Open Bisimulation, Revisited

S. Briais U. Nestmann

School of Computer and Communication Sciences
École Polytechnique Fédérale de Lausanne

COMETE-PARSIFAL Seminar
11 May, 2006
Paris, FRANCE

Outline

Outline

Syntax

- Processes

$$P,Q \quad ::= \quad \mathbf{0} \quad \mid \quad !\,P \quad \mid \quad P \| Q \quad \mid \quad P + Q$$
$$\mid \quad \pi.P \quad \mid \quad (\nu z)\,P \quad \mid \quad [x = y]P$$

- Prefixes

$$\pi \quad ::= \quad \tau \quad \mid \quad x(z) \quad \mid \quad \overline{x}\langle z \rangle$$

Syntax

- Processes

$$P, Q \quad ::= \quad \mathbf{0} \;\mid\; !P \;\mid\; P \| Q \;\mid\; P + Q \;\mid\; \pi.P \;\mid\; (\nu z)P \;\mid\; [x = y]P$$

- Prefixes

$$\pi \quad ::= \quad \tau \;\mid\; x(z) \;\mid\; \overline{x}\langle z\rangle$$

- Only names

Late Labelled Semantics

$$\textsc{Input}\ \frac{}{a(x).P \xrightarrow{a(x)} P} \qquad \textsc{Open}\ \frac{P \xrightarrow{\bar{a}\,z} P'}{(\nu z)\,P \xrightarrow{(\nu z)\,\bar{a}\,z} P'}\ z \neq a$$

$$\textsc{Close-L}\ \frac{P \xrightarrow{a(x)} P' \qquad Q \xrightarrow{(\nu z)\,\bar{a}\,z} Q'}{P \| Q \xrightarrow{\tau} (\nu z)\,(P'\{^{z}/_{x}\} \| Q')}\ z \notin \mathrm{fn}(P)$$

Outline

Bisimulation

- Proof techniques for showing process equivalence

Bisimulation

- Proof techniques for showing process equivalence
- Wide variety of bisimulations: ground, early, late, open, ...

Bisimulation

- Proof techniques for showing process equivalence
- Wide variety of bisimulations: ground, early, late, open, . . .
- The above cited differ on how they handle substitutions

Bisimulation

- Proof techniques for showing process equivalence
- Wide variety of bisimulations: ground, early, late, open, . . .
- The above cited differ on how they handle substitutions
- For example, ground: no substitutions at all

Bisimulation

- Proof techniques for showing process equivalence
- Wide variety of bisimulations: ground, early, late, open, . . .
- The above cited differ on how they handle substitutions
- For example, ground: no substitutions at all

$$P$$
$$|$$
$$Q$$

Bisimulation

- Proof techniques for showing process equivalence
- Wide variety of bisimulations: ground, early, late, open, . . .
- The above cited differ on how they handle substitutions
- For example, ground: no substitutions at all

$$\begin{array}{ccc} P & \xrightarrow{\alpha} & P' \\ | & & \\ Q & & \end{array}$$

Bisimulation

- Proof techniques for showing process equivalence
- Wide variety of bisimulations: ground, early, late, open, . . .
- The above cited differ on how they handle substitutions
- For example, ground: no substitutions at all

$$\begin{array}{ccc} P & \xrightarrow{\alpha} & P' \\ | & & \\ Q & \xrightarrow{\alpha} & \end{array}$$

Bisimulation

- Proof techniques for showing process equivalence
- Wide variety of bisimulations: ground, early, late, open, ...
- The above cited differ on how they handle substitutions
- For example, ground: no substitutions at all

$$\begin{array}{ccc} P & \xrightarrow{\alpha} & P' \\ | & & \\ Q & \xrightarrow{\alpha} & Q' \end{array}$$

Bisimulation

- Proof techniques for showing process equivalence
- Wide variety of bisimulations: ground, early, late, open, ...
- The above cited differ on how they handle substitutions
- For example, ground: no substitutions at all

$$\begin{array}{ccc} P & \xrightarrow{\alpha} & P' \\ | & & | \\ Q & \xrightarrow{\alpha} & Q' \end{array}$$

Substitutions

A substitution

- has finite domain

Substitutions

A substitution

- has finite domain
- replaces something (a name) by something (e.g.: a name)

Substitutions

A substitution

- has finite domain
- replaces something (a name) by something (e.g.: a name)
- can be lifted to bigger entities (e.g.: messages)

Substitutions

A substitution

- has finite domain
- replaces something (a name) by something (e.g.: a name)
- can be lifted to bigger entities (e.g.: messages)

Some questions when designing a bisimulation:

Substitutions

A substitution

- has finite domain
- replaces something (a name) by something (e.g.: a name)
- can be lifted to bigger entities (e.g.: messages)

Some questions when designing a bisimulation:

- When should substitutions be applied?

Substitutions

A substitution

- has finite domain
- replaces something (a name) by something (e.g.: a name)
- can be lifted to bigger entities (e.g.: messages)

Some questions when designing a bisimulation:

- When should substitutions be applied?
- Which names are substitutable?

Substitutions

A substitution

- has finite domain
- replaces something (a name) by something (e.g.: a name)
- can be lifted to bigger entities (e.g.: messages)

Some questions when designing a bisimulation:

- When should substitutions be applied?
- Which names are substitutable?
- By what?

Early and late bisimulation

- The *symmetric* relation $\mathcal{R} \subset \mathcal{P} \times \mathcal{P}$ is

Early and late bisimulation

- The *symmetric* relation $\mathcal{R} \subset \mathcal{P} \times \mathcal{P}$ is
  - an *early* bisimulation if for all $(P, Q) \in \mathcal{R}$, if $P \xrightarrow{\alpha} P'$ then

Early and late bisimulation

- The *symmetric* relation $\mathcal{R} \subset \mathcal{P} \times \mathcal{P}$ is
  - an *early* bisimulation if for all $(P, Q) \in \mathcal{R}$, if $P \xrightarrow{\alpha} P'$ then
    i) if α is not an input,

Early and late bisimulation

- The *symmetric* relation $\mathcal{R} \subset \mathcal{P} \times \mathcal{P}$ is
  - an *early* bisimulation if for all $(P, Q) \in \mathcal{R}$, if $P \xrightarrow{\alpha} P'$ then
    i) if α is not an input,
    there exists Q' such that $Q \xrightarrow{\alpha} Q'$ and $P' \mathcal{R} \, Q'$

Early and late bisimulation

- The *symmetric* relation $\mathcal{R} \subset \mathcal{P} \times \mathcal{P}$ is
  - an *early* bisimulation if for all $(P, Q) \in \mathcal{R}$, if $P \xrightarrow{\alpha} P'$ then
    i) if α is not an input,
    there exists Q' such that $Q \xrightarrow{\alpha} Q'$ and $P' \mathcal{R}\, Q'$
    ii) if $\alpha = a(x)$,

Early and late bisimulation

- The *symmetric* relation $\mathcal{R} \subset \mathcal{P} \times \mathcal{P}$ is
  - an *early* bisimulation if for all $(P, Q) \in \mathcal{R}$, if $P \xrightarrow{\alpha} P'$ then
    i) if α is not an input,
    there exists Q' such that $Q \xrightarrow{\alpha} Q'$ and $P' \mathcal{R}\, Q'$
    ii) if $\alpha = a(x)$,
    then for all $u \in \mathcal{N}$, there exists Q' such that $Q \xrightarrow{\alpha} Q'$ and $(P'\{^u/_x\}, Q'\{^u/_x\}) \in \mathcal{R}$

Early and late bisimulation

- The *symmetric* relation $\mathcal{R} \subset \mathcal{P} \times \mathcal{P}$ is
  - an *early* bisimulation if for all $(P, Q) \in \mathcal{R}$, if $P \xrightarrow{\alpha} P'$ then
    - i) if α is not an input,
      there exists Q' such that $Q \xrightarrow{\alpha} Q'$ and $P' \mathcal{R}\, Q'$
    - ii) if $\alpha = a(x)$,
      then for all $u \in \mathcal{N}$, there exists Q' such that $Q \xrightarrow{\alpha} Q'$ and $(P'\{^u/_x\}, Q'\{^u/_x\}) \in \mathcal{R}$
  - a *late* bisimulation if instead of ii), it satisfies

