

# Concurrency Theory

## Strong bisimulation

Joost-Pieter Katoen and Thomas Noll

Lehrstuhl für Informatik 2  
Software Modeling and Verification Group

<http://moves.rwth-aachen.de/teaching/ws-1516/ct>

January 11, 2016

# Overview

- 1 Introduction
- 2 Strong bisimilarity as a game
- 3 Simulation equivalence
- 4 Bisimulation as a fixed point
- 5 Summary

# Overview

- 1 Introduction
- 2 Strong bisimilarity as a game
- 3 Simulation equivalence
- 4 Bisimulation as a fixed point
- 5 Summary

# Summary so far

1. Strong bisimulation is based on mutual mimicking each other

# Summary so far

1. Strong bisimulation is based on mutual mimicking each other
2. Strong bisimilarity  $\sim$ :

# Summary so far

1. Strong bisimulation is based on mutual mimicking each other
2. Strong bisimilarity  $\sim$ :
  - 2.1 is the largest strong bisimulation
  - 2.2 is an equivalence
  - 2.3 is a congruence (for CCS)
  - 2.4 is strictly finer than trace equivalence
  - 2.5 is deadlock sensitive

# Summary so far

1. Strong bisimulation is based on mutual mimicking each other
2. Strong bisimilarity  $\sim$ :
  - 2.1 is the largest strong bisimulation
  - 2.2 is an equivalence
  - 2.3 is a congruence (for CCS)
  - 2.4 is strictly finer than trace equivalence
  - 2.5 is deadlock sensitive

## Aims of this lecture

1. Using games to show non-bisimilarity of two processes
2. Strong simulation: one-way bisimulation
3. Using fixed points to compute  $\sim$

# Strong bisimulation

## Strong bisimulation

[Park, 1981, Milner, 1989]

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

# Strong bisimulation

## Strong bisimulation

[Park, 1981, Milner, 1989]

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

1. if  $P \xrightarrow{\alpha} P'$  then there exists  $Q' \in Prc$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$
2. if  $Q \xrightarrow{\alpha} Q'$  then there exists  $P' \in Prc$  s.t.  $P \xrightarrow{\alpha} P'$  and  $(P', Q') \in \mathcal{R}$ .

# Strong bisimulation

## Strong bisimulation

[Park, 1981, Milner, 1989]

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

1. if  $P \xrightarrow{\alpha} P'$  then there exists  $Q' \in Prc$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$
2. if  $Q \xrightarrow{\alpha} Q'$  then there exists  $P' \in Prc$  s.t.  $P \xrightarrow{\alpha} P'$  and  $(P', Q') \in \mathcal{R}$ .

## Strong bisimilarity

The processes  $P$  and  $Q$  are **strongly bisimilar**, denoted  $P \sim Q$ , iff there is a strong bisimulation  $\mathcal{R}$  with  $(P, Q) \in \mathcal{R}$ .

# Strong bisimulation

## Strong bisimulation

[Park, 1981, Milner, 1989]

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

1. if  $P \xrightarrow{\alpha} P'$  then there exists  $Q' \in \text{Prc}$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$
2. if  $Q \xrightarrow{\alpha} Q'$  then there exists  $P' \in \text{Prc}$  s.t.  $P \xrightarrow{\alpha} P'$  and  $(P', Q') \in \mathcal{R}$ .

## Strong bisimilarity

The processes  $P$  and  $Q$  are **strongly bisimilar**, denoted  $P \sim Q$ , iff there is a strong bisimulation  $\mathcal{R}$  with  $(P, Q) \in \mathcal{R}$ . Thus,

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

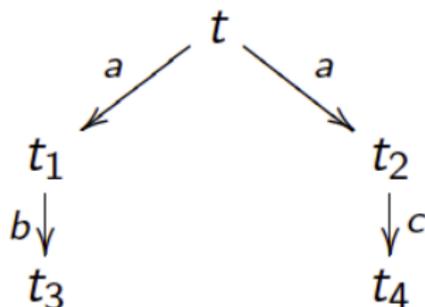
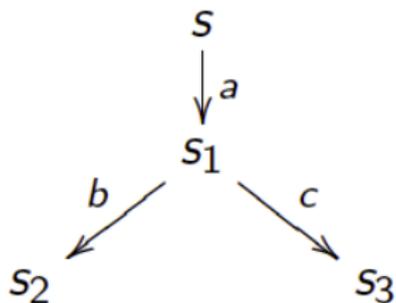
Relation  $\sim$  is called **strong bisimilarity**.

# Overview

- 1 Introduction
- 2 Strong bisimilarity as a game**
- 3 Simulation equivalence
- 4 Bisimulation as a fixed point
- 5 Summary

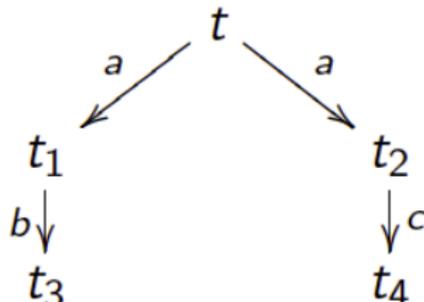
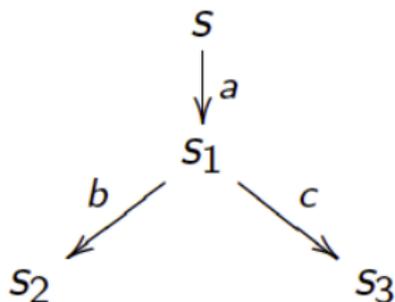
# How to show non-bisimilarity?

