Properties of strong bisimilarity (reprise)
Bisimulation games
Weak bisimilarity and weak bisimulation games
Properties of weak bisimilarity
Example: a communication protocol and its modelling in CCS
Concurrency workbench (CWB)
Let \((\text{Proc}, \text{Act}, \{ \xrightarrow{a} \mid a \in \text{Act} \})\) be an LTS.

**Strong Bisimulation**

A binary relation \(R \subseteq \text{Proc} \times \text{Proc}\) is a strong bisimulation iff whenever \((s, t) \in R\) then for each \(a \in \text{Act}\):

- if \(s \xrightarrow{a} s'\) then \(t \xrightarrow{a} t'\) for some \(t'\) such that \((s', t') \in R\)
- if \(t \xrightarrow{a} t'\) then \(s \xrightarrow{a} s'\) for some \(s'\) such that \((s', t') \in R\).

**Strong Bisimilarity**

Two processes \(p_1, p_2 \in \text{Proc}\) are strongly bisimilar \((p_1 \sim p_2)\) if and only if there exists a strong bisimulation \(R\) such that \((p_1, p_2) \in R\).

\[\sim = \bigcup \{R \mid R \text{ is a strong bisimulation}\}\]
Basic Properties of Strong Bisimilarity

**Theorem**

\[ \sim \text{ is an equivalence relation (reflexive, symmetric and transitive)} \]

**Theorem**

\[ \sim \text{ is the largest strong bisimulation} \]

**Theorem**

\[ s \sim t \text{ if and only if for each } a \in \text{Act:} \]

- if \( s \xrightarrow{a} s' \) then \( t \xrightarrow{a} t' \) for some \( t' \) such that \( s' \sim t' \)
- if \( t \xrightarrow{a} t' \) then \( s \xrightarrow{a} s' \) for some \( s' \) such that \( s' \sim t' \).
How to Show Nonbisimilarity?

To prove that \( s \not\sim t \):

- Enumerate all binary relations and show that none of them at the same time contains \((s, t)\) and is a strong bisimulation. (Expensive: \(2^{|Proc|^2}\) relations.)
- Make certain observations which enable us to disqualify many bisimulation candidates in one step.
- Use the game characterization of strong bisimilarity.
To prove that \( s \not\sim t \):

- Enumerate all binary relations and show that none of them at the same time contains \((s, t)\) and is a strong bisimulation. (Expensive: \(2^{|Proc|}^2\) relations.)
- Make certain observations which enable us to disqualify many bisimulation candidates in one step.
- Use the game characterization of strong bisimilarity.
To prove that $s \not\sim t$:

- Enumerate all binary relations and show that none of them at the same time contains $(s, t)$ and is a strong bisimulation. (Expensive: $2^{|Proc|^2}$ relations.)

- Make certain observations which enable us to disqualify many bisimulation candidates in one step.

- Use the game characterization of strong bisimilarity.
To prove that $s \not\sim t$:

- Enumerate **all binary relations** and show that none of them at the same time contains $(s, t)$ and is a strong bisimulation. (Expensive: $2^{|Proc|^2}$ relations.)
- Make certain **observations** which enable us to disqualify many bisimulation candidates in one step.
- Use the **game characterization** of strong bisimilarity.
Let \((\text{Proc}, \text{Act}, \{ \xrightarrow{a} \mid a \in \text{Act}\})\) be an LTS and \(s, t \in \text{Proc}\).

We define a two-player game of an ‘attacker’ and a ‘defender’ starting from \(s\) and \(t\).

- The game is played in **rounds**, and configurations of the game are pairs of states from \(\text{Proc} \times \text{Proc}\).
- In every round exactly one configuration is called **current**. Initially the configuration \((s, t)\) is the current one.

**Intuition**

The defender wants to show that \(s\) and \(t\) are strongly bisimilar while the attacker aims at proving the opposite.
Let \((Proc, Act, \{a \rightarrow | a \in Act\})\) be an LTS and \(s, t \in Proc\).

We define a two-player game of an ‘attacker’ and a ‘defender’ starting from \(s\) and \(t\).

- The game is played in \textit{rounds}, and configurations of the game are pairs of states from \(Proc \times Proc\).
- In every round exactly one configuration is called \textit{current}. Initially the configuration \((s, t)\) is the current one.

