Protokollsicherheit

Prozesskalkül

Communication und Concurrency

Analyse kryptographische Protokolle erfordert:

- Formulierung der Protokolle in Sprache mit formaler Semantik
- Modellierung von interaktivem, dynamischen Kommunikationsverhalten
- Abstraktion von Details (z.B. Netzwerk)

Eine Möglichkeit: **Prozesskalküle**

- Calculus of Communicating Systems (CCS) [Milner, $\sim$1980]
- Communicating Sequential Processes (CSP) [Hoare, $\sim$1980]
- π-Kalkül [Milner, Parrow, Walker, 1992]
- spi-Kalkül [Abadi, Gordon, 1999]
- Angewandter π-Kalkül [Abadi, Forunet, 2001]

Communication und Concurrency

Programme und Systeme mit nichtsequentiellen und interaktivem Verhalten sind allgegenwärtig.

Abstrakte Theorie kommunizierender Systeme wird seit den 1970er Jahren intensiv studiert.

Ziel: grundlegende Ergebnisse analog zur Charakterisierung der berechenbaren Funktionen durch Turing Maschinen, rekursive Funktionen, ...

Typische Fragen:

- Wie kann man die Gleichheit von interaktiven Systemen erfassen?
- Welche algebraische Struktur haben solche Systeme?
- Wie ist die Ausdrucksstärke verschiedener Formalismen? Welche Analysemethoden gibt es?

CCS — Syntax

Sei $\mathcal{N}$ eine unendliche Menge von Namen.

CCS-Prozesse:

$$P, Q ::= 0 \mid a.P \mid \overline{a}.P \mid \nu a.P \mid P + Q \mid P|Q$$

Dabei läuft a über $\mathcal{N}$.

Die Prozesskonstruktoren sind in absteigender Bindungsstärke angegeben. Beispiel:

$$a.(b.0 + c.0) \mid \overline{a}.0$$

steht für

$$(a.((b.0) + (c.0))) \mid (\overline{a}.0)$$

N.B.: Die Originaldefiniton von CCS enthält noch einen Prozess $\tau.P$.

CCS — Informell

Deadlock: 0 ist ein Prozess, der keine Aktion ausführen kann.

Eingabe: Der Prozess $a.P$ wartet auf eine Eingabe auf Kanal a und verhält sich dann wie Prozess P.

Ausgabe: Der Prozess $\overline{a}.P$ gibt ein Signal über Kanal a aus und verhält sich dann wie Prozess P.

Auswahl: Der Prozess $P + Q$ hat die Wahl, sich so zu verhalten wie P oder wie Q.

Beispiele:
$a.(\overline{c}.0 + \overline{b}.0)$ erwartet eine Eingabe auf Kanal a und kann dann auf Kanal b oder c ausgeben.
$a.\overline{c}.0 + a.\overline{b}.0$ erwartet entweder erst eine Eingabe auf Kanal a und gibt auf Kanal b aus, oder erst eine Eingabe auf Kanal a und gibt dann auf Kanal c aus.

CCS — Informell

Parallele Komposition: Der Prozess $P \mid Q$ erlaubt die Aktionen,

- die P und Q jeweils alleine ausführen können; sowie
- die Interaktion von P und Q auf einem beliebigen Kanal a, wenn einer der Prozesse eine Ausgabe der andere eine Eingabe auf diesem Kanal ausführen kann.

Beispiel:

- $a.(b.0 + c.0) \mid \overline{a}.0$ kann ein a eingeben und verhält sich danach wie $(b.0 + c.0) \mid \overline{a}.0$.
- $a.(b.0 + c.0) \mid \overline{a}.0$ kann ein a ausgeben und verhält sich danach wie $a.(b.0 + c.0) \mid 0$.
- $a.(b.0 + c.0) \mid \overline{a}.0$ kann eine interne Aktion ausführen und wird dann zu $(b.0 + c.0) \mid 0$.

CCS — Informell

Restriktion: Der Prozess $\nu a.P$ erlaubt alle Aktionen von P außer der Ein- und Ausgabe über Kanal a.

Beispiel: $\nu a.\,(a.(b.0 + c.0) \mid \overline{a}.0)$ kann eine interne Aktion ausführen, aber weder ein a eingeben noch ausgeben.

