Strong Bisimilarity – Summary

Properties of $\sim$

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

Question

Should we look any further???

Strong Bisimilarity – Summary

Properties of $\sim$

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

Question

Should we look any further???

Problems with Internal Actions

Question

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

Problem

Strong bisimilarity does not abstract away from τ actions.

Example: SmUni $\not\sim$ Spec

Problems with Internal Actions

Question

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

Problem

Strong bisimilarity does not abstract away from τ actions.

Example: SmUni $\not\sim$ Spec

Problems with Internal Actions

Question

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

Problem

Strong bisimilarity does not abstract away from τ actions.

Example: SmUni $\not\sim$ Spec

Problems with Internal Actions

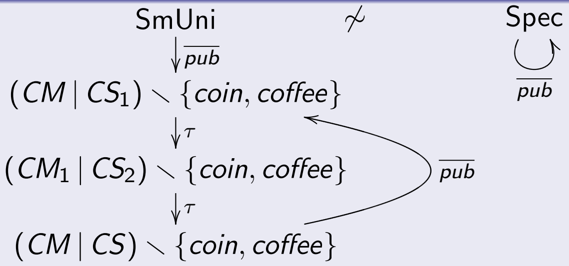
Question

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

Problem

Strong bisimilarity does not abstract away from τ actions.

Example: SmUni $\not\sim$ Spec



Weak Transition Relation

Let $(Proc, Act, \{\stackrel{a}{\longrightarrow} \mid a \in Act\})$ be an LTS such that $\tau \in Act$.

Definition of Weak Transition Relation

$$\stackrel{a}{\Longrightarrow} \; = \begin{cases} (\stackrel{\tau}{\longrightarrow})^* \circ \stackrel{a}{\longrightarrow} \circ (\stackrel{\tau}{\longrightarrow})^* & \text{if } a \neq \tau \\ (\stackrel{\tau}{\longrightarrow})^* & \text{if } a = \tau \end{cases}$$

What does $s \stackrel{a}{\Longrightarrow} t$ informally mean?

- If $a \neq \tau$ then $s \stackrel{a}{\Longrightarrow} t$ means that
  from s we can get to t by doing zero or more τ actions, followed by the action a, followed by zero or more τ actions.
- If $a = \tau$ then $s \stackrel{\tau}{\Longrightarrow} t$ means that
  from s we can get to t by doing zero or more τ actions.

Weak Transition Relation

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

Definition of Weak Transition Relation

$$\stackrel{a}{\Longrightarrow} \; = \begin{cases} (\xrightarrow{\tau})^* \circ \xrightarrow{a} \circ (\xrightarrow{\tau})^* & \text{if } a \neq \tau \\ (\xrightarrow{\tau})^* & \text{if } a = \tau \end{cases}$$

What does $s \stackrel{a}{\Longrightarrow} t$ informally mean?

- If $a \neq \tau$ then $s \stackrel{a}{\Longrightarrow} t$ means that
  from s we can get to t by doing zero or more τ actions, followed by the action a, followed by zero or more τ actions.
- If $a = \tau$ then $s \stackrel{\tau}{\Longrightarrow} t$ means that
  from s we can get to t by doing zero or more τ actions.

Weak Bisimilarity

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

Weak Bisimulation

A binary relation $R \subseteq Proc \times Proc$ is a weak bisimulation iff whenever $(s, t) \in R$ then for each $a \in Act$ (including τ):

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

Weak Bisimilarity

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

$$\approx \; = \; \cup\{R \mid R \text{ is a weak bisimulation}\}$$

Weak Bisimilarity

Let $(Proc, Act, \{\stackrel{a}{\longrightarrow} \mid a \in Act\})$ be an LTS such that $\tau \in Act$.

Weak Bisimulation

A binary relation $R \subseteq Proc \times Proc$ is a weak bisimulation iff whenever $(s, t) \in R$ then for each $a \in Act$ (including τ):

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

Weak Bisimilarity

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

$$\approx \; = \; \cup\{R \mid R \text{ is a weak bisimulation}\}$$

Weak Bisimulation Game

Definition

All the same except that

- defender can now answer using $\stackrel{a}{\Longrightarrow}$ moves.

