



# Concurrency Theory

Winter Semester 2019/20

Lecture 12: Strong Bisimulation

Joost-Pieter Katoen and Thomas Noll  
Software Modeling and Verification Group  
RWTH Aachen University

<https://moves.rwth-aachen.de/teaching/ws-19-20/ct/>

# Recap: Trace Equivalence

---

## Outline of Lecture 12

Recap: Trace Equivalence

Bisimulation

Bisimulation and Trace Equivalence

Bisimulation is a Congruence

# Recap: Trace Equivalence

---

## Trace Equivalence

- Trace equivalence is a possible behavioural equivalence, is a congruence, but does **not preserve deadlocks**.

# Recap: Trace Equivalence

---

## Trace Equivalence

- Trace equivalence is a possible behavioural equivalence, is a congruence, but does **not preserve deadlocks**.
- Main problem:

$$\alpha.(P + Q) \equiv \alpha.P + \alpha.Q,$$

whereas their deadlock behaviour in a context can differ.

# Recap: Trace Equivalence

---

## Trace Equivalence

- Trace equivalence is a possible behavioural equivalence, is a congruence, but does **not preserve deadlocks**.
- Main problem:

$$\alpha.(P + Q) \equiv \alpha.P + \alpha.Q,$$

whereas their deadlock behaviour in a context can differ.

- Solution: consider finer behavioural equivalences such that:

$$\alpha.(P + Q) \not\equiv \alpha.P + \alpha.Q$$

# Recap: Trace Equivalence

---

## Trace Equivalence

- Trace equivalence is a possible behavioural equivalence, is a congruence, but does **not preserve deadlocks**.
- Main problem:

$$\alpha.(P + Q) \equiv \alpha.P + \alpha.Q,$$

whereas their deadlock behaviour in a context can differ.

- Solution: consider finer behavioural equivalences such that:

$$\alpha.(P + Q) \not\equiv \alpha.P + \alpha.Q$$

- Our (serious) attempt today: Milner's **strong bisimulation**.



Robin Milner (1934–2010)

# Bisimulation

---

## Outline of Lecture 12

Recap: Trace Equivalence

Bisimulation

Bisimulation and Trace Equivalence

Bisimulation is a Congruence

## Rationale

### Observation

In order for a behavioural equivalence to be deadlock sensitive, it has to take the **branching structure** of processes into account.



# Bisimulation

---

## Rationale

### Observation

In order for a behavioural equivalence to be deadlock sensitive, it has to take the **branching structure** of processes into account.

This is achieved by an equivalence that is defined according to the following scheme:

### Bisimulation scheme

$P, Q \in Prc$  are equivalent iff, for every action  $\alpha$ , every  $\alpha$ -successor of  $P$  is equivalent to some  $\alpha$ -successor of  $Q$ , and vice versa.

# Bisimulation

---

## Rationale

### Observation

In order for a behavioural equivalence to be deadlock sensitive, it has to take the **branching structure** of processes into account.

This is achieved by an equivalence that is defined according to the following scheme:

### Bisimulation scheme

$P, Q \in \text{Prc}$  are equivalent iff, for every action  $\alpha$ , every  $\alpha$ -successor of  $P$  is equivalent to some  $\alpha$ -successor of  $Q$ , and vice versa.

Three versions will be considered in this course:

1. **Strong** bisimulation: ignore the special function of  $\tau$ -actions
2. **Weak** bisimulation: treat  $\tau$ -actions as invisible
3. **Simulation** relations: unidirectional versions of bisimulation

## Strong Bisimulation I

Definition 12.1 (Strong bisimulation)

(Park 1981, Milner 1989)

A binary relation  $\rho \subseteq Prc \times Prc$  is a **strong bisimulation** whenever for every  $(P, Q) \in \rho$  and  $\alpha \in Act$ :

1. if  $P \xrightarrow{\alpha} P'$ , then there exists  $Q' \in Prc$  such that  $Q \xrightarrow{\alpha} Q'$  and  $P' \rho Q'$ , and
2. if  $Q \xrightarrow{\alpha} Q'$ , then there exists  $P' \in Prc$  such that  $P \xrightarrow{\alpha} P'$  and  $P' \rho Q'$ .

