow[l, -{Triangle[open]}, "R\pi^{\mathrm{op}}"'] \arrow[d, "|" marking, -{Triangle[open]}, "aM"] \arrow[dl, phantom, "\circlearrowleft"] & \text{iff} & Q \arrow[d, "|" marking, -{Triangle[open]}, "aL"'] & S \arrow[l, "Q\pi"'] \arrow[r, "R\pi"] \arrow[d, "|" marking, -{Triangle[open]}, "aU"] \arrow[dl, phantom, "="] \arrow[dr, phantom, "\circlearrowright"] & R \arrow[d, "|" marking, -{Triangle[open]}, "aM"] \\
Q & R \arrow[l, -{Triangle[open]}, "S^{\mathrm{op}}"] & & Q & S \arrow[l, "Q\pi"] & R \arrow[l, -{Triangle[open]}, "R^{\mathrm{op}}\pi"] & & Q & S \arrow[l, "Q\pi"] \arrow[r, "R\pi"'] & R
\end{tikzcd}$$

Sub-structures, or bisimulations, have an equality in the rightmost square. Oplax transforms, or “op-simulations” take care of incoming transitions:

$$\begin{tikzcd}
Q \arrow[r, -{Triangle[open]}, "S"] \arrow[d, "|" marking, -{Triangle[open]}, "aL"'] \arrow[dr, phantom, "\circlearrowright"] & R \arrow[d, "|" marking, -{Triangle[open]}, "aM"] & \text{iff} & Q \arrow[r, -{Triangle[open]}, "Q\pi^{\mathrm{op}}"] \arrow[d, "|" marking, -{Triangle[open]}, "aL"'] \arrow[dr, phantom, "="] & S \arrow[r, "R\pi"] \arrow[d, "|" marking, -{Triangle[open]}, "aV"] \arrow[dr, phantom, "\circlearrowright"] & R \arrow[d, "|" marking, -{Triangle[open]}, "aM"] & \text{iff} & Q \arrow[r, -{Triangle[open]}, "Q\pi^{\mathrm{op}}"] \arrow[d, "|" marking, -{Triangle[open]}, "aL"'] \arrow[dr, phantom, "="] & S \arrow[d, "|" marking, -{Triangle[open]}, "aV"] \arrow[dr, phantom, "\circlearrowleft"] & R \arrow[l, -{Triangle[open]}, "R\pi^{\mathrm{op}}"'] \arrow[d, "|" marking, -{Triangle[open]}, "aM"] \\
Q \arrow[r, -{Triangle[open]}, "S"'] & R & & Q \arrow[r, -{Triangle[open]}, "Q\pi^{\mathrm{op}}"'] & S \arrow[r, "R\pi"'] & R & & Q \arrow[r, -{Triangle[open]}, "Q\pi^{\mathrm{op}}"'] & S & R \arrow[l, -{Triangle[open]}, "R\pi^{\mathrm{op}}"]
\end{tikzcd}$$

13. Taking simulations apart (also works over spn)

Weak homomorphisms/simulations, are precisely those “weak substructures”, where the existence of outgoing transitions with label $a \in X$ is equivalent to the existence of such transitions in the first component. Then the Australian Mate Calculus becomes applicable:

$$\begin{tikzcd}
Q \arrow[d, "aL" description, -{Triangle[open]}] \arrow[dr, phantom, "\supseteq"] & R \arrow[l, "S^{\mathrm{op}}"', -{Triangle[open]}] \arrow[d, "aM" description, -{Triangle[open]}] \\
Q & R \arrow[l, "S^{\mathrm{op}}", -{Triangle[open]}]
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[d, "aL" description, -{Triangle[open]}] \arrow[dr, phantom, "="] & S \arrow[l, "Q\pi"'] \arrow[d, "aU" description, -{Triangle[open]}] \arrow[dr, phantom, "\supseteq"] & R \arrow[l, "R\pi^{\mathrm{op}}"', -{Triangle[open]}] \arrow[d, "aM" description, -{Triangle[open]}] \\
Q & S \arrow[l, "Q\pi"] & R \arrow[l, "R^{\mathrm{op}}\pi", -{Triangle[open]}]
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[d, "aL" description, -{Triangle[open]}] \arrow[dr, phantom, "="] & S \arrow[l, "Q\pi"'] \arrow[r, "R\pi"] \arrow[d, "aU" description, -{Triangle[open]}] \arrow[dr, phantom, "\subseteq"] & R \arrow[d, "aM" description, -{Triangle[open]}] \\
Q & S \arrow[l, "Q\pi"] \arrow[r, "R\pi"'] & R
\end{tikzcd}$$

Sub-structures, or bisimulations, have an equality in the rightmost square. Oplax transforms, or “op-simulations” take care of incoming transitions:

$$\begin{tikzcd}
Q \arrow[r, "S", -{Triangle[open]}] \arrow[d, "aL" description, -{Triangle[open]}] \arrow[dr, phantom, "\subseteq"] & R \arrow[d, "aM" description, -{Triangle[open]}] \\
Q \arrow[r, "S"', -{Triangle[open]}] & R
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[r, "Q\pi^{\mathrm{op}}", -{Triangle[open]}] \arrow[d, "aL" description, -{Triangle[open]}] \arrow[dr, phantom, "="] & S \arrow[r, "R\pi"] \arrow[d, "aV" description, -{Triangle[open]}] \arrow[dr, phantom, "\subseteq"] & R \arrow[d, "aM" description, -{Triangle[open]}] \\
Q \arrow[r, "Q\pi^{\mathrm{op}}"', -{Triangle[open]}] & S \arrow[r, "R\pi"'] & R
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[r, "Q\pi^{\mathrm{op}}", -{Triangle[open]}] \arrow[d, "aL" description, -{Triangle[open]}] \arrow[dr, phantom, "="] & S \arrow[d, "aV" description, -{Triangle[open]}] \arrow[dr, phantom, "\supseteq"] & R \arrow[l, "R\pi^{\mathrm{op}}"', -{Triangle[open]}] \arrow[d, "aM" description, -{Triangle[open]}] \\
Q \arrow[r, "Q\pi^{\mathrm{op}}"', -{Triangle[open]}] & S & R \arrow[l, "R\pi^{\mathrm{op}}", -{Triangle[open]}]
\end{tikzcd}$$