# How to show non-bisimilarity?



To prove that  $s \not\sim t$

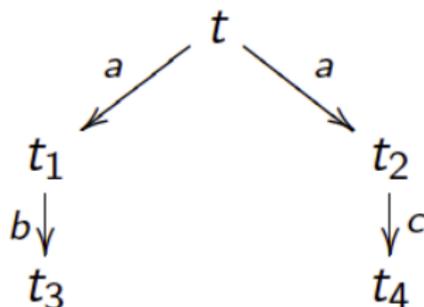
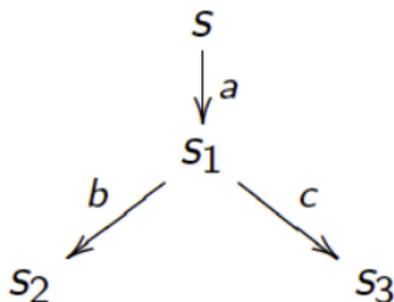
# How to show non-bisimilarity?



To prove that  $s \not\sim t$

- ▶ Enumerate **all binary relations** and show that none of them containing  $(s, t)$  is a strong bisimulation.

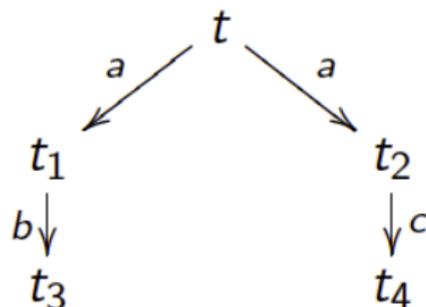
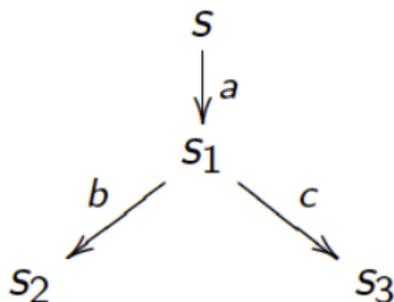
# How to show non-bisimilarity?



To prove that  $s \not\sim t$

- ▶ Enumerate **all binary relations** and show that none of them containing  $(s, t)$  is a strong bisimulation. This is expensive, as there are  $2^{k^2}$  binary relations on  $Prc$  with  $|Prc| = k$ .

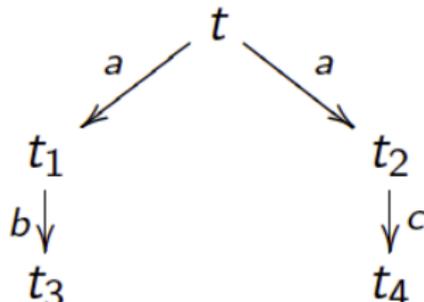
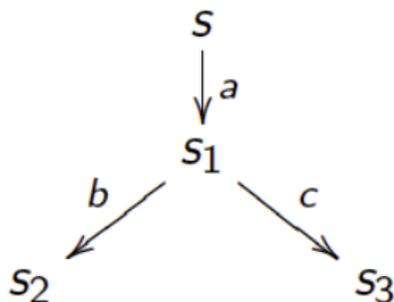
# How to show non-bisimilarity?



To prove that  $s \not\sim t$

- ▶ Enumerate **all binary relations** and show that none of them containing  $(s, t)$  is a strong bisimulation. This is expensive, as there are  $2^{k^2}$  binary relations on  $Prc$  with  $|Prc| = k$ .
- ▶ Make certain **observations** which will enable to disqualify many bisimulation candidates in one step.

# How to show non-bisimilarity?



## To prove that $s \not\sim t$

- ▶ Enumerate **all binary relations** and show that none of them containing  $(s, t)$  is a strong bisimulation. This is expensive, as there are  $2^{k^2}$  binary relations on  $Prc$  with  $|Prc| = k$ .
- ▶ Make certain **observations** which will enable to disqualify many bisimulation candidates in one step.
- ▶ Use **game characterization** of strong bisimilarity.

# Strong bisimulation game

Let  $(Prc, Act, \rightarrow)$  be an LTS and  $s, t \in Prc$ . Aim: does  $s \sim t$ ?

## Strong bisimulation game

Let  $(Prc, Act, \rightarrow)$  be an LTS and  $s, t \in Prc$ . Aim: does  $s \sim t$ ?

We define a game with two players: an “attacker” and “defender”.

## Strong bisimulation game

Let  $(Prc, Act, \rightarrow)$  be an LTS and  $s, t \in Prc$ . Aim: does  $s \sim t$ ?

We define a game with two players: an “attacker” and “defender”.

- ▶ The game is played in **rounds** and configurations of the game are pairs of states from  $Prc \times Prc$ .

# Strong bisimulation game

Let  $(Prc, Act, \rightarrow)$  be an LTS and  $s, t \in Prc$ . Aim: does  $s \sim t$ ?

We define a game with two players: an “attacker” and “defender”.

- ▶ The game is played in **rounds** and configurations of the game are pairs of states from  $Prc \times Prc$ .
- ▶ In each round exactly one configuration is called **current**.

# Strong bisimulation game

Let  $(Prc, Act, \rightarrow)$  be an LTS and  $s, t \in Prc$ . Aim: does  $s \sim t$ ?

We define a game with two players: an “attacker” and “defender”.

- ▶ The game is played in **rounds** and configurations of the game are pairs of states from  $Prc \times Prc$ .
- ▶ In each round exactly one configuration is called **current**.
- ▶ Initially, the configuration  $(s, t)$  is the current one.

