

1 Definitionen

Stand: 31. Juli 2017

Kombination aus [BV15b] und [Sch16] mit Einflüssen von [BFLV16]:

Definition 1.1 (*Modal Error-I/O-Transitionssystem*). Ein Modal Error-I/O-Transitionssystem (MEIO) ist ein Tupel $(P, I, O, \longrightarrow, \dashrightarrow, p_0, E)$ mit:

- P : Menge der Zustände,
- $p_0 \in P$: Startzustand,
- I, O : disjunkte Mengen der (sichtbaren) Input- und Output-Aktionen,
- $\longrightarrow \subseteq P \times \Sigma_\tau \times P$: must-Transitions-Relation,
- $\dashrightarrow \subseteq P \times \Sigma_\tau \times P$: may-Transitions-Relation,
- $E \subseteq P$: Menge der Fehler-Zustände.

Es wird vorausgesetzt, dass $\longrightarrow \subseteq \dashrightarrow$ (syntaktische Konsistenz) gilt.

Das Alphabet bzw. die Aktionsmenge eines MEIO ist $\Sigma = I \cup O$. Die interne Aktion τ ist nicht in Σ enthalten. Jedoch wird $\Sigma_\tau := \Sigma \cup \{\tau\}$ definiert. Die Signatur eines MEIOs entspricht $\text{Sig}(P) = (I, O)$.

Falls $\longrightarrow = \dashrightarrow$ gilt, wird P auch Implementierung genannt.

Implementierungen entsprechen den in [Sch16] behandelten EIOs.

Must-Transitions sind Transitions, die von einer Verfeinerung implementiert werden müssen. Die may-Transitions sind hingegen die zulässigen Transitions für eine Verfeinerung.

MEIOs werden in dieser Arbeit durch ihre Zustandsmenge (z.B. P) identifiziert und falls notwendig werden damit auch die Komponenten indiziert (z.B. I_P anstatt I). Falls das MEIO selbst bereits einen Index hat (z.B. P_1) kann an der Komponente die Zustandsmenge als Index wegfallen und nur noch der Index des gesamten Transitionssystems verwendet werden (z.B. I_1 anstatt I_{P_1}). Zusätzlich stehen i, o, a, ω und α für Buchstaben aus den Alphabeten $I, O, \Sigma, O \cup \{\tau\}$ und Σ_τ .

Es wird die Notation $p \xrightarrow{\alpha} p'$ für $(p, \alpha, p') \in \dashrightarrow$ und $p \xrightarrow{\alpha}$ für $\exists p' : (p, \alpha, p') \in \dashrightarrow$ verwendet. Dies kann entsprechend auf Buchstaben-Sequenzen $w \in \Sigma_\tau^*$ erweitert werden: für $p \xrightarrow{w} p'$ ($p \dashrightarrow^w p'$) steht für die Existenz eines Laufes $p \xrightarrow{\alpha_1} p_1 \xrightarrow{\alpha_2} \dots p_{n-1} \xrightarrow{\alpha_n} p'$ ($p \dashrightarrow^{\alpha_1} p_1 \dashrightarrow^{\alpha_2} \dots p_{n-1} \dashrightarrow^{\alpha_n}$) mit $w = \alpha_1 \dots \alpha_n$.

1 Definitionen

Desweiteren soll $w|_B$ die Aktions-Sequenz bezeichnen, die man erhält, wenn man aus w alle Aktionen löscht, die nicht in $B \subseteq \Sigma$ enthalten sind. \hat{w} steht für $w|_\Sigma$. Es wird $p \xRightarrow{w} p'$ für ein $w \in \Sigma^*$ geschrieben, falls $\exists w' \in \Sigma_\tau^* : \hat{w}' = w \wedge p \xrightarrow{w'} p'$, und $p \xRightarrow{w}$, falls $p \xRightarrow{w} p'$ für ein beliebiges p' gilt. Falls $p_0 \xRightarrow{w} p$ gilt, dann wird w *Trace* genannt und p ist ein *erreichbarer Zustand*.

Analog zu \dashrightarrow und \Rightarrow werden \longrightarrow und \Longrightarrow für die entsprechenden Relationen der must-Transition verwendet.

Outputs und die interne Aktion werden *lokale Aktionen* genannt, da sie lokal vom ausführenden MEIO kontrolliert sind. Um eine Erleichterung der Notation zu erhalten, soll gelten, dass $p \not\stackrel{a}{\rightarrow}$ und $p \not\stackrel{a}{\dashrightarrow}$ für $\neg\exists p' : p \stackrel{a}{\rightarrow} p'$ und $\neg\exists p' : p \stackrel{a}{\dashrightarrow} p'$ stehen soll. $p \stackrel{a}{\rightarrow} \stackrel{\varepsilon}{\Rightarrow} p'$ wird geschrieben, wenn p'' existiert, so dass $p \stackrel{a}{\rightarrow} p'' \stackrel{\varepsilon}{\Rightarrow} p'$ gilt. Diese Transition wird auch als *schwach-nachlaufende must-Transition* bezeichnet. Entsprechend steht $\dashrightarrow \stackrel{\varepsilon}{\Rightarrow}$ für die *schwach-nachlaufende may-Transition*.

In Graphiken wird eine Aktion a als $a?$ notiert, falls $a \in I$ und $a!$, falls $a \in O$. Must-Transitionen (may-Transitionen) werden als durchgezogener Pfeil gezeichnet (gestrichelter Pfeil). Entsprechend der syntaktischen Konsistenz repräsentiert jede gezeichnete must-Transition auch gleichzeitig die zugrundeliegende may-Transitionen.

Definition 1.2 (Parallelkomposition). Zwei MEIOs $P_1 = (P_1, I_1, O_1, \longrightarrow_1, \dashrightarrow_1, p_{01}, E_1)$ und $P_2 = (P_2, I_2, O_2, \longrightarrow_2, \dashrightarrow_2, p_{02}, E_2)$ sind komponierbar, falls $O_1 \cap O_2 = \emptyset$. Für solche MEIOs ist die Parallelkomposition $P_{12} := P_1 \parallel P_2 = ((P_1 \times P_2), I, O, \longrightarrow_{12}, \dashrightarrow_{12}, (p_{01}, p_{02}), E)$ definiert mit: TODO: erzwungenen Zeilenumbrüche kontrollieren

- $I = (I_1 \cup I_2) \setminus (O_1 \cup O_2),$
- $O = (O_1 \cup O_2),$
- $\longrightarrow_{12} = \left\{ ((p_1, p_2), \alpha, (p'_1, p'_2)) \mid p_1 \xrightarrow{\alpha}_1 p'_1, \alpha \in \Sigma_\tau \setminus \text{Synch}(P_1, P_2) \right\} \\ \cup \left\{ ((p_1, p_2), \alpha, (p_1, p'_2)) \mid p_2 \xrightarrow{\alpha}_2 p'_2, \alpha \in \Sigma_\tau \setminus \text{Synch}(P_1, P_2) \right\} \\ \cup \left\{ ((p_1, p_2), \alpha, (p'_1, p'_2)) \mid p_1 \xrightarrow{\alpha}_1 p'_1, p_2 \xrightarrow{\alpha}_2 p'_2, \alpha \in \text{Synch}(P_1, P_2) \right\},$
- $\dashrightarrow_{12} = \left\{ ((p_1, p_2), \alpha, (p'_1, p'_2)) \mid p_1 \dashrightarrow_1 p'_1, \alpha \in \Sigma_\tau \setminus \text{Synch}(P_1, P_2) \right\} \\ \cup \left\{ ((p_1, p_2), \alpha, (p_1, p'_2)) \mid p_2 \dashrightarrow_2 p'_2, \alpha \in \Sigma_\tau \setminus \text{Synch}(P_1, P_2) \right\} \\ \cup \left\{ ((p_1, p_2), \alpha, (p'_1, p'_2)) \mid p_1 \dashrightarrow_1 p'_1, p_2 \dashrightarrow_2 p'_2, \alpha \in \text{Synch}(P_1, P_2) \right\},$
- $E = (P_1 \times E_2) \cup (E_1 \times P_2)$ *geerbte Fehler*
 $\left. \begin{array}{l} \cup \left\{ (p_1, p_2) \mid \exists a \in O_1 \cap I_2 : p_1 \dashrightarrow_1^a \wedge p_2 \not\stackrel{a}{\rightarrow}_2 \right\} \\ \cup \left\{ (p_1, p_2) \mid \exists a \in I_1 \cap O_2 : p_1 \not\stackrel{a}{\rightarrow}_1 \wedge p_2 \dashrightarrow_2^a \right\} \end{array} \right\}$ *neue Kommunikationsfehler.*

Dabei bezeichnet $\text{Synch}(P_1, P_2) = (I_1 \cap O_2) \cup (O_1 \cap I_2) \cup (I_1 \cap I_2)$ die Menge der zu synchronisierenden Aktionen. Die synchronisierten Aktionen werden als Inputs, in dem Fall $(I_1 \cap I_2)$, bzw. Outputs, in allen anderen Fällen, der Komposition beibehalten.

P_1 wird Partner von P_2 genannt, wenn die Parallelkomposition von P_1 und P_2 geschlossen ist. Eine Parallelkomposition von zwei MEIOs P_1 und P_2 ist geschlossen, wenn P_1 und P_2 duale Signaturen $\text{Sig}(P_1) = (I, O)$ und $\text{Sig}(P_2) = (O, I)$ haben.

Ein neuer Fehler entsteht, wenn eines der MEIOs die Möglichkeit für einen Output hat (may-Transition) und das andere MEIO den passenden Input nicht sicher stellt (keine must-Transition vorhanden). Es muss also in möglichen Implementierungen nicht wirklich zu diesem Fehler kommen, da die Output-Transition nicht zwingendermaßen implementiert werden muss und die may-Input-Transition trotzdem erlaubt sein kann. Wie bereits in [Sch16] kann es durch die Synchronisation von Inputs zu keinen neuen Fehler kommen, da die Inputs in beiden Transitionssystemen keine lokal kontrollierten Aktionen sind. Falls jedoch nur eines der Transitionssysteme die Möglichkeit für einen Input hat, der synchronisiert wird, besteht diese Möglichkeit in der Parallelkomposition nicht mehr. Es kann also in der Kommunikation mit einem weiteren MEIO dort zu einem neuen Fehler kommen.

Definition 1.3 (alternierende Simulation). Eine (starke) alternierende Simulation ist eine Relation $\mathcal{R} \subseteq P \times Q$ auf zwei MEIOs P und Q , wenn für alle $(p, q) \in \mathcal{R}$ mit $q \notin E_Q$ und $\alpha \in \Sigma_\tau$ gilt:

1. $p \notin E_P$,
2. $q \xrightarrow{\alpha}_Q q'$ impliziert $p \xrightarrow{\alpha}_P p'$ für ein p' mit $p' \mathcal{R} q'$,
3. $p \xrightarrow{\alpha}_P p'$ impliziert $q \xrightarrow{\alpha}_Q q'$ für ein q' mit $p' \mathcal{R} q'$.

Die Vereinigung \sqsubseteq_{as} aller dieser Relationen wird als (starke) as-Verfeinerung(-s Relation) (auch modal Verfeinerung) bezeichnet. Es wird $P \sqsubseteq_{\text{as}} Q$ geschrieben, falls $p_0 \sqsubseteq_{\text{as}} q_0$ gilt. $P \sqsubseteq_{\text{as}} Q$ steht dabei dafür, dass P Q (stark) as-verfeinert oder dass P eine (starke) as-Verfeinerung von Q ist.

Für ein MEIO Q und eine Implementierung P mit $P \sqsubseteq_{\text{as}} Q$, ist P eine as-Implementierung von Q und es wird $\text{as-impl}(Q)$ für die Menge aller as-Implementierungen von Q verwendet.

Da für zwei MEIOs P und Q und alle möglichen Zustands-Tupel (p, q) in einer alternierenden Simulationen \mathcal{R} gelten muss, dass aus $q \notin E_Q$ folgt, dass auch p nicht in E_P enthalten ist, gilt auch die Implikation $p \in E_P \Rightarrow q \in E_Q$.

Definition 1.4 (schwache Simulation). Eine schwache alternierende Simulation ist eine Relation $\mathcal{R} \subseteq P \times Q$ auf zwei MEIOs P und Q , wenn für alle $(p, q) \in \mathcal{R}$ mit $q \notin E_Q$ und $i \in I$ und $\omega \in O \cup \{\tau\}$ gilt:

1. $p \notin E_P$,
2. $q \xrightarrow{i}_Q q'$ impliziert $p \xrightarrow{i}_P \xRightarrow{\varepsilon}_P p'$ für ein p' mit $p' \mathcal{R} q'$,
3. $q \xrightarrow{\omega}_Q q'$ impliziert $p \xRightarrow{\hat{\omega}}_P p'$ für ein p' mit $p' \mathcal{R} q'$,
4. $p \xrightarrow{i}_P p'$ impliziert $q \xrightarrow{i}_Q \xRightarrow{\varepsilon}_Q q'$ für ein q' mit $p' \mathcal{R} q'$,

5. $p \xrightarrow{\omega}_P p'$ impliziert $q \xRightarrow{\omega}_Q q'$ für ein q' mit $p' \mathcal{R} q'$.

Analog zur starken alternierenden Simulation, wird hier \sqsubseteq_{w-as} als Relationssymbol verwendet und man kann auch entsprechend schwache as-Verfeinerung betrachten.