Early and late bisimulation

- The *symmetric* relation $\mathcal{R} \subset \mathcal{P} \times \mathcal{P}$ is
  - an *early* bisimulation if for all $(P, Q) \in \mathcal{R}$, if $P \xrightarrow{\alpha} P'$ then
    - i) if α is not an input, there exists Q' such that $Q \xrightarrow{\alpha} Q'$ and $P' \mathcal{R}\, Q'$
    - ii) if $\alpha = a(x)$, then for all $u \in \mathcal{N}$, there exists Q' such that $Q \xrightarrow{\alpha} Q'$ and $(P'\{^u/_x\}, Q'\{^u/_x\}) \in \mathcal{R}$
  - a *late* bisimulation if instead of ii), it satisfies
    - ii') if $\alpha = a(x)$, then there exists Q' such that $Q \xrightarrow{\alpha} Q'$ and for all $u \in \mathcal{N}$, $(P'\{^u/_x\}, Q'\{^u/_x\}) \in \mathcal{R}$

From late to open

- Late bisimulation

From late to open

- Late bisimulation

$$\begin{array}{c} P \\ | \\ Q \end{array}$$

From late to open

- Late bisimulation

$$
\begin{array}{ccc}
P & \xrightarrow{a(x)} & P' \\
| & & \\
Q & &
\end{array}
$$

From late to open

- Late bisimulation

$$\begin{array}{ccc} P & \xrightarrow{a(x)} & P' \\ | & & \\ Q & \xrightarrow{a(x)} & \end{array}$$

From late to open

- Late bisimulation

$$\begin{array}{ccc} P & \xrightarrow{a(x)} & P' \\ | & & \\ Q & \xrightarrow{a(x)} & Q' \end{array}$$

From late to open

- Late bisimulation

$$\begin{array}{ccccc} P & \xrightarrow{a(x)} & P' & & P'\{^z/_x\} \\ | & & & & | \\ Q & \xrightarrow{a(x)} & Q' & & Q'\{^z/_x\} \end{array}$$

From late to open

- Late bisimulation

$$\begin{array}{ccccc} P & \xrightarrow{a(x)} & P' & & P'\{^z/_x\} \\ | & & & & | \\ Q & \xrightarrow{a(x)} & Q' & & Q'\{^z/_x\} \end{array}$$

- Open bisimulation

From late to open

- Late bisimulation

$$\begin{array}{ccccc} P & \xrightarrow{a(x)} & P' & & P'\{^z/_x\} \\ | & & & & | \\ Q & \xrightarrow{a(x)} & Q' & & Q'\{^z/_x\} \end{array}$$

- Open bisimulation

$$\begin{array}{c} P \\ | \\ Q \end{array}$$

From late to open

- Late bisimulation

$$\begin{array}{cccc} P & \xrightarrow{a(x)} & P' & P'\{^z/_x\} \\ | & & & | \\ Q & \xrightarrow{a(x)} & Q' & Q'\{^z/_x\} \end{array}$$

- Open bisimulation

$$\begin{array}{cc} P & P\sigma \\ | & \\ Q & \end{array}$$

From late to open

- Late bisimulation

$$\begin{array}{ccccc} P & \xrightarrow{a(x)} & P' & & P'\{^z/_x\} \\ | & & & & | \\ Q & \xrightarrow{a(x)} & Q' & & Q'\{^z/_x\} \end{array}$$

- Open bisimulation

$$\begin{array}{ccccc} P & & P\sigma & \xrightarrow{\alpha} & P' \\ | & & & & \\ Q & & & & \end{array}$$

From late to open

- Late bisimulation

$$\begin{array}{ccccc} P & \xrightarrow{a(x)} & P' & & P'\{z/x\} \\ | & & & & | \\ Q & \xrightarrow{a(x)} & Q' & & Q'\{z/x\} \end{array}$$

- Open bisimulation

$$\begin{array}{ccccc} P & & P\sigma & \xrightarrow{\alpha} & P' \\ | & & & & \\ Q & & Q\sigma & & \end{array}$$

From late to open

- Late bisimulation

$$
\begin{array}{ccccc}
P & \xrightarrow{a(x)} & P' & & P'\{^z/_x\} \\
| & & & & | \\
Q & \xrightarrow{a(x)} & Q' & & Q'\{^z/_x\}
\end{array}
$$

- Open bisimulation

$$
\begin{array}{ccccc}
P & & P\sigma & \xrightarrow{\alpha} & P' \\
| & & & & \\
Q & & Q\sigma & \xrightarrow{\alpha} & Q'
\end{array}
$$

From late to open

- Late bisimulation

$$\begin{array}{ccccc} P & \xrightarrow{a(x)} & P' & & P'\{{}^{z}\!/\!_{x}\} \\ | & & & & | \\ Q & \xrightarrow{a(x)} & Q' & & Q'\{{}^{z}\!/\!_{x}\} \end{array}$$

- Open bisimulation

$$\begin{array}{ccccc} P & & P\sigma & \xrightarrow{\alpha} & P' \\ | & & & & | \\ Q & & Q\sigma & \xrightarrow{\alpha} & Q' \end{array}$$

From late to open

- Late bisimulation

$$\begin{array}{ccccc} P & \xrightarrow{a(x)} & P' & & P'\{^z/_x\} \\ | & & & & | \\ Q & \xrightarrow{a(x)} & Q' & & Q'\{^z/_x\} \end{array}$$

- Open bisimulation

$$\begin{array}{cccccc} & P & & P\sigma & \xrightarrow{\alpha} & P' \\ \sigma \triangleright D & | & & & & | \\ & Q & & Q\sigma & \xrightarrow{\alpha} & Q' \end{array}$$

Indexed by a *distinction* D.

Distinctions

- A distinction D is an irreflexive and symmetric relation between names (finite list of inequalities)

Distinctions

- A distinction D is an irreflexive and symmetric relation between names (finite list of inequalities)
- A substitution σ respects D ($\sigma \triangleright D$) if $x\sigma \neq y\sigma$ for all $(x, y) \in D$.

Distinctions

- A distinction D is an irreflexive and symmetric relation between names (finite list of inequalities)
- A substitution σ respects D ($\sigma \triangleright D$) if $x\sigma \neq y\sigma$ for all $(x, y) \in D$.
- If $\sigma \triangleright D$, we define the updated distinction $D\sigma$.

Distinctions

- A distinction D is an irreflexive and symmetric relation between names (finite list of inequalities)
- A substitution σ respects D ($\sigma \triangleright D$) if $x\sigma \neq y\sigma$ for all $(x, y) \in D$.
- If $\sigma \triangleright D$, we define the updated distinction $D\sigma$.
- For example, if $D = \{(x, y), (x, z), (y, x), (z, x)\}$ then
  - $x \mapsto u$ respects D and

Distinctions

- A distinction D is an irreflexive and symmetric relation between names (finite list of inequalities)
- A substitution σ respects D ($\sigma \triangleright D$) if $x\sigma \neq y\sigma$ for all $(x, y) \in D$.
- If $\sigma \triangleright D$, we define the updated distinction $D\sigma$.
- For example, if $D = \{(x, y), (x, z), (y, x), (z, x)\}$ then
    - $x \mapsto u$ respects D and the updated distinction is $\{(u, y), (u, z), (y, u), (z, u)\}$

Distinctions

- A distinction D is an irreflexive and symmetric relation between names (finite list of inequalities)
- A substitution σ respects D ($\sigma \triangleright D$) if $x\sigma \neq y\sigma$ for all $(x, y) \in D$.
- If $\sigma \triangleright D$, we define the updated distinction $D\sigma$.
- For example, if $D = \{(x, y), (x, z), (y, x), (z, x)\}$ then
  - $x \mapsto u$ respects D and the updated distinction is $\{(u, y), (u, z), (y, u), (z, u)\}$
  - On the contrary, $x \mapsto u, y \mapsto u$ does not respect D

Open Bisimulation

- An open bisimulation is a "symmetric" relation $\mathcal{R} \subset \mathcal{D} \times \mathcal{P} \times \mathcal{P}$ such that

Open Bisimulation

- An open bisimulation is a "symmetric" relation $\mathcal{R} \subset \mathcal{D} \times \mathcal{P} \times \mathcal{P}$ such that for all $(D, P, Q) \in \mathcal{R}$ and $\sigma \triangleright D$, if $P\sigma \xrightarrow{\alpha} P'$ then