In der Restriktion $\nu a.P$ wird der Name a in P gebunden.

Freie Namen eines Prozesses:

$$fn(0) = \emptyset \qquad fn(a.P) = fn(\overline{a}.P) = \{a\} \cup fn(P)$$
$$fn(\nu a.\,P) = fn(P) \setminus \{a\} \qquad fn(P + Q) = fn(P \mid Q) = fn(P) \cup fn(Q)$$

Wir identifizieren Prozesse bis auf Umbenennung von gebundenen Variablen, d.h. wir identifizieren zum Beispiel $\nu a.\,(a.(b.0 + c.0) \mid \overline{a}.0)$ und $\nu d.\,(d.(b.0 + c.0) \mid \overline{d}.0)$.

CCS — Operationelle Semantik

Wir definieren die operationelle Semantik von CCS durch eine binäre Relation $\longrightarrow$ auf Prozessen, welche **Prozessreduktion** ausdrückt.

Grundidee: Erlaube folgende Reduktion

$$a.P + Q \mid \overline{a}.R + S \longrightarrow P \mid R$$

für alle $a \in \mathcal{N}$ und alle Prozesse P, Q, R, S.

Eine Ausgabe (bzw. Eingabe) kann nur ausgeführt werden, wenn es einen Kommunikationsparter gibt, der die zugehörige Eingabe (bzw. Ausgabe) ausführt.

Beispiel: $b.0 + c.0 \mid \overline{b}.0 + 0 \longrightarrow 0 \mid 0$
Problem: $b.0 \mid \overline{b}.0 \not\longrightarrow$ und $b.0 + c.0 \mid \overline{b}.0 + 0 \mid 0 \longrightarrow 0 \mid 0$, ...

Kontexte

Ein **Kontext** ist ein „Prozessterm mit einem Loch $[-]$“.

$$C ::= [-] \mid 0 \mid a.C \mid \overline{a}.C \mid \nu a.C \\ \mid C + P \mid P + C \mid C|P \mid P|C$$

Schreibe $C[P]$ für den Prozess, den man durch Einsetzen von P in das Loch des Kontexts $C[-]$ erhält.

Beispiel:

$$C[-] = \nu a.\,(a.(b.0 + c.0) \mid \overline{a}.[-])$$
$$C[a.0 + d.0] = \nu a.\,(a.(b.0 + c.0) \mid \overline{a}.(a.0 + d.0))$$

Beachte: Durch Einsetzung in Kontexte können Variablen gebunden werden (hier: a).

Kontexte

Ein **Auswertungskontext** ist ein spezieller Kontext:

$$E ::= [-] \mid \nu a.E \mid E|P \mid P|E$$

Mit Auswertungskontexten identifizert man die Subterme eines Prozessterms, die reduziert werden können.
D.h. ein Subterm P' von P kann reduziert werden, wenn man P als $E[P']$ schreiben kann.

Beispiel:
$\nu a.\,(a.(b.0 + c.0) \mid \overline{a}.(a.0 + d.0))$ lässt sich schreiben als:

$E_1[a.(b.0 + c.0) \mid \overline{a}.(a.0 + d.0)]$ für $E_1 = \nu a.\,[-]$,

$E_2[a.(b.0 + c.0)]$ für $E_2 = \nu a.\,([-] \mid \overline{a}.(a.0 + d.0))$,

$E_3[a.(a.0 + d.0)]$ für $E_3 = \nu a.\,(a.(b.0 + c.0) \mid [-])$.

Strukturelle Äquivalenz

Strukturelle Äquivalenz ist die kleinste Äquivalenzrelation $\equiv$ auf Prozessen, für die gilt:

1.

$$P \mid 0 \equiv P \qquad P \mid Q \equiv Q \mid P \qquad (P \mid Q) \mid R \equiv P \mid (Q \mid R)$$

$$P + 0 \equiv P \qquad P + Q \equiv Q + P \qquad (P + Q) + R \equiv P + (Q + R)$$

$$\nu a.\, 0 \equiv 0 \qquad \nu a.\, \nu b.\, P \equiv \nu b.\, \nu a.\, P$$

$$\nu a.\, (P \mid Q) \equiv (\nu a.\, P) \mid Q \quad \text{wenn } a \notin fn(Q)$$

