Concurrency Theory

Winter Semester 2017/18

Lecture 11: Strong Bisimulation

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

http://moves.rwth-aachen.de/teaching/ws-1718/ct/

Software Modeling and Verification Chair

RWTH AACHEN UNIVERSITY

Outline of Lecture 11

Trace Equivalence

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

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.

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:

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

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)

Outline of Lecture 11

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 scheme:

Bisimulation scheme

$P, Q \in Prc$ are equivalent iff, for every action α, every α-successor of P is equivalent to some α-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 scheme:

Bisimulation scheme

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

Three versions will be considered in this course:

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

Strong Bisimulation I

Definition 11.1 (Strong bisimulation) (Park 1981, Milner 1989)

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

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

Strong Bisimulation I

Definition 11.1 (Strong bisimulation) (Park 1981, Milner 1989)

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

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

Definition 11.2 (Strong bisimilarity)

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

Strong Bisimulation I

Definition 11.1 (Strong bisimulation) (Park 1981, Milner 1989)

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

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

Definition 11.2 (Strong bisimilarity)

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

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

Strong Bisimulation I

Definition 11.1 (Strong bisimulation) (Park 1981, Milner 1989)

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

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

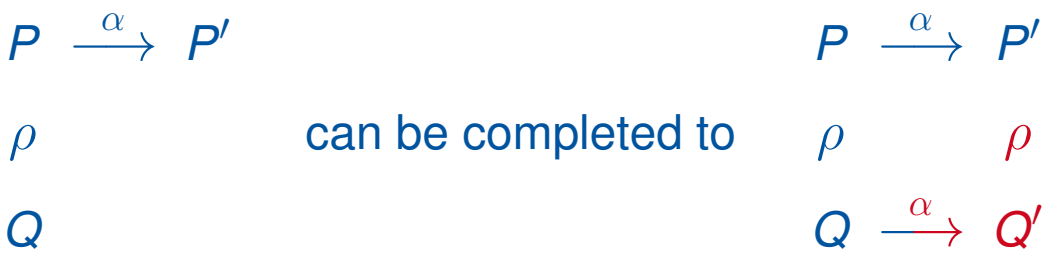
Definition 11.2 (Strong bisimilarity)

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

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

Relation $\sim$ is called strong bisimilarity.

Strong Bisimulation II



Strong Bisimulation II

$$\begin{array}{ccc} P & \xrightarrow{\alpha} & P' \\ \rho & & \\ Q & & \end{array} \quad \text{can be completed to} \quad \begin{array}{ccc} P & \xrightarrow{\alpha} & P' \\ \rho & & \rho \\ Q & \xrightarrow{\alpha} & Q' \end{array}$$

and

$$\begin{array}{ccc} P & & \\ \rho & & \\ Q & \xrightarrow{\alpha} & Q' \end{array} \quad \text{can be completed to} \quad \begin{array}{ccc} P & \xrightarrow{\alpha} & P' \\ \rho & & \rho \\ Q & \xrightarrow{\alpha} & Q' \end{array}$$

Examples

Example 11.3 (A first example)

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

$P_1 = b.P_2$ $\quad Q_1 = b.Q_1$

$P_2 = b.P_2$

Examples

Example 11.3 (A first example)

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

$P_1 = b.P_2$ $\quad Q_1 = b.Q_1$

$P_2 = b.P_2$

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

Examples

Example 11.3 (A first example)

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

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

Example 11.4 (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 $Q = a.Q$.

Examples

Example 11.3 (A first example)

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

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

Example 11.4 (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.

Examples

Example 11.3 (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 11.4 (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 11.5 (Counterexample; cf. Example 10.10)

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

$$CTM = coin.\left(\overline{coffee}.CTM + \overline{tea}.CTM\right)$$
$$CTM' = coin.\overline{coffee}.CTM' + coin.\overline{tea}.CTM'.$$

Properties of Strong Bisimilarity

Lemma 11.6 (Properties of $\sim$)

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

Properties of Strong Bisimilarity

Lemma 11.6 (Properties of $\sim$)

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

Properties of Strong Bisimilarity

Lemma 11.6 (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 □

Outline of Lecture 11

Bisimulation on Paths

Lemma 11.7 (Bisimulation on paths)

Whenever we have:

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

$$\rho$$

$$Q_0$$

Bisimulation on Paths

Lemma 11.7 (Bisimulation on paths)