Ebenso kann \sqsubseteq_{w-as} für ein MEIO Q und eine Implementierung P definiert werden mit $P \sqsubseteq_{w-as} Q$, ist P eine w-as-Implementierung von Q und es wird $w-as-impl(Q)$ für die Menge aller w-as-Implementierungen von Q verwendet.

Die schwache Simulation erlaubt interne Aktionen beim MEIO, das die entsprechende Aktion matchen muss. Jedoch ist es zwingen notwendig, dass ein Input sofort aufgeführt wird und erst dann interne Aktinen möglich sind. Da ein Input die Reaktion auf eine Aktion ist, die die Umwelt auslöst und die nicht auf das Transitionssystem warten kann. Outputs hingegen können auch verzögert werden, da die Umgebung dies dann als Inputs aufnimmt und für diese somit nicht lokal kontrolliert ist.

Auch für alle Tupel (p, q) in einer schwach alternierenden Simulations-Relation \mathcal{R} gilt $p \in E_P \Rightarrow q \in E_Q$.

Die Parallelkomposition von Wörtern und Mengen kann z.B. aus [BV15a] übernommen werden.

Definition 1.5 (Parallelkomposition auf Traces).

- Für zwei Wörter $w_1 \in \Sigma_1$ und $w_2 \in \Sigma_2$ ist deren Parallelkomposition definiert als: $w_1 \| w_2 := \{w \in (\Sigma_1 \cup \Sigma_2)^* \mid w|_{\Sigma_1} = w_1 \wedge w|_{\Sigma_2} = w_2\}$.
- Für zwei Mengen von Wörtern bzw. Sprachen $W_1 \subseteq \Sigma_1^*$ und $W_2 \subseteq \Sigma_2^*$ ist deren Parallelkomposition definiert als: $W_1 \| W_2 := \bigcup \{w_1 \| w_2 \mid w_1 \in W_1 \wedge w_2 \in W_2\}$.

Ebenso können die Definitionen der Funktionen `prune` und `cont` zum Abschneiden und Verlängern von Traces aus [Sch16] übernommen werden. Hierbei ist zu beachten, dass in dieser Arbeit ε das leere Wort und $\mathfrak{P}(M)$ die Potenzmenge der Menge M bezeichnet.

Definition 1.6 (Pruning- und Fortsetzungs-Funktion).

- $\text{prune} : \Sigma^* \rightarrow \Sigma^*, w \mapsto u$, mit $w = uv, u = \varepsilon \vee u \in \Sigma^* \cdot I$ und $v \in O^*$,
- $\text{cont} : \Sigma^* \rightarrow \mathfrak{P}(\Sigma^*), w \mapsto \{wu \mid u \in \Sigma^*\}$,
- $\text{cont} : \mathfrak{P}(\Sigma^*) \rightarrow \mathfrak{P}(\Sigma^*), L \mapsto \bigcup \{\text{cont}(w) \mid w \in L\}$.

Definition 1.7 (Sprache). Die Sprache eines MEIOs P ist $L(P) := \{w \in \Sigma^* \mid p_0 \xRightarrow{w}_P\}$.

Somit entspricht die Sprache eines MEIOs, das eine Implementierung ist der Definition aus [Sch16] für EIOs. Jedoch muss die Sprache einer as-Verfeinerung eine MEIOs nicht mehr Teilmenge der Sprache des MEIOs sein, da Definition 1.3 beliebiges Verhalten nach einem Fehler-Zustand in dem zu verfeinernden MEIO zulässt. Falls jedoch der MEIO bereits fehler frei ist, ist seine Sprache die Vereinigung der Sprachen all seine möglichen as-Implementierungen.

Von der Sprache einer as-Verfeinerung eines MEIOs kann man keine Rückschlüsse mehr auf die ursprüngliche Sprache ziehen, da man nicht weiß, welche Fehler-Zustände in der Verfeinerung übernommen wurden und welche als normale Zustände mit beliebigen Verhalten umgesetzt wurden. Es kann also im Allgemeinen kein Zusammenhang zwischen den Sprachen der as-Implementierungen eines MEIOs und der Sprache des MEIOs hergestellt werden.

Man hätte alternativ der Sprache eines MEIOs anders definieren können um weiterhin einen Zusammenhang zwischen dieser und den Sprachen der as-Implementierungen zu haben, jedoch wäre dann die Äquivalenz zur EIO Sprach-Definition in [Sch16] verloren gegangen. Eine Implementierung mit Fehler-Zuständen hätte dann auch eine Sprache mit Wörtern, die sie nicht mal ausführen können muss.

2 allgemeine Folgerungen

Proposition 2.1 (*Sprache und Implementierungen*). *Für die Sprache eines MEIOs P gilt $L(P) \subseteq \left\{ w \in \Sigma^* \mid \exists P' \in \text{as-impl}(P) : p'_0 \xRightarrow{w}_{P'} \right\} = \bigcup_{P' \in \text{as-impl}(P)} L(P').$*

Beweis.

Sei P' die as-Implementierung von P , die alle may- und must-Transitionen von P implementiert. Die entsprechende starke as-Verfeinerungs-Relation, die zwischen P' und P gilt, ist die Identitätsabbildung zwischen den Zuständen der Transitionssysteme. Die Definition von P' lautet dann:

- $P' = P$ für die Menge der Zustände,
- $p'_0 = p_0$,
- $I_{P'} = I_P$ und $O_{P'} = O_P$,
- $\longrightarrow_{P'} = \dashrightarrow_{P'} = \dashrightarrow_P$,
- $E'_{P'} = \emptyset$.

Für alle $p \in P$, die auch von nicht Fehler-Zuständen aus erreichbar sind, muss $p \sqsubseteq_{\text{as}} p'$ wegen 1.3 3. für entsprechenden p' in P' , die durch die analogen Transitionen erreichbar sind. Diese Tupel sind bereits in der Identitätsabbildung, die als as-Verfeinerungs-Relation vorausgesetzt wurde, enthalten. Da P' alle Transitionen von P implementiert, gilt 1.3 2. bereits durch die Tupel, die durch die Identitätsabbildung in \sqsubseteq_{as} enthalten sein müssen. Der erste Punkt von 1.3 gilt, da E'_P leer ist. Für alle $w \in L(P) = \left\{ w \in \Sigma^* \mid p_0 \xRightarrow{w}_P \right\}$ folgt nun $w \in L(P') = \left\{ w \in \Sigma^* \mid p'_0 \xRightarrow{w}_{P'} \right\}$, da alle Transitionen von P in P' implementiert werden. \square

Proposition 2.2 (*Sprache der Parallelkomposition*). *Für zwei komponierbare MEIOs P_1 und P_2 gilt: $L_{12} := L(P_{12}) = L_1 \parallel L_2$.*

Beweis. Jedes Wort, dass in L_{12} enthalten ist, hat einen einen entsprechenden Ablauf, der in P_{12} ausführbar ist. Dieser Ablauf kann auf Abläufe von P_1 und P_2 projiziert werden und die Projektionen sind dann in L_1 und L_2 enthalten.

In einer Parallelkomposition werden die Wörter der beiden MEIOs gemeinsam ausgeführt, falls es sich um synchronisierte Aktionen handelt, und verschränkt sequenziell, wenn es sich um unsynchronisierte Aktionen handelt. Somit sind alle Wörter aus $L_1 \parallel L_2$ auch Wörter der Parallelkomposition $L(P_{12})$. \square

Lemma 2.3 (*w-as-Verfeinerung und Parallelkomposition*). *Für zwei komponierbar MEIOs P_1 und P_2 gilt, falls P'_1 und P'_2 schwache as-Verfeinerungen von P_1 bzw. P_2 sind, dann ist auch $P'_1 \parallel P'_2$ eine schwache as-Verfeinerung von $P_1 \parallel P_2$.*

Beweis. Es gelte $j \in \{1, 2\}$. Da P'_j eine schwache as-Verfeinerung von P_j ist, gibt es nach Definition 1.4 eine schwache as-Verfeinerungs-Relation \mathcal{R}_j , die beschreibt, wie P'_j P_j verfeinert. Die Parallelkomposition werden auf Basis von Definition 1.2 gebildet. Die Zustände sind also Tupel der Zustände der Komponenten. In dem man aus den Zuständen, die die \mathcal{R}_j in Relation setzt auch solche Tupel zusammensetzt, kann man eine neue schwache as-Verfeinerungs-Relation für die Verfeinerung von $P_1 \parallel P_2$ durch $P'_1 \parallel P'_2$ erstellen. Die neue schwache as-Verfeinerungs-Relation soll \mathcal{R}_{12} heißen und wie folgt definiert sein: $\forall p'_1, p'_2, p_1, p_2 : ((p'_1, p'_2), (p_1, p_2)) \in \mathcal{R}_{12} \Leftrightarrow (p'_1, p_1) \in \mathcal{R}_1 \wedge (p'_2, p_2) \in \mathcal{R}_2$. Es bleibt nun zu zeigen, dass \mathcal{R}_{12} eine zulässige schwache as-Verfeinerungs-Relation nach Definition 1.4 ist. Für alle folgenden Fälle wird $((p'_1, p'_2), (p_1, p_2)) \in \mathcal{R}_{12}$ mit $(p_1, p_2) \notin E_{12}$ vorausgesetzt.

1. Für den ersten Punkt ist zu zeigen, dass (p'_1, p'_2) kein Element von $E_{P'_1 \parallel P'_2}$ ist. Dies folgt direkt aus der Voraussetzung, dass $(p_1, p_2) \notin E_{P_1 \parallel P_2}$ für das Tupel $((p'_1, p'_2), (p_1, p_2))$ aus \mathcal{R}_{12} gilt. In dem man auf das \mathcal{R}_{12} die Definition anwendet, erhält man $(p'_j, p_j) \in \mathcal{R}_j$ für beide j Werte. Die p_j dürfen beide keine Fehler-Zustände sein, da sonst auch (p_1, p_2) ein solcher wäre. Somit folgt mit Definition 1.4 1. $p'_j \notin E_j$ für beide j Werte. Die beiden gestrichenen Zustände in Parallelkomposition können also keinen geerbten Fehler produzieren. Jedoch könnte (p'_1, p'_2) aufgrund eines nicht erzwungenen Inputs ein neuer Fehler-Zustand sein. Dafür müsste oBdA $p'_1 \not\rightarrow_{P'_1}^q$ und $p'_2 \not\rightarrow_{P'_2}^a$ für ein a aus $I_1 \cap O_2$ gelten. \mathcal{R}_2 erzwingt mit 1.4 5. die schwache Ausführbarkeit des Outputs a in P_2 , d.h. $p_2 \Rightarrow^a_2$. Mit 1.4 2. von \mathcal{R}_1 folgt $p_1 \rightarrow^q_1$. Somit müsste auch $(p_1, p_2) \in E_{12}$ gelten, was ein Widerspruch zur Voraussetzung wäre. (p'_1, p'_2) kann also weder ein geerbter noch ein neuer Fehler-Zustand in $P'_1 \parallel P'_2$ sein und deshalb gilt $(p'_1, p'_2) \notin E_{P'_1 \parallel P'_2}$.
2. Aus der schwache Simulations-Definition 1.4 folgt, dass für diesen Punkt folgendes zu zeigen ist: $(p_1, p_2) \xrightarrow{i}_{12} (q_1, q_2)$ impliziert $(p'_1, p'_2) \xrightarrow{i}_{P'_1 \parallel P'_2} \xRightarrow{\varepsilon}_{P'_1 \parallel P'_2} (q'_1, q'_2)$ für ein (q'_1, q'_2) mit $((q'_1, q'_2), (q_1, q_2)) \in \mathcal{R}_{12}$. Die i -must-Transition in $P_1 \parallel P_2$ kann entweder aus der Synchronisation von zwei must-Inputs entstanden sein oder als unsynchronisierte Aktion aus einem P_1 übernommen worden sein.

- Fall 1 ($i \notin \text{Synch}(P_1 \parallel P_2)$): OBdA ist i in I_1 enthalten. Es muss also in P_1 die i -Transition als must-Transition von p_1 ausgehen, es gilt $p_1 \xrightarrow{i}_1 q_1$. $p_2 = q_2$ muss gelten, da i nicht in Σ_2 enthalten ist. Mit der Relation \mathcal{R}_1 und 1.4 2. folgt, dass in P'_1 i als schwache Transition in der Form $p'_1 \xrightarrow{i}_{P'_1} \xRightarrow{\varepsilon}_{P'_1} q'_1$ ausführbar sein muss und $q'_1 \mathcal{R}_1 q_1$ gelten muss. Aus der Voraussetzung folgt $(p'_2, p_2) = (q'_2, q_2) \in \mathcal{R}_2$, da i wenn es kein Element der Aktionen von P_2 ist auch keine Aktion der schwachen as-Verfeinerung P'_2 sein kann. Mit der

2 allgemeine Folgerungen

Definition von \mathcal{R}_{12} kann dann daraus $((q'_1, q'_2), (q_1, q_2)) \in \mathcal{R}_{12}$ gefolgert werden. In der Parallelkomposition von P'_1 und P'_2 entsteht die Transitionsfolge $(p'_1, p'_2) \xrightarrow{i} P'_1 \| P'_2 \xRightarrow{\varepsilon} P'_1 \| P'_2 (q'_1, q'_2)$.