Open Bisimulation

- An open bisimulation is a "symmetric" relation $\mathcal{R} \subset \mathcal{D} \times \mathcal{P} \times \mathcal{P}$ such that for all $(D, P, Q) \in \mathcal{R}$ and $\sigma \triangleright D$, if $P\sigma \xrightarrow{\alpha} P'$ then $Q\sigma \xrightarrow{\alpha} Q'$ and

Open Bisimulation

- An open bisimulation is a "symmetric" relation $\mathcal{R} \subset \mathcal{D} \times \mathcal{P} \times \mathcal{P}$ such that for all $(D, P, Q) \in \mathcal{R}$ and $\sigma \triangleright D$, if $P\sigma \xrightarrow{\alpha} P'$ then $Q\sigma \xrightarrow{\alpha} Q'$ and
  - if α is not a bound output, then $(D\sigma, P', Q') \in \mathcal{R}$

Open Bisimulation

- An open bisimulation is a "symmetric" relation $\mathcal{R} \subset \mathcal{D} \times \mathcal{P} \times \mathcal{P}$ such that for all $(D, P, Q) \in \mathcal{R}$ and $\sigma \triangleright D$, if $P\sigma \xrightarrow{\alpha} P'$ then $Q\sigma \xrightarrow{\alpha} Q'$ and
  - if α is not a bound output, then $(D\sigma, P', Q') \in \mathcal{R}$
  - otherwise, if $\alpha = (\nu z)\,\overline{a}\,z$, then

Open Bisimulation

- An open bisimulation is a "symmetric" relation $\mathcal{R} \subset \mathcal{D} \times \mathcal{P} \times \mathcal{P}$ such that for all $(D, P, Q) \in \mathcal{R}$ and $\sigma \triangleright D$, if $P\sigma \xrightarrow{\alpha} P'$ then $Q\sigma \xrightarrow{\alpha} Q'$ and
  - if α is not a bound output, then $(D\sigma, P', Q') \in \mathcal{R}$
  - otherwise, if $\alpha = (\nu z)\,\overline{a}\,z$, then $(D', P', Q') \in \mathcal{R}$ where $D' = D\sigma$

Open Bisimulation

- An open bisimulation is a "symmetric" relation $\mathcal{R} \subset \mathcal{D} \times \mathcal{P} \times \mathcal{P}$ such that for all $(D, P, Q) \in \mathcal{R}$ and $\sigma \triangleright D$, if $P\sigma \xrightarrow{\alpha} P'$ then $Q\sigma \xrightarrow{\alpha} Q'$ and
  - if α is not a bound output, then $(D\sigma, P', Q') \in \mathcal{R}$
  - otherwise, if $\alpha = (\nu z)\,\overline{a}\,z$, then $(D', P', Q') \in \mathcal{R}$ where $D' = D\sigma \cup \{z\} \otimes (\mathrm{fn}((P + Q)\sigma))$

Open Bisimulation

- An open bisimulation is a "symmetric" relation $\mathcal{R} \subset \mathcal{D} \times \mathcal{P} \times \mathcal{P}$ such that for all $(D, P, Q) \in \mathcal{R}$ and $\sigma \triangleright D$, if $P\sigma \xrightarrow{\alpha} P'$ then $Q\sigma \xrightarrow{\alpha} Q'$ and
  - if α is not a bound output, then $(D\sigma, P', Q') \in \mathcal{R}$
  - otherwise, if $\alpha = (\nu z)\,\overline{a}\,z$, then $(D', P', Q') \in \mathcal{R}$ where $D' = D\sigma \cup \{z\} \otimes (\mathrm{fn}((P+Q)\sigma))$
- Distinctions are used to forbid the fusing of fresh names with other names

The lazy flavour of open

$$P \stackrel{\text{def}}{=} c(x).(\tau + \tau.\tau + \tau.[x = a]\tau)$$
$$Q \stackrel{\text{def}}{=} c(x).(\tau + \tau.\tau)$$

The lazy flavour of open

$$
\begin{aligned}
P &\stackrel{\text{def}}{=} c(x).(\tau + \tau.\tau + \tau.[x = a]\tau) \\
Q &\stackrel{\text{def}}{=} c(x).(\tau + \tau.\tau)
\end{aligned}
$$

- P and Q are late bisimilar but not open

The lazy flavour of open

$$\begin{array}{rcl} P & \stackrel{\text{def}}{=} & c(x).(\tau + \tau.\tau + \tau.[x = a]\tau) \\ Q & \stackrel{\text{def}}{=} & c(x).(\tau + \tau.\tau) \end{array}$$

- P and Q are late bisimilar but not open
- In open, the instantiation of x can be delayed until x is used

Some properties of open

Some properties of open

- Contrary to early or late, it is a full congruence

Some properties of open

- Contrary to early or late, it is a full congruence
- More precisely, open D-bisimilarity is a D-congruence

Some properties of open

- Contrary to early or late, it is a full congruence
- More precisely, open D-bisimilarity is a D-congruence
- It is easily implementable (Mobility Workbench, ABC)

Some properties of open

- Contrary to early or late, it is a full congruence
- More precisely, open D-bisimilarity is a D-congruence
- It is easily implementable (Mobility Workbench, ABC)

For these reasons, we wanted to extend open to the spi-calculus.

Outline

The spi-calculus

The spi-calculus

- To model and study cryptographic protocols.

The spi-calculus

- To model and study cryptographic protocols.
- Messages

$$M, N \quad ::= \quad x \quad \mid \quad (M\,.\,N) \quad \mid \quad \mathrm{E}_N(M)$$

The spi-calculus

- To model and study cryptographic protocols.
- Messages

$$M, N \quad ::= \quad x \quad \mid \quad (M \,.\, N) \quad \mid \quad \mathsf{E}_N(M)$$

- Expressions

$$\begin{array}{rcllll} E, F & ::= & x & & & \\ & \mid & (E \,.\, F) & \mid \ \pi_1(E) & \mid \ \pi_2(E) \\ & \mid & \mathsf{E}_F(E) & \mid \ \mathsf{D}_F(E) & \end{array}$$

The spi-calculus

- To model and study cryptographic protocols.
- Messages

$$M, N \quad ::= \quad x \quad \mid \quad (M\,.\,N) \quad \mid \quad \mathrm{E}_N(M)$$

- Expressions

$$\begin{array}{rcllll} E, F & ::= & x & & & \\ & \mid & (E\,.\,F) & \mid \ \pi_1(E) & \mid & \pi_2(E) \\ & \mid & \mathrm{E}_F(E) & \mid \ \mathrm{D}_F(E) & & \end{array}$$

- Guards

$$\phi \quad ::= \quad [E = F] \quad \mid \quad [E : \mathcal{N}]$$

Evaluation of expressions and formulae

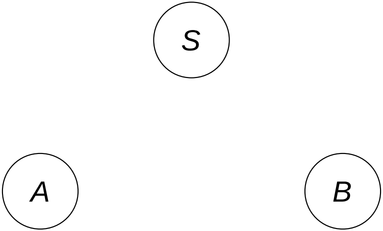
$$
\begin{aligned}
[\![a]\!] &\stackrel{\text{def}}{=} a \\
[\![\mathsf{E}_F(E)]\!] &\stackrel{\text{def}}{=} \mathsf{E}_N(M) && \text{if } [\![E]\!] = M \in \mathcal{M} \text{ and } [\![F]\!] = N \in \mathcal{M} \\
[\![\mathsf{D}_F(E)]\!] &\stackrel{\text{def}}{=} M && \text{if } [\![E]\!] = \mathsf{E}_N(M) \in \mathcal{M} \text{ and } [\![F]\!] = N \in \mathcal{M} \\
[\![E]\!] &\stackrel{\text{def}}{=} \bot && \text{in all other cases}
\end{aligned}
$$