\textbf{Intuition}

The defender wants to show that \(s\) and \(t\) are strongly bisimilar while the attacker aims at proving the opposite.
Rules of the Bisimulation Games

Game Rules

In each round the players change the current configuration as follows:

1. the attacker chooses one of the processes in the current configuration and makes an $a \xrightarrow{a}$-move for some $a \in \text{Act}$, and
2. the defender must respond by making an $a \xrightarrow{a}$-move in the other process under the same action $a$.

The newly reached pair of processes becomes the current configuration. The game then continues by another round.

Result of the Game

- If one player cannot move, the other player wins.
- If the game is infinite, the defender wins.
Game Rules

In each round the players change the current configuration as follows:

1. the attacker chooses one of the processes in the current configuration and makes an $\xrightarrow{a}$-move for some $a \in \text{Act}$, and
2. the defender must respond by making an $\xrightarrow{a}$-move in the other process under the same action $a$.

The newly reached pair of processes becomes the current configuration. The game then continues by another round.

Result of the Game

- If one player cannot move, the other player wins.
- If the game is infinite, the defender wins.
Theorem

- States $s$ and $t$ are strongly bisimilar if and only if the defender has a universal winning strategy starting from the configuration $(s, t)$.
- States $s$ and $t$ are not strongly bisimilar if and only if the attacker has a universal winning strategy starting from the configuration $(s, t)$.

Remark

The bisimulation game can be used to prove both bisimilarity and nonbisimilarity of two processes. It very often provides elegant arguments for the negative case.
**Theorem**

- States $s$ and $t$ are strongly bisimilar if and only if the defender has a *universal* winning strategy starting from the configuration $(s, t)$.
- States $s$ and $t$ are not strongly bisimilar if and only if the attacker has a *universal* winning strategy starting from the configuration $(s, t)$.

**Remark**

The bisimulation game can be used to prove both bisimilarity and nonbisimilarity of two processes. It very often provides elegant arguments for the negative case.
Theorem

Let $P$ and $Q$ be CCS processes such that $P \sim Q$. Then

- $\alpha . P \sim \alpha . Q$ for each action $\alpha \in \text{Act}$
- $P + R \sim Q + R$ and $R + P \sim R + Q$ for each CCS process $R$
- $P \mid R \sim Q \mid R$ and $R \mid P \sim R \mid Q$ for each CCS process $R$
- $P[f] \sim Q[f]$ for each relabelling function $f$
- $P \setminus L \sim Q \setminus L$ for each set of labels $L$. 
Other Properties of Strong Bisimilarity

The Following Properties Hold for all CCS Processes $P, Q, R$

- $P + Q \sim Q + P$
- $P | Q \sim Q | P$
- $P + Nil \sim P$
- $P | Nil \sim P$
- $(P + Q) + R \sim P + (Q + R)$
- $(P | Q) | R \sim P | (Q | R)$
Example – Buffer

Buffer of Capacity 1

\[ B_0^1 \overset{\text{def}}{=} in.B_1^1 \]
\[ B_1^1 \overset{\text{def}}{=} out.B_0^1 \]

Buffer of Capacity \( n \)

\[ B_0^n \overset{\text{def}}{=} in.B_1^n \]
\[ B_i^n \overset{\text{def}}{=} in.B_{i+1}^n + out.B_{i-1}^n \quad \text{for } 0 < i < n \]
\[ B_n^n \overset{\text{def}}{=} out.B_{n-1}^n \]

Example: \( B_2^2 \sim B_0^1 \| B_0^1 \)
Example – Buffer

**Buffer of Capacity 1**

\[ B_0^1 \overset{\text{def}}{=} \text{in}.B_1^1 \]
\[ B_1^1 \overset{\text{def}}{=} \text{out}.B_0^1 \]

**Buffer of Capacity** \( n \)

\[ B_0^n \overset{\text{def}}{=} \text{in}.B_1^n \]
\[ B_i^n \overset{\text{def}}{=} \text{in}.B_{i+1}^n + \text{out}.B_{i-1}^n \quad \text{for } 0 < i < n \]
\[ B_n^n \overset{\text{def}}{=} \text{out}.B_{n-1}^n \]

**Example:** \( B_0^2 \sim B_0^1 \parallel B_1^1 \)

![Diagram of buffer transitions](image-url)
Example – Buffer

Buffer of Capacity 1

\[ B_0^1 \overset{\text{def}}{=} in.B_1^1 \]
\[ B_1^1 \overset{\text{def}}{=} out.B_0^1 \]