# Strong bisimulation game

Let  $(Prc, Act, \rightarrow)$  be an LTS and  $s, t \in Prc$ . Aim: does  $s \sim t$ ?

We define a game with two players: an “attacker” and “defender”.

- ▶ The game is played in **rounds** and configurations of the game are pairs of states from  $Prc \times Prc$ .
- ▶ In each round exactly one configuration is called **current**.
- ▶ Initially, the configuration  $(s, t)$  is the current one.

## Intuition

The defender wants to show that  $s \sim t$  while the attacker aims to show the opposite.

# Rules of the bisimulation game

# Rules of the bisimulation game

## Game rules

In each round the current configuration  $(s, t)$  is changed as follows:

1. the attacker chooses one of the processes in the current configuration, say  $t$ , and makes an  $\xrightarrow{\alpha}$ -move for some  $\alpha \in Act$  to  $t'$ , say, and

# Rules of the bisimulation game

## Game rules

In each round the current configuration  $(s, t)$  is changed as follows:

1. the attacker chooses one of the processes in the current configuration, say  $t$ , and makes an  $\xrightarrow{\alpha}$ -move for some  $\alpha \in Act$  to  $t'$ , say, and
2. the defender must respond by making an  $\xrightarrow{\alpha}$ -move in the other process  $s$  of the current configuration under the same action  $\alpha$ , yielding  $s \xrightarrow{\alpha} s'$ .

# Rules of the bisimulation game

## Game rules

In each round the current configuration  $(s, t)$  is changed as follows:

1. the attacker chooses one of the processes in the current configuration, say  $t$ , and makes an  $\xrightarrow{\alpha}$ -move for some  $\alpha \in Act$  to  $t'$ , say, and
2. the defender must respond by making an  $\xrightarrow{\alpha}$ -move in the other process  $s$  of the current configuration under the same action  $\alpha$ , yielding  $s \xrightarrow{\alpha} s'$ .

The new pair of processes  $(s', t')$  becomes the current configuration.

# Rules of the bisimulation game

## Game rules

In each round the current configuration  $(s, t)$  is changed as follows:

1. the attacker chooses one of the processes in the current configuration, say  $t$ , and makes an  $\xrightarrow{\alpha}$ -move for some  $\alpha \in Act$  to  $t'$ , say, and
2. the defender must respond by making an  $\xrightarrow{\alpha}$ -move in the other process  $s$  of the current configuration under the same action  $\alpha$ , yielding  $s \xrightarrow{\alpha} s'$ .

The new pair of processes  $(s', t')$  becomes the current configuration. The game continues with another round.

# Rules of the bisimulation game

## Game rules

In each round the current configuration  $(s, t)$  is changed as follows:

1. the attacker chooses one of the processes in the current configuration, say  $t$ , and makes an  $\xrightarrow{\alpha}$ -move for some  $\alpha \in Act$  to  $t'$ , say, and
2. the defender must respond by making an  $\xrightarrow{\alpha}$ -move in the other process  $s$  of the current configuration under the same action  $\alpha$ , yielding  $s \xrightarrow{\alpha} s'$ .

The new pair of processes  $(s', t')$  becomes the current configuration. The game continues with another round.

## Game results

1. If one player cannot move, the other player wins.

# Rules of the bisimulation game

## Game rules

In each round the current configuration  $(s, t)$  is changed as follows:

1. the attacker chooses one of the processes in the current configuration, say  $t$ , and makes an  $\xrightarrow{\alpha}$ -move for some  $\alpha \in Act$  to  $t'$ , say, and
2. the defender must respond by making an  $\xrightarrow{\alpha}$ -move in the other process  $s$  of the current configuration under the same action  $\alpha$ , yielding  $s \xrightarrow{\alpha} s'$ .

The new pair of processes  $(s', t')$  becomes the current configuration. The game continues with another round.

## Game results

1. If one player cannot move, the other player wins.
2. If the game can be played *ad infinitum*, the defender wins.

# Game characterization of bisimulation

---

<sup>1</sup>In the next lectures, we will present yet another method to check this.

# Game characterization of bisimulation

## Theorem

[Stirling, 1995]. [Thomas, 1993]

1.  $s \sim t$  iff the defender has a **universal** winning strategy from configuration  $(s, t)$ .
2.  $s \not\sim t$  iff the attacker has a **universal** winning strategy from configuration  $(s, t)$ .

(By means of a universal winning strategy, a player can always win, regardless of how the other player selects her moves.)

---

<sup>1</sup>In the next lectures, we will present yet another method to check this.

# Game characterization of bisimulation

## Theorem

[Stirling, 1995], [Thomas, 1993]

1.  $s \sim t$  iff the defender has a **universal** winning strategy from configuration  $(s, t)$ .
2.  $s \not\sim t$  iff the attacker has a **universal** winning strategy from configuration  $(s, t)$ .

(By means of a universal winning strategy, a player can always win, regardless of how the other player selects her moves.)

## Proof.

Left as an exercise. □

---

<sup>1</sup>In the next lectures, we will present yet another method to check this.

# Game characterization of bisimulation

## Theorem

[Stirling, 1995], [Thomas, 1993]

1.  $s \sim t$  iff the defender has a **universal** winning strategy from configuration  $(s, t)$ .
2.  $s \not\sim t$  iff the attacker has a **universal** winning strategy from configuration  $(s, t)$ .

(By means of a universal winning strategy, a player can always win, regardless of how the other player selects her moves.)

## Proof.

Left as an exercise. □

A bisimulation game can be used to prove bisimilarity as well as non-bisimilarity.<sup>1</sup> It often provides elegant arguments for  $s \not\sim t$ .