13. Taking simulations apart (also works over spn)

Weak homomorphisms/simulations, are precisely those “weak sub-structures”, where the existence of outgoing transitions with label $a \in X$ is equivalent to the existence of such transitions in the first component. Then the Australian Mate Calculus becomes applicable:

$$\begin{tikzcd}
Q \arrow[d, "aL" description, -{Triangle[open]}] & R \arrow[l, "S^{\mathrm{op}}"', -{Triangle[open]}] \arrow[d, "aM" description, -{Triangle[open]}] \arrow[dl, phantom, "\supseteq"] \\
Q & R \arrow[l, "S^{\mathrm{op}}", -{Triangle[open]}]
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[d, "aL" description, -{Triangle[open]}] & S \arrow[l, "Q\pi"'] \arrow[d, "aU" description, -{Triangle[open]}] \arrow[dl, phantom, "="] & R \arrow[l, "R\pi^{\mathrm{op}}"', -{Triangle[open]}] \arrow[d, "aM" description, -{Triangle[open]}] \arrow[dl, phantom, "\supseteq"] \\
Q & S \arrow[l, "Q\pi"] & R \arrow[l, "R^{\mathrm{op}}\pi", -{Triangle[open]}]
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[d, "aL" description, -{Triangle[open]}] & S \arrow[l, "Q\pi"'] \arrow[r, "R\pi"] \arrow[d, "aU" description, -{Triangle[open]}] \arrow[dl, phantom, "="] \arrow[dr, phantom, "\subseteq"] & R \arrow[d, "aM" description, -{Triangle[open]}] \\
Q & S \arrow[l, "Q\pi"] \arrow[r, "R\pi"'] & R
\end{tikzcd}$$

Sub-structures, or bisimulations, have an equality in the rightmost square. Oplax transforms, or “op-simulations” take care of incoming transitions:

$$\begin{tikzcd}
Q \arrow[r, "S", -{Triangle[open]}] \arrow[d, "aL" description, -{Triangle[open]}] \arrow[dr, phantom, "\subseteq"] & R \arrow[d, "aM" description, -{Triangle[open]}] \\
Q \arrow[r, "S"', -{Triangle[open]}] & R
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[r, "Q\pi^{\mathrm{op}}", -{Triangle[open]}] \arrow[d, "aL" description, -{Triangle[open]}] \arrow[dr, phantom, "="] & S \arrow[r, "R\pi"] \arrow[d, "aV" description, -{Triangle[open]}] \arrow[dr, phantom, "\subseteq"] & R \arrow[d, "aM" description, -{Triangle[open]}] \\
Q \arrow[r, "Q\pi^{\mathrm{op}}"', -{Triangle[open]}] & S \arrow[r, "R\pi"'] & R
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[r, "Q\pi^{\mathrm{op}}", -{Triangle[open]}] \arrow[d, "aL" description, -{Triangle[open]}] \arrow[dr, phantom, "="] & S \arrow[d, "aV" description, -{Triangle[open]}] & R \arrow[l, "R\pi^{\mathrm{op}}"', -{Triangle[open]}] \arrow[d, "aM" description, -{Triangle[open]}] \arrow[dl, phantom, "\supseteq"] \\
Q \arrow[r, "Q\pi^{\mathrm{op}}"', -{Triangle[open]}] & S & R \arrow[l, "R\pi^{\mathrm{op}}", -{Triangle[open]}]
\end{tikzcd}$$

13. Taking simulations apart (also works over $\boldsymbol{spn}$)

Weak homomorphisms/simulations, are precisely those “weak sub-structures”, where the existence of outgoing transitions with label $a \in X$ is equivalent to the existence of such transitions in the first component. Then the Australian Mate Calculus becomes applicable:

$$\begin{tikzcd}
Q \arrow[d, -{Triangle[open]}, "aL"] \arrow[dr, phantom, "\supseteq"] & R \arrow[l, -{Triangle[open]}, "S^{\mathrm{op}}"'] \arrow[d, -{Triangle[open]}, "aM"] \\
Q & R \arrow[l, -{Triangle[open]}, "S^{\mathrm{op}}"]
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[d, -{Triangle[open]}, "aL"] \arrow[dr, phantom, "="] & S \arrow[l, "Q\pi"'] \arrow[d, -{Triangle[open]}, "aU"] \arrow[dr, phantom, "\supseteq"] & R \arrow[l, -{Triangle[open]}, "R\pi^{\mathrm{op}}"'] \arrow[d, -{Triangle[open]}, "aM"] \\
Q & S \arrow[l, "Q\pi"] & R \arrow[l, -{Triangle[open]}, "R^{\mathrm{op}}\pi"]
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[d, -{Triangle[open]}, "aL"] \arrow[dr, phantom, "="] & S \arrow[l, "Q\pi"'] \arrow[r, "R\pi"] \arrow[d, -{Triangle[open]}, "aU"] \arrow[dr, phantom, "\subseteq"] & R \arrow[d, -{Triangle[open]}, "aM"] \\
Q & S \arrow[l, "Q\pi"] \arrow[r, "R\pi"'] & R
\end{tikzcd}$$

Sub-structures, or bisimulations, have an equality in the rightmost square. Oplax transforms, or “op-simulations” take care of incoming transitions:

$$\begin{tikzcd}
Q \arrow[r, -{Triangle[open]}, "S"] \arrow[d, -{Triangle[open]}, "aL"] \arrow[dr, phantom, "\subseteq"] & R \arrow[d, -{Triangle[open]}, "aM"] \\
Q \arrow[r, -{Triangle[open]}, "S"'] & R
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[r, -{Triangle[open]}, "Q\pi^{\mathrm{op}}"] \arrow[d, -{Triangle[open]}, "aL"] \arrow[dr, phantom, "="] & S \arrow[r, "R\pi"] \arrow[d, -{Triangle[open]}, "aV"] \arrow[dr, phantom, "\subseteq"] & R \arrow[d, -{Triangle[open]}, "aM"] \\
Q \arrow[r, -{Triangle[open]}, "Q\pi^{\mathrm{op}}"'] & S \arrow[r, "R\pi"'] & R
\end{tikzcd}
\quad \text{iff} \quad
\begin{tikzcd}
Q \arrow[r, -{Triangle[open]}, "Q\pi^{\mathrm{op}}"] \arrow[d, -{Triangle[open]}, "aL"] \arrow[dr, phantom, "="] & S \arrow[d, -{Triangle[open]}, "aV"] \arrow[dr, phantom, "\supseteq"] & R \arrow[l, -{Triangle[open]}, "R\pi^{\mathrm{op}}"'] \arrow[d, -{Triangle[open]}, "aM"] \\
Q \arrow[r, -{Triangle[open]}, "Q\pi^{\mathrm{op}}"'] & S & R \arrow[l, -{Triangle[open]}, "R\pi^{\mathrm{op}}"]
\end{tikzcd}$$