# Bisimulation

---

## Strong Bisimulation I

Definition 12.1 (Strong bisimulation)

(Park 1981, Milner 1989)

A binary relation  $\rho \subseteq Prc \times Prc$  is a **strong bisimulation** whenever for every  $(P, Q) \in \rho$  and  $\alpha \in Act$ :

1. if  $P \xrightarrow{\alpha} P'$ , then there exists  $Q' \in Prc$  such that  $Q \xrightarrow{\alpha} Q'$  and  $P' \rho Q'$ , and
2. if  $Q \xrightarrow{\alpha} Q'$ , then there exists  $P' \in Prc$  such that  $P \xrightarrow{\alpha} P'$  and  $P' \rho Q'$ .

**Note:** strong bisimulations are not necessarily equivalences

# Bisimulation

## Strong Bisimulation I

Definition 12.1 (Strong bisimulation)

(Park 1981, Milner 1989)

A binary relation  $\rho \subseteq \text{Prc} \times \text{Prc}$  is a **strong bisimulation** whenever for every  $(P, Q) \in \rho$  and  $\alpha \in \text{Act}$ :

1. if  $P \xrightarrow{\alpha} P'$ , then there exists  $Q' \in \text{Prc}$  such that  $Q \xrightarrow{\alpha} Q'$  and  $P' \rho Q'$ , and
2. if  $Q \xrightarrow{\alpha} Q'$ , then there exists  $P' \in \text{Prc}$  such that  $P \xrightarrow{\alpha} P'$  and  $P' \rho Q'$ .

**Note:** strong bisimulations are not necessarily equivalences

Definition 12.2 (Strong bisimilarity)

Processes  $P$  and  $Q$  are **strongly bisimilar**, denoted  $P \sim Q$ , iff there is a strong bisimulation  $\rho$  with  $P \rho Q$ .

# Bisimulation

## Strong Bisimulation I

Definition 12.1 (Strong bisimulation)

(Park 1981, Milner 1989)

A binary relation  $\rho \subseteq \text{Prc} \times \text{Prc}$  is a **strong bisimulation** whenever for every  $(P, Q) \in \rho$  and  $\alpha \in \text{Act}$ :

1. if  $P \xrightarrow{\alpha} P'$ , then there exists  $Q' \in \text{Prc}$  such that  $Q \xrightarrow{\alpha} Q'$  and  $P' \rho Q'$ , and
2. if  $Q \xrightarrow{\alpha} Q'$ , then there exists  $P' \in \text{Prc}$  such that  $P \xrightarrow{\alpha} P'$  and  $P' \rho Q'$ .

**Note:** strong bisimulations are not necessarily equivalences

Definition 12.2 (Strong bisimilarity)

Processes  $P$  and  $Q$  are **strongly bisimilar**, denoted  $P \sim Q$ , iff there is a strong bisimulation  $\rho$  with  $P \rho Q$ . Thus,

$$\sim = \bigcup \{ \rho \mid \rho \text{ is a strong bisimulation} \}.$$

# Bisimulation

## Strong Bisimulation I

Definition 12.1 (Strong bisimulation)

(Park 1981, Milner 1989)

A binary relation  $\rho \subseteq \text{Prc} \times \text{Prc}$  is a **strong bisimulation** whenever for every  $(P, Q) \in \rho$  and  $\alpha \in \text{Act}$ :

1. if  $P \xrightarrow{\alpha} P'$ , then there exists  $Q' \in \text{Prc}$  such that  $Q \xrightarrow{\alpha} Q'$  and  $P' \rho Q'$ , and
2. if  $Q \xrightarrow{\alpha} Q'$ , then there exists  $P' \in \text{Prc}$  such that  $P \xrightarrow{\alpha} P'$  and  $P' \rho Q'$ .