- Fall 2 ($i \in \text{Synch}(P_1 \| P_2)$): Damit i auch in $P_1 \| P_2$ ein Input ist, muss $i \in I_1 \cap I_2$ gelten. Um die Transition $(p_1, p_2) \xrightarrow{i} P_1 \| P_2 (q_1, q_2)$ in der Komposition möglich zu machen, muss in beiden P_j 's $p_j \xrightarrow{i} P_j q_j$ gelten. Durch \mathcal{R}_j und die Definition 1.4 2., die für diese Relationen gilt, folgt für beide j Werte $p'_j \xrightarrow{i} P'_j \xRightarrow{\varepsilon} P'_j q'_j$ mit $(q'_j, q_j) \in \mathcal{R}_j$. Es folgt $((q'_1, q'_2), (q_1, q_2)) \in \mathcal{R}_{12}$ mit der Definition von \mathcal{R}_{12} . Durch die Synchronisation des i 's in der Komposition von P'_1 und P'_2 gilt $(p'_1, p'_2) \xrightarrow{i} P'_1 \| P'_2 \xRightarrow{\varepsilon} P'_1 \| P'_2 (q'_1, q'_2)$.
3. Analog zu 2. kann für diesen Punkt $(p_1, p_2) \xrightarrow{\omega} P_1 \| P_2 (q_1, q_2)$ impliziert $(p'_1, p'_2) \xRightarrow{\hat{\omega}} P'_1 \| P'_2 (q'_1, q'_2)$ für ein (q'_1, q'_2) mit $((q'_1, q'_2), (q_1, q_2)) \in \mathcal{R}_{12}$ gezeigt werden. Die ω Transition in $P_1 \| P_2$ ist entweder aus einem synchronisierten oder aus einem unsynchronisierten ω entstanden.

- Fall 1 ($\omega \notin \text{Synch}(P_1 \| P_2)$): OBdA ist ω in $O_1 \cup \{\tau\}$ enthalten. Um in $P_1 \| P_2$ die must-Transition zu erhalten muss bereits für die Transition in P_1 $p_1 \xrightarrow{\omega} P_1 q_1$ gelten. Mit 1.4 3. kann für \mathcal{R}_1 gefolgert werden, dass $p'_1 \xRightarrow{\hat{\omega}} P'_1 q'_1$ mit $(q'_1, q_1) \in \mathcal{R}_1$ gilt. In der Komposition folgt dann $(p'_1, p'_2) \xRightarrow{\hat{\omega}} P'_1 \| P'_2 (q'_1, q'_2)$, da $\omega \notin \Sigma_2$ ist und somit $(p'_2, p_2) = (q'_2, q_2) \in \mathcal{R}_2$ gilt. Es folgt insgesamt auch noch die Zugehörigkeit des Zustands-Tupels $((q'_1, q'_2), (q_1, q_2))$ zur schwachen as-Verfeinerungs-Relation \mathcal{R}_{12} .
 - Fall 2 ($\omega \in \text{Synch}(P_1 \| P_2)$): Da in der Menge $\text{Synch}(P_1 \| P_2)$ nur Inputs und Outputs enthalten sein kann, muss in diesem Fall $\omega \neq \tau$ gelten. Um einen Output ω in der Parallelkomposition von P_1 und P_2 zu erhalten, muss oBdA $\omega \in I_1 \cap O_2$ gelten. Es folgt also $p_1 \xrightarrow{\omega} P_1 q_1$ mit $\omega \in I_1$ und $p_2 \xrightarrow{\omega} P_2 q_2$ mit $\omega \in O_2$ für die einzelnen Transitionssysteme. Mit \mathcal{R}_1 und 1.4 2. folgt $p'_1 \xrightarrow{\omega} P'_1 \xRightarrow{\varepsilon} P'_1 q'_1$ und $q'_1 \mathcal{R}_1 q_1$. Wenn man \mathcal{R}_2 mit 1.4 3. angewendet erhält man $p'_2 \xRightarrow{\omega} P'_2 q'_2$ mit $q'_2 \mathcal{R}_2 q_2$. Da ω in P'_2 ein Output ist, gilt $\omega = \hat{\omega}$. In der Parallelkomposition von P'_1 und P'_2 werden zuerst die internen Aktionen von P'_2 ausgeführt, bis dort der Output erreicht ist, dann wird ω synchronisiert und danach werden die internen Aktionen beider Komponenten ausgeführt, bis man bei den Zuständen q'_1 und q'_2 angekommen ist. Es folgt also die Transitionsfolge $(p'_1, p'_2) \xRightarrow{\hat{\omega}} P'_1 \| P'_2 (q'_1, q'_2)$ und das Tupel $((q'_1, q'_2), (q_1, q_2))$ in der Relation \mathcal{R}_{12} .
4. $(p'_1, p'_2) \xrightarrow{i} P'_1 \| P'_2 (q'_1, q'_2)$ impliziert $(p_1, p_2) \xrightarrow{i} P_1 \| P_2 \xRightarrow{\varepsilon} P_1 \| P_2 (q_1, q_2)$ für ein (q_1, q_2) mit $((q'_1, q'_2), (q_1, q_2)) \in \mathcal{R}_{12}$ ist die Voraussetzung des 4. Punktes, um zu beweisen, dass \mathcal{R}_{12} eine schwache as-Verfeinerungs-Relation ist.

Die Transition i kann wiederum durch Synchronisation von zwei Transitionen entstanden sein oder durch eine Transition aus einer Komponenten und $i \notin \text{Synch}(P'_1 \| P'_2)$.

2 allgemeine Folgerungen

- Fall 1 ($i \notin \text{Synch}(P'_1 \| P'_2)$): OBdA ist i in I_1 enthalten. Es muss also in P'_1 eine ausgehende i -Transition von Zustand p'_1 geben, so dass $p'_1 \xrightarrow{i} q'_1$ gilt. $p_2 = q_2$ muss gelten, da i nicht in Σ_2 enthalten ist. Mit der Relation \mathcal{R}_1 und 1.4 4. folgt, dass in P_1 i als schwache Transition in der Form $p_1 \xrightarrow{i} q_1$ ausführbar sein muss und $q'_1 \mathcal{R}_1 q_1$ gelten muss. Aus der Voraussetzung folgt $(p'_2, p_2) = (q'_2, q_2) \in \mathcal{R}_2$, da i wenn es kein Element der Aktionen von P_2 ist auch keine Aktion der schwachen as-Verfeinerung P'_2 sein kann. Mit der Definition von \mathcal{R}_{12} kann dann daraus $((q'_1, q'_2), (q_1, q_2)) \in \mathcal{R}_{12}$ gefolgert werden. In der Parallelkomposition von P_1 und P_2 entsteht die Transitionsfolge $(p_1, p_2) \xrightarrow{i} q_1, q_2$.
 - Fall 2 ($i \in \text{Synch}(P'_1 \| P'_2)$): Damit i auch in $P'_1 \| P'_2$ ein Input ist, muss $i \in I_1 \cap I_2$ gelten. Um die Transition $(p'_1, p'_2) \xrightarrow{i} (q'_1, q'_2)$ in der Komposition möglich zu machen, muss in beiden P'_j 's $p_j \xrightarrow{i} q'_j$ gelten. Durch \mathcal{R}_j und die Definition 1.4 4., die für diese Relationen gilt, folgt für beide j Werte $p_j \xrightarrow{i} q_j$ mit $(q'_j, q_j) \in \mathcal{R}_j$. Es folgt $((q'_1, q'_2), (q_1, q_2)) \in \mathcal{R}_{12}$ mit der Definition von \mathcal{R}_{12} . Durch die Synchronisation des i 's in der Komposition von P_1 und P_2 gilt $(p_1, p_2) \xrightarrow{i} q_1, q_2$.
5. Analog zu 3. und 4. kann für diesen Punkt $(p'_1, p'_2) \xrightarrow{\omega} (q'_1, q'_2)$ impliziert $(p_1, p_2) \xrightarrow{\omega} q_1, q_2$ für ein (q_1, q_2) mit $((q'_1, q'_2), (q_1, q_2)) \in \mathcal{R}_{12}$ gezeigt werden. Die ω Transition in $P'_1 \| P'_2$ ist entweder aus einem synchronisierten oder aus einem unsynchronisierten ω entstanden.
- Fall 1 ($\omega \notin \text{Synch}(P'_1 \| P'_2)$): OBdA ist ω in $O_1 \cup \{\tau\}$ enthalten. Um in $P'_1 \| P'_2$ die may-Transition zu erhalten muss bereits in P'_1 die Transition $p'_1 \xrightarrow{\omega} q'_1$ möglich gewesen sein. Mit 1.4 5. kann für \mathcal{R}_1 gefolgert werden, dass $p_1 \xrightarrow{\omega} q_1$ mit $(q'_1, q_1) \in \mathcal{R}_1$ gilt. In der Komposition folgt dann $(p_1, p_2) \xrightarrow{\omega} q_1, q_2$, da $\omega \notin \Sigma_2$ ist und somit $(p'_2, p_2) = (q'_2, q_2) \in \mathcal{R}_2$ gilt. Es folgt insgesamt auch noch die Zugehörigkeit des Zustands-Tupels $((q'_1, q'_2), (q_1, q_2))$ zur schwachen as-Verfeinerungs-Relation \mathcal{R}_{12} .
 - Fall 2 ($\omega \in \text{Synch}(P'_1 \| P'_2)$): Es muss $\omega \neq \tau$ gelten und somit muss oBdA $\omega \in I_1 \cap O_2$ gelten. Es folgt also $p'_1 \xrightarrow{\omega} q'_1$ mit $\omega \in I_1$ und $p'_2 \xrightarrow{\omega} q'_2$ mit $\omega \in O_2$ für die einzelnen Transitionssysteme. Mit \mathcal{R}_1 und 1.4 4. folgt $p_1 \xrightarrow{\omega} q_1$ und $q'_1 \mathcal{R}_1 q_1$. Wenn man \mathcal{R}_2 mit 1.4 5. angewendet erhält man $p_2 \xrightarrow{\omega} q_2$ mit $q'_2 \mathcal{R}_2 q_2$. Da ω in P_2 ein Output ist, gilt $\omega = \hat{\omega}$. In der Parallelkomposition von P_1 und P_2 werden zuerst die internen Aktionen von P_2 ausgeführt, bis dort der Output erreicht ist, dann wird ω synchronisiert und danach werden die internen Aktionen beider Komponenten ausgeführt, bis man bei den Zuständen q_1 und q_2 angekommen ist. Es folgt also die Transitionsfolge $(p_1, p_2) \xrightarrow{\omega} q_1, q_2$ und das Tupel $((q'_1, q'_2), (q_1, q_2))$ in der Relation \mathcal{R}_{12} .

□

Korollar 2.4 (*w-as-Implementierungen und Parallelkomposition*). Für zwei komponierbare MEIOs P_1 und P_2 gilt: $P'_1 \in \text{w-as-impl}(P_1) \wedge P'_2 \in \text{w-as-impl}(P_2) \Rightarrow (P'_1 \parallel P'_2) \in \text{w-as-impl}(P_1 \parallel P_2)$.

Beweis. P'_1 und P'_2 sind aufgrund der Definition 1.4 auch schwache as-Verfeinerungen von P_1 bzw. P_2 . Somit ist die Parallelkomposition $P'_1 \parallel P'_2$ auch eine schwache as-Verfeinerung von $P_1 \parallel P_2$, wegen Lemma 2.3. Für Implementierungen gilt $\rightarrow = \dashv\dashv$. Durch die Definition der Parallelkomposition in 1.2 können aus zwei komponierbaren Implementierungen in der Komposition keine may-Transitionen ohne zugehörige must-Transitionen entstehen. Es gilt also auch $\rightarrow_{P'_1 \parallel P'_2} = \dashv\dashv_{P'_1 \parallel P'_2}$ und somit ist $P'_1 \parallel P'_2$ eine Implementierung und eine as-Verfeinerung von $P_1 \parallel P_2$. Dies entspricht der Definition der schwachen as-Implementierung, sodass $(P'_1 \parallel P'_2) \in \text{w-as-impl}(P_1 \parallel P_2)$ gilt. □

Korollar 2.5 (*as-Verfeinerungen und Parallelkomposition*). Für zwei komponierbar MEIOs P_1 und P_2 gilt, falls P'_1 und P'_2 as-Verfeinerungen von P_1 bzw. P_2 sind, dann ist auch $P'_1 \parallel P'_2$ eine as-Verfeinerung von $P_1 \parallel P_2$.

Beweis. Falls die Relationen \mathcal{R}_1 und \mathcal{R}_2 aus dem Beweis von Lemma 2.3 keine schwachen as-Verfeinerungs-Relationen sondern as-Verfeinerungs-Relation sind, ist auch \mathcal{R}_{12} eine as-Verfeinerungs-Relation zwischen $P'_1 \parallel P'_2$ und $P_1 \parallel P_2$. Es ist also nur zu zeigen, wie aus den einzeln Beweispunkten des Beweises von 2.3 folgt, dass \mathcal{R}_{12} eine as-Verfeinerungs-Relation ist. Es gilt hier ebenso für alle Punkte die Voraussetzung $((p'_1, p'_2), (p_1, p_2)) \in \mathcal{R}_{12}$ mit $(p_1, p_2) \notin E_{12}$.