14. Graph comprehension revisited

Contrast a lax transform $L \Longrightarrow M$ with the rightmost diagram for simulations, translated into the world of $\boldsymbol{X}$-controlled processes:

$$\begin{tikzcd}
\boldsymbol{X} \arrow[d, bend right=60, "L"'] \arrow[d, bend left=60, "M"] \arrow[d, phantom, "\Leftarrow"] \\
\boldsymbol{spn}
\end{tikzcd}
\quad \text{vs.} \quad
\begin{tikzcd}
& \boldsymbol{S} \arrow[ld, "Q\pi"'] \arrow[rd, "R\pi"] \arrow[dd, dashed, "U"] & \\
\boldsymbol{Q} \arrow[rd, "\ell"'] & & \boldsymbol{R} \arrow[ld, "m"] \\
& \boldsymbol{X} &
\end{tikzcd}$$

where $Q\pi$ is $(1 \xrightarrow{0} 2)$-injective

There are three other such correspondences. Pavlović was aware of one of these, restricted to lax functors into $\boldsymbol{spn}$ /functors into $\boldsymbol{X}$.

For systems the precedent of $\mathcal{W}$-enriched categories suggestes the feasibility of change-of-control (seldom considered for processes)

$$\begin{tikzcd}
\boldsymbol{X} \arrow[rr, bend left, "H"] \arrow[rd, bend right, "L"'] & \arrow[d, phantom, "\overset{\swarrow}{\sigma}"] & \boldsymbol{Y} \arrow[ld, bend left, "M"] \\
& \mathcal{W} &
\end{tikzcd}
\quad \text{or} \quad
\begin{tikzcd}
\boldsymbol{X} \arrow[rr, bend left, "H"] \arrow[rd, bend right, "L"'] & \arrow[d, phantom, "\overset{\kappa}{\nearrow}"] & \boldsymbol{Y} \arrow[ld, bend left, "M"] \\
& \mathcal{W} &
\end{tikzcd}$$

The precise nature of H (graph morphism, (pro)functor, or even a span of those?) remains to be determined.

14. Graph comprehension revisited

Contrast a lax transform $L \Longrightarrow M$ with the rightmost diagram for simulations, translated into the world of $\boldsymbol{X}$-controlled processes:

$$\begin{tikzcd}
\boldsymbol{X} \arrow[d, bend right=50, "L"'] \arrow[d, bend left=50, "M"] \arrow[d, phantom, "\Leftarrow"] \\
\boldsymbol{spn}
\end{tikzcd}
\qquad \text{vs.} \qquad
\begin{tikzcd}
 & \boldsymbol{S} \arrow[dl, "Q\pi"'] \arrow[dr, "R\pi"] \arrow[d, dashed, "U"] & \\
\boldsymbol{Q} \arrow[r, "\ell"'] & \boldsymbol{X} & \boldsymbol{R} \arrow[l, "m"]
\end{tikzcd}$$

where $\boldsymbol{Q}\pi$ is $(1 \xrightarrow{0} 2)$-injective

There are three other such correspondences. Pavlović was aware of one of these, restricted to lax functors into $\boldsymbol{spn}$ /functors into $\boldsymbol{X}$.

For systems the precedent of $\mathcal{W}$-enriched categories suggestes the feasibility of change-of-control (seldom considered for processes)

$$\begin{tikzcd}
\boldsymbol{X} \arrow[rr, "H"] \arrow[dr, "L"'] & \arrow[d, phantom, "\overset{}{\Swarrow}\,\sigma"] & \boldsymbol{Y} \arrow[dl, "M"] \\
 & \mathcal{W} &
\end{tikzcd}
\qquad \text{or} \qquad
\begin{tikzcd}
\boldsymbol{X} \arrow[rr, "H"] \arrow[dr, "L"'] & \arrow[d, phantom, "\kappa\,\Nearrow"] & \boldsymbol{Y} \arrow[dl, "M"] \\
 & \mathcal{W} &
\end{tikzcd}$$

The precise nature of H (graph morphism, (pro)functor, or even a span of those?) remains to be determined.

14. Graph comprehension revisited

Contrast a lax transform $L \Longrightarrow M$ with the rightmost diagram for simulations, translated into the world of $\boldsymbol{X}$-controlled processes:

$$\begin{tikzcd}
\boldsymbol{X} \arrow[d, bend right=50, "L"'] \arrow[d, bend left=50, "M"] \arrow[d, phantom, "\Leftarrow"] \\
\boldsymbol{spn}
\end{tikzcd}
\quad \text{vs.} \quad
\begin{tikzcd}
 & \boldsymbol{S} \arrow[dl, "Q\pi"'] \arrow[dr, "R\pi"] \arrow[dd, dashed, "U"] & \\
\boldsymbol{Q} \arrow[dr, "\ell"'] & & \boldsymbol{R} \arrow[dl, "m"] \\
 & \boldsymbol{X} &
\end{tikzcd}$$

where $Q\pi$ is $(1 \xrightarrow{0} 2)$-injective

There are three other such correspondences. Pavlović was aware of one of these, restricted to lax functors into $\boldsymbol{spn}$ /functors into $\boldsymbol{X}$.

