

# Congruence for SOS with Data

Mohammad Mousavi, Michel Reniers, Jan Friso Groote

Department of Computer Science,  
Eindhoven University of Technology,

19<sup>th</sup> LICS, July 2004,  
Turku, Finland

## Overview

- Introduction: A Motivating Example
- 2. Transition System Specification and Bisimulation (with Data)
- 3. Congruence Formats (with Data)
- 4. Conclusion and Future Perspective (with Data!)

## A Motivating Example: Syntax

$$\text{prog} ::= \text{nop} \mid$$
$$\text{if } (bexp) \text{ then } v \leftarrow c \text{ fi} ; \text{prog}$$
$$bexp ::= \text{false} \mid \text{true} \mid \text{var} == \text{const}$$

N.B.

$$\text{if } (bexp) \text{ then } p \text{ fi} =_{def} \text{if } (bexp) \text{ then } p \text{ fi} ; \text{nop}$$
$$v \leftarrow c ; \text{prog} =_{def} \text{if } (\text{true}) \text{ then } v \leftarrow c \text{ fi} ; \text{prog}$$

## A Motivating Example: Semantics

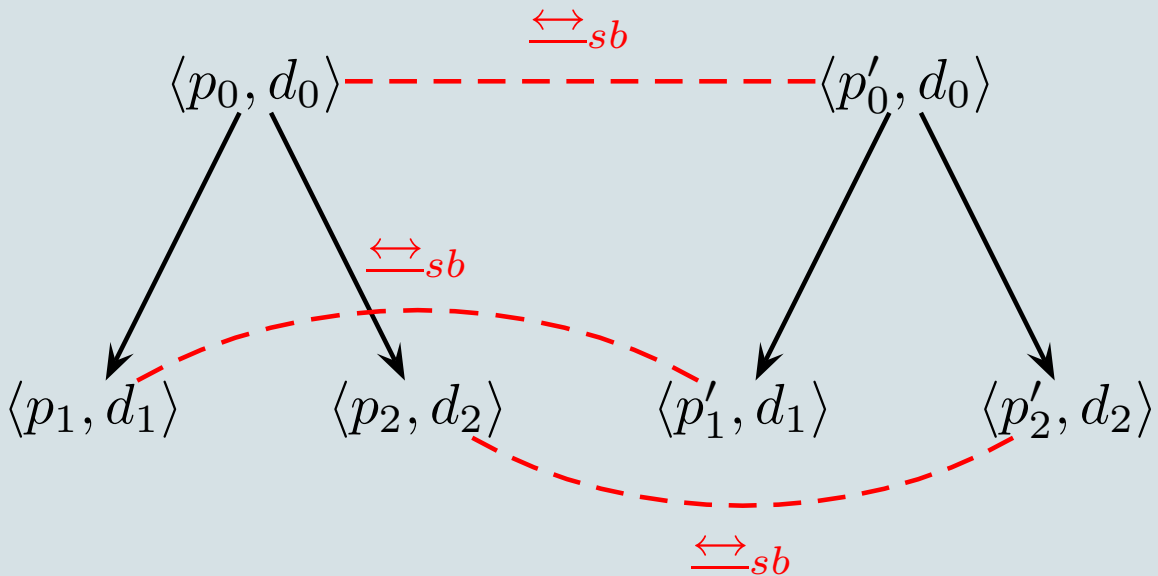
$$(\sigma : vars \mapsto const)$$

Semantics:

$$(S0) \quad \frac{\sigma \models bexp}{\langle if (bexp) then v \leftarrow c fi ; q, \sigma \rangle \rightarrow \langle q, \sigma[v \mapsto c] \rangle}$$

$$(S1) \quad \frac{\sigma \not\models bexp \quad \langle q, \sigma \rangle \rightarrow \langle q', \sigma' \rangle}{\langle if (bexp) then v \leftarrow c fi ; q, \sigma \rangle \rightarrow \langle q', \sigma' \rangle}$$

# A Motivating Example: Statebased Bisimulation



## A Motivating Example: A Disappointing Result

$\langle \text{if } (x == 2) \text{ then } x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$

$\xleftrightarrow{sb}$   
 $\langle \text{if } (x == 3) \text{ then } x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$

## A Motivating Example: A Disappointing Result

$$\langle \text{if } (x == 2) \text{ then } x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$$