Buffer of Capacity \( n \)

\[ B_0^n \overset{\text{def}}{=} in.B_1^n \]
\[ B_i^n \overset{\text{def}}{=} in.B_{i+1}^n + \overline{out}.B_{i-1}^n \quad \text{for } 0 < i < n \]
\[ B_n^n \overset{\text{def}}{=} \overline{out}.B_{n-1}^n \]

Example: \( B_0^2 \sim B_0^1 \mid B_1^1 \)
Example – Buffer

**Theorem**

For all natural numbers $n$: $B^n_0 \sim B^1_0|B^1_0|\cdots|B^1_0$

$n$ times

**Proof.**

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

$$R = \{(B^n_{i_1}, B^1_{i_2}|B^1_{i_2}|\cdots|B^1_{i_n}) \mid \sum_{j=1}^{n} i_j = i\}$$

- $(B^n_0, B^1_0|B^1_0|\cdots|B^1_0) \in R$
- $R$ is strong bisimulation
Theorem

For all natural numbers $n$: $B^n_0 \sim \overbrace{B^1_0|B^1_0|\cdots|B^1_0}^{n \text{ times}}$

Proof.

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

$$R = \{(B^n_i, B^1_{i_1}|B^1_{i_2}|\cdots|B^1_{i_n}) \mid \sum_{j=1}^{n} i_j = i\}$$

- $(B^n_0, B^1_0|B^1_0|\cdots|B^1_0) \in R$
- $R$ is strong bisimulation
Properties of $\sim$

- an equivalence relation
- the largest strong bisimulation
- a congruence
- enough to prove some natural rules like
  - $P|Q \sim Q|P$
  - $P|\text{Nil} \sim P$
  - $(P|Q)|R \sim Q|(P|R)$
  - $\ldots$

Question

Should we look any further???
## Properties of \( \sim \)

- an equivalence relation
- the largest strong bisimulation
- a congruence
- enough to prove some natural rules like
  - \( P|Q \sim Q|P \)
  - \( P|\text{Nil} \sim P \)
  - \( (P|Q)|R \sim Q|(P|R) \)
  - \( \ldots \)

## Question

Should we look any further???
Question
Does $a.\tau.Nil \sim a.Nil$ hold? NO!

Problem
Strong bisimilarity does not abstract away from $\tau$ actions.

Example: SmUni $\not\sim$ Spec

SmUni
\[\downarrow^{pub}\]
$(CM_1 \mid CS_1) \setminus \{\text{coin, coffee}\}$
\[\downarrow^{\tau}\]
$(CM_1 \mid CS_2) \setminus \{\text{coin, coffee}\}$

Spec
\[\uparrow^{pub}\]
$(CM \mid CS_1 \setminus \{\text{coin, coffee}\})$

EWSCS'07–Lecture 4 Reactive Systems: Modelling, Specification and Verification
Problems with Internal Actions

Question
Does \( a.\tau.\text{Nil} \sim a.\text{Nil} \) hold? NO!

Problem
Strong bisimilarity does not abstract away from \( \tau \) actions.

Example: SmUni \( \not\sim \) Spec

\[
\begin{align*}
\text{SmUni} & \downarrow_{\text{pub}} \\
(CM \mid CS_1) & \downarrow \{\text{coin, coffee}\} \downarrow_\tau \\
(CM_1 \mid CS_2) & \downarrow \{\text{coin, coffee}\} \downarrow_\tau \\
(CM \mid CS) & \downarrow \{\text{coin, coffee}\} \\
\end{align*}
\]

\[
\begin{align*}
\text{Spec} & \uparrow_{\text{pub}} \\
\end{align*}
\]
Problems with Internal Actions

Question

Does $a.\tau.\text{Nil} \sim a.\text{Nil}$ hold? NO!

Problem

Strong bisimilarity does not abstract away from $\tau$ actions.

Example: SmUni $\not\sim$ Spec

\[
\begin{align*}
\text{SmUni} & \xrightarrow{\text{pub}} (CM | CS_1) \setminus \{\text{coin, coffee}\} \\
& \xrightarrow{\tau} (CM_1 | CS_2) \setminus \{\text{coin, coffee}\} \\
& \xrightarrow{\tau} (CM | CS) \setminus \{\text{coin, coffee}\} \\
\text{Spec} & \xrightarrow{\text{pub}}
\end{align*}
\]
Problems with Internal Actions

Question
Does $a.\tau.Nil \sim a.Nil$ hold? NO!

Problem
Strong bisimilarity does not abstract away from $\tau$ actions.