For systems the precedent of $\mathcal{W}$-enriched categories suggestes the feasibility of change-of-control (seldom considered for processes)

$$\begin{tikzcd}
\boldsymbol{X} \arrow[r, "H"] \arrow[dr, bend right, "L"'] & \boldsymbol{Y} \arrow[d, bend left, "M"] \arrow[dl, phantom, "\overset{\sigma}{\Swarrow}"] \\
 & \mathcal{W}
\end{tikzcd}
\quad \text{or} \quad
\begin{tikzcd}
\boldsymbol{X} \arrow[r, "H"] \arrow[dr, bend right, "L"'] & \boldsymbol{Y} \arrow[d, bend left, "M"] \arrow[dl, phantom, "\overset{\kappa}{\Nearrow}"] \\
 & \mathcal{W}
\end{tikzcd}$$

The precise nature of H (graph morphism, (pro)functor, or even a span of those?) remains to be determined.

14. Graph comprehension revisited

Contrast a lax transform $L \Longrightarrow M$ with the rightmost diagram for simulations, translated into the world of $\boldsymbol{X}$-controlled processes:

$$\begin{tikzcd}
\boldsymbol{X} \arrow[d, bend right=50, "L"'] \arrow[d, bend left=50, "M"] \arrow[d, phantom, "\Leftarrow"] \\
\boldsymbol{spn}
\end{tikzcd}
\qquad \text{vs.} \qquad
\begin{tikzcd}
& \boldsymbol{S} \arrow[dl, "Q\pi"'] \arrow[dr, "R\pi"] \arrow[dd, dashed, "U"] & \\
\boldsymbol{Q} \arrow[dr, "\ell"'] & & \boldsymbol{R} \arrow[dl, "m"] \\
& \boldsymbol{X} &
\end{tikzcd}$$

where $Q\pi$ is $(1 \xrightarrow{0} 2)$-injective

There are three other such correspondences. Pavlović was aware of one of these, restricted to lax functors into $\boldsymbol{spn}$ /functors into $\boldsymbol{X}$.

For systems the precedent of $\mathcal{W}$-enriched categories suggestes the feasibility of change-of-control (seldom considered for processes)

$$\begin{tikzcd}
\boldsymbol{X} \arrow[r, "H"] \arrow[dr, bend right, "L"'] & \boldsymbol{Y} \arrow[d, bend left, "M"] \arrow[dl, phantom, "\overset{\Swarrow}{\sigma}"] \\
& \mathcal{W}
\end{tikzcd}
\qquad \text{or} \qquad
\begin{tikzcd}
\boldsymbol{X} \arrow[r, "H"] \arrow[dr, bend right, "L"'] & \boldsymbol{Y} \arrow[d, bend left, "M"] \arrow[dl, phantom, "\overset{\kappa}{\Nearrow}"] \\
& \mathcal{W}
\end{tikzcd}$$

The precise nature of H (graph morphism, (pro)functor, or even a span of those?) remains to be determined.

14. Graph comprehension revisited

Contrast a lax transform $L \Longrightarrow M$ with the rightmost diagram for simulations, translated into the world of $\boldsymbol{X}$-controlled processes:

$$\begin{tikzcd} \boldsymbol{X} \arrow[d, bend right=60, "L"'] \arrow[d, bend left=60, "M"] \arrow[d, phantom, "\Leftarrow"] \\ \boldsymbol{spn} \end{tikzcd} \qquad \text{vs.} \qquad \begin{tikzcd} & \boldsymbol{S} \arrow[dl, "\color{green}{\boldsymbol{Q}\pi}"'] \arrow[dr, "\boldsymbol{R}\pi"] \arrow[dd, dashed, "U"] & \\ \boldsymbol{Q} \arrow[dr, "\ell"'] & & \boldsymbol{R} \arrow[dl, "m"] \\ & \boldsymbol{X} & \end{tikzcd}$$

where $\color{green}{\boldsymbol{Q}\pi}$ is $(1 \xrightarrow{0} 2)$-injective

There are three other such correspondences. Pavlović was aware of one of these, restricted to lax functors into $\boldsymbol{spn}$ /functors into $\boldsymbol{X}$.

For systems the precedent of $\mathcal{W}$-enriched categories suggestes the feasibility of change-of-control (seldom considered for processes)

$$\begin{tikzcd} \boldsymbol{X} \arrow[rr, "H"] \arrow[dr, bend right, "L"'] & \arrow[d, phantom, "\overset{\Swarrow}{\sigma}"] & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd} \qquad \text{or} \qquad \begin{tikzcd} \boldsymbol{X} \arrow[rr, "H"] \arrow[dr, bend right, "L"'] & \arrow[d, phantom, "\overset{\kappa}{\Nearrow}"] & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$

The precise nature of H (graph morphism, (pro)functor, or even a span of those?) remains to be determined.

14. Graph comprehension revisited

Contrast a lax transform $L \Longrightarrow M$ with the rightmost diagram for simulations, translated into the world of $\boldsymbol{X}$-controlled processes:

$$\begin{tikzcd}
\boldsymbol{X} \arrow[d, bend right, "L"'] \arrow[d, bend left, "M"] \arrow[d, phantom, "\Leftarrow"] \\
\boldsymbol{spn}
\end{tikzcd}
\quad \text{vs.} \quad
\begin{tikzcd}
& \boldsymbol{S} \arrow[dl, "Q\pi"'] \arrow[dr, "R\pi"] \arrow[d, dashed, "U"] & \\
\boldsymbol{Q} \arrow[r, "\ell"'] & \boldsymbol{X} & \boldsymbol{R} \arrow[l, "m"]
\end{tikzcd}$$

where $\boldsymbol{Q}\pi$ is
$(1 \xrightarrow{0} 2)$-injective

There are three other such correspondences. Pavlović was aware of one of these, restricted to lax functors into $\boldsymbol{spn}$ /functors into $\boldsymbol{X}$.