The attacker is still using only $\stackrel{a}{\longrightarrow}$ 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 Bisimulation Game

Definition

All the same except that

- defender can now answer using $\stackrel{a}{\Longrightarrow}$ moves.

The attacker is still using only $\stackrel{a}{\longrightarrow}$ 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 $\approx$

- an equivalence relation
- the largest weak bisimulation
- validates lots of natural laws, e.g.
  - $a.\tau.P \approx a.P$
  - $P + \tau.P \approx \tau.P$
  - $a.(P + \tau.Q) \approx a.(P + \tau.Q) + a.Q$
  - $P + Q \approx Q + P \qquad P|Q \approx Q|P \qquad P + Nil \approx P \qquad \ldots$
- strong bisimilarity is included in weak bisimilarity ($\sim \subseteq \approx$)
- abstracts from τ loops

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* $\alpha \in Act$
- $P \mid R \approx Q \mid R$ *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.

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* $\alpha \in Act$
- $P \mid R \approx Q \mid R$ *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.

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* $\alpha \in Act$
- $P \mid R \approx Q \mid R$ *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.

Case Study: Communication Protocol

$$\begin{aligned}
\mathsf{Send} &\stackrel{\text{def}}{=} \mathsf{acc.Sending} & \mathsf{Rec} &\stackrel{\text{def}}{=} \mathsf{trans.Del} \\
\mathsf{Sending} &\stackrel{\text{def}}{=} \overline{\mathsf{send}}.\mathsf{Wait} & \mathsf{Del} &\stackrel{\text{def}}{=} \overline{\mathsf{del}}.\mathsf{Ack} \\
\mathsf{Wait} &\stackrel{\text{def}}{=} \mathsf{ack.Send} + \mathsf{error.Sending} & \mathsf{Ack} &\stackrel{\text{def}}{=} \overline{\mathsf{ack}}.\mathsf{Rec}
\end{aligned}$$

$$\begin{aligned}
\mathsf{Med} &\stackrel{\text{def}}{=} \mathsf{send.Med}' \\
\mathsf{Med}' &\stackrel{\text{def}}{=} \tau.\mathsf{Err} + \overline{\mathsf{trans}}.\mathsf{Med} \\
\mathsf{Err} &\stackrel{\text{def}}{=} \overline{\mathsf{error}}.\mathsf{Med}
\end{aligned}$$

Case Study: Communication Protocol

$$\mathsf{Send} \stackrel{\text{def}}{=} \mathsf{acc.Sending}$$
$$\mathsf{Sending} \stackrel{\text{def}}{=} \overline{\mathsf{send}}.\mathsf{Wait}$$
$$\mathsf{Wait} \stackrel{\text{def}}{=} \mathsf{ack.Send} + \mathsf{error.Sending}$$

$$\mathsf{Rec} \stackrel{\text{def}}{=} \mathsf{trans.Del}$$
$$\mathsf{Del} \stackrel{\text{def}}{=} \overline{\mathsf{del}}.\mathsf{Ack}$$
$$\mathsf{Ack} \stackrel{\text{def}}{=} \overline{\mathsf{ack}}.\mathsf{Rec}$$

$$\mathsf{Med} \stackrel{\text{def}}{=} \mathsf{send.Med}'$$
$$\mathsf{Med}' \stackrel{\text{def}}{=} \tau.\mathsf{Err} + \overline{\mathsf{trans}}.\mathsf{Med}$$
$$\mathsf{Err} \stackrel{\text{def}}{=} \overline{\mathsf{error}}.\mathsf{Med}$$

Verification Question

$$\mathrm{Impl} \stackrel{\mathrm{def}}{=} (\mathrm{Send} \mid \mathrm{Med} \mid \mathrm{Rec}) \smallsetminus \{\mathrm{send}, \mathrm{trans}, \mathrm{ack}, \mathrm{error}\}$$

$$\mathrm{Spec} \stackrel{\mathrm{def}}{=} \mathrm{acc}.\overline{\mathrm{del}}.\mathrm{Spec}$$