**Note:** strong bisimulations are not necessarily equivalences

Definition 12.2 (Strong bisimilarity)

Processes  $P$  and  $Q$  are **strongly bisimilar**, denoted  $P \sim Q$ , iff there is a strong bisimulation  $\rho$  with  $P \rho Q$ . Thus,

$$\sim = \bigcup \{ \rho \mid \rho \text{ is a strong bisimulation} \}.$$

Relation  $\sim$  is called **strong bisimilarity**.

# Bisimulation

---

## Strong Bisimulation II

$$P \xrightarrow{\alpha} P'$$

$\rho$

$Q$

can be completed to

$$P \xrightarrow{\alpha} P'$$

$\rho$

$\rho'$

$$Q \xrightarrow{\alpha} Q'$$



# Bisimulation

---

## Strong Bisimulation II

$$P \xrightarrow{\alpha} P'$$

$\rho$

$Q$

can be completed to

$$P \xrightarrow{\alpha} P'$$

$\rho$        $\rho$

$$Q \xrightarrow{\alpha} Q'$$

and

$P$

$\rho$

$$Q \xrightarrow{\alpha} Q'$$

can be completed to

$$P \xrightarrow{\alpha} P'$$

$\rho$        $\rho$

$$Q \xrightarrow{\alpha} Q'$$

# Bisimulation

---

## Examples

### Example 12.3 (A first example)

Claim:  $P \sim Q$  where

$$\begin{array}{ll} P = a.P_1 + a.P_2 & Q = a.Q_1 \\ P_1 = b.P_2 & Q_1 = b.Q_1 \\ P_2 = b.P_2 & \end{array}$$

# Bisimulation

---

## Examples

### Example 12.3 (A first example)

Claim:  $P \sim Q$  where

$$\begin{array}{ll} P = a.P_1 + a.P_2 & Q = a.Q_1 \\ P_1 = b.P_2 & Q_1 = b.Q_1 \\ P_2 = b.P_2 & \end{array}$$

Proof:  $\rho = \{(P, Q), (P_1, Q_1), (P_2, Q_1)\}$  is a strong bisimulation

# Bisimulation

---

## Examples

### Example 12.3 (A first example)

Claim:  $P \sim Q$  where

$$\begin{array}{ll} P = a.P_1 + a.P_2 & Q = a.Q_1 \\ P_1 = b.P_2 & Q_1 = b.Q_1 \\ P_2 = b.P_2 & \end{array}$$

Proof:  $\rho = \{(P, Q), (P_1, Q_1), (P_2, Q_1)\}$  is a strong bisimulation

### Example 12.4 (Relating a finite to an infinite-state process)

Claim:  $P_0 \sim Q$  where  $P_i = a.P_{i+1}$  for  $i \in \mathbb{N}$  and  $Q = a.Q$ .

# Bisimulation

---

## Examples

### Example 12.3 (A first example)

Claim:  $P \sim Q$  where

$$\begin{array}{ll} P = a.P_1 + a.P_2 & Q = a.Q_1 \\ P_1 = b.P_2 & Q_1 = b.Q_1 \\ P_2 = b.P_2 & \end{array}$$

Proof:  $\rho = \{(P, Q), (P_1, Q_1), (P_2, Q_1)\}$  is a strong bisimulation

### Example 12.4 (Relating a finite to an infinite-state process)

Claim:  $P_0 \sim Q$  where  $P_i = a.P_{i+1}$  for  $i \in \mathbb{N}$  and  $Q = a.Q$ .

Proof:  $\rho = \{(P_i, Q) \mid i \in \mathbb{N}\}$  is a strong bisimulation.

# Bisimulation

## Examples

### Example 12.3 (A first example)

Claim:  $P \sim Q$  where  $P = a.P_1 + a.P_2$        $Q = a.Q_1$   
 $P_1 = b.P_2$        $Q_1 = b.Q_1$   
 $P_2 = b.P_2$

Proof:  $\rho = \{(P, Q), (P_1, Q_1), (P_2, Q_1)\}$  is a strong bisimulation

### Example 12.4 (Relating a finite to an infinite-state process)

Claim:  $P_0 \sim Q$  where  $P_i = a.P_{i+1}$  for  $i \in \mathbb{N}$  and  $Q = a.Q$ .