For systems the precedent of $\mathcal{W}$-enriched categories suggestes the feasibility of change-of-control (seldom considered for processes)

$$\begin{tikzcd}
\boldsymbol{X} \arrow[rr, "H"] \arrow[dr, "L"'] & \arrow[d, phantom, "\overset{\sigma}{\Swarrow}"] & \boldsymbol{Y} \arrow[dl, "M"] \\
& \mathcal{W} &
\end{tikzcd}
\quad \text{or} \quad
\begin{tikzcd}
\boldsymbol{X} \arrow[rr, "H"] \arrow[dr, "L"'] & \arrow[d, phantom, "\overset{\kappa}{\Nearrow}"] & \boldsymbol{Y} \arrow[dl, "M"] \\
& \mathcal{W} &
\end{tikzcd}$$

The precise nature of H (graph morphism, (pro)functor, or even a span of those?) remains to be determined.

14. Graph comprehension revisited

Contrast a lax transform $L \Longrightarrow M$ with the rightmost diagram for simulations, translated into the world of $\boldsymbol{X}$-controlled processes:

$$\begin{tikzcd}
\boldsymbol{X} \arrow[d, bend right, "L"'] \arrow[d, bend left, "M"] \arrow[d, phantom, "\Leftarrow"] \\
\boldsymbol{spn}
\end{tikzcd}
\quad \text{vs.} \quad
\begin{tikzcd}
& \boldsymbol{S} \arrow[dl, "Q\pi"'] \arrow[dr, "R\pi"] \arrow[dd, dashed, "U"] & \\
\boldsymbol{Q} \arrow[dr, "\ell"'] & & \boldsymbol{R} \arrow[dl, "m"] \\
& \boldsymbol{X} &
\end{tikzcd}$$

where $Q\pi$ is $(1 \xrightarrow{0} 2)$-injective

There are three other such correspondences. Pavlović was aware of one of these, restricted to lax functors into $\boldsymbol{spn}$ /functors into $\boldsymbol{X}$.

For systems the precedent of $\mathcal{W}$-enriched categories suggestes the feasibility of change-of-control (seldom considered for processes)

$$\begin{tikzcd}
\boldsymbol{X} \arrow[rr, "H"] \arrow[dr, bend right, "L"'] & \Swarrow_{\sigma} & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\
& \mathcal{W} &
\end{tikzcd}
\quad \text{or} \quad
\begin{tikzcd}
\boldsymbol{X} \arrow[rr, "H"] \arrow[dr, bend right, "L"'] & \Nearrow^{\kappa} & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\
& \mathcal{W} &
\end{tikzcd}$$

The precise nature of H (graph morphism, (pro)functor, or even a span of those?) remains to be determined.

14. Graph comprehension revisited

Contrast a lax transform $L \Longrightarrow M$ with the rightmost diagram for simulations, translated into the world of $\boldsymbol{X}$-controlled processes:

$$\begin{tikzcd} \boldsymbol{X} \arrow[d, bend right=60, "L"'] \arrow[d, bend left=60, "M"] \arrow[d, phantom, "\Leftarrow"] \\ \boldsymbol{spn} \end{tikzcd}$$

vs.

$$\begin{tikzcd} & \boldsymbol{S} \arrow[ld, "\boldsymbol{Q}\pi"'] \arrow[rd, "\boldsymbol{R}\pi"] \arrow[dd, dashed, "U"] & \\ \boldsymbol{Q} \arrow[rd, "\ell"'] & & \boldsymbol{R} \arrow[ld, "m"] \\ & \boldsymbol{X} & \end{tikzcd}$$

where $\boldsymbol{Q}\pi$ is $(1 \xrightarrow{0} 2)$-injective

There are three other such correspondences. Pavlović was aware of one of these, restricted to lax functors into $\boldsymbol{spn}$ /functors into $\boldsymbol{X}$.

For systems the precedent of $\mathcal{W}$-enriched categories suggestes the feasibility of change-of-control (seldom considered for processes)

$$\begin{tikzcd} \boldsymbol{X} \arrow[rr, "H"] \arrow[rd, bend right, "L"'] & \Swarrow_{\sigma} & \boldsymbol{Y} \arrow[ld, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$

or

$$\begin{tikzcd} \boldsymbol{X} \arrow[rr, "H"] \arrow[rd, bend right, "L"'] & \overset{\kappa}{\Nearrow} & \boldsymbol{Y} \arrow[ld, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$

The precise nature of H (graph morphism, (pro)functor, or even a span of those?) remains to be determined.

14. Graph comprehension revisited

Contrast a lax transform $L \Longrightarrow M$ with the rightmost diagram for simulations, translated into the world of $\boldsymbol{X}$-controlled processes:

$$\begin{tikzcd} \boldsymbol{X} \arrow[d, bend right, "L"'] \arrow[d, bend left, "M"] \arrow[d, phantom, "\Leftarrow"] \\ \boldsymbol{spn} \end{tikzcd} \quad \text{vs.} \quad \begin{tikzcd} & \boldsymbol{S} \arrow[dl, "Q\pi"'] \arrow[dr, "R\pi"] \arrow[dd, dashed, "U"] & \\ \boldsymbol{Q} \arrow[dr, "\ell"'] & & \boldsymbol{R} \arrow[dl, "m"] \\ & \boldsymbol{X} & \end{tikzcd}$$

where $\boldsymbol{Q}\pi$ is $(1 \xrightarrow{0} 2)$-injective

There are three other such correspondences. Pavlović was aware of one of these, restricted to lax functors into $\boldsymbol{spn}$ /functors into $\boldsymbol{X}$.

For systems the precedent of $\mathcal{W}$-enriched categories suggestes the feasibility of change-of-control (seldom considered for processes)

$$\begin{tikzcd} \boldsymbol{X} \arrow[rr, "H"] \arrow[dr, bend right, "L"'] & {} \arrow[d, phantom, "\Swarrow \sigma"] & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd} \quad \text{or} \quad \begin{tikzcd} \boldsymbol{X} \arrow[rr, "H"] \arrow[dr, bend right, "L"'] & {} \arrow[d, phantom, "\kappa \Nearrow"] & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$

The precise nature of H (graph morphism, (pro)functor, or even a span of those?) remains to be determined.

15. A change of paradigm!

Worytkiewizc interprets certain functors $\boldsymbol{X} \longrightarrow \mathcal{W}$ into a bicategory of spans as $\mathcal{W}$-controlled processes. The motivation goes back to Burstall's treatment of flow charts [Bur72]. Some advantages are:

- A "universal control" eliminates the need to change control;
- as a bicategory, $\mathcal{W}$ provides new types of control (rewriting?).