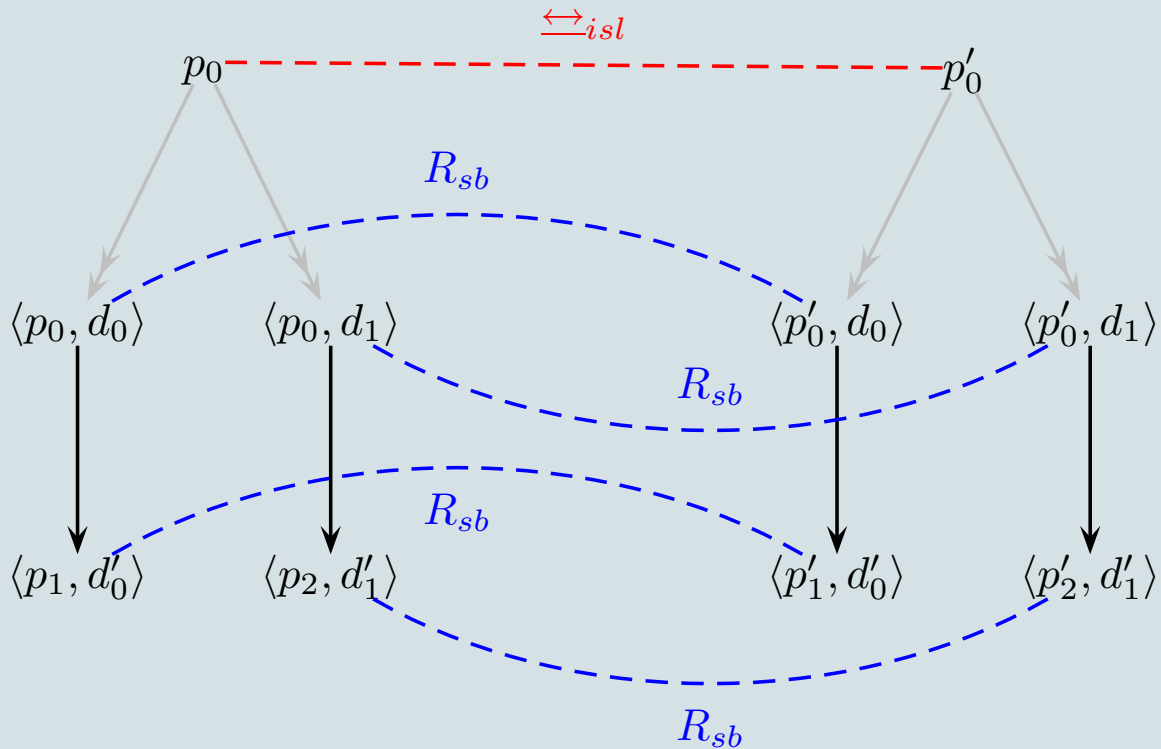
$$\langle \text{if } (x == 3) \text{ then } x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$$

But it **does not hold** that:

$$\langle x \leftarrow 2 ; \text{if } (x == 2) \text{ then } x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$$

$$\langle x \leftarrow 2 ; \text{if } (x == 3) \text{ then } x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$$

# A Motivating Example: Initially Stateless Bisimulation



# A Motivating Example: Introducing Parallelism

$$\begin{aligned} \text{prog} ::= & \text{nop} \quad | \\ & \text{if } \text{bexp} \text{ then } v \leftarrow c \text{ fi ;prog} \quad | \\ & \text{prog} \parallel \text{prog} \end{aligned}$$
$$\text{bexp} ::= \text{false} \mid \text{true} \mid (\text{var} == \text{const})$$

# A Motivating Example: Enriched Semantics

$$(P0) \frac{\langle p, \sigma \rangle \rightarrow \langle p', \sigma' \rangle}{\langle p \parallel q, \sigma \rangle \rightarrow \langle p' \parallel q, \sigma' \rangle}$$

$$(P1) \frac{\langle q, \sigma \rangle \rightarrow \langle q', \sigma' \rangle}{\langle p \parallel q, \sigma \rangle \rightarrow \langle p \parallel q', \sigma' \rangle}$$

# A Motivating Example: Another Disappointment

$x \leftarrow 2 ; \text{if } (x == 3) \text{ then } x \leftarrow 1 \text{ fi}$

$x \leftarrow 2 ; \text{if } (x \xrightarrow{isl} == 4) \text{ then } x \leftarrow 1 \text{ fi}$