2. Wenn $P \equiv Q$ dann $E[P] \equiv E[Q]$ für alle Auswertungskontexte $E[-]$.

CCS — Operationelle Semantik

Die Reduktionsrelation $\longrightarrow$ zwischen Prozesse ist nun induktiv durch folgende Regeln definiert.

$$\frac{}{a.P + Q \mid \overline{a}.R + S \longrightarrow P \mid R}$$ Kommunikation

$$\frac{P \longrightarrow Q}{P + P' \longrightarrow Q}$$ Auswahl

$$\frac{P \longrightarrow Q}{E[P] \longrightarrow E[Q]}$$ Kontextabschluss

$$\frac{P' \equiv P \qquad P \longrightarrow Q \qquad Q \equiv Q'}{P' \longrightarrow Q'}$$ Struktur

Beispiel

Beispiel 1:

$$\begin{aligned}
&€.(\overline{coffee}.0 + €.\overline{chocolate}.0) \mid \overline{€}.coffee.0 \\
&\longrightarrow \overline{coffee}.0 + €.\overline{chocolate}.0 \mid coffee.0 \\
&\longrightarrow \overline{coffee}.0 + €.\overline{chocolate}.0 \mid coffee.0 \\
&\longrightarrow 0
\end{aligned}$$

(Nachfolger sind hier bis auf strukturelle Äquivalenz eindeutig bestimmt.)

Beispiel 2:

$$\begin{aligned}
&€.\overline{coffee}.0 + €.€.\overline{chocolate}.0 \mid \overline{€}.coffee.0 \\
&\longrightarrow €.\overline{chocolate}.0 \mid coffee.0 \\
&\not\longrightarrow
\end{aligned}$$

Prozessäquivalenz

Grundlegende Frage:
Wann sollen zwei Prozesse mit unterschiedlicher Implementierung und unterschiedlichem internen Verhalten als gleich erachtet werden?

Aspekte:

- Prozessalgebra: Gleiche Prozesse können in verschiedenen Kontexten durch gleiche ersetzt werden.
- Für welche Zwecke soll die Äquivalenz benuzt werden?
- Wie kann man Gleichheit von Prozessen beweisen?

Beispiele:

1. $\mathit{protocol}(key) \mid \mathit{attacker} \approx \mathit{protocol}(key') \mid \mathit{attacker}$
2. Spezifikation und Verifikation: Spezifiziere gewünschtes Verhalten durch einen Prozess; leite aus Implementierung einen Prozess ab und prüfe Gleichheit mit der Spezifikation.

Spuräquivalenz

Zwei Prozesse können die gleichen Folgen von Aktionen ausführen.

Problem:
Damit sind folgende Prozesse gleich:

$$€.(coffee.0 + chocolate.0)$$

$$€.coffee.0 + €.chocolate.0$$

Spuräquivalenz kann nützlich sein, ist aber oft zu grob.

Observationen

Wir betrachten stattdessen *observationelle Äquivalenz*, in der zwei Progesse gleich sind, wenn sie von der Umgebung nicht unterschieden werden können.

Observationen

Wir betrachten stattdessen *observationelle Äquivalenz*, in der zwei Progesse gleich sind, wenn sie von der Umgebung nicht unterschieden werden können.

Als mögliche Observation betrachten wir ein **Strong Barb** genanntes Prädikat $P \downarrow a$ (etwa: Widerhaken).

$P \downarrow a$ gilt falls P sofort auf dem Kanal a senden kann.

Formal: $P \downarrow a$ falls $P \equiv E[\overline{a}.P_1 + P_2]$ für geeignete Prozesse P_1, P_2 und Auwertungskontext $E[-]$, der a nicht bindet.

Beispiele:

$$\overline{a}.(b.0 + c.0) \downarrow a$$

$$\overline{a}.(b.0 + c.0) \not\downarrow b$$

Observationen

Man könnte nun versuchen Prozesse P und Q als gleich auffassen, wenn sie in beliebigen Kontexten die gleichen Observationen zulassen, d.h.

$$E[P] \downarrow a \quad \text{gdw} \quad E[Q] \downarrow a$$

für alle Auswertungskontexte $E[-]$ und alle a.

