1 Grundlagen

Stand: 11. September 2017

1.1 Definitionen

Das Definitions-Kapitel wurde auf Grundlage von [BV15b] und [Sch16] zusammengestellt und teilweise von [BFLV16] beeinflusst.

Definition 1.1 (*Modal Error-I/O-Transitionssystem*). *Ein* Modales 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* $\mathrm{Sig}(P) = (I, O)$*.*
Falls $\longrightarrow = \dashrightarrow$ *gilt, wird* P *auch* Implementierung *genannt.*

Implementierungen entsprechen den z.B. in [Sch16] behandelten EIOs.
Must-Transitionen sind Transitionen, die von einer Verfeinerung implementiert werden müssen. Die may-Transitionen sind hingegen die zulässigen Transitionen für eine Verfeinerung. Alle nicht vorhandenen must- und may-Transitionen dürfen auch in keiner Verfeinerung einer Spezifikation in MEIO-Form auftauchen. Diese Forderungen werden in den Definitionen der Verfeinerungen in 1.3 und 1.4 berücksichtigt.
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 $p \overset{\alpha}{\dashrightarrow} p'$ für $(p, \alpha, p') \in \dashrightarrow$ und $p \overset{\alpha}{\dashrightarrow}$ für $\exists p' : (p, \alpha, p') \in \dashrightarrow$ verwendet. Dies kann entsprechend auf Buchstaben-Sequenzen $w \in \Sigma_\tau^*$ erweitert werden: $p \overset{w}{\dashrightarrow} p'$ ($p \overset{w}{\dashrightarrow}$) steht für die Existenz eines *Laufes* $p \overset{\alpha_1}{\dashrightarrow} p_1 \overset{\alpha_2}{\dashrightarrow} \dots p_{n-1} \overset{\alpha_n}{\dashrightarrow} p'$ ($p \overset{\alpha_1}{\dashrightarrow} p_1 \overset{\alpha_2}{\dashrightarrow} \dots p_{n-1} \overset{\alpha_n}{\dashrightarrow}$) mit $w = \alpha_1 \dots \alpha_n$.
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. $\widehat{w}$ steht für $w|_\Sigma$. Es wird $p \overset{w}{\Longrightarrow} p'$ für ein $w \in \Sigma^*$ geschrieben, falls $\exists w' \in \Sigma_\tau^* : \widehat{w'} = w \wedge p \overset{w'}{\dashrightarrow} p'$, und $p \overset{w}{\Longrightarrow}$, falls $p \overset{w}{\Longrightarrow} p'$ für ein beliebiges p' gilt. Falls $p_0 \overset{w}{\Longrightarrow} p$ gilt, dann wird w *Trace* genannt und p ist ein *erreichbarer Zustand*.
Analog zu $\dashrightarrow$ und $\Longrightarrow$ 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 die Notation zu vereinfachen soll $p \overset{a}{\not\longrightarrow}$ und $p \overset{a}{\not\dashrightarrow}$ für $\neg\exists p' : p \overset{a}{\longrightarrow} p'$ und $\neg\exists p' : p \overset{a}{\dashrightarrow} p'$ stehen. $p \overset{a}{\longrightarrow}\overset{\varepsilon}{\Longrightarrow} p'$ wird geschrieben, wenn p'' existiert, so dass $p \overset{a}{\longrightarrow} p'' \overset{\varepsilon}{\Longrightarrow} p'$ gilt. Diese Transition wird auch als *schwach-nachlaufende must-Transition* bezeichnet. Entsprechend steht $\overset{a}{\dashrightarrow}\overset{\varepsilon}{\Longrightarrow}$ für die *schwach-nachlaufende may-Transition*.
In Grafiken 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 $P_{12} := P_1 \| P_2 = ((P_1 \times P_2), I, O, \longrightarrow_{12}, \dashrightarrow_{12}, (p_{01}, p_{02}), E)$ *definiert mit:* TODO: erzwungenen Zeilenumbrüche kontrollieren

- $I = (I_1 \cup I_2) \backslash (O_1 \cup O_2)$,
- $O = (O_1 \cup O_2)$,
- $$\begin{aligned} \longrightarrow_{12} = &\left\{((p_1, p_2), \alpha, (p_1', p_2)) \mid p_1 \overset{\alpha}{\longrightarrow}_1 p_1', \alpha \in \Sigma_\tau \backslash \mathrm{Synch}(P_1, P_2)\right\} \\ &\cup \left\{((p_1, p_2), \alpha, (p_1, p_2')) \mid p_2 \overset{\alpha}{\longrightarrow}_2 p_2', \alpha \in \Sigma_\tau \backslash \mathrm{Synch}(P_1, P_2)\right\} \\ &\cup \left\{((p_1, p_2), \alpha, (p_1', p_2')) \mid p_1 \overset{\alpha}{\longrightarrow}_1 p_1', p_2 \overset{\alpha}{\longrightarrow}_2 p_2', \alpha \in \mathrm{Synch}(P_1, P_2)\right\}, \end{aligned}$$
- $$\begin{aligned} \dashrightarrow_{12} = &\left\{((p_1, p_2), \alpha, (p_1', p_2)) \mid p_1 \overset{\alpha}{\dashrightarrow}_1 p_1', \alpha \in \Sigma_\tau \backslash \mathrm{Synch}(P_1, P_2)\right\} \\ &\cup \left\{((p_1, p_2), \alpha, (p_1, p_2')) \mid p_2 \overset{\alpha}{\dashrightarrow}_2 p_2', \alpha \in \Sigma_\tau \backslash \mathrm{Synch}(P_1, P_2)\right\} \\ &\cup \left\{((p_1, p_2), \alpha, (p_1', p_2')) \mid p_1 \overset{\alpha}{\dashrightarrow}_1 p_1', p_2 \overset{\alpha}{\dashrightarrow}_2 p_2', \alpha \in \mathrm{Synch}(P_1, P_2)\right\}, \end{aligned}$$

- $E = (P_1 \times E_2) \cup (E_1 \times P_2)$ } *geerbte Fehler*
  $\cup \left\{(p_1, p_2) \mid \exists a \in O_1 \cap I_2 : p_1 \overset{a}{\dashrightarrow}_1 \wedge p_2 \overset{a}{\not\rightarrow}_2\right\}$
  $\cup \left\{(p_1, p_2) \mid \exists a \in I_1 \cap O_2 : p_1 \overset{a}{\not\rightarrow}_1 \wedge p_2 \overset{a}{\dashrightarrow}_2\right\}$ } *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 zu Inputs (im Fall* $(I_1 \cap I_2)$*) bzw. Outputs (alle anderen Fälle) der Komposition.*
P_1 *ist ein* Partner *von* P_2*, wenn* P_1 *und* P_2 *duale Signaturen* $\text{Sig}(P_1) = (I, O)$ *und* $\text{Sig}(P_2) = (O, I)$ *haben.*

Die Definition der Parallelkomposition sagt aus, dass 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.
Durch die Synchronisation von Inputs kann es zu keinen neuen Fehler kommen, da die Inputs in beiden Transitionssystemen keine lokal kontrollierten Aktionen sind. Falls jedoch nur eines der Transitionssysteme einen Input sicherstellt, der synchronisiert wird, wird dieser Input in der Parallelkomposition nicht mehr sicher gestellt. Es kann also in der Kommunikation mit einem weiteren MEIO dort zu einem neuen Fehler kommen.

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

1. $p \notin E_P$*,*
2. *falls* $q \xrightarrow{\alpha}_Q q'$ *gilt, gibt es eine Transition* $p \xrightarrow{\alpha}_P p'$ *für ein* p' *mit* $p' \mathcal{R} q'$*,*
3. *falls* $p \overset{\alpha}{\dashrightarrow}_P p'$ *gilt, gibt es eine Transition* $q \overset{\alpha}{\dashrightarrow}_Q q'$ *für ein* q' *mit* $p' \mathcal{R} q'$*.*

Die Vereinigung $\sqsubseteq_{\text{as}}$ *aller dieser Simulations-Relationen wird als* (starke) as-Verfeinerung(-s Relation) *(auch modal Verfeinerung) bezeichnet. Falls für die Startzustände von zwei MEIOs* $p_0 \sqsubseteq_{\text{as}} q_0$ *gilt, wird auch* $P \sqsubseteq_{\text{as}} Q$ *für die Transitionssysteme geschrieben.* $P \sqsubseteq_{\text{as}} Q$ *bedeutet, 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* 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 Simulations-Relation $\mathcal{R}$ gelten muss, dass aus $q \notin E_Q$ folgt, dass auch p nicht in E_P enthalten ist, gilt auch die Implikation $p \in E_P \Rightarrow q \in E_Q$.

Für LTS (Labelled Transition Systems), die nach Definition keine Modalitäten und keine Fehler-Zustände enthalten, entspricht die as-Verfeinerung einer Bisimulation. Dafür müssen die Transitionen eines LTS als must-Transitionen aufgefasst werden. Man kann also auf LTS mit einer as-Verfeinerungs-Relation zwischen zwei Systemen deren Äquivalenz beweisen.

TODO: falls möglich passendes Quellen Beispiel einfügen

Auf den EIO, die z.B. in [Sch16] betrachtet wurden, lässt die as-Verfeinerungs-Relation zu, dass es in einer Verfeinerung möglicherweise weniger Fehler gibt und zusätzliches Verhalten, dass die Spezifikation nicht hatte. Die EIOs entsprechen Implementierungen von MEIOs, es ist also möglich, eine Implementierung mit Fehler durch eine andere as zu verfeinern, die keine Fehler enthält, aber potentiell zusätzliches Verhalten aufweist.

TODO: Beispiel einfügen

Falls eine Implementierung bereits frei von Fehler-Zuständen ist, entspricht sie einem LTS und kann somit nur noch durch äquivalente Implementierungen „verfeinert“ werden.

Definition 1.4 (*schwache alternierende Simulation*). *Eine Relation $\mathcal{R} \subseteq P \times Q$ auf zwei MEIOs P und Q mit gleicher Signatur ist eine* schwache alternierende Simulation*, 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. *falls* $q \xrightarrow{i}_Q q'$ *gilt, gibt es eine Transition* $p \xrightarrow{i}_P \overset{\varepsilon}{\Longrightarrow}_P p'$ *für ein* p' *mit* $p' \mathcal{R} q'$,
3. *falls* $q \xrightarrow{\omega}_Q q'$ *gilt, gibt es eine Transition* $p \overset{\hat{\omega}}{\Longrightarrow}_P p'$ *für ein* p' *mit* $p' \mathcal{R} q'$,
4. *falls* $p \overset{i}{\dashrightarrow}_P p'$ *gilt, gibt es eine Transition* $q \overset{i}{\dashrightarrow}_Q \overset{\varepsilon}{\Longrightarrow}_Q q'$ *für ein* q' *mit* $p' \mathcal{R} q'$,
5. *falls* $p \overset{\omega}{\dashrightarrow}_P p'$ *gilt, gibt es eine Transition* $q \overset{\hat{\omega}}{\Longrightarrow}_Q q'$ *für ein* q' *mit* $p' \mathcal{R} q'$.

Analog zur starken alternierenden Simulation, wird hier $\sqsubseteq_{\text{w-as}}$ als Relationssymbol für die Vereinigung aller schwachen Simulations-Relationen verwendet und man kann auch entsprechend schwache as-Verfeinerungen *definieren.*
Ebenso kann $\sqsubseteq_{\text{w-as}}$ für ein MEIO Q und eine Implementierung P definiert werden mit $P \sqsubseteq_{\text{w-as}} Q$, 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.* TODO: ausführen oder umformulieren

Die schwache Simulation erlaubt interne Aktionen des MEIO, das eine Aktion matchen muss. Jedoch ist es zwingend notwendig, dass ein Input sofort ausgeführt wird und erst dann interne Aktinen möglich sind, da ein Input die Reaktion auf eine Aktion ist, die die Umwelt auslöst, welche nicht auf das Transitionssystem warten kann. Outputs hingegen können auch verzögert werden, da die Umgebung diese dann als Inputs aufnimmt und für die Umgebung diese Aktionen nicht lokal kontrolliert sind.
Auch für alle Tupel (p, q) in einer schwach alternierenden Simulations-Relation $\mathcal{R}$ gilt $p \in E_P \Rightarrow q \in E_Q$.

Wie üblich kann man zeigen, dass $\sqsubseteq_{\text{as}}$ (bzw. $\sqsubseteq_{\text{w-as}}$) eine starke (bzw. schwache) alternierende Simulation ist und die Eigenschaft der Transitivität erfüllt. Die Beweise für diese Aussagen sollen hier entfallen.

Die in der folgende Definition vorgestellte 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:* $W_1 \| W_2 := \bigcup \{w_1 \| w_2 \mid w_1 \in W_1 \wedge w_2 \in W_2\}$.

w_1 und w_2 sind für gewöhnlich Traces von MEIO, die Komponiert werden. Das Wort $w_1 \| w_2$ ist dann eine Aktionsfolge aus Input und Outputs. Es existiert in einer Parallelkomposition ein Transitionsfolge vom Startzustand aus, die mit den Aktionen aus $w_1 \| w_2$ beschriftet ist. Der Trace $w_1 \| w_2$ der Parallelkomposition kann also auch als Wort aufgefasst werden, dass verarbeitet wird während es Ablauf des Systems.

Definitionen der Funktionen prune und cont zum Abschneiden und Verlängern von Traces können ebenso wie die letzte Definition aus z.B. [BV15a] 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*).

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

Definition 1.7 (*Sprache*). *Die* Sprache *eines MEIOs* P *ist definiert als:*

$$L(P) := \left\{ w \in \Sigma^* \mid p_0 \overset{w}{\Rightarrow}_P \right\}.$$

Somit entspricht die Sprache einer Implementierung der Definition aus [Sch16] für EI-Os. Jedoch muss die Sprache einer as-Verfeinerung eines 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 das MEIO bereits frei von Fehler-Zuständen ist, ist seine Sprache die Vereinigung der Sprachen all seine möglichen as-Implementierungen.
Von der Sprache einer as-Verfeinerung eines MEIOs kann man nur wenig Rückschlüsse 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.

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

1.2 Allgemeine Folgerungen

Proposition 1.8 (*Sprache und Implementierungen*). *Für die Sprache eines MEIOs P gilt* $L(P) \subseteq \bigcup_{P' \in \text{as-impl}(P)} L(P')$.

Beweis.
Sei P' die as-Implementierung von P, die alle may- und must-Transitionen von P implementiert. Die Definition von P' lautet also:

- $P' = P$,
- $p'_0 = p_0$,
- $I_{P'} = I_P$ und $O_{P'} = O_P$,
- $\longrightarrow_{P'} = \dashrightarrow_{P'} = \dashrightarrow_P$,
- $E_{P'} = \emptyset$.

Die entsprechende starke as-Verfeinerungs-Relation $\mathcal{R}$, die zwischen P' und P gilt, ist die Identitäts-Relation zwischen den Zuständen der Transitionssysteme. Für 1.3.3 betrachten wir $(p, p) \in \mathcal{R}$ mit $p \overset{\alpha}{\dashrightarrow}_{P'} p'$ in P'. Diese Transition muss auch in P existieren und da $\mathcal{R}$ die Identitäts-Relation ist auch $(p', p') \in \mathcal{R}$ gelten. Alle must-Transitionen in P haben zugrunde liegende may-Transitionen. Es gibt also zu jeder must-Transition $p \xrightarrow{\alpha}_P p'$ in P auch die Transition $p \xrightarrow{\alpha}_{P'} p'$. Es folgt dann auch $(p', p') \in \mathcal{R}$ und dadurch ist auch 1.3.2 erfüllt. Der erste Punkt von 1.3 gilt, da $E_{P'}$ leer ist.
Für alle $w \in L(P) = \left\{ w \in \Sigma^* \mid p_0 \overset{w}{\Longrightarrow}_P \right\}$ gilt $w \in L(P') = \left\{ w \in \Sigma^* \mid p'_0 \overset{w}{\Longrightarrow}_{P'} \right\}$, da alle Transitionen von P in P' implementiert werden. □

Proposition 1.9 (*Sprache der Parallelkomposition*). *Für zwei komponierbare MEIOs P_1 und P_2 gilt:* $L_{12} := L(P_{12}) = L_1 \| L_2$.

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 $L_1 \| L_2$ auch Wörter der Parallelkomposition $L(P_{12})$. □

Lemma 1.10 (*w-as-Verfeinerung und Parallelkomposition*). *Seien P_1 und P_2 komponierbar MEIOs und P_1' eine schwache as-Verfeinerung von P_1 ist mit der schwachen as-Verfeinerungs-Relation $\mathcal{R}_1$. Dann gelten für die Relation $\mathcal{R}_{12} = \{((p_1', p_2), (p_1, p_2)) \mid (p_1', p_1) \in \mathcal{R}_1, p_2 \in P_2\}$ die Aussagen 2. bis 5. aus der Definition 1.4.*

Beweis. Um zu beweisen, dass $\mathcal{R}_{12}$ die Aussagen 2. bis 5. der Definition 1.4 erfüllt, müssen diese geprüft werden.
Für alle folgenden Fälle wird $((p_1', p_2), (p_1, p_2))$ immer aus $\mathcal{R}_{12}$ mit $(p_1, p_2) \notin E_{12}$ gewählt.

2. Aus der Definition der schwachen alternierenden Simulation in 1.4 folgt, dass für diesen Punkt zu zeigen ist: $(p_1, p_2) \xrightarrow{i}_{12} (q_1, q_2)$ 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 $((q_1', q_2), (q_1, q_2)) \in \mathcal{R}_{12}$.
   Die i-must-Transition in $P_1\|P_2$ kann entweder aus der Synchronisation von zwei must-Inputs entstanden sein oder als unsynchronisierte Aktion aus einem der komponierten MEIOs übernommen worden sein.

   - Fall 1 ($i \notin \text{Synch}(P_1, P_2)$): Für den Fall i in I_2 ist der Input in P_2 via must-Transition ausführbar und somit sowohl in der Parallelkomposition $P_1'\|P_2$ wie auch in $P_1\|P_2$ als must-Transition vorhanden. Es gilt dann auch $p_1 = q_1$ und $p_1' = q_1'$, da i kein Input von P_1 ist. Es gilt also die geforderte Transitionsfolge und das geforderte Element der Relation $\mathcal{R}_{12}$ direkt durch die Voraussetzungen.
     Für den Rest dieses Punktes wird davon ausgegangen, dass i in I_1 enthalten ist. Es muss also in P_1 die i-Transition als must-Transition von p_1 ausgehen ($p_1 \xrightarrow{i}_1 q_1$). 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 ist und $q_1'\mathcal{R}_1 q_1$ gelten muss. $p_2 = q_2 \in P_2$ muss erfüllt sein, da i nicht in Σ_2 enthalten ist und p_2 nach Voraussetzung ein Zustand von P_2 sein muss. In der Komposition von P_1' mit P_2 entsteht die Transitionsfolge $(p_1', p_2) \xrightarrow{i}_{P_1'\|P_2} \xRightarrow{\varepsilon}_{P_1'\|P_2} (q_1', q_2)$. Mit der Definition von $\mathcal{R}_{12}$ kann daraus $((q_1', q_2), (q_1, q_2)) \in \mathcal{R}_{12}$ gefolgert werden.

   - 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 $(p_1, p_2) \xrightarrow{i}_{12} (q_1, q_2)$ in der Komposition möglich zu machen, muss in beiden Transitionssystemen P_j $p_j \xrightarrow{i}_j q_j$ gelten. Durch $\mathcal{R}_1$ und die Definition 1.4.2 folgt $p_1' \xrightarrow{i}_{P_1'} \xRightarrow{\varepsilon}_{P_1'} q_1'$ mit $(q_1', q_1) \in \mathcal{R}_1$. Daraus ergibt sich $((q_1', q_2), (q_1, q_2)) \in \mathcal{R}_{12}$ mit der Definition von $\mathcal{R}_{12}$. Durch die Synchronisation der i-Inputs 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 $(p_1, p_2) \xrightarrow{\omega}_{12} (q_1, q_2)$ 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 ($\omega \notin \text{Synch}(P_1, P_2)$): Der Fall ω ist eine Aktion, die in P_2 ausgeführt wird, verläuft analog zum Fall i in I_2 im Fall 1 des zweiten Punkt dieses Beweises. Es sei also im folgenden ω ein Output oder eine interen Aktion von P_1. Um in der Komposition $P_1 \| P_2$ die must-Transition zu erhalten muss bereits für die Transition in P_1 $p_1 \xrightarrow{\omega}_1 q_1$ gelten sowie $p_2 = q_2$ in P_2. Mit 1.4.3 kann für $\mathcal{R}_1$ gefolgert werden, dass $p_1' \stackrel{\hat{\omega}}{\Longrightarrow}_{P_1'} q_1'$ mit $(q_1', q_1) \in \mathcal{R}_1$ gilt. In der Komposition folgt dann $(p_1', p_2) \stackrel{\hat{\omega}}{\Longrightarrow}_{P_1' \| P_2} (q_1', q_2)$. Zusätzlich gilt auch die Zugehörigkeit des Zustands-Tupels $((q_1', q_2), (q_1, q_2))$ zur 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 können, muss in diesem Fall $\omega \neq \tau$ gelten. Um einen Output ω in der Parallelkomposition von P_1 und P_2 zu erhalten, muss entweder $\omega \in I_1 \cap O_2$ oder $\omega \in O_1 \cap I_2$ gelten. Für beide Fälle müssen die Transitionen $p_1 \xrightarrow{\omega}_1 q_1$ und $p_2 \xrightarrow{\omega}_2 q_2$ in den einzelnen Komponenten enthalten sein. Mit $\mathcal{R}_1$ und 1.4.2 folgt im Fall $\omega \in I_1$ $p_1' \xrightarrow{\omega}_{P_1'} \stackrel{\varepsilon}{\Longrightarrow}_{P_1'} q_1'$ und $q_1' \mathcal{R}_1 q_1$. Im Fall $\omega \in O_1$ erhält man durch $\mathcal{R}_1$ und 1.4.3 die Transition $p_1' \stackrel{\omega}{\Longrightarrow}_{P_1'} q_1'$ mit $q_1' \mathcal{R}_1 q_1$. Da ω in beiden Fällen keine interne Aktion ist, gilt $\omega = \hat{\omega}$. In der Parallelkomposition von P_1' und P_2 werden zuerst die internen Aktionen von P_1' ausgeführt, falls diese existieren, bis dort die Aktion ω erreicht ist, dann wird ω synchronisiert und danach werden die restlichen internen Aktionen ausgeführt, bis man beim Zuständen q_1 angekommen ist. Es ergibt also die Transitionsfolge $(p_1', p_2) \stackrel{\hat{\omega}}{\Longrightarrow}_{P_1' \| P_2} (q_1', q_2)$ und das Tupel $((q_1', q_2), (q_1, q_2))$ in der Relation $\mathcal{R}_{12}$.

4. $(p_1', p_2) \overset{i}{\dashrightarrow}_{P_1' \| P_2} (q_1', q_2)$ impliziert $(p_1, p_2) \overset{i}{\dashrightarrow}_{12} \stackrel{\varepsilon}{\Longrightarrow}_{12} (q_1, q_2)$ 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, bis auf die Erfüllung von 1. aus der Definition 1.4, ist.
   Die Transition i kann durch Synchronisation von zwei Transitionen entstanden sein oder durch eine Transition aus einer der beiden Komponenten mit der Voraussetzung $i \notin \text{Synch}(P_1', P_2)$.

   - Fall 1 ($i \notin \text{Synch}(P_1', P_2)$): Der Fall i in I_2 verläuft analog zum selben Fall im Fall 1 des Beweis des zweiten Punktes. Es muss nur must durch may ersetzt werden. Es kann also für den Rest diese Punktes davon ausgegangen werden, dass i in I_1 enthalten ist. Es muss in P_1' eine ausgehende i-Transition von Zustand p_1' geben, so dass $p_1' \overset{i}{\dashrightarrow}_1 q_1'$ gilt. Mit der Relation $\mathcal{R}_1$ und 1.4.4 folgt, dass in P_1 i als schwache Transition der Form $p_1 \overset{i}{\dashrightarrow}_1 \stackrel{\varepsilon}{\Longrightarrow}_1 q_1$ ausführbar sein und $q_1' \mathcal{R}_1 q_1$ gelten muss. Mit der Definition von $\mathcal{R}_{12}$ kann dann $((q_1', q_2), (q_1, q_2)) \in \mathcal{R}_{12}$ gefolgert werden. In der Parallelkomposition von P_1 und P_2 entsteht die Transitionsfolge $(p_1, p_2) \overset{i}{\dashrightarrow}_{12} \stackrel{\varepsilon}{\Longrightarrow}_{12} (q_1, q_2)$ mit $p_2 = q_2$.

   - 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 $(p_1', p_2) \overset{i}{\dashrightarrow}_{P_1' \| P_2} (q_1', q_2)$ in der Komposition möglich

zu machen, muss in den Transitionssystemen P_1' und P_2 $p_1' \overset{i}{\dashrightarrow}_{P_1'} q_1'$ bzw. $p_2 \overset{i}{\dashrightarrow}_2 q_2$ gelten. Durch $\mathcal{R}_1$ und die Definition 1.4.4, die für diese Relation gilt, folgt $p_1 \overset{i}{\dashrightarrow}_1 \overset{\varepsilon}{\Rightarrow}_1 q_1$ mit $(q_1', q_1) \in \mathcal{R}_1$. Es gilt also $((q_1', q_2), (q_1, q_2)) \in \mathcal{R}_{12}$. Durch die Synchronisation des Inputs i in der Komposition von P_1 und P_2 ergibt sich $(p_1, p_2) \overset{i}{\dashrightarrow}_{12} \overset{\varepsilon}{\Rightarrow}_{12} (q_1, q_2)$.

5. Analog zu 3. und 4. kann für diesen Punkt $(p_1', p_2) \overset{\omega}{\dashrightarrow}_{P_1' \| P_2} (q_1', q_2)$ impliziert $(p_1, p_2) \overset{\hat{\omega}}{\Rightarrow}_{12} (q_1, q_2)$ 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 ($\omega \notin \text{Synch}(P_1', P_2)$): Im Fall ω ist eine Aktion von P_2 folgt das zu zeigende direkt aus den Voraussetzungen, ebenso wie in allen vorangegangenen Punkten. Somit wird im folgenden davon ausgegangen, dass ω in O_1 enthalten oder eine interne Aktion ist, die von P_1' geerbt wurde. Um in $P_1' \| P_2$ die may-Transition zu erhalten muss also bereits in P_1' die Transition $p_1' \overset{\omega}{\dashrightarrow}_{P_1'} q_1'$ möglich gewesen sein. Mit 1.4.5 kann für $\mathcal{R}_1$ gefolgert werden, dass $p_1 \overset{\hat{\omega}}{\Rightarrow}_1 q_1$ mit $(q_1', q_1) \in \mathcal{R}_1$ gilt. Für die Komposition folgt daraus $(p_1, p_2) \overset{\hat{\omega}}{\Rightarrow}_{12} (q_1, q_2)$ mit $p_2 = q_2$. Es gilt auch die Zugehörigkeit des Zustands-Tupels $((q_1', q_2), (q_1, q_2))$ zur Relation $\mathcal{R}_{12}$.
   - Fall 2 ($\omega \in \text{Synch}(P_1', P_2)$): Es muss $\omega \neq \tau$ gelten und somit können die Fälle $\omega \in I_1 \cap O_2$ und $\omega \in O_1 \cap I_1$ unterschieden werden. Es folgt in beiden Fällen $p_1' \overset{\omega}{\dashrightarrow}_{P_1'} q_1'$ und $p_2 \overset{\omega}{\dashrightarrow}_2 q_2$. Mit $\mathcal{R}_1$ und 1.4.4 folgt im Fall $\omega \in I_1$ $p_1 \overset{\omega}{\dashrightarrow}_1 \overset{\varepsilon}{\Rightarrow}_1 q_1$ und $q_1' \mathcal{R}_1 q_1$. Im Fall $\omega \in O_1$ wendet man $\mathcal{R}_1$ mit 1.4.5 an und erhält $p_1 \overset{\omega}{\Rightarrow}_1 q_1$ und $q_1' \mathcal{R}_1 q_1$. Da ω eine sichtbare Aktion ist, gilt $\omega = \hat{\omega}$. In der Parallelkomposition von P_1 und P_2 werden zuerst mögliche internen Aktionen von P_1 ausgeführt, bis dort die sichtbare Aktion erreicht ist, dann wird ω synchronisiert und danach werden die restlichen internen Aktionen ausgeführt, bis man beim Zustand q_1 angekommen ist. Es ergibt sich also die Transitionsfolge $(p_1, p_2) \overset{\hat{\omega}}{\Rightarrow}_{12} (q_1, q_2)$ und das Tupel $((q_1', q_2), (q_1, q_2))$ in der Relation $\mathcal{R}_{12}$.