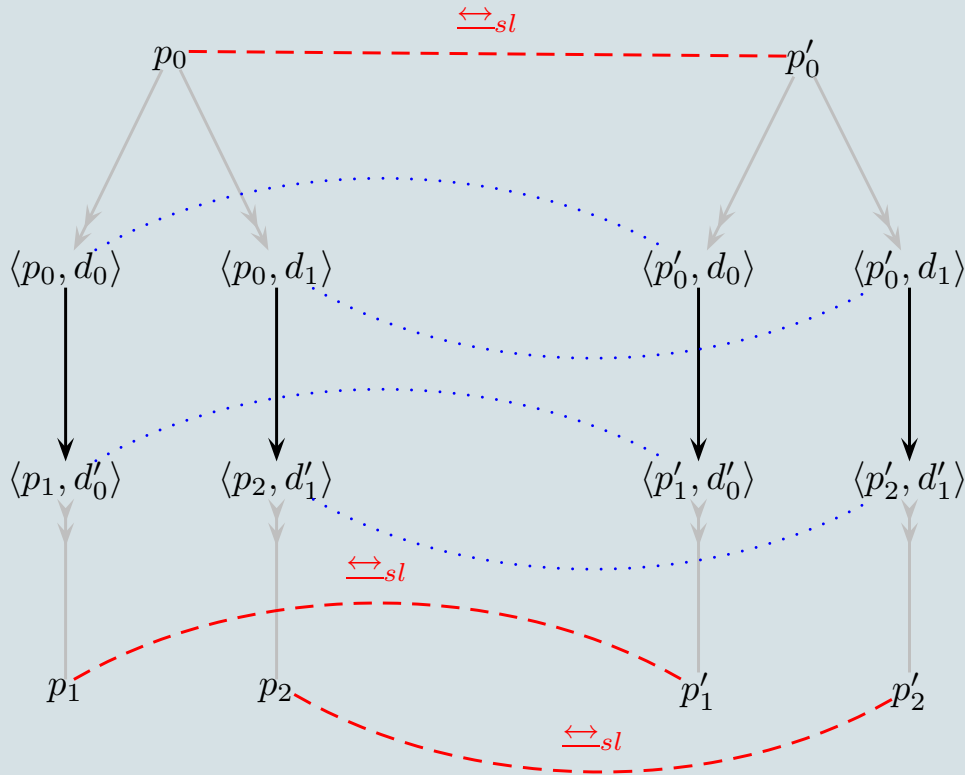
## A Motivating Example: Another Disappointment

$$x \leftarrow 2 ; \text{if } (x == 3) \text{ then } x \leftarrow 1 \text{ fi}$$
$$x \leftarrow 2 ; \text{if } (x \stackrel{\leftrightarrow_{isl}}{=} 4) \text{ then } x \leftarrow 1 \text{ fi}$$

But it **does not hold** that:

$$x \leftarrow 4 \parallel (x \leftarrow 2 ; \text{if } (x == 3) \text{ then } x \leftarrow 1 \text{ fi})$$
$$x \leftarrow 4 \parallel (x \leftarrow 2 ; \text{if } (x \stackrel{\leftrightarrow_{isl}}{=} 4) \text{ then } x \leftarrow 1 \text{ fi})$$

# A Motivating Example: Stateless Bisimulation



## Overview

- ✓ Introduction: A Motivating Example
- Transition System Specification and Bisimulation (with Data)
- 3. Congruence Formats (with Data)
- 4. Conclusion and Future Perspective (with Data!)

# Transition System Specification (with Data)

Syntax for processes:

- process variables :  $x_p, x_q, x_{p_0}, \dots \in V_p$
- process functions (with their arity):  $f_p, g_p, \dots \in \Sigma_p$
- open and closed process terms:  $t_p, t'_p, \dots \in T(\Sigma_p), p, q, \dots \in C(\Sigma_p)$

Syntax for data:

- data variables :  $x_d, x_{d_0}, \dots \in V_d$
- data functions (with their arity):  $f_d, f'_d, \dots \in \Sigma_d$
- open and closed process terms:  $t_d, t'_d, \dots \in T(\Sigma_d), d, d', \dots \in C(\Sigma_d)$

Syntax for states: pair of a process term and a data term.

$vars(t)$ : set of variables appearing in  $t$

# Transition system specification (with Data)

TSS with Data: A set of rules of the following form

$$\frac{\{\langle t_{p_i}, t_{d_i} \rangle \xrightarrow{l_i} \langle t'_{p_i}, t'_{d_i} \rangle \mid i \in I\}}{\langle t_p, t_d \rangle \xrightarrow{l} \langle t'_p, t'_d \rangle}$$

