Concurrency Theory

Winter Semester 2017/18

Lecture 12: Properties of 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/
Recap: Strong Bisimulation

Outline of Lecture 12

Recap: Strong Bisimulation

Congruence and Deadlock Sensitivity

Buffers Revisited

Epilogue
Recap: Strong Bisimulation

Strong Bisimulation I

Definition (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 $(P', Q') \in \rho$, and

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

Definition (Strong bisimilarity)

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

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

Relation $\sim$ is called strong bisimilarity.
Recap: Strong Bisimulation

Summary So Far

1. $\sim$ is an equivalence relation.
2. $\sim$ is less distinctive than LTS isomorphism.
3. $P \sim Q$ implies that $P$ and $Q$ are trace equivalent.
4. For deterministic $P$ and $Q$: $P \sim Q$ iff $Tr(P) = Tr(Q)$.

Remaining interesting properties:

- $\sim$ is a CCS congruence.
- $\sim$ preserves deadlocks.
Outline of Lecture 12

Recap: Strong Bisimulation

Congruence and Deadlock Sensitivity

Buffers Revisited

Epilogue
Congruence and Deadlock Sensitivity

Congruence

Theorem 12.1 (CCS congruence property of $\sim$)

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

- $\alpha . P \sim \alpha . Q$ for every action $\alpha$
- $P + R \sim Q + R$ for every process $R$
- $P \parallel R \sim Q \parallel R$ for every process $R$
- $P \backslash L \sim Q \backslash L$ for every set $L \subseteq A$
- $P[f] \sim Q[f]$ for every relabelling $f : A \rightarrow A$

Proof.
- for $\parallel$: on the board
- for other CCS operators: left as an exercise
Congruence and Deadlock Sensitivity

Congruence

Theorem 12.1 (CCS congruence property of $\sim$)

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

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

Proof.

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

**Definition (Deadlock; cf. Definition 10.6)**

Let $P, Q \in Prc$ and $w \in Act^*$ such that $P \xrightarrow{w} Q$ and $Q \not\xrightarrow{}$. Then $Q$ is called a $w$-deadlock of $P$.


Concrence and Deadlock Sensitivity

Deadlock Sensitivity of ~

Definition (Deadlock; cf. Definition 10.6)
Let \( P, Q \in Prc \) and \( w \in Act^* \) such that \( P \xrightarrow{w} Q \) and \( Q \not\xrightarrow{} \). Then \( Q \) is called a \( w \)-deadlock of \( P \).

Definition (Deadlock sensitivity; cf. Definition 10.8)
Relation \( \equiv \subseteq Prc \times Prc \) is deadlock sensitive whenever:

\[
P \equiv Q \text{ implies } (\forall w \in Act^*. P \text{ has a } w\text{-deadlock iff } Q \text{ has a } w\text{-deadlock}).
\]
Deadlock Sensitivity of $\sim$

Definition (Deadlock; cf. Definition 10.6)

Let $P, Q \in Prc$ and $w \in Act^*$ such that $P \xrightarrow{w} Q$ and $Q \not\rightarrow$. Then $Q$ is called a $w$-deadlock of $P$.

Definition (Deadlock sensitivity; cf. Definition 10.8)

Relation $\equiv \subseteq Prc \times Prc$ is deadlock sensitive whenever:

$P \equiv Q$ implies $(\forall w \in Act^*. P \text{ has a } w\text{-deadlock iff } Q \text{ has a } w\text{-deadlock})$.

Theorem 12.2

$\sim$ is deadlock sensitive.
Deadlock Sensitivity of \sim

Definition (Deadlock; cf. Definition 10.6)
Let \( P, Q \in \text{Prc} \) and \( w \in \text{Act}^* \) such that \( P \xrightarrow{w} Q \) and \( Q \not\rightarrow \). Then \( Q \) is called a \( w \)-deadlock of \( P \).

Definition (Deadlock sensitivity; cf. Definition 10.8)
Relation \( \equiv \subseteq \text{Prc} \times \text{Prc} \) is deadlock sensitive whenever:
\[
P \equiv Q \text{ implies } (\forall w \in \text{Act}^*. P \text{ has a } w\text{-deadlock iff } Q \text{ has a } w\text{-deadlock}).
\]

Theorem 12.2
\( \sim \) is deadlock sensitive.