Example: SmUni $\not\sim$ Spec

\[
\begin{align*}
\text{SmUni} & \xrightarrow{\text{pub}} (CM | CS_1) \xleftarrow{\text{pub}} \{\text{coin, coffee}\} \\
& \xrightarrow{\tau} (CM_1 | CS_2) \xleftarrow{\text{pub}} \{\text{coin, coffee}\} \\
& \xrightarrow{\tau} (CM | CS) \xleftarrow{\text{pub}} \{\text{coin, coffee}\}
\end{align*}
\]

\[
\begin{align*}
\text{Spec} & \xrightarrow{\tau} (CM_1 | CS_2) \xleftarrow{\text{pub}} \{\text{coin, coffee}\} \\
& \xrightarrow{\tau} (CM | CS) \xleftarrow{\text{pub}} \{\text{coin, coffee}\}
\end{align*}
\]
Let \((\text{Proc}, \text{Act}, \{ \xrightarrow{a} \mid a \in \text{Act} \})\) be an LTS such that \(\tau \in \text{Act}\).

**Definition of Weak Transition Relation**

\[
\xrightarrow{a} = \begin{cases} 
(\xrightarrow{\tau})^* \circ \xrightarrow{a} \circ (\xrightarrow{\tau})^* & \text{if } a \neq \tau \\
(\xrightarrow{\tau})^* & \text{if } a = \tau
\end{cases}
\]

**What does \(s \xrightarrow{a} t\) informally mean?**

- If \(a \neq \tau\) then \(s \xrightarrow{a} t\) means that from \(s\) we can get to \(t\) by doing zero or more \(\tau\) actions, followed by the action \(a\), followed by zero or more \(\tau\) actions.
- If \(a = \tau\) then \(s \xrightarrow{\tau} t\) means that from \(s\) we can get to \(t\) by doing zero or more \(\tau\) actions.
Let \((\text{Proc}, \text{Act}, \{ \xrightarrow{a} \mid a \in \text{Act} \})\) be an LTS such that \(\tau \in \text{Act}\).

**Definition of Weak Transition Relation**

\[
\xrightarrow{a} = \begin{cases} 
(\tau \xrightarrow{\cdot})^* \xrightarrow{a} \circ (\tau \xrightarrow{\cdot})^* & \text{if } a \neq \tau \\
(\tau \xrightarrow{\cdot})^* & \text{if } a = \tau 
\end{cases}
\]

**What does \(s \xrightarrow{a} t\) informally mean?**

- If \(a \neq \tau\) then \(s \xrightarrow{a} t\) means that from \(s\) we can get to \(t\) by doing zero or more \(\tau\) actions, followed by the action \(a\), followed by zero or more \(\tau\) actions.
- If \(a = \tau\) then \(s \xrightarrow{\tau} t\) means that from \(s\) we can get to \(t\) by doing zero or more \(\tau\) actions.
Weak Bisimilarity

Let \((\text{Proc}, \text{Act}, \{a \rightarrow \mid a \in \text{Act}\})\) be an LTS such that \(\tau \in \text{Act}\).

**Weak Bisimulation**

A binary relation \(R \subseteq \text{Proc} \times \text{Proc}\) is a **weak bisimulation** iff whenever \((s, t) \in R\) then for each \(a \in \text{Act}\) (including \(\tau\)):

- if \(s \xrightarrow{a} s'\) then \(t \xrightarrow{a} t'\) for some \(t'\) such that \((s', t') \in R\)
- if \(t \xrightarrow{a} t'\) then \(s \xrightarrow{a} s'\) for some \(s'\) such that \((s', t') \in R\).

**Weak Bisimilarity**

Two processes \(p_1, p_2 \in \text{Proc}\) are **weakly bisimilar** \((p_1 \approx p_2)\) if and only if there exists a weak bisimulation \(R\) such that \((p_1, p_2) \in R\).

\[
\approx = \bigcup \{R \mid R \text{ is a weak bisimulation}\}
\]
Weak Bisimilarity

Let \((\mathit{Proc}, \mathit{Act}, \{ \xrightarrow{a} \mid a \in \mathit{Act} \})\) be an LTS such that \(\tau \in \mathit{Act}\).

Weak Bisimulation

A binary relation \(R \subseteq \mathit{Proc} \times \mathit{Proc}\) is a **weak bisimulation** iff whenever \((s, t) \in R\) then for each \(a \in \mathit{Act}\) (including \(\tau\)):

- if \(s \xrightarrow{a} s'\) then \(t \xrightarrow{a} t'\) for some \(t'\) such that \((s', t') \in R\)
- if \(t \xrightarrow{a} t'\) then \(s \xrightarrow{a} s'\) for some \(s'\) such that \((s', t') \in R\).