## Proof

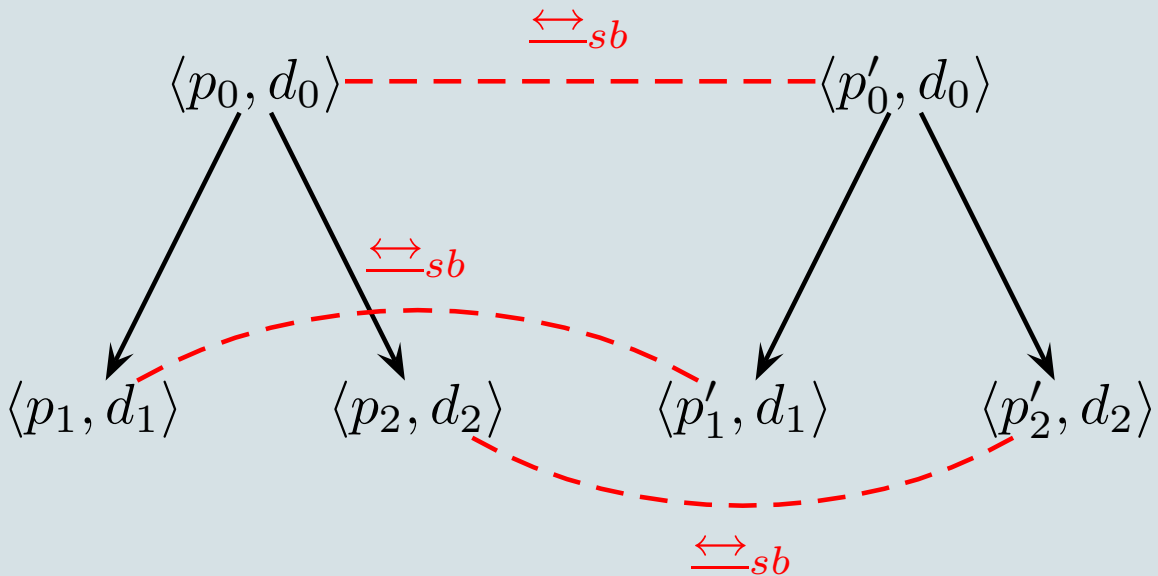
A transition is **provable** using a TSS if and only if:

there exists a **tree of finite depth** with nodes containing

**deduction rules** from the TSS and a **substitution** such that:

1. the valuation of the **conclusion of the root** matches **the proved transition**;
2. the valuation of an arbitrary **premise** of a node matches valuation of **conclusion of one of its offspring**.

# Statebased Bisimulation (Revisited)



## Statebased Bisimulation

$R_{sb}$  on states (with data) is a *statebased bisimulation* relation iff  
 $\forall_{p,q,d} (\langle p, d \rangle, \langle q, d' \rangle) \in R_{sb} \Rightarrow d = d' \wedge$

$$1. \quad \forall_{l,p',d''} \langle p, d \rangle \xrightarrow{l} \langle p', d'' \rangle \Rightarrow \exists_{q'} \quad \langle q, d \rangle \xrightarrow{l} \langle q', d'' \rangle \wedge \\ (\langle p', d'' \rangle, \langle q', d'' \rangle) \in R_{sb};$$

$$2. \quad \forall_{l,p',d''} \langle p, d \rangle \xrightarrow{l} \langle p', d'' \rangle \Rightarrow \exists_{q'} \quad \langle q, d \rangle \xrightarrow{l} \langle q', d'' \rangle \wedge \\ (\langle p', d'' \rangle, \langle q', d'' \rangle) \in R_{sb}.$$

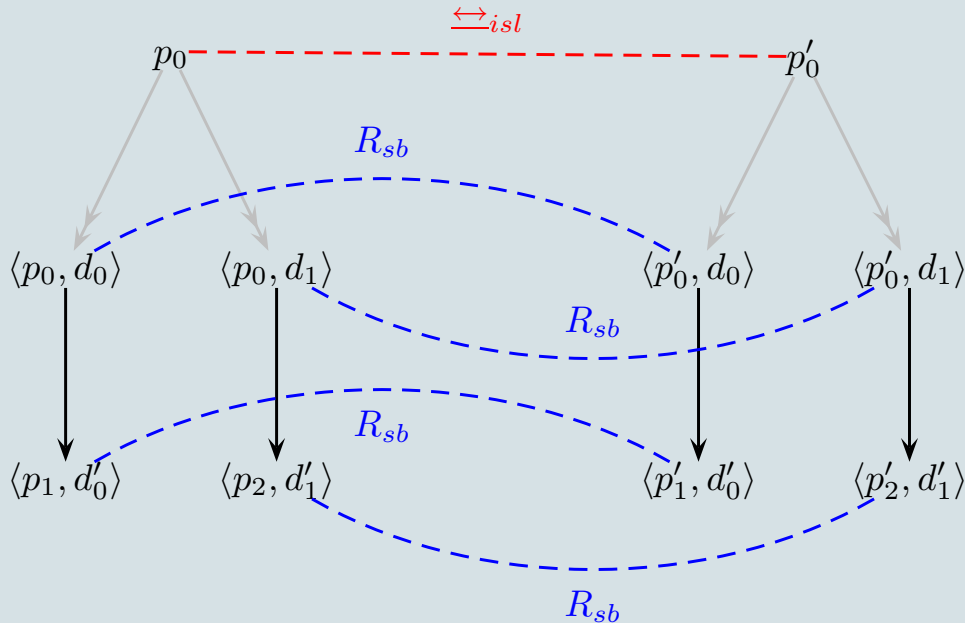
$\langle p, d \rangle$  and  $\langle q, d \rangle$  are *statebased bisimilar* ( $\langle p, d \rangle \xleftrightarrow{sb} \langle q, d \rangle$ ), iff there exists a  $R_{sb}$  s.t.  $(\langle p, d \rangle, \langle q, d \rangle) \in R_{sb}$ .