<sup>1</sup>In the next lectures, we will present yet another method to check this.

# Example

## A first example

Use the game characterization to show  $P \sim Q$  where:

$$P = a.P_1 + a.P_2$$

$$P_1 = b.P_2$$

$$P_2 = b.P_2$$

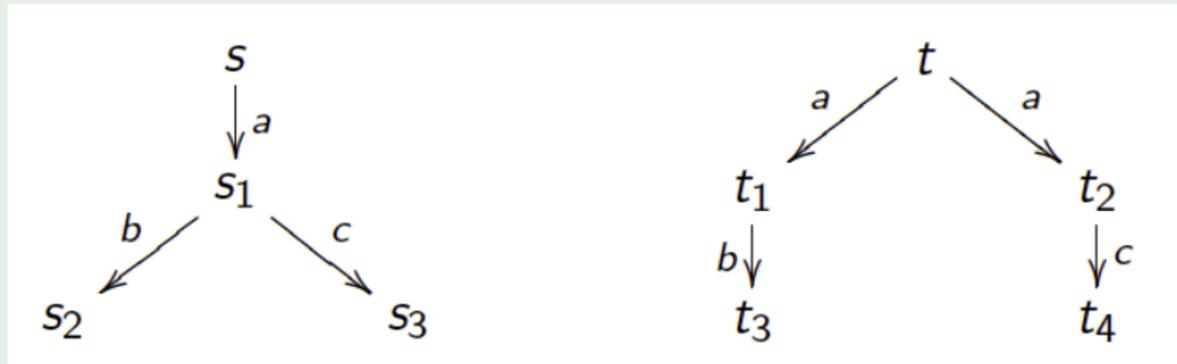
$$Q = a.Q_1$$

$$Q_1 = b.Q_1.$$

# Example

## Another example

Use the game characterization to show that  $s \not\sim t$  where:



# Overview

- 1 Introduction
- 2 Strong bisimilarity as a game
- 3 Simulation equivalence**
- 4 Bisimulation as a fixed point
- 5 Summary

# Strong simulation

**Observation:** sometimes, the concept of strong bisimulation is **too strong** (example: extending a system by new features).

# Strong simulation

**Observation:** sometimes, the concept of strong bisimulation is **too strong** (example: extending a system by new features).

## Strong simulation

Relation  $\mathcal{R} \subseteq Prc \times Prc$  is a **strong simulation** if, whenever  $(P, Q) \in \mathcal{R}$  and  $P \xrightarrow{\alpha} P'$ , there exists  $Q' \in Prc$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$ .

# Strong simulation

**Observation:** sometimes, the concept of strong bisimulation is **too strong** (example: extending a system by new features).

## Strong simulation

Relation  $\mathcal{R} \subseteq Prc \times Prc$  is a **strong simulation** if, whenever  $(P, Q) \in \mathcal{R}$  and  $P \xrightarrow{\alpha} P'$ , there exists  $Q' \in Prc$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$ .

$Q$  **strongly simulates**  $P$ , denoted  $P \sqsubseteq Q$ , if there exists a strong simulation  $\mathcal{R}$  such that  $(P, Q) \in \mathcal{R}$ . Relation  $\sqsubseteq$  is called **strong similarity**.

# Strong simulation

**Observation:** sometimes, the concept of strong bisimulation is **too strong** (example: extending a system by new features).

## Strong simulation

Relation  $\mathcal{R} \subseteq Prc \times Prc$  is a **strong simulation** if, whenever  $(P, Q) \in \mathcal{R}$  and  $P \xrightarrow{\alpha} P'$ , there exists  $Q' \in Prc$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$ .

$Q$  **strongly simulates**  $P$ , denoted  $P \sqsubseteq Q$ , if there exists a strong simulation  $\mathcal{R}$  such that  $(P, Q) \in \mathcal{R}$ . Relation  $\sqsubseteq$  is called **strong similarity**.

Thus: if  $Q$  strongly simulates  $P$ , then whatever transition  $P$  takes,  $Q$  can match it which retains all of  $P$ 's options.

# Strong simulation

**Observation:** sometimes, the concept of strong bisimulation is **too strong** (example: extending a system by new features).

## Strong simulation

Relation  $\mathcal{R} \subseteq Prc \times Prc$  is a **strong simulation** if, whenever  $(P, Q) \in \mathcal{R}$  and  $P \xrightarrow{\alpha} P'$ , there exists  $Q' \in Prc$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$ .

$Q$  **strongly simulates**  $P$ , denoted  $P \sqsubseteq Q$ , if there exists a strong simulation  $\mathcal{R}$  such that  $(P, Q) \in \mathcal{R}$ . Relation  $\sqsubseteq$  is called **strong similarity**.

Thus: if  $Q$  strongly simulates  $P$ , then whatever transition  $P$  takes,  $Q$  can match it which retains all of  $P$ 's options. But:  $P$  does not need to be able to match each transition of  $Q$ !

# Simulation: example

## Strong simulation

Relation  $\mathcal{R} \subseteq Proc \times Proc$  is a **strong simulation** if, whenever  $(P, Q) \in \mathcal{R}$  and  $P \xrightarrow{\alpha} P'$ , there exists  $Q' \in Proc$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$ .

$Q$  **strongly simulates**  $P$ , denoted  $P \sqsubseteq Q$ , if there exists a strong simulation  $\mathcal{R}$  such that  $(P, Q) \in \mathcal{R}$ .

# Simulation: example

## Strong simulation

Relation  $\mathcal{R} \subseteq Proc \times Proc$  is a **strong simulation** if, whenever  $(P, Q) \in \mathcal{R}$  and  $P \xrightarrow{\alpha} P'$ , there exists  $Q' \in Proc$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$ .

