Communicating Sequential Processes

Thomas Strathmann <thomas@pdp7.org>

6. Juli 2007

Gliederung

Modell

- parallele Komposition von Prozessen
- Prozesse können sich auf *Events* synchronisieren
  $\longrightarrow$ Unterscheidung in
  - *externe Events* – sichtbar für andere Prozesse
  - *interne Events* – ausschließlich prozesslokal, keine Synchronisation
- blockierendes Warten auf Events

Prozesse

Alphabet

Ein CSP Prozess P verfügt über ein Alphabet αP von externen Events.
Beispiel: $\alpha P = \{a, b, c\}$

Trace

Die *Traces* eines CSP Prozesses P sind Tupel aus αP^n.
Beispiel: $\langle a, b, c, a, b \rangle$

Prozesse

Alphabet

Ein CSP Prozess P verfügt über ein Alphabet αP von externen Events.
Beispiel: $\alpha P = \{a, b, c\}$

Trace

Die *Traces* eines CSP Prozesses P sind Tupel aus αP^n.
Beispiel: $\langle a, b, c, a, b \rangle$

Prozessgleichungen

Prozessgleichungen

Prozesse werden über *Prozessgleichungen* beschrieben:

$$p \stackrel{def}{=} P$$
$$p[i : \mathcal{D}] \stackrel{def}{=} P$$

Es sind

- p – Prozess Bezeichner
- i – Parameter mit
- $\mathcal{D}$ – Wertebereichen
- P – Prozessausdruck

Prozessausdrücke: Syntax

Syntax

Die Syntax von Prozessausdrücken in BNF:

$$P ::= STOP_A \mid p \mid p[e] \mid a \rightarrow P \mid P \sqcap P \mid P \,[]\, P \mid$$
$$P \parallel_A P \mid P \,|||\, P \mid P[d/c] \mid P \setminus A$$

Prozessausdrücke: Syntax II

$STOP_A$

$STOP_A$ kann keine Aktionen aus A ausführen (Deadlock)
Anmerkung: Index A weglassen, wenn $A = \alpha P$

p

verhält sich wie der vorher definierte Prozess mit Bezeichner p

$p[e]$

parametrisierter Prozess: Alle Vorkommen von i in $p[i : \mathcal{D}]$ durch e (aus $\mathcal{D}$) ersetzen.

Prozessausdrücke: Syntax II

$STOP_A$

$STOP_A$ kann keine Aktionen aus A ausführen (Deadlock)
Anmerkung: Index A weglassen, wenn $A = \alpha P$

p

verhält sich wie der vorher definierte Prozess mit Bezeichner p

$p[e]$

parametrisierter Prozess: Alle Vorkommen von i in $p[i : \mathcal{D}]$ durch e (aus $\mathcal{D}$) ersetzen.

Prozessausdrücke: Syntax III

$a \rightarrow P$

Präfix a: Prozess muss initial auf Event a synchronisieren. Das weitere Verhalten ist durch P gegeben.

$P \sqcap Q$

interner Nicht-Determinismus: beliebige Wahl aus Verhalten von P oder Q

$P \,[]\, Q$

externer Nicht-Determinismus: Umgebung beeinflusst Verhalten durch Wahl eines Präfix von P oder Q
Anmerkung: nur definiert, wenn $\alpha P = \alpha Q$

Prozessausdrücke: Syntax III

$a \rightarrow P$

Präfix a: Prozess muss initial auf Event a synchronisieren. Das weitere Verhalten ist durch P gegeben.

$P \sqcap Q$

interner Nicht-Determinismus: beliebige Wahl aus Verhalten von P oder Q

$P \,[\!]\, Q$

externer Nicht-Determinismus: Umgebung beeinflusst Verhalten durch Wahl eines Präfix von P oder Q
Anmerkung: nur definiert, wenn $\alpha P = \alpha Q$

Prozessausdrücke: Syntax IV

$P \parallel_A Q$

parallele Komposition: P und Q synchronisieren auf Events aus A. Alphabet ist $\alpha P \cap \alpha Q$.

$P \parallel\parallel Q$

Speziallfall von $P \parallel_A Q$, wenn $A = \emptyset$.
Beliebige Abfolge der Verhalten von P und Q.

$P[d/c]$

Umbenennung: Alle Vorkommen von c durch d ersetzen.

$P \setminus A$

Beschränkung: Events aus A sind für die Umgebung unsichtbar.

Prozessausdrücke: Syntax IV

$P \parallel_A Q$

parallele Komposition: P und Q synchronisieren auf Events aus A. Alphabet ist $\alpha P \cap \alpha Q$.

$P \parallel\parallel Q$

Speziallfall von $P \parallel_A Q$, wenn $A = \emptyset$.
Beliebige Abfolge der Verhalten von P und Q.

$P[d/c]$

Umbenennung: Alle Vorkommen von c durch d ersetzen.

$P \setminus A$

Beschränkung: Events aus A sind für die Umgebung unsichtbar.

Beispiel: Kaffeeautomat

$$AUTOMAT \stackrel{def}{=} (Geld \rightarrow (Kaffee \rightarrow AUTOMAT \,[]\, Espresso \rightarrow AUTOMAT))$$

$$KUNDE \stackrel{def}{=} (Kaffee \rightarrow KUNDE \,[]\, Geld \rightarrow (Espresso \rightarrow KUNDE))$$

$$EINKAUFEN \stackrel{def}{=} (AUTOMAT \|_{\{Kaffee, Espresso\}} KUNDE)$$

Modell

- Prozesse kommunizieren über benannte *Channels*
- Beschränkung des Alphabets αP auf *Channel Alphabet* $\alpha_c P$
- Für jeden Channel c_i ein (möglicherweise leeres) Channel Alphabet α_{c_i}
  $\rightarrow$ Abbildung der Netztopologie

Prozessausdrücke für Prozessnetzwerke: Syntax

Syntax

Die Syntax von Prozessausdrücken für Prozessnetzwerke in BNF:

$$P ::= STOP_A \mid p \mid p[e] \mid \mathbf{c}!\mathbf{e} \rightarrow \mathbf{P} \mid \mathbf{c}?\mathbf{x} : \mathbf{M} \rightarrow \mathbf{P} \mid P \sqcap P \mid P \,[\!]\, P \mid P \parallel_A P \mid P \,|||\, P \mid P[d/c] \mid P \setminus L \mid (\textbf{if } b \textbf{ then } P \textbf{ else } P)$$

(Änderungen/Erweiterung **hervorgehoben**)

Prozessausdrücke: Syntax II

$c!e \rightarrow P$

gibt Event e auf Channel c aus und verhält sich dann wie P.
Alphabet: $\alpha P \cup \{c.e\}$

$c?x : M \rightarrow P$

liest von Channel c einen Wert vom Typ M ein und bindet ihn an die Variable x. Alphabet: $\alpha P \cup \{c.v | v \in M\}$

if b then P else Q

verhält sich wie P, wenn b wahr, ansonsten wie Q
Anmerkung: nur definiert, wenn $\alpha P = \alpha Q$

Prozessausdrücke: Syntax II

$c!e \rightarrow P$

gibt Event e auf Channel c aus und verhält sich dann wie P.
Alphabet: $\alpha P \cup \{c.e\}$

$c?x : M \rightarrow P$

liest von Channel c einen Wert vom Typ M ein und bindet ihn an die Variable x. Alphabet: $\alpha P \cup \{c.v | v \in M\}$

if b **then** P **else** Q

verhält sich wie P, wenn b wahr, ansonsten wie Q
Anmerkung: nur definiert, wenn $\alpha P = \alpha Q$