Weak Bisimilarity

Two processes \(p_1, p_2 \in \mathit{Proc}\) are **weakly bisimilar** \((p_1 \approx p_2)\) if and only if there exists a weak bisimulation \(R\) such that \((p_1, p_2) \in R\).

\[\approx = \bigcup \{ R \mid R \text{ is a weak bisimulation} \}\]
Definition

All the same except that
- defender can now answer using $\xrightarrow{a}$ moves.
The attacker is still using only $\xrightarrow{a}$ moves.

Theorem

- States $s$ and $t$ are weakly bisimilar if and only if the defender has a universal winning strategy starting from the configuration $(s, t)$.
- States $s$ and $t$ are not weakly bisimilar if and only if the attacker has a universal winning strategy starting from the configuration $(s, t)$. 
**Definition**

All the same except that
- defender can now answer using $\xrightarrow{a}$ moves.

The attacker is still using only $\xrightarrow{a}$ moves.

**Theorem**

- States $s$ and $t$ are weakly bisimilar if and only if the defender has a *universal* winning strategy starting from the configuration $(s, t)$.

- States $s$ and $t$ are not weakly bisimilar if and only if the attacker has a *universal* winning strategy starting from the configuration $(s, t)$.
Weak Bisimilarity – Properties

Properties of $\approxeq$

- an equivalence relation
- the largest weak bisimulation
- validates lots of natural laws, e.g.
  - $a.\tau. P \approxeq a. P$
  - $P + \tau. P \approxeq \tau. P$
  - $a.(P + \tau. Q) \approxeq a.(P + \tau. Q) + a. Q$
  - $P + Q \approxeq Q + P$ $P\mid Q \approxeq Q\mid P$ $P + \text{Nil} \approxeq P$ ...
- strong bisimilarity is included in weak bisimilarity ($\sim \subseteq \approxeq$)
- abstracts from $\tau$ loops

\[ \begin{array}{c}
\tau \quad \Downarrow \quad \text{a} \\
\approx \quad \Downarrow \quad \text{a}
\end{array} \]
Case Study: Communication Protocol

Send $\text{def} = \text{acc.Sending}$

Sending $\text{def} = \text{send.Wait}$

Wait $\text{def} = \text{ack.Send + error.Sending}$

Rec $\text{def} = \text{trans.Del}$

Del $\text{def} = \text{del.Ack}$

Ack $\text{def} = \text{ack.Rec}$

Med $\text{def} = \text{send.Med'}$

Med' $\text{def} = \tau.\text{Err + trans.Med}$

Err $\text{def} = \text{error.Med}$
Case Study: Communication Protocol

Send \( \overset{\text{def}}{=} \) acc.Sending
Send \( \overset{\text{def}}{=} \) send.Wait
Wait \( \overset{\text{def}}{=} \) ack.Send + error.Sending
Rec \( \overset{\text{def}}{=} \) trans.Del
Del \( \overset{\text{def}}{=} \) del.Ack
Ack \( \overset{\text{def}}{=} \) ack.Rec

Med \( \overset{\text{def}}{=} \) send.Med'
Med' \( \overset{\text{def}}{=} \) \(\tau\).Err + trans.Med
Err \( \overset{\text{def}}{=} \) error.Med
\text{Impl} \overset{\text{def}}{=} (\text{Send} \mid \text{Med} \mid \text{Rec}) \setminus \{\text{send, trans, ack, error}\}

\text{Spec} \overset{\text{def}}{=} \text{acc.del.Spec}

\textbf{Question}

\text{Impl} \approx \text{Spec}

1. Draw the LTS of \text{Impl} and \text{Spec} and prove (by hand) the equivalence.

2. Use Concurrency WorkBench (CWB).
\[ \text{Impl} \triangleq (\text{Send} \mid \text{Med} \mid \text{Rec}) \setminus \{\text{send, trans, ack, error}\} \]

\[ \text{Spec} \triangleq \text{acc.del.Spec} \]

1. Draw the LTS of Impl and Spec and prove (by hand) the equivalence.
2. Use Concurrency WorkBench (CWB).
Impl $\overset{\text{def}}{=} (\text{Send} \mid \text{Med} \mid \text{Rec}) \setminus \{\text{send}, \text{trans}, \text{ack}, \text{error}\}$