A good criterion for judging the suitability of different choices for H would seem to be the existence of saturations for σ and κ.

Proposition (for small $\boldsymbol{X}$ and $\mathcal{W}$ with local coproducts)

The saturation $\boldsymbol{X} \xrightarrow{L^{\wedge}} \mathcal{W}$ of $\boldsymbol{X} \xrightarrow{L} \mathcal{W}$ leaves the objects invariant and maps $x \xrightarrow{a} y$ in $\boldsymbol{X}$ to the coproduct of all $a_0L; a_1L; \ldots; a_{n-1}L$, where $a_0; a_1; \ldots; a_{n-1} = a$ in $\boldsymbol{X}$.

Saturation for $\mathcal{W} = \boldsymbol{rel}$ is idempotent; not so for $\mathcal{W} = \boldsymbol{spn}$.

15. A change of paradigm!

Worytkiewizc interprets certain functors $\boldsymbol{X} \longrightarrow \mathcal{W}$ into a bicategory of spans as $\mathcal{W}$-controlled processes. The motivation goes back to Burstall's treatment of flow charts [Bur72]. Some advantages are:

- A "universal control" eliminates the need to change control;
- as a bicategory, $\mathcal{W}$ provides new types of control (rewriting?).

A good criterion for judging the suitability of different choices for H would seem to be the existence of saturations for σ and κ.

Proposition (for small $\boldsymbol{X}$ and $\mathcal{W}$ with local coproducts)

The saturation $\boldsymbol{X} \xrightarrow{L^{\wedge}} \mathcal{W}$ of $\boldsymbol{X} \xrightarrow{L} \mathcal{W}$ leaves the objects invariant and maps $x \xrightarrow{a} y$ in $\boldsymbol{X}$ to the coproduct of all $a_0L; a_1L; \ldots; a_{n-1}L$, where $a_0; a_1; \ldots; a_{n-1} = a$ in $\boldsymbol{X}$.

Saturation for $\mathcal{W} = \boldsymbol{rel}$ is idempotent; not so for $\mathcal{W} = \boldsymbol{spn}$.

15. A change of paradigm!

Worytkiewizc interprets certain functors $\boldsymbol{X} \longrightarrow \mathcal{W}$ into a bicategory of spans as $\mathcal{W}$-controlled processes. The motivation goes back to Burstall's treatment of flow charts [Bur72]. Some advantages are:

- A "universal control" eliminates the need to change control;
- as a bicategory, $\mathcal{W}$ provides new types of control (rewriting?).

A good criterion for judging the suitability of different choices for H would seem to be the existence of saturations for σ and κ.

Proposition (for small $\boldsymbol{X}$ and $\mathcal{W}$ with local coproducts)

The saturation $\boldsymbol{X} \xrightarrow{L^{\wedge}} \mathcal{W}$ of $\boldsymbol{X} \xrightarrow{L} \mathcal{W}$ leaves the objects invariant and maps $x \xrightarrow{a} y$ in $\boldsymbol{X}$ to the coproduct of all $a_0 L; a_1 L; \ldots; a_{n-1} L$, where $a_0; a_1; \ldots; a_{n-1} = a$ in $\boldsymbol{X}$.

Saturation for $\mathcal{W} = \boldsymbol{rel}$ is idempotent; not so for $\mathcal{W} = \boldsymbol{spn}$.

15. A change of paradigm!

Worytkiewizc interprets certain functors $\boldsymbol{X} \longrightarrow \mathcal{W}$ into a bicategory of spans as $\mathcal{W}$-controlled processes. The motivation goes back to Burstall's treatment of flow charts [Bur72]. Some advantages are:

- A "universal control" eliminates the need to change control;
- as a bicategory, $\mathcal{W}$ provides new types of control (rewriting?).

A good criterion for judging the suitability of different choices for H would seem to be the existence of saturations for σ and κ.

Proposition (for small $\boldsymbol{X}$ and $\mathcal{W}$ with local coproducts)

The saturation $\boldsymbol{X} \xrightarrow{L^\wedge} \mathcal{W}$ of $\boldsymbol{X} \xrightarrow{L} \mathcal{W}$ leaves the objects invariant and maps $x \xrightarrow{a} y$ in $\boldsymbol{X}$ to the coproduct of all $a_0L; a_1L; \ldots; a_{n-1}L$, where $a_0; a_1; \ldots; a_{n-1} = a$ in $\boldsymbol{X}$.

Saturation for $\mathcal{W} = \boldsymbol{rel}$ is idempotent; not so for $\mathcal{W} = \boldsymbol{spn}$.

15. A change of paradigm!

Worytkiewizc interprets certain functors $\boldsymbol{X} \longrightarrow \mathcal{W}$ into a bicategory of spans as $\mathcal{W}$-controlled processes. The motivation goes back to Burstall's treatment of flow charts [Bur72]. Some advantages are:

- A "universal control" eliminates the need to change control;
- as a bicategory, $\mathcal{W}$ provides new types of control (rewriting?).

A good criterion for judging the suitability of different choices for H would seem to be the existence of saturations for σ and κ .

Proposition (for small $\boldsymbol{X}$ and $\mathcal{W}$ with local coproducts)

The saturation $\boldsymbol{X} \xrightarrow{L^{\wedge}} \mathcal{W}$ of $\boldsymbol{X} \xrightarrow{L} \mathcal{W}$ leaves the objects invariant and maps $x \xrightarrow{a} y$ in $\boldsymbol{X}$ to the coproduct of all $a_0 L; a_1 L; \ldots; a_{n-1} L$, where $a_0; a_1; \ldots; a_{n-1} = a$ in $\boldsymbol{X}$.

Saturation for $\mathcal{W} = \boldsymbol{rel}$ is idempotent; not so for $\mathcal{W} = \boldsymbol{spn}$.

15. A change of paradigm!

Worytkiewizc interprets certain functors $\boldsymbol{X} \longrightarrow \mathcal{W}$ into a bicategory of spans as $\mathcal{W}$-controlled processes. The motivation goes back to Burstall's treatment of flow charts [Bur72]. Some advantages are:

- A "universal control" eliminates the need to change control;
- as a bicategory, $\mathcal{W}$ provides new types of control (rewriting?).