$Q$  **strongly simulates**  $P$ , denoted  $P \sqsubseteq Q$ , if there exists a strong simulation  $\mathcal{R}$  such that  $(P, Q) \in \mathcal{R}$ .

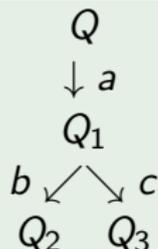
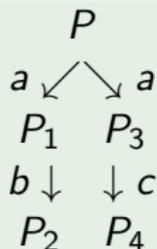
# Simulation: example

## Strong simulation

Relation  $\mathcal{R} \subseteq \text{Prc} \times \text{Prc}$  is a **strong simulation** if, whenever  $(P, Q) \in \mathcal{R}$  and  $P \xrightarrow{\alpha} P'$ , there exists  $Q' \in \text{Prc}$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$ .

$Q$  **strongly simulates**  $P$ , denoted  $P \sqsubseteq Q$ , if there exists a strong simulation  $\mathcal{R}$  such that  $(P, Q) \in \mathcal{R}$ .

## Example



$Q$  strongly simulates  $P$ ,

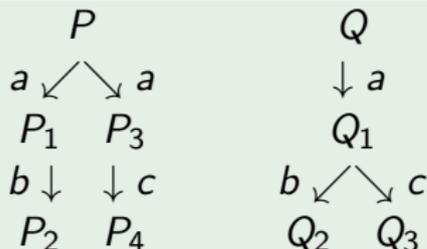
# Simulation: example

## Strong simulation

Relation  $\mathcal{R} \subseteq \text{Prc} \times \text{Prc}$  is a **strong simulation** if, whenever  $(P, Q) \in \mathcal{R}$  and  $P \xrightarrow{\alpha} P'$ , there exists  $Q' \in \text{Prc}$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$ .

$Q$  **strongly simulates**  $P$ , denoted  $P \sqsubseteq Q$ , if there exists a strong simulation  $\mathcal{R}$  such that  $(P, Q) \in \mathcal{R}$ .

## Example



$Q$  strongly simulates  $P$ ,  
but not vice versa

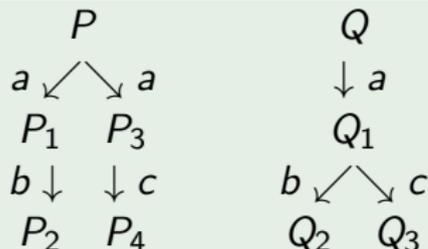
# Simulation: example

## Strong simulation

Relation  $\mathcal{R} \subseteq \text{Prc} \times \text{Prc}$  is a **strong simulation** if, whenever  $(P, Q) \in \mathcal{R}$  and  $P \xrightarrow{\alpha} P'$ , there exists  $Q' \in \text{Prc}$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$ .

$Q$  **strongly simulates**  $P$ , denoted  $P \sqsubseteq Q$ , if there exists a strong simulation  $\mathcal{R}$  such that  $(P, Q) \in \mathcal{R}$ .

## Example



$Q$  strongly simulates  $P$ ,  
but not vice versa

This yields that:

$$a.b.\text{nil} + a.c.\text{nil} \sqsubseteq a.(b.\text{nil} + c.\text{nil})$$

$$a.(b.\text{nil} + c.\text{nil}) \not\sqsubseteq a.b.\text{nil} + a.c.\text{nil}.$$

# Strong simulation and bisimilarity

## Proposition

If  $P \sim Q$ , then  $Q \sqsubseteq P$  and  $P \sqsubseteq Q$ .

# Strong simulation and bisimilarity

## Proposition

If  $P \sim Q$ , then  $Q \sqsubseteq P$  and  $P \sqsubseteq Q$ .

## Proof.

A strong bisimulation  $\mathcal{R} \subseteq Prc \times Prc$  for  $P \sim Q$  is a strong simulation for both directions. □

# Strong simulation and bisimilarity

## Proposition

If  $P \sim Q$ , then  $Q \sqsubseteq P$  and  $P \sqsubseteq Q$ .

## Proof.

A strong bisimulation  $\mathcal{R} \subseteq Prc \times Prc$  for  $P \sim Q$  is a strong simulation for both directions. □

**Caveat:** the converse does not generally hold!

# Strong simulation and bisimilarity

## Proposition

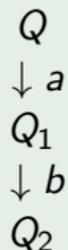
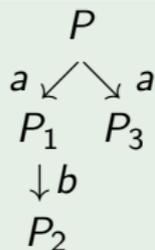
If  $P \sim Q$ , then  $Q \sqsubseteq P$  and  $P \sqsubseteq Q$ .

## Proof.

A strong bisimulation  $\mathcal{R} \subseteq \text{Prc} \times \text{Prc}$  for  $P \sim Q$  is a strong simulation for both directions. □

**Caveat:** the converse does not generally hold!

## Example



$P \sqsubseteq Q$  and  $Q \sqsubseteq P$ ,

# Strong simulation and bisimilarity

## Proposition

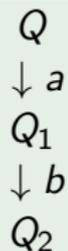
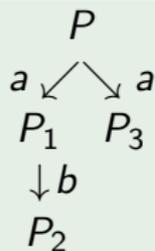
If  $P \sim Q$ , then  $Q \sqsubseteq P$  and  $P \sqsubseteq Q$ .