Spec $\overset{\text{def}}{=} \text{acc.del.Spec}$

**Question**

Impl $\approx$ Spec

1. Draw the LTS of Impl and Spec and prove (by hand) the equivalence.
2. Use Concurrency WorkBench (CWB).
Impl \overset{\text{def}}{=} (\text{Send} \mid \text{Med} \mid \text{Rec}) \setminus \{\text{send, trans, ack, error}\}

Spec \overset{\text{def}}{=} \text{acc.del.Spec}

\begin{enumerate}
  \item Draw the LTS of Impl and Spec and prove (by hand) the equivalence.
  \item Use Concurrency WorkBench (CWB).
\end{enumerate}
\[ \text{Impl} \overset{\text{def}}{=} (\text{Send} \mid \text{Med} \mid \text{Rec}) \setminus \{\text{send, trans, ack, error}\} \]

\[ \text{Spec} \overset{\text{def}}{=} \text{acc.del.Spec} \]

**Question**

1. Draw the LTS of Impl and Spec and prove (by hand) the equivalence.
2. Use Concurrency WorkBench (CWB).
CCS Expressions in CWB

**CCS Definitions**

\[
\begin{align*}
\text{Med} & \equiv \text{send}.\text{Med}' \\
\text{Med}' & \equiv \tau.\text{Err} + \text{trans}.\text{Med} \\
\text{Err} & \equiv \text{error}.\text{Med} \\
\text{Impl} & \equiv (\text{Send} | \text{Med} | \text{Rec}) \setminus \{\text{send}, \text{trans}, \text{ack}, \text{error}\} \\
\text{Spec} & \equiv \text{acc}.\text{del}.\text{Spec}
\end{align*}
\]

**CWB Program (protocol.cwb)**

```
agent Med = send.Med';
agent Med' = (tau.Err + 'trans.Med);
agent Err = 'error.Med;
agent Impl = (Send | Med | Rec) \ set L = \{send, trans, ack, error\};
agent Spec = acc.'del.Spec;
```
[luca@vel5638 CWB]$ ./xccscwb.x86-linux

> help;

> input "protocol.cwb";

> vs(5,Impl);

> sim(Spec);

> eq(Spec,Impl);  ** weak bisimilarity **

> strongeq(Spec,Impl);  ** strong bisimilarity **
Is Weak Bisimilarity a Congruence for CCS?

Theorem

Let $P$ and $Q$ be CCS processes such that $P \approx Q$. Then

- $\alpha.P \approx \alpha.Q$ for each action $\alpha \in \text{Act}$
- $P | R \approx Q | R$ and $R | P \approx R | Q$ for each CCS process $R$
- $P[f] \approx Q[f]$ for each relabelling function $f$
- $P \setminus L \approx Q \setminus L$ for each set of labels $L$.

What about choice?

$\tau.a.Nil \approx a.Nil$ but $\tau.a.Nil + b.Nil \not\approx a.Nil + b.Nil$

Conclusion

Weak bisimilarity is not a congruence for CCS.
Theorem

Let $P$ and $Q$ be CCS processes such that $P \approx Q$. Then

- $\alpha.P \approx \alpha.Q$ for each action $\alpha \in \text{Act}$
- $P \mid R \approx Q \mid R$ and $R \mid P \approx R \mid Q$ for each CCS process $R$
- $P[f] \approx Q[f]$ for each relabelling function $f$
- $P \setminus L \approx Q \setminus L$ for each set of labels $L$.

What about choice?

$\tau.a.Nil \approx a.Nil$ but $\tau.a.Nil + b.Nil \not\approx a.Nil + b.Nil$

Conclusion

Weak bisimilarity is not a congruence for CCS.
### Theorem

Let $P$ and $Q$ be CCS processes such that $P \approx Q$. Then

- $\alpha.P \approx \alpha.Q$ for each action $\alpha \in \text{Act}$
- $P \parallel R \approx Q \parallel R$ and $R \parallel P \approx R \parallel Q$ for each CCS process $R$
- $P[f] \approx Q[f]$ for each relabelling function $f$
- $P \backslash L \approx Q \backslash L$ for each set of labels $L$.

### What about choice?

$\tau.a.Nil \approx a.Nil$ but $\tau.a.Nil + b.Nil \not\approx a.Nil + b.Nil$

### Conclusion

Weak bisimilarity is **not** a congruence for CCS.