1. Dieser Punkt kann genauso wie 1. aus dem Beweis von Lemma 2.3 bewiesen werden. Nur für p_2 ist der Output a nicht nur schwach ausführbar, sondern direkt.
2. Da α kann sowohl Input, Output wie auch internen Aktion sein. Um diesen Punkt der zu beweisen muss man die Beweise von 2. und 3. aus dem Beweis von Lemma 2.3 kombinieren. Da die \mathcal{R}_j für $j \in \{1, 2\}$ jedoch die Transition in den P'_j ohne zusätzliche τ -Transitionen fordern, entstehen in den einzelnen Komponenten keine schwachen Transitionen für die α s und somit ist α auch in der Parallelkomposition $P'_1 \parallel P'_2$ eine direkte Transition ohne zusätzliche τ s. Es folgt also das zu zeigende für diesen die strikte as-Verfeinerungsrelation für diesen Punkt.
3. Hierfür werden die Punkte 3. und 4. aus dem Beweis des Lemmas 2.3 kombiniert. Analog wie bei 2. diese Beweises fallen die zusätzlichen τ -Transitionen durch die stärkere Forderung an \mathcal{R}_1 und \mathcal{R}_2 weg. Dieser Punkt gilt also ebenfalls.

□

Korollar 2.6 (*as-Implementierungen und Parallelkomposition*). Für zwei komponierbare MEIOs P_1 und P_2 gilt: $P'_1 \in \text{as-impl}(P_1) \wedge P'_2 \in \text{as-impl}(P_2) \Rightarrow (P'_1 \parallel P'_2) \in \text{as-impl}(P_1 \parallel P_2)$.

2 allgemeine Folgerungen

Beweis. Analog zum Beweis des Korollars 2.4 kann hier auch begründet werden, dass $P_1 \parallel P'_2$ eine Implementierung ist. Zusätzlich ist $P'_1 \parallel P'_2$ eine as-Verfeinerung, wegen der Voraussetzungen diese Korollars in Kombination mit der Aussage des Korollars 2.5. Es gilt also $(P'_1 \parallel P'_2) \in \text{as-impl}(P_1 \parallel P_2)$. \square

Die entgegengesetzte Richtung von Korollar 2.5 gilt im allgemeinen nicht, d.h. es muss zu einer as-Verfeinerung P' einer Parallelkomposition $P_1 \parallel P_2$ keine as-Verfeinerungen P'_1 bzw. P'_2 der einzelnen Komponenten P_1 bzw. P_2 geben, deren Parallelkomposition $P'_1 \parallel P'_2$ der as-Verfeinerung der Parallelkomposition P' entsprechen. Die Problematik wird in Abbildung 2.1 an einem Beispiel dargestellt. In der Parallelkomposition wird die may-Transition von P_2 zu zwei may-Transitionen, für die in einer as-Verfeinerung unabhängig entschieden werden kann, ob sie übernommen, implementiert oder weggelassen werden. Somit kommt es in P' zu dem Problem, dass keine as-Verfeinerung von P_2 (entweder keine Transition oder die o' Transition wird als may- oder must-Transition ausgeführt) in Parallelkomposition mit der Implementierung P_1 P' ergeben würde.

Da jede as-Verfeinerungs-Relation auch eine schwache as-Verfeinerungs-Relation ist, folgt draus auch, dass die entgegengesetzte Richtung von Lemma 2.3 ebenfalls nicht gelten kann. Auch im Spezialfall von as-Implementierungen bzw. w-as-Implementierungen kann das Gegenbeispiel angewendet werden, da P' auch eine Implementierung von $P_1 \parallel P_2$ ist und es auch keine passende as-Implementierung bzw. w-as-Implementierung von P_2 geben kann, wenn es schon keine passende Verfeinerung gibt.

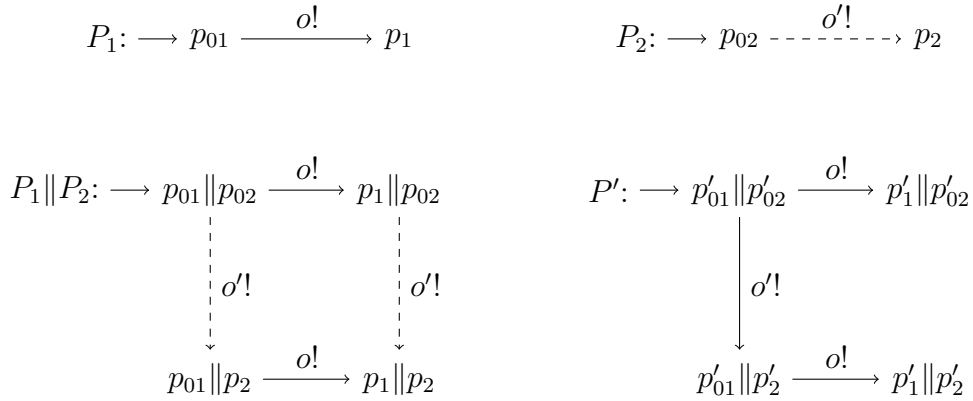


Abbildung 2.1: Gegenbeispiel für Umkehrung von Lemma 2.5

Ein neuer Fehler in einer Parallelkomposition zweier MEIOs muss in einer Implementierung (as oder w-as) dieser Parallelkomposition nicht auftauchen, auch nicht in der Parallelkomposition von Implementierungen der einzelnen Komponenten. Dies liegt daran, dass für den Input nur gesagt wird, dass keine must-Transition für die Synchronisation der Aktion vorhanden ist. Es kann trotzdem eine may-Transition für den Input geben, die auch implementiert werden kann. Falls es aber in der Parallelkomposition

2 allgemeine Folgerungen

zweier MEIO zu einem neuen Fehler kommt, dann gibt es auch immer mindestens eine Implementierung, die diesen Fehler enthält und es gibt auch immer mindestens ein Implementierungs-Paar der Komponenten, in deren Parallelkomposition sich dieser Fehler ebenfalls zeigt.

3 Verfeinerungen für Kommunikationsfehler-Freiheit

3.1 Erweiterungs-Ansatz

Dieses Kapitel versucht die Präkongruenz für Error bei EIOs aus [Sch16] auf die hier betrachten MEIOs zu erweitern.

Definition 3.1 (fehler-freie Kommunikation). *Ein Fehler-Zustand ist lokal erreichbar in einem MEIO P , wenn ein $w \in O^*$ existiert mit $p_0 \xRightarrow{w}_P p \in E$. Zwei MEIOs P_1 und P_2 kommunizieren fehler-frei, wenn keine as-Implementierungen ihrer Parallelkomposition P_{12} einen Fehler-Zustände lokal erreichen kann.*

Definition 3.2 (Kommunikationsfehler-Verfeinerungs-Basirelation). *Für zwei MEIOs P_1 und P_2 mit der gleichen Signatur wird $P_1 \sqsubseteq_E^B P_2$ geschrieben, wenn nur dann ein Fehler-Zustand in einer as-Implementierung von P_1 lokal erreichbar ist, wenn es auch eine as-Implementierung von P_2 gibt, in der ein Fehler-Zustand auch lokal erreichbar ist. Die Basisrelation stellt eine Verfeinerung-Relation bezüglich Kommunikationsfehler-Freiheit dar.*

\sqsubseteq_E^C bezeichnet die vollständig abstrakte Präkongruenz von \sqsubseteq_E^B bezüglich $\cdot\|\cdot$, d.h. die größte Präkongruenz bezüglich $\cdot\|\cdot$, die in \sqsubseteq_E^B enthalten ist.

Für as-Implementierungen P_1 und P_2 entspricht \sqsubseteq_E^B der Relation \sqsubseteq_E^B aus [Sch16].

Wie in [Sch16] werden die Fehler hier Trace-basiert betrachtet.

Definition 3.3 (Kommunikationsfehler-Traces). *Für ein MEIO P wird definiert:*

- strikte Fehler-Traces: $StET(P) := \{w \in \Sigma^* \mid p_0 \xRightarrow{w}_P p \in E\}$,
- gekürzte Fehler-Traces: $PrET(P) := \{\text{prune}(w) \mid w \in StET(P)\}$,
- Input-kritische-Traces: $MIT(P) := \{wa \in \Sigma^* \mid p_0 \xRightarrow{w}_P p \wedge a \in I \wedge p \not\xrightarrow{a}_P\}$.

Da die Basisrelation über as-Implementierungen spricht, ist es wichtig bereits in den Trace-Mengen eine Beziehung zwischen der allgemeinen Definition für MEIOs und deren as-Implementierungen herzustellen. Deshalb wird in der folgenden Proposition eine alternative Sichtweise auf die Trace-Definitionen dargestellt. Die Traces eines MEIOs entsprechen somit der Vereinigung der Traces aller seiner as-Implementierungen.

Proposition 3.4 (Kommunikationsfehler-Traces und Implementierungen).

Sei P ein MEIO.

1. Für die strikten Fehler-Traces von P gilt: $StET(P) \subseteq \{w \in \Sigma^* \mid \exists P' \in \text{as-impl}(P) : p'_0 \xRightarrow{w}_{P'} p' \in E\} = \bigcup_{P' \in \text{as-impl}(P)} StET(P').$ TODO: erzwungen Zeilenumbruch kontrollieren
2. Für die gekürzten Fehler-Traces von P gilt: $PrET(P) \subseteq \{\text{prune}(w) \mid \exists P' \in \text{as-impl}(P) : w \in StET(P')\} = \bigcup_{P' \in \text{as-impl}(P)} PrET(P').$ TODO: erzwungen Zeilenumbruch kontrollieren
3. Für Input-kritischen-Traces von P gilt: $MIT(P) \subseteq \{wa \in \Sigma^* \mid \exists P' \in \text{as-impl}(P) : p'_0 \xRightarrow{w}_{P'} p' \wedge a \in I \wedge p' \not\xrightarrow{a}_{P'}\} = \bigcup_{P' \in \text{as-impl}(P)} MIT(P').$ TODO: erzwungen Zeilenumbruch kontrollieren

Beweis.

1. Um die Inklusion zu zeigen wird eine Implementierung P' angegeben, die die strikten Fehler-Traces von P implementiert und zusätzlich auch noch eine passende as-Verfeinerungs-Relation zwischen den beiden Transitionssystemen. P' implementiert wie im Beweis zu Proposition 2.1 alle Transitionen von P . Das P' wird hier jedoch im Gegensatz zu Beweis von 2.1 auch noch alle Fehler-Zustände aus P implementieren. Die entsprechende as-Verfeinerungs-Relation ist jedoch hier ebenfalls wieder die Identitätsabbildung zwischen den Zuständen der Transitionssysteme. Die Definition von P' lautet also:

- $P' = P$ für die Menge der Zustände,
- $p'_0 = p_0$,
- $I_{P'} = I_P$ und $O_{P'} = O_P$,
- $\longrightarrow_{P'} = \dashrightarrow_{P'} = \dashrightarrow_P$,
- $E'_P = E_P$.

Die Tupel, die von 1.3 2. und 3. gefordert werden, dass sie in der as-Verfeinerungs-Relation enthalten sind, werden bereits durch die Identitätsabbildung garantiert, wie im Beweis 2.1. Für 1.3 1. muss für jedes in der as-Verfeinerungs-Relation enthaltene Zustands-Paar gelten, wenn der Zustand aus P kein Fehler-Zustand ist, dann ist auch der Zustand aus P' keiner, die folgt aus der Gleichheit der Mengen E_P und E'_P . Jeder Trace aus $StET(P)$ ist via may-Transitionen in P ausführbar und führt dort zu einem Fehler-Zustand. Der analoge Trace ist auch in P' möglich, da alle may-Transitionen aus P in P' als must-Transitionen implementiert wurden. Der dabei erreichte Zustand steht mit dem Fehler-Zustand in P in der Identitätsabbildung, die Zustände entsprechen sich also. Es gilt also mit $E'_P = E_P$, dass auch der in P' erreichte Zustand ein Fehler-Zustand ist. Für die as-Implementierung P' von P und der Identitätsabbildung als starke as-Verfeinerungs-Relation zwischen den Transitionssystemen gilt also $StET(P) = StET(P')$.

2. Da der erste Punkt dieser Proposition bereits bewiesen wurde, gilt bereits, dass alle strikten Fehler-Traces von P in der Vereinigung aller strikten Fehler-Traces der as-Implementierungen von P enthalten sind. Wenn auf alle Wörter in beiden Mengen die prune-Funktion angewendet wird, gilt die Inklusion der daraus entstanden Mengen weiterhin. Dies entspricht der Behauptung des Punktes.
3. Auch für diese Inklusion wird eine starke as-Verfeinerungs-Relation und eine Implementierung P' angegeben. Jedoch werden nicht wie bei 1. alle Transitionen von P in P' implementiert. Es wird auch für alle wa aus $MIT(P)$ eine eigene Implementierung P' geben und nicht eine für alle. Es werden alle must-Transitionen aus P in P' implementiert und zusätzlich die may-Transitionen, die zum Ausführen von w benötigt werden, so dass das a danach in P nicht gefordert wird. Es kann aufgrund von Schleifen in P auch nicht mehr die Identitätsabbildung als as-Verfeinerungs-Relation gewählt werden. Der w Trace wird entsprechend seiner Länge abgewickelt, so dass sicher gestellt wird, dass der Zustand am Ende dieses Traces wirklich ruhig ist und nicht Teil einer Schlinge in w . Für das Abwickeln werden die Zustände entsprechend ihrer Position in w durchnummeriert. Für ein w , für das $wa \in MIT(P)$, gilt $\exists w' \in \Sigma_\tau^*, \exists \alpha_1, \alpha_2, \dots, \alpha_n, \exists p_1, p_2, \dots, p_n : \hat{w}' = w \wedge w' = \alpha_1 \alpha_2 \dots \alpha_n \wedge p_0 \xrightarrow{\alpha_1} p_1 \xrightarrow{\alpha_2} \dots p_{n-1} \xrightarrow{\alpha_n} p_n$. Die starke as-Verfeinerungs-Relation \mathcal{R} enthält in diesem Fall Tupel $((p, j), p)$ für alle $0 \leq j \leq n$. Die entsprechende Definition für das P' , das wa als Input-kritischen-Trace enthalten soll lautet:

- $P' = P \times \{0, 1, \dots, n\}$,
- $p'_0 = (p_0, 0)$,
- $I'_P = I_P$ und $O'_P = O_P$,
- $\xrightarrow{a}_P = \xrightarrow{a}_{P'} = \left\{ ((p, j), \alpha, (p', j+1)) \mid p \xrightarrow{\alpha}_P p', 0 \leq j < n \right\} \cup \left\{ ((p, j), \alpha, (p', j)) \mid p \xrightarrow{\alpha}_P p', 0 \leq j \leq n \right\}$,
- $E'_P = \emptyset$.