## Proof.

A strong bisimulation  $\mathcal{R} \subseteq Prc \times Prc$  for  $P \sim Q$  is a strong simulation for both directions. □

**Caveat:** the converse does not generally hold!

## Example



$P \sqsubseteq Q$  and  $Q \sqsubseteq P$ ,  
but  $P \not\sim Q$

## Ready simulation

If  $P \sqsubseteq Q$  and  $P$  has a deadlock,  $Q$  does not necessarily have a deadlock.  
This anomaly of  $\sqsubseteq$  can be remedied as follows

## Ready simulation

If  $P \sqsubseteq Q$  and  $P$  has a deadlock,  $Q$  does not necessarily have a deadlock. This anomaly of  $\sqsubseteq$  can be remedied as follows

### Ready simulation

Relation  $\mathcal{R} \subseteq Prc \times Prc$  is a **ready simulation** if, whenever  $(P, Q) \in \mathcal{R}$  and  $\alpha \in Act$ :

1. if  $P \xrightarrow{\alpha} P'$ , then there exists  $Q' \in Prc$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$ ,

## Ready simulation

If  $P \sqsubseteq Q$  and  $P$  has a deadlock,  $Q$  does not necessarily have a deadlock. This anomaly of  $\sqsubseteq$  can be remedied as follows

### Ready simulation

Relation  $\mathcal{R} \subseteq Prc \times Prc$  is a **ready simulation** if, whenever  $(P, Q) \in \mathcal{R}$  and  $\alpha \in Act$ :

1. if  $P \xrightarrow{\alpha} P'$ , then there exists  $Q' \in Prc$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$ , and
2. if  $Q \xrightarrow{\alpha}$ , then  $P \xrightarrow{\alpha}$ .

## Ready simulation

If  $P \sqsubseteq Q$  and  $P$  has a deadlock,  $Q$  does not necessarily have a deadlock. This anomaly of  $\sqsubseteq$  can be remedied as follows

### Ready simulation

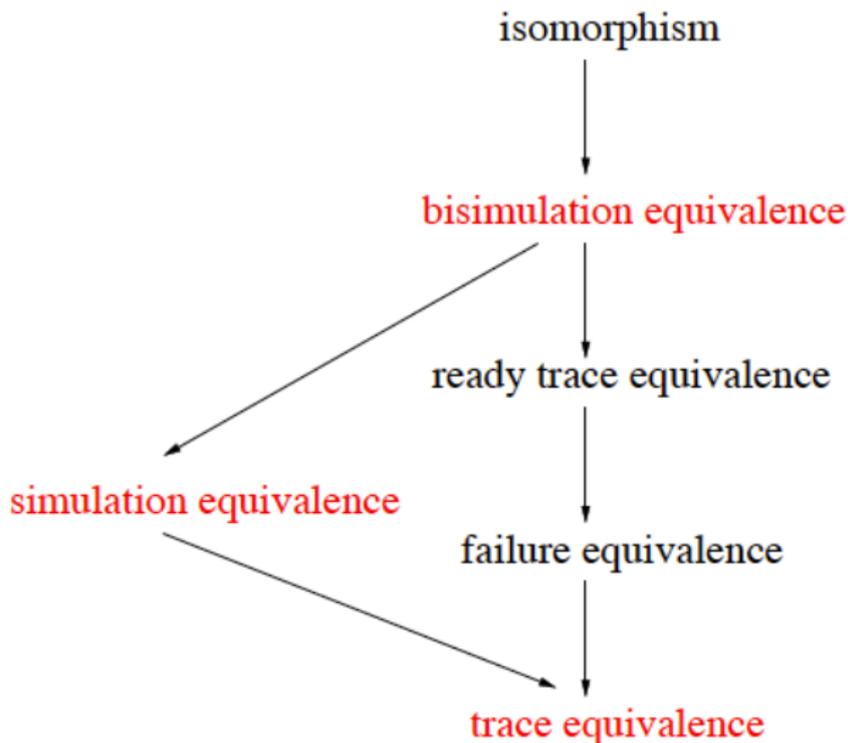
Relation  $\mathcal{R} \subseteq Prc \times Prc$  is a **ready simulation** if, whenever  $(P, Q) \in \mathcal{R}$  and  $\alpha \in Act$ :

1. if  $P \xrightarrow{\alpha} P'$ , then there exists  $Q' \in Prc$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$ , and
2. if  $Q \xrightarrow{\alpha}$ , then  $P \xrightarrow{\alpha}$ .

$Q$  **ready simulates**  $P$ , denoted  $P \sqsubseteq_{rs} Q$ , if there exists a ready simulation  $\mathcal{R}$  such that  $(P, Q) \in \mathcal{R}$ .

In addition to the requirement for  $\sqsubseteq$ , ready simulation requires that if  $Q$  does not deadlock on  $\alpha$ , then  $P$  neither does.

# Overview of some behavioural equivalences



# Overview

- 1 Introduction
- 2 Strong bisimilarity as a game
- 3 Simulation equivalence
- 4 Bisimulation as a fixed point**
- 5 Summary

# Strong bisimilarity

Recall:  $\sim$  implies trace equivalence, and checking trace equivalence is PSPACE-complete.

---

<sup>2</sup>A complete lattice is a partial order s.t. all its sets have a glb and a lub.

# Strong bisimilarity

Recall:  $\sim$  implies trace equivalence, and checking trace equivalence is PSPACE-complete.

What about checking  $\sim$  between two processes?

---

<sup>2</sup>A complete lattice is a partial order s.t. all its sets have a glb and a lub.

# Strong bisimilarity

Recall:  $\sim$  implies trace equivalence, and checking trace equivalence is PSPACE-complete.

What about checking  $\sim$  between two processes?