## Process-Congruence

For  $\sim \subseteq (C(\Sigma_p) \times C(\Sigma_d)) \times (C(\Sigma_p) \times C(\Sigma_d))$ ,  $\sim$  is called a *process-congruence* w.r.t.  $f \in \Sigma_p$  iff for all  $p_i, q_i$ , for all  $d$ , if  $\langle p_i, d \rangle \sim \langle q_i, d \rangle$  then  $\langle f(p_0, \dots, p_{n-1}), d \rangle \sim \langle f(q_0, \dots, q_{n-1}), d \rangle$ .

$\sim$  is a *process-congruence for a TSS* iff it is a process-congruence w.r.t. all process functions of the process signature.

# Initially Stateless Bisimulation (Revisited)



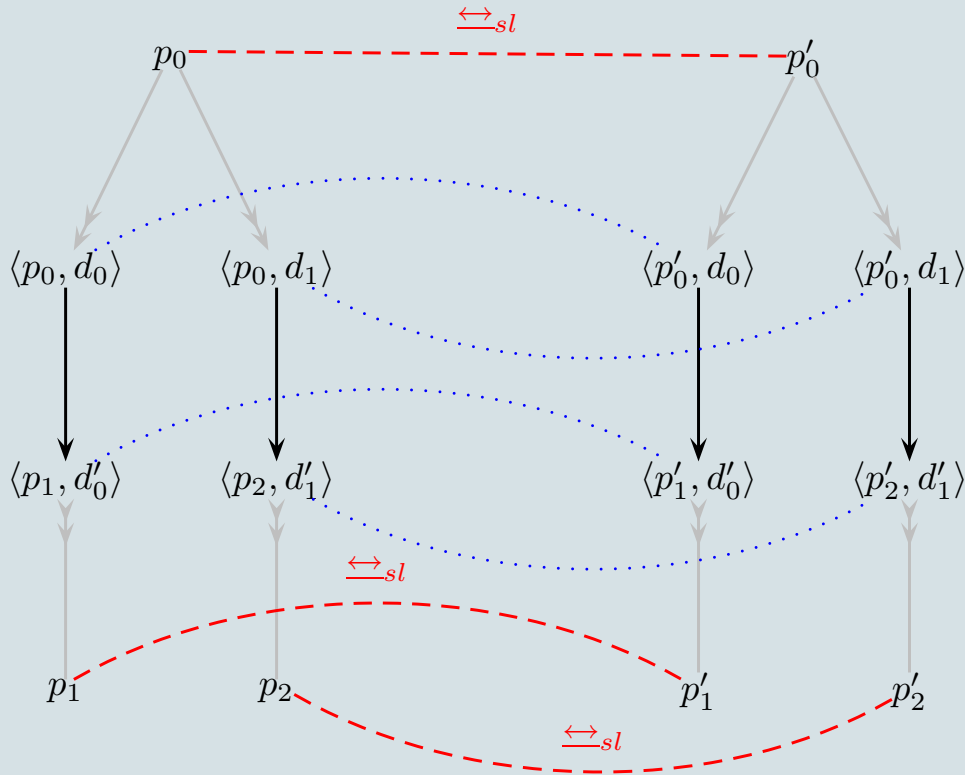
$p$  and  $q$  are *initially stateless bisimilar* ( $p \leftrightarrow_{isl} q$ ), iff there exists a  $R_{sb}$  such that  $(\langle p, d \rangle, \langle q, d \rangle) \in R_{sb}$ , for all  $d$ .

# Congruence

For  $\sim \subseteq C(\Sigma_p) \times C(\Sigma_p)$ ,  $\sim$  is called a *congruence w.r.t.  $f \in \Sigma_p$*  iff for all  $p_i, q_i$ , if  $p_i \sim q_i$  then  $f(p_0, \dots, p_{n-1}) \sim f(q_0, \dots, q_{n-1})$ .

$\sim$  is a *congruence for a TSS* iff it is a congruence w.r.t. all process functions of the process signature.

# Stateless Bisimulation (Revisited)



## Stateless Bisimulation

$R_{sl}$  on processes is a *stateless bisimulation* relation iff  $\forall_{p,q} (p, q) \in R_{sl} \Rightarrow$

1.  $\forall_{d,l,p',d'} \langle p, d \rangle \xrightarrow{l} \langle p', d' \rangle \Rightarrow \exists_{q'} \langle q, d \rangle \xrightarrow{l} \langle q', d' \rangle \wedge (p', q') \in R_{sl};$
2.  $\forall_{d,l,q',d'} \langle q, d \rangle \xrightarrow{l} \langle q', d' \rangle \Rightarrow \exists_{p'} \langle p, d \rangle \xrightarrow{l} \langle p', d' \rangle \wedge (p', q') \in R_{sl}.$

$p$  and  $q$  are *stateless bisimilar* ( $p \leftrightarrow_{sl} q$ ), iff there exists a  $R_{sl}$  s.t.  $(p, q) \in R_{sl}$ .