Observationen

Man könnte nun versuchen Prozesse P und Q als gleich auffassen, wenn sie in beliebigen Kontexten die gleichen Observationen zulassen, d.h.

$$E[P] \downarrow a \quad \text{gdw} \quad E[Q] \downarrow a$$

für alle Auswertungskontexte $E[-]$ und alle a.

Auf diese Art erfasst man aber nur einen Auswertungsschritt.

Beispiel: $\overline{a}.b.0$ und $\overline{a}.c.0$ wären äquivalent.

Observationen

Man könnte nun versuchen Prozesse P und Q als gleich auffassen, wenn sie in beliebigen Kontexten die gleichen Observationen zulassen, d.h.

$$E[P] \downarrow a \quad \text{gdw} \quad E[Q] \downarrow a$$

für alle Auswertungskontexte $E[-]$ und alle a.

Auf diese Art erfasst man aber nur einen Auswertungsschritt.

Beispiel: $a.b.0$ und $a.c.0$ wären äquivalent.

Auch wenn man sagt, dass die Prozesse eine beliebige Anzahl von Schritten ausführen dürfen, bevor die Observation gemacht wird, d.h.

$$\exists P'.\, E[P] \longrightarrow^* P' \downarrow a \quad \text{gdw} \quad \exists Q'.\, E[Q] \longrightarrow^* Q' \downarrow a$$

(für alle $E[-]$ und a), so kann man $a.(b.0 + c.0)$ und $a.b.0 + a.c.0$ nicht unterscheiden.

Bisimulation

Idee: Jede Evolution von P muss sich einer äquivalenten von Q zuordnen lassen und umgekehrt.

Beispiel:
Semaphore S_1, die *einen* Prozess in kritischen Bereich lässt.
Semaphore S_2, die *zwei* Prozesse in kritischen Bereich lässt.

$$S_1 = p.S_1' \qquad S_2 = p.S_2'$$
$$S_1' = v.S_1 \qquad S_2' = p.S_2'' + v.S_2$$
$$S_2'' = v.S_2'$$

$S_1 \mid S_1$ und S_2 sollten äquivalent sein.
Evolutionen dieses Prozesse können zugeordnet werden:

$E[S_1 \mid S_1]$ und $E[S_2]$ $\qquad$ $E[S_1' \mid S_1]$ und $E[S_2']$

$E[S_1 \mid S_1']$ und $E[S_2']$ $\qquad$ $E[S_1' \mid S_1']$ und $E[S_2'']$

Bisimulation

Idee: Jede Evolution von P muss sich einer äquivalenten von Q zuordnen lassen und umgekehrt.

Beispiel:
Bei den Prozessen $P = a.(\overline{b}.0 + \overline{c}.0) \mid \overline{a}.0$ und $Q = (a.\overline{b}.0 + a.\overline{c}.0) \mid \overline{a}.0$ kann man die Redukte nicht äquivalent zuordnen.

Wir haben $Q \longrightarrow \overline{b}.0$ und $Q \longrightarrow \overline{c}.0$, aber nur $P \longrightarrow \overline{b}.0 + \overline{c}.0$.

Es gilt $(\overline{b}.0 + \overline{c}.0) \downarrow b$ aber $\overline{c}.0 \not\downarrow b$ sowie $(\overline{b}.0 + \overline{c}.0) \downarrow c$ aber $\overline{b}.0 \not\downarrow c$.

$\Rightarrow$ Kann die Redukte nicht so zuordnen, dass die zugeordneten Prozesse stets nur die gleichen Observationen zulassen.

Barbed Equivalence

Definition:
Strong Barbed Equivalence $\simeq$ ist die größte symmetrische Relation mit folgenden Eigenschaften (für alle P, Q, P'):

1. Aus $P \simeq Q$ und $P \downarrow a$ folgt $Q \downarrow a$ für alle a.
2. Gilt $P \simeq Q$ und $P \longrightarrow P'$, so gibt es Q' mit $Q \longrightarrow Q'$ und $P' \simeq Q'$.
3. Aus $P \simeq Q$ folgt $E[P] \simeq E[Q]$ für jeden Auswertungskontext $E[-]$.

Satz: $\simeq$ ist wohldefiniert und eine Äquivalenzrelation.