## Strong bisimilarity

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

---

<sup>2</sup>A complete lattice is a partial order s.t. all its sets have a glb and a lub.

# Strong bisimilarity

Recall:  $\sim$  implies trace equivalence, and checking trace equivalence is PSPACE-complete.

What about checking  $\sim$  between two processes?

## Strong bisimilarity

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

Note that  $(2^{Proc \times Proc}, \subseteq)$  is a complete lattice<sup>2</sup> with  $\cup$  and  $\cap$  as least upper bound and greatest lower bound.

---

<sup>2</sup>A complete lattice is a partial order s.t. all its sets have a glb and a lub.

# Strong bisimilarity

Recall:  $\sim$  implies trace equivalence, and checking trace equivalence is PSPACE-complete.

What about checking  $\sim$  between two processes?

## Strong bisimilarity

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

Note that  $(2^{Proc \times Proc}, \subseteq)$  is a complete lattice<sup>2</sup> with  $\bigcup$  and  $\bigcap$  as least upper bound and greatest lower bound. We will show that  $\sim$  can be characterized as a **fixed point of a monotonic function** on this lattice.

---

<sup>2</sup>A complete lattice is a partial order s.t. all its sets have a glb and a lub.

# Fixed point characterization of $\sim$

# Fixed point characterization of $\sim$

## Function on relations

# Fixed point characterization of $\sim$

## Function on relations

Let  $\mathcal{R} \subseteq Proc \times Proc$ .

# Fixed point characterization of $\sim$

## Function on relations

Let  $\mathcal{R} \subseteq Proc \times Proc$ . Let  $\mathcal{F} : 2^{Proc \times Proc} \rightarrow 2^{Proc \times Proc}$

# Fixed point characterization of $\sim$

## Function on relations

Let  $\mathcal{R} \subseteq Proc \times Proc$ . Let  $\mathcal{F} : 2^{Proc \times Proc} \rightarrow 2^{Proc \times Proc}$  be defined as follows:  
 $(P, Q) \in \mathcal{F}(\mathcal{R})$  for all  $P, Q \in Proc$  iff:

1. if  $P \xrightarrow{\alpha} P'$  then there exists  $Q' \in Proc$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$
2. if  $Q \xrightarrow{\alpha} Q'$  then there exists  $P' \in Proc$  s.t.  $P \xrightarrow{\alpha} P'$  and  $(P', Q') \in \mathcal{R}$ .

# Fixed point characterization of $\sim$

## Function on relations

Let  $\mathcal{R} \subseteq Proc \times Proc$ . Let  $\mathcal{F} : 2^{Proc \times Proc} \rightarrow 2^{Proc \times Proc}$  be defined as follows:  
 $(P, Q) \in \mathcal{F}(\mathcal{R})$  for all  $P, Q \in Proc$  iff:

1. if  $P \xrightarrow{\alpha} P'$  then there exists  $Q' \in Proc$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$
2. if  $Q \xrightarrow{\alpha} Q'$  then there exists  $P' \in Proc$  s.t.  $P \xrightarrow{\alpha} P'$  and  $(P', Q') \in \mathcal{R}$ .

## Intuition

$\mathcal{F}(\mathcal{R})$  contains all pairs of processes from which, in one round of the bisimulation game, the defender can ensure that the players reach a current configuration that is contained in  $\mathcal{R}$ .

# Fixed point characterization of $\sim$

## Function on relations

Let  $\mathcal{R} \subseteq \text{Prc} \times \text{Prc}$ . Let  $\mathcal{F} : 2^{\text{Prc} \times \text{Prc}} \rightarrow 2^{\text{Prc} \times \text{Prc}}$  be defined as follows:  
 $(P, Q) \in \mathcal{F}(\mathcal{R})$  for all  $P, Q \in \text{Prc}$  iff:

1. if  $P \xrightarrow{\alpha} P'$  then there exists  $Q' \in \text{Prc}$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$
2. if  $Q \xrightarrow{\alpha} Q'$  then there exists  $P' \in \text{Prc}$  s.t.  $P \xrightarrow{\alpha} P'$  and  $(P', Q') \in \mathcal{R}$ .

## Intuition

$\mathcal{F}(\mathcal{R})$  contains all pairs of processes from which, in one round of the bisimulation game, the defender can ensure that the players reach a current configuration that is contained in  $\mathcal{R}$ .

## Proposition

$\mathcal{R}$  is a strong bisimulation iff  $\mathcal{R} \subseteq \mathcal{F}(\mathcal{R})$ ,

# Fixed point characterization of $\sim$

## Function on relations

Let  $\mathcal{R} \subseteq \text{Prc} \times \text{Prc}$ . Let  $\mathcal{F} : 2^{\text{Prc} \times \text{Prc}} \rightarrow 2^{\text{Prc} \times \text{Prc}}$  be defined as follows:  
 $(P, Q) \in \mathcal{F}(\mathcal{R})$  for all  $P, Q \in \text{Prc}$  iff:

1. if  $P \xrightarrow{\alpha} P'$  then there exists  $Q' \in \text{Prc}$  s.t.  $Q \xrightarrow{\alpha} Q'$  and  $(P', Q') \in \mathcal{R}$
2. if  $Q \xrightarrow{\alpha} Q'$  then there exists  $P' \in \text{Prc}$  s.t.  $P \xrightarrow{\alpha} P'$  and  $(P', Q') \in \mathcal{R}$ .

## Intuition

$\mathcal{F}(\mathcal{R})$  contains all pairs of processes from which, in one round of the bisimulation game, the defender can ensure that the players reach a current configuration that is contained in  $\mathcal{R}$ .