Whenever we have:

$$\begin{array}{ccccccccccc} P_0 & \xrightarrow{\alpha_1} & P_1 & \xrightarrow{\alpha_2} & P_2 & \xrightarrow{\alpha_3} & P_3 & \xrightarrow{\alpha_4} & P_4 & \ldots\ldots \\ \rho & & & & & & & & & \\ Q_0 & & & & & & & & & \end{array}$$

this can be completed to

$$\begin{array}{ccccccccccc} P_0 & \xrightarrow{\alpha_1} & P_1 & \xrightarrow{\alpha_2} & P_2 & \xrightarrow{\alpha_3} & P_3 & \xrightarrow{\alpha_4} & P_4 & \ldots\ldots \\ \rho & & \rho & & \rho & & \rho & & \rho & \\ Q_0 & \xrightarrow{\alpha_1} & Q_1 & \xrightarrow{\alpha_2} & Q_2 & \xrightarrow{\alpha_3} & Q_3 & \xrightarrow{\alpha_4} & Q_4 & \ldots\ldots \end{array}$$

Bisimulation on Paths

Lemma 11.7 (Bisimulation on paths)

Whenever we have:

$$\begin{array}{ccccccccccc} P_0 & \xrightarrow{\alpha_1} & P_1 & \xrightarrow{\alpha_2} & P_2 & \xrightarrow{\alpha_3} & P_3 & \xrightarrow{\alpha_4} & P_4 & \ldots\ldots \\ \rho & & & & & & & & & \\ Q_0 & & & & & & & & & \end{array}$$

this can be completed to

$$\begin{array}{ccccccccccc} P_0 & \xrightarrow{\alpha_1} & P_1 & \xrightarrow{\alpha_2} & P_2 & \xrightarrow{\alpha_3} & P_3 & \xrightarrow{\alpha_4} & P_4 & \ldots\ldots \\ \rho & & \rho & & \rho & & \rho & & \rho & \\ Q_0 & \xrightarrow{\alpha_1} & Q_1 & \xrightarrow{\alpha_2} & Q_2 & \xrightarrow{\alpha_3} & Q_3 & \xrightarrow{\alpha_4} & Q_4 & \ldots\ldots \end{array}$$

Proof.

by induction on the length of the path □

Strong Bisimulation vs. Trace Equivalence

Theorem 11.8

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

Strong Bisimulation vs. Trace Equivalence

Theorem 11.8

$P \sim Q$ 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.

Strong Bisimulation vs. Trace Equivalence

Theorem 11.8

$P \sim Q$ 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.

Strong Bisimulation vs. Trace Equivalence

Theorem 11.8

$P \sim Q$ 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 $P_1 = b.\text{nil} + c.\text{nil}$ and $Q = a.b.nil + a.c.\text{nil}$.

Strong Bisimulation vs. Trace Equivalence

Theorem 11.8

$P \sim Q$ 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 $P_1 = b.\text{nil} + c.\text{nil}$ and $Q = a.b.nil + a.c.\text{nil}$.
Then: $Tr(P) = \{\epsilon, a, ab, ac\} = Tr(Q)$.

Strong Bisimulation vs. Trace Equivalence

Theorem 11.8

$P \sim Q$ 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 $P_1 = b.\text{nil} + c.\text{nil}$ and $Q = a.b.nil + a.c.\text{nil}$.
Then: $Tr(P) = \{\epsilon, a, ab, ac\} = Tr(Q)$.
Thus, P and Q are trace equivalent.

Strong Bisimulation vs. Trace Equivalence

Theorem 11.8

$P \sim Q$ 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 $P_1 = b.\text{nil} + c.\text{nil}$ and $Q = a.b.nil + a.c.\text{nil}$.
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.

Strong Bisimulation vs. Trace Equivalence

Theorem 11.8

$P \sim Q$ 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 $P_1 = b.\text{nil} + c.\text{nil}$ and $Q = a.b.nil + a.c.\text{nil}$.
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.
Why? No state in Q can perform both b and c. □

Deterministic Transition Systems

Definition 11.9 (Determinism)

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

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

Deterministic Transition Systems

Definition 11.9 (Determinism)

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

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

Theorem 11.10 (Determinism implies coincidence of $\sim$ and trace equivalence) (Park)

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

Deterministic Transition Systems

Definition 11.9 (Determinism)

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

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

Theorem 11.10 (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.

Deterministic Transition Systems

Definition 11.9 (Determinism)

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

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

Theorem 11.10 (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, and ready trace equivalence all coincide. □