Der Ablauf von w aus P wird in P' durch $(p_0, 0) \xrightarrow{\alpha_1} (p_1, 1) \xrightarrow{\alpha_2} \dots (p_{n-1}, n-1) \xrightarrow{\alpha_n} (p_n, n)$ simuliert. w ist also in P' ausführbar. a ist für (p_n, n) nicht ausführbar, da in P für p_n $p_n \not\xrightarrow{a}$ gilt und für die Zustände mit der Nummer n in P' nur die must-Transitionen implementiert werden. Die implementierten Transitionen in P' haben nach Definition alle zugrundeliegende may-Transitionen in P an den in Relation \mathcal{R} stehenden Zuständen. Es werden alle durch die zweite Menge der Transitions-Definition von P' sind alle must-Transitionen aus P entsprechend in P' implementiert an den Zuständen, die in der starken as-Verfeinerungs-Relation stehen. Da die Menge E'_P leer ist, gelten alle Bedingungen, damit \mathcal{R} eine as-Verfeinerungs-Relation zwischen P' und P ist. Mit der Begründung von oben folgt auch $wa \in MIT(P')$. Da für alle wa aus $MIT(P)$ eine entsprechende Implementierung mit as-Verfeinerungs-Relation angegeben werden kann, gilt die Inklusion.

□

Definition 3.5 (Kommunikationsfehler-Semantik). Sei P ein MEIO.

- Die Menge der Fehler-Traces von P ist $ET(P) := \text{cont}(\text{PrET}(P)) \cup \text{cont}(\text{MIT}(P))$.
- Die Fehler-geflutete Sprache von P ist $EL(P) := L(P) \cup ET(P)$.

Für zwei MEIOs P_1, P_2 mit der gleichen Signatur wird $P_1 \sqsubseteq_E P_2$ geschrieben, wenn $ET_1 \subseteq ET_2$ und $EL_1 \subseteq EL_2$ gilt.

Hierbei ist zu beachten, dass die Mengen $StET$, $PrET$, MIT , ET und EL nur denen aus [Sch16] entsprechen, wenn P bereits eine as-Implementierung ist.

Aus den Propositionen für die Sprache und die Traces konnte für die Vereinigung der gleichen Mengen über die Implementierungen immer nur eine Inklusionsrichtung gefolgert werden, da die Definition 1.3 nach einen Fehler-Zustand in P beliebiges Verhalten ins dessen Implementierungen zulässt. Mit dem Einsatz der cont -Funktion zum beliebigen fortsetzen der Traces kann dies ausgeglichen werden. Somit gilt wie der nächsten Proposition behauptet für die Fehler-Traces und die Fehler-geflutete Sprache Gleichheit und nicht nur die Inklusion, die aus den Propositionen vorher bereits folgt.

Proposition 3.6 (Kommunikationsfehler-Semantik und Implementierungen). Sie P ein MEIO.

1. Für die Menge der Fehler-Traces von P gilt $ET(P) = \bigcup_{P' \in \text{as-impl}(P)} ET(P')$.
2. Für die Fehler-geflutete Sprache von P gilt $EL(P) = \bigcup_{P' \in \text{as-impl}(P)} EL(P')$.

Beweis.

1. „ \subseteq “:

$$\begin{aligned}
 ET(P) &\stackrel{3.5}{=} \text{cont}(\text{PrET}(P)) \cup (\text{MIT}(P)) \\
 &\stackrel{3.4}{\subseteq} \text{cont} \left(\bigcup_{P' \in \text{as-impl}(P)} \text{PrET}(P') \right) \cup \text{cont} \left(\bigcup_{P' \in \text{as-impl}(P)} \text{MIT}(P') \right) \\
 &\stackrel{\text{cont monoton}}{=} \bigcup_{P' \in \text{as-impl}(P)} \text{cont}(\text{PrET}(P')) \cup (\text{MIT}(P')) \\
 &\stackrel{3.5}{=} \bigcup_{P' \in \text{as-impl}(P)} ET(P').
 \end{aligned}$$

1. „ \supseteq “:

Da für P $ET(P) = \text{cont}(\text{PrET}(P)) \cup \text{cont}(\text{MIT}(P))$ gilt und für alle as-Implementierungen P' von P analog, genügt es ein präfix-minimales w aus $ET(P')$ für eine as-Implementierung P' von P zu betrachten. Da w präfix-minimal ist für P' ist entweder vollständig oder bis auf den letzten Buchstaben in P' ausführbar. Dies hängt davon ab, ob $w \in \text{PrET}(P')$ oder $w \in \text{MIT}(P')$ gilt. Für P muss jedoch nicht mal das Präfix

von w ohne den letzten Buchstaben von w ausführbar sein. Für den weiteren Teil dieses Beweises soll gelten $w = va$ für $a \in \Sigma$ und $v \in \Sigma^*$.

- Fall 1 ($v \notin L(P)$): Das v kann in P nicht ausgeführt werden. Dies kann nur den Fall sein, wenn ein Präfix von v zu einem Fehler-Zustand in P führt, da sonst 1.3 3. verletzt wäre. Es ist also ein Präfix von w in $StET(P)$ enthalten und somit gilt wegen der cont-Funktion $w \in ET(P)$.
- Fall 2 ($v \in L(P)$ und $w \in PrET(P')$): In diesem Fall kann das Wort v in P ausgeführt werden. Falls a in P nicht ausgeführt werden kann, ist w aufgrund der gleichen Argumentation wie im 1. Fall in $ET(P)$ enthalten. Es wird also davon ausgegangen, dass $w \in L(P)$ gilt. Die Definition 1.3 fordert, dass alle Zustände p_j und p'_j aus P bzw. P' auf dem w -Trace in der starken as-Verfeinerungs-Relation \mathcal{R} stehen. In P' existiert eine Verlängerung $v \in O^*$ von w , so dass $wv \in StET(P')$. Diese Verlängerung kann in P möglicherweise nicht ausführbar sein, dann ist jedoch analog zu Fall 1 ein Präfix von wv in $StET(P)$ enthalten. Da v nur aus Outputs besteht gilt $prune(wv) = prune(w)$. Es folgt also für ein Präfix von w die Zugehörigkeit zur Menge $PrET(P)$. Somit ist w in $ET(P)$ enthalten. Falls jedoch wv in P ausführbar, kann immer noch ein echtes Präfix von wv in P zu einem Fehler-Zustand führen, wodurch wieder $w \in ET(P)$ folgt. Falls jedoch alle Zustände (bis auf den letzten), die im Trace wv auftauchen keine Fehler-Zustände sind, gilt für p und p' mit $p_0 \xRightarrow{wv}_P p$ und $p'_0 \xRightarrow{wv}_{P'} p'$ ($p', p \in \mathcal{R}$). Da $wv \in StET(P')$ gilt, muss $p' \in E_{P'}$ gelten. Daraus folgt mit 1.3 1., dass p in E_P enthalten sein muss. Es gilt also $wv \in StET(P)$. Mit $prune(w) = prune(wv)$ folgt daraus $w \in ET(P)$.
- Fall 3 ($v \in L(P)$ und $w \in MIT(P')$): Da w in $MIT(P')$ enthalten ist, $a \in I$. Falls ein Präfix von v oder v selbst in P zu einem Fehler Zustand führt, gilt wie in den beiden vorangegangenen Fällen $w \in ET(P)$. Es wird also davon ausgegangen, dass auf dem v Trace in P kein Fehler-Zustand erreicht wird. Es gilt also $p_0 \xRightarrow{v}_P p$ und $p'_0 \xRightarrow{v}_{P'} p'$ mit $(p', p) \in \mathcal{R}$. Wobei \mathcal{R} die as-Verfeinerungs-Relation ist, die P' zu einer as-Implementierung von P macht. Falls a für p eine ausgehende must-Transition wäre, würde 1.3 2. auch für p' die Implementierung der a Transition fordern und somit würde nicht $wa \in MIT(P')$ gelten. Somit gilt $p \not\rightarrow_a$ und $va \in MIT(P)$. Daraus ergibt sich direkt $w \in ET(P)$.

2. „ \subseteq “:

$$\begin{aligned}
 EL(P) &\stackrel{3.5}{=} L(P) \cup ET(P) \\
 &\stackrel{2.1}{\subseteq} \left(\bigcup_{P' \in \text{as-impl}(P)} L(P') \right) \cup ET(P) \\
 &\stackrel{1.}{=} \left(\bigcup_{P' \in \text{as-impl}(P)} L(P') \right) \cup \left(\bigcup_{P' \in \text{as-impl}(P)} ET(P') \right) \\
 &= \bigcup_{P' \in \text{as-impl}(P)} L(P') \cup ET(P')
 \end{aligned}$$

$$\stackrel{3.5}{=} \bigcup_{P' \in \text{as-impl}(P)} EL(P').$$

2. „ \supseteq “:

Da der erste Punkt dieser Proposition bereits bewiesen ist, reicht es aus für diesen Punkt zu zeigen, dass $\bigcup_{P' \in \text{as-impl}(P)} EL(P') \setminus ET(P')$ eine Teilmenge von $EL(P)$ ist. Die Menge $EL(P') \setminus ET(P')$ entspricht $L(P') \setminus ET(P')$. Es muss also ein Wort w aus der Sprache einer as-Implementierung von P betrachtet werden, dass nicht in den Fehler-Traces dieser as-Implementierung enthalten ist. Das Wort w ist also in P' ausführbar. Falls das w in P jedoch nicht ausführbar ist, folgt wiederum $w \in ET(P) \subseteq EL(P)$, da ein Präfix von w in P zu einem Fehler-Zustand führen muss. Falls w jedoch ausführbar ist in P gilt $w \in L(P) \subseteq EL(P)$. \square

Satz 3.7 (Kommunikationsfehler-Semantik für Parallelkompositionen). Für zwei komponierbare MEIOs P_1, P_2 und ihre Komposition P_{12} gilt:

1. $ET_{12} = \text{cont}(\text{prune}((ET_1 \parallel EL_2) \cup (EL_1 \parallel ET_2))),$
2. $EL_{12} = (EL_1 \parallel EL_2) \cup ET_{12}.$

Beweis.

1. „ \subseteq “:

Da beide Seiten der Gleichung unter der Fortsetzung cont abgeschlossen sind, genügt es ein präfix-minimales Element w von ET_{12} zu betrachten. Diese Element ist aufgrund der Definition der Menge der Fehler-Traces in MIT_{12} oder in $PrET_{12}$ enthalten.

- Fall 1 ($w \in MIT_{12}$): Aus der Definition von MIT folgt, dass es eine Aufteilung $w = xa$ gibt mit $(p_{01}, p_{02}) \xRightarrow{x}_{12} (p_1, p_2) \wedge a \in I_{12} \wedge (p_1, p_2) \not\xrightarrow{a}_{12}$. Da $I_{12} = (I_1 \cup I_2) \setminus (O_1 \cup O_2)$ ist, folgt $a \in (I_1 \cup I_2)$ und $a \notin (O_1 \cup O_2)$. Es wird unterschieden, ob $a \in (I_1 \cap I_2)$ oder $a \in (I_1 \cup I_2) \setminus (I_1 \cap I_2)$ ist.
 - Fall 1a) ($a \in (I_1 \cap I_2)$): Durch Projektion des Ablaufes auf die einzelnen Transitionssysteme erhält man oBdA $p_{01} \xRightarrow{x_1}_1 p_1 \not\xrightarrow{a}_1$ und $p_{02} \xRightarrow{x_2}_2 p_2 \not\xrightarrow{a}_2$ oder $p_{02} \xRightarrow{x_2}_2 p_2 \xrightarrow{a}_2$ mit $x \in x_1 \parallel x_2$. Daraus kann $x_1 a \in \text{cont}(MIT_1) \subseteq ET_1$ und $x_2 a \in EL_2$ ($x_2 a \in MIT_2$ oder $x_2 a \in L_2$) gefolgert werden. Damit folgt $w \in (x_1 \parallel x_2) \cdot \{a\} \subseteq (x_1 a) \parallel (x_2 a) \subseteq ET_1 \parallel EL_2$, und somit ist w in der rechten Seite der Gleichung enthalten.
 - Fall 1b) ($a \in (I_1 \cup I_2) \setminus (I_1 \cap I_2)$): OBdA gilt $a \in I_1$. Durch die Projektion auf die einzelnen Komponenten erhält man: $p_{01} \xRightarrow{x_1}_1 p_1 \not\xrightarrow{a}_1$ und $p_{02} \xRightarrow{x_2}_2 p_2$ mit $x \in x_1 \parallel x_2$. Daraus folgt $x_1 a \in \text{cont}(MIT_1) \subseteq ET_1$ und $x_2 \in L_2 \subseteq EL_2$. Somit gilt $w \in (x_1 \parallel x_2) \cdot \{a\} \subseteq (x_1 a) \parallel x_2 \subseteq ET_1 \parallel EL_2$. Dies ist eine Teilmenge der rechten Seite der Gleichung.