A good criterion for judging the suitability of different choices for H would seem to be the existence of saturations for σ and κ.

Proposition (for small $\boldsymbol{X}$ and $\mathcal{W}$ with local coproducts)

The saturation $\boldsymbol{X} \xrightarrow{L\hat{}} \mathcal{W}$ of $\boldsymbol{X} \xrightarrow{L} \mathcal{W}$ leaves the objects invariant and maps $x \xrightarrow{a} y$ in $\boldsymbol{X}$ to the coproduct of all $a_0 L; a_1 L; \ldots; a_{n-1} L$, where $a_0; a_1; \ldots; a_{n-1} = a$ in $\boldsymbol{X}$.

Saturation for $\mathcal{W} = \boldsymbol{rel}$ is idempotent; not so for $\mathcal{W} = \boldsymbol{spn}$.

16. The universal property of saturation

Theorem

Saturation $|\boldsymbol{X}, \mathcal{W}|_{\mathrm{olx}} \longrightarrow \lfloor \boldsymbol{X}, \mathcal{W} \rfloor_{\mathrm{olx}}$ *(* $|\boldsymbol{X}, \mathcal{W}|_{\mathrm{lax}} \longrightarrow \lfloor \boldsymbol{X}, \mathcal{W} \rfloor_{\mathrm{lax}}$ *) is left (right) adjoint with (co)units based on coprojections.* □

- $$\begin{tikzcd} \boldsymbol{X} \arrow[rr, "H"] \arrow[dr, bend right, "L"'] & \arrow[d, phantom, "\overset{}{\Swarrow}\sigma"] & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$ or $$\begin{tikzcd} \boldsymbol{X} \arrow[rr, "H"] \arrow[dr, bend right, "L"'] & \arrow[d, phantom, "\kappa\Nearrow"] & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$ saturate, if σ is lax, respectively, κ is oplax
- $$\begin{tikzcd} & \boldsymbol{Z} \arrow[dl, bend right, "H_0"'] \arrow[dr, bend left, "H_1"] \arrow[dd, dashed, "U"] & \\ \boldsymbol{X} \arrow[r, phantom, "\overset{\sigma}{\Rightarrow}"] \arrow[dr, bend right, "L"'] & {} \arrow[r, phantom, "\overset{\kappa}{\Rightarrow}"] & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$ saturates, if σ is lax and κ is oplax
- $$\begin{tikzcd} & \boldsymbol{Z} \arrow[dl, bend right, "H_0"'] \arrow[dr, bend left, "H_1"] & \\ \boldsymbol{X} \arrow[dr, bend right, "L"'] & \underset{\sigma}{\Leftarrow} & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$ saturates, if σ is lax and H_0 is UFL, or if σ is oplax and H_1 is UFL.

16. The universal property of saturation

Theorem

Saturation $\lvert \boldsymbol{X}, \mathcal{W} \rvert_{\mathrm{olx}} \longrightarrow \lfloor \boldsymbol{X}, \mathcal{W} \rfloor_{\mathrm{olx}}$ *(* $\lvert \boldsymbol{X}, \mathcal{W} \rvert_{\mathrm{lax}} \longrightarrow \lfloor \boldsymbol{X}, \mathcal{W} \rfloor_{\mathrm{lax}}$ *) is left (right) adjoint with (co)units based on coprojections.* ☐

- $$\begin{tikzcd} \boldsymbol{X} \arrow[rr, "H"] \arrow[dr, bend right, "L"'] & \arrow[d, phantom, "\Swarrow \sigma"] & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$ or $$\begin{tikzcd} \boldsymbol{X} \arrow[rr, "H"] \arrow[dr, bend right, "L"'] & \arrow[d, phantom, "\kappa \Nearrow"] & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$ saturate, if σ is lax, respectively, κ is oplax

- $$\begin{tikzcd} & Z \arrow[dl, bend right, "H_0"'] \arrow[dr, bend left, "H_1"] \arrow[dd, dashed, "U" description] & \\ \boldsymbol{X} \arrow[dr, bend right, "L"'] \arrow[r, phantom, "\overset{\sigma}{\Rightarrow}"] & \arrow[r, phantom, "\overset{\kappa}{\Rightarrow}"] & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$ saturates, if σ is lax and κ is oplax

- $$\begin{tikzcd} & Z \arrow[dl, bend right, "H_0"'] \arrow[dr, bend left, "H_1"] & \\ \boldsymbol{X} \arrow[dr, bend right, "L"'] & \underset{\sigma}{\Leftarrow} & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$ saturates, if σ is lax and H_0 is UFL, or if σ is oplax and H_1 is UFL.

16. The universal property of saturation

Theorem

Saturation $|\boldsymbol{X},\mathcal{W}|_{\mathrm{olx}} \longrightarrow \lfloor \boldsymbol{X},\mathcal{W}\rfloor_{\mathrm{olx}}$ *(* $|\boldsymbol{X},\mathcal{W}|_{\mathrm{lax}} \longrightarrow \lfloor \boldsymbol{X},\mathcal{W}\rfloor_{\mathrm{lax}}$ *) is left (right) adjoint with (co)units based on coprojections.* □

- $$\begin{tikzcd} \boldsymbol{X} \arrow[rr, "H"] \arrow[dr, bend right, "L"'] & {} \arrow[d, phantom, "\Swarrow \sigma"] & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$ or $$\begin{tikzcd} \boldsymbol{X} \arrow[rr, "H"] \arrow[dr, bend right, "L"'] & {} \arrow[d, phantom, "\overset{\kappa}{\Nearrow}"] & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$ saturate, if σ is lax, respectively, κ is oplax

- $$\begin{tikzcd} & \boldsymbol{Z} \arrow[dl, bend right, "H_0"'] \arrow[dr, bend left, "H_1"] \arrow[d, dashed, "U"] & \\ \boldsymbol{X} \arrow[r, phantom, "\overset{\sigma}{\Rightarrow}"] \arrow[dr, bend right, "L"'] & U \arrow[d, dashed] \arrow[r, phantom, "\overset{\kappa}{\Rightarrow}"] & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$ saturates, if σ is lax and κ is oplax

- $$\begin{tikzcd} & \boldsymbol{Z} \arrow[dl, bend right, "H_0"'] \arrow[dr, bend left, "H_1"] & \\ \boldsymbol{X} \arrow[dr, bend right, "L"'] & \underset{\sigma}{\Leftarrow} & \boldsymbol{Y} \arrow[dl, bend left, "M"] \\ & \mathcal{W} & \end{tikzcd}$$ saturates, if σ is lax and H_0 is UFL, or if σ is oplax and H_1 is UFL.