## Overview

- ✓ Introduction: A Motivating Example
- ✓ Transition System Specification and Bisimulation (with Data)
- Congruence Formats (with Data)
- 4. Conclusion and Future Perspective (with Data!)

## Congruence Formats (with Data)

Tyft format: a set of rules of the following form

$$\frac{\{t_{p_i} \xrightarrow{l_i} y_{p_i} \mid i \in I\}}{f_p(x_{p_0}, \dots, x_{p_{n-1}}) \xrightarrow{l} t'_p}$$

where all  $y_i$ 's and  $x_{p_i}$ 's are distinct variables.

Extensions of TSS (not dealt with in this talk but present in the paper):

- deduction rules in tyxt format
- predicates
- negative premises

# Transition System Specification (with Data)

Process-tyft:

$$(dr) \frac{\{\langle t_{p_i}, t_{d_i} \rangle \xrightarrow{l_i} \langle y_{p_i}, t'_{d_i} \rangle \mid i \in I\}}{\langle f_p(x_{p_0}, \dots, x_{p_{n-1}}), t_d \rangle \xrightarrow{l} \langle t'_p, t'_d \rangle}$$

Theorem: if all deduction rules of a TSS are in **process-tyft** format, then  $\xleftrightarrow{sl}$  is a **congruence**.

## Example: programming language

$$(S0) \frac{\sigma \models bexp}{\langle \text{if } (bexp) \text{ then } v \leftarrow c \text{ fi} ; q, \sigma \rangle \rightarrow \langle q, \sigma[v \mapsto c] \rangle}$$

$$(S1) \frac{\sigma \not\models bexp \quad \langle q, \sigma \rangle \rightarrow \langle q', \sigma' \rangle}{\langle \text{if } (bexp) \text{ then } v \leftarrow c \text{ fi} ; q, \sigma \rangle \rightarrow \langle q', \sigma' \rangle}$$

$$(P0) \frac{\langle p, \sigma \rangle \rightarrow \langle p', \sigma' \rangle}{\langle p \parallel q, \sigma \rangle \rightarrow \langle p' \parallel q, \sigma' \rangle}$$

$$(P1) \frac{\langle q, \sigma \rangle \rightarrow \langle q', \sigma' \rangle}{\langle p \parallel q, \sigma \rangle \rightarrow \langle p \parallel q', \sigma' \rangle}$$

All deduction rules are in process-tyft format.

Does this format also apply to  $\xrightarrow{sb}$  and  $\xrightarrow{isl}$ ? Let's re-visit them.

# A Motivating Example: A Disappointing Result (revisited)

$$\langle \text{if } (x == 2) \text{ then } x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$$
$$\xleftrightarrow{sb}$$
$$\langle \text{if } (x == 3) \text{ then } x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$$

But it **does not hold** that:

$$\langle x \leftarrow 2 ; \text{if } (x == 2) \text{ then } x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$$
$$\xleftrightarrow{sb}$$
$$\langle x \leftarrow 2 ; \text{if } (x == 3) \text{ then } x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$$

## A Motivating Example: A Disappointing Result (revisited)

$$\langle \text{if } (x == 2) \text{ then } x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$$

$$\langle \text{if } (x == 3) \text{ then } \xleftrightarrow{sb} x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$$

But it **does not hold** that:

**Why?**

$$\langle x \leftarrow 2 ; \text{if } (x == 2) \text{ then } x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$$

$$\langle x \leftarrow 2 ; \text{if } (x == 3) \text{ then } \xleftrightarrow{sb} x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$$

# A Motivating Example: A Disappointing Result

Since:

1. Original process terms

$(if\ (x == 2)\ then\ x \leftarrow 1\ fi)$  and

$(if\ (x == 3)\ then\ x \leftarrow 1\ fi)$  are only related using  $[x \mapsto 1]$

2. and for example rule (S0) allows for change of the data state while keeping the process part:

$$(S0) \frac{\sigma \models bexp}{\langle if\ (bexp)\ then\ v \leftarrow c\ fi ; q, \sigma \rangle \rightarrow \langle q, \sigma[v \mapsto c] \rangle}$$

To have **congruence for statebased** bisimilarity, the rules should **maintain the data dependencies**.