- Fall 2 ($w \in PrET_{12}$): Aus der Definition von $PrET$ und $prune$ folgt, dass ein $v \in O_{12}^*$ gibt, so dass $(p_{01}, p_{02}) \xRightarrow{w}_{12} (p_1, p_2) \xRightarrow{v}_{12} (p'_1, p'_2)$ gilt mit $(p'_1, p'_2) \in E_{12}$ und $w = prune(wv)$. Durch Projektion auf die Komponenten erhält man $p_{01} \xRightarrow{w_1}_1 p_1 \xRightarrow{v_1}_1 p'_1$ und $p_{02} \xRightarrow{w_2}_2 p_2 \xRightarrow{v_2}_2 p'_2$ mit $w \in w_1 \| w_2$ und $v \in v_1 \| v_2$. Aus $(p'_1, p'_2) \in ET_{12}$ folgt, dass es sich entweder um einen geerbten oder einen neuen Fehler handelt. Bei einem geerbten wäre bereits einer der beiden Zustände p'_1 bzw. p'_2 ein Fehler-Zustand gewesen. Ein neuer Fehler hingegen wäre durch das fehlen der Synchronisations-Erzwingung (fehlende must-Transition) in einer der Komponenten entstanden.
 - Fall 2a) (geerbter Fehler): OBdA gilt $p'_1 \in E_1$. Daraus folgt, $w_1 v_1 \in StET_1 \subseteq cont(PrET_1) \subseteq ET_1$. Da $p_{02} \xRightarrow{w_2 v_2}_2$ gilt, erhält man $w_2 v_2 \in L_2 \subseteq EL_2$. Dadurch ergibt sich $wv \in ET_1 \| EL_2$ mit $w = prune(wv)$ und somit ist w in der rechten Seite der Gleichung enthalten.
 - Fall 2b) (neuer Fehler): OBdA gilt $a \in I_1 \cap O_2$ mit $p'_1 \not\xrightarrow{a}_1 \wedge p'_2 \not\xrightarrow{a}_1$. Daraus folgt $w_1 v_1 a \in MIT_1 \subseteq ET_1$ und $w_2 v_2 a \in L_2 \subseteq EL_2$. Damit ergibt sich $wva \in ET_1 \| EL_2$, da $a \in O_1 \subseteq O_{12}$ gilt $w = prune(wva)$ und somit ist w in der rechten Seite der Gleichung enthalten.

1. „ \supseteq “:

Wegen der Abgeschlossenheit beider Seiten der Gleichung gegenüber $cont$ wird auch in diesem Fall nur ein präfix-minimales Element $x \in prune((ET_1 \| EL_2) \cup (EL_1 \| ET_2))$ betrachtet. Da x durch die Anwendung der $prune$ -Funktion entstanden ist, existiert ein $y \in O_{12}^*$ mit $xy \in (ET_1 \| EL_2) \cup (EL_1 \| ET_2)$. OBdA wird davon ausgegangen, dass $xy \in ET_1 \| EL_2$ gilt, d.h. es gibt $w_1 \in ET_1$ und $w_2 \in EL_2$ mit $xy \in w_1 \| w_2$.

Im Folgenden wird für alle Fälle von xy gezeigt, dass es ein $v \in PrET_{12} \cup MIT_{12}$ gibt, das ein Präfix von xy ist und v entweder auf einen Input I_{12} endet oder $v = \varepsilon$. Damit muss v ein Präfix von x sein. ε ist Präfix von jedem Wort und sobald v mindestens einen Buchstaben enthält, muss das Ende von v vor dem Anfang von $y \in O_{12}^*$ liegen. Dadurch ist ein Präfix von x in $PrET_{12} \cup MIT_{12}$ enthalten und somit gilt $x \in ET_{12}$, da ET die Fortsetzung der Mengenvereinigung aus $PrET$ und MIT ist.

Sei v_1 das kürzeste Präfix von w_1 in $PrET_1 \cup MIT_1$. Falls $w_2 \in L_2$, so sei $v_2 = w_2$, sonst soll v_2 das kürzeste Präfix von w_2 in $PrET_2 \cup MIT_2$ sein. Jede Aktion in v_1 und v_2 hängt mit einer aus xy zusammen. Es kann nun davon ausgegangen werden, dass entweder $v_2 = w_2 \in L_2$ gilt oder die letzte Aktion von v_1 vor oder gleichzeitig mit der letzten Aktion von v_2 statt findet. Ansonsten endet $v_2 \in PrET_2 \cup MIT_2$ vor v_1 und somit ist dieser Fall analog zu v_1 endet vor v_2 .

- Fall 1 ($v_1 = \varepsilon$): Da $\varepsilon \in PrET_1 \cup MIT_1$, ist bereits in P_1 ein Fehler-Zustand lokal erreichbar. $\varepsilon \in MIT_1$ ist nicht möglich, da jedes Element aus MIT nach Definition mindestens die Länge 1 haben muss. Mit der Wahl $v'_2 = v' = \varepsilon$ ist v'_2 ein Präfix von v_2 .
- Fall 2 ($v_1 \neq \varepsilon$): Aufgrund der Definitionen von $PrET$ und MIT endet v_1 auf ein $a \in I_1$, d.h. $v_1 = v'_1 a$. v' sei das Präfix von xy , das mit der letzten Aktion von v_1

endet, d.h. mit a und $v'_2 = v'|_{\Sigma_2}$. Falls $v_2 = w_2 \in L_2$, dann ist v'_2 ein Präfix von v_2 . Falls $v_2 \in PrET_2 \cup MIT_2$ gilt, dann ist durch die Annahme, dass v_2 nicht vor v_1 endet, v'_2 ein Präfix von v_2 . Im Fall $v_2 \in MIT_2$ weiß man zusätzlich, dass v_2 auf $b \in I_2$ endet. Es kann jedoch $a = b$ gelten.

In allen Fällen erhält man $v'_2 = v'|_{\Sigma_2}$ ist ein Präfix von v_2 und $v' \in v_1 \| v'_2$ ist ein Präfix von xy . Es kann nur für die Fälle $a \notin I_2$ gefolgert werden, dass $p_{02} \xRightarrow{v'_2}_2$ gilt.

- Fall I ($v_1 \in MIT_1$ und $v_1 \neq \varepsilon$): Es gibt einen Ablauf der Form $p_{01} \xRightarrow{v'_1}_1 p_1 \not\xrightarrow{a}_1$ und es gilt $v' = v''a$.
 - Fall Ia) ($a \notin \Sigma_2$): Es gilt $p_{02} \xRightarrow{v'_2}_2 p_2$ mit $v'' \in v'_1 \| v'_2$. Dadurch erhält man $(p_{01}, p_{02}) \xRightarrow{v''}_{12} (p_1, p_2) \not\xrightarrow{a}_{12}$ mit $a \in I_{12}$. Somit wird $v := v''a = v' \in MIT_{12}$ gewählt.
 - Fall Ib) ($a \in I_2$ und $v'_2 \in MIT_2$): Es gilt $v'_2 = v''_2 a$ mit $p_{02} \xRightarrow{v''_2}_2 \not\xrightarrow{a}_2$ und $v'' \in v'_1 \| v''_2$. a ist für P_2 , ebenso wie für P_1 , ein nicht erzwungener Input. Daraus folgt, dass $(p_1, p_2) \not\xrightarrow{a}_{12}$ gilt. Es wird ebenfalls $v := v''a = v' \in MIT_{12}$ gewählt.
 - Fall Ic) ($a \in I_2$ und $v'_2 \in L_2 \setminus MIT_2$): Es gilt $p_{02} \xRightarrow{v''_2}_2 p_2 \not\xrightarrow{a}_2$ mit $v'_2 = v''_2 a$. Da die gemeinsamen Inputs synchronisiert werden, folgt $(p_1, p_2) \not\xrightarrow{a}_{12}$ bereits aus $q_1 \not\xrightarrow{a}_1$. Somit kann hier nochmals $v := v''a = v' \in MIT_{12}$ gewählt werden.
 - Fall Id) ($a \in O_2$): Es gilt $v'_2 = v''_2 a$ und $p_{02} \xRightarrow{v''_2}_2$. Man erhält also $p_{02} \xRightarrow{v''_2}_2 p_2 \not\xrightarrow{a}_2$ mit $v'' \in v'_1 \| v''_2$. Daraus ergibt sich $(p_{01}, p_{02}) \xRightarrow{v''}_{12} (p_1, p_2)$ mit $p_2 \not\xrightarrow{a}_2$, $p_1 \not\xrightarrow{a}_1$, $a \in I_1$ und $a \in O_2$, somit gilt $(p_1, p_2) \in E_{12}$. Es wird $v := \text{prune}(v'') \in PrET_{12}$ gewählt.
- Fall II ($v_1 \in PrET_1$): $\exists u_1 \in O_1^* : p_{01} \xRightarrow{v_1}_1 p_1 \xRightarrow{u_1}_1 p'_1$ mit $p'_1 \in E_1$. Im Fall $v'_1 \neq \varepsilon$ kann das a , auf das v_1 endet, ebenfalls der letzte Buchstabe von v_2 sein. Im Fall von $v_2 \in MIT_2$ kann somit $a = b$ gelten, wodurch $v_2 = v'_2$ gilt. Dieser Fall verläuft jedoch analog zu Fall Ic) und wird hier nicht weiter betrachtet. Es gilt für alle anderen Fälle $p_{02} \xRightarrow{v'_2}_2 p_2$ mit $(p_{01}, p_{02}) \xRightarrow{v'}_{12} (p_1, p_2)$.
 - Fall IIa) ($u_2 \in (O_1 \cap I_2)^*, c \in (O_1 \cap I_2)$, sodass $u_2 c$ Präfix von $u_1|_{I_2}$ mit $p_2 \xRightarrow{u_2}_2 p'_2 \not\xrightarrow{c}_2$): Für das Präfix $u'_1 c$ von u_1 mit $(u'_1 c)|_{I_2} = u_2 c$ weiß man, dass $q_1 \xRightarrow{u'_1}_1 q'_1 \not\xrightarrow{c}_1$. Somit gilt $u'_1 \in u_1 \| u_2$ und $(p_1, p_2) \xRightarrow{u'_1}_{12} (q'_1, q'_2) \in E_{12}$, der für P_2 der entsprechende Input nicht erzwungen wird, der mit dem c Output von P_1 zu koppeln wäre. Es handelt sich also um einen neuen Fehler. Es wird $v := \text{prune}(v' u'_1) \in PrET_{12}$ gewählt, dies ist ein Präfix von v' , da $u_1 \in O_1^*$.

- Fall IIb) ($p_2 \xRightarrow{u_2}_2 p'_2$ mit $u_2 = u_1|_{I_2}$): Es gilt $u_1 \in u_1 \| u_2$ und $(p_1, p_2) \xRightarrow{u_1}_{12} (p'_1, p'_2) \in E_{12}$, da $p'_1 \in E_1$ und somit handelt es sich in P_{12} um einen geerbten Fehler. Nun wird $v := \text{prune}(v'u_1) \in \text{PrET}_{12}$ gewählt, das wiederum ein Präfix von v' ist.

2.:

Durch die Definitionen ist klar, dass $L_i \subseteq EL_i$ und $ET_i \subseteq EL_i$ gilt. Die Argumentation startet auf den rechten Seite der Gleichung:

$$\begin{aligned}
 (EL_1 \| EL_2) \cup ET_{12} &\stackrel{3.5}{=} ((L_1 \cup ET_1) \| (L_2 \cup ET_2)) \cup ET_{12} \\
 &= (L_1 \| L_2) \cup \underbrace{(L_1 \| ET_2)}_{\substack{\subseteq (EL_1 \| ET_2) \\ \stackrel{1.}{\subseteq} ET_{12}}} \cup \underbrace{(ET_1 \| L_2)}_{\substack{\subseteq (ET_1 \| EL_2) \\ \stackrel{1.}{\subseteq} ET_{12}}} \cup \underbrace{(ET_1 \| ET_2)}_{\substack{\subseteq (EL_1 \| ET_2) \\ \stackrel{1.}{\subseteq} ET_{12}}} \cup ET_{12} \\
 &= (L_1 \| L_2) \cup ET_{12} \\
 &\stackrel{2.2}{=} L_{12} \cup ET_{12} \\
 &\stackrel{3.5}{=} EL_{12}.
 \end{aligned}$$

□

Korollar 3.8 (Kommunikationsfehler-Präkongruenz). Die Relation \sqsubseteq_E ist eine Präkongruenz bezüglich $\cdot \| \cdot$.