Proof.
on the board
Buffers Revisited

Outline of Lecture 12

Recap: Strong Bisimulation

Congruence and Deadlock Sensitivity

Buffers Revisited

Epilogue
Buffers Revisited

Two Buffers

Example 12.3 (One-place buffer)

\[ B^1_0 = in.B^1_1 \]
\[ B^1_1 = out.B^1_0. \]
Buffers Revisited

Two Buffers

Example 12.3 (One-place buffer)

\[
\begin{align*}
B_0^1 &= \text{in}.B_1^1 \\
B_1^1 &= \overline{\text{out}}.B_0^1.
\end{align*}
\]

Example 12.4 (Two-place buffer)

\[
\begin{align*}
B_0^2 &= \text{in}.B_1^2 \\
B_1^2 &= \text{in}.B_2^2 + \overline{\text{out}}.B_0^2 \\
B_2^2 &= \overline{\text{out}}.B_1^2.
\end{align*}
\]
Buffers Revisited

Two Buffers

**Example 12.3 (One-place buffer)**

\[
\begin{align*}
B_0^1 &= \text{in}.B_1^1 \\
B_1^1 &= \text{out}.B_0^1.
\end{align*}
\]

**Example 12.4 (Two-place buffer)**

\[
\begin{align*}
B_0^2 &= \text{in}.B_1^2 \\
B_1^2 &= \text{in}.B_2^2 + \text{out}.B_0^2 \\
B_2^2 &= \text{out}.B_1^2.
\end{align*}
\]

\[B_0^2 \sim B_0^1 \parallel B_0^1\]
Buffers Revisited

Semaphores: A Generalisation

Example 12.5 (An $n$-ary semaphore)

Let $S^n_i$ stand for a semaphore for $n$ resources $i$ of which are taken:

- $S^n_0 = \text{get}.S^n_1$
- $S^n_i = \text{get}.S^n_{i+1} + \text{put}.S^n_{i-1}$ for $0 < i < n$
- $S^n_n = \text{put}.S^n_{n-1}$

This process is strongly bisimilar to $n$ parallel binary semaphores:

Lemma 12.6
For every $n \in \mathbb{N}^+$, we have: $S^n_0 \sim S^1_0 \parallel \cdots \parallel S^1_0 \parallel_n$
Buffers Revisited

Semaphores: A Generalisation

Example 12.5 (An $n$-ary semaphore)

Let $S^n_i$ stand for a semaphore for $n$ resources $i$ of which are taken:

\[
S^n_0 = \text{get}.S^n_1 \\
S^n_i = \text{get}.S^n_{i+1} + \text{put}.S^n_{i-1} \quad \text{for } 0 < i < n \\
S^n_n = \text{put}.S^n_{n-1}
\]

This process is strongly bisimilar to $n$ parallel binary semaphores:

Lemma 12.6

For every $n \in \mathbb{N}_+$, we have: $S^n_0 \sim \underbrace{S^1_0 \parallel \cdots \parallel S^1_0}_n$.
Buffers Revisited

Semaphores II

Lemma

For every $n \in \mathbb{N}_+$, we have: $S^n_0 \sim S^1_0 \parallel \cdots \parallel S^1_0$.  

Proof.
Consider the following binary relation where $i_1, i_2, \ldots, i_n \in \{0, 1\}$:

$\rho = \begin{cases} 
(S^n_0, S^1_0 \parallel \cdots \parallel S^1_0) 
\end{cases}$

Then: $\rho$ is a strong bisimulation and $(S^n_0, S^1_0 \parallel \cdots \parallel S^1_0) \in \rho$. 

n times
Buffers Revisited

Semaphores II

Lemma

For every $n \in \mathbb{N}_+$, we have: $S_0^n \sim S_0^1 \parallel \cdots \parallel S_0^1$ $n$ times.

Proof.

Consider the following binary relation where $i_1, i_2, \ldots, i_n \in \{0, 1\}$:

$$
\rho = \left\{ (S_0^n, S_1^1 \parallel \cdots \parallel S_1^1) \left| \sum_{j=1}^{n} i_j = i \right. \right\}
$$
Lemma

For every $n \in \mathbb{N}_+$, we have: $S^n_0 \sim S^1_0 \parallel \cdots \parallel S^1_0$. 

Proof.

Consider the following binary relation where $i_1, i_2, \ldots, i_n \in \{0, 1\}$:

$$\rho = \left\{ (S^n_i, S^1_{i_1} \parallel \cdots \parallel S^1_{i_n}) \bigg| \sum_{j=1}^n i_j = i \right\}$$

Then: $\rho$ is a strong bisimulation and $(S^n_0, S^1_0 \parallel \cdots \parallel S^1_0) \in \rho$. \qed
Epilogue

Outline of Lecture 12

Recap: Strong Bisimulation

Congruence and Deadlock Sensitivity

Buffers Revisited

Epilogue
Overview of Some Behavioural Equivalences

- Isomorphism
- Bisimulation equivalence
- Ready trace equivalence
- Simulation equivalence
- Failure equivalence
- Trace equivalence
Summary

- Strong bisimulation of processes is based on mutually mimicking each other
Epilogue

Summary

- Strong bisimulation of processes is based on mutually mimicking each other
- Strong bisimilarity $\sim$:
  1. is the largest strong bisimulation
  2. is an equivalence
  3. is a CCS congruence
  4. is strictly finer than trace equivalence
  5. is deadlock sensitive