# Standard Format for Statebased

Data dependency:

$\langle t_p, t_d \rangle \models x_p \Rightarrow t'_d$ , if and only if  $x_p \in vars(t_p)$  and  $t'_d = t_d$

I.  $\forall x_p \in X_p \ (t'_p, t'_d) \models x_p \Rightarrow t'_d \Rightarrow \langle f_p(x_{p_0}, \dots, x_{p_{n-1}}), t_d \rangle \models x_p \Rightarrow t'_d$

Data dependencies should flow:

$$(dr) \quad \frac{\{ \langle t_{p_i}, t_{d_i} \rangle \xrightarrow{l_i} \langle y_{p_i}, t'_{d_i} \rangle \mid i \in I \}}{\langle f_p(x_{p_0}, \dots, x_{p_{n-1}}), t_d \rangle \xleftarrow{l} \langle t'_p, t'_d \rangle}$$

## Standard Format for Statebased

1.  $\forall x_p \in X_p \langle t'_p, t'_d \rangle \models x_p \Rightarrow t'_d \Rightarrow \langle f_p(x_{p_0}, \dots, x_{p_{n-1}}), t_d \rangle \models x_p \Rightarrow t'_d$
2.  $\forall x_p \in Y_p \langle t'_p, t'_d \rangle \models x_p \Rightarrow t'_d \Rightarrow \exists_{i \in I} \langle y_{p_i}, t'_{d_i} \rangle \models x_p \Rightarrow t'_d$

Data dependencies should flow:

$$(dr) \quad \frac{\{ \langle t_{p_i}, t_{d_i} \rangle \xrightarrow{l_i} \langle y_{p_i}, t'_{d_i} \rangle \mid i \in I \}}{\langle f_p(x_{p_0}, \dots, x_{p_{n-1}}), t_d \rangle \xrightarrow{l} \langle t'_p, t'_d \rangle}$$

# Standard Format for Statebased

1.  $\forall x_p \in X_p \ (t'_p, t'_d) \models x_p \Rightarrow t'_d \Rightarrow \langle f_p(x_{p_0}, \dots, x_{p_{n-1}}), t_d \rangle \models x_p \Rightarrow t'_d$
2.  $\forall x_p \in Y_p \ (t'_p, t'_d) \models x_p \Rightarrow t'_d \Rightarrow \exists i \in I \ \langle y_{p_i}, t'_{d_i} \rangle \models x_p \Rightarrow t'_d$
3.  $\forall i \in I, x_p \in X_p \ (t_{p_i}, t_{d_i}) \models x_p \Rightarrow t_{d_i} \Rightarrow \langle f_p(x_{p_0}, \dots, x_{p_{n-1}}), t_d \rangle \models x_p \Rightarrow t_{d_i}$

Data dependencies should flow:

$$(dr) \quad \frac{\{\langle t_{p_i}, t_{d_i} \rangle \xrightarrow{l_i} \langle y_{p_i}, t'_{d_i} \rangle \mid i \in I\}}{\langle f_p(x_{p_0}, \dots, x_{p_{n-1}}), t_d \rangle \xrightarrow{l} \langle t'_p, t'_d \rangle}$$

# Standard Format for Statebased

1.  $\forall x_p \in X_p \ (t'_p, t'_d) \models x_p \Rightarrow t'_d \Rightarrow \langle f_p(x_{p_0}, \dots, x_{p_{n-1}}), t_d \rangle \models x_p \Rightarrow t'_d$
2.  $\forall x_p \in Y_p \ (t'_p, t'_d) \models x_p \Rightarrow t'_d \Rightarrow \exists_{i \in I} \langle y_{p_i}, t'_{d_i} \rangle \models x_p \Rightarrow t'_d$
3.  $\forall_{i \in I, x_p \in X_p} \ (t_{p_i}, t_{d_i}) \models x_p \Rightarrow t_{d_i} \Rightarrow \langle f_p(x_{p_0}, \dots, x_{p_{n-1}}), t_d \rangle \models x_p \Rightarrow t_{d_i}$
4.  $\forall_{i \in I, x_p \in Y_p} \ (t_{p_i}, t_{d_i}) \models x_p \Rightarrow t_{d_i} \Rightarrow \exists_{j \in I} \langle y_{p_j}, t'_{d_j} \rangle \models x_p \Rightarrow t_{d_i}$

Data dependencies should flow:

$$(dr) \quad \frac{\left\{ \langle t_{p_i}, t_{d_i} \rangle \xrightarrow{l_i} \langle y_{p_i}, t'_{d_i} \rangle \mid i \in I \right\}}{\langle f_p(x_{p_0}, \dots, x_{p_{n-1}}), t_d \rangle \xrightarrow{l} \langle t'_p, t'_d \rangle}$$

## Standard Format for Statebased

Theorem: if all deduction rules of a TSS are in **process-tyft** format and **the data-dependencies flow** as before, then  $\xrightarrow{sb}$  is a congruence.

The above format works for  $\xrightarrow{isl}$ , as well. However, we can **relax** some of the **data-dependency** constraints there:

$$(S0) \quad \frac{\sigma \models bexp}{\langle if (bexp) then v \leftarrow c fi; q, \sigma \rangle \rightarrow \langle q, \sigma[v \mapsto c] \rangle}$$

the argument of  $;$  are related for all data states.