Beispiele:

$$a.(\overline{b}.0 + \overline{c}.0) \not\simeq a.\overline{b}.0 + a.\overline{c}.0$$

$$S_1 \mid S_1 \simeq S_2$$

$$\nu a.(a.0) \simeq 0$$

Labelled Transitions

Die Quantifizierung über (unendlich viele) Kontexte macht es schwierig, konkrete Äquivalenzen zu beweisen.

Beweistechnik: Bisimulationen mit „Labelled Transitions“.

Idee: Formalisiere nicht nur die tatsächlichen, sondern auch die *möglichen* Interaktionen mit der Umgebung.

Reduktionsrelation mit Labels $P \xrightarrow{\alpha} Q$, wobei $\alpha \in \{a, \overline{a}, \tau \mid a \in \mathcal{N}\}$.

- Bisherige Reduktion $P \longrightarrow Q$ tauchen (bis auf $\equiv$) als $P \xrightarrow{\tau} Q$ auf.
- Eine Reduktion $P \xrightarrow{a} Q$ sagt, dass P auf Kanal a empfangen kann (von einem bisher unbekannten Prozess) und dann zu Q wird.
- Eine Reduktion $P \xrightarrow{\overline{a}} Q$ sagt, dass P auf Kanal $\overline{a}$ senden kann (an einen bisher unbekannten Prozess) und dann zu Q wird.

Labelled Transitions

Transitionsrelation wird induktiv durch folgende Regeln definiert.

$$\frac{}{a.P \xrightarrow{a} P} \qquad \frac{}{\overline{a}.P \xrightarrow{\overline{a}} P}$$

$$\frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'} \qquad \frac{Q \xrightarrow{\alpha} Q'}{P + Q \xrightarrow{\alpha} Q'}$$

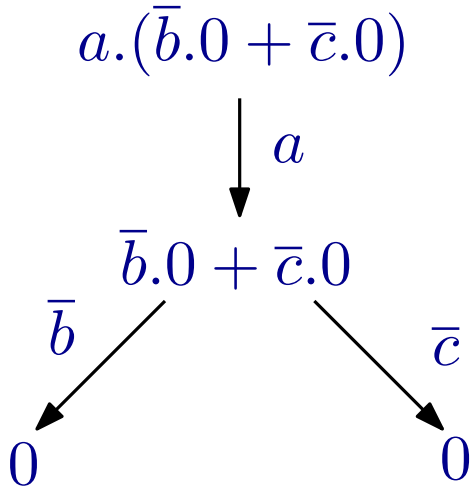
$$\frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q'} \qquad \frac{Q \xrightarrow{\alpha} Q'}{P \mid Q \xrightarrow{\alpha} P' \mid Q'} \qquad \frac{P \xrightarrow{a} P' \qquad Q \xrightarrow{\overline{a}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$$

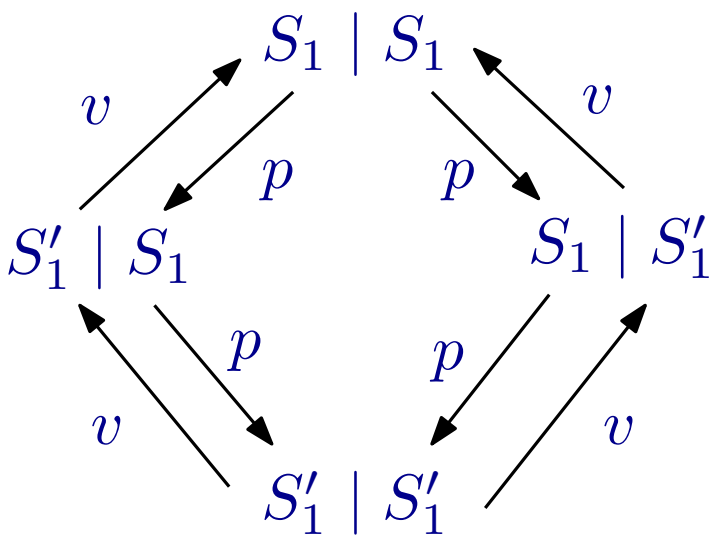
$$\frac{P \xrightarrow{\alpha} P'}{\nu a.\, P \xrightarrow{\alpha} \nu a.P'} \; \alpha \notin \{a, \overline{a}\}$$

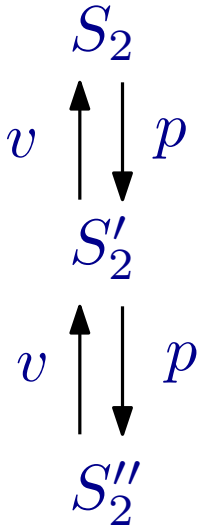
α läuft über $\{a, \overline{a}, \tau \mid a \in \mathcal{N}\}$.