Proof:  $\rho = \{(P_i, Q) \mid i \in \mathbb{N}\}$  is a strong bisimulation.

### Example 12.5 (Counterexample; cf. Example 11.10)

Show on board that  $CTM \not\sim CTM'$  where

$$CTM = \text{coin.}(\overline{\text{coffee.}CTM} + \overline{\text{tea.}CTM})$$

$$CTM' = \text{coin.}\overline{\text{coffee.}CTM'} + \text{coin.}\overline{\text{tea.}CTM'}$$

## Properties of Strong Bisimilarity

### Lemma 12.6 (Properties of $\sim$ )

1.  $\sim$  is an *equivalence relation* (i.e., reflexive, symmetric, and transitive)

## Properties of Strong Bisimilarity

### Lemma 12.6 (Properties of $\sim$ )

1.  $\sim$  is an *equivalence relation* (i.e., reflexive, symmetric, and transitive)
2.  $\sim$  is the *coarsest* strong bisimulation



# Bisimulation

---

## Properties of Strong Bisimilarity

### Lemma 12.6 (Properties of $\sim$ )

1.  $\sim$  is an *equivalence relation* (i.e., reflexive, symmetric, and transitive)
2.  $\sim$  is the *coarsest* strong bisimulation

Proof.

on the board



# Bisimulation and Trace Equivalence

---

## Outline of Lecture 12

Recap: Trace Equivalence

Bisimulation

Bisimulation and Trace Equivalence

Bisimulation is a Congruence

# Bisimulation and Trace Equivalence

---

## Bisimulation on Paths

### Lemma 12.7 (Bisimulation on paths)

Whenever we have:

$$P_0 \xrightarrow{\alpha_1} P_1 \xrightarrow{\alpha_2} P_2 \xrightarrow{\alpha_3} P_3 \xrightarrow{\alpha_4} P_4 \dots\dots\dots$$

$\rho$

$Q_0$

# Bisimulation and Trace Equivalence

## Bisimulation on Paths

### Lemma 12.7 (Bisimulation on paths)

Whenever we have:

$$P_0 \xrightarrow{\alpha_1} P_1 \xrightarrow{\alpha_2} P_2 \xrightarrow{\alpha_3} P_3 \xrightarrow{\alpha_4} P_4 \dots\dots\dots$$

$\rho$

$Q_0$

this can be completed to

$$P_0 \xrightarrow{\alpha_1} P_1 \xrightarrow{\alpha_2} P_2 \xrightarrow{\alpha_3} P_3 \xrightarrow{\alpha_4} P_4 \dots\dots\dots$$

$\rho$

$\rho$

$\rho$

$\rho$

$\rho$

$$Q_0 \xrightarrow{\alpha_1} Q_1 \xrightarrow{\alpha_2} Q_2 \xrightarrow{\alpha_3} Q_3 \xrightarrow{\alpha_4} Q_4 \dots\dots\dots$$

# Bisimulation and Trace Equivalence

## Bisimulation on Paths

### Lemma 12.7 (Bisimulation on paths)

Whenever we have:

$$P_0 \xrightarrow{\alpha_1} P_1 \xrightarrow{\alpha_2} P_2 \xrightarrow{\alpha_3} P_3 \xrightarrow{\alpha_4} P_4 \dots\dots\dots$$

$\rho$

$Q_0$

this can be completed to

$$P_0 \xrightarrow{\alpha_1} P_1 \xrightarrow{\alpha_2} P_2 \xrightarrow{\alpha_3} P_3 \xrightarrow{\alpha_4} P_4 \dots\dots\dots$$

$\rho$

$\rho$

$\rho$

$\rho$

$\rho$

$$Q_0 \xrightarrow{\alpha_1} Q_1 \xrightarrow{\alpha_2} Q_2 \xrightarrow{\alpha_3} Q_3 \xrightarrow{\alpha_4} Q_4 \dots\dots\dots$$

Proof.

by induction on the length of the path □

# Bisimulation and Trace Equivalence

---

## Strong Bisimulation vs. Trace Equivalence

### Theorem 12.8

*$P \sim Q$  implies that  $P$  and  $Q$  are trace equivalent. The reverse does generally not hold.*

# Bisimulation and Trace Equivalence

---

## Strong Bisimulation vs. Trace Equivalence

### Theorem 12.8

*$P \sim Q$  implies that  $P$  and  $Q$  are trace equivalent. The reverse does generally not hold.*

### Proof.

The implication from left to right follows from the previous slide.