Question

$$\mathrm{Impl} \stackrel{?}{\approx} \mathrm{Spec}$$

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

Verification Question

$$\mathsf{Impl} \stackrel{\mathrm{def}}{=} (\mathsf{Send} \mid \mathsf{Med} \mid \mathsf{Rec}) \smallsetminus \{\mathsf{send}, \mathsf{trans}, \mathsf{ack}, \mathsf{error}\}$$

$$\mathsf{Spec} \stackrel{\mathrm{def}}{=} \mathsf{acc}.\overline{\mathsf{del}}.\mathsf{Spec}$$

Question

$$\mathsf{Impl} \stackrel{?}{\approx} \mathsf{Spec}$$

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

Verification Question

$$\mathrm{Impl} \stackrel{\mathrm{def}}{=} (\mathrm{Send} \mid \mathrm{Med} \mid \mathrm{Rec}) \smallsetminus \{\mathrm{send}, \mathrm{trans}, \mathrm{ack}, \mathrm{error}\}$$

$$\mathrm{Spec} \stackrel{\mathrm{def}}{=} \mathrm{acc}.\overline{\mathrm{del}}.\mathrm{Spec}$$

Question

$$\mathrm{Impl} \stackrel{?}{\approx} \mathrm{Spec}$$

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

Verification Question

$$\mathsf{Impl} \stackrel{\text{def}}{=} (\mathsf{Send} \mid \mathsf{Med} \mid \mathsf{Rec}) \smallsetminus \{\mathsf{send}, \mathsf{trans}, \mathsf{ack}, \mathsf{error}\}$$

$$\mathsf{Spec} \stackrel{\text{def}}{=} \mathsf{acc}.\overline{\mathsf{del}}.\mathsf{Spec}$$

Question

$$\mathsf{Impl} \stackrel{?}{\approx} \mathsf{Spec}$$

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

Verification Question

$$\mathsf{Impl} \stackrel{\mathrm{def}}{=} (\mathsf{Send} \mid \mathsf{Med} \mid \mathsf{Rec}) \smallsetminus \{\mathsf{send}, \mathsf{trans}, \mathsf{ack}, \mathsf{error}\}$$

$$\mathsf{Spec} \stackrel{\mathrm{def}}{=} \mathsf{acc}.\overline{\mathsf{del}}.\mathsf{Spec}$$

Question

$$\mathsf{Impl} \stackrel{?}{\approx} \mathsf{Spec}$$

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

$\mathrm{Med} \stackrel{\mathrm{def}}{=} \mathrm{send}.\mathrm{Med}'$

$\mathrm{Med}' \stackrel{\mathrm{def}}{=} \tau.\mathrm{Err} + \overline{\mathrm{trans}}.\mathrm{Med}$

$\mathrm{Err} \stackrel{\mathrm{def}}{=} \overline{\mathrm{error}}.\mathrm{Med}$

$\vdots$

$\mathrm{Impl} \stackrel{\mathrm{def}}{=} (\mathrm{Send} \mid \mathrm{Med} \mid \mathrm{Rec}) \smallsetminus \{\mathrm{send}, \mathrm{trans}, \mathrm{ack}, \mathrm{error}\}$

$\mathrm{Spec} \stackrel{\mathrm{def}}{=} \mathrm{acc}.\overline{\mathrm{del}}.\mathrm{Spec}$

CWB Program (protocol.cwb)

```
agent Med = send.Med';
agent Med' = (tau.Err + 'trans.Med);
agent Err = 'error.Med;
⋮
set L = {send, trans, ack, error};
agent Impl = (Send | Med | Rec) \ L;

agent Spec = acc.'del.Spec;
```

CWB Session

```
fire1$ /pack/FS/CWB/cwb

> help;

> input "protocol.cwb";

> vs(5,Impl);

> sim(Spec);

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

> strongeq(Spec,Impl);        ** strong bisimilarity **
```