Labelled Transitions

Beispiele:







Labelled Bisimulation

Definition:
Eine Relation $\mathcal{S}$ ist eine **starke Bisimulation** falls aus $P\mathcal{S}Q$ folgt:

1. Wenn $P \xrightarrow{\alpha} P'$, dann gibt es Q' mit $Q \xrightarrow{\alpha} Q'$ und $P'\mathcal{S}Q'$.
2. Wenn $Q \xrightarrow{\alpha} Q'$, dann gibt es P' mit $P \xrightarrow{\alpha} P'$ und $P'\mathcal{S}Q'$.

Satz: Es existiert eine **größte starke Bisimulationsrelation** $\sim$, die alle anderen Bisimulationsrelationen enthält, d.h. es gilt $\mathcal{S} \subseteq \sim$ für jede Bisimulationsrelation $\mathcal{S}$.

Definition:
Zwei Prozesse P und Q mit $P \sim Q$ heißen **stark bisimilar**.

Bemerkung: Um zu zeigen, dass P und Q stark bisimilar sind, genügt es *eine Bisimulation* $\mathcal{S}$ zu finden, für die $P\mathcal{S}Q$ gilt.

Labelled Bisimulation

Beispiel:

$$S_2 \sim S_1 \mid S_1$$

Es genügt, eine Bisimulationsrelation $\mathcal{S}$ mit $S_2 \; \mathcal{S} \; (S_1 \mid S_1)$ zu finden.

Wähle:

$$\mathcal{S} = \{(S_2, S_1 \mid S_1), (S_2', S_1' \mid S_1), (S_2', S_1 \mid S_1'), (S_2'', S_1' \mid S_1')\}$$

Labelled Bisimulation

Beispiel:

$$a.(b.0 + c.0) \not\sim a.b.0 + a.c.0$$

Angenommen es gäbe eine Bisimulation $\mathcal{S}$ mit $a.(b.0 + c.0)\ \mathcal{S}\ (a.b.0 + a.c.0)$.

- Es gilt $a.(b.0 + c.0) \xrightarrow{a} (b.0 + c.0)$.
- Der zweite Prozess hat folgende a-Transitionen: $(a.b.0 + a.c.0) \xrightarrow{a} b.0$ und $(a.b.0 + a.c.0) \xrightarrow{a} c.0$.
- Da $\mathcal{S}$ eine Bisimulation ist, muss dann auch $(b.0 + c.0)\ \mathcal{S}\ b.0$ oder $(b.0 + c.0)\ \mathcal{S}\ c.0$ gelten.
- $(b.0 + c.0)\mathcal{S}b.0$ ist nicht möglich: $(b.0 + c.0) \xrightarrow{c} 0$, aber $b.0 \not\xrightarrow{c}$.
- $(b.0 + c.0)\mathcal{S}c.0$ ist nicht möglich: $(b.0 + c.0) \xrightarrow{b} 0$, aber $c.0 \not\xrightarrow{c}$.

Barbed Equivalence und Labelled Bisimulation

Satz: Bisimulation $\sim$ ist eine Kongruenz, d.h. aus $P \sim Q$ folgt $C[P] \sim C[Q]$ für alle Kontexte $C[-]$.

Satz: Es gilt $\equiv \subseteq \sim$.

Bisimulation kann benutzt werden, um Äquivalenzen zu zeigen:

Satz: Aus $P \sim Q$ (Bisimulation) folgt $P \simeq Q$ (Barbed Equivalence).

Unter vernünftigen Annahmen (Rekursion ist so eingeschränkt, dass jeder Prozess höchstens endlich viele Nachfolger haben kann) kann Bisimulation alle Äquivalenzen zeigen:

Satz: Aus $P \simeq Q$ (Barbed Equivalence) folgt $P \sim Q$ (Bisimulation).

Bisimulation quantifiziert nicht über alle Kontexte.