Evaluation of expressions and formulae

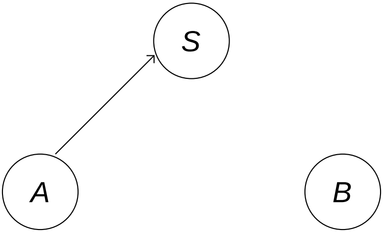
$$
\begin{array}{rcll}
[\![a]\!] & \stackrel{\text{def}}{=} & a & \\
[\![\mathsf{E}_F(E)]\!] & \stackrel{\text{def}}{=} & \mathsf{E}_N(M) & \text{if } [\![E]\!] = M \in \mathcal{M} \text{ and } [\![F]\!] = N \in \mathcal{M} \\
[\![\mathsf{D}_F(E)]\!] & \stackrel{\text{def}}{=} & M & \text{if } [\![E]\!] = \mathsf{E}_N(M) \in \mathcal{M} \text{ and } [\![F]\!] = N \in \mathcal{M} \\
[\![E]\!] & \stackrel{\text{def}}{=} & \perp & \text{in all other cases}
\end{array}
$$

$$
\begin{array}{rcll}
[\![tt]\!] & \stackrel{\text{def}}{=} & \textbf{true} & \\
[\![\phi \wedge \psi]\!] & \stackrel{\text{def}}{=} & [\![\phi]\!] \text{ and } [\![\psi]\!] & \\
[\![[E = F]]\!] & \stackrel{\text{def}}{=} & \textbf{true} & \text{if } [\![E]\!] = [\![F]\!] = M \in \mathcal{M} \\
[\![[E : \mathcal{N}]]\!] & \stackrel{\text{def}}{=} & \textbf{true} & \text{if } [\![E]\!] = a \in \mathcal{N} \\
[\![\phi]\!] & \stackrel{\text{def}}{=} & \textbf{false} & \text{in all other cases}
\end{array}
$$

The wide-mouthed frog protocol

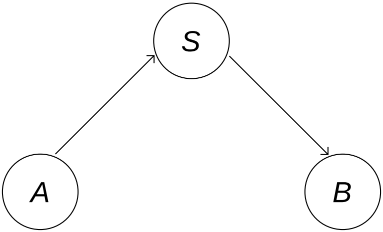


The wide-mouthed frog protocol



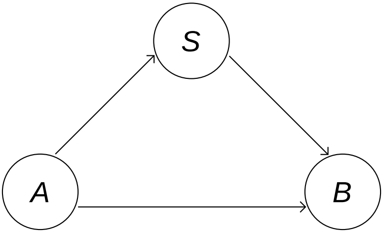
1. $A \rightarrow S : (A \,.\, \mathrm{E}_{k_{AS}}((B \,.\, k_{AB})))$

The wide-mouthed frog protocol



1. $A \rightarrow S : (A \,.\, \mathrm{E}_{k_{AS}}((B \,.\, k_{AB})))$
2. $S \rightarrow B : \mathrm{E}_{k_{BS}}(((A \,.\, B) \,.\, k_{AB}))$

The wide-mouthed frog protocol



1. $A \rightarrow S : (A \,.\, \mathrm{E}_{k_{AS}}((B \,.\, k_{AB})))$
2. $S \rightarrow B : \mathrm{E}_{k_{BS}}(((A \,.\, B) \,.\, k_{AB}))$
3. $A \rightarrow B : \mathrm{E}_{k_{AB}}(m)$

.. in spi-calculus

$$
\begin{array}{l}
(\nu k_{AS}, k_{BS}) \\
\quad (\nu k_{AB})\,\overline{S}\langle(A\,.\,\mathrm{E}_{k_{AS}}((B\,.\,k_{AB})))\rangle.\overline{B}\langle\mathrm{E}_{k_{AB}}(m)\rangle.\,\mathbf{0} \\
\quad \| B(x_1).\phi_1 B(x_2).\phi_2\,\mathbf{0} \\
\quad \| S(x_0).\phi_0 \overline{B}\langle\mathrm{E}_{k_{BS}}(((A\,.\,B)\,.\,\pi_2(\mathrm{D}_{k_{AS}}(\pi_2(x_0)))))\rangle.\,\mathbf{0}
\end{array}
$$

.. in spi-calculus

$$
\begin{array}{l}
(\nu k_{AS}, k_{BS}) \\
\quad (\nu k_{AB})\, \overline{S}\langle (A\,.\,\mathrm{E}_{k_{AS}}((B\,.\,k_{AB})))\rangle.\overline{B}\langle \mathrm{E}_{k_{AB}}(m)\rangle.\,\mathbf{0} \\
\quad \| B(x_1).\phi_1 B(x_2).\phi_2\, \mathbf{0} \\
\quad \| S(x_0).\phi_0 \overline{B}\langle \mathrm{E}_{k_{BS}}(((A\,.\,B)\,.\,\pi_2(\mathrm{D}_{k_{AS}}(\pi_2(x_0))))))\rangle.\,\mathbf{0}
\end{array}
$$

$$
\begin{array}{lcl}
\phi_0 & = & [B = \pi_1(\mathrm{D}_{k_{AS}}(\pi_2(x_0)))] \wedge [A = \pi_1(x_0)] \\
\phi_1 & = & [B = \pi_1(\pi_2(\mathrm{D}_{k_{BS}}(x_1)))] \wedge [A = \pi_1(\mathrm{D}_{k_{BS}}(x_1))] \\
\phi_2 & = & [\mathrm{D}_{\pi_2(\pi_2(\mathrm{D}_{k_{BS}}(x_1)))}(x_2) : \mathcal{M}]
\end{array}
$$

Open in spi?

- Consider

$$P \stackrel{\text{def}}{=} (\nu k)\,(\nu m)\,\overline{a}\langle \mathrm{E}_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \parallel [x = k]\overline{a}\langle a\rangle)$$

Open in spi?

- Consider

$$P \stackrel{\text{def}}{=} (\nu k)\,(\nu m)\,\overline{a}\langle \mathrm{E}_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \parallel [x = k\,]\overline{a}\langle a\rangle)$$

- The guard $[x = k\,]$ can never be true.

Open in spi?

- Consider

$$P \stackrel{\text{def}}{=} (\nu k)\,(\nu m)\,\overline{a}\langle \mathrm{E}_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \parallel [x = k]\overline{a}\langle a\rangle)$$

- The guard $[x = k]$ can never be true.
- The name k has been extruded when performing $\overline{a}\,\mathrm{E}_k(m)$.

Open in spi?

- Consider

$$P \stackrel{\text{def}}{=} (\nu k)\,(\nu m)\,\overline{a}\langle \mathrm{E}_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \,\|\, [x = k]\overline{a}\langle a\rangle)$$

- The guard $[x = k]$ can never be true.
- The name k has been extruded when performing $\overline{a}\,\mathrm{E}_k(m)$.
- What are the possible values for x?

Open in spi?

- Consider

$$P \stackrel{\text{def}}{=} (\nu k)\,(\nu m)\,\overline{a}\langle \mathrm{E}_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \parallel [x = k]\overline{a}\langle a\rangle)$$

- The guard $[x = k]$ can never be true.
- The name k has been extruded when performing $\overline{a}\,\mathrm{E}_k(m)$.
- What are the possible values for x?
  a

Open in spi?

- Consider

$$P \stackrel{\text{def}}{=} (\nu k)\,(\nu m)\,\overline{a}\langle \mathrm{E}_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \,\|\, [x = k]\overline{a}\langle a\rangle)$$

- The guard $[x = k]$ can never be true.
- The name k has been extruded when performing $\overline{a}\,\mathrm{E}_k(m)$.
- What are the possible values for x?
  a, z for any z fresh

Open in spi?

- Consider

$$P \stackrel{\text{def}}{=} (\nu k)\,(\nu m)\,\overline{a}\langle \mathrm{E}_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \parallel [x = k]\overline{a}\langle a\rangle)$$

- The guard $[x = k]$ can never be true.
- The name k has been extruded when performing $\overline{a}\,\mathrm{E}_k(m)$.
- What are the possible values for x?
  a, z for any z fresh(not in $\{k, m, a\}$)

Open in spi?

- Consider

$$P \stackrel{\text{def}}{=} (\nu k)\,(\nu m)\,\overline{a}\langle \mathrm{E}_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \,\|\, [x = k]\overline{a}\langle a\rangle)$$

- The guard $[x = k]$ can never be true.
- The name k has been extruded when performing $\overline{a}\,\mathrm{E}_k(m)$.
- What are the possible values for x?
  a, z for any z fresh(not in $\{k, m, a\}$), $\mathrm{E}_k(m)$

Open in spi?