□

Proposition 1.11 (*w-as-Verfeinerung und Parallelkomposition*). *Für zwei komponierbare MEIOs P_1 und P_2 und eine schwache as-Verfeinerung P_1' von P_1 muss $P_1' \| P_2$ keine schwache as-Verfeinerung von $P_1 \| P_2$ sein. Spezielle erfüllt die Relation $\mathcal{R}_{12}$ aus dem Lemma 1.10 den ersten Punkt der Definition 1.4 im allgemeinen nicht.*

Beweis. Die Aussage 1. der Definition 1.4 ist im Allgemeinen für die Relation $\mathcal{R}_{12}$ aus 1.10 nicht erfüllt. Es gilt also für ein $((p_1', p_2), (p_1, p_2))$ aus $\mathcal{R}_{12}$ mit $(p_1, p_2) \notin E_{12}$ nicht unbedingt, dass (p_1', p_2) kein Element der Menge $E_{P_1' \| P_2}$ ist. $P_1' \| P_2$ muss also keine schwache as-Verfeinerung von $P_1 \| P_2$ sein.

Die Voraussetzung besagt, dass $(p_1, p_2) \notin E_{12}$ für das Zustands-Tupel $((p_1', p_2), (p_1, p_2))$ aus $\mathcal{R}_{12}$ gilt. Nach Definition von $\mathcal{R}_{12}$ erhält man $(p_1', p_1) \in \mathcal{R}_1$ und $p_2 \in P_2$. Die p_j $(j \in \{1, 2\})$ dürfen keine Fehler-Zustände sein, da sonst auch (p_1, p_2) ein solcher wäre. Somit folgt mit Definition 1.4.1 auch $p_1' \notin E_1$. Die Zustände p_1' und p_2 vererben also keinen Fehler. Jedoch könnte (p_1', p_2) aufgrund eines nicht erzwungenen Inputs ein neuer Fehler-Zustand sein. Der nicht sichergestellte Input kann in beiden Systemen auftreten. Für den Fall, dass P_1' einem vom Zustand p_1' ausgehenden Output hat, für den P_2 im Zustand p_2 nicht den passenden Input sicherstellt gilt $p_1' \overset{a}{\dashrightarrow}_{P_1'}$ und $p_2 \overset{a}{\not\dashrightarrow}_2$ für ein a aus $O_1 \cap I_2$. $\mathcal{R}_1$ erzwingt nach Definition nur die schwache Ausführbarkeit des Outputs a in P_1 vom Zustand p_1 ausgehend, d.h. $p_1 \overset{a}{\Rightarrow}_1$. Dadurch kann es in der Parallelkomposition von $P_1' \| P_2$ zu einem neuen Kommunikationsfehler kommen, der in $P_1 \| P_2$ keiner ist.
Um zu zeigen, dass dieser Fall wirklich auftreten kann, ist ein Beispiel mit diesem Verhalten in Abbildung 1.1 dargestellt. Dabei soll a im Schnitt der Outputs O_1 von P_1 bzw. P_1' und der Inputs I_2 von P_2 enthalten sein. Die Relation $\mathcal{R}_1$ enthält die Zustands-Tupel (p_{01}', p_{01}) und (p_1', p_1). Somit ist $((p_{01}', p_{02}), (p_{01}, p_{02}))$ in $\mathcal{R}_{12}$ enthalten und es gilt $(p_{01}, p_{02}) \notin E_{12}$. Jedoch ist (p_{01}', p_{02}) trotzdem ein Fehler-Zustand in der Parallelkomposition von P_1' und P_2.
Es kann auch keine andere schwache as-Verfeinerungs-Relation $\mathcal{R}$ für $P_1' \| P_2$ und $P_1 \| P_2$ geben, da $(P_1' \| P_2) \mathcal{R} (P_1 \| P_2)$ nur gilt, wenn die Startzustände der Transitionssysteme in der Relation $\mathcal{R}$ stehen. Das Tupel $((p_{01}', p_{02}), (p_{01}, p_{02}))$ muss also in $\mathcal{R}$ enthalten sein. Für diese Tupel ist jedoch der erste Punkt der Definition 1.4 nicht erfüllt mit der analogen Begründung wie für die Relation $\mathcal{R}_{12}$.

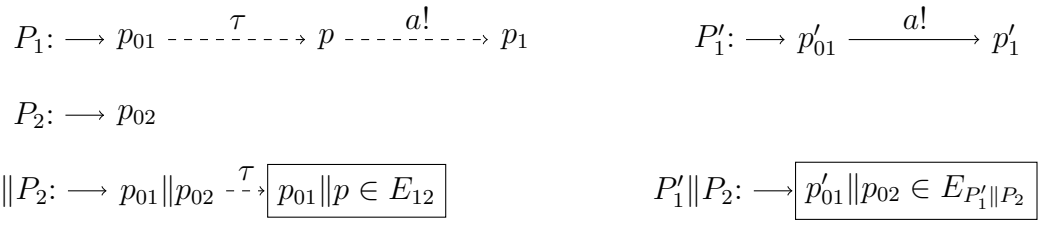

Abbildung 1.1: Gegenbeispiel für 1. von $\mathcal{R}_{12}$ bzgl. Definition 1.4 mit $a \in O_1 \cap I_2$

Es wäre auch möglich, dass P_1 ebenfalls eine Implementierung ist. Die must-τ-Transition würde dann eine schwache Implementierung in P_1' fordern, jedoch muss es dafür keine echte Transition in P_1' geben, da τ die interne Aktion ist. Die Implementierung von a würde ebenfalls schwach gefordert werden, ist jedoch bereits stark vorhanden. $P_1 \| P_2$ würde mit der must-τ-Transition dann ebenfalls zu einer Implementierung werden. Die must-a-Transition in P_1' könnte auch eine may-Transition sein, solange die a-Transition in P_1 keine must-Transition ist.
Um dieses Problem zu lösen könnte man die Definition der Parallelkomposition verändern. Es wäre denkbar, dass alle Zustände, die lokal Fehler-Zustände erreichen können ebenfalls bereits als Fehler angesehen werden. Jedoch wird erwartet dass die Definition der Parallelkomposition dann eine stärkere Forderung an die Transitionssysteme stellt. Für Implementierungen wäre die Forderung sogar stärkere, wie die der EIOs in

z.B. [Sch16]. Dieser Ansatz käme jedoch dem Vorgehen des Abschneidens der Fehler-Zustände mit ihren lokalen Vorgängern in [BFLV16] näher. □

Korollar 1.12 (*as-Verfeinerungen und Parallelkomposition*). *Für zwei komponierbar MEIOs P_1 und P_2 gilt, falls P_1' eine as-Verfeinerung von P_1 ist, dann ist auch $P_1'\|P_2$ eine as-Verfeinerung von $P_1\|P_2$.*

Beweis. Falls die Relation $\mathcal{R}_1$ aus dem Lemma 1.10 keine schwachen as-Verfeinerungs-Relation sondern eine starke as-Verfeinerungs-Relation ist, ist auch $\mathcal{R}_{12}$ eine as-Verfeinerungs-Relation zwischen $P_1'\|P_2$ und $P_1\|P_2$. Dazu ist also nur zu zeigen, wie aus den einzelnen Beweispunkten des Beweises von 1.10 folgt, dass $\mathcal{R}_{12}$ eine starke as-Verfeinerungs-Relation ist und dass hier zusätzlich der erste Punkt erfüllt ist. Es wird hier ebenso für alle Punkte jeweils ein $((p_1', p_2), (p_1, p_2))$ aus $\mathcal{R}_{12}$ mit $(p_1, p_2) \notin E_{12}$ gewählt.

1. Dieser Punkt kann im Gegensatz zum ersten Punkt der Definition 1.4 für die schwache as-Verfeinerungs-Relation $\mathcal{R}_{12}$ aus Lemma 1.10 für die starke as-Verfeinerungs-Relation bewiesen werden. Dies ist möglich, da für p_1 im Falle eines Outputs a dieser nicht nur schwach sondern direkt ausführbar ist. Es ist also zu zeigen, dass (p_1', p_2) kein Element von $E_{P_1'\|P_2}$ ist.
   In dem man auf $\mathcal{R}_{12}$ die Definition anwendet, erhält man $(p_1', p_1) \in \mathcal{R}_1$ und $p_2 \in P_2$. Die p_j dürfen beide keine Fehler-Zustände sein, da sonst auch (p_1, p_2) ein solcher wäre. Somit folgt mit Definition 1.3.1 $p_1' \notin E_{P_1'}$. Die Zustände p_1' und p_2 in Parallelkomposition können also keinen geerbten Fehler produzieren. Jedoch könnte (p_1', p_2) aufgrund eines nicht sichergestellten Inputs ein neuer Fehler-Zustand sein. Dafür müsste entweder $p_1' \overset{a}{\not\rightarrow}_{P_1'}$ und $p_2 \overset{a}{\dashrightarrow}_2$ für ein a aus $I_1 \cap O_2$ oder $p_1' \overset{a}{\dashrightarrow}_{P_1'}$ und $p_2 \overset{a}{\not\rightarrow}_2$ für ein a aus $O_1 \cap I_2$ gelten. Mit 1.3.2 und $\mathcal{R}_1$ folgt im Fall $a \in I_1$ $p_1 \overset{a}{\not\rightarrow}_1$. $\mathcal{R}_1$ erzwingt mit 1.3.3 die direkte Ausführbarkeit des Outputs a in P_1 im Fall $a \in O_1$, d.h. $p_1 \overset{a}{\dashrightarrow}_1$. Somit müsste auch $(p_1, p_2) \in E_{12}$ in beiden Fällen gelten, was ein Widerspruch zur Voraussetzung wäre. (p_1', p_2') kann also weder ein geerbter noch ein neuer Fehler in $P_1'\|P_2$ sein und deshalb gilt $(p_1', p_2) \notin E_{P_1'\|P_2}$.

2. α kann sowohl Input, Output wie auch interen Aktion sein. Um diesen Punkt zu beweisen muss man 2. und 3. aus dem Beweis von Lemma 1.10 kombinieren. Da $\mathcal{R}_1$ die Transition in P_1' ohne zusätzliche τ-Transitionen fordern, entstehen keine schwachen Transitionen für die αs und somit ist α auch in der Parallelkomposition $P_1'\|P_2$ eine direkte Transition ohne zusätzliche τs. Es folgt das $\mathcal{R}_{12}$ die Forderungen für die strake as-Verfeinerungs-Relation dieses Punktes erfüllt.

3. Hierfür werden die Punkte 3. und 4. aus dem Beweis des Lemmas 1.10 kombiniert. Analog wie bei 2. diese Beweises fallen die zusätzlichen τ-Transitionen durch die stärkere Forderung an $\mathcal{R}_1$ weg. Dieser Punkt gilt also ebenfalls.

□

Die drei vorangegangenen Ergebnisse fordern nur die Verfeinerung der ersten Komponente. Die Parallelkomposition wurde so definiert, dass sie kommutativ ist. Somit

ist ebenso die Verfeinerung der zweiten Komponente möglich. Da man beide Komponenten nach einander verfeinern kann und jede Verfeinerung einer Verfeinerung auch eine Verfeinerung das ursprünglichen Systems ist, kann man auch beide Komponenten gleichzeitig verfeinern und erhält in der Parallelkomposition die gleiche Verfeinerung.

Korollar 1.13 (*as-Implementierungen und Parallelkomposition*). *Für zwei komponierbare MEIOs P_1 und P_2 gilt:* $P_1' \in \text{as-impl}(P_1) \wedge P_2' \in \text{as-impl}(P_2) \Rightarrow (P_1' \| P_2') \in \text{as-impl}(P_1 \| P_2)$.

Beweis. P_1' und P_2' sind aufgrund der Definition 1.3 auch starke as-Verfeinerungen von P_1 bzw. P_2. Somit ist die Parallelkomposition $P_1' \| P_2'$ auch eine starke as-Verfeinerung von $P_1 \| P_2$, nach Korollar 1.12. Für Implementierungen gilt $\longrightarrow = \dashrightarrow$. 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 $\longrightarrow_{P_1' \| P_2'} = \dashrightarrow_{P_1' \| P_2'}$ und somit ist $P_1' \| P_2'$ eine Implementierung und eine as-Verfeinerung von $P_1 \| P_2$. Dies entspricht der Definition der starken as-Implementierung, sodass $(P_1' \| P_2') \in \text{as-impl}(P_1 \| P_2)$ gilt. □

Für schwache as-Implementierungen kann es kein analoges Korollar zu 1.13 gegen, da die Verfeinerung im allgemeinen bereits scheitert (Proposition 1.11); man beachte auch die Diskussion des Beispiels 1.1. Die Parallelkomposition von Implementierungen ist jedoch immer eine Implementierung. Somit würde in den Fällen, in denen auch 1.4.1 erfüllt ist für die Parallelkomposition schwacher as-Implementierungen, eine analoge Aussage gelten.
Die umgekehrte Richtung von Korollar 1.12 gilt im allgemeinen nicht, d.h. es muss zu einer as-Verfeinerung P' einer Parallelkomposition $P_1 \| P_2$ keine as-Verfeinerungen P_1' und P_2' der einzelnen Komponenten geben, deren Parallelkomposition $P_1' \| P_2'$ der as-Verfeinerung der Parallelkomposition P' entsprechen. Die Problematik wird in Abbildung 1.2 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. Für eine as-Verfeinerung von P_2 ist es nur möglich, dass keine Transition umgesetzt wird oder die o' Transition entweder als may- oder must-Transition in die Verfeinerung übernommen wird. Somit kommt es in P' zu dem Problem, dass keine as-Verfeinerung von P_2 in Parallelkomposition mit der Implementierung P_1 den geforderten MEIO P' ergeben würde. Auch im Spezialfall von as-Implementierungen kann dieses Gegenbeispiel angewendet werden, da P' auch eine Implementierung von $P_1 \| P_2$ ist und es auch keine passende as-Implementierung von P_2 geben kann, wenn es schon keine passende Verfeinerung gibt.

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,

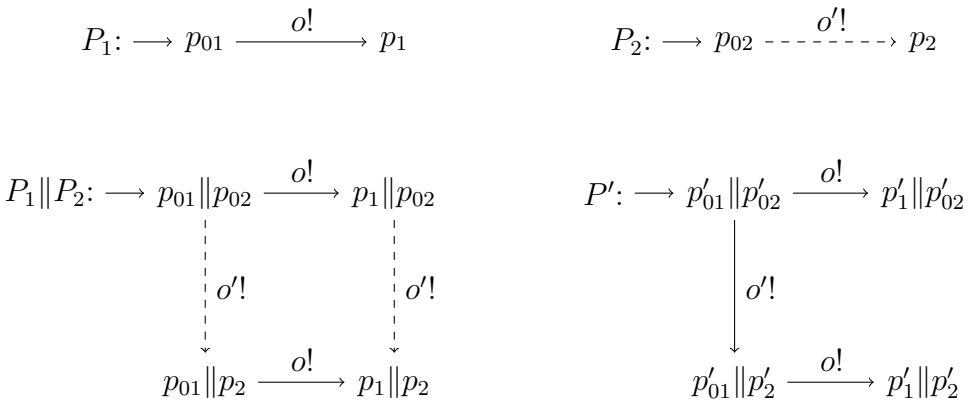

Abbildung 1.2: Gegenbeispiel für Umkehrung von Lemma 1.12

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

2 Verfeinerungen für Kommunikationsfehler-Freiheit

2.1 gröbster Präkongruenz-Ansatz

Dieses Kapitel hat das Ziel die Präkongruenz für Error bei EIOs aus z.B. [Sch16] auf die hier betrachten MEIOs zu erweitern.

Definition 2.1 (*fehler-freie Kommunikation*). *Ein Fehler-Zustand ist* lokal erreichbar *in einem MEIO P, wenn ein $w \in O^*$ existiert mit $p_0 \stackrel{w}{\Rightarrow}_P p \in E$. Ein MEIO P ist lokal fehler-frei, wenn $\neg\exists w \in O^* : p_0 \stackrel{w}{\Rightarrow}_P p \in E$.*
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 2.2 (*Kommunikationsfehler-Verfeinerungs-Basirelation*). *Für zwei MEIOs P_1 und P_2 mit der gleichen Signatur wird $P_1 \sqsubseteq_E^{\mathrm{B}} P_2$ 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 ebenfalls lokal erreichbar ist. Die $\sqsubseteq_E^{\mathrm{B}}$ stellt als Basisrelation eine* Verfeinerung *bezüglich* Kommunikationsfehler-Freiheit *dar.*
$\sqsubseteq_E^{\mathrm{C}}$ bezeichnet die vollständig abstrakte Präkongruenz *von $\sqsubseteq_E^{\mathrm{B}}$ bezüglich $\cdot\|\cdot$, d.h. die gröbste Präkongruenz bezüglich $\cdot\|\cdot$, die in $\sqsubseteq_E^{\mathrm{B}}$ enthalten ist.*

Für as-Implementierungen P_1 und P_2 entspricht $\sqsubseteq_E^{\mathrm{B}}$ der Basisrelation $\sqsubseteq_E^{\mathrm{B}}$ aus [Sch16].

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

Definition 2.3 (*Kommunikationsfehler-Traces*). *Für ein MEIO P wird definiert:*

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

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. Die nächste Proposition stellt eine Teilmengen-

Beziehung zwischen den Traces eines MEIOs und den Traces seiner as-Implementierungen dar. Die Teilmengen-Beziehung in die andere Richtung gilt im allgemeinen nicht, da as-Verfeinerungs-Relationen beliebiges Verhalten nach Fehler-Zuständen in der Spezifikation zulassen.

Proposition 2.4 (*Kommunikationsfehler-Traces und Implementierungen*). *Sei P ein MEIO.*

1. *Für die strikten Fehler-Traces von P gilt: $StET(P) \subseteq \bigcup_{P' \in \text{as-impl}(P)} StET(P')$.*
2. *Für die gekürzten Fehler-Traces von P gilt: $PrET(P) \subseteq \bigcup_{P' \in \text{as-impl}(P)} PrET(P')$.*
3. *Für Input-kritischen-Traces von P gilt: $MIT(P) \subseteq \bigcup_{P' \in \text{as-impl}(P)} MIT(P')$.*

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 $\mathcal{R}$ zwischen den beiden Transitionssystemen. P' implementiert wie im Beweis zu Proposition 1.8 alle Transitionen von P. Das P' wird hier jedoch im Gegensatz zu Beweis von 1.8 auch noch alle Fehler-Zustände aus P implementieren. Die entsprechende as-Verfeinerungs-Relation $\mathcal{R}$ ist hier ebenfalls die Identitäts-Relation zwischen den Zuständen der Transitionssysteme. Die Definition von P' lautet:

   - $P' = P$,
   - $p'_0 = p_0$,
   - $I_{P'} = I_P$ und $O_{P'} = O_P$,
   - $\longrightarrow_{P'} = \dashrightarrow_{P'} = \dashrightarrow_P$,
   - $E_{P'} = E_P$.

   Die Tupel, die von 1.3.2 und 3. als Elemente der as-Verfeinerungs-Relation $\mathcal{R}$ gefordert werden, sind bereits durch die Identitäts-Relation garantiert, wie im Beweis von 1.8. Für 1.3.1 muss für jedes in der as-Verfeinerungs-Relation $\mathcal{R}$ enthaltene Zustands-Paar gelten, wenn der Zustand aus P kein Fehler-Zustand ist, dann ist auch der Zustand aus P' keiner. Dies 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 Identitäts-Relation $\mathcal{R}$, die Zustände entsprechen sich also. Es gilt mit $E_{P'} = E_P$, dass auch der in P' erreichte Zustand ein Fehler-Zustand ist. Für die as-Implementierung P' von P und die Identitäts-Relation $\mathcal{R}$ als starke as-Verfeinerungs-Relation zwischen den Transitionssystemen gilt also $StET(P) = StET(P')$.

2. Da der erste Punkt dieser Proposition bereits beweisen 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 dies Punktes.

3. Auch für diese Inklusion wird eine starke as-Verfeinerungs-Relation $\mathcal{R}$ und eine Implementierung P' angegeben. Jedoch werden nicht wie bei 1. alle Transitionen von P in P' implementiert. Es wird auch für jedes 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 möglicher Schleifen in P auch nicht mehr die Identitäts-Relation als as-Verfeinerungs-Relation gewählt werden. Der Trace w wird entsprechend seiner Länge abgewickelt, so dass sicher gestellt wird, dass der Zustand am Ende dieses Traces wirklich einen fehlenden Input aufweist. Für das Abwickeln werden die Zustände entsprechend ihrer Position im Ablauf, auf dem w ausgeführt wird, durchnummeriert. Für ein w, für das $wa \in MIT(P)$, gilt: $\exists w' \in \Sigma_\tau^*, \exists \alpha_1, \alpha_2, \ldots, \alpha_n, \exists p_1, p_2, \ldots, p_n : \hat{w'} = w \wedge w' = \alpha_1\alpha_2 \ldots \alpha_n \wedge p_0 \overset{\alpha_1}{\dashrightarrow}_P p_1 \overset{\alpha_2}{\dashrightarrow}_P \ldots p_{n-1} \overset{\alpha_n}{\dashrightarrow}_P p_n \overset{a}{\not\longrightarrow}_P$. 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:

   - $P' = P \times \{0, 1, \ldots n\}$,
   - $p'_0 = (p_0, 0)$,
   - $I'_P = I_P$ und $O'_P = O_P$,
   - $$\longrightarrow'_P = \dashrightarrow'_P = \left\{((p, j), \alpha, (p', j+1)) \mid p \overset{\alpha}{\dashrightarrow}_P p', 0 \leq j < n\right\} \cup \left\{((p, j), \alpha, (p', j)) \mid p \overset{\alpha}{\longrightarrow}_P p', 0 \leq j \leq n\right\},$$
   - $E_{P'} = \emptyset$.

   Der Ablauf von w aus P wird in P' durch $(p_0, 0) \overset{\alpha_1}{\longrightarrow}_{P'} (p_1, 1) \overset{\alpha_2}{\longrightarrow}_{P'} \ldots (p_{n-1}, n-1) \overset{\alpha_n}{\longrightarrow}_{P'} (p_n, n)$ simuliert. w ist also in P' ausführbar. a ist für (p_n, n) nicht ausführbar, da in P für p_n $p_n \overset{a}{\not\longrightarrow}_P$ gilt und für die Zustände mit der Nummer n in P' nur die must-Transitionen implementiert werden. Die Transitionen $p \overset{\alpha}{\longrightarrow}_P p'$ aus P werden in P' durch die Transitionen $(p, j) \overset{\alpha}{\longrightarrow}_{P'} (p', j)$ gematched. Für die may-Transitionen $(p, j) \overset{\alpha}{\dashrightarrow}_{P'} (p', j+1)$ (bzw. (p', j)) in P' sind die entsprechenden matchenden Transitionen in P die Transition $p \overset{\alpha}{\dashrightarrow}_P p'$. 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 $\mathcal{R}$ angegeben werden kann, gilt die Inklusion.

□

Definition 2.5 (*Kommunikationsfehler-Semantik*). *Sei P ein MEIO.*

- *Die Menge der* Fehler-Traces *von P ist $ET(P) := \text{cont}(PrET(P)) \cup \text{cont}(MIT(P))$.*
- *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.
Im weiteren Verlauf wird immer wieder eine Aussage den Zusammenhang von Abläufen in as-Verfeinerungen und ihren Spezifikationen benötigt. Um nicht immer wieder ähnliche Argumentationen zu benötigen, wird diese Aussage nun hier allgemein bewiesen.

Proposition 2.6 (*Abläufe in as-Verfeinerungen und Spezifikationen*). *Sei P' eine as-Verfeinerung eines MEIOs P und $\mathcal{R}$ die as-Verfeinerungs-Relation zwischen den beiden Transitionssystemen. Ein Ablauf $p'_0 \overset{\alpha_1}{\dashrightarrow}_{P'} p'_1 \overset{\alpha_2}{\dashrightarrow}_{P'} \dots p'_{n-1} \overset{\alpha_n}{\dashrightarrow}_{P'} p'_n$ aus P' kann durch einen Ablauf $p_0 \overset{\alpha_1}{\dashrightarrow}_P p_1 \overset{\alpha_2}{\dashrightarrow}_P \dots p_{n-1} \overset{\alpha_n}{\dashrightarrow}_P p_n$ mit $p'_j \mathcal{R} p_j$ für alle $j \in \{0, 1, \dots n\}$ in P gematched werden oder es gibt wird ein p_j für $0 \leq j < n$ erreicht, dass in der Menge E_P enthalten ist. Aus $w \in L(P')$ folgt also $w \in L(P)$ oder $w \in \text{cont}(StET(P)) \subseteq ET(P)$.*

Beweis. Die Existenz des Traces in P soll induktiv über die einzelnen Transitionen bewiesen werden. Die entsprechenden Behauptung für die Induktion lautet: Für alle $j \in \{0, 1, \dots n-1\}$ mit $p'_j \mathcal{R} p_j$ gibt es eine Transition $p_j \dashrightarrow \alpha_{j+1} {}_P p_{j+1}$ mit $p'_{j+1} \mathcal{R} p_{j+1}$ oder p_j ist in E_P enthalten.
Die Verankerung $p'_0 \mathcal{R} p_0$ folgt direkt aus der Voraussetzung, das $\mathcal{R}$ eine as-Verfeinerungs-Relation für P' und P ist.
Es kann also nun davon ausgegangen werden, dass $p'_j \mathcal{R} p_j$ für ein j mit $0 \leq j < n$ erfüllt ist.

- Fall 1 ($p_j \in E_P$): Die Behauptung der Induktion ist bereits erfüllt.
- Fall 2 ($p_j \notin E_P$): In der Verfeinerung P' existiert die Transition $p'_j \overset{\alpha_{j+1}}{\dashrightarrow}_{P'} p'_{j+1}$. Mit Definition 1.3.3 folgt daraus die Existenz der Transition $p_j \overset{\alpha_{j+1}}{\dashrightarrow}_P p_{j+1}$ in der Spezifikation P mit $p'_{j+1} \mathcal{R} p_{j+1}$.

Für jedes w aus $L(P')$ gibt es einen Ablauf wie er in dieser Proposition vorausgesetzt wurde, so dass $w = (\alpha_1 \alpha_2 \dots \alpha_n)|_\Sigma$ gilt. Im Fall 1 wurden bis p_j die Aktionen α in ihrer Reihenfolge ausgeführt. Es gibt also ein Präfix v von w, dass in P zu einem Fehler-Zustand führt und somit $v \in StET(P)$ erfüllt. w ist eine Fortsetzung von v somit gilt die Aussage $w \in \text{cont}(StET(P)) \subseteq ET(P)$ ebenfalls. Falls der Fall 2 für $j = n - 1$

angewendet werden kann, folgt daraus $p'_n \mathcal{R} p_n$ und $\alpha_1 \alpha_2 \dots \alpha_n$ sind die Transitionsbeschriftungen, die vom Startzustand p_0 zum Zustand p_n in P führen. Das Wort w ist also in P ausführbar und somit in der Sprache $L(P)$ enthalten. □

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 in dessen as-Implementierungen zulässt. Mit dem Einsatz der cont-Funktion zum beliebigen fortsetzten der Traces kann dies ausgeglichen werden. Somit gilt wie die nächsten Proposition behauptet für die Fehler-Traces und die Fehler-geflutete Sprache Gleichheit und nicht nur die Inklusion, die aus den vorangegangenen Propositionen bereits folgt.

Proposition 2.7 (*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. „$\subseteq$“:

$$\begin{aligned} ET(P) &\overset{2.5}{=} \text{cont}(PrET(P)) \cup \text{cont}(MIT(P)) \\ &\overset{2.4}{\subseteq} \text{cont}\left(\bigcup_{P' \in \text{as-impl}(P)} PrET(P')\right) \cup \text{cont}\left(\bigcup_{P' \in \text{as-impl}(P)} MIT(P')\right) \\ &\overset{\text{cont monoton}}{=} \bigcup_{P' \in \text{as-impl}(P)} \text{cont}(PrET(P')) \cup \text{cont}(MIT(P')) \\ &\overset{2.5}{=} \bigcup_{P' \in \text{as-impl}(P)} ET(P'). \end{aligned}$$

1. „$\supseteq$“:
Da für P $ET(P) = \text{cont}(PrET(P)) \cup \text{cont}(MIT(P))$ gilt und für alle as-Implementierungen P' von P die analogen Gleichungen für $ET(P')$ gelten, 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 w entweder vollständig oder bis auf den letzten Buchstaben in P' ausführbar. Dies hängt davon ab, ob $w \in PrET(P')$ oder $w \in MIT(P')$ gilt. Für P muss jedoch nicht mal das Präfix von w ohne den letzten Buchstaben von w ausführbar sein. Da P' eine as-Implementierung von P ist, gibt es eine as-Verfeinerungs-Relation $\mathcal{R}$ zwischen den beiden Transitionssystemen. Es muss $p'_0 \mathcal{R} p_0$ gelten für die Startzustände von P' und P.