# Bisimulation and Trace Equivalence

---

## Strong Bisimulation vs. Trace Equivalence

### Theorem 12.8

*$P \sim Q$  implies that  $P$  and  $Q$  are trace equivalent. The reverse does generally not hold.*

### Proof.

The implication from left to right follows from the previous slide.

Consider the other direction.



# Bisimulation and Trace Equivalence

---

## Strong Bisimulation vs. Trace Equivalence

### Theorem 12.8

$P \sim Q$  implies that  $P$  and  $Q$  are trace equivalent. The reverse does generally not hold.

### Proof.

The implication from left to right follows from the previous slide.

Consider the other direction.

Take  $P = a.P_1$  with  $P_1 = b.nil + c.nil$  and  $Q = a.b.nil + a.c.nil$ .

# Bisimulation and Trace Equivalence

---

## Strong Bisimulation vs. Trace Equivalence

### Theorem 12.8

$P \sim Q$  implies that  $P$  and  $Q$  are trace equivalent. The reverse does generally not hold.

### Proof.

The implication from left to right follows from the previous slide.

Consider the other direction.

Take  $P = a.P_1$  with  $P_1 = b.nil + c.nil$  and  $Q = a.b.nil + a.c.nil$ .

Then:  $Tr(P) = \{\epsilon, a, ab, ac\} = Tr(Q)$ .

# Bisimulation and Trace Equivalence

---

## Strong Bisimulation vs. Trace Equivalence

### Theorem 12.8

$P \sim Q$  implies that  $P$  and  $Q$  are trace equivalent. The reverse does generally not hold.

### Proof.

The implication from left to right follows from the previous slide.

Consider the other direction.

Take  $P = a.P_1$  with  $P_1 = b.nil + c.nil$  and  $Q = a.b.nil + a.c.nil$ .

Then:  $Tr(P) = \{\epsilon, a, ab, ac\} = Tr(Q)$ .

Thus,  $P$  and  $Q$  are trace equivalent.

# Bisimulation and Trace Equivalence

---

## Strong Bisimulation vs. Trace Equivalence

### Theorem 12.8

$P \sim Q$  implies that  $P$  and  $Q$  are trace equivalent. The reverse does generally not hold.

### Proof.

The implication from left to right follows from the previous slide.

Consider the other direction.

Take  $P = a.P_1$  with  $P_1 = b.nil + c.nil$  and  $Q = a.b.nil + a.c.nil$ .

Then:  $Tr(P) = \{\epsilon, a, ab, ac\} = Tr(Q)$ .

Thus,  $P$  and  $Q$  are trace equivalent.

But:  $P \not\sim Q$ , as there is no state in the LTS of  $Q$  that is bisimilar to  $P_1$  (cf. Example 12.5).

# Bisimulation and Trace Equivalence

---

## Strong Bisimulation vs. Trace Equivalence

### Theorem 12.8

$P \sim Q$  implies that  $P$  and  $Q$  are trace equivalent. The reverse does generally not hold.

### Proof.

The implication from left to right follows from the previous slide.

Consider the other direction.

Take  $P = a.P_1$  with  $P_1 = b.nil + c.nil$  and  $Q = a.b.nil + a.c.nil$ .

Then:  $Tr(P) = \{\epsilon, a, ab, ac\} = Tr(Q)$ .

Thus,  $P$  and  $Q$  are trace equivalent.

But:  $P \not\sim Q$ , as there is no state in the LTS of  $Q$  that is bisimilar to  $P_1$  (cf. Example 12.5).