16. The universal property of saturation

Theorem

Saturation $|\boldsymbol{X}, \mathcal{W}|_{\mathrm{olx}} \longrightarrow \lfloor \boldsymbol{X}, \mathcal{W} \rfloor_{\mathrm{olx}}$ *(* $|\boldsymbol{X}, \mathcal{W}|_{\mathrm{lax}} \longrightarrow \lfloor \boldsymbol{X}, \mathcal{W} \rfloor_{\mathrm{lax}}$ *) is left (right) adjoint with (co)units based on coprojections.* $\square$

- $$\begin{tikzcd} \boldsymbol{X} \arrow[rr, "H"] \arrow[dr, "L"'] & \overset{}{\underset{\sigma}{\Leftarrow}} & \boldsymbol{Y} \arrow[dl, "M"] \\ & \mathcal{W} & \end{tikzcd}$$ or $$\begin{tikzcd} \boldsymbol{X} \arrow[rr, "H"] \arrow[dr, "L"'] & \overset{\kappa}{\Rightarrow} & \boldsymbol{Y} \arrow[dl, "M"] \\ & \mathcal{W} & \end{tikzcd}$$ saturate, if σ is lax, respectively, κ is oplax
- $$\begin{tikzcd} & & \boldsymbol{Z} \arrow[dll, "H_0"'] \arrow[drr, "H_1"] \arrow[dd, dashed, "U"] & & \\ \boldsymbol{X} \arrow[drr, "L"'] & \overset{\sigma}{\Rightarrow} & & \overset{\kappa}{\Rightarrow} & \boldsymbol{Y} \arrow[dll, "M"] \\ & & \mathcal{W} & & \end{tikzcd}$$ saturates, if σ is lax and κ is oplax
- $$\begin{tikzcd} & & \boldsymbol{Z} \arrow[dll, "H_0"'] \arrow[drr, "H_1"] & & \\ \boldsymbol{X} \arrow[drr, "L"'] & & \underset{\sigma}{\Leftarrow} & & \boldsymbol{Y} \arrow[dll, "M"] \\ & & \mathcal{W} & & \end{tikzcd}$$ saturates, if σ is lax and H_0 is UFL, or if σ is oplax and H_1 is UFL.

18. Conclusion

- Simulations and bisimulations form a valuable contribution of theoretical computer science to general mathematics.
- They are too good to be left to computer scientists!

18. Conclusion

- Simulations and bisimulations form a valuable contribution of theoretical computer science to general mathematics.
- They are too good to be left to computer scientists!

Bibliography 0

[AM89] Peter Aczel and Max Mendler.
A final coalgebra theorem.
In *Category theory and computer science (Manchester, 1989)*, volume 389 of *Lecture Notes in Computer Science*, pages 357–365, Berlin, Heidelberg and New York, 1989. Springer-Verlag.

[AP97] Samson Abramsky and Dusko Pavlović.
Specifying interaction categories.
In E. Moggi and G. Rossolini, editors, *CTCS '97*, number 1290 in Lecture Notes in Computer Science, pages 147–158, Berlin, Heidelberg and New York, 1997. Springer-Verlag.

[BF00] Marta Bunge and Marcelo Fiore.
Unique factorization lifting functors and categories of linearly controlled processes.
Math. Structures Comput. Sci., 10:137–163, 2000.

[Bur72] Rod Burstall.
An algebraic description of programs with assertions, verification and simulation.
In J. Mack Adams, John Johnston, and Richard Stark, editors, *Conference on Proving Assertions about Programs*, pages 7–14. ACM, 1972.

[CKSW03] J.R.B. Cockett, J. Koslowski, R.A.G. Seely, and R.J. Wood.
Modules.
Theory Appl. Categ., 11(17):375–396, 2003.

[CS97] J. Robin B. Cockett and David A. Spooner.
Constructing process categories.
Theoret. Comput. Sci., 177(1):73–109, 1997.

Bibliography 1

[Err99] Lindsay Errington.
Twisted Systems.
PhD thesis, Imperial College, London, 1999.

[Gin68] Abraham Ginzburg.
Algebraic Theory of Automata.
Academic Press, 1968.

[GY63] Abraham Ginzburg and Michael Yoeli.
Products of Automata and the Problem of Covering.
Technical Report 15, Technion, Haifa, July 1963.

[GY65] Abraham Ginzburg and Michael Yoeli.
Products of Automata and the Problem of Covering.
Trans. Amer. Math. Soc., 116:253–266, April 1965.

[JNW94] André Joyal, Mogens Nielsen, and Glynn Winskel.
Bisimulation from open maps.
BRICS Reportseries RS-94-7, BRICS, May 1994.

[Lam58] Joachim Lambek,
Goursat's Theorem and the Zassenhaus Lemma.
Canad. J. Math., 10:45–56, 1958.

Bibliography 2

[Mil71] Robin Milner.
An algebraic definition of simulation between programs.
In *Proceedings of the 2nd International Joint Conference on Artificial Intelligence*, pages 481–489, London, sep 1971. British Computer Society.

[Par81] David Park.
Concurrency and automata on infinite sequences.
In P. Deussen, editor, *Theoretical Computer Science (Karlsruhe, 1981)*, volume 104 of *Lecture Notes in Computer Science*, pages 167–183. Springer-Verlag, 1981.

[Wor03] Krzysztof Worytkiewicz.
Paths and simulations.
In Rick Blute and Peter Selinger, editors, *Electronic Notes in Theoretical Computer Science*, volume 69. Elsevier, 2003.