- Consider

$$P \stackrel{\text{def}}{=} (\nu k)\,(\nu m)\,\overline{a}\langle \mathrm{E}_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \,\|\, [x = k]\overline{a}\langle a\rangle)$$

- The guard $[x = k]$ can never be true.
- The name k has been extruded when performing $\overline{a}\,\mathrm{E}_k(m)$.
- What are the possible values for x?
  a, z for any z fresh(not in $\{k, m, a\}$), $\mathrm{E}_k(m)$
  and any message built with these "bricks"

Bisimulations in spi

- Bisimulations of π-calculus are two strong

Bisimulations in spi

- Bisimulations of π-calculus are two strong

$$P(m) \stackrel{\text{def}}{=} (\nu k)\,\overline{a}\langle \mathrm{E}_k(m)\rangle$$

Bisimulations in spi

- Bisimulations of π-calculus are two strong

$$P(m) \stackrel{\text{def}}{=} (\nu k)\,\overline{a}\langle \mathrm{E}_k(m)\rangle$$

For any m and n, we want $P(m)$ and $P(n)$ equivalent.

Bisimulations in spi

- Bisimulations of π-calculus are two strong

$$P(m) \stackrel{\text{def}}{=} (\nu k)\,\overline{a}\langle \mathsf{E}_k(m)\rangle$$

  For any m and n, we want $P(m)$ and $P(n)$ equivalent.
- Abadi and Gordon have introduced environment-sensitive bisimulation.

Outline

Different kinds of free names

$$P \stackrel{\text{def}}{=} a(x).(\nu k)\,\overline{b}\langle k\rangle.\overline{x}\langle k\rangle.\mathbf{0}$$

A free name is

Different kinds of free names

$$P \stackrel{\text{def}}{=} a(x).(\nu k)\,\overline{b}\langle k\rangle.\overline{x}\langle k\rangle.\,\mathbf{0}$$

A free name is

- either initially free

Different kinds of free names

$$P \stackrel{\text{def}}{=} a(x).(\nu k)\,\overline{b}\langle k\rangle.\overline{x}\langle k\rangle.\,\mathbf{0}$$

A free name is

- either initially free
- or becomes free after an input

Different kinds of free names

$$P \stackrel{\text{def}}{=} a(x).(\nu k)\,\overline{b}\langle k\rangle.\overline{x}\langle k\rangle.\,\mathbf{0}$$

A free name is

- either initially free
- or becomes free after an input
- or becomes free by scope extrusion

Different kinds of free names

$$P \stackrel{\text{def}}{=} a(x).(\nu k)\,\overline{b}\langle k\rangle.\overline{x}\langle k\rangle.\,\mathbf{0}$$

A free name is

- either initially free
- or becomes free after an input
- or becomes free by scope extrusion

- The first two kinds are substitutable:

Different kinds of free names

$$P \stackrel{\text{def}}{=} a(x).(\nu k)\,\overline{b}\langle k\rangle.\overline{x}\langle k\rangle.\,\mathbf{0}$$

A free name is

- either initially free
- or becomes free after an input
- or becomes free by scope extrusion

- The first two kinds are substitutable:
  - by any name that was known at the moment they became free or

Different kinds of free names

$$P \stackrel{\text{def}}{=} a(x).(\nu k)\,\overline{b}\langle k\rangle.\overline{x}\langle k\rangle.\,\mathbf{0}$$

A free name is

- either initially free
- or becomes free after an input
- or becomes free by scope extrusion

- The first two kinds are substitutable:
  - by any name that was known at the moment they became free or
  - any fresh name.

Refining distinctions

- A distinction is a finite list of *in*equalities between names.

Refining distinctions

- A distinction is a finite list of *in*equalities between names.
- We take a dual approach for constraining admissible substitutions.

Refining distinctions

- A distinction is a finite list of *in*equalities between names.
- We take a dual approach for constraining admissible substitutions.
- $e = (C, V, \prec)$

Refining distinctions

- A distinction is a finite list of *in*equalities between names.
- We take a dual approach for constraining admissible substitutions.
- $e = (C, V, \prec)$
  - C contains the emitted names (or messages) not in V

Refining distinctions

- A distinction is a finite list of *in*equalities between names.
- We take a dual approach for constraining admissible substitutions.
- $e = (C, V, \prec)$
  - C contains the emitted names (or messages) not in V
  - V contains the input names and the initially free ones

Refining distinctions

- A distinction is a finite list of *in*equalities between names.
- We take a dual approach for constraining admissible substitutions.
- $e = (C, V, \prec)$
  - C contains the emitted names (or messages) not in V
  - V contains the input names and the initially free ones
  - $\prec$ indicates for each $x \in V$ which names in C were known before

Environments

$$P \stackrel{\text{def}}{=} (\nu k)\,\overline{a}\langle k\rangle.a(x).((\nu l)\,\overline{b}\langle l\rangle \| [x = k]\overline{a}\langle a\rangle)$$

C	V	$\prec$
$\emptyset$	$\{a, b\}$	$\emptyset$

$$D = \emptyset$$

Environments

$$P \stackrel{\text{def}}{=} (\nu k)\,\overline{a}\langle k\rangle.a(x).((\nu l)\,\overline{b}\langle l\rangle \| [x = k]\overline{a}\langle a\rangle)$$

C	V	$\prec$
$\emptyset$	$\{a, b\}$	$\emptyset$
$\{k\}$	$\{a, b\}$	$\emptyset$

$$D = k \neq a, k \neq b$$

Environments

$$P \stackrel{\text{def}}{=} (\nu k)\,\overline{a}\langle k\rangle.a(x).((\nu l)\,\overline{b}\langle l\rangle \parallel [x = k]\overline{a}\langle a\rangle)$$

C	V	$\prec$
$\emptyset$	$\{a, b\}$	$\emptyset$
$\{k\}$	$\{a, b\}$	$\emptyset$
$\{k\}$	$\{a, b, x\}$	$\{(k, x)\}$

$$D = k \neq a, k \neq b$$

Environments

$$P \stackrel{\text{def}}{=} (\nu k)\,\overline{a}\langle k\rangle.a(x).((\nu l)\,\overline{b}\langle l\rangle \| [x = k]\overline{a}\langle a\rangle)$$

C	V	$\prec$
$\emptyset$	$\{a, b\}$	$\emptyset$
$\{k\}$	$\{a, b\}$	$\emptyset$
$\{k\}$	$\{a, b, x\}$	$\{(k, x)\}$

$$D = k \neq a, k \neq b$$

Environments

$$P \stackrel{\text{def}}{=} (\nu k)\,\overline{a}\langle k\rangle.a(x).((\nu l)\,\overline{b}\langle l\rangle \,\|\, [x = l]\overline{a}\langle a\rangle)$$

C	V	$\prec$
$\emptyset$	$\{a, b\}$	$\emptyset$
$\{k\}$	$\{a, b\}$	$\emptyset$
$\{k\}$	$\{a, b, x\}$	$\{(k, x)\}$

$$D = k \neq a, k \neq b$$

Environments

$$P \stackrel{\text{def}}{=} (\nu k)\,\overline{a}\langle k\rangle.a(x).((\nu l)\,\overline{b}\langle l\rangle \parallel [x=k]\overline{a}\langle a\rangle)$$

C	V	$\prec$
$\emptyset$	$\{a,b\}$	$\emptyset$
$\{k\}$	$\{a,b\}$	$\emptyset$
$\{k\}$	$\{a,b,x\}$	$\{(k,x)\}$
$\{k,l\}$	$\{a,b,x\}$	$\{(k,x)\}$

$$D = k \neq a, k \neq b, l \neq a, l \neq b, l \neq x, k \neq l$$

Environments

$$P \stackrel{\text{def}}{=} (\nu k)\,\overline{a}\langle k\rangle.a(x).((\nu l)\,\overline{b}\langle l\rangle \| [x = k]\overline{a}\langle a\rangle)$$

C	V	$\prec$
$\emptyset$	$\{a, b\}$	$\emptyset$
$\{k\}$	$\{a, b\}$	$\emptyset$
$\{k\}$	$\{a, b, x\}$	$\{(k, x)\}$
$\{k, l\}$	$\{a, b, x\}$	$\{(k, x)\}$

$$D = k \neq a, k \neq b, l \neq a, l \neq b, l \neq x, k \neq l$$

Environments

$$P \stackrel{\text{def}}{=} (\nu k)\,\overline{a}\langle k\rangle.a(x).((\nu l)\,\overline{b}\langle l\rangle \parallel [x = l]\overline{a}\langle a\rangle)$$