- Fall 1 ($w \in PrET(P')$): In P' existiert eine Verlängerung $v \in O^*$ von w, so dass $wv \in StET(P')$ gilt. Das Wort wv ist in P' ausführbar ($wv \in L(P')$). Es kann somit die Proposition 2.6 angewendet werden, wobei vorausgesetzt werden kann, dass der Trace in P' zu einem Zustand p'_n führt, der in $E_{P'}$ enthalten ist. Falls ein p_j mit $0 \leq j < n$ in E_P enthalten ist, dann ist ein Präfix von wv ein strikter Fehler-Trace in P und mit $w = \text{prune}(wv)$ ist somit ein Präfix von w in $ET(P)$ enthalten. Da ET unter Fortsetzung der Traces abgeschlossen ist, gilt auch $w \in ET(P)$. Falls für alle p_j mit $j < n$ $p_j \notin E_P$ gilt, kann $p'_n \mathcal{R} p_n$ gefolgert werden durch den Beweis von 2.6. Da $p'_n \in E_{P'}$ gilt, folgt draus mit 1.3.1, dass bereits p_n in E_P enthalten sein muss. Es gilt also $wv \in StET(P)$ und mit der Argumentation von oben folgt daraus $w \in ET(P)$.
- Fall 2 ($w \in MIT(P') \backslash PrET(P')$): Da w in $MIT(P')$ enthalten ist, gibt es ein $a \in I$ für das $w = va$ gilt mit $v \in \Sigma^*$. Analog zu Fall 1 kann das v in P' ausgeführt werden und davor die Proposition 2.6 angewendet werden, wobei das erreicht p'_n den Input a nicht als ausgehenden must-Transition besitzen soll. Falls es ein $p_j \in E_P$ mit $0 \leq j \leq n$ gibt, gilt $v \in ET(P)$ und wegen des Abschlusses unter cont auch $w \in ET(P)$. Ansonsten kann davon ausgegangen werden, dass $p'_n \mathcal{R} p_n$ für einen Zustand p_n aus P erfüllt ist. Falls a für p_n eine ausgehende must-Transition wäre, würde 1.3.2 auch für p'_n die Implementierung der a Transition fordern, dies würde einen Widerspruch zu $va \in MIT(P')$ darstellen. Es gilt also $p \overset{a}{\not\rightarrow}$ und $va \in MIT(P)$. Daraus ergibt sich direkt $w \in ET(P)$.

2. „$\subseteq$“:

$$
\begin{aligned}
EL(P) &\overset{2.5}{=} L(P) \cup ET(P) \\
&\overset{1.8}{\subseteq} \left(\bigcup_{P' \in \text{as-impl}(P)} L(P') \right) \cup ET(P) \\
&\overset{1.}{=} \left(\bigcup_{P' \in \text{as-impl}(P)} L(P') \right) \cup \left(\bigcup_{P' \in \text{as-impl}(P)} ET(P') \right) \\
&= \bigcup_{P' \in \text{as-impl}(P)} L(P') \cup ET(P') \\
&\overset{2.5}{=} \bigcup_{P' \in \text{as-impl}(P)} EL(P').
\end{aligned}
$$

2. „$\supseteq$“:
Da der erste Punkt dieser Proposition bereits beweisen ist, reicht es aus für diesen Punkt zu zeigen, dass $\bigcup_{P' \in \text{as-impl}(P)} EL(P') \backslash ET(P')$ eine Teilmenge von $EL(P)$ ist. Die Menge $EL(P') \backslash ET(P')$ entspricht $L(P') \backslash ET(P')$. Es musst 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 wie zuvor $w \in ET(P) \subseteq EL(P)$, da

ein Präfix von w in P zu einem Fehler-Zustand führen musst. Falls w ausführbar ist in P gilt $w \in L(P) \subseteq EL(P)$. □

Korollar 2.8 (*lokale Fehler Erreichbarkeit*).

(i) Es ist ein Fehler lokal erreichbar in einem MEIO $P \Leftrightarrow \exists$ as-Implementierung von P, in der ein Fehler lokal erreichbar ist.

(ii) Falls für zwei MEIOs $P_1 \sqsubseteq_E^{\mathrm{B}} P_2$ gilt und in P_1 ein Fehler lokal erreichbar ist, dann ist auch in P_2 ein Fehler lokal erreichbar.

Beweis.
(i) $\Rightarrow$:
Da ein Fehler in P lokal erreichbar ist, gilt $\varepsilon \in PrET_P \subseteq ET_P$. Es muss aufgrund von Proposition 2.7.1 mindestens ein $P' \in$ as-impl(P) geben, für dass $\varepsilon \in ET_{P'}$ gilt. Dies kann nur der Fall sein, wenn durch lokale Aktionen in P' ein Fehler-Zustand erreicht werden kann. Es ist also auch in der as-Implikation P' ein Fehler lokal erreichbar, da ET die Fortsetzungen von $PrET$ und MIT enthält und Elemente aus MIT mindestens die Länge 1 haben müssen.

(i) $\Leftarrow$:
Sei P' die as-Implementierung von P, in der ein Fehler lokal erreichbar ist. Es gilt dann $\varepsilon \in PrET(P') \subseteq ET_{P'}$. Mit Proposition 2.7.1 folgt daraus $\varepsilon \in ET_P$. Es muss also auch in P ein Fehler-Zustand lokal erreichbar sein.

(ii):
Da für P_1 ein Fehler lokale erreichbar ist folgt mit 1. dass es auch eine as-Implikation P_1' von P_1 gibt, für die ein Fehler lokal erreichbar ist. Mit $P_1 \sqsubseteq_E^{\mathrm{B}} P_2$ ergibt sich daraus, dass es auch eine as-Implementierung P_2' von P_2 geben muss, die lokal einen Fehler-Zustand erreichen kann. P_2 muss aufgrund von 1. ebenfalls einen Fehler lokal erreichen können, da es eine as-Implementierung gibt, die dies kann. □

Satz 2.9 (*Kommunikationsfehler-Semantik für Parallelkompositionen*). *Für zwei komponierbare MEIOs P_1, P_2 und ihre Komposition P_{12} gilt:*

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

Beweis.
1. „$\subseteq$“:
Da beide Seiten der Gleichung unter der Fortsetzung cont abgeschlossen sind, genügt es ein präfix-minimales Element w aus 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 $(p_{01}, p_{02}) \stackrel{x}{\Rightarrow}_{12} (p_1, p_2) \wedge a \in I_{12} \wedge (p_1, p_2) \stackrel{a}{\not\rightarrow}_{12}$. Da $I_{12} = (I_1 \cup I_2) \backslash (O_1 \cup O_2)$ 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 $a \in (I_1 \cup I_2) \backslash (I_1 \cap I_2)$ ist.
  - Fall 1a) ($a \in (I_1 \cap I_2)$): Durch Projektion des Ablaufes auf die einzelnen Transitionssysteme erhält man oBdA $p_{01} \stackrel{x_1}{\Rightarrow}_1 p_1 \stackrel{a}{\not\rightarrow}_1$ und $p_{02} \stackrel{x_2}{\Rightarrow}_2 p_2 \stackrel{a}{\not\rightarrow}_2$ oder $p_{02} \stackrel{x_2}{\Rightarrow}_2 p_2 \stackrel{a}{\longrightarrow}_2$ mit $x \in x_1 \| x_2$. Daraus kann $x_1 a \in MIT_1 \subseteq ET_1$ und $x_2 a \in EL_2$ ($x_2 a \in MIT_2$ oder $x_2 a \in L_2$) gefolgert werden. Damit folgt $w \in (x_1 \| x_2) \cdot \{a\} = (x_1 a) \| (x_2 a) \subseteq ET_1 \| EL_2$, und somit ist w in der rechten Seite der Gleichung enthalten.
  - Fall 1b) ($a \in (I_1 \cup I_2) \backslash (I_1 \cap I_2)$): OBdA gilt $a \in I_1$. Durch die Projektion auf die einzelnen Komponenten erhält man: $p_{01} \stackrel{x_1}{\Rightarrow}_1 p_1 \stackrel{a}{\not\rightarrow}_1$ und $p_{02} \stackrel{x_2}{\Rightarrow}_2 p_2$ mit $x \in x_1 \| x_2$. Daraus folgt $x_1 a \in MIT_1 \subseteq ET_1$ und $x_2 \in L_2 \subseteq EL_2$. Somit gilt $w \in (x_1 \| x_2) \cdot \{a\} \subseteq (x_1 a) \| x_2 \subseteq ET_1 \| EL_2$. 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 $v \in O_{12}^*$ existiert, so dass $(p_{01}, p_{02}) \stackrel{w}{\Rightarrow}_{12} (p_1, p_2) \stackrel{v}{\Rightarrow}_{12} (p_1', p_2')$ gilt mit $(p_1', p_2') \in E_{12}$ und $w = \text{prune}(wv)$. Durch Projektion auf die Komponenten erhält man $p_{01} \stackrel{w_1}{\Rightarrow}_1 p_1 \stackrel{v_1}{\Rightarrow}_1 p_1'$ und $p_{02} \stackrel{w_2}{\Rightarrow}_2 p_2 \stackrel{v_2}{\Rightarrow}_2 p_2'$ 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 fehlende Sicherstellen der Synchronisation (fehlende must-Input-Transition) in einer der Komponenten entstanden.
  - Fall 2a) (geerbter Fehler): OBdA gilt $p_1' \in E_1$. Daraus folgt, $w_1 v_1 \in StET_1 \subseteq \text{cont}(PrET_1) \subseteq ET_1$. Da $p_{02} \stackrel{w_2 v_2}{\Rightarrow}_2$ gilt, erhält man $w_2 v_2 \in L_2 \subseteq EL_2$. Dadurch ergibt sich $wv \in ET_1 \| EL_2$ mit $w = \text{prune}(wv)$ 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 $p_1' \stackrel{a}{\not\rightarrow}_1$ und $p_2' \stackrel{a}{\dashrightarrow}_2$. 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 $w = \text{prune}(wva)$ 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 $x \in \text{prune}((ET_1 \| EL_2) \cup (EL_1 \| ET_2))$ 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. Also v entweder auf einen Input I_{12} endet oder $v = \varepsilon$. Damit muss v ein Präfix von x sein, denn 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. Es gilt dann $w_2 \in ET_2$ 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 den beiden vorangegangen 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 $p_{02} \overset{v_2'}{\Longrightarrow}_2$ gilt. Falls $p_{02} \overset{v_2'}{\Longrightarrow}$ nicht gilt, ist $v_2' = v_2 \in MIT_2$ und v_2' endet auf $b = a \in I_2$.

- Fall I ($v_1 \in MIT_1$): Da $v_1 \in MIT_1$ gilt, muss v_1 ungleich ε sein. Es gibt einen Ablauf der Form $p_{01} \overset{v_1'}{\Longrightarrow}_1 p_1 \overset{a}{\not\longrightarrow}_1$ und es gilt $v' = v''a$.

  - Fall Ia) ($a \notin \Sigma_2$): Es gilt $p_{02} \overset{v_2'}{\Longrightarrow}_2 p_2$ mit $v'' \in v_1' \| v_2'$. Dadurch erhält man $(p_{01}, p_{02}) \overset{v''}{\Longrightarrow}_{12} (p_1, p_2) \overset{a}{\not\longrightarrow}_{12}$ 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 $p_{02} \overset{v_2''}{\Longrightarrow}_2 p_2 \overset{a}{\not\longrightarrow}_2$ und $v'' \in v_1' \| v_2''$. a ist für P_2, ebenso wie für P_1, ein nicht sichergestellter Input. Daraus folgt, dass $(p_1, p_2) \overset{a}{\not\longrightarrow}_{12}$ gilt. Es wird ebenfalls $v := v''a = v' \in MIT_{12}$ gewählt.

  - Fall Ic) ($a \in I_2$ und $v_2' \notin MIT_2$): Es gilt $p_{02} \overset{v_2''}{\Longrightarrow}_2 p_2 \overset{a}{\longrightarrow}_2$ mit $v_2' = v_2''a \in L_2$. Da die gemeinsamen Inputs synchronisiert werden, folgt $(p_1, p_2) \overset{a}{\not\longrightarrow}_{12}$ bereits aus $q_1 \overset{a}{\not\longrightarrow}_1$. 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 $p_{02} \overset{v_2'}{\Longrightarrow}_2$. Man erhält also $p_{02} \overset{v_2''}{\Longrightarrow}_2 p_2 \overset{a}{\dashrightarrow}_2$ mit $v'' \in v_1' \| v_2''$. Daraus ergibt sich $(p_{01}, p_{02}) \overset{v''}{\Longrightarrow}_{12} (p_1, p_2)$ mit $p_2 \overset{a}{\dashrightarrow}_1$,

$p_1 \overset{a}{\not\rightarrow}_1, a \in I_1$ 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 ($v_1 \in PrET_1$): $\exists u_1 \in O_1^* : p_{01} \overset{v_1}{\Longrightarrow}_1 p_1 \overset{u_1}{\Longrightarrow}_1 p_1'$ 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 $p_{02} \overset{v_2'}{\Longrightarrow}_2 p_2$ mit $(p_{01}, p_{02}) \overset{v'}{\Longrightarrow}_{12} (p_1, p_2)$.
  - Fall IIa) $\Big(u_2 \in (O_1 \cap I_2)^*, c \in (O_1 \cap I_2)$, sodass $u_2 c$ Präfix von $u_1|_{I_2}$ mit $p_2 \overset{u_2}{\Longrightarrow}_2 p_2' \overset{c}{\not\rightarrow}_2\Big)$: Für das Präfix $u_1' c$ von u_1 mit $(u_1' c)|_{I_2} = u_2 c$ weiß man, dass $p_1 \overset{u_1'}{\Longrightarrow}_1 p_1'' \overset{c}{\dashrightarrow}_1$. Somit gilt $u_1' \in u_1' \| u_2$ und $(p_1, p_2) \overset{u_1'}{\Longrightarrow}_{12} (p_1'', p_2') \in E_{12}$, da für P_2 der entsprechende Input nicht sichergestellt 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) $\Big(p_2 \overset{u_2}{\Longrightarrow}_2 p_2'$ mit $u_2 = u_1|_{I_2}\Big)$: Es gilt $u_1 \in u_1 \| u_2$ und $(p_1, p_2) \overset{u_1}{\Longrightarrow}_{12} (p_1', p_2') \in E_{12}$, da $p_1' \in E_1$ und somit handelt es sich in P_{12} um einen geerbten Fehler. Nun wird $v := \text{prune}(v' u_1) \in PrET_{12}$ gewählt, das wiederum ein Präfix von v' ist.

2.:
Durch die Definitionen ist klar, dass $L_j \subseteq EL_j$ und $ET_j \subseteq EL_j$ gilt. Die Argumentation startet auf den rechten Seite der Gleichung:

$$
\begin{aligned}
(EL_1 \| EL_2) \cup ET_{12} &\overset{2.5}{=} ((L_1 \cup ET_1) \,\|\, (L_2 \cup ET_2)) \cup ET_{12} \\
&= (L_1 \| L_2) \cup \underbrace{(L_1 \| ET_2)}_{\substack{\subseteq (EL_1 \| ET_2) \\ \overset{1.}{\subseteq} ET_{12}}} \cup \underbrace{(ET_1 \| L_2)}_{\substack{\subseteq (ET_1 \| EL_2) \\ \overset{1.}{\subseteq} ET_{12}}} \cup \underbrace{(ET_1 \| ET_2)}_{\substack{\subseteq (EL_1 \| ET_2) \\ \overset{1.}{\subseteq} ET_{12}}} \cup ET_{12} \\
&= (L_1 \| L_2) \cup ET_{12} \\
&\overset{1.9}{=} L_{12} \cup ET_{12} \\
&\overset{2.5}{=} EL_{12}.
\end{aligned}
$$

□

Korollar 2.10 (*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:

- $ET_{31} \overset{2.9.1}{=} \mathrm{cont}\,(\mathrm{prune}\,((ET_3 \| EL_1) \cup (EL_3 \| ET_1)))$
  $\overset{\substack{ET_1 \subseteq ET_2 \\ \text{und} \\ EL_1 \subseteq EL_2}}{\subseteq} \mathrm{cont}\,(\mathrm{prune}\,((ET_3 \| EL_2) \cup (EL_3 \| ET_2)))$
  $\overset{2.9.1}{=} ET_{32},$
- $EL_{31} \overset{2.9.2}{=} (EL_3 \| EL_1) \cup E_{31}$
  $\overset{\substack{EL_1 \subseteq EL_2 \\ \text{und} \\ ET_{31} \subseteq ET_{32}}}{\subseteq} (EL_3 \| EL_2) \cup ET_{32}$
  $\overset{2.9.2}{=} EL_{32}.$

□

Lemma 2.11 (*Verfeinerung mit Kommunikationsfehlern*). *Gegeben sind zwei MEIOs P_1 und P_2 mit der gleichen Signatur. Wenn $U \| P_1 \sqsubseteq_E^{\mathrm{B}} U \| P_2$ 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 $U \| P_1$. Wegen 2.8 (ii) erreicht auch $U \| P_2$ lokal einen Fehler-Zustand. 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 2.1):
  - $U = \{p_0, p_1, \dots, p_{n+1}\}$,
  - $p_{0U} = p_0$,
  - $\dashrightarrow_U = \longrightarrow_U = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j \leq n\}$
    $\cup \{(p_j, x, p_{n+1}) \mid x \in I_U \backslash \{x_{j+1}\}, 0 \leq j \leq n\}$
    $\cup \{(p_{n+1}, x, p_{n+1}) \mid x \in I_U\}$,

– $E_U = \emptyset$.

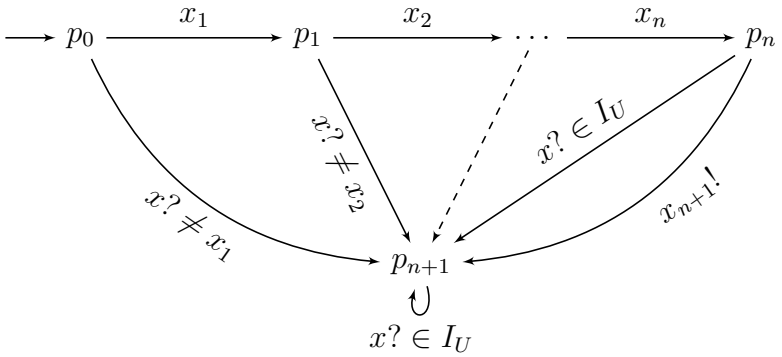

Abbildung 2.1: $x? \neq x_j$ steht für alle $x \in I_U \backslash \{x_j\}$

Für w können nun zwei Fälle unterschieden werden. Aus beiden wird folgen, dass für die Parallelkomposition $U \| P_1$ $\varepsilon \in PrET(U \| P_1)$ gilt.

– Fall 2a) ($w \in MIT_1$): In $U \| P_1$ erhält man $(p_0, p_{01}) \stackrel{x_1 \ldots x_n}{=\!=\!\Rightarrow}_{U \| P_1} (p_n, p')$ mit $p' \stackrel{x_{n+1}}{\not\longrightarrow}_1$ und $p_n \stackrel{x_{n+1}}{\longrightarrow}_U$. Deshalb gilt $(p_n, p') \in E_{U \| P_1}$. Da alle Aktionen aus w bis auf x_{n+1} synchronisiert werden und $I \cap I_U = \emptyset$, gilt $x_1, \ldots, x_n \in O_{U \| P_1}$. Da $(p_n, p') \in E_{U \| P_1}$ in $U \| P_1$ lokal erreichbar ist gilt $\varepsilon \in PrET(U \| P_1)$.

– Fall 2b) ($w \in PrET_1$): In der Parallelkomposition von U und P_1 erhält man $(p_0, p_{01}) \stackrel{w}{=\!=\!\Rightarrow}_{U \| P_1} (p_{n+1}, p'') \stackrel{u}{=\!=\!\Rightarrow}_{U \| P_1} (p_{n+1}, p')$ für $u \in O^*$ und $p' \in E_1$. Daraus folgt $(p_{n+1}, p') \in E_{U \| P_1}$ und somit $wu \in StET(U \| P_1)$. Da alle Aktionen in w synchronisiert werden und $I \cap I_U = \emptyset$, gilt $x_1, \ldots, x_n, x_{n+1} \in O_{U \| P_1}$ und, da $u \in O^*$, folgt $u \in O^*_{U \| P_1}$. Somit ergibt sich $\varepsilon \in PrET(U \| P_1)$.

Da $\varepsilon \in PrET(U \| P_1)$ gilt, kann durch $U \| P_1 \sqsubseteq^{\mathrm{B}}_E U \| P_2$ unter zuhilfenahme von 2.8 (ii) geschlossen werden, dass auch in $U \| P_2$ ein Fehler-Zustand lokal erreichbar sein muss.
Der in $U \| P_2$ erreichbare 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 sicherstellt, muss ein lokal erreichbarer Fehler-Zustand der Form sein, dass ein Output $a \in O_U$ von U möglich ist, dessen Synchronisation in P_2 mit einem passenden Input nicht sichergestellt ist (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 $i \leq n, p' \stackrel{x_{i+1}}{\not\longrightarrow}_2$ und $x_{i+1} \in O_U = I$. Durch Projektion erhält man dann $p_{02} \stackrel{x_1 \ldots x_i}{=\!=\!\Rightarrow}_2 p' \stackrel{x_{i+1}}{\not\longrightarrow}_2$ und damit gilt $x_1 \ldots x_{i+1} \in MIT_2 \subseteq ET_2$. Somit ist ein Präfix von w in ET_2 enthalten.

– Fall 2ii) (geerbter Fehler): U hat $x_1 \ldots x_i u$ mit $u \in I^*_U = O^*$ ausgeführt und ebenso hat P_2 dieses Wort abgearbeitet. Durch dies hat P_2 einen Zustand

in E_2 erreicht, da von U keine Fehler geerbt werden können. Es gilt dann $\text{prune}(x_1 \dots x_i u) = \text{prune}(x_1 \dots x_i) \in PrET_2 \subseteq ET_2$. Da $x_1 \dots x_i$ ein Präfix von w ist, führt in diesem Fall eine Verlängerung um lokale Aktionen eines 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 auch 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 $L_1 \backslash ET_1 \subseteq EL_2$ gilt.
Es wird dafür ein beliebiges $w \in L_1 \backslash ET_1$ 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 2.2):
  - $U = \{p_0, p_1, \dots, p_n, p\}$,
  - $p_{0U} = p_0$,
  - $\dashrightarrow_U = \longrightarrow_U = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j < n\}$
    $\cup \{(p_j, x, p) \mid x \in I_U \backslash \{x_{j+1}\}, 0 \leq j < n\}$
    $\cup \{(p, x, p) \mid x \in I_U\}$,
  - $E_U = \{p_n\}$.

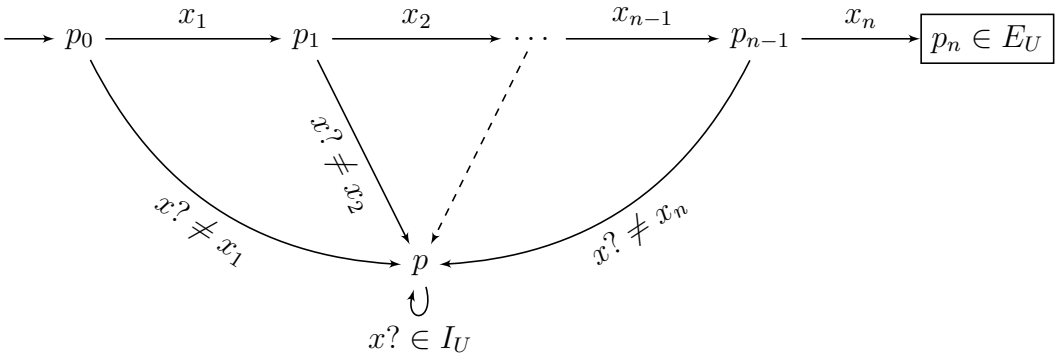

Abbildung 2.2: $x? \neq x_j$ steht für alle $x \in I_U \backslash \{x_j\}$, p_n ist der einzige Fehler-Zustand

Da $p_{01} \stackrel{w}{\Longrightarrow}_1 p'$ gilt, kann man schließen, dass $U \| P_1$ einen lokal erreichbaren geerbten Fehler hat. Aufgrund von Korollar 2.8 (ii) muss ebenfalls ein Fehler-Zustand in $U \| P_2$ lokal erreichbar sein.

- Fall 2a) (neuer Fehler aufgrund von $x_i \in O_U$ und $p_{02} \stackrel{x_1 \dots x_{i-1}}{\Longrightarrow}_2 p'' \stackrel{x_i}{\not\longrightarrow}_2$): 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 könnten.

- Fall 2b) (neuer Fehler aufgrund von $a \in O = I_U$): Der einzige Zustand, in dem U nicht alle Inputs erlaubt sind, ist p_n, der bereits ein Fehler-Zustand ist. Da in diesem Fall der Fehler-Zustand in $U \| P_2$ 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 $p_{02} \stackrel{x_1 \dots x_n}{=\!=\!\Rightarrow}_2$ gilt. In diesem Fall ist w ausführbar und es gilt $w \in L_2 \subseteq EL_2$.

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

□

Der folgende Satz sagt aus, dass $\sqsubseteq_E$ die gröbste Präkongruenz ist, die charakterisiert werden soll, also gleich der vollständig abstrakten Präkongruenz $\sqsubseteq_E^{\mathrm{C}}$.

Satz 2.12 (*Vollständige Abstraktheit für Kommunikationsfehler-Semantik*). *Für zwei MEIOs P_1 und P_2 mit derselben Signatur gilt $P_1 \sqsubseteq_E^{\mathrm{C}} P_2 \Leftrightarrow P_1 \sqsubseteq_E P_2$.*

Beweis.
„⇐“: 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. Diese as-Implementierung muss es geben, wenn P_1 und P_2 lokal erreichbare Fehler enthalten, wegen Korollar 2.8 (i). Daraus folgt, dass $P_1 \sqsubseteq_E^{\mathrm{B}} P_2$ gilt, da $\sqsubseteq_E^{\mathrm{B}}$ in Definition 2.2 über die lokale Erreichbarkeit der Fehler-Zustande in den as-Implementierungen definiert wurde. Es ist also $\sqsubseteq_E$ in $\sqsubseteq_E^{\mathrm{B}}$ enthalten. Wie in Korollar 2.10 gezeigt, ist $\sqsubseteq_E$ eine Präkongruenz bezüglich $\cdot \| \cdot$. Da $\sqsubseteq_E^{\mathrm{C}}$ die gröbste Präkongruenz bezüglich $\cdot \| \cdot$ ist, die in $\sqsubseteq_E^{\mathrm{B}}$ enthalten ist, muss $\sqsubseteq_E$ in $\sqsubseteq_E^{\mathrm{C}}$ enthalten sein. Es folgt also aus $P_1 \sqsubseteq_E P_2$, dass auch $P_1 \sqsubseteq_E^{\mathrm{C}} P_2$ gilt.

„⇒“: Durch die Definition von $\sqsubseteq_E^{\mathrm{C}}$ als Präkongruenz in 2.2 folgt aus $P_1 \sqsubseteq_E^{\mathrm{C}} P_2$, dass $U \| P_1 \sqsubseteq_E^{\mathrm{C}} U \| P_2$ für alle MEIOs U gilt, die mit P_1 komponierbar sind. Da $\sqsubseteq_E^{\mathrm{C}}$ nach Definition auch in $\sqsubseteq_E^{\mathrm{B}}$ enthalten sein soll, folgt aus $U \| P_1 \sqsubseteq_E^{\mathrm{C}} U \| P_2$ auch die Gültigkeit von $U \| P_1 \sqsubseteq_E^{\mathrm{B}} U \| P_2$ für alle diese MEIOs U. Mit Lemma 2.11 folgt dann $P_1 \sqsubseteq_E P_2$. □

Es wurde somit eine Kette an Folgerungen gezeigt, die sich zu einem Ring schließt. Dies ist in Abbildung 2.3 dargestellt.