## Proposition

$\mathcal{R}$  is a strong bisimulation iff  $\mathcal{R} \subseteq \mathcal{F}(\mathcal{R})$ , and thus:

$$\sim = \bigcup \{ \mathcal{R} \in \text{Prc} \times \text{Prc} \mid \mathcal{R} \subseteq \mathcal{F}(\mathcal{R}) \}.$$

# Computation of $\sim$

## Proposition

$\mathcal{R}$  is a bisimulation iff  $\mathcal{R} \subseteq \mathcal{F}(\mathcal{R})$ , and thus:

$$\sim = \bigcup \{ \mathcal{R} \in \text{Prc} \times \text{Prc} \mid \mathcal{R} \subseteq \mathcal{F}(\mathcal{R}) \}.$$

# Computation of $\sim$

## Proposition

$\mathcal{R}$  is a bisimulation iff  $\mathcal{R} \subseteq \mathcal{F}(\mathcal{R})$ , and thus:

$$\sim = \bigcup \{ \mathcal{R} \in \text{Prc} \times \text{Prc} \mid \mathcal{R} \subseteq \mathcal{F}(\mathcal{R}) \}.$$

## Theorem

For **finite-state** process  $P$  with state space  $S$ ,  $\sim$  can be computed by:

$$\begin{aligned} \sim &= \bigcap_{i=0}^{\infty} \sim_i \quad \text{where } \sim_i \text{ is defined by} \\ \sim_0 &= S \times S \\ \sim_{i+1} &= \mathcal{F}(\sim_i). \end{aligned}$$

## Proof.

Using the facts that  $\mathcal{F}$  is monotonic on  $(2^{\text{Prc} \times \text{Prc}}, \subseteq)$  and Tarski's fixed point theorem. □

# Naive algorithm for checking $\sim$

---

<sup>3</sup>Recall that checking trace equivalence is PSPACE-complete.

# Naive algorithm for checking $\sim$

The fixed point characterisation suggests a **polynomial-time** algorithm.

---

<sup>3</sup>Recall that checking trace equivalence is PSPACE-complete.

## Naive algorithm for checking $\sim$

The fixed point characterisation suggests a **polynomial-time** algorithm.

Let  $P, Q$  be finite-state processes. Aim is to check whether  $P \sim Q$ .

---

<sup>3</sup>Recall that checking trace equivalence is PSPACE-complete.

## Naive algorithm for checking $\sim$

The fixed point characterisation suggests a **polynomial-time** algorithm.

Let  $P, Q$  be finite-state processes. Aim is to check whether  $P \sim Q$ .

1. Start with  $\sim = (S_P \cup S_Q) \times (S_P \cup S_Q)$

---

<sup>3</sup>Recall that checking trace equivalence is PSPACE-complete.

# Naive algorithm for checking $\sim$

The fixed point characterisation suggests a **polynomial-time** algorithm.

Let  $P, Q$  be finite-state processes. Aim is to check whether  $P \sim Q$ .

1. Start with  $\sim = (S_P \cup S_Q) \times (S_P \cup S_Q)$
2. As long as there is  $(s, t) \in \sim$  with  $s \xrightarrow{\alpha} s'$  and there is no  $t'$  such that  $t \xrightarrow{\alpha} t'$  and  $(s', t') \in \sim$ , then
$$\sim := \sim \setminus \{(s, t)\}$$

---

<sup>3</sup>Recall that checking trace equivalence is PSPACE-complete.

# Naive algorithm for checking $\sim$

The fixed point characterisation suggests a **polynomial-time** algorithm.

Let  $P, Q$  be finite-state processes. Aim is to check whether  $P \sim Q$ .

1. Start with  $\sim = (S_P \cup S_Q) \times (S_P \cup S_Q)$
2. As long as there is  $(s, t) \in \sim$  with  $s \xrightarrow{\alpha} s'$  and there is no  $t'$  such that  $t \xrightarrow{\alpha} t'$  and  $(s', t') \in \sim$ , then
$$\sim := \sim \setminus \{(s, t)\}$$
3. More efficient schemes do exist, but are not topic of this lecture.

---

<sup>3</sup>Recall that checking trace equivalence is PSPACE-complete.

# Naive algorithm for checking $\sim$

The fixed point characterisation suggests a **polynomial-time** algorithm.

Let  $P, Q$  be finite-state processes. Aim is to check whether  $P \sim Q$ .

1. Start with  $\sim = (S_P \cup S_Q) \times (S_P \cup S_Q)$
2. As long as there is  $(s, t) \in \sim$  with  $s \xrightarrow{\alpha} s'$  and there is no  $t'$  such that  $t \xrightarrow{\alpha} t'$  and  $(s', t') \in \sim$ , then  

$$\sim := \sim \setminus \{(s, t)\}$$
3. More efficient schemes do exist, but are not topic of this lecture.

## Complexity

[Balcázar et al., 1992]

Deciding strong bisimilarity between finite LTSs is P-complete.<sup>3</sup>

<sup>3</sup>Recall that checking trace equivalence is PSPACE-complete.

# Overview

- 1 Introduction
- 2 Strong bisimilarity as a game
- 3 Simulation equivalence
- 4 Bisimulation as a fixed point
- 5 Summary**

# Summary

1. Checking (non-)bisimilarity can be done using a two-player game
2. Strong simulation is a one-way strong bisimulation
3. Strong simulation equivalence is strictly coarser than  $\sim$
4. Ready simulation takes deadlocks into account
5. Strong bisimilarity can be characterised as a fixed point
6. This yields a polynomial-time procedure for determining  $\sim$ .