## Standard Format for Initially Stateless

1.  $\forall_{x_p \in Y_p} (t'_p, t'_d) \models x_p \Rightarrow t'_d \Rightarrow \exists_{i \in I} \langle y_{p_i}, t'_{d_i} \rangle \models x_p \Rightarrow t'_d$
2.  $\forall_{i \in I, x_p \in Y_p} (t_{p_i}, t_{d_i}) \models x_p \Rightarrow t_{d_i} \Rightarrow \exists_{j \in I} \langle y_{p_j}, t'_{d_j} \rangle \models x_p \Rightarrow t_{d_i}$

Data dependencies should flow:

$$(dr) \quad \frac{\left\{ \langle t_{p_i}, t_{d_i} \rangle \xrightarrow{l_i} \langle y_{p_i}, t'_{d_i} \rangle \mid i \in I \right\}}{\langle f_p(x_{p_0}, \dots, x_{p_{n-1}}), t_d \rangle \xrightarrow{l} \langle t'_p, t'_d \rangle}$$

A rule may violate the two other data-dependency flows if:

1. those variables **violating the property** are the **initial** ones;
2. the **places** containing **initial** variables are **fixed** in all rules.

## Example: programming language (revisited)

$$(S0) \frac{\sigma \models bexp}{\langle \text{if } (bexp) \text{ then } v \leftarrow c \text{ fi} ; q, \sigma \rangle \rightarrow \langle q, \sigma[v \mapsto c] \rangle}$$

$$(S1) \frac{\sigma \not\models bexp \quad \langle q, \sigma \rangle \rightarrow \langle q', \sigma' \rangle}{\langle \text{if } (bexp) \text{ then } v \leftarrow c \text{ fi} ; q, \sigma \rangle \rightarrow \langle q', \sigma' \rangle}$$

$$(P0) \frac{\langle p, \sigma \rangle \rightarrow \langle p', \sigma' \rangle}{\langle p \parallel q, \sigma \rangle \rightarrow \langle p' \parallel q, \sigma' \rangle}$$

$$(P1) \frac{\langle q, \sigma \rangle \rightarrow \langle q', \sigma' \rangle}{\langle p \parallel q, \sigma \rangle \rightarrow \langle p \parallel q', \sigma' \rangle}$$

## A Motivating Example: Another Disappointment (revisited)

$$\langle x \leftarrow 2 ; \text{if } (x == 3) \text{ then } x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$$

$$\langle x \leftarrow 2 ; \text{if } (x == 4) \text{ then } x \leftarrow 1 \text{ fi}, [x \mapsto 1] \rangle$$

$\xleftrightarrow{isl}$

But it **does not hold** that:

$$\langle x \leftarrow 4 \parallel (x \leftarrow 2 ; \text{if } (x == 3) \text{ then } x \leftarrow 1 \text{ fi}), [x \mapsto 1] \rangle$$

$$\langle x \leftarrow 4 \parallel (x \leftarrow 2 ; \text{if } (x == 4) \text{ then } x \leftarrow 1 \text{ fi}), [x \mapsto 1] \rangle$$

$\xleftrightarrow{isl}$

## Overview

- ✓ Introduction: A Motivating Example
- ✓ Transition System Specification and Bisimulation (with Data)
- ✓ Congruence Formats (with Data)
- Conclusion and Future Perspective (with Data!)

## Conclusion

- formats for establishing **congruence** for **three kinds of bisimilarity** on TSS with **data**
- **stateless and initially stateless** bisimilarity are encountered in literature (for example in **HyPA, discrete and hybrid  $\chi$ , timed  $\mu$ CRL**)
- case studies:
  - HyPA (uses  $\leftrightarrow_{sl}$ ):  $\leftrightarrow_{sl}$  is a congruence,  $\leftrightarrow_{isl}$  is a congruence for sequential-HyPA
  - timed  $\mu$ CRL (uses  $\leftrightarrow_{isl}$ ):  $\leftrightarrow_{sl}$  is a congruence,  $\leftrightarrow_{isl}$  is a congruence (after reformulation of the deduction rules)
  - Linda:  $\leftrightarrow_{sl}$  is a congruence,  $\leftrightarrow_{isl}$  is a congruence with respect to the sequential subset

## Future (Ongoing) Work

- categorical and co-algebraic interpretation of bisimulation and the role of data
- extension of other (weaker) notions of bisimulation
- deriving equational theories from TSS

## Overview

- ✓ Introduction: A Motivating Example
- ✓ Transition System Specification and Bisimulation (with Data)
- ✓ Congruence Formats (with Data)
- ✓ Conclusion and Future Perspective (with Data!)

**Thank You!**