Beweis. Es muss gezeigt werden: Wenn $P_1 \sqsubseteq_E P_2$ gilt, dann für jedes komponierbare P_3 auch $P_{31} \sqsubseteq_E P_{32}$. D.h. es ist zu zeigen, dass aus $ET_1 \subseteq ET_2$ und $EL_1 \subseteq EL_2$, $ET_{31} \subseteq ET_{32}$ und $EL_{31} \subseteq EL_{32}$ folgt. Dies ergibt sich aus der Monotonie von cont , prune und $\cdot \| \cdot$ auf Sprachen wie folgt:

- $ET_{31} \stackrel{3.7}{=}^1 \text{cont}(\text{prune}((ET_3 \| EL_1) \cup (EL_3 \| ET_1)))$
 $\begin{array}{l} ET_1 \subseteq ET_2 \\ \text{und} \\ EL_1 \subseteq EL_2 \end{array} \subseteq \text{cont}(\text{prune}((ET_3 \| EL_2) \cup (EL_3 \| ET_2)))$
 $\stackrel{3.7}{=}^1 ET_{32},$
- $EL_{31} \stackrel{3.7}{=}^2 (EL_3 \| EL_1) \cup E_{31}$
 $\begin{array}{l} EL_1 \subseteq EL_2 \\ \text{und} \\ ET_{31} \subseteq ET_{32} \end{array} \subseteq (EL_3 \| EL_2) \cup ET_{32}$
 $\stackrel{3.7}{=}^2 EL_{32}.$

□

Lemma 3.9 (Verfeinerung mit Kommunikationsfehlern). Gegeben sind zwei MEIOs P_1 und P_2 mit der gleichen Signatur. Wenn $U \| P_1 \sqsubseteq_E^B U \| P_2$ für alle Partner U gilt, dann folgt daraus die Gültigkeit von $P_1 \sqsubseteq_E P_2$.

Beweis. Da P_1 und P_2 die gleiche Signaturen haben wird $I := I_1 = I_2$ und $O := O_1 = O_2$ definiert. Für jeden Partner U gilt $I_U = O$ und $O_U = I$.

Um $P_1 \sqsubseteq_E P_2$ zu zeigen, wird nachgeprüft, ob folgendes gilt:

- $ET_1 \subseteq ET_2$,
- $EL_1 \subseteq EL_2$.

Für ein gewähltes präfix-minimales Element $w \in ET_1$ wir gezeigt, dass dieses w oder eines seiner Präfixe in ET_2 enthalten ist. Dies ist möglich, da die beiden Mengen ET_1 und ET_2 durch cont abgeschlossen sind.

- Fall 1 ($w = \varepsilon$): Es handelt sich um einen lokal erreichbaren Fehler-Zustand in P_1 . Für U wird ein Transitionssystem verwendet, das nur aus dem Startzustand und einer must-Schleife für alle Inputs $x \in I_U$ besteht. Somit kann P_1 die im Prinzip gleichen Fehler-Zustände lokal erreichen wie alle möglichen as-Implementierungen von $U \parallel P_1$ zusammen. Daraus folgt, dass auch mindestens eine as-Implementierung von $U \parallel P_2$ einen lokal erreichbaren Fehler-Zustand haben muss. Durch die Definition von U kann dieser Fehler nur von P_2 geerbt sein. Es muss also in P_2 ein Fehler-Zustand durch interne Aktionen und Outputs erreichbar sein, d.h. es gilt $\varepsilon \in PrET_2$.
- Fall 2 ($w = x_1 \dots x_n x_{n+1} \in \Sigma^+$ mit $n \geq 0$ und $x_{n+1} \in I = O_U$): Es wird der folgende Partner U betrachtet (siehe auch Abbildung 3.1):

- $U = \{p_0, p_1, \dots, p_{n+1}\}$,
- $p_{0U} = p_0$,
- $\rightarrow_U = \{(p_i, x_{i+1}, p_{i+1}) \mid 0 \leq i \leq n\}$
 $\cup \{(p_i, x, p_{n+1}) \mid x \in I_U \setminus \{x_{i+1}\}, 0 \leq i \leq n\}$
 $\cup \{(p_{n+1}, x, p_{n+1}) \mid x \in I_U\}$,
- $E_U = \emptyset$.

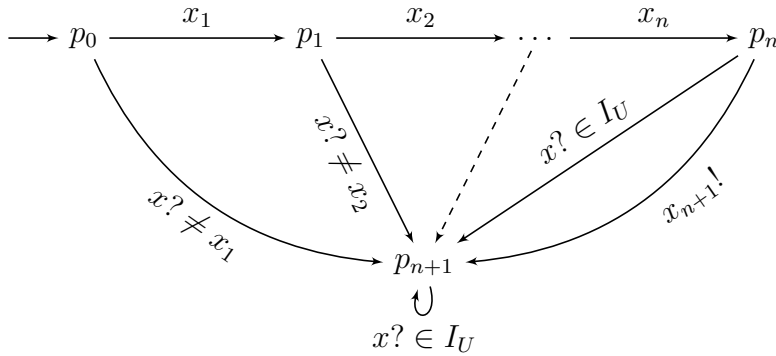


Abbildung 3.1: $x? \neq x_i$ steht für alle $x \in I_U \setminus \{x_i\}$

Für w können nun zwei Fälle unterschieden werden. Aus beiden wird folgen, dass für mindestens eine as-Implementierung P' von $U \parallel P_1$ $\varepsilon \in PrET(P')$.

- Fall 2a) ($w \in MIT_1$): In $U \parallel P_1$ erhält man $(p_0, p_{01}) \xrightarrow{x_1 \dots x_n} U \parallel P_1 (p_n, p')$ mit $p' \not\xrightarrow{x_{n+1}}_1$ und $p_n \xrightarrow{x_{n+1}}_U$. Deshalb gilt $(p_1, p') \in E_{U \parallel P_1}$. Da alle Aktionen aus w bis auf x_{n+1} synchronisiert werden und $I \cap I_U = \emptyset$, gilt $x_1, \dots, x_n \in O_{U \parallel P_1}$. Da $(p_1, p') \in E_{U \parallel P_1}$ gibt es mindestens ein P' in $as\text{-}impl(U \parallel P_1)$, die diesen Fehler-Zustand ebenfalls enthält. Daraus ergibt sich dann $\varepsilon \in PrET(P')$.
- Fall 2b) ($w \in PrET_1$): In $U \parallel P_1$ erhält man $(p_0, p_{01}) \xRightarrow{w} U \parallel P_1 (p_{n+1}, p'') \xRightarrow{u} U \parallel P_1 (p_{n+1}, p')$ für $u \in O^*$ und $p' \in E_1$. Daraus folgt $(p_{n+1}, p') \in E_{U \parallel P_1}$ und somit $wu \in StET(U \parallel P_1)$. Da alle Aktionen in w synchronisiert werden und $I \cap I_U = \emptyset$, gilt $x_1, \dots, x_n, x_{n+1} \in O_{U \parallel P_1}$ und, da $u \in O^*$, folgt $u \in O_{U \parallel P_1}^*$. Somit ergibt sich für eine as-Implementierung P' von $U \parallel P_1$ $\varepsilon \in PrET(P')$.

Da $\varepsilon \in PrET(P')$ für ein P' aus $as\text{-}impl(U \parallel P_1)$ gilt, kann durch $U \parallel P_1 \sqsubseteq_E^B U \parallel P_2$ geschlossen werden, dass auch in mindestens einer as-Implementierung von $U \parallel P_2$ ein Fehler-Zustand lokal erreichbar sein muss. Da as-Implementierungen die Definition 1.3 erfüllen müssen, muss jeder in einer as-Implementierung von $U \parallel P_2$ lokal erreichbare Fehler auch in $U \parallel P_2$ lokal erreichbar sein.

Dieser Fehler kann geerbt oder neu sein.

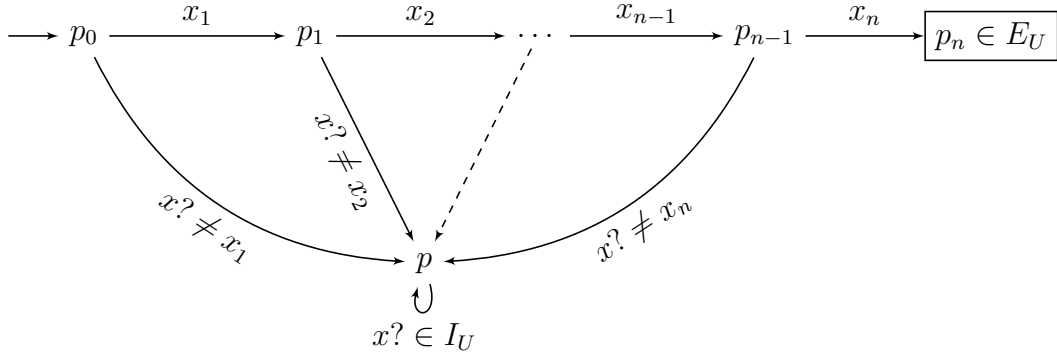
- Fall 2i) (neuer Fehler): Da jeder Zustand von U alle Inputs $x \in O = I_U$ durch must-Transitionen erzwingt, muss ein lokal erreichbarer Fehler-Zustand der Form sein, dass ein Output $a \in O_U$ von U möglich ist, der nicht mit einem passenden Input aus P_2 synchronisiert werden muss (P_2 enthält die entsprechende a Transitionen nicht als must-Transition). Durch die Konstruktion von U sind in p_{n+1} keine Outputs möglich. Ein neuer Fehler muss also die Form (p_i, p') haben mit $i \leq n, p' \not\xrightarrow{x_{i+1}}_2$ und $x_{i+1} \in O_U = I$. Durch Projektion erhält man dann $p_{02} \xRightarrow{x_1 \dots x_i} p' \not\xrightarrow{x_{i+1}}_2$ und damit gilt $x_1 \dots x_{i+1} \in MIT_2 \subseteq ET_2$. Somit ist ein Präfix von w in ET_2 enthalten.
- Fall 2ii) (geerbter Fehler): U hat $x_1 \dots x_i u$ mit $u \in I_U^* = O^*$ ausgeführt und ebenso hat P_2 dieses Wort abgearbeitet. Durch dies hat P_2 einen Zustand E_2 erreicht, da von U kleine Fehler geerbt werden können. Es gilt dann $prune(x_1 \dots x_i u) = prune(x_1 \dots x_i) \in PrET_2 \subseteq ET_2$. Da $x_1 \dots x_i$ ein Präfix von w ist, führt in diesem Fall eine Verlängerung um lokale Aktionen von einem Präfix von w zu einem Fehler-Zustand. Da ET der Menge aller Verlängerungen von gekürzten Fehler-Traces entspricht, ist $x_1 \dots x_i$ in ET_2 enthalten und somit ist ein Präfix von w in ET_2 enthalten.

Um die andere Inklusion zu beweisen, reicht es aufgrund der ersten Inklusion und der Definition von EL aus zu zeigen, dass $L_1 \setminus ET_1 \subseteq EL_2$ gilt.

Es wird dafür ein beliebiges $w \in L_1 \setminus ET_1$ gewählt und gezeigt, dass es in EL_2 enthalten ist.

- Fall 1 ($w = \varepsilon$): Da ε immer in EL_2 enthalten ist, muss hier nichts gezeigt werden.

- Fall 2 ($w = x_1 \dots x_n$ mit $n \geq 1$): Es wird ein Partner U wie folgt konstruiert (siehe dazu auch Abbildung 3.2)
 - $U = \{p_0, p_1, \dots, p_n, p\}$,
 - $p_{0U} = p_0$,
 - $\rightarrow_U = \{(p_i, x_{i+1}, p_{i+1}) \mid 0 \leq i < n\}$
 $\cup \{(p_i, x, p) \mid x \in I_U \setminus \{x_{i+1}\}, 0 \leq i < n\}$
 $\cup \{(p, x, p) \mid x \in I_U\}$,
 - $E_U = \{p_n\}$.


 Abbildung 3.2: $x? \neq x_i$ steht für alle $x \in I_U \setminus \{x_i\}$, p_n ist der einzige Fehler-Zustand

Da $p_{01} \xRightarrow{w}_1 p'$ gilt, kann man schließen, dass $U \parallel P_1$ einen lokal erreichbaren geerbten Fehler hat. Es gibt also auch mindestens eine Implementierung in $\text{as-impl}(U \parallel P_1)$, die diesen lokal erreichbaren Fehler implementiert. Somit muss es eine as-Implementierung von $U \parallel P_2$ geben, die ebenfalls einen lokal erreichbaren Fehler-Zustand hat. Aufgrund von Definition 1.3 3. muss dieser Fehler-Zustand auch in $U \parallel P_2$ lokal erreichbar sein.

- Fall 2a) (neuer Fehler aufgrund von $x_i \in O_U$ und $p_{02} \xRightarrow{x_1 \dots x_{i-1}}_2 q'' \not\xrightarrow{x_i}_2$): Es gilt $x_1 \dots x_i \in MIT_2$ und somit $w \in EL_2$. Anzumerken ist, dass es nur auf diesem Weg Outputs von U möglich sind, deshalb gibt es keine anderen Outputs von U , die zu einem neuen Fehler führen können.
- Fall 2b) (neuer Fehler aufgrund von $a \in O = I_U$): Der einzige Zustand, in dem U nicht alle Input erlaubt sind, ist p_n , der bereits ein Fehler-Zustand ist. Da in diesem Fall dieser Zustand in $U \parallel P_2$ erreichbar ist, besitzt das komponierte MEIO einen geerbten Fehler und es gilt $w \in L_2 \subseteq EL_2$, wegen dem folgenden Fall 2c).
- Fall 2c) (geerbter Fehler von U): Da p_n der einzige Fehler-Zustand in U ist und alle Aktionen synchronisiert sind, ist dies nur möglich, wenn $p_{02} \xRightarrow{x_1 \dots x_n}_2$ gilt. In diesem Fall gilt $w \in L_2 \subseteq EL_2$.

