Theorie reaktiver Systeme

Algorithmus

Traces

Algorithmus

Gegeben sind zwei LTS $P_1 = (S_1, s_1^0, T_1, \Sigma)$ und $P_2 = (S_2, s_2^0, T_2, \Sigma)$. Der Zustandsraum S ist die Vereinigung $S_1 \cup S_2$. Ziel ist eine Partitionierung in Äquivalenzklassen vom Zuständen.

Initialisierung:
Partition := $\{S\}$;
Splitter := Label $\times$ Partition;

Liegen bei Terminierung s_1^0 und s_2^0 in einer Äquivalenzklasse, so sind P_1 und P_2 bisimilar.

while (Splitter $\neq \emptyset$)

choose $(a, C_{spl}) \in$ Splitter;

forall $C \in$ Partition

$split(C, a, C_{spl}$, Partition, Splitter);

Splitter := Splitter - (a, C_{spl});

procedure $split(C, a, C_{spl}$, Partition, Splitter$)$

$C^+ := \{P \mid P \in C \wedge \exists\, Q.(P \xrightarrow{a} Q \wedge Q \in C_{spl})\}$;

if $(C^+ \neq C \wedge C^+ \neq \emptyset)$

$C^- := C - C^+$;

Partition : $=$ Partition $\cup \{C^+, C^-\} - \{C\}$;

Splitter := Splitter $\cup$ (Label$\times\{C^+, C^-\})-$ Label $\times\{C\}$;

Beispiel:

$P = (a \rightarrow \ b \rightarrow \ \text{STOP}) \ \square \ (a \rightarrow \ c \rightarrow \ \text{STOP})$

$Q = a \rightarrow ((b \rightarrow \ \text{STOP}) \ \square \ (c \rightarrow \ \text{STOP}))$

Initialisierung:
Partition $= \{\{S_0, S_1, S_2, S_3, S_4, R_0, R_1, R_2, R_3\}\}$ /* S
Splitter $= \{(a, S), \ (b, S), \ (c, S)\}$

Bei Terminierung des Algorithmus:

Partition $= \{\{S_0\}, \{R_0\}, \{S_1\}, \{R_1\}, \{S_2\}, \{S_3, S_4, R_2, R_3\}\}$

Splitter $= \emptyset$

Da S_0 und R_0 in verschiedenen Äquivalenzklassen liegen, sind P und Q nicht bisimilar.

Bemerkung: $\sim_{BS}$ ist kompositionell, d.h. wenn $P \sim_{BS} P'$ gilt, so gilt auch

- $a \rightarrow P \sim_{BS} a \rightarrow P'$
- $P; Q \sim_{BS} P'; Q$
- $P \Box Q \sim_{BS} P' \Box Q$
- $P \underset{\Sigma}{\|} Q \sim_{BS} P' \underset{\Sigma}{\|} Q$
- $P \,|||\, Q \sim_{BS} P' \,|||\, Q$

u.s.w.

Bemerkung: Wenn $P \sim_{BS} Q$ gilt, so sind P und Q in beliebiger Umgebung nicht unterscheidbar.

Aber: Betrachte Prozesse

$P = (a \rightarrow b \rightarrow \text{STOP}) \,\square\, (a \rightarrow c \rightarrow \text{STOP})$

$Q = a \rightarrow ((b \rightarrow \text{STOP}) \sqcap (c \rightarrow \text{STOP}))$

P und Q sind nicht bisimilar, P und Q sind in beliebiger Umgebung nicht unterscheidbar.

Unterscheidbarkeit in (beliebiger) Umgebung
Welches Verhalten kann beobachtet werden?

- Kommunikation: Events
- Blockade: Deadlock
- Terminierung

Traces

$TRACE = (\Sigma \cup \{\checkmark\})^*$
Zu einem Prozess P ist $trace(P) \subseteq TRACE$ die Menge der möglichen (Teil-)Folgen sichtbarer Events dieses Prozesses.

Für alle Prozesse P gilt:
$trace(P) \neq \emptyset$
$trace(P)$ ist präfix-abgeschlossen.

Notation

$\langle\rangle$ – leere trace

$\#tr$ – Länge von tr

$tr_1 \frown tr_2$ – Konkatenation, oft auch $tr_1 \cdot tr_2$ oder einfach $tr_1\, tr_2$

$tr = \langle a \rangle \frown tr'$ – a wird dann auch als $head(tr)$ oder $first(tr)$ bezeichnet, tr' als $tail(tr)$.

Präfix $tr \preceq tr'$: $\exists\, tr''.tr \frown tr'' = tr'$. Dabei darf $tr'' = \langle\rangle$ gelten. Bei einem *echten* Präfix $tr \prec tr'$ ist $tr'' \neq \langle\rangle$.

Projektion (Restriktion) $tr \restriction_A$ von tr auf Eventmenge A:

$$\langle\rangle \restriction_A = \langle\rangle$$

$$(\langle a \rangle \frown tr) \restriction_A = \begin{cases} \langle a \rangle \frown (tr \restriction_A), & \text{falls } a \in A \\ tr \restriction_A & \text{falls } a \notin A \end{cases}$$

to be continued...