C	V	$\prec$
$\emptyset$	$\{a, b\}$	$\emptyset$
$\{k\}$	$\{a, b\}$	$\emptyset$
$\{k\}$	$\{a, b, x\}$	$\{(k, x)\}$
$\{k, l\}$	$\{a, b, x\}$	$\{(k, x)\}$

$$D = k \neq a, k \neq b, l \neq a, l \neq b, l \neq x, k \neq l$$

Environments

$$P \stackrel{\text{def}}{=} (\nu k)\,\overline{a}\langle k\rangle.a(x).((\nu l)\,\overline{b}\langle l\rangle \,\|\, [x = k]\overline{a}\langle a\rangle)$$

C	V	$\prec$
$\emptyset$	$\{a, b\}$	$\emptyset$
$\{k\}$	$\{a, b\}$	$\emptyset$
$\{k\}$	$\{a, b, x\}$	$\{(k, x)\}$
$\{k, l\}$	$\{a, b, x\}$	$\{(k, x)\}$

$$D = k \neq a, k \neq b, l \neq a, l \neq b, l \neq x, k \neq l$$

Refining distinctions

- A distinction is a finite list of *in*equalities between names.
- We take a dual approach for constraining admissible substitutions.
- $e = (C, V, \prec)$
  - C contains the emitted names (or messages) not in V
  - V contains the input names and the initially free ones
  - $\prec$ indicates for each $x \in V$ which names in C were known before
- A substitution σ respects e if
  $\mathrm{supp}(\sigma) \subseteq V$ and σ does not "contradict" $\prec$

Refining distinctions

- A distinction is a finite list of *in*equalities between names.
- We take a dual approach for constraining admissible substitutions.
- $e = (C, V, \prec)$
  - C contains the emitted names (or messages) not in V
  - V contains the input names and the initially free ones
  - $\prec$ indicates for each $x \in V$ which names in C were known before
- A substitution σ respects e if
  $\mathrm{supp}(\sigma) \subseteq V$ and σ does not "contradict" $\prec$
- The corresponding distinction is

$$D(C, V, \prec) \stackrel{\text{def}}{=} C^{\neq} \cup \{n \neq x \mid n \in C \wedge \neg(n \prec x)\}$$

Some results

We have

Some results

We have

- $$P \sim_{\mathrm{K}}^{(C,V,\prec)} Q \Rightarrow P \sim_{\mathrm{O}}^{D(C,V,\prec)} Q$$

Some results

We have

- $$P \sim_{\mathrm{K}}^{(C,V,\prec)} Q \Rightarrow P \sim_{\mathrm{O}}^{D(C,V,\prec)} Q$$

- $$P \sim_{\mathrm{O}}^{D(C,V,\prec)} Q \Rightarrow P \sim_{\mathrm{K}}^{(C,V,\prec)} Q$$

Some results

We have

- $$P \sim_{\mathrm{K}}^{(C,V,\prec)} Q \Rightarrow P \sim_{\mathrm{O}}^{D(C,V,\prec)} Q$$
- $$P \sim_{\mathrm{O}}^{D(C,V,\prec)} Q \Rightarrow P \sim_{\mathrm{K}}^{(C,V,\prec)} Q$$
- In particular
$$P \sim_{\mathrm{O}}^{\emptyset} Q \Leftrightarrow P \sim_{\mathrm{K}}^{(\emptyset,\mathrm{fn}(P+Q),\emptyset)} Q$$

Some results

We have

- $$P \sim_{\mathrm{K}}^{(C,V,\prec)} Q \Rightarrow P \sim_{\mathrm{O}}^{D(C,V,\prec)} Q$$

- $$P \sim_{\mathrm{O}}^{D(C,V,\prec)} Q \Rightarrow P \sim_{\mathrm{K}}^{(C,V,\prec)} Q$$

- In particular
$$P \sim_{\mathrm{O}}^{\emptyset} Q \Leftrightarrow P \sim_{\mathrm{K}}^{(\emptyset,\mathrm{fn}(P+Q),\emptyset)} Q$$

- if e is an environment, then open $D(e)$-bisimilarity is an e-congruence

Outline

The intruder knowledge (1/2)

- A hedge h is a finite set of pairs of message

The intruder knowledge (1/2)

- A hedge h is a finite set of pairs of message
- The synthesis $\mathcal{S}(h)$ is the smallest set that contains h and satisfies

$$(\text{SYN-ENC})\ \frac{(M,N) \in \mathcal{S}(h) \qquad (K,L) \in \mathcal{S}(h)}{(\mathrm{E}_K(M), \mathrm{E}_L(N)) \in \mathcal{S}(h)}$$

The intruder knowledge (1/2)

- A hedge h is a finite set of pairs of message
- The synthesis $\mathcal{S}(h)$ is the smallest set that contains h and satisfies

$$\text{(SYN-ENC)}\ \frac{(M,N) \in \mathcal{S}(h) \qquad (K,L) \in \mathcal{S}(h)}{(\mathrm{E}_K(M), \mathrm{E}_L(N)) \in \mathcal{S}(h)}$$

- For example, if $h = \{(a,a),(k,k)\}$, we have $(\mathrm{E}_k(a), \mathrm{E}_k(a)) \in \mathcal{S}(h)$

The intruder knowledge (1/2)

- A hedge h is a finite set of pairs of message
- The synthesis $\mathcal{S}(h)$ is the smallest set that contains h and satisfies

$$(\textsc{syn-enc})\ \frac{(M,N) \in \mathcal{S}(h) \qquad (K,L) \in \mathcal{S}(h)}{(\mathrm{E}_K(M), \mathrm{E}_L(N)) \in \mathcal{S}(h)}$$

- For example, if $h = \{(a,a),(k,k)\}$, we have $(\mathrm{E}_k(a), \mathrm{E}_k(a)) \in \mathcal{S}(h)$
- In general, $\mathcal{S}(h)$ is not a hedge since it is not finite.

The intruder knowledge (2/2)

- The analysis $\mathcal{A}(h)$ is the smallest hedge that contains h and satisfies

$$(\text{ANA-DEC})\ \frac{(\mathsf{E}_K(M), \mathsf{E}_L(N)) \in \mathcal{A}(h) \qquad (K, L) \in \mathcal{S}(\mathcal{A}(h))}{(M, N) \in \mathcal{A}(h)}$$

The intruder knowledge (2/2)

- The analysis $\mathcal{A}(h)$ is the smallest hedge that contains h and satisfies

$$\text{(ANA-DEC)}\ \frac{(\mathrm{E}_K(M), \mathrm{E}_L(N)) \in \mathcal{A}(h) \qquad (K, L) \in \mathcal{S}(\mathcal{A}(h))}{(M, N) \in \mathcal{A}(h)}$$

- For example, if $h = \{(k, k), (\mathrm{E}_k(a), \mathrm{E}_k(a))\}$, we have $\mathcal{A}(h) = \{(k, k), (\mathrm{E}_k(a), \mathrm{E}_k(a)), (a, a)\}$.

The intruder knowledge (2/2)

- The analysis $\mathcal{A}(h)$ is the smallest hedge that contains h and satisfies

$$(\text{ANA-DEC})\ \frac{(\mathrm{E}_K(M), \mathrm{E}_L(N)) \in \mathcal{A}(h) \qquad (K, L) \in \mathcal{S}(\mathcal{A}(h))}{(M, N) \in \mathcal{A}(h)}$$

- For example, if $h = \{(k, k), (\mathrm{E}_k(a), \mathrm{E}_k(a))\}$, we have $\mathcal{A}(h) = \{(k, k), (\mathrm{E}_k(a), \mathrm{E}_k(a)), (a, a)\}$.
- The irreducibles $\mathcal{I}(h)$ is a “minimal” hedge “equivalent” to $\mathcal{A}(h)$

Late hedged bisimulation

- A “symmetric” relation $\mathcal{R} \subset \mathcal{H} \times \mathcal{P} \times \mathcal{P}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$,

Late hedged bisimulation

- A “symmetric” relation $\mathcal{R} \subset \mathcal{H} \times \mathcal{P} \times \mathcal{P}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$,
  - h is consistent