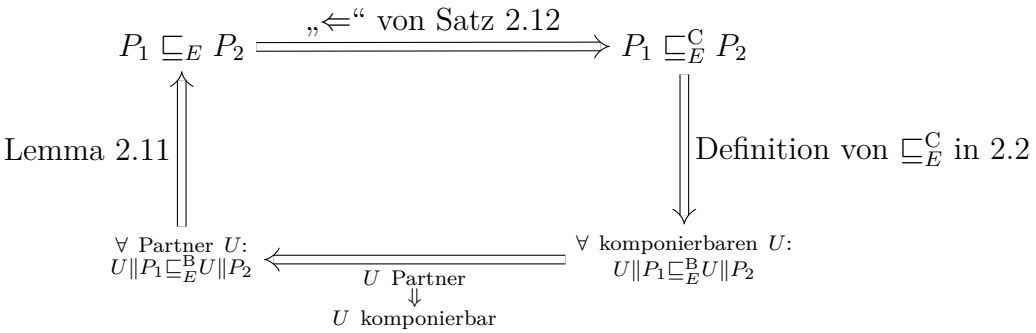

Abbildung 2.3: Folgerungskette der Fehler-Relationen

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 2.13. *Es gilt:* $P_1 \sqsubseteq_E P_2 \Leftrightarrow U \| P_1 \sqsubseteq_E^{\mathrm{B}} U \| P_2$ *für alle Partner* U*.*

2.2 Testing-Ansatz

Der gröbste Präkongruenz-Ansatz aus dem letzten Teil stützt sich auf die Parallelkomposition von MEIOs, die wie bereits in Kapitel 1.2 erwähnt auch anders gestaltet hätte werden können. Spezielle bei neuen Fehler ist dort gezeigt worden, dass es zu einem ungewöhnlichen Verfeinerungs-Verhalten kommen kann, das möglicherweise gar nicht so gewollt ist. Deshalb ist es sinnvoll sich nur auf die Definition der Parallelkomposition von EIO zu stützen, die hier für die Implementierungen gilt. Dies führt zu dem in diesem Teil behandelten Testing-Ansatz. Dieser lässt jedoch keine Aussage zu gröbste Präkongruenz zu, wie das im letzen Teil möglich war. Der Test ersetzt die Basisrelation als grundlegende Definition für den Ansatz. Die Präkongruenz, die sich ergibt, ist jedoch die selbe.
Der Testing-Ansatz stützt sich auf das Vorgehen, das in [BV15b] angewendet wurde. Jedoch sind Tests dort Tupel aus einer Implementierung und einer Menge an Aktionen, über denen synchronisiert werden soll. Die MEIOs, die hier komponiert werden bringen die Menge Synch an Aktionen, die synchronisiert werden bereits mit. Es wird im Gegensatz zu [BV15b] mit Inputs und Outputs gearbeitet und dadurch scheint der Ansatz die gemeinsamen Aktionen zu synchronisieren natürlicher, wie eine Menge an Aktionen beim Test vorzugeben.
Die Definition von lokaler Erreichbarkeit eines Fehler-Zustandes soll aus dem Erweiterungs-Ansatz übernommen werden. Es wird also trotzdem noch optimistisch davon

ausgegangen, dass Fehler erst zu Problemen führen, wenn eine hilfreiche Umgebung dies nicht mehr verhindern kann.

Definition 2.14 (*Test und Verfeinerung für Kommunikationsfehler*). *Ein* Test T *ist eine Implementierung. Ein MEIO* P as-erfüllt *einen Kommunikationsfehler-Test* T*, falls* $S\|T$ *fehler-frei ist für alle* $S \in \text{as-impl}(P)$*. Es wird dann* $P\, sat^E_{\text{as}}\, T$ *geschrieben. Die Parallelkomposition* $S\|T$ *ist* fehler-frei*, wenn kein Fehler lokal erreichbar ist. Ein MEIO* P Fehler-verfeinert P'*, falls für alle Tests* T*:* $P'\, sat^E_{\text{as}}\, T \Rightarrow P\, sat^E_{\text{as}}\, T$*.*

Abgesehen von Korollar 2.8 (ii) erweisen sich Definition 2.3 bis Korollar 2.10 auch hier als nützlich.
Die Basisrelation aus dem gröbsten Präkongruenz-Ansatz gibt es in diesem Teil nicht, somit kann das Lemma 2.11 nicht in dieser Art formuliert werden. Jedoch kann hier mit Tests gearbeitet werden und unter deren zuhilfenahme ein ähnliches Lemma formuliert werden.

Lemma 2.15 (*Testing-Verfeinerung mit Kommunikationsfehlern*). *Gegeben sind zwei MEIOs* P_1 *und* P_2 *mit der gleichen Signatur. Wenn für alle Tests* T*, die Partner von* P_1 *bzw.* P_2 *sind,* $P_2\, sat^E_{\text{as}}\, T \Rightarrow P_1\, sat^E_{\text{as}}\, T$ *gilt, dann folgt daraus die Gültigkeit von* $P_1 \sqsubseteq_E P_2$*.*

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 unter cont abgeschlossen sind.
Mit Proposition 2.7 folgt aus $w \in ET_1$, 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 $P_1\, sat^E_{\text{as}}\, T$ nicht. Es darf also auch P_2 den Test T nicht as-erfüllen, wegen der Implikation $P_2\, sat^E_{\text{as}}\, T \Rightarrow P_1\, sat^E_{\text{as}}\, T$. Damit P_2 T nicht as-erfüllt muss es eine as-Implementierungen P_2' geben, die in Parallelkomposition mit T zu einem nicht fehler-freien System führt. Es muss also 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 interen 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 2.7 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 Benennungen der Mengen dem Transitionssystem U aus Abbildung 2.1):

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

  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 $(p_{01}', p_0) \stackrel{x_1 \dots x_n}{\Longrightarrow}_{P_1' \| T} (p', p_n)$ mit $p' \stackrel{x_{n+1}}{\not\longrightarrow}_{P_1'}$ und $p_n \stackrel{x_{n+1}}{\longrightarrow}_T$. Deshalb gilt $(p', p_n) \in E_{P_1' \| T}$. Da alle Aktionen aus w bis auf x_{n+1} synchronisiert werden und $I \cap I_T = \emptyset$, gilt $x_1, \dots, x_n \in O_{P_1' \| T}$. Es folgt also $\varepsilon \in PrET(P_1' \| T)$.

  - Fall 2b) ($w \in PrET_{P_1'}$): In der Parallelkomposition $P_1' \| T$ erhält man die Transitionsfolge $(p_{01}, p_0) \stackrel{w}{\Longrightarrow}_{P_1' \| T} (p'', p_{n+1}) \stackrel{u}{\Longrightarrow}_{P_1' \| T} (p', p_{n+1})$ für $u \in O^*$ und $p' \in E_{P_1'}$. Daraus folgt $(p', p_{n+1}) \in E_{P_1' \| T}$ und somit $wu \in StET(P_1' \| T)$. Da alle Aktionen in w synchronisiert werden und $I \cap I_T = \emptyset$, gilt $x_1, \dots, x_n, x_{n+1} \in O_{P_1' \| T}$ und, da $u \in O^*$, folgt $u \in O^*_{P_1' \| T}$. Somit ergibt sich $\varepsilon \in PrET(P_1' \| T)$.

  Da $\varepsilon \in PrET(P_1' \| T)$ gilt, kann durch $P_2 \, sat^E_{\text{as}} \, T \Rightarrow P_1 \, sat^E_{\text{as}} \, T$ geschlossen werden, dass auch $P_2 \, sat^E_{\text{as}} \, T$ nicht gelten kann. Die Relation gilt für P_2 nicht, da es eine as-Implementierung P_2' von P_2 gibt, so dass $P_2' \| T$ nicht fehler-frei ist.
  Der lokal erreichbar Fehler in $P_2' \| T$ kann geerbt oder neu sein.

  - Fall 2i) (neuer Fehler): Da jeder Zustand von T alle Inputs $x \in O = I_T$ zulässt, muss ein lokal erreichbarer Fehler-Zustand der Form sein, dass ein Outputs $a \in O_T$ von T möglich ist, der nicht mit einem passenden Input aus P_2' synchronisiert werden werden kann. Durch die Konstruktion von T sind in p_{n+1} keine Outputs möglich. Ein neuer Fehler muss also die Form (p', p_i) haben mit $i \leq n$, $p' \stackrel{x_{i+1}}{\not\longrightarrow}_{P_2'}$ und $x_{i+1} \in O_T = I$. Durch Projektion erhält man dann $p_{02} \stackrel{x_1 \dots x_i}{\Longrightarrow}_{P_2'} p' \stackrel{x_{i+1}}{\not\longrightarrow}_{P_2'}$ und damit gilt $x_1 \dots x_{i+1} \in MIT_{P_2'} \subseteq ET_{P_2'}$. Es ist also ein Präfix von w in $ET_{P_2'}$ enthalten und mit Proposition 2.7 auch in ET_2.

– Fall 2ii) (geerbter Fehler): T hat $x_1 \dots x_i u$ mit $u \in I_T^* = O^*$ ausgeführt und ebenso hat P_2' dieses Wort abgearbeitet. Durch dies hat P_2' einen Zustand in $E_{P_2'}$ erreicht, da von T keine Fehler geerbt werden können. Es gilt dann $\text{prune}(x_1 \dots x_i u) = \text{prune}(x_1 \dots x_i) \in PrET_{P_2'} \subseteq ET_{P_2'}$. Da $x_1 \dots x_i$ ein Präfix von w ist, führt in diesem Fall eine Verlängerung um lokale Aktionen eines 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 $ET_{P_2'}$ enthalten und somit ist mit Proposition 2.7 ein Präfix von w in ET_2 enthalten.

Um die andere Inklusion zu beweisen, reich es aufgrund der ersten Inklusion und der Definition von EL aus zu zeigen, dass $L_1 \backslash ET_1 \subseteq EL_2$ gilt.
Es wird dafür ein beliebiges $w \in L_1 \backslash ET_1$ gewählt und gezeigt, dass es in EL_2 enthalten ist. Das w ist wegen der Propositionen 1.8 und 2.7 auch für eine as-Implementierung P_1' von P_1 in $L_{P_1'} \backslash ET_{P_1'}$ enthalten.

- 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 T wie folgt konstruiert (T entspricht dabei U aus Abbildung 2.2 bis auf die Benennung der Mengen):

  – $T = \{p_0, p_1, \dots, p_n, p\}$,

  – $p_{0T} = p_0$,

  – $\dashrightarrow_T = \longrightarrow_T = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j < n\}$
  $\cup \{(p_j, x, p) \mid x \in I_T \backslash \{x_{j+1}\}, 0 \leq j < n\}$
  $\cup \{(p, x, p) \mid x \in I_T\}$,

  – $E_T = \{p_n\}$.

  Da $p_{01} \overset{w}{\Rightarrow}_{P_1'} p'$ gilt, kann man schließen, dass $P_1' \| T$ ein lokal erreichbaren geerbten Fehler hat. Es muss also auch eine as-Implementierung P_2' von P_2 geben, für die $P_2' \| T$ einen lokal erreichbaren Fehler-Zustand hat.

  – Fall 2a) (neuer Fehler aufgrund von $x_i \in O_T$ und $p_{02} \overset{x_1 \dots x_{i-1}}{\Rightarrow}_{P_2'} p'' \not\xrightarrow{x_i}_{P_2'}$): Es gilt $x_1 \dots x_i \in MIT_{P_2'}$ und somit $w \in EL_{P_2'}$. Anzumerken ist, dass es nur auf diesem Weg Outputs von T möglich sind, deshalb gibt es keine anderen Outputs von T, die zu einem neuen Fehler führen können. Es gilt $w \in EL_2$ wegen Proposition 2.7.

  – Fall 2b) (neuer Fehler aufgrund von $a \in O = I_T$): Der einzige Zustand, in dem T nicht alle Inputs erlaubt sind, ist p_n, der bereits ein Fehler-Zustand ist. Da in diesem Fall der Fehler-Zustand in $P_2' \| T$ erreichbar ist, besitzt der komponierte MEIO einen geerbten Fehler und es gilt $w \in L_{P_2'} \subseteq EL_2$, wegen dem folgenden Fall 2c).

– Fall 2c) (geerbter Fehler von T): Da p_n der einzige Fehler-Zustand in T ist und alle Aktionen synchronisiert sind, ist dies nur möglich, wenn $p_{02} \stackrel{x_1 \dots x_n}{\Longrightarrow}_{P_2'}$ gilt. In diesem Fall ist $w \in L_{P_2'} \subseteq EL_{P_2'}$. Daraus folgt mit Proposition 2.7 $w \in EL_2$.

– Fall 2d) (geerbter Fehler von P_2'): Es gilt $p_{02} \stackrel{x_1 \dots x_n}{\Longrightarrow}_{P_2'} p' \in E_{P_2'}$ für ein $i \geq 0$ und $u \in O^*$. Somit ist $x_1 \dots x_i u \in StET_{P_2'}$ und damit $\text{prune}(x_1 \dots x_i u) = \text{prune}(x_1 \dots x_i) \in PrET_{P_2'} \subseteq EL_{P_2'}$. Mit Hilfe von Proposition 2.7 folgt $w \in EL_{P_2'} \subseteq EL_2$.

□

Satz 2.16. *Falls $P_1 \sqsubseteq_E P_2$ gilt folgt draus, dass P_1 P_2 Fehler-verfeinert.*

Beweis. 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. Aufgrund von Proposition 2.8 (i) gibt es dann auch entsprechende as-Implementierungen P_1' und P_2' für P_1 bzw. P_2, die ebenfalls Fehler-Zustände lokal erreichen können, wenn dies in P_1 bzw. P_2 möglich ist. Für alle Tests T gilt, dass $P_j' \| T$ für $j \in \{1, 2\}$ einen lokal erreichbaren Fehler hat, wenn P_j' einen solchen hat. Dies Schlussfolgerung ist möglich, da Satz 2.9.1 aussagt, dass die Parallelkomposition von einem Fehler-Trace aus $ET_{P_j'}$ mit einem Wort aus EL_T in den Fehler-Traces von $ET_{P_j' \| T}$ enthalten ist. ε ist ein Element von $ET_{P_j'}$, da ein Fehler lokal erreichbar ist und $\varepsilon \in EL_T$ gilt für alle Test T und somit folgt $\varepsilon \in ET_{P_j' \| T}$. Dies impliziert die lokal Fehler Erreichbarkeit in $P_j' \| T$. Falls also P_1 einen lokal erreichbaren Fehler-Zustand enthält, dann zeigt sich dies Verhalten auch in einer as-Implementierungen und ebenfalls in der Parallelkomposition der as-Implementierung mit allen Tests T. Aus einem lokal erreichbaren Fehler in P_1 folgt auch ein lokal erreichbarer Fehler-Zustand in P_2 und somit auch ein lokal erreichbarer Fehler-Zustand in der Parallelkomposition einer as-Implementierung von P_2 mit allen Tests T. Aus $P_1 \sqsubseteq_E P_2$ folgt also die Implikation $\neg P_1 \, sat^E_{\text{as}} \, T \Rightarrow \neg P_2 \, sat^E_{\text{as}} \, T$ für alle Test T. Wenn man die Negationen entfernt, ergibt sich für alle Tests T: $P_2 \, sat^E_{\text{as}} \, T \Rightarrow P_1 \, sat^E_{\text{as}} \, T$. Dies entspricht der Definition von P_1 Fehler-verfeinert P_2. □

Auch die in diesem Abschnitt gezeigten Folgerungen schließen sich zu einem Ring. Dies ist in Abbildung 2.4 dargestellt.

2.3 Zusammenhänge

Satz 2.17 (*Zusammenhang der Verfeinerungs-Relationen mit der Fehler-Relation*). *Für MEIOs P und Q gilt $P \sqsubseteq_{\text{as}} Q \Rightarrow P \sqsubseteq_{\text{w-as}} Q \Rightarrow P \sqsubseteq_E Q$. Die Implikationen in die andere Richtung gelten jedoch nicht.*

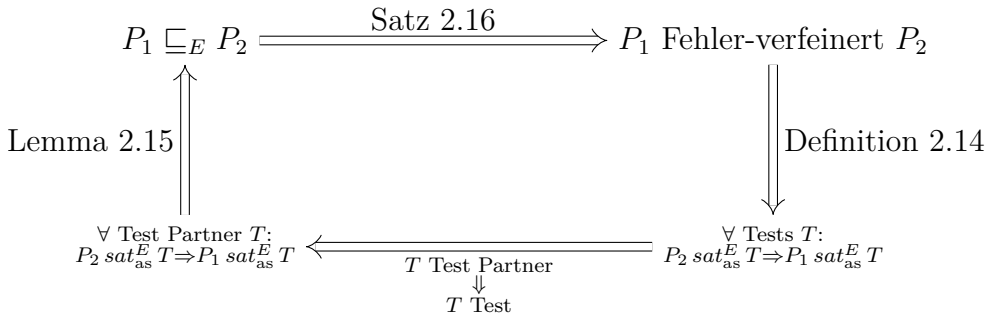

Abbildung 2.4: Folgerungskette der Testing-Verfeinerung und Fehler-Relation

Beweis.
$P \sqsubseteq_{\text{as}} Q \Rightarrow P \sqsubseteq_{\text{w-as}} Q$:
Um diese Implikation zu zeigen, muss man nachweisen, dass jede starke as-Verfeinerungs-Relation auch die Definition 1.4 der schwachen as-Verfeinerungs-Relation erfüllt. In beiden Simulations-Definitionen (1.3 und 1.4) müssen die Punkte für alle $(p, q) \in \mathcal{R}$ mit $q \notin E_Q$ gelten. Sei $\mathcal{R}$ nun eine as-Verfeinerungs-Relation. Es gilt also mit 1.3.1, dass p kein Fehler-Zustand von P ist. Somit ist auch 1. von 1.4 erfüllt. Für alle $\alpha \in \Sigma_\tau$ impliziert 1.3.2 $q \xrightarrow{\alpha}_Q q'$ $p \xrightarrow{\alpha}_P p'$ für ein p' mit $p' \mathcal{R} q'$. Da $\Sigma_\tau = I \cup O \cup \{\tau\}$ gilt, sind dadurch 2. und 3. der Definition 1.4 erfüllt. Die schwache ε-Transition aus 2. führt keine echten Transitionen aus, sondern bleibt beim Zustand p'. Die schwache $\hat{\omega}$-Transition aus 3. entspricht in $\mathcal{R}$ nur einer einzigen Transition für ω. Die Punkte 4. und 5. aus Definition 1.4 werden durch 1.3.3 erfüllt. Es gilt für $\mathcal{R}$ $p \overset{\alpha}{\dashrightarrow}_P p'$ impliziert $q \overset{\alpha}{\dashrightarrow}_Q q'$ für ein q' mit $p' \mathcal{R} q'$. Die in 1.4 geforderten schwachen may-Transitionen werden hier jeweils stark durch eine einzige Transition umgesetzt. $\mathcal{R}$ ist also auch eine schwache as-Verfeinerungs-Relation.

$P \sqsubseteq_{\text{w-as}} Q \Rightarrow P \sqsubseteq_E Q$:
Um diese Implikation zu beweisen wird gezeigt, dass eine beliebige schwache as-Verfeinerungs-Relation $\mathcal{R}$ zwischen P und Q auch die Eigenschaften der Relation $\sqsubseteq_E$ erfüllt. Da $\mathcal{R}$ eine schwache as-Verfeinerungs-Relation zwischen P und Q ist, muss $p_0 \mathcal{R} q_0$ gelten. Es sind die folgenden Punkte nachzuweisen:

- $ET_P \subseteq ET_Q$,
- $EL_P \subseteq EL_Q$.

Für den ersten Punkt wird ein beliebiges w aus ET_P betrachtet und gezeigt, dass dieses auch in ET_Q enthalten ist. Es kann davon ausgegangen werden, dass w präfix-minimal ist, da beide ET-Mengen unter cont abgeschlossen sind. w kann ein Element aus $PrET(P)$ sein oder ein Element aus $MIT(P)$.

- Fall 1 ($w \in PrET(P)$): Es existiert ein $v \in O_P$, sodass das Wort wv in P einen Fehler-Zustand erreicht. Für die entsprechende Transitionsfolge existieren Zustän-

de $p_1, p_2, \ldots, p_n$ in P und Aktionen $\alpha_1, \alpha_2, \ldots, \alpha_n$, die zusammengesetzt ein Wort $w' = \alpha_1\alpha_2 \ldots \alpha_n$ bilden, dass ohne die internen Aktionen wv entspricht. Die entsprechende Transitionsfolge in P ist dann $p_0 \xrightarrow{\alpha_1}_P p_1 \xrightarrow{\alpha_2}_P \ldots p_{n-1} \xrightarrow{\alpha_n}_P p_n \in E_P$. Abhängig davon, ob α_j ein Input oder eine lokale Aktion ist, kann mit 1.4.4 bzw. 1.4.5 argumentiert werden, dass das jeweilige α_j auch in Q schwach ausführbar ist, solange der entsprechende Zustand q_{j-1} kein Fehler-Zustand ist für $j \in \{1, 2, \ldots n\}$. Falls ein q_j für $j < n$ in E_Q enthalten ist, wurde bis dort ein Präfix von wv ausgeführt. Dieses Präfix ist in $StET_Q$ enthalten. Es gilt also mit $w = \text{prune}(wv)$ und dem Abschluss von ET unter cont $w \in ET_Q$. Ansonsten gibt es in Q einen Trace $q_0 \stackrel{\widehat{\alpha_1}}{\Longrightarrow}_Q q_1 \stackrel{\widehat{\alpha_2}}{\Longrightarrow}_Q \ldots q_{n-1} \stackrel{\widehat{\alpha_n}}{\Longrightarrow}_Q q_n$, wobei $p_j \mathcal{R} q_j$ für alle $0 \leq j \leq n$ gilt. Mit 1.4.1 folgt, dass $q_n \in E_Q$ gelten muss und somit auch $w \in ET_Q$ mit der Begründung von oben.

- Fall 2 ($w \in MIT(P)$): w ist in P ein Input-kritischer Trace. Es existiert also eine Aufteilung von w in va mit $v \in \Sigma^*$ und $a \in I$. Der Trace in P kann wie folgt dargestellt werden: $\exists v' \in \Sigma_\tau^*, \exists p_1, p_2, \ldots, p_n, \exists \alpha_1, \alpha_2, \ldots, \alpha_n : \hat{v'} = v \wedge v' = \alpha_1\alpha_2 \ldots \alpha_n \wedge p_0 \xrightarrow{\alpha_1}_P p_1 \xrightarrow{\alpha_2}_P \ldots p_{n-1} \xrightarrow{\alpha_n}_P p_n \stackrel{a}{\not\longrightarrow}_P$. Basierend auf 1.4.4 bzw. 1.4.5 kann daraus gefolgert werden, dass die α_j auch schwach ausführbar sind in Q, falls kein q_{j-1} angetroffen wird, dass in E_Q enthalten ist für $j \in \{1, 2, \ldots n\}$. Falls eines der q_j mit $0 \leq j \leq n$ ein Fehler-Zustand ist, dann ist w in ET_Q enthalten, da ein Präfix von w ein strikter Fehler-Trace in Q ist. Ansonsten gilt $q_0 \stackrel{\widehat{\alpha_1}}{\Longrightarrow}_Q q_1 \stackrel{\widehat{\alpha_2}}{\Longrightarrow}_Q \ldots q_{n-1} \stackrel{\widehat{\alpha_n}}{\Longrightarrow}_Q q_n$ mit $p_j \mathcal{R} q_j$ für alle $j \in \{0, 1, \ldots, n\}$. Da a für p_n in P keine ausgehende must-Transition sein kann, gilt mit 1.4.2 auch $q_n \stackrel{a}{\not\longrightarrow}$. w ist in $MIT_Q \subseteq ET_Q$ enthalten.

Für den zweiten Punkt kann man sich auf die Inklusion $EL_P \backslash ET_P \subseteq EL_Q$ einschränken, da der erste Punkt bereits vorausgesetzt werden kann. $EL_P \backslash ET_P$ ist eine Teilmenge der Sprache L_P. Somit ist ein w in P ausführbar. Es gibt also einen Trace $p_0 \xrightarrow{\alpha_1}_P p_1 \xrightarrow{\alpha_2}_P \ldots p_{n-1} \xrightarrow{\alpha_n}_P p_n$ in P, wobei w der Aktionsfolge $\alpha_1\alpha_2 \ldots \alpha_n$ entspricht bis auf die internen Aktionen. Erneut können hier 1.4.4 und 1.4.5 dazu verwendet werden einen analogen Trace in Q zu finden. Falls ein q_j für $0 \leq j \leq n$ in E_Q enthalten ist, gilt $w \in ET_Q \subseteq EL_Q$. Es wird also im Folgenden davon ausgegangen, dass kein q_j ein Fehler-Zustand ist. Es gilt dann $q_0 \stackrel{\widehat{\alpha_1}}{\Longrightarrow}_Q q_1 \stackrel{\widehat{\alpha_2}}{\Longrightarrow}_Q \ldots q_{n-1} \stackrel{\widehat{\alpha_n}}{\Longrightarrow}_Q q_n$ in Q mit $p_j \mathcal{R} q_j$ für $0 \leq j \leq n$ und somit $w \in L_Q \subseteq EL_Q$.

$P \sqsubseteq_{\text{as}} Q \not\Leftarrow P \sqsubseteq_{\text{w-as}} Q$:
Im Abbildung 2.5 wird ein Gegenbeispiel dargestellt mit einem MEIO Q und einer schwachen as-Verfeinerung P von Q, die jedoch keine starke as-Verfeinerung von Q ist. Die schwache as-Verfeinerungs-Relationen $\mathcal{R}$ zwischen P und Q enthält die Tupel (p_0, q_0) und (p_1, q_{12}). Damit $\mathcal{R}$ eine schwache Simulations-Relation zwischen P und Q sein kann müssen die Startzustände in Relation stehen. Dies ist durch $(p_0, q_0) \in \mathcal{R}$ erfüllt. Es sind keine Fehler-Zustände in Q und P enthalten, somit ist 1. der Definition 1.4 bereits für beide Zustands-Tupel erfüllt. Für das Tupel (p_1, q_{12}) sind auch 2.-5. von 1.4 erfüllt, da weder p_1 noch q_{12} ausgehende Transitionen besitzen. Für $(p_0, q_0) \in \mathcal{R}$ gibt es keine

ausgehende must-Transitionen. Also ist 2. und 3. von 1.4 bereits erfüllt. Falls α ein Input ist, fordert 1.4.4, dass die Transition $p_0 \overset{\alpha}{\dashrightarrow}_P p_1$ in Q schwach ausführbar ist in der Form $q_0 \overset{\alpha}{\dashrightarrow}_Q \overset{\varepsilon}{\Rightarrow}_Q q$. Ein entsprechendes q ist in diesem Fall q_{12} und es gilt $p_1 \mathcal{R} q_{12}$. Falls α eine lokale Aktion ist, lautet die Forderung $q_0 \overset{\hat{\alpha}}{\Rightarrow}_Q q$ für Q und q_{12} ist wieder der passende Zustand für q, der mit p_1 in Relation stehen. $\mathcal{R}$ ist also eine schwache as-Verfeinerungs-Relation zwischen P und Q.
Angenommen es gibt auch eine starke as-Verfeinerungs-Relation $\mathcal{R}'$ zwischen P und Q, dann muss $p_0 \mathcal{R}' q_0$ gelten. Mit 1.3.3 wird gefordert, dass die Transition $p_0 \overset{\alpha}{\dashrightarrow}_P p_1$ durch eine Transition der Form $q_0 \overset{\alpha}{\dashrightarrow}_Q q$ in Q gematched werden muss. Für den Zustand q kommt dieses mal nur q_{11} in Frage. Es muss also $(p_1, q_{11}) \in \mathcal{R}$ gelten. Der zweite Punkt der Definition 1.3 fordert, dass die τ-must-Transition aus Q auch in P auftauchen muss. Es müsste also ein p geben, für dass $p_1 \xrightarrow{\tau}_P p$ gilt und das Tupel (p, q_{12}) müsste in $\mathcal{R}$ enthalten sein. Da es keine solche Transition gibt, tritt ein Widerspruch zur Annahme auf. Es kann also keine starke as-Verfeinerungs-Relation zwischen P und Q geben.

Abbildung 2.5: Gegenbeispiel zu $\sqsubseteq_{\text{as}} \Leftarrow \sqsubseteq_{\text{w-as}}$

$P \sqsubseteq_{\text{w-as}} Q \not\Leftarrow P \sqsubseteq_E Q$:
Die nicht Gültigkeit dieser Implikation beruht darauf, dass Simulationen strenger sind als Sprach Inklusionen. Das Gegenbeispiel hier ist also so aufgebaut, dass $ET(P) = ET(Q) = \emptyset$ und $L(P) \subseteq L(Q)$ gilt, jedoch keine schwache as-Verfeinerungs-Relation zwischen P und Q existieren kann. Q und P sind in der Abbildung 2.6 dargestellt. Damit $ET(P) = ET(Q) = \emptyset$ gilt, dürfen keine der Zustände Fehler-Zustände sein und es muss gefordert werden, dass die Menge I der Inputs für die MEIOs leer ist, ansonsten würde es Input-kritische Traces gegen. P kann keine Aktionen ausführen und Q nur die Output Aktion o somit gilt für die Sprachen $\{\varepsilon\} = L(P) \subset L(Q) = \{\varepsilon, o\}$.
Angenommen es gibt eine schwache as-Verfeinerungs-Relation $\mathcal{R}$ zwischen P und Q. Dafür muss $(p_0, q_0) \in \mathcal{R}$ gelten. Da es keine Fehler-Zustände in P gibt, ist 1.4.1 erfüllt. Die Punkte 2., 4. und 5. der Definition 1.4 stellen keine Forderungen an die Relation $\mathcal{R}$. Da jedoch die Transition $q_0 \xrightarrow{o}_Q q_1$ in Q vorhanden ist, wird die Verfeinerung dieser in P gefordert es müsste also auch eine must-Output-Transition in P geben. Da diese nicht vorhanden ist, stellt dies einen Widerspruch zur Annahme dar und es folgt, dass es keine schwache as-Verfeinerungs-Relation zwischen P und Q geben kann.