- Fall 2d) (geerbter Fehler von P_2): Es gilt dann $p_{02} \stackrel{x_1 \dots x_i u}{\equiv}_2 p' \in E_2$ für $i \geq 0$ und $u \in O^*$. Somit ist $x_1 \dots x_i u \in StET_2$ und damit $\text{prune}(x_1 \dots x_i u) = \text{prune}(x_1 \dots x_i) \in PrET_2 \subseteq EL_2$. Somit gilt $w \in EL_2$.

□

Der folgende Satz sagt aus, dass \sqsubseteq_E die größte Präkongruenz ist, die charakterisiert werden soll, also gleich der vollständig abstrakten Präkongruenz \sqsubseteq_E^C .

Satz 3.10 (Vollständige Abstraktheit für Kommunikationsfehler-Semantik).
Für zwei MEIOs P_1 und P_2 mit derselben Signatur gilt $P_1 \sqsubseteq_E^C P_2 \Leftrightarrow P_1 \sqsubseteq_E P_2$.

Beweis.

„ \Leftarrow “: Nach Definition gilt, genau dann wenn $\varepsilon \in ET(P)$, ist ein Fehler-Zustand lokal erreichbar in P . $P_1 \sqsubseteq_E P_2$ impliziert, dass $\varepsilon \in ET_2$ gilt, wenn $\varepsilon \in ET_1$. Somit ist ein Fehler-Zustand in P_1 nur dann lokal erreichbar, wenn dieser auch in P_2 lokal erreichbar ist. Falls es also eine as-Implementierung von P_1 gibt, in der ein Fehler-Zustand lokal erreichbar ist, dann gibt es auch mindestens eine as-Implementierung von P_2 , die einen Fehler-Zustand lokal erreichen kann. Dadurch folgt, dass $P_1 \sqsubseteq_E^B P_2$ gilt, da \sqsubseteq_E^B in Definition 3.2 über die lokale Erregbarkeit der Fehler-Zustände in den as-Implementierungen definiert wurde und die ET -Mengen von P_1 und P_2 auch durch die Vereinigung der Traces ihrer as-Implementierungen, wie in Proposition 3.4, ausgedrückt werden können. Es ist also \sqsubseteq_E in \sqsubseteq_E^B enthalten. Wie in Korollar 3.8 gezeigt, ist \sqsubseteq_E eine Präkongruenz bezüglich $\cdot\|\cdot$. Da \sqsubseteq_E^C die größte Präkongruenz bezüglich $\cdot\|\cdot$ ist, die in \sqsubseteq_E^B enthalten ist, muss \sqsubseteq_E in \sqsubseteq_E^C enthalten sein. Es folgt also aus $P_1 \sqsubseteq_E P_2$, dass auch $P_1 \sqsubseteq_E^C P_2$ gilt.

„ \Rightarrow “: Durch die Definition von \sqsubseteq_E^C als Präkongruenz in 3.2 folgt aus $P_1 \sqsubseteq_E^C P_2$, dass $U\|P_1 \sqsubseteq_E^C U\|P_2$ für alle MEIOs U gilt, die mit P_1 komponierbar sind. Da \sqsubseteq_E^C nach Definition auch in \sqsubseteq_E^B enthalten sein soll, folgt aus $U\|P_1 \sqsubseteq_E^C U\|P_2$ auch die Gültigkeit von $U\|P_1 \sqsubseteq_E^B U\|P_2$ für alle diese MEIOs U . Mit Lemma 3.9 folgt dann $P_1 \sqsubseteq_E P_2$. □

Es wurde somit jetzt eine Kette an Folgerungen gezeigt, die sich zu einem Ring schließt. Dies ist in Abbildung 3.3 dargestellt.

Angenommen man definiert, dass $P_1 P_2$ verfeinern soll, genau dann wenn für alle Partner MEIOs U , für die P_2 fehler-frei mit U kommuniziert, folgt, dass P_1 ebenfalls fehler-frei mit U kommuniziert. Dann wird auch diese Verfeinerung durch \sqsubseteq_E charakterisiert.

Korollar 3.11. Es gilt: $P_1 \sqsubseteq_E P_2 \Leftrightarrow U\|P_1 \sqsubseteq_E^B U\|P_2$ für alle Partner U .

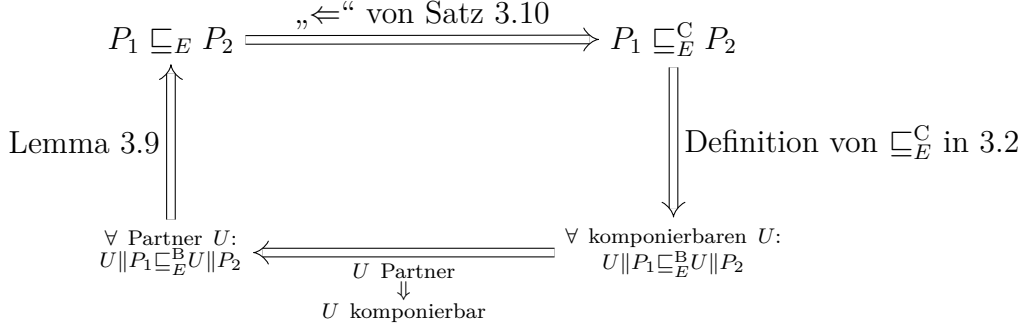


Abbildung 3.3: Folgerungskette der Fehler-Relationen

3.2 Testing-Ansatz

Der Testing-Ansatz stützt auf den Ansatz, der in [BV15b] angewendet wurde. Jedoch sind Tests hier Tupel aus einer Implementierung und einer Menge an Aktionen, über denen synchronisiert werden soll, da die MEIOs, die komponiert werden die Menge Synch an Aktionen automatisch synchronisieren. Es wird hier im Gegensatz zu [BV15b] mit Inputs und Outputs gearbeitet und dadurch scheint der Ansatz die gemeinsamen Aktionen zu synchronisieren natürlicher, wie eine Menge vorzugeben.

Die Definition von lokaler Erreichbarkeit eines Fehler-Zustandes soll aus dem Erweiterungs-Ansatz übernommen werden. Hier ist vor allem wichtig, dass für Implementierungen ein $w \in O^*$ existieren muss mit $p_0 \xRightarrow{w} p \in E$, falls in der Implementierung ein Fehler lokal erreichbar ist.

Definition 3.12 (Test und Verfeinerung). Ein Test T ist eine Implementierung. Ein MEIO P as-erfüllt einen Test T , falls $S \parallel T$ fehler-frei ist für $S \in \text{as-impl}(P)$. Es wird dann $P \text{ sat}_{\text{as}} T$ geschrieben.
Ein MEIO P verfeinert P' , falls für alle Tests T : $P' \text{ sat}_{\text{as}} T \Rightarrow P \text{ sat}_{\text{as}} T$.

Die Definition 3.3 der Kommunikationsfehler-Traces eines MEIOs P kann aus dem Erweiterungs-Ansatz übernommen werden. Ebenso wie die Definition der Kommunikationsfehler-Semantik aus Definition 3.5. Daraus kann dann auch in diesem Ansatz der Satz 3.7, das Korollar 3.8 und die Propositionen 3.4 und 3.6 bewiesen werden.

Die Basisrelation aus dem Erweiterungs-Ansatz gibt es in diesem Ansatz nicht, somit kann das Lemma 3.9 nicht in dieser Art formuliert werden. Jedoch kann hier jetzt mit Test gearbeitet werden.

Lemma 3.13 (Testing-Verfeinerung mit Kommunikationsfehlern). Gegeben sind zwei MEIOs P_1 und P_2 mit der gleichen Signatur. Wenn für alle Test T , die Partner von P_1 bzw. P_2 sind, $P_2 \text{ sat}_{\text{as}} T \Rightarrow P_1 \text{ sat}_{\text{as}} T$ gilt, dann folgt daraus die Gültigkeit von $P_1 \subseteq_E P_2$.

Beweis. Da P_1 und P_2 die gleichen Signaturen haben wird $I := I_1 = I_2$ und $O := O_1 = O_2$ definiert. Für jeden Test Partner T gilt $I_T = O$ und $O_T = I$.

Um $P_1 \sqsubseteq_E P_2$ zu zeigen, wird nachgeprüft, ob folgendes gilt:

- $ET_1 \subseteq ET_2$,
- $EL_1 \subseteq EL_2$.

Für ein gewähltes präfix-minimales Element $w \in ET_1$ wird gezeigt, dass dies w oder eines seiner Präfixe in ET_2 enthalten ist. Dies ist möglich, da die beiden Mengen ET_1 und ET_2 durch cont abgeschlossen sind.

Mit Proposition 3.6 folgt mit $w \in ET_1$, dass es auch eine as-Implementierung P'_1 von P_1 geben muss, für die w ebenfalls in $ET_{P'_1}$ enthalten ist.

- Fall 1 ($w = \varepsilon$): Es ist ein Fehler-Zustand in P'_1 lokal erreichbar. Für T wird ein Transitionssystem verwendet, das nur aus dem Startzustand und einer must-Schleife für alle Inputs $x \in I_T$ besteht. Somit kann P'_1 die im Prinzip gleichen Fehler-Zustände lokal erreichen wie $P'_1 \| T$. P'_1 ist in Parallelkomposition mit T nicht fehler-frei somit gilt $P_1 \text{ sat}_{\text{as}} T$ nicht. Es muss also auch P_2 den Test T nicht as-erfüllen, wegen der Implikation $P_2 \text{ sat}_{\text{as}} T \Rightarrow P_1 \text{ sat}_{\text{as}} T$. Damit P_2 T nicht as-erfüllt muss es eine as-Implementierung P'_2 geben, die in Parallelkomposition mit T nicht fehler-frei ist. Da T ein Partner von P'_2 ist, gibt es in der Parallelkomposition nur lokale Aktionen. Somit muss in $P'_2 \| T$ ein Fehler lokal erreichbar sein. Durch die Definition von T kann dieser Fehler nur von P'_2 geerbt sein. In P'_2 kann dieser Fehler-Zustand nur durch internen Aktionen und Outputs erreichbar sein, da T keine Outputs besitzt, die man mit Inputs aus P'_2 synchronisieren könnte und unsynchronisierte Aktionen sind in einer Parallelkomposition von Partner nicht möglich. Somit gilt $\varepsilon \in PrET_{P'_2} \subseteq ET_{P'_2}$. Mit Proposition 3.6 folgt daraus $\varepsilon \in ET_2$.
- Fall 2 ($w = x_1 \dots x_n x_{n+1} \in \Sigma^+$ mit $n \geq 0$ und $x_{n+1} \in I = O_T$): Es wird der folgende Partner T betrachtet (dieser entspricht bis auf die Benennung Mengen U aus Abbildung 3.1):

- $T = \{p_0, p_1, \dots, p_{n+1}\}$,
- $p_0 T = p_0$,
- $\longrightarrow_T = \{(p_i, x_{i+1}, p_{i+1}) \mid 0 \leq i \leq n\}$
 $\cup \{(p_i, x, p_{n+1}) \mid x \in I_T \setminus \{x_{i+1}\}, 0 \leq i \leq n\}$
 $\cup \{(p_{n+1}, x, p_{n+1}) \mid x \in I_T\}$,
- $E_T = \emptyset$.

Für w können nun zwei Fälle unterschieden werden, für die beide $\varepsilon \in PrET(P'_1 \| T)$ folgen wird.

- Fall 2a) ($w \in MIT_{P'_1}$): In $P'_1 \| T$ erhält man $(p'_{01}, p_0) \xrightarrow{x_1 \dots x_n}_{P'_1 \| T} (p', p_n)$ mit $p' \xrightarrow{x_{n+1}}_{P'_1}$ und $p_n \xrightarrow{x_{n+1}}_T$. Deshalb gilt TODO: Fehler-Zustand in anderem Beweis überprüfen

3 Verfeinerungen für Kommunikationsfehler-Freiheit

- Fall 2b) ($w \in PrET_{P'_1}$):

TODO: zu beweisen

□

Literaturverzeichnis

- [BFLV16] Ferenc Bujtor, Sascha Fendrich, Gerald Lüttgen, und Walter Vogler, *Non-deterministic Modal Interfaces*, Theor. Comput. Sci. **642** (2016), 24–53.
- [BV15a] Ferenc Bujtor und Walter Vogler, *Error-pruning in interface automata*, Theor. Comput. Sci. **597** (2015), 18–39.
- [BV15b] ———, *Failure Semantics for Modal Transition Systems*, ACM Trans. Embedded Comput. Syst. **14** (2015), no. 4, 67:1–67:30.
- [Sch16] Ayleen Schinko, *Kommunikationsfehler, Verklemmung und Divergenz bei Interface-Automaten*, Bachelorarbeit, Universität Augsburg, 2016.