Late hedged bisimulation

- A “symmetric” relation $\mathcal{R} \subset \mathcal{H} \times \mathcal{P} \times \mathcal{P}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$,
  - h is consistent
  - if $P \xrightarrow{\alpha} P'$

Late hedged bisimulation

- A “symmetric” relation $\mathcal{R} \subset \mathcal{H} \times \mathcal{P} \times \mathcal{P}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$,
  - h is consistent
  - if $P \xrightarrow{\alpha} P'$
    1. if $\alpha = \tau$,

Late hedged bisimulation

- A “symmetric” relation $\mathcal{R} \subset \mathcal{H} \times \mathcal{P} \times \mathcal{P}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$,
  - h is consistent
  - if $P \xrightarrow{\alpha} P'$ then $Q \xrightarrow{\beta} Q'$ and
    1. if $\alpha = \tau$,

Late hedged bisimulation

- A “symmetric” relation $\mathcal{R} \subset \mathcal{H} \times \mathcal{P} \times \mathcal{P}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$,
  - h is consistent
  - if $P \xrightarrow{\alpha} P'$ then $Q \xrightarrow{\beta} Q'$ and
    1. if $\alpha = \tau$, then $\beta = \tau$ and $(h, P', Q') \in \mathcal{R}$

Late hedged bisimulation

- A "symmetric" relation $\mathcal{R} \subset \mathcal{H} \times \mathcal{P} \times \mathcal{P}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$,
  - h is consistent
  - if $P \xrightarrow{\alpha} P'$ then $Q \xrightarrow{\beta} Q'$ and
    1. if $\alpha = \tau$, then $\beta = \tau$ and $(h, P', Q') \in \mathcal{R}$
    2. if $\alpha = a(x)$,

Late hedged bisimulation

- A "symmetric" relation $\mathcal{R} \subset \mathcal{H} \times \mathcal{P} \times \mathcal{P}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$,
  - h is consistent
  - if $P \xrightarrow{\alpha} P'$ and $\text{ch}(\alpha) \in \pi_1(h)$ then $Q \xrightarrow{\beta} Q'$ and
    1. if $\alpha = \tau$, then $\beta = \tau$ and $(h, P', Q') \in \mathcal{R}$
    2. if $\alpha = a(x)$,

Late hedged bisimulation

- A “symmetric” relation $\mathcal{R} \subset \mathcal{H} \times \mathcal{P} \times \mathcal{P}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$,
  - h is consistent
  - if $P \xrightarrow{\alpha} P'$ and $\mathrm{ch}(\alpha) \in \pi_1(h)$ then $Q \xrightarrow{\beta} Q'$ and
    1. if $\alpha = \tau$, then $\beta = \tau$ and $(h, P', Q') \in \mathcal{R}$
    2. if $\alpha = a(x)$, then $\beta = b(x)$ and $(a, b) \in \mathcal{S}(h)$

Late hedged bisimulation

- A "symmetric" relation $\mathcal{R} \subset \mathcal{H} \times \mathcal{P} \times \mathcal{P}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$,
  - h is consistent
  - if $P \xrightarrow{\alpha} P'$ and $\mathrm{ch}(\alpha) \in \pi_1(h)$ then $Q \xrightarrow{\beta} Q'$ and
    1. if $\alpha = \tau$, then $\beta = \tau$ and $(h, P', Q') \in \mathcal{R}$
    2. if $\alpha = a(x)$, then $\beta = b(x)$ and $(a, b) \in \mathcal{S}(h)$

       for all $(M, N) \in \mathcal{S}(h \quad)$, $(h \quad, P'\{^M/_x\}, Q'\{^N/_x\}) \in \mathcal{R}$

Late hedged bisimulation

- A “symmetric” relation $\mathcal{R} \subset \mathcal{H} \times \mathcal{P} \times \mathcal{P}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$,
  - h is consistent
  - if $P \xrightarrow{\alpha} P'$ and $\mathrm{ch}(\alpha) \in \pi_1(h)$ then $Q \xrightarrow{\beta} Q'$ and
    1. if $\alpha = \tau$, then $\beta = \tau$ and $(h, P', Q') \in \mathcal{R}$
    2. if $\alpha = a(x)$, then $\beta = b(x)$ and
       $(a, b) \in \mathcal{S}(h)$
       for all $B \subset \mathcal{N} \times \mathcal{N}$ consistent (and minimal, and fresh)
       for all $(M, N) \in \mathcal{S}(h \cup B)$, $(h \cup B, P'\{^M/_x\}, Q'\{^N/_x\}) \in \mathcal{R}$

Late hedged bisimulation

- A "symmetric" relation $\mathcal{R} \subset \mathcal{H} \times \mathcal{P} \times \mathcal{P}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$,
  - h is consistent
  - if $P \xrightarrow{\alpha} P'$ and $\text{ch}(\alpha) \in \pi_1(h)$ then $Q \xrightarrow{\beta} Q'$ and
    1. if $\alpha = \tau$, then $\beta = \tau$ and $(h, P', Q') \in \mathcal{R}$
    2. if $\alpha = a(x)$, then $\beta = b(x)$ and
       $(a, b) \in \mathcal{S}(h)$
       for all $B \subset \mathcal{N} \times \mathcal{N}$ consistent (and minimal, and fresh)
       for all $(M, N) \in \mathcal{S}(h \cup B)$, $(h \cup B, P'\{^M/_x\}, Q'\{^N/_x\}) \in \mathcal{R}$
    3. if $\alpha = (\nu\tilde{c})\,\overline{a}\,M$,

Late hedged bisimulation

- A "symmetric" relation $\mathcal{R} \subset \mathcal{H} \times \mathcal{P} \times \mathcal{P}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$,
  - h is consistent
  - if $P \xrightarrow{\alpha} P'$ and $\mathrm{ch}(\alpha) \in \pi_1(h)$ then $Q \xrightarrow{\beta} Q'$ and
    1. if $\alpha = \tau$, then $\beta = \tau$ and $(h, P', Q') \in \mathcal{R}$
    2. if $\alpha = a(x)$, then $\beta = b(x)$ and
       $(a, b) \in \mathcal{S}(h)$
       for all $B \subset \mathcal{N} \times \mathcal{N}$ consistent (and minimal, and fresh)
       for all $(M, N) \in \mathcal{S}(h \cup B)$, $(h \cup B, P'\{^M/_x\}, Q'\{^N/_x\}) \in \mathcal{R}$
    3. if $\alpha = (\nu\tilde{c})\,\overline{a}\,M$, then $\beta = (\nu\tilde{d})\,\overline{b}\,N$ and
       $(a, b) \in \mathcal{S}(h)$

Late hedged bisimulation

- A “symmetric” relation $\mathcal{R} \subset \mathcal{H} \times \mathcal{P} \times \mathcal{P}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$,
  - h is consistent
  - if $P \xrightarrow{\alpha} P'$ and $\mathrm{ch}(\alpha) \in \pi_1(h)$ then $Q \xrightarrow{\beta} Q'$ and
    1. if $\alpha = \tau$, then $\beta = \tau$ and $(h, P', Q') \in \mathcal{R}$
    2. if $\alpha = a(x)$, then $\beta = b(x)$ and
       $(a, b) \in \mathcal{S}(h)$
       for all $B \subset \mathcal{N} \times \mathcal{N}$ consistent (and minimal, and fresh)
       for all $(M, N) \in \mathcal{S}(h \cup B)$, $(h \cup B, P'\{^M/_x\}, Q'\{^N/_x\}) \in \mathcal{R}$
    3. if $\alpha = (\nu\tilde{c})\,\overline{a}\,M$, then $\beta = (\nu\tilde{d})\,\overline{b}\,N$ and
       $(a, b) \in \mathcal{S}(h)$
       $(\mathcal{I}(h \cup \{(M, N)\}), P', Q') \in \mathcal{R}$

From late to open hedged

- An environment is now composed of

From late to open hedged

- An environment is now composed of
  - a hedge h: the emitted messages messages

From late to open hedged

- An environment is now composed of
  - a hedge h: the emitted messages messages
  - a finite set of pair of names v: the input names

From late to open hedged

- An environment is now composed of
  - a hedge h: the emitted messages messages
  - a finite set of pair of names v: the input names
  - $\prec$: precedence relation to indicate which part of h was available (for each input)