Why? No state in  $Q$  can perform both  $b$  and  $c$ . □

# Bisimulation and Trace Equivalence

---

## Deterministic Transition Systems

### Definition 12.9 (Determinism)

$P \in Prc$  is **deterministic** whenever for every of its states  $s$  it holds:

$$\left( s \xrightarrow{\alpha} t \text{ and } s \xrightarrow{\alpha} u \right) \text{ implies } t = u.$$

# Bisimulation and Trace Equivalence

---

## Deterministic Transition Systems

### Definition 12.9 (Determinism)

$P \in Prc$  is **deterministic** whenever for every of its states  $s$  it holds:

$$\left( s \xrightarrow{\alpha} t \text{ and } s \xrightarrow{\alpha} u \right) \text{ implies } t = u.$$

Theorem 12.10 (Determinism implies coincidence of  $\sim$  and trace equivalence) (Park)

*For deterministic  $P$  and  $Q$ :  $P \sim Q$  iff  $Tr(P) = Tr(Q)$ .*

# Bisimulation and Trace Equivalence

---

## Deterministic Transition Systems

### Definition 12.9 (Determinism)

$P \in Prc$  is **deterministic** whenever for every of its states  $s$  it holds:

$$\left( s \xrightarrow{\alpha} t \text{ and } s \xrightarrow{\alpha} u \right) \text{ implies } t = u.$$

Theorem 12.10 (Determinism implies coincidence of  $\sim$  and trace equivalence) (Park)

*For deterministic  $P$  and  $Q$ :  $P \sim Q$  iff  $Tr(P) = Tr(Q)$ .*

Proof.

Left as an exercise.



# Bisimulation and Trace Equivalence

---

## Deterministic Transition Systems

### Definition 12.9 (Determinism)

$P \in Prc$  is **deterministic** whenever for every of its states  $s$  it holds:

$$\left( s \xrightarrow{\alpha} t \text{ and } s \xrightarrow{\alpha} u \right) \text{ implies } t = u.$$

Theorem 12.10 (Determinism implies coincidence of  $\sim$  and trace equivalence) (Park)

*For deterministic  $P$  and  $Q$ :  $P \sim Q$  iff  $Tr(P) = Tr(Q)$ .*

Proof.

Left as an exercise. In fact, for deterministic processes, trace equivalence, complete trace, failure trace, ready trace, and bisimulation equivalence all coincide.  $\square$

# Bisimulation is a Congruence

---

## Outline of Lecture 12

Recap: Trace Equivalence

Bisimulation

Bisimulation and Trace Equivalence

**Bisimulation is a Congruence**

# Bisimulation is a Congruence

## Congruence

Theorem 12.11 (CCS congruence property of  $\sim$ )

*Strong bisimilarity  $\sim$  is a CCS congruence, that is, whenever  $P, Q \in \text{Prc}$  such that  $P \sim Q$ ,*

$$\begin{array}{ll} \alpha.P \sim \alpha.Q & \text{for every action } \alpha \\ P + R \sim Q + R & \text{for every process } R \\ P \parallel R \sim Q \parallel R & \text{for every process } R \\ P \setminus L \sim Q \setminus L & \text{for every set } L \subseteq A \\ P[f] \sim Q[f] & \text{for every relabelling } f : A \rightarrow A \end{array}$$

# Bisimulation is a Congruence

## Congruence

Theorem 12.11 (CCS congruence property of  $\sim$ )

*Strong bisimilarity  $\sim$  is a CCS congruence, that is, whenever  $P, Q \in \text{Prc}$  such that  $P \sim Q$ ,*

$$\begin{array}{ll} \alpha.P \sim \alpha.Q & \text{for every action } \alpha \\ P + R \sim Q + R & \text{for every process } R \\ P \parallel R \sim Q \parallel R & \text{for every process } R \\ P \setminus L \sim Q \setminus L & \text{for every set } L \subseteq A \\ P[f] \sim Q[f] & \text{for every relabelling } f : A \rightarrow A \end{array}$$

## Proof.

- for  $\parallel$ : on the board
- for other CCS operators: left as an exercise