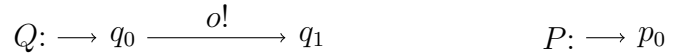

Abbildung 2.6: Gegenbeispiel zu $\sqsubseteq_{\text{w-as}} \Leftarrow \sqsubseteq_E$ mit $I_P = I_Q = \emptyset$

□

In dieser Arbeit werden im Gegensatz zu [BFLV16] die Fehler bei einer Parallelkomposition beibehalten bzw. aus dieser entstehend angesehen. Es werden dabei alle Transitionen, die nicht durch fehlende Synchronisations-Möglichkeiten wegfallen, übernommen. In [BFLV16] hingegen wird der Ansatz verfolgt alle Fehler zu entfernen und durch einen universal Zustand zu ersetzten, der nur eingehende may-Input-Transitionen zulässt. Auf diese Normierung wurde hier mit Absicht verzichtete um sehen zu können, dass die Fehler einen Ursprung haben, den man später auch noch einsehen kann. Jedoch gibt es trotzdem einen Zusammenhang zwischen diesen beiden Ansätzen.
Die Unterscheidung, die die Basisrelation $\sqsubseteq_E^{\mathrm{B}}$ hier herbei führt, würde dort der Unterscheidung zwischen Transitionssystemen, die den universal Zustand e als Startzustand haben und denen, die einen Startzustand ungleich e besitzen, entsprechen. Falls hier in einer as-Implementierung von P ein Fehler lokal erreichbar ist, dann muss auch in P ein Fehler-Zustand lokal erreichbar sein, wegen 2.8 (i). Dies entspricht $\varepsilon \in PrET(P)$. Das Abschneiden den lokalen Aktionen wird hier nur in der Trace Menge praktiziert, in [BFLV16] jedoch direkt auf den Transitionssystemen. Im Fall der lokalen Fehler-Erreichbarkeit bleibt also nur noch der Zustand e als universal Zustand übrig, für den jedes Verhalten zulässig ist. Falls man hier dieses Pruning mit beliebigem Verhalten nachmachen wollen würde, müsste man an den Zustand e noch eine may-Schleife für alle Aktionen aus I und O hinzufügen.
Da auch der Testing-Ansatz die lokale Fehler-Erreichbarkeit verwendet, existiert der Zusammenhang auch dort für die Parallelkomposition mit dem entsprechenden Test.
Falls es keine Input-kritischen Traces gibt, entspricht $\sqsubseteq_E$ der Relation $\sqsubseteq$ aus [BFLV16], falls auf die MEIOs das entsprechende Abschneiden der Fehler-Traces angewendet würde und diese dann durch einen universal Zustand mit may-Schleife für alle Inputs und Outputs ersetzt würden. Gibt es in Q jedoch einen Input-kritische Traces, würde $P \sqsubseteq_E Q$ zulassen, dass P einen gekürzten Fehler-Trace anstatt dessen besitzt. Diese Art der Verfeinerung lässt $\sqsubseteq$ nicht zu. Jedoch sind die Input-kritischen Traces auch dort die potentiellen Auslöser für neue Fehler-Zustände in einer Parallelkomposition. Diese Problem basiert auf dem Problem, dass $\sqsubseteq_E$ keine schwache as-Verfeinerungs-Relation sein muss, die in $\sqsubseteq_{\text{w-as}}$ enthalten ist.

2.4 Konjunktion

Neben der Parallelkomposition ist die Konjunktion einer der wichtigsten Operatoren, die man auf Transitionssysteme anwenden kann, die wie hier als Spezifikationen und Verfeinerungen aufgefasst werden. Dieser Operator erlaubt es unterschiedliche Verhaltensweisen eines Systems separat zu definieren und durch Konjunktion daraus dann eine vollständige Spezifikation des Gesamtsystems zu erhalten. Die Konjunktion sollte also die gröbste Spezifikation sein, die alle gegebenen Spezifikationen, der einzel Komponenten, verfeinert, d.h. sie sollte die größte untere Schranke der Verfeinerungs Präkongruenz charakterisieren.
Um die Konjunktion zwischen zwei MEIOs bilden zu können muss man diese zuerst

normalisieren, da sonst nicht gewolltes Verhalten auftreten kann. Die Konjunktion soll auf dem Kreuzprodukt der beiden Transitionssysteme beruhen, also für MEIOs Q und P eine MEIO der Art $Q \times P$ mit möglicherweise kleinen Abwandlungen sein. Eine Verfeinerung, die beide Spezifikationen erfüllt, kann dabei nur Fehler enthalten, die beide Spezifikationen zulassen. Es sollte also für die Konjunktion $ET_Q \cap ET_P$ und insbesondere $\mathrm{cont}(PrET_Q) \cap \mathrm{cont}(PrET_P)$ gelten. Im Beispiel in Abbildung 2.7 soll verdeutlicht werden, wieso es bei MEIOs ohne Normalisierung zu Problemen mit dieser Forderung kommen kann. Für P un Q ist i ein gekürzter Fehler-Trace. Da jedoch die Output-Transitionen o bzw. o' nicht in das Kreuzprodukt übernommen werden, enthält $Q \times P$ kleinen Fehler-Zustand und somit ist auch die Menge $ET_{Q\times P}$ leer. Durch eine Normalisierung, in der die lokalen Aktionen bereits von einem strikten Fehler-Trace entfernt würden und der letzten Input bereits zu dem Fehler-Zustand führen würde, könnte dieses Problem verhindert werden. Es wären dann also bereits die Zustände die durch die Traces in $PrET$ erreicht werden Fehler-Zustände.

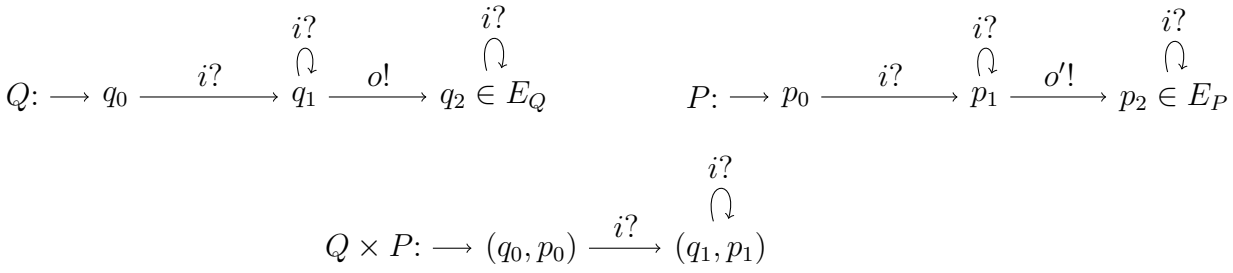

Abbildung 2.7: Beispiel um die Notwendigkeit der Normalisierung zu verdeutlichen

Definition 2.18 (*Normal Form*). *Ein MEIO P ist in* Normal Form *(NF), falls die Menge E_P der Fehler-Zustände nur ein einziges Element enthält $E_P = \{e\}$ und es sollen zusätzlich die folgenden Bedingungen gelten:*

- $e \xrightarrow{a} e$ *für alle* $a \in \Sigma$,
- *falls* $p \overset{\alpha}{\dashrightarrow} e$ *mit* $p \neq e$ *gilt, soll auch* α *ein Element der Menge der Input-Aktionen* I *sein.*

Die Normal Form *von P ist ein MEIO $NF(P)$, das man aus P durch die nachfolgenden zwei Schritte erhält:*

(i) Definiere $\overline{E_P} = \{p \mid \exists p' \in E_P, w \in O^* : p \overset{w}{\Longrightarrow}_P p'\}$ *und ersetzte* E_P *durch* $\overline{E_P}$.

(ii) Falls $p_0 \in \overline{E_P}$, *ist* $NF(P) = (\{p_0\}, I, O, \{p_0 \xrightarrow{a} p_0 \mid a \in \Sigma\}, p_0, \{p_0\})$.
Ansonsten, füge einen neuen Zustand e hinzu, der der einzige Fehler-Zustand wird. Wann immer $p \overset{\alpha}{\dashrightarrow}_P p' \in \overline{E_P}$ *und* $p \notin \overline{E_P}$ *für ein* $\alpha \in \Sigma \cup \{\tau\}$ *gilt (dann gilt zwingendermaßen* $\alpha \in I$*), entferne alle* α*-Transitionen, die von P ausgehen und füge die Transition* $p \overset{\alpha}{\dashrightarrow}_{NF(P)} e$ *und falls* $p \xrightarrow{\alpha}_P p' \in \overline{E_P}$ *galt auch* $p \xrightarrow{\alpha}_{NF(P)} e$ *hinzu. Am Ende füge alle* $e \xrightarrow{a}_{NF(P)} e$ *mit* $a \in \Sigma$ *hinzu.*

Schritt (i) verändert nichts an den Transitionen und auch nichts daran, ob ein Zustand einen nicht sichergestellten Input hat oder nicht. Die Menge $StET$ wird verändert jedoch bleibt die Menge $PrET$ gleich, da $StET$ am Ende alle Traces enthält, die direkt oder durch lokale Aktionen zu einem Fehler-Zustand des ursprünglichen MEIOs P führen. Für die Menge $MIT \backslash \text{cont}(PrET)$ ändert sich nichts, da sich an Zuständen ohne einen Zusammenhang mit Fehler-Zuständen nichts ändert. Das Ergebnis von (i) entspricht also dem MEIO P bezüglich der Relation $\sqsubseteq_E$. Im ersten Fall von Schritt (ii) gilt $ET_P = \Sigma^* = ET(NF(P))$. Im zweiten Fall ändert der Schritt (ii) nichts an Zuständen, die nicht sichergestellte Inputs haben, solange diese nicht in der Menge $\overline{E_P}$ enthalten sind, da eine must-Transition die zum Zustand e führt nur hinzu gefügt wird, wenn der entsprechende Input davor bereits eine ausgehende must-Transition des Zustands p war. Die Menge MIT verkleinert sich jedoch, falls der Schnitt mit $\text{cont}(PrET)$ nicht leer war. Die Menge der Fehler-Traces ET bleibt somit im Schritt (ii) gleich. Die Sprache des MEIOs P wird durch die Konstruktion in (ii) zunächst um die Traces in $\text{cont}(PrET)$ verkleinert. Jedoch durch die must-Schlinge für alle sichtbaren Aktionen am Fehler-Zustand, enthält die Sprache von $NF(P)$ em Ende sogar alle Traces aus $\text{cont}(PrET)$ zusätzlich zu den Wörtern, die bereits in $L(P)$ enthalten waren. Die Menge EL wird mit ET geflutet und es gilt deshalb für EL auch $EL(P) = EL(NF(P))$. Somit ist $NF(P)$ äquivalent zu P bezüglich der Relation $\sqsubseteq_E$. Es gilt also die folgende Proposition.

Proposition 2.19 (*Normal Form*). *Jedes MEIO P ist äquivalent zu seiner Normal Form $NF(P)$ bezüglich der Relation $\sqsubseteq_E$.*

Somit können wird davon ausgehen, dass ein MEIO in Normal Form ist, falls dies notwendig oder hilfreich ist.
Die Transformation in die Normal Form aus der Definition 2.18 ist effizient für endliche MEIOs: Für Schritt (i) kann eine Tiefensuche (DFS) in linearer Zeit von den Zuständen in E über umgekehrte mit lokalen Aktionen beschriftete Transitionen ausgeführt werden. Der erste Fall von Schritt (ii) benötigt nur konstante Zeit, da ausschließlich der Startzustand überprüft werden muss. Für den zweiten Fall kann erneut eine DFS ausgeführt werden. Dies DFS soll vom Startzustand p_0 ablaufen ohne dabei die Transitionen zu benutzen, die gelöscht werden sollen. Alle Zustände, die dadurch nicht besucht werden sind nicht erreichbar in $NF(P)$ und können entfernt werden. $NF(P)$ ist also möglicherweise deutlich kleiner wie P.
Es wurden durch die Normalisierung der MEIOs das Problem mit den Traces in $\text{cont}(PrET)$ gelöst, und man erhält dann Transitionssysteme, die die Eigenschaften der MIAs aus z.B. [BFLV16] erfüllen. Jedoch kann die Konjunktion hier trotzdem nicht auf die gleiche Art gebildet werden, wie dort. Es gibt noch ein weiteres ungelöstes Problem. Es beruht auf den Input-kritischen Traces. In [BFLV16] kann ein Zustand in der Konjunktion, der für eines der Transitionssysteme P einen must-Input hat und für des andere Transitionssystem Q keine entsprechende may-Transition besitzt, als inkonsistent angesehen werden und aus der Konjunktion entfernt werden. Dies würde jedoch hier zu einem Problem mit den Traces führen. Es soll für die Komposition $ET_{Q \wedge P} = ET_Q \cap ET_P$ und $EL_{Q \wedge P} = EL_Q \cap EL_P$ gelten. Der Trace in Q mit einem nicht sichergestellten Input ist

ein Input-kritischer Trace. Er ist also in der Menge $ET_Q \subseteq EL_Q$ enthalten. Da es in P ein must-Transition gibt ist der entsprechende Trace in der Sprache von P enthalten, also auch in der Menge EL_P. Es muss also nach den geforderten Eigenschaften der Konjunktion auch gelten, dass der Trace in der Fehler-gefluteten Sprache der Konjunktion enthalten ist. Durch das entfernen des Zustandes wie in [BFLV16] würde der Trace in den meisten Fällen kein Element der Fehler-gefluteten Sprache sein (siehe dazu auch das Beispiel in Abbildung 2.8). Im nachfolgenden Beispiel würde die Komposition von Q und P für [BFLV16] nicht existieren, da der Zustand (q_0, p_0) als (logisch) inkonsistent angesehen wird.

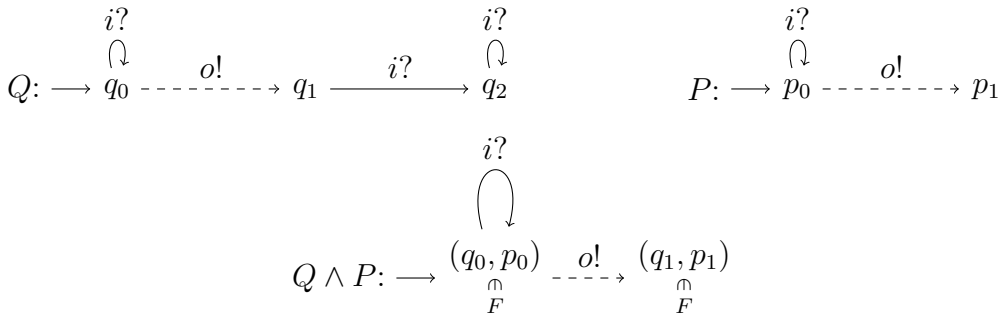

Abbildung 2.8: Beispiel für MIT-Problem, $\wedge$ bezeichnet die Konjunktion aus [BFLV16]

Man kann diese Problem lösen, in dem man die Zustände aus dem in Konjunktion entstehenden Transitionssystemen nicht streicht sondern beibehält und die Transition zu einem Zustands-Tupel aus dem Zustand, der durch die must-Transition in P erreicht wird und dem Fehler-Zustand von Q führt. Falls in Q der Input als may-Transition ausführbar ist. Zählt der Trace in Q bereits als Input-kritisch und es kommt nicht mehr darauf an, ob die Verlängerungen ausführbar sind. Die Verlängerungen sind alle in der Menge ET_Q enthalten. Die Ausführbarkeit von Aktionen nach der Input-Transition in der Komposition hängt also nur von P ab. Die Transitionen der Komposition sollten also nach dem Input direkt von P geerbt werden, dieses Verhalten erhält man, in dem man den may-Input von Q auf den Zustand e_Q umbiegt.
Es ergibt sich das folgende Lemma.

Lemma 2.20. *Sei P ein MEIO in Normal Form (mit dem Fehler-Zustand e). $w \in \text{cont}(PrET(P))$ gilt genau dann, wenn w sich aus einem Ablauf ergibt, der e erreicht. $va = w \in MIT \backslash \text{cont}(PrET(P))$ mit $v \in \Sigma^*$ und $a \in I$ gilt genau dann, wenn v ein Ablauf von P ist, der zu einem Zustand führt, für den a keine ausgehende must-Transition ist.*

Auch die Behandlung von Outputs in Fällen, in denen das eine MEIO eine must-Transition und das andere keine Möglichkeit für den Output besitzt, führt zu Problemen mit den Traces, die möglich sein sollten, da sie im Schnitt der EL-Mengen enthalten sind. Es wird in dieser Arbeit also verzichtet (logisch) inkonsistente Zustände wie z.B.

in [BFLV16] zu definiere und diese aus den Transitionssystemen zu entfernen. Man kann sich von der Korrektheit dieses Vorgehens recht leicht dadurch überzeugen, dass für alle MEIOs ε in der Menge EL enthalten ist, also muss auch eine Konjunktion zweier MEIOs das leere Wort immer ausführen können. Es gibt also immer eine Konjunktion. Mit inkonsistenten Zuständen kann es jedoch ,wie im Beispiel in Abbildung 2.8 gesehen, dazu kommen, dass der Startzustand inkonsistent ist und somit die Konjunktion nicht existieren würde. Die Definition der Konjunktion orientiert sich somit eher an der des konjunktiven Produkts aus [BFLV16], wie an der Definition der Konjunktion aus [BFLV16]. Jedoch mussten für die Input-Transitionen einige Veränderungen vorgenommen werden.

Definition 2.21 (*Konjunktion*). *Für zwei MEIOs* $P_1 = (P_1, I, O, \longrightarrow_1, \dashrightarrow_1, p_{01}, \{e_1\})$ *und* $P_2 = (P_2, I, O, \longrightarrow_2, \dashrightarrow_2, p_{02}, \{e_2\})$ *in Normal Form mit der selben Signatur ist die* Konjunktion *definiert als* $P_1 \wedge P_2 = (P_1 \times P_2, I, O, \longrightarrow, \dashrightarrow, (p_{01}, p_{02}), \{(e_1, e_2)\})$ *mit den folgenden Regeln für die Transitionen:*

(OMust) $(p_1, p_2) \xrightarrow{\omega} (p_1', p_2')$, *falls* $p_1 \xrightarrow{\omega}_1 p_1'$ *und* $p_2 \stackrel{\hat{\omega}}{\Longrightarrow}_2 p_2'$ *oder* $p_1 \stackrel{\hat{\omega}}{\Longrightarrow}_1 p_1'$ *und* $p_2 \xrightarrow{\omega}_2 p_2'$,

(IMust1) $(p_1, p_2) \xrightarrow{i} (p_1', p_2')$, *falls* $p_1 \xrightarrow{i}_1 p_1'$ *und* $p_2 \xrightarrow{i}_2 \stackrel{\varepsilon}{\Longrightarrow}_2 p_2'$ *oder* $p_1 \xrightarrow{i}_1 \stackrel{\varepsilon}{\Longrightarrow}_1 p_1'$ *und* $p_2 \xrightarrow{i}_2 p_2'$,

(IMust2) $(p_1, p_2) \xrightarrow{i} (p_1', e_2)$, *falls* $p_1 \xrightarrow{i}_1 p_1'$ *und* $p_2 \not\xrightarrow{i}_2$,

(IMust3) $(p_1, p_2) \xrightarrow{i} (e_1, p_2')$, *falls* $p_1 \not\xrightarrow{i}_1$ *und* $p_2 \xrightarrow{i}_2 p_2'$,

(EMust1) $(p_1, e_2) \xrightarrow{\alpha} (p_1', e_2)$, *falls* $p_1 \xrightarrow{\alpha}_1 p_1'$,

(EMust2) $(e_1, p_2) \xrightarrow{\alpha} (e_1, p_2')$, *falls* $p_2 \xrightarrow{\alpha}_2 p_2'$,

(May1) $(p_1, p_2) \stackrel{\tau}{\dashrightarrow} (p_1', p_2)$, *falls* $p_1 \stackrel{\tau}{\Longrightarrow}_1 p_1'$,

(May2) $(p_1, p_2) \stackrel{\tau}{\dashrightarrow} (p_1, p_2')$, *falls* $p_2 \stackrel{\tau}{\Longrightarrow}_2 p_2'$,

(OMay) $(p_1, p_2) \stackrel{\omega}{\dashrightarrow} (p_1', p_2')$, *falls* $p_1 \stackrel{\omega}{\Longrightarrow}_1 p_1'$ *und* $p_2 \stackrel{\omega}{\Longrightarrow}_2 p_2'$,

(IMay1) $(p_1, p_2) \stackrel{i}{\dashrightarrow} (p_1', p_2')$, *falls* $p_1 \xrightarrow{i}_1 \stackrel{\varepsilon}{\Longrightarrow}_1 p_1'$ *und* $p_2 \xrightarrow{i}_2 \stackrel{\varepsilon}{\Longrightarrow}_2 p_2'$,

(IMay2) $(p_1, p_2) \stackrel{i}{\dashrightarrow} (p_1', e_2)$, *falls* $p_1 \stackrel{i}{\dashrightarrow}_1 p_1'$ *und* $p_2 \not\xrightarrow{i}_2$,

(IMay3) $(p_1, p_2) \stackrel{i}{\dashrightarrow} (e_1, p_2')$, *falls* $p_1 \not\xrightarrow{i}_1$ *und* $p_2 \stackrel{i}{\dashrightarrow}_2 p_2'$,

(EMay1) $(p_1, e_2) \stackrel{\alpha}{\dashrightarrow} (p_1', e_2)$, *falls* $p_1 \stackrel{\alpha}{\dashrightarrow}_1 p_1'$,

(EMay2) $(e_1, p_2) \stackrel{\alpha}{\dashrightarrow} (e_1, p_2')$, *falls* $p_2 \stackrel{\alpha}{\dashrightarrow}_2 p_2'$.

Aus schwachen Transitionen der einzelnen MEIOs entstehen im der Konjunktion einzelnen Transitionen (z.B. (OMust), (IMust1), (May)). τ-Transitionen können via der Regel (OMay) synchronisiert werden. In den Regel (EMust) und (EMay) wirken die Fehler-Zustände als neutrales Element. In der Parallelkomposition hingeben hatten die Fehler-Zustände absorbierende Wirkung.
P_1 und P_2 müssen für die Definition 2.21 in Normal Form sein. Der daraus resultierende MEIO $P_1 \wedge P_2$ ist ebenfalls wieder in Normal Form. Es gibt nur den Zustand (e_1, e_2) als Fehler-Zustand. Da für e_1 und e_2 bereits alle Aktionen via must-Schleife ausführbar sind, hat auch der Zustand (e_1, e_2) die Möglichkeit alle Aktionen via must-Transition zu sich selbst auszuführen. Die Transitionen, die in P_1 und P_2 zu den Fehler-Zustand führen sind bereits alle Inputs. Es kann also in der Konjunktion durch das gemeinsame Ausführen der Transitionen nur Inputs zum Zustand (e_1, e_2) führen. Es kann jedoch auch durch Input-kritische Traces der Komponenten möglich sein, den Zustand (e_1, e_2) zu erreicht. Hierfür kommen aber nur die Regel (IMust2), (IMust3), (IMay2) und (IMay3) in Frage. Diese befassen sich jedoch nur mit Input-Transitionen. Es kann also durch diese Regeln auch keine zu (e_1, e_2) führende Transition entstehen, die mit einer lokalen Aktion beschriftet wäre.
Auf Grundlage dieser Definitionen kann nun gezeigt werden, dass es sich bei der Konjunktion $\wedge$ um die gewünschte größte untere Schranke bezüglich der Relation $\sqsubseteq_E$ handelt.

Satz 2.22 (*Konjunktion*). *Für drei MEIOs S, P_1 und P_2 mit der gleichen Signatur gilt $S \sqsubseteq_E P_1 \wedge P_2$ genau dann, wenn $S \sqsubseteq_E P_1$ und $S \sqsubseteq_E P_2$ erfüllt ist.*

Beweis. Die Definition von $\wedge$ setzt normalisierte MEIOs voraus. Deshalb ist es sinnvoll auch hier von MEIOs P_1 und P_2 in Normal Form auszugehen. Dies ist möglich durch die Aussage der Proposition 2.19.

„$\Leftarrow$“:
Um diese Richtung zu beweisen muss gezeigt werden, dass $ET_S \subseteq ET_{P_1 \wedge P_2}$ und $EL_S \subseteq EL_{P_1 \wedge P_2}$ gilt, wenn $ET_S \subseteq ET_1$, $EL_S \subseteq EL_1$, $ET_S \subseteq ET_2$ und $EL_S \subseteq EL_2$ vorausgesetzt wird.

Als erstes wird versucht die Teilmengenbeziehung $ET_S \subseteq ET_{P_1 \wedge P_2}$ zu beweisen. Dafür wird ein beliebiges w aus der Menge ET_S gewählt. Das w muss ebenfalls in $ET_1 \cap ET_2$ enthalten. In Lemma 2.20 wurde ein Unterschied zwischen Elementen der Menge $\text{cont}(PrET(P))$ und $\text{cont}(MIT(P))$ deutlich. Dies ist auch der Grund für die nachfolgende Fallunterscheidung.

- Fall 1 ($w \in (\text{cont}(PrET_1) \cap \text{cont}(PrET_2))$): Es führt also nach Lemma 2.20 sowohl in P_1 wie auch in P_2 der Ablauf von w zum Fehler-Zustand e_1 bzw. e_2. Da in beiden Transitionssystemen der Ablauf von w möglich ist, enthält auch $P_1 \wedge P_2$ die dafür notwendigen Transitionen. Das w führt in der Konjunktion zum Zustand (e_1, e_2) und ist wegen 2.20 in der Menge $ET_{P_1 \wedge P_2}$ enthalten.

- Fall 2 ($w \in (\text{cont}(MIT_1) \cap \text{cont}(MIT_2))$): Es existieren Präfixe w_1 und w_2 von w für die $w_1 \in MIT_1$ und $w_2 \in MIT_2$ gilt.

  - Fall 2a) ($w_1 = w_2$): In beiden P_j ist das v, für das es einen Input a gibt, so dass $w_1 = w_2 = va$ gilt, Element der Sprache L_j. Das v muss wegen 2.20 in den MEIOs P_j zu jeweils einem Zustand p_j führen, in dem das a nicht sichergestellt ist. In der Komposition muss es einen Ablauf von v geben, der zu dem Zustand (p_1, p_2) führt. Da für keinen der beiden Zustände p_j der Input a eine ausgehende must-Transition ist, kann es mit Definition 2.21 auch keine mit a beschriftete ausgehende must-Transition für (p_1, p_2) geben. Es gilt also mit Lemma 2.20 $w_1 = w_2 \in MIT_{P_1 \wedge P_2}$ und somit $w \in ET_{P_1 \wedge P_2}$.

  - Fall 2b) ($w_1 \neq w_2$): OBdA ist w_1 kürzer wie w_2. Es gibt Wörter v_1, v_2 und Inputs a_1, a_2, so dass $w_1 = v_1 a_1$ und $w_2 = v_2 a_2$ gilt. v_1 ist ein echtes Präfix von v_2. Es ist also v_1 in beiden MEIOs P_j ausführbar. In P_1 wird dadurch ein Zustand p_1 erreicht für den $p_1 \xrightarrow{a_1}\!\!\!\!\!\!/\;$ gilt. Für den Zustand p_2, den P_2 durch v_1 erreicht gilt jedoch $p_2 \xrightarrow{a_1}$. In der Konjunktion kommt also die Regel (IMust3) der Definition 2.21 zur Anwendung. Die restlichen Transitionen von v_2 nach der Ausführung von w_1 erbt die Konjunktion also direkt von P_2. In $P_1 \wedge P_2$ führt der Ablauf von v_2 zu einem Zustand der Form (e_1, p_2'). Für p_2' ist a_2 keine ausgehende must-Transition. Da für e_1 alle Aktionen eine must-Transition zu e_1 besitzen kann erneut die Regel (IMust3) der Definition 2.21 angewendet werden. Durch w_2 wird in $P_1 \wedge P_2$ der Zustand (e_1, e_2) erreicht. Es gilt durch Lemma 2.20 $w_2 \in PrET_{P_1 \wedge P_2}$ und daraus folgt dann auch $w \in ET_{P_1 \wedge P_2}$.

- Fall 3 ($w \in (\text{cont}(PrET_1) \cap \text{cont}(MIT_2))$ oder $w \in (\text{cont}(MIT_1) \cap \text{cont}(PrET_2))$): Die beiden hier Vereinigten Fälle sind symmetrisch, somit kann wegen der Kommunativität der Konjunktioin oBdA davon ausgegangen werden, dass $w \in (\text{cont}(PrET_1) \cap \text{cont}(MIT_2))$ gilt. In P_1 führt der Ablauf w zum Fehler-Zustand e_1. Es gibt ein Präfix w' von w, dass in MIT_2 enthalten ist. In P_2 ist das Präfix v von w' ausführbar, für das es einen Input a gibt, so dass $w' = va$ gilt. va ist entweder in MIT_1 enthalten und es kann Fall 2 angewendet werden, oder der Input a ist in P_1 nach der Ausführung von v durch eine must-Transition möglich. Falls in P_1 die a-must-Transtion enthalten ist, folgt mit der Definition 2.21, dass in der Konjunktion durch w' ein Zustand erreicht werden muss, dessen zweites Element des Tupels e_2 ist. Somit wird in $P_1 \wedge P_2$ durch w der Zustand (e_1, e_2) erreicht. Mit Lemma 2.20 folgt daraus $w \in ET_{P_1 \wedge P_2}$.

Ein beliebiges w der fehler-gefluteten Sprache des MEIOs S muss in P_1 und P_2 ebenfalls ein Wort aus der jeweiligen Menge EL_j sein. Da die fehler-geflutete Sprache die Vereinigung der Sprache L und der Fehler-Traces ET ist, können unterschiedliche Fälle für das Wort w eintreten.

- Fall I ($w \in ET_1 \cap ET_2$): In diesem Fall gilt aufgrund des Beweises der vorangegangen Inklusion $w \in ET_{P_1 \wedge P_2} \subseteq EL_{P_1 \wedge P_2}$.

- Fall II ($w \in ET_1 \cap L_2 \backslash ET_2$ oder $w \in L_1 \backslash ET_1 \cap ET_2$): Dieses beiden Fälle sind symmetrisch und da die Konjunktion kommunaitv ist kann oBdA davon ausgegangen werden, dass $w \in ET_1 \cap L_2 \backslash ET_2$ gilt. Es gibt also in P_2 einen Ablauf des Wortes w.
  - Fall IIa) ($w \in \text{cont}(PrET_1)$): Es gibt wegen Lemma 2.20 auch in P_1 einen Ablauf, der mit w beschriftet ist. In diesem Fall gilt also $w \in L_{P_1 \wedge P_2} \subseteq EL_{P_1 \wedge P_2}$.
  - Fall IIb) ($w \in \text{cont}(MIT_1) \backslash \text{cont}(PrET_1)$): w ist in P_1 nicht ausführbar. Es gibt ein echtes Präfix v von w, dass in P_1 zu einem Zustand führt, der einen Input a nicht sicherstellt, so dass va ebenfalls ein Präfix von w ist. Da $w \notin ET_2$ gilt, kann va in P_2 kein Input-kritischer Trace sein. Es muss also nach dem Ablauf v in P_2 ein Zustand erreicht werden, der für den Input a einen ausgehende must-Transition besitzt. Es folgt mit Definition 2.21, dass in der Konjunktion durch va ein Zustand erreicht wird, der als erstes Element des Tupels den Zustand e_1 enthält. Die noch fehlenden Transitionen des Wortes w nach der Ausführung von va werden also direkt von P_2 geerbt. Somit gibt es in $P_1 \wedge P_2$ einen mit w beschrifteten Ablauf und es gilt $w \in L_{P_1 \wedge P_2} \subseteq EL_{P_1 \wedge P_2}$.
- Fall III ($w \in L_1 \backslash ET_1 \cap L_2 \backslash ET_2$): Das Wort w ist in beiden MEIOs P_j ausführbar. Die dafür verwendeten Transitionen werden durch die Definition 2.21 in die Konjunktion übernommen. Es gibt also auch in $P_1 \wedge P_2$ einen Ablauf, der mit w beschriftet ist. Somit gilt $w \in L_{P_1 \wedge P_2} \subseteq EL_{P_1 \wedge P_2}$.

„$\Rightarrow$“:
Für diese Richtung des Beweises können die Inklusionen $ET_S \subseteq ET_{P_1 \wedge P_2}$ und $EL_S \subseteq EL_{P_1 \wedge P_2}$ vorausgesetzt werden um $ET_S \subseteq ET_1$, $EL_S \subseteq EL_1$, $ET_S \subseteq ET_2$ und $EL_S \subseteq EL_2$ zu zeigen. Da die Konjunktion aufgrund ihrer Definition in 2.21 kommutativ ist, reicht es die Inklusionen für einen der beiden MEIOs zu zeigen.

Als erstes soll die Inklusion $ET_S \subseteq ET_1$ betrachtet werden. Durch die Voraussetzungen gilt für ein beliebiges w aus ET_S auch die Zugehörigkeit zur Menge $ET_{P_1 \wedge P_2}$. Es wird also durch w entweder der Fehler-Zustand (e_1, e_2) in der Konjunktion von P_1 und P_2 erreicht oder durch ein Präfix v von w ein Zustand, in dem der Input a, der in w auf das v folgen würde, nicht sicher gestellt ist.

- Fall 1 ($w \in \text{cont}(PrET_{P_1 \wedge P_2})$): Der Ablauf von w führt zum Zustand (e_1, e_2). Es muss also durch die Konjunktion von P_1 und P_2 der Zustand e_1 durch das Wort w in der ersten Komponente der Zustands-Tupel erreicht worden sein. Dies ist möglich durch eine nicht sichergestellte Input-Transition innerhalb des ws in P_1 oder dadurch, dass w ein Ablauf in P_1 ist, der zum Fehler-Zustand e_1 führt. Das Wort w ist in beiden Fällen in der Menge ET_1 enthalten.
- Fall 2 ($w \in \text{cont}(MIT_{P_1 \wedge P_2})$): Das Präfix v von w ist in $P_1 \wedge P_2$ ausführbar zu einem Zustand (p_1, p_2), für den $(p_1, p_2) \overset{a}{\not\rightarrow}_{P_1 \wedge P_2}$ gilt. Es darf also für die Zustände p_1 und p_2 ebenfalls keine ausgehenden a-must-Transition geben, da sonst 2.21 auch

eine must-Transition in der Konjunktion der beiden MEIOs fordern würde. va ist also für P_1 ein Input-kritischer Trace mit $va \in MIT_1 \subseteq ET_1$. Da die Menge ET unter Fortsetzung abgeschlossen ist, gilt auch $w \in ET_1$.

Um die Inklusion $EL_S \subseteq EL_1$ zu beweisen, wird ein beliebiges w aus der Menge EL_S gewählt. Nach Voraussetzung ist das w auch in der fehler-gefluteten Sprache der Konjunktion $P_1 \wedge P_2$ enthalten. Falls es in $P_1 \wedge P_2$ keinen Ablauf für w gibt, gilt $w \in \text{cont}(MIT_{P_1 \wedge P_2})$, dieser Fall führt zu $w \in ET_1 \subseteq EL_1$, wie bereits in Fall 2 der letzten Inklusion gezeigt wurde. Es kann also davon ausgegangen werden, dass w in der Konjunktion ausführbar ist. Der Fall w führt in $P_1 \wedge P_2$ zum Fehler-Zustand wurde auch bereits von der letzten Inklustion in Fall 1 behandelt und führt zu $w \in ET_1 \subseteq EL_1$. Man kann sich für diesen Beweisteil deshalb sogar auf Abläufe von w einschränken, die zu einem Zustand in $P_1 \wedge P_2$ führen, der nicht der Fehler-Zustand ist. Jedoch ist es trotzdem möglich, dass durch w ein Zustand erreicht wird, in dessen Tupel einer der Zustände der Fehler-Zustand der jeweiligen Komponente ist.

- Fall I ($(p_{01}, p_{02}) \stackrel{w}{\Rightarrow}_{P_1 \wedge P_2} (p_1, p_2)$ mit $p_1 \neq e_1$): Die Konjunktion kann für das Wort w nur Transitionen verwendet haben, die in P_1 mindestens via schwacher may-Transitionen möglich waren. Es muss also auch in P_1 einen Ablauf des Wortes w geben, der ebenfalls zum Zustand p_1 führt. Es gilt also $w \in L_1 \subseteq EL_1$.
- Fall II ($(p_{01}, p_{02}) \stackrel{w}{\Rightarrow}_{P_1 \wedge P_2} (e_1, p_2)$): Da das Zustands-Tupel (e_1, p_2) den Fehler-Zustand von P_1 in der ersten Komponente enthält, gibt es zwei Möglichkeiten für P_1 wie dieser Zustand in der Konjunktion erreicht wurde.
  - Fall IIa) (w ausführbar in P_1): P_1 kann durch w den Zustand e_1 erreichen. Es gilt also $w \in L_1 \cap ET_1 \subseteq EL_1$.
  - Fall IIb) (w nicht vollständig ausführbar in P_1): Es gibt ein Präfix w' von w, das man in das Wort v und den Input a aufteilen kann, so dass $w' = va$ gilt und durch v in P_1 ein Zustand erreicht wird, für den der Input a nicht sichergestellt ist. In der Konjunktion kam die Transitions-Regel (IMust3) oder (IMay3) der Definition 2.21 zum Einsatz um einen Zustand zu erreichen, der e_1 als erste Komponente enthält. Es gilt in diesem Fall also $va \in MIT_1 \subseteq ET_1$. Wegen der Abgeschlossenheit der Menge ET unter Fortsetzungen gilt $w \in ET_1 \subseteq EL_1$.

□

3 Verfeinerungen für Kommunikationsfehler- und Stillstand-Freiheit

In diesem Kapitel wird die Menge der betrachteten Zustandsmengen von den Kommunikations-basierten Fehlern im letzten Kapitel erweitert um stille Zustände. Es wird nur noch der Testing-Ansatz des letzten Kapitels fortgeführt, da dieser sich auf die Parallelkomposition von EIOs stützt und nicht auf die Definition der Parallelkomposition von MEIOs, die auch anders gestaltet hätte werden können.

3.1 Testing-Ansatz

Zustände, die keine must-Outputs ohne einen Input ausführen können, werden als in einer Art Verklemmung angesehen, da sie ohne Zutun von Außen den Zustand nicht mehr verlassen können, falls ein möglicherweise vorhandener may-Output nicht implementiert wird. So ein Zustand hat also keine must-Transitions-Möglichkeiten für einen Output. Falls dieser Zustand die Möglichkeit für eine interne Aktion via einer must-Transition hat, darf durch die τs niemals ein Zustand erreicht werden, von dem aus ein Output in Implementierungen sicher gestellt wird. Ein Zustand, der keine Outputs und τs via must-Transitionen ausführen kann, ist also ein Deadlock-Zustand, in denen das System nichts mehr tun können muss ohne einen Input. Wenn man eine Erweiterung um τs zu Zuständen ohne must-Outputs zulässt, hat man zusätzlich noch Verklemmungen der Art Livelock, da diese Zustände möglicherweise beliebig viele interne Aktionen ausführen können, jedoch nie aus eigener Kraft einen wirklichen Fortschritt in Form eines Outputs bewirken können müssen. Die Menge der Zustände, die sich in einer Verklemmung befinden, würde also durch $\left\{p \in P \mid \forall a \in O : p \overset{a}{\Rightarrow\!\!\!\!\!/}_P\right\}$ beschrieben werden. Somit wären dies alle Zustände, die keine Möglichkeit haben ohne einen Input von Außen oder eine implementierte may-Output-Transition je wieder einen Output machen zu können. Falls man diese Definition verwenden würde, müsste man immer alle Zustände betrachten, die durch τs erreichbar sind. Dies würde einige Betrachtungen deutlich aufwendiger machen und soll deshalb hier nicht behandelt werden. Die Definition für die betrachteten Verklemmungen, hier Stille genannt, beschränkt sich auf Zustände, die keine Outputs und τs via must-Transitionen ausführen können.

Definition 3.1 (*Stillstand*). *Ein* stiller Zustand *ist ein Zustand in einem MEIO P, der keine Outputs und kein τ zulässt via must-Transitionen.*
Somit ist die Menge der stillen Zustände in einem MEIO P wie folgt formal definiert: $Qui(P) := \left\{p \in P \mid \forall \alpha \in (O \cup \{\tau\}) : p \overset{\alpha}{\not\rightarrow}_P\right\}$.

Für die Erreichbarkeit wird wie im letzten Kapitel ein optimistischer Anzahl der lokalen Erreichbarkeit für die Fehler-Zustände verwendet. Stille ist kein unabwendbare „Fehler-Art“, sondern kann durch einen Input repariert werden oder im Fall von vorhandenen may-Output-Transitionen oder may-τ-Transitionen, durch eine Implementierung dieser lokalen Aktionen . Daraus ergibt sich, dass Stille im Vergleich zu den Fehlern aus dem letzten Kapitel als weniger „schlimmer Fehler“ anzusehen ist. Somit ist ein stiller Zustand ebenso wie ein Fehler-Zustand erreichbar, sobald er durch Outputs und τs erreicht werden kann, jedoch ist nicht jede beliebige Fortsetzung eines Traces, das durch lokale Aktionen zu einem stillen Zustand führt ein Stille-Trace.

Definition 3.2 (*Test und Verfeinerung für Stillstand*). *Ein* Test T *ist eine Implementierung. Ein MEIO P* as-erfüllt *einen Stillstands-Test T, falls $S \| T$ fehler- und stillstand-frei ist für alle $S \in$ as-impl(P). Es wird dann $P \, sat^{Qui}_{\text{as}} \, T$ geschrieben. Die Parallelkomposition $S \| T$ ist* fehler- *und* stillstand-frei*, wenn kein Fehler- und kein stiller Zustand lokal erreichbar ist.*
Ein MEIO P Stille-verfeinert *P', falls für alle Tests T: $P' \, sat^{Qui}_{\text{as}} \, T \Rightarrow P \, sat^{Qui}_{\text{as}} \, T$.*

Um eine genauere Auseinandersetzung mit den Präkongruenzen zu ermöglichen, benötigt man wie im letzten Kapitel die Definition von Traces auf der Struktur. Wie bereits oben erwähnt, ist Stille ein reparierbares Fehlverhalten im Gegensatz zu Fehlern. Es genügt deshalb für Stille die strikten Traces ohne Kürzung zu betrachten.

Definition 3.3 (*Stillstands-Traces*). *Sei P ein MEIO und definiere:*

- strikte Stille-Traces*:* $StQT(P) := \left\{w \in \Sigma^* \mid p_0 \overset{w}{\Rightarrow}_P p \in Qui(P)\right\}$.

Die Definition der strikten Stille-Traces befasst sich nur mit MEIOs im allgemeinen. Jedoch ist auch in diesem Kapitel der Zusammenhang mi den entsprechenden Traces der as-Implementierungen relevant.

Proposition 3.4 (*Stillstands-Traces und Implementierungen*). *Für ein MEIO P gilt für die strikte Stille-Traces:* $StQT(P) \subseteq \bigcup_{P' \in \text{as-impl}(P)} StQT(P')$.

Beweis. Analog zu den Propositionen 1.8 und 2.4 ist die Inklusion am Besten mit einer as-Implementierung zu zeigen und der entsprechenden as-Verfeinerungs-Relation $\mathcal{R}$. Falls der Startzustand p_0 von P ein stiller Zustand ist, muss man zwei as-Implementierungen betrachten, ansonsten genügt es eine für alle w aus $StQT(P)$ anzugeben, wobei w möglicherweise nicht ε entsprechend darf.
Die as-Implementierung P' für den Fall $p_0 \in Qui(P)$ implementiert alle must-Transitionen, keine may-Transitionen und keine Fehler-Zustände von P und hat die Identitäts-Relation als starke as-Verfeinerungs-Relation $\mathcal{R}$. In diesem Fall gilt $\varepsilon \in StQT(P)$. Für

alle $a \in \Sigma$ folgt, wenn in P für einen Zustand p $p \overset{a}{\not\longrightarrow}_P$, gilt auch in P' $p' \overset{a}{\not\longrightarrow}_{P'}$ für den Zustand, der mit p in Relation steht. Der Startzustand von P' ist mit ε erreichbar und ebenfalls still. Es gilt also $p'_0 \in Qui(P')$ und $\varepsilon \in StQT(P')$. Der 2. Punkt der Definition 1.3 ist für die Identitäts-Relation als as-Verfeinerungs-Relation $\mathcal{R}$ erfüllt, da alle must-Transitionen aus P entsprechend in P' umgesetzt wurden. Alle must-Transitionen in P müssen zugrundeliegende may-Transitionen haben, somit gilt auch 3. von 1.3. Der 1. Punkt der Definition ist auch erfüllt, da $E_{P'} = \emptyset$ gilt, wenn keine Fehler-Zustände implementiert werden.
Für alle $w \neq \varepsilon$ und für $w = \varepsilon$, dass zu einem anderen stillen Zustand führt wie p_0, mit $w \in StQT(P)$ kann P' als die folgende as-Implementierung gewählt werden:

- $P' = \{q \mid p \in P\} \cup \{q' \mid p \in P\}$,
- $p'_0 = q_0$,
- $I_{P'} = I_P$ und $O_{P'} = O_P$,
- $\longrightarrow_{P'} = \dashrightarrow_{P'} = \Big\{(q_0, \alpha, q_j) \mid p_0 \overset{\alpha}{\dashrightarrow} p_j\Big\}$
  $\cup \Big\{(q_0, \alpha, q'_j) \mid p_0 \overset{\alpha}{\dashrightarrow} p_j\Big\}$
  $\cup \Big\{(q_j, \alpha, q_k) \mid p_j \overset{\alpha}{\longrightarrow} p_k\Big\}$
  $\cup \Big\{(q'_j, \alpha, q'_k) \mid p_j \overset{\alpha}{\longrightarrow} p_k\Big\}$
  $\cup \Big\{(q'_j, \alpha, q_k) \mid j \neq 0, p_j \overset{\alpha}{\dashrightarrow} p_k, p_j \overset{\alpha}{\not\longrightarrow} p_k\Big\}$
  $\cup \Big\{(q'_j, \alpha, q'_k) \mid j \neq 0, p_j \overset{\alpha}{\dashrightarrow} p_k, p_j \overset{\alpha}{\not\longrightarrow} p_k\Big\}$,
- $E_{P'} = \emptyset$.

Als as-Verfeinerungs-Relation zwischen P und P' wird die Relation $\mathcal{R} = \{(q_j, p_j) \mid p_j \in P\} \cup \{(q'_j, p_j) \mid p_j \in P\}$ verwendet. Es werden in P' für die ungestrichenen Zustände q nur die must-Transitionen und für die gestrichenen Zustände q' werden die must- und may-Transitionen implementiert. Die must-Transitionen werden nur zu den Zuständen der „gleichen Sorte“ umgesetzt, wohingegen die may-Transitionen, zu denen es keine entsprechende must-Transition in P gibt, von den Zuständen q' zu dem entsprechenden ungestrichenen und gestrichenen Zustand implementiert werden. Da die Menge der Fehler-Zustände leer ist, gilt 1.3.1 für $\mathcal{R}$. Die must-Transitionen werden für die ungestrichenen und gestrichenen Zustände umgesetzt, dies erfüllt zusammen mit $\mathcal{R}$ die Definition 1.3.2. Ebenso wird der dritte Punkt der Definition 1.3 erfüllt, da sowohl die gestrichenen wie auch die ungestrichenen Zustände mit den entsprechenden Zuständen aus P in der Relation $\mathcal{R}$ stehen. $\mathcal{R}$ ist also eine starke alternierende Simulations-Relation zwischen P' und P. Falls ein Zustand p_j in P still ist, ist es auch der entsprechenden Zustand q_j in P', da für q_j alle ausgehenden must-Transitionen von p_j implementiert wurden, aber keine einzige may-Transition, die keine der must-Transitionen entspricht. Wenn also für p_j keine Outputs und kein τ möglich waren via must-Transitionen, dann ist es dies auch für q_j nicht. q_j ist in P' mit den selben Traces erreichbar wie p_j in P, da jeder ungestrichene und gestrichene Zustand in P' die selben eingehenden Transitionen

hat wie der entsprechende Zustand in P. Falls der Trace zu p_j may-Transitionen ohne entsprechende must-Transitionen enthält, kann der Trace in P' ausgeführt werden, in dem von q_0 aus der Trace über die gestrichenen Zustände genommen wird bis zur letzten Transition, die zu einem ungestrichenen Zuständen führt in dem auszuführenden Wort. Ab da hat der Trace in P nur must-Transitionen genommen und kann somit in den ungestrichenen Zuständen in P' nachgefolgt werden. Falls der Trace in P insgesamt nur aus must-Transitionen bestanden hat, ist direkt von q_0 aus der Weg über ungestrichene Zustände zu q_j möglich. Es gilt also $StQT(P)\backslash\{\varepsilon\} = StQT(P')\backslash\{\varepsilon\}$. Falls ε zu einem stillen Zustand $p \neq p_0$ in P geführt hat, gilt sogar $StQT(P) = StQT(P')$, da ein Trace aus internen Aktionen in P und P' zu dem entsprechenden stillen Zustand p bzw. q führt. □

Man hätte anstatt des hier verwendeten P' mit zwei Arten von Zuständen auch den Ansatz des ausrollens der betrachteten Traces analog zu Proposition 2.4.3 anwenden können. Es wäre auch möglich die Inklusion der Input-kritischen Traces aus Proposition 2.4.3 mit dem P' aus dem Beweis der Proposition 3.4 zu begründen.
Für ET und EL gelten die Definitionen aus dem letzten Kapitel. Es wir nur für Stille eine neue Semantik definiert.

Definition 3.5 (*Stillstands-Semantik*). *Sei P ein MEIO.*

- *Die Menge der* fehler-gefluteten Stille-Traces *von P ist $QET(P) := StQT(P) \cup ET(P)$.*

Für zwei MEIOs P_1, P_2 mit der gleichen Signatur wird $P_1 \sqsubseteq_{Qui} P_2$ geschrieben, wenn $P_1 \sqsubseteq_E P_2$ und $QET_1 \subseteq QET_2$ gilt.

Proposition 3.6 (*Stillstands-Semanik und Implementierungen*). *Für die Menge der fehler-gefluteten Stille-Traces von P gilt $QET(P) = \bigcup_{P' \in \text{as-impl}(P)} QET(P')$.*

Beweis.
„$\subseteq$“:

$$
\begin{aligned}
QET(P) &\overset{3.5}{=} StQT(P) \cup ET(P) \\
&\overset{3.4}{\subseteq} \left(\bigcup_{P' \in \text{as-impl}(P)} StET(P') \right) \cup ET(P) \\
&\overset{2.7}{=} \left(\bigcup_{P' \in \text{as-impl}(P)} StET(P') \right) \cup \left(\bigcup_{P' \in \text{as-impl}(P)} ET(P') \right) \\
&= \bigcup_{P' \in \text{as-impl}(P)} StQT(P') \cup ET(P') \\
&\overset{3.5}{=} \bigcup_{P' \in \text{as-impl}(P)} QET(P').
\end{aligned}
$$

„$\supseteq$“:
Es wird hier für ein $w \in QET(P')$ einer beliebigen as-Implementierung P' von P gezeigt, dass das Wort w auch in $QET(P)$ enthalten ist. Es kann danach unterschieden werden, ob w aus $StQT(P')\backslash ET(P')$ stammt oder aus $ET(P')$. Falls $w \in ET(P')$ gilt, folgt mit Proposition 2.7 bereits, dass $w \in ET(P) \subseteq QET(P)$ gilt. Somit wird für den Rest des Beweises davon ausgegangen, dass $w \in StQT(P')\backslash ET(P')$ ist. w führt in P' also nur zu einem stillen Zustand und hat nichts mit Fehler-Zuständen in P' zu tun. Der Trace, der durch w in P' beschrieben wird, hat die folgende Form: $\exists w' \in \Sigma_\tau, \exists \alpha_1, \alpha_2, \ldots, \alpha_n, \exists p'_1, p'_2, \ldots p'_n : \hat{w'} = w \wedge w' = \alpha_1\alpha_2 \ldots \alpha_n \wedge p'_0 \stackrel{\alpha_1}{\dashrightarrow}_{P'} p'_1 \stackrel{\alpha_2}{\dashrightarrow}_{P'} \ldots p'_{n-1} \stackrel{\alpha_n}{\dashrightarrow}_{P'} p'_n \in Qui_{P'}$. Da es eine as-Verfeinerungs-Relation $\mathcal{R}$ geben muss, die beweist, dass P' P as-verfeinert, muss es ein Präfix von w geben, dass auch in P ausführbar ist. Falls w nicht vollständig ausführbar ist in P, muss auf dem Weg, auf dem das Präfix von w ausgeführt wird ein Zustand p_j mit $0 \leq j \leq n$ liegen, der ein Fehler-Zustand ist. Es gilt dann $w \in ET(P) \subseteq QET(P)$. Falls jedoch w in P ausführbar ist ohne einen Fehler-Zustand zu erreichen, gibt es einen analogen Trace zu dem in P', der Form: $p_0 \stackrel{\alpha_1}{\dashrightarrow}_P p_1 \stackrel{\alpha_2}{\dashrightarrow}_P \ldots p_{n-1} \stackrel{\alpha_n}{\dashrightarrow}_P p_n$, wobei $p'_j \mathcal{R} p_j$ für alle j aus $\{0, 1, \ldots, n\}$ gilt. Es wird also durch w ein Zustand p_n erreicht, der mit dem Zustand p'_n in der starken as-Verfeinerungs-Relation $\mathcal{R}$ stehen. p'_n ist still, nach Voraussetzung. Es gilt also für alle $\omega \in O \cup \{\tau\}$ $p'_n \stackrel{\omega}{\not\rightarrow}$. Da $(p'_n, p_n) \in \mathcal{R}$ gilt und beide Zustände keine Fehler-Zustände sind, muss auch $p_n \stackrel{\omega}{\not\rightarrow}$ für alle $\omega \in O \cup \{\tau\}$ gelten, da sonst 1.3.2 verletzt würde. Es gilt also in diesem Fall $w \in StQT(P) \subseteq QET(P)$. □

Wie im letzten Kapitel kann aus der vorangegangen Proposition über die Gleichheit der betrachteten Trace Mengen in der Relation $\sqsubseteq_{Qui}$ auch eine Aussage über die lokale Erreichbarkeit „fehlerhafter Zustände“ in einer Spezifikation und den zugehörigen Implementierungen getroffen werden.

Korollar 3.7 (*lokale Stillstands Erreichbarkeit*).

(i) Falls in einem MEIO P ein Fehler lokal erreichbar ist, dann existiert auch eine as-Implementierung, in der ein Fehler lokal erreichbar ist.

(ii) Falls ein MEIO P einen lokal erreichbaren stillen Zustand besitzt, dann existiert auch eine as-Implementierung, in der ein stiller Zustand lokal erreichbar ist.

(iii) Falls es eine as-Implementierung von P gibt, die einen Fehler oder Stille lokal erreicht, dann ist auch ein Fehler oder Stille in P lokal erreichbar.

Beweis.

(i) Dieser Punkt folgt direkt aus Korollar 2.8 (i).

(ii) Da ein stiller Zustand in P lokal erreichbar ist, gilt $w \in StQT_P$ für $w \in O$. Es muss wegen Proposition 3.4 mindestens ein $P' \in$ as-impl(P) geben, für dass $w \in StQT_{P'}$ gilt. Da w nur aus lokalen Aktion bestehen kann, ist auch in P' ein stiller Zustand lokal erreichbar.

(iii) Sei P' die as-Implementierung von P, in der ein Fehler- oder stiller Zustand lokal erreichbar ist. Es gilt dann $w \in QET_{P'}$ für $w \in O$. Mit Proposition 3.6 folgt draus $w \in QET_P$. Es muss also auch in P ein Fehler- oder stiller Zustand lokal erreichbar sein.

□

Für spätere Beweise werden noch Zusammenhänge zwischen Stille-Zuständen in den einzelnen Komponenten und in einer Parallelkomposition dieses Komponenten benötigt.

Lemma 3.8 (*Stillstands-Zustände unter Parallelkomposition*).

1. *Ein Zustand (p_1, p_2) aus der Parallelkomposition P_{12} ist still, wenn es auch die Zustände p_1 und p_2 in P_1 bzw. P_2 sind.*
2. *Wenn der Zustand (p_1, p_2) still ist und nicht in E_{12} enthalten ist, dann sind auch die auf die Teilsysteme projizierten Zustände p_1 und p_2 still.*

Beweis.

1. Da $p_1 \in Qui_1$ und $p_2 \in Qui_2$ gilt, haben diese beiden Zustände jeweils höchstens die Möglichkeit für Input-Transitionen oder Output- und τ-may-Transitionen, jedoch keine Möglichkeit für Outputs oder τs als must-Transitionen.
   Angenommen der Zustand, der durch die Parallelkomposition aus den Zuständen p_1 und p_2 entsteht, ist nicht still, d.h. er hat eine ausgehende must-Transition für einen Output oder ein τ.

   - Fall 1 $\left((p_1, p_2) \xrightarrow{\tau}_{12}\right)$: Ein τ ist eine interne Aktion und kann in der Parallelkomposition nicht durch das Verbergen von Aktionen bei der Synchronisation entstehen. Ein τ in der Parallelkomposition ist also auch nur möglich, wenn dies bereits als must-Transition in einer Komponente möglich war für einen der Zustände, aus denen (p_1, p_2) zusammensetzt ist. Jedoch verbietet die Voraussetzung, dass p_1 oder p_2 eine ausgehende τ-must-Transition haben, deshalb kann auch (p_1, p_2) keine solche Transition besitzen.

   - Fall 2 $\left((p_1, p_2) \xrightarrow{a}_{12} \text{ mit } a \in O_{12} \backslash \text{Synch}(P_1, P_2)\right)$: Da es sich bei a um einen Output handelt, der nicht in $\text{Synch}(P_1, P_2)$ enthalten ist, kann dieser nicht aus der Synchronisation von zwei Aktionen entstanden sein, sondern muss bereits für P_1 oder P_2 als must-Transition ausführbar gewesen sein. Es gilt also oBdA $p_1 \xrightarrow{a}_1$ mit $a \in O_1$. Dies ist jedoch aufgrund der Voraussetzung nicht möglich. Somit kann die Parallelkomposition diese Transition für (p_1, p_2) ebenfalls nicht als must-Transition enthalten sein.

   - Fall 3 $\left((p_1, p_2) \xrightarrow{a}_{12} \text{ mit } a \in O_{12} \cap \text{Synch}(P_1, P_2)\right)$: Der Output a ist in diesem Fall durch Synchronisation von einem Output mit einem Input entstanden. OBdA gilt $a \in O_1 \cap I_2$. Für die einzelnen Systeme muss also gelten, dass

$p_1 \xrightarrow{a}_1$ und $p_2 \xrightarrow{a}_2$. Die Transition für das System P_1 ist jedoch in der Voraussetzung ausgeschlossen worden. Somit ist es nicht möglich, dass P_{12} diese in diesem Fall angenommene must-Transition für den Zustand (p_1, p_2) ausführen kann.

Da alle diese Fälle zu einem Widerspruch mit der Voraussetzung führen folgt, dass bereits die Annahme, dass der Zustand (p_1, p_2) nicht still ist, falsch war. Es gilt also, dass aus $p_j \in Qui_j$ für $j \in \{1, 2\}$ $(p_1, p_2) \in Qui_{12}$ folgt.

2. Es gilt $(p_1, p_2) \in Qui_{12} \backslash E_{12}$, somit hat dieser Zustand allenfalls die Möglichkeit für must-Transitionen, die mit Inputs beschriftet sind.
   Angenommen $p_1 \notin Qui_1$, dann ist für p_1 entweder eine τ-must-Transition oder eine Output-must-Transition möglich.

   - Fall 1 $\left(p_1 \xrightarrow{\tau}_1\right)$: Da die Transition für P_1 möglich ist, hat auch P_{12} die Möglichkeit für eine τ-must-Transition. Dies ist jedoch durch die Voraussetzung verboten und somit kann dieser Fall nicht eintreten.
   - Fall 2 $\left(p_1 \xrightarrow{a}_1 \text{ mit } a \in O_1 \backslash \text{Synch}(P_1, P_2)\right)$: Da es sich bei a um einen must-Output handelt, der nicht zu synchronisieren ist, wird dieser einfach in die Parallelkomposition übernommen. Es müsste also $(p_1, p_2) \xrightarrow{a}_{12}$ mit $a \in O_{12}$ gelten, was jedoch verboten ist. Somit kann die Transition für P_1 in diesem Fall nicht möglich sein.
   - Fall 3 $\left(p_1 \xrightarrow{a}_1 \text{ mit } a \in O_1 \cap \text{Synch}(P_1, P_2) \text{ und } p_2 \xrightarrow{a}_2\right)$: In diesem Fall ist die Synchronisation des Outputs a von P_1 mit dem Input a von P_2 möglich, so dass in der Parallelkomposition der Output a als must-Transition für (p_1, p_2) entsteht. Diese must-Transition ist jedoch für P_{12} nach Voraussetzung nicht erlaubt. Es folgt also auch, dass dieser Fall nicht eintreten kann.
   - Fall 4 $\left(p_1 \xrightarrow{a}_1 \text{ mit } a \in O_1 \cap \text{Synch}(P_1, P_2) \text{ und } p_2 \not\xrightarrow{a}_2\right)$: Da P_2 die a Transition nicht als must-Transition enthält, handelt es sich hier um einen neuen Fehler. Das a kann für P_2 kein Output sein, da sonst P_1 und P_2 nicht komponierbar wäre. Der neue Fehler kann dadurch entstehen, dass die Synchronisation des Outputs a von P_1 mit dem Input a von P_2 an dieser Stelle nicht möglich ist, oder da der Input a für p_2 nur als may-Transition vorliegt und somit die Gefahr besteht, dass dieser in einer Implementierung nicht vorhanden ist. Im zweiten Fall synchronisieren die beiden Transitionen zu einer a Output-may-Transition, die in P_{12} zulässig wäre. Jedoch wird der Zustand (p_1, p_2) in beiden Fällen in die Menge E_{12} der Parallelkomposition eingefügt (Definition 1.2). Dies wurde in der Voraussetzung für den Zustand ausgeschlossen und dieser Fall ist somit nicht möglich.

   Alle aufgeführten Fälle führen zu einem Widerspruch mit der Voraussetzung, somit folgt, dass die Annahme bereits falsch war und $p_1 \in Qui_1$ gelten muss. Analog kann für p_2 argumentiert werden, so dass dann auch $p_2 \in Qui_2$ folgt.

□

In dem folgenden Satz sind die Punkte 1. und 3. nur zur Vollständigkeit aufgeführt. Sie entsprechen Punkt 1. und 2. aus Satz 2.9.

Satz 3.9 (*Kommunikationsfehler- und Stillstands-Semantik für Parallelkompositionen*). *Für zwei komponierbare MEIOs P_1, P_2 und ihre Komposition P_{12} gilt:*

1. $ET_{12} = \text{cont}(\text{prune}((ET_1 \| EL_2) \cup (EL_1 \| ET_2)))$,
2. $QET_{12} = (QET_1 \| QET_2) \cup ET_{12}$,
3. $EL_{12} = (EL_1 \| EL_2) \cup ET_{12}$.

Beweis. Es wird nur der 2. Punkt beweisen.
„$\subseteq$“:
Hier muss unterschieden werden, ob ein $w \in StQT_{12} \backslash ET_{12}$ oder ein $w \in ET_{12}$ betrachtet wird. Im zweiten Fall ist das w offensichtlich in der rechten Seite enthalten. Somit wird im Folgenden ein $w \in StQT_{12} \backslash ET_{12}$ betrachtet und es wird versucht dessen Zugehörigkeit zur rechten Menge zu zeigen. Aufgrund von Definition 3.3 weiß man, dass $(p_{01}, p_{02}) \overset{w}{\Rightarrow}_{12} (p_1, p_2)$ gilt mit $(p_1, p_2) \in Qui_{12} \backslash E_{12}$. Durch Projektion erhält man $p_{01} \overset{w_1}{\Rightarrow}_1 p_1$ und $p_{02} \overset{w_2}{\Rightarrow}_2 p_2$ mit $w \in w_1 \| w_2$. Aus $(p_1, p_2) \in Qui_{12} \backslash E_{12}$ kann mit dem zweiten Punkt von Lemma 3.8 gefolgert werden, dass $q_1 \in Qui_1$ und $q_2 \in Qui_2$ gilt. Somit gilt $w_1 \in StQT_1 \subseteq QET_1$ und $w_2 \in StQT_2 \subseteq QET_2$. Daraus folgt $w \in QET_1 \| QET_2$ und somit ist w in der rechten Seite der Gleichung enthalten.

„$\supseteq$“:
Es muss wieder danach unterschieden werden aus welcher Menge das betrachtete Element stammt. Falls $w \in ET_{12}$ gilt, so kann die Zugehörigkeit zur linken Seite direkt gefolgert werden. Somit wird für den weiteren Beweis dieser Inklusionsrichtung ein Element $w \in QET_1 \| QET_2$ betrachtet und gezeigt, dass es in der linken Menge enthalten ist. Da $QET_i = StQT_i \cup ET_i$ gilt, existieren für w_1 und w_2 mit $w \in w_1 \| w_2$ unterschiedliche Möglichkeiten:

- Fall 1 ($w_1 \in ET_1 \vee w_2 \in ET_2$): OBdA gilt $w_1 \in ET_1$. Nun kann $w_2 \in StQT_2 \subseteq L_2$ oder $w_2 \in ET_2$ gelten und somit ist auf jeden Fall w_2 in EL_2 enthalten. Daraus kann dann mit dem ersten Punkt von Satz 2.9 gefolgert werden, dass $w \in ET_{12}$ gilt und damit ist w in der linken Seite der Gleichung enthalten.
- Fall 2 ($w_1 \in StQT_1 \backslash ET_1 \wedge w_2 \in StQT_2 \backslash ET_2$): Es gilt in diesem Fall $p_{01} \overset{w_1}{\Rightarrow}_1 p_1 \in Qui_1$ und $p_{02} \overset{w_2}{\Rightarrow}_2 p_2 \in Qui_2$. Da p_1 und p_2 in der jeweiligen Stillstands-Menge enthalten sind, ist auch der Zustand, der aus ihnen zusammengesetzt ist, in der Parallelkomposition still, wie bereits im ersten Punkt von Lemma 3.8 gezeigt. Es gilt also für die Komposition $(p_{01}, p_{02}) \overset{w}{\Rightarrow}_{12} (p_1, p_2) \in Qui_{12}$ und dadurch ist w in der linken Seite der Gleichung enthalten, da $w \in StQT_{12} \subseteq QET_{12}$ gilt.

□

Aus diesem Satz kann direkt gefolgert werden, dass $\sqsubseteq_{Qui}$ eine Präkongruenz ist. Den Beweis dazu liefert das folgende Korollar.

Korollar 3.10 (*Stillstands-Präkongruenz*). *Die Relation $\sqsubseteq_{Qui}$ ist eine Präkongruenz bezüglich $\cdot\|\cdot$.*

Beweis. Es muss gezeigt werden: Wenn $P_1 \sqsubseteq_{Qui} P_2$ gilt, so auch $P_{31} \sqsubseteq_{Qui} P_{32}$ für jedes komponierbare System P_3. D.h. es ist zu zeigen, dass aus $P_1 \sqsubseteq_E P_2$ und $QET_1 \subseteq QET_2$ sowohl $P_{31} \sqsubseteq_E P_{32}$ als auch $QET_{31} \subseteq QET_{32}$ folgt. Dies ergibt sich, wie im Beweis zu Korollar 2.10, aus der Monotonie von $\cdot\|\cdot$ auf Sprachen wie folgt:

- $P_{31} \overset{\substack{\text{Korollar 2.10}\\ \text{und}\\ P_1 \sqsubseteq_E P_2}}{\sqsubseteq_E} P_{32}$,
- $QET_{31} \overset{3.9.2}{=} (QET_3 \| QET_1) \cup ET_{31}$
  $\overset{\substack{ET_{31} \subseteq ET_{32}\\ \text{und}\\ QET_1 \subseteq QET_2}}{\subseteq} (QET_3 \| QET_2) \cup ET_{32}$
  $\overset{3.9.2}{=} QET_{32}$. □

Im nächsten Lemma soll eine Verfeinerung bezüglich guter Kommunikation mit Partnern betrachtet werden. Die gute Kommunikation stützt sich dabei auf die Definition von Tests und der daraus resultierenden Verfeinerung in 3.2.

Lemma 3.11 (*Testing-Verfeinerung mit Stillstand*). *Gegeben sind zwei MEIOs P_1 und P_2 mit der gleichen Signatur. Wenn für alle Tests T, die Partner von P_1 bzw. P_2 sind, $P_2\, sat_{\text{as}}^{Qui}\, T \Rightarrow P_1\, sat_{\text{as}}^{Qui}\, T$ gilt, dann folgt daraus die Gültigkeit von $P_1 \sqsubseteq_{Qui} P_2$.*

Beweis. Da P_1 und P_2 die gleiche Signatur 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 zu zeigen, dass die Relation $P_1 \sqsubseteq_{Qui} P_2$ gilt, müssen die folgenden Punkte nachgewiesen werden:

- $P_1 \sqsubseteq_E P_2$,
- $QET_1 \subseteq QET_2$.

In Lemma 2.15 wurde bereits etwas Ähnliches gezeigt, jedoch wurde dort als Voraussetzung $P_2\, sat_{\text{as}}^{E}\, T \Rightarrow P_1\, sat_{\text{as}}^{E}\, T$ für alle Test Partner T verwendet und hier dieselbe Aussage mit der Test Erfüllung für Stillstand. Die hier verwendeten Tests sagen nichts darüber aus, welche Art von fehlerhaftem Zustand enthalten ist. Die Aussage des Lemmas 2.15 kann hier also nicht verwendet werden. Aus der lokalen Erreichbarkeit eines Fehler-Zustandes in der Parallelkomposition einer as-Implementierung von P_1 mit T lässt sich nur schließen, dass P_2 den Test T ebenfalls nicht as-erfüllt. Dies kann aber aufgrund einer as-Implementierung von P_2 sein, die in Parallelkomposition mit T einen Fehler- oder stillen Zustand lokal erreicht. Analoges gilt auch für die lokale Erreichbarkeit eines

stillen Zustandes in der Komposition einer as-Implementierung von P_1 mit einem Test T.
Es muss also für den ersten Punkt noch folgendes nachgewiesen werden:

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

Es wird nun damit begonnen, den ersten Unterpunkt des ersten Beweispunktes zu zeigen, d.h. es wird unter der Voraussetzung $P_2\, sat_{\text{as}}^{Qui}\, T \Rightarrow P_1\, sat_{\text{as}}^{Qui}\, T$ gezeigt, dass $ET_1 \subseteq ET_2$ gilt. Da beide ET-Mengen unter cont abgeschlossen sind, reicht es ein präfix-minimales Element $w \in ET_1$ zu betrachten und zu zeigen, dass dieses w oder eines seiner Präfixe in ET_2 enthalten ist. w muss, wegen Proposition 2.7, in einer as-Implementierung P_1' von P_1 ebenfalls ein präfix-minimales Element in $ET_{P_1'}$ sein.

- Fall 1 ($w = \varepsilon$): Es handelt sich um einen lokal erreichbaren Fehler-Zustand in P_1'. Für T wird ein Transitionssystem verwendet, das nur aus dem Startzustand, einer must-Schleife für alle Inputs $x \in I_T$ und einer must-Schlinge für τ besteht. Somit kann P_1' die im Prinzip gleichen Fehler-Zustände lokal erreichen wie $P_1' \| T$. Es gibt also einen lokal erreichbaren Zustand von $P_1' \| T$, der in $E_{P_1' \| T}$ enthalten ist. Somit erfüllt P_1 nicht alle Tests T und es muss somit auch mindestens eine as-Implementierung P_2' von P_2 geben, die den Test T ebenfalls nicht erfüllt. Da eine Implementierung den Test T erfüllt, wenn die Parallelkomposition der Implementierung mit dem Test fehler- und stillstand-frei ist, kann die nicht Erfüllung eines Testes sowohl an einem Fehler- wie auch einem stillen Zustand liegen. Bei dem lokal erreichbaren fehlerhaften Zustand kann es sich nur um einen Fehler handeln, da es in der Komposition mit T keine Stille geben kann. Da T keinen Fehler-Zustand und auch keine fehlenden Input-Möglichkeiten enthält, kann der Fehler nur von P_2' geerbt sein. Somit muss in P_2' ein Fehler-Zustand lokal erreichbar sein. Es gilt also $\varepsilon \in PrET_{P_2'} \subseteq ET_{P_2'}$ und mit Proposition 2.7 auch $\varepsilon \in ET_2$.
- Fall 2 ($w = x_1 \dots x_n x_{n+1} \in \Sigma^+$ mit $n \geq 0$ und $x_{n+1} \in I$): Es wird der folgende Partner T betrachtet (siehe auch Abbildung 3.1):
  - $T = \{p_0, p_1, \dots, p_{n+1}\}$,
  - $p_{0T} = p_0$,
  - $\dashrightarrow_T = \Longrightarrow_T = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j \leq n\}$
    $\cup \{(p_j, x, p_{n+1}) \mid x \in I_T \backslash \{x_{j+1}\}, 0 \leq j \leq n\}$
    $\cup \{(p_{n+1}, x, p_{n+1}) \mid x \in I_T\}$
    $\cup \{(p_j, \tau, p_j) \mid 0 \leq j \leq n+1\}$,
  - $E_T = \emptyset$.

  Die Menge der stillen Zustände des hier betrachteten Ts ist leer. Da im Vergleich zum Transitionssystem in Abbildung 2.1 nur die τ-Schlingen ergänzt wurden und die Umbenennung der Mengen, ändert sich nichts an den Fällen 2a) und 2b) aus

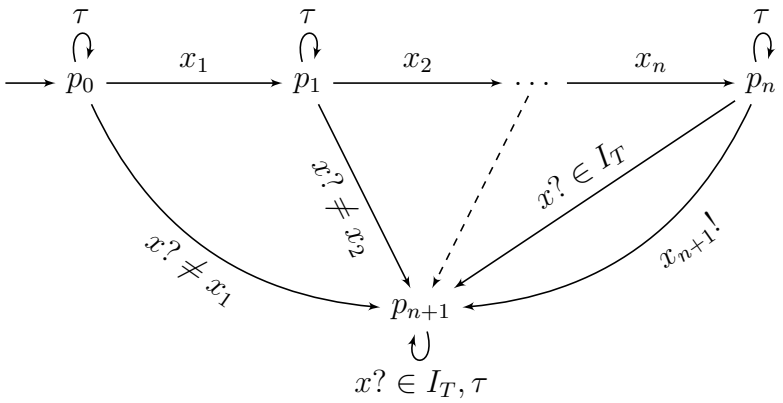

Abbildung 3.1: $x? \neq x_j$ steht für alle $x \in I_T \backslash \{x_j\}$

dem Beweis der selben Inklusion von Lemma 2.15. Die Begründungen, wieso in den beiden Fällen $\varepsilon \in PrET(P_1' \| T)$ gilt, bleibt also analog zum Beweis des ersten Punktes des Lemmas aus dem vorangegangenen Kapitel. Durch die must-τ-Schlingen wurde, genau wie im letzten Fall nur erreicht, das in einer Parallelkomposition mit T keine stillen Zustände möglich sind. Es kann also auch hier aus der lokalen Erreichbarkeit eines Fehler in $P_1' \| T$ auf die lokale Erreichbarkeit eines Fehler-Zustandes in $P_2' \| T$ für eine as-Implementierung P_2' von P_2 geschlossen werden. Die weitere Argumentation verläuft analog zu Fall 2, derselben Inklusion im Beweis von Lemma 2.15. Da τs nur interne Aktionen einer einzelnen Komponente sind, verändert sich auch nichts an den Traces über die argumentiert wird. Es können zwar möglicherweise τ-Transitionen ausgeführt werden, diese können jedoch weder zu einem Fehler führen noch beeinflussen, dass ein anderer Trace nicht ausgeführt werden kann.

Nun wird mit dem zweiten Unterpunkt des ersten Beweispunktes begonnen. Genau wie im Beweis zu 2.15 ist hier jedoch aufgrund des bereits geführten Beweisteils nur noch $L_1 \backslash ET_1 \subseteq EL_2$ zu zeigen. Es wird also für ein beliebig gewähltes $w \in L_1 \backslash ET_1$ gezeigt, dass dieses auch in EL_2 enthalten ist. Aufgrund der Propositionen 1.8 und 2.7 gibt es auch eine as-Implementierung P_1' von P_1 für die $w \in L_{P_1'} \backslash ET_{P_1'}$ gilt.

- Fall 1 ($w = \varepsilon$): Ebenso wie in 2.15 gilt auch hier, dass ε immer in EL_2 enthalten ist.

- Fall 2 ($w = x_1 \ldots x_n$ mit $n \geq 1$): Die Konstruktion des Partners T weicht wie im letzten Beweisteil nur durch die τ-must-Schleifen an den Zuständen des Transitionssystems vom Beweis des zweiten Punktes aus Lemma 2.15 ab. Somit ist der Partner T dann wie folgt definiert (siehe dazu auch Abbildung 3.2):

  – $T = \{p_0, p_1, \ldots, p_n, p\}$,

  – $p_{0T} = p_0$,

– $\dashrightarrow_T = \longrightarrow_T = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j < n\}$
$\cup \{(p_j, x, p) \mid x \in I_T \backslash \{x_{j+1}\}, 0 \leq j < n\}$
$\cup \{(p_j, \tau, p_j) \mid 0 \leq j \leq n\}$
$\cup \{(p, \alpha, p) \mid \alpha \in I_T \cup \{\tau\}\}$,

– $E_T = \{p_n\}$.

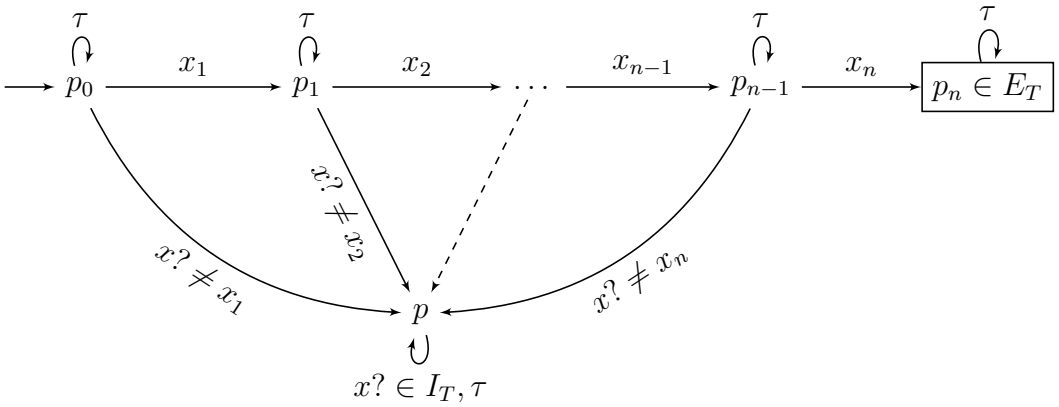

Abbildung 3.2: $x? \neq x_j$ steht für alle $x \in I_T \backslash \{x_j\}$, p_n ist der einzige Fehler-Zustand

> Da durch die τ-must-Schlingen an den Zuständen wie oben vermieden wird, dass es in einer Komposition mit T und auch in T selbst stille Zustände gibt, verläuft der Rest des Beweises dieses Punktes analog zum Beweis der selben Inklusions von Lemma 2.15. Und somit gilt für alle Fälle (2a) bis 2d)), dass w in EL_2 enthalten ist.

So bleibt nur noch der letzt Beweispunkt zu zeigen, d.h. die Inklusion $QET_1 \subseteq QET_2$. Diese kann jedoch, analog zum Beweis der Inklusion der Fehler-gefluteten Sprache, noch weiter eingeschränkt werden. Da bereits bekannt ist, dass $ET_1 \subseteq ET_2$ gilt, muss nur noch $StQT_1 \backslash ET_1 \subseteq QET_2$ gezeigt werden.
Es wird ein $w \in StQT_1 \backslash ET_1$ gewählt und gezeigt, dass dieses auch in QET_2 enthalten ist. Mit den Propositionen 2.7 und 3.4 kann gefolgert werden, dass es auch eine as-Implementierung P_1' von P_1 gibt, für die $w \in StQT_{P_1'} \backslash ET_{P_1'}$ gilt.
Durch die Wahl des ws wird vom Startzustand von P_1' durch das Wort w ein stiller Zustand erreichbar. Dies hat nur Auswirkungen auf die Parallelkomposition $P_1' \| T$, wenn in T ebenfalls ein stiller Zustand durch w erreichbar ist.
Das betrachtete w hat also die Form $w = x_1 \ldots x_n \in \Sigma^*$ mit $n \geq 0$. Es wird der folgende Partner T betrachtet (siehe auch Abbildung 3.3):

- $T = \{p_0, p_1, \ldots, p_n, p\}$,
- $p_{0T} = p_0$,

- $\dashrightarrow_T = \longrightarrow_T = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j < n\}$
  $\cup \{(p_j, x, p) \mid x \in I_T \backslash \{x_{j+1}\}, 0 \leq j < n\}$
  $\cup \{(p_j, \tau, p_j) \mid 0 \leq j < n\}$
  $\cup \{(p_n, x, p) \mid x \in I_T\}$
  $\cup \{(p, \alpha, p) \mid \alpha \in I_T \cup \{\tau\}\},$
- $E_T = \emptyset$.

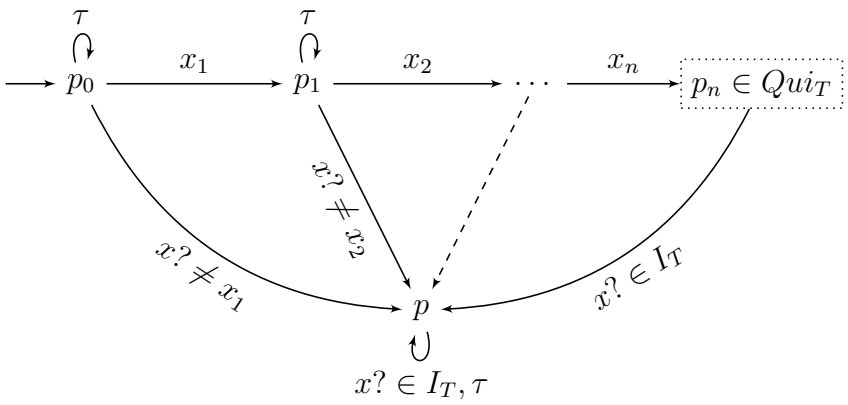

Abbildung 3.3: $x? \neq x_j$ steht für alle $x \in I_T \backslash \{x_j\}$, p_n ist der einzige stille Zustand

Falls für das betrachtete $w = \varepsilon$ gilt, reduziert sich der Partner T auf den Zustand $p_n = p_0$ und den Zustand p. Es ist also in diesem Fall der Startzustand gleich dem stillen Zustand.
Allgemein ist der Zustand p_n aus T der einzig stille Zustand in T. Es gilt wegen des ersten Punktes von Lemma 3.8, dass auch in der Parallelkomposition $P_1' \| T$ ein stiller Zustand mit w erreicht wird. Bei allen in w befindlichen Aktionen handelt es sich um synchronisierte Aktionen und es gilt $I_T \cap I = \emptyset$. Daraus folgt $w \in O_{P_1' \| T}$ und $w \in StQT(P_1' \| T)$. Es kann also in der Parallelkomposition durch w ein stiller Zustand lokal erreicht werden. Da $w \notin ET_{P_1'}$ gilt, kann auf dem Weg, der mit w im Transitionssystem P_1' zurückgelegt wird, kein Fehler lokal erreicht werden. Es kann also weder von P_1' noch von T ein Fehler auf diesem Weg geerbt werden und durch den Aufbau von T kann auch kein neuer Fehler in der Parallelkomposition der beiden Systeme entstehen. Da ein stiller Zustand in $P_1' \| T$ lokal erreichbar ist, muss auch in $P_2' \| T$ für eine as-Implementierung P_2' von P_2 ein fehlerhafter Zustand lokal erreichbar sein. Es kann zunächst keine Aussage getroffen werden, ob das w in $P_2' \| T$ ausführbar ist und ob es sich bei dem fehlerhaften Zustand um Stille oder einen Fehler handelt.

- Fall a) ($\varepsilon \in ET(P_2' \| T)$): Es handelt sich bei dem lokal erreichbaren fehlerhaften Zustand um einen Fehler. Es ist somit nicht relevant, ob w ausführbar ist. Der Fehler-Zustand kann sowohl von P_2' geerbt sein, wie auch durch fehlende Input-must-Transitionen als neuer Fehler in der Parallelkomposition entstanden sein. Es gilt, dass in P_2' ein Präfix von w in $ET_{P_2'}$ enthalten ist, wegen des Beweises des

ersten Punktes aus Lemma 2.15 und da T nur neue Fehler auf dem Trace w zulässt. Die Menge ET ist unter cont abgeschlossen, somit gilt $w \in ET_{P_1'} \subseteq QET_{P_2'}$. Mit Proposition 3.6 folgt daraus $w \in QET_2$.

- Fall b) (stiller Zustand lokal erreichbar in $P_2' \| T$ und $\varepsilon \notin ET(P_2' \| T)$): Da in T nur durch w ein stiller Zustand erreicht werden kann, muss es sich bei dem lokal erreichbaren stillen Zustand in $P_2' \| T$ um einen handeln, der mit w erreicht werden kann. Mit Lemma 3.8 kann somit gefolgert werden, dass auch in P_2' ein stiller Zustand mit w erreichbar sein muss. Es gilt $w \in StQT_{P_2'} \subseteq QET_{P_2'} \subseteq QET_2$, wegen Proposition 3.6.

□

Satz 3.12. *Falls $P_1 \sqsubseteq_{Qui} P_2$ gilt folgt draus auch, dass P_1 P_2 Stille-verfeinert.*

Beweis. Nach Definition gilt $w \in QET(P)$ mit $w \in O(P)^*$ genau dann, wenn in P ein stiller Zustand oder ein Fehler-Zustand lokal erreichbar ist. $P_1 \sqsubseteq_{Qui} P_2$ impliziert, dass $w \in QET_2$ gilt, wenn $w \in QET_1$ gilt. Somit ist ein stiller- oder Fehler-Zustand nur dann in P_1 lokal erreichbar, wenn auch ein stiller- oder Fehler-Zustand in P_2 lokal erreichbar ist. Daraus folgt, dass es as-Implementierungen P_1' und P_2' von P_1 bzw. P_2 gibt, die analoge fehlerhafte Zustände lokal erreichen wegen 3.7. Es gilt dann auch, dass $P_j' \| T$ einen lokal erreichbaren Fehler oder lokal erreichbare Stille hat, wenn P_j' dies hat, für $j \in \{1, 2\}$ und einen Test T. Falls P_1 einen fehlerhaften Zustand lokal erreicht, dann zeigt sich auch in einer as-Implementierung von P_1 dieser durch einen Fehler-Zustand oder einen stillen Zustand. In der Parallelkomposition der as-Implementierung mit einem Test T tritt der fehlerhafte Zustand auf. Falls es sich um einen Fehler handelt, dann tritt mit allen Tests T ein Fehler in der Parallelkomposition auf. Bei Stille hingegen zeigt sich das Fehlverhalten nur, mit Tests T, die einen analogen stillen Zustand enthalten. Mit der Relation $\sqsubseteq_{Qui}$ gibt es auch in P_2 einen lokal erreichbaren fehlerhaften Zustand, wenn es in P_1 einen solchen gibt. Es tritt dann auch in einer as-Implementierung Stille oder ein Fehler auf, dies zeigt sich dann auch in der Parallelkomposition mit Tests T. Im Fall von Fehlern, zeigt sich sowohl in der Parallelkomposition einer as-Implementierung von P_1 wie auch einer as-Implementierung von P_2 mit beliebigen Tests T, der Fehler, da $\sqsubseteq_E$ gilt und somit die Argumentationen aus Satz 2.16 anwendbar sind. Stille tritt in P_2 nur ohne einen Fehler auf, wenn in P_1 auch nur Stille und kein Fehler auf dem Trace vorhanden war. Ansonsten hätte $\sqsubseteq_E$ auch einen Fehler in P_2 gefordert. In der Parallelkomposition einer as-Implementierung von P_1 mit T hat sich die Stille bereits gezeigt, also tut sie dies auch in der Parallelkomposition einer as-Implementierung von P_2 mit T. Es gilt also $\neg P_1\, sat_{\text{as}}^{Qui}\, T \Rightarrow \neg P_2\, sat_{\text{as}}^{Qui}\, T$ für alle Tests T. Daraus ergibt sich für alle Tests T die Implikation $P_2\, sat_{\text{as}}^{Qui}\, T \Rightarrow P_1\, sat_{\text{as}}^{Qui}\, T$ und somit Stille-verfeinert P_1 P_2. □

Es wurde, wie im letzten Kapitel, eine Kette an Folgerungen gezeigt, die sich zu einem Ring schließen. Dies ist in Abbildung 3.4 dargestellt.

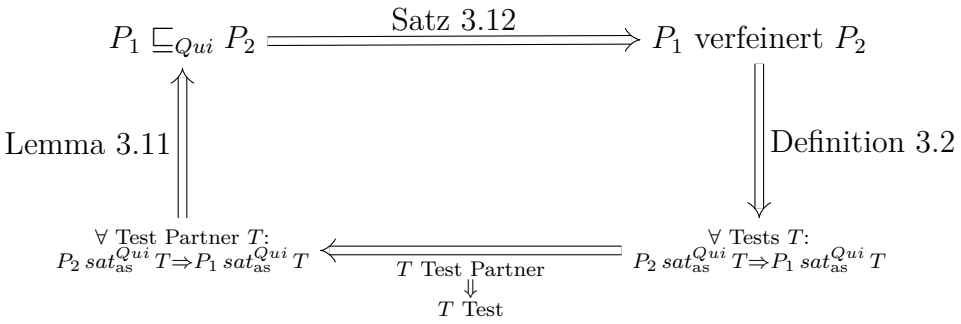

Abbildung 3.4: Folgerungskette der Testing-Verfeinerung und Stillstands-Relation

3.2 Zusammenhänge

Satz 3.13 (*Zusammenhang der Verfeinerungs-Relationenen mit der Stillstands-Relation*). *Für MEIOs P und Q gilt $P \sqsubseteq_{\text{w-as}} Q \Rightarrow P \sqsubseteq_{Qui} Q \Rightarrow P \sqsubseteq_E Q$. Die Implikationen in die andere Richtung gelten jedoch nicht.*

Beweis.
$P \sqsubseteq_{\text{w-as}} Q \Rightarrow P \sqsubseteq_{Qui} Q$:
Im Beweis des Satzes 2.17 wurde bereits bewiesen, dass eine schwache as-Verfeinerungs-Relation $\mathcal{R}$ zwischen P und Q die Eigenschaften der Fehler-Relation $\sqsubseteq_E$ erfüllt. Es fehlt für diese Implikation also nur noch der Beweis der Inklusion $QET_P \subseteq QET_Q$. Da bereits $ET_P \subseteq ET_Q$ beweisen wurde, reicht es aus zu beweisen, dass $StQT_P \backslash ET_P \subseteq QET_Q$ gilt. Für ein Wort w aus $StQT_P \backslash ET_P$ gilt: $\exists w' \in \Sigma_\tau^*, \exists p_1, p_2, \ldots, p_n, \exists \alpha_1, \alpha_2, \ldots, \alpha_n$: $\hat{w'} = w \wedge w' = \alpha_1\alpha_2 \ldots \alpha_n \wedge p_0 \overset{\alpha_1}{\dashrightarrow}_P p_1 \overset{\alpha_2}{\dashrightarrow}_P \ldots p_{n-1} \overset{\alpha_n}{\dashrightarrow}_P p_n \in Qui_P$. Aufgrund von 1.4.4 bzw. 1.4.5 gibt es einen analogen Trace in Q, falls kein q_j für $0 \leq j \leq n$ in E_Q angetroffen wird. Es gilt $w \in StET_Q \subseteq ET_Q \subseteq QET_Q$, falls ein Fehler-Zustand in Q auf einem Präfix-Trace von w auftritt. Im folgenden wird davon ausgegangen, das w in Q ohne das erreichen eines Fehler-Zustandes ausführbar ist. Es gibt also einen Trace der Form $q_0 \overset{\widehat{\alpha_1}}{\Longrightarrow}_Q q_1 \overset{\widehat{\alpha_2}}{\Longrightarrow}_Q \ldots q_{n-1} \overset{\widehat{\alpha_n}}{\Longrightarrow}_Q q_n$ in Q mit $p_j \mathcal{R} q_j$ für alle j aus $\{0, 1, \ldots n\}$. p_n ist für P ein stiller Zustand, es gilt also für alle $\omega \in (O \cup \{\tau\})$ $p_n \overset{\omega}{\not\longrightarrow}_P$. Da $\mathcal{R}$ eine schwache as-Verfeinerungs-Relation ist, muss wegen 1.4.3 auch $q_n \overset{\omega}{\not\longrightarrow}_Q$ gelten für alle $\omega \in (O \cup \{\tau\})$. q_n ist also auch ein stiller Zustand. Somit ist w in $StQT_Q \subseteq QET_Q$ enthalten.

$P \sqsubseteq_{Qui} Q \Rightarrow P \sqsubseteq_E Q$:
Diese Implikation folgt direkt aus der Definition von $\sqsubseteq_{Qui}$ in 3.5. Da $P \sqsubseteq_{Qui} Q$ dort definiert wurde als Relation, die $P \sqsubseteq_E Q$ und $QET_P \subseteq QET_Q$ erfüllt. Es gilt also $P \sqsubseteq_E Q$.

$P \sqsubseteq_{\text{w-as}} Q \not\Leftarrow P \sqsubseteq_{Qui} Q$:
Wie im Gegenbeispiel für die analoge Implikation aus der Relation $\sqsubseteq_E$ beruht der Grund

für die nicht Gültigkeit hier auch darauf, dass Simulationen strenger sind als Sprach Inklusionen. Jedoch funktioniert hier nicht das gleiche Gegenbeispiel wie im letzten Kapitel, da es zu Problemen mit den Stille-Traces führen würde. Um diese zu vermeiden, wird wieder die Technik angewendet an alle Zustände eine τ-Schleife anzufügen. Das daraus entstehende Gegenbeispiel ist in Abbildung 3.5 dargestellt. Die Menge der Inputs I der beiden MEIOs ist leer. Es gilt also $ET_P = ET_Q = QET_P = QET_Q = \emptyset$ und $\{\varepsilon\} = L(P) \subset L(Q) = \{\varepsilon, o\}$.
Angenommen es gib eine schwache as-Verfeinerungs-Relation $\mathcal{R}$ zwischen P und Q. Dann stehen die Startzustände in dieser Relation, es gilt also $p_0 \mathcal{R} q_0$. Da es keine Fehler-Zustände in P gibt, ist 1.4.1 erfüllt. Die Transition $q_0 \xrightarrow{o}_Q q_1$ fordert durch 1.4.3 ihre Verfeinerung in P. Da es jedoch keine mit o beschriftete Transition in P gibt, kann $\mathcal{R}$ keine as-Verfeinerungs-Relation zwischen P und Q sein.

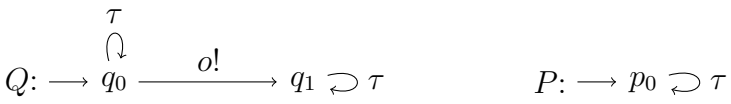

Abbildung 3.5: Gegenbeispiel zu $\sqsubseteq_{\text{w-as}} \Leftarrow \sqsubseteq_{Qui}$

$P \sqsubseteq_{Qui} Q \not\Leftarrow P \sqsubseteq_E Q$:
Die Relation $\sqsubseteq_{Qui}$ stützt sich auf die Definition der Relation $\sqsubseteq_E$. Jedoch erweitert sich die Definition noch um eine weiter Voraussetzung. Es muss also in einem entsprechenden Gegenbeispiel die Inklusion $QET_P \subseteq QET_Q$ verletzt sein. Das Gegenbeispiel ist in Abbildung 3.6 dargestellt. Es wird $I = \emptyset$ vorausgesetzt, damit keine Input-kritischen Traces auftreten. Es gilt also $ET_P = ET_Q = \emptyset$ und $L(P) = L(Q) = \{\varepsilon\}$. Es gilt also $P \sqsubseteq_E Q$.
Jedoch gilt für die strickten Stille-Traces $StQT_P = \{\varepsilon\}$ und $StQT_Q = \emptyset$. Es folgt also $QET_P \not\subseteq QET_Q$ und somit ist die Relation $\sqsubseteq_{Qui}$ zwischen P und Q auch nicht erfüllt.

Abbildung 3.6: Gegenbeispiel zu $\sqsubseteq_{Qui} \Leftarrow \sqsubseteq_E$ mit $I_P = I_Q = \emptyset$

□

3.3 Konjunktion

Die Relation $\sqsubseteq_{Qui}$ verfeinert die Relation $\sqsubseteq_E$ nur um die zusätzlich Inklusion $QET_P \subseteq QET_Q$. Es kann also die Konjunktion und die dafür notwendige Normal Form nur mit kleinen Abwandlungen aus dem Fehler-Kapitel übernommen werden.
Als erstes sollte überprüft werden, ob die Konstruktion der Normal Form etwas an den

Traces aus der Menge QET verändert. Da die Normalisierung in 2.18 die Menge der Fehler-Traces in ET unverändert lässt, reicht es aus die Traces aus $StQT(P)\backslash ET(P)$ zu betrachten. Schritt (i) lässt alle Zustände, die nicht in E_P oder $\overline{E_P}$ enthalten sind unverändert, vor allem verändert dieser Schritt noch keine Transitionen. Alle stillen Zustände sind also auch noch nach dem Schritt (i) still. Im ersten Fall des Schrittes (ii) ist die Menge $StQT(P)\backslash ET(P)$ leer und ebenso ist es die Menge $StQT(NF(P))\backslash ET(NF(P))$. Der zweite Fall von Schritt (ii) verändert nur Transitionen, zu Zuständen führen, die mit Traces aus $ET(P)$ erreichbar sind. Es bleiben also auch die durch $StQT(P)\backslash ET(P)$ erreichbaren Zustände im zweiten Schritt der Konstruktion in 2.18 still. Es gilt also auch für die Trace Menge QET $QET(P) = QET(NF(P))$ und somit ist $NF(P)$ ebenfalls äquivalent zu P bezüglich der Relation $\sqsubseteq_{Qui}$. Es ergibt sich eine analoge Folgerung zu Proposition 2.19.

Proposition 3.14 (*Normal Form*). *Jedes MEIO P ist äquivalent zu seiner Normal Form $NF(P)$ bezüglich der Relation $\sqsubseteq_{Qui}$.*

Das Lemma 2.20 kann für die Stille-Traces erweitert werden. Alle Aussagen, die Lemma 2.20 trifft sind auch in diesem Kapitel weiterhin gültig.

Lemma 3.15. *Sei P ein MEIO in Normal Form. $w \in StET(P)\backslash ET(P)$ gilt genau dann, wenn w sich aus einem Ablauf ergibt, der einen Zustand erreicht, für denen es keine ausgehenden must-Transitionen für lokale Aktionen gibt.*

Die Normal Form hat das Problem mit dem Fehler-Zuständen gelöst. In der Definition 2.21 gab es eine spezielle Behandlung der Input-Transitionen um in der Konjunktion die Input-kritischen Traces entsprechend zu berücksichtigen. Auch für die in diesem Kapitel neu eingeführte Trace Menge QET kann es zu Problemen kommen, die durch die Normal Form und Konjunktion im Fehler-Kapitel noch nicht gelöst wurden. Es soll wiederum $QET_{P\wedge Q} = QET_P \cap QET_Q$ gelten. Für die ET Mengen gilt bereits eine analoge Aussage. Es ist also nur noch notwendig sich mit Traces auseinander zu setzten, die in mindestens einer Komponente der Konjunktion einen stillen Zustand erreichen. Falls P und Q beide einen stillen Zustand erreichen, kann es in der Konjunktion aus den beiden stillen Zuständen auch keine ausgehenden lokalen Aktionen geben, da in $P \wedge Q$ nur Transitionen möglich sind, die mindestens an einem der beteiligten Zustände möglich ist. Es sind also alle im Schritt der fehler-gefluteten Stille-Traces enthaltenen Elemente auch in der Menge $QET_{P\wedge Q}$ enthalten. Jedoch kann die Menge $QET_{P\wedge Q}$ echt größer sein, wenn man die Definitionen aus dem letzten Kapitel unverändert anwendet. Falls P eine must-Output-Transition besitzt und Q keine passende Output-Transition enthält. Hat der zusammengesetzt keine ausgehende Transition für den Output. Somit kann es dazu kommen, dass ein Zustand, der aus zwei Zuständen zusammen gesetzt ist, von denen mindestens einer nicht still ist, in der Konjunktion trotzdem still ist. An den Zuständen, an denen das beschriebene Problem auftritt kann diese durch neu hinzugefügte τ-Schlingen gelöst werden.

Definition 3.16 (*Konjunktion*). *Es werden die Voraussetzungen und die Konstruktion der Konjunktion aus 2.21 in diese Definition übernommen. Man erhält $P_1 \wedge P_2$ in diesem Kapitel aus der konstruierten Konjunktion in dem man eine τ-Schleife an jeden Zustand $(p_1, p_2) \in Qui$ anfügt, für den keiner der p_j ein Fehler-, Input-kritischer oder stiller Zustand ist.*

Für dies abgewandelte Konjunktions-Definition kann nun auch für die Relation $\sqsubseteq_{Qui}$ gezeigt werden, dass $\wedge$ die gewünscht größte untere Schranke ist.

Satz 3.17 (*Konjunktion*). *Für drei MEIOs S, P_1 und P_2 mit der gleichen Signatur gilt $S \sqsubseteq_{Qui} P_1 \wedge P_2$ genau dann, wenn $S \sqsubseteq_{Qui} P_1$ und $S \sqsubseteq_{Qui} P_2$ erfüllt ist.*

Beweis. Man kann sich hier, wie auch im Beweis von Satz 2.22 auf MEIOs in Normal Form beschränken. Dies ist möglich, da jeder MEIO einen bezüglich $\sqsubseteq_{Qui}$ äquivalenten MEIO in Normal Form besitzt nach Proposition 3.14.
Damit $P \sqsubseteq_{Qui} Q$ gilt, muss $P \sqsubseteq_E Q$ und $QET_P \subseteq QET_Q$ erfüllt sein (Definition 3.5). Für die Relation $\sqsubseteq_E$ wurde bereits im Fehler-Kapitel ein analoger Satz (2.22) bewiesen. Somit reicht es um diesen Satz zu zeigen, sich auf die Inklusionen der QET-Mengen zu beschränken.

„$\Leftarrow$“:
Für diese Richtung muss die Inklusion $QET_S \subseteq QET_{P_1 \wedge P_2}$ beweisen wurden unter der Voraussetzung, dass $QET_S \subseteq QET_1$ und $QET_S \subseteq QET_2$ gilt. Auch der Satz 2.22 wird im laufe des Beweises benötigt werden, dies ist zulässig, da $\sqsubseteq_{Qui}$ eine Verfeinerung der Fehler-Relation $\sqsubseteq_E$ ist.
Für ein beliebiges $w \in QET_S$ gilt $w \in QET_1 \cap QET_2$. Da QET die Vereinigung einiger Trace-Mengen ist, müssen gibt es unterschiedliche Fälle für das Wort w.

- Fall 1 ($w \in ET_1 \cap ET_2$): Durch den Beweis von Satz 2.22 folgt in diesem Fall $w \in ET_{P_1 \wedge P_2} \subseteq QET_{P_1 \wedge P_2}$.
- Fall 2 ($w \in StQT_1 \backslash ET_1 \cap ET_2$ oder $w \in ET_1 \cap StQT_2 \backslash ET_2$): Durch die Kommutativität der Konjunktion kann man sich in diesem Fall oBdA auf $w \in StQT_1 \backslash ET_1 \cap ET_2$ einschränken. Es gibt in P_1 einen Ablauf für das Wort w, der zu einem Zustand führt, der keine ausgehenden must-Transitionen für lokale Aktionen besitzt. Für $w \in ET_2$ gibt es jedoch unterschiedliche Fälle.

  TODO: Behandlung lokaler Aktionen in der Konjunktions-Definition ähnlich zu MIT-Traces anpassen

  - Fall 2a) ($w \in \text{cont}(PrET_2)$): Es ist ein mit w beschrifteter Trace in P_2 ausführbar, der zum Zustand e_2 führt. e_2 hat für alle Output-Aktionen eine must-Schleife zu sich selbst. TODO: Argumentation nach Anpassung der Definition beenden
  - Fall 2b) ($w \in \text{cont}(MIT_2)$): Es gibt ein echtes Präfix v von w, dass in P_2 ausführbar ist zu einem Zustand, der für einen Input a keine ausgehende must-Transition besitzt. va muss dabei ein Präfix von w sein. Da w kein Element von ET_1 ist, muss P_1 nach dem Wort v den Input a via einer must-Transition sicherstellen. Es kommt also die Regel (IMust2) aus der Definition 2.21 in der

Konjunktion aus dieser Stelle zum Einsatz. Dadurch wird in der Komponente P_2 der Zustand e_2 erreicht. Der Rest der Argumentation verläuft analog zu Fall 2a).

- Fall 3 ($w \in StQT_1 \backslash ET_1 \cap StQT_2 \backslash ET_2$): In P_1 und P_2 gibt es einen Ablauf für das Wort w, dass einen stillen Zustand erreicht. Der Zustand, der aus diesem beiden Zuständen in Konjunktion entsteht, kann keine ausgehenden must-Transitionen für Outputs oder die interne Aktion τ haben. Der Zustand muss also ebenfalls still sein. Die Transitionen, die für w in P_1 und P_2 verwendet werden, gibt es auch in der Konjunktion und durch dies Transitionen wird der stille Zustand erreicht. w ist also ein strikter Stille-Trace in $P_1 \wedge P_2$. Es gilt also $w \in StQT_{P_1 \wedge P_2} \subseteq QET_{P_1 \wedge P_2}$.

„⇒“:
Die Inklusionen $QET_S \subseteq QET_1$ und $QET_S \subseteq QET_2$ müssen für diesen Punkt unter der Voraussetzung $S \sqsubseteq_{Qui} P_1 \wedge P_2$ beweisen werden. Es reicht wie in Satz 2.22 aus nur die Inklusion für P_1 zu begründen, da daraus mit der Kommutativität der Konjunktion auch die Inklusion für P_2 folgt.
Sei w beliebig aus QET_2. w muss auch in der Menge $QET_{P_1 \wedge P_2}$ enthalten sein. Falls $w \in ET_{P_1 \wedge P_2}$ gilt folgt mit Satz 2.22 $w \in ET_1 \subseteq QET_1$. Somit müssen im folgenden nur noch w aus der Menge $StQT_{P_1 \wedge P_2} \backslash ET_{P_1 \wedge P_2}$ betrachtet werden. Das w ist in der Konjunktion von P_1 und P_2 zu einem Zustand ausführbar, der still ist.

TODO: Argumentation nach Veränderung der Definition beenden □

4 Verfeinerungen für Kommunikationsfehler-, Stillstand- und Divergenz-Freiheit

In diesem Kapitel soll die Menge der betrachteten Zustände noch einmal erweitert werden. Somit werden dann Fehler-, stille und Divergenz-Zustände betrachtet. Wie im letzten Kapitel wird auch hier nur der Testing-Ansatz betrachtet.

4.1 Testing-Ansatz

Definition 4.1 (*Divergenz*). *Ein* Divergenz-Zustand *ist ein Zustand in einem MEIO P, der via may-Transitionen eine unendliche Folge an τs ausführen kann.*
Die Menge $Div(P)$ besteht aus all diesen divergenten Zuständen des MEIOs P.

Die unendliche Folge an τs kann durch eine Schleife an einem durch interne Aktionen erreichbaren Zustand ausführbar sein oder durch einen Weg, der mit τs ausführbar ist, der unendliche viele Zustände durchläuft. Es ist jedoch zu beachten, dass ein Zustand, von dem aus unendlich viele Zustände durch τs erreichbar sind, nicht divergent sein muss. Es ist auch möglich, dass dieser Zustand eine unendliche Verzweigung hat und somit keine unendlichen Folgen an τs ausführen kann.
Als Erreichbarkeitsbegriff wird wieder die lokale Erreichbarkeit verwendet und somit eine optimistische Betrachtungsweise. Da das Divergieren eines Systems nicht mehr verhindert werden kann, sobald ein divergenter Zustand lokal erreicht werden kann, ist Divergenz als ähnlich „schlimm“ zu bewerten wie ein Fehler-Zustand.

Definition 4.2 (*Test und Verfeinerung für Divergenz*). *Ein* Test *T ist eine Implementierung. Ein MEIO P* as-erfüllt *einen Divergenz-Test T, falls $S \| T$ fehler-, stillstand- und divergenz-frei ist für alle $S \in$ as-impl(P). Es wird dann $P\, sat^{Div}_{\text{as}}\, T$ geschrieben. Die Parallelkomposition $S \| T$ ist* fehler-, stillstand- *und* divergenz-frei*, wenn kein Fehler-, stiller- oder Divergenz-Zustand lokal erreichbar ist.*
Ein MEIO P Divergenz-verfeinert *P', falls für alle Tests T: $P'\, sat^{Div}_{\text{as}}\, T \Rightarrow P\, sat^{Div}_{\text{as}}\, T$.*

Da nun die grundlegenden Definitionen für Divergenz festgehalten sind, kann man sich einen Begriff für die Traces zu divergenten Zuständen bilden. Im letzten Absatz wurde bereits festgestellt, dass Divergenz als ähnlich „schlimmes“ Fehlverhalten anzusehen ist

wie Fehler und dass das Divergieren eines Systems nicht mehr verhinderbar ist, sobald ein divergenter Zustand lokal erreichbar ist, kommt für die Divergenz-Traces wieder die prune-Funktion zum Einsatz. Ein System, das unendliche viele τs ausführen kann, ist von außen nicht von so einem System zu unterscheiden, das einen Fehler-Zustand erreicht. Somit wird in den Trace-Mengen auch nicht zwischen Fehler-Traces und Divergenz-Traces explizit unterschieden. Dadurch genügt es nicht mehr nur mit den Fehler-Traces die Sprache zu fluten, sondern es muss sowohl mit den Fehler-Traces wie auch den Divergenz-Traces geflutet werden. Ebenso werden die strikten Stille-Traces mit diesen beiden Trace-Mengen geflutet.

Definition 4.3 (*Divergenz-Traces*). *Sei P ein MEIO und definiere:*

- strikte Divergenz-Traces*:* $StDT(P) := \left\{w \in \Sigma^* \mid p_0 \overset{w}{\Rightarrow}_P p \in Div(P)\right\}$,
- gekürzte Divergenz-Traces*:* $PrDT(P) := \{\text{prune}(w) \mid w \in StDT(P)\}$.

Analog zu den Propositionen 2.4 und 3.4 gibt es hier auch eine Proposition, die die Divergenz-Traces eines MEIOs mit den Divergenz-Traces seiner as-Implementierungen verbindet. Die Begründung verläuft analog zu den Propositionen der vorangegangenen Kapitel.

Proposition 4.4 (*Divergenz-Traces und Implementierungen*). *Sei P ein MEIO.*

1. *Für die strikten Divergenz-Traces gilt:* $StDT(P) \subseteq \bigcup_{P' \in \text{as-impl}(P)} StDT(P')$.
2. *Für die gekürzten Divergenz-Traces von P gilt:* $PrDT(P) \subseteq \bigcup_{P' \in \text{as-impl}(P)} PrDT(P')$.

Beweis.

1. Um diese Inklusion beweisen zu können wird wieder eine as-Implementierung P' von P und eine passende as-Verfeinerungs-Relation $\mathcal{R}$ angegeben, so dass alle strikten Divergenz-Traces von P auch in P' enthalten sind. In diesem Fall funktioniert der Ansatz alle Traces aus P in P' zu implementieren und keine Fehler-Zustände zu übernehmen. Die Definition von P' lautet also:
   - $P' = P$,
   - $p'_0 = p_0$,
   - $I_{P'} = I_P$ und $O_{P'} = O_P$,
   - $\longrightarrow_{P'} = \dashrightarrow_{P'} = \dashrightarrow_P$,
   - $E_{P'} = \emptyset$.

Die passende as-Verfeinerungs-Relation $\mathcal{R}$ ist die Identität-Relation. Wie bereits im Beweis zu Proposition 1.8 begründet erfüllt $\mathcal{R}$ alle Punkte der Definition 1.3 um eine as-Verfeinerungs-Relation zu sein. Es wird ein w aus $StDT(P)$ betrachtet. Es gibt also einen Trace in P auf dem das Wort w ausgeführt wird und der einen divergenten Zustand erreicht. Es gilt also $\exists w' \in \Sigma_\tau^*, \exists \alpha_1, \alpha_2, \ldots, \alpha_n, \exists p_1, p_2, \ldots, p_n : \hat{w'} = w \wedge w' = \alpha_1\alpha_2 \ldots \alpha_n \wedge p_0 \xrightarrow{\alpha_1}_P p_1 \xrightarrow{\alpha_2}_P \ldots p_{n-1} \xrightarrow{\alpha_n}_P p_n \in Div_P$. Die Identitäts-Relation $\mathcal{R}$ setzt die Zustände des Traces mit den analogen Zuständen aus P' in Relation. Zusätzlich mit der Implementierung aller Transitionen aus P in P' ergibt sich der selbe Trace in P'. Es gilt also $p'_0 \xrightarrow{\alpha_1}_{P'} p'_1 \xrightarrow{\alpha_2}_{P'} \ldots p'_{n-1} \xrightarrow{\alpha_n}_{P'} p'_n$ mit $(p'_j, p_j) \in \mathcal{R}$ für $0 \leq j \leq n$. Da p_n in Div_P enthalten ist, gibt es von diesem Zustand in P aus die Möglichkeit eine unendliche Folge an τs auszuführen. Die Ausführbarkeit muss sich dabei auf mit τ beschriftete may-Transitionen in P stützen. Da diese Transitionen in P' alle übernommen wurden, ist