From late to open hedged

- An environment is now composed of
  - a hedge h: the emitted messages messages
  - a finite set of pair of names v: the input names
  - $\prec$: precedence relation to indicate which part of h was available (for each input)
  - two sets of names (γ_l, γ_r): type constraints for input names

Environments

$$P \stackrel{\text{def}}{=} (\nu k)\,(\nu m)\,\overline{a}\langle \mathsf{E}_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \parallel [x = k]\overline{a}\langle a\rangle)$$

$h^{\frac{1}{2}}$	$v^{\frac{1}{2}}$	$\prec^{\frac{1}{2}}$	γ_l
$\emptyset$	$\{a\}$	$\emptyset$	$\emptyset$

Environments

$$P \stackrel{\text{def}}{=} (\nu k)\,(\nu m)\,\overline{a}\langle \mathsf{E}_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \,\|\, [x = k]\overline{a}\langle a\rangle)$$

$h^{\frac{1}{2}}$	$v^{\frac{1}{2}}$	$\prec^{\frac{1}{2}}$	γ_l
$\emptyset$	$\{a\}$	$\emptyset$	$\emptyset$
$\{\mathsf{E}_k(m)\}$	$\{a\}$	$\emptyset$	$\{a\}$

Environments

$$P \stackrel{\text{def}}{=} (\nu k)(\nu m)\,\overline{a}\langle E_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \parallel [x = k]\overline{a}\langle a\rangle)$$

$h^{\frac{1}{2}}$	$v^{\frac{1}{2}}$	$\prec^{\frac{1}{2}}$	γ_l
$\emptyset$	$\{a\}$	$\emptyset$	$\emptyset$
$\{E_k(m)\}$	$\{a\}$	$\emptyset$	$\{a\}$
$\{E_k(m)\}$	$\{a, x\}$	$E_k(m) \prec x$	$\{a\}$

Environments

$$P \stackrel{\text{def}}{=} (\nu k)(\nu m)\,\overline{a}\langle \mathrm{E}_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \parallel [x = k]\overline{a}\langle a\rangle)$$

$h^{\frac{1}{2}}$	$v^{\frac{1}{2}}$	$\prec^{\frac{1}{2}}$	γ_l
$\emptyset$	$\{a\}$	$\emptyset$	$\emptyset$
$\{\mathrm{E}_k(m)\}$	$\{a\}$	$\emptyset$	$\{a\}$
$\{\mathrm{E}_k(m)\}$	$\{a, x\}$	$\mathrm{E}_k(m) \prec x$	$\{a\}$

Environments

$$P \stackrel{\text{def}}{=} (\nu k)\,(\nu m)\,\overline{a}\langle \mathsf{E}_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \,\|\, [x = k\,]\overline{a}\langle a\rangle)$$

$h^{\frac{1}{2}}$	$v^{\frac{1}{2}}$	$\prec^{\frac{1}{2}}$	γ_l
$\emptyset$	$\{a\}$	$\emptyset$	$\emptyset$
$\{\mathsf{E}_k(m)\}$	$\{a\}$	$\emptyset$	$\{a\}$
$\{\mathsf{E}_k(m)\}$	$\{a, x\}$	$\mathsf{E}_k(m) \prec x$	$\{a\}$
$\{\mathsf{E}_k(m), k, m\}$	$\{a, x\}$	$\mathsf{E}_k(m) \prec x$	$\{a\}$

Environments

$$P \stackrel{\text{def}}{=} (\nu k)\,(\nu m)\,\overline{a}\langle \mathrm{E}_k(m)\rangle.a(x).(\overline{a}\langle k\rangle \| [x = k]\overline{a}\langle a\rangle)$$

$h^{\frac{1}{2}}$	$v^{\frac{1}{2}}$	$\prec^{\frac{1}{2}}$	γ_l
$\emptyset$	$\{a\}$	$\emptyset$	$\emptyset$
$\{\mathrm{E}_k(m)\}$	$\{a\}$	$\emptyset$	$\{a\}$
$\{\mathrm{E}_k(m)\}$	$\{a, x\}$	$\mathrm{E}_k(m) \prec x$	$\{a\}$
$\{\mathrm{E}_k(m), k, m\}$	$\{a, x\}$	$\mathrm{E}_k(m) \prec x$	$\{a\}$

From late to open hedged

- An environment is now composed of
  - a hedge h: the emitted messages messages
  - a finite set of pair of names v: the input names
  - $\prec$: precedence relation to indicate which part of h was available (for each input)
  - two sets of names (γ_l, γ_r): type constraints for input names
- Moreover, we define
  - the sets of pair of respectful substitutions (σ, ρ)

From late to open hedged

- An environment is now composed of
  - a hedge h: the emitted messages messages
  - a finite set of pair of names v: the input names
  - $\prec$: precedence relation to indicate which part of h was available (for each input)
  - two sets of names (γ_l, γ_r): type constraints for input names
- Moreover, we define
  - the sets of pair of respectful substitutions (σ, ρ)
  - the consistency of an environment

From late to open hedged

- An environment is now composed of
  - a hedge h: the emitted messages messages
  - a finite set of pair of names v: the input names
  - $\prec$: precedence relation to indicate which part of h was available (for each input)
  - two sets of names (γ_l, γ_r): type constraints for input names
- Moreover, we define
  - the sets of pair of respectful substitutions (σ, ρ)
  - the consistency of an environment
  - the updating of an environment

From late to open hedged

- An environment is now composed of
  - a hedge h: the emitted messages messages
  - a finite set of pair of names v: the input names
  - $\prec$: precedence relation to indicate which part of h was available (for each input)
  - two sets of names (γ_l, γ_r): type constraints for input names
- Moreover, we define
  - the sets of pair of respectful substitutions (σ, ρ)
  - the consistency of an environment
  - the updating of an environment
- ... and we finally define the bisimulation.

From late to open hedged

- An environment is now composed of
  - a hedge h: the emitted messages messages
  - a finite set of pair of names v: the input names
  - $\prec$: precedence relation to indicate which part of h was available (for each input)
  - two sets of names (γ_l, γ_r): type constraints for input names
- Moreover, we define
  - the sets of pair of respectful substitutions (σ, ρ)
  - the consistency of an environment
  - the updating of an environment
- ... and we finally define the bisimulation.
- The definition obtained is sound.

Conclusion

Conclusion

- Definition of K-open bisimulation

Conclusion

- Definition of K-open bisimulation
  - Coincides with open bisimulation

Conclusion

- Definition of K-open bisimulation
  - Coincides with open bisimulation
  - Defined of bigger set of contexts that preserves open

Conclusion

- Definition of K-open bisimulation
  - Coincides with open bisimulation
  - Defined of bigger set of contexts that preserves open
- Open hedged bisimulation

Conclusion

- Definition of K-open bisimulation
  - Coincides with open bisimulation
  - Defined of bigger set of contexts that preserves open
- Open hedged bisimulation
  - Sound w.r.t. late hedged bisimulation

Future work

Future work

- Study open hedged bisimulation

Future work

- Study open hedged bisimulation
  - Link with symbolic bisimulation of [BBN04]

Future work

- Study open hedged bisimulation
  - Link with symbolic bisimulation of [BBN04]
  - Congruence properties?

Thank you!

Thank you!

Questions?

Bibliography

D. Sangiorgi
A Theory of Bisimulation for the π-calculus.

J. Borgström, S. Briais and U. Nestmann
Symbolic Bisimulations in the Spi Calculus

e-respectful contexts

If $e = (O, V, \prec)$, a context $C[\cdot]$ respects e if it is generated by

$$
\begin{array}{rcll}
C_N[\cdot] & ::= & [\cdot] & \text{if } N = \emptyset \\
& | & P \| C_N[\cdot] \,|\, C_N[\cdot] \| P & \\
& | & P + C_N[\cdot] \,|\, C_N[\cdot] + P & \\
& | & !\, C_N[\cdot] & \\
& | & \phi C_N[\cdot] & \\
& | & (\nu x)\, C_{N \setminus \{x\}}[\cdot] & \\
& | & \overline{a}\langle z \rangle . C_N[\cdot] & \\
& | & a(x).C_N[\cdot] & \text{if } x \notin O \cup V \\
& | & a(x).C_{N \cup N'}[\cdot] & \text{if } x \in V \text{ and } N' = \{ n \in O \mid \neg n \prec x \}
\end{array}
$$

with $N \subset O$ and $C_\emptyset[\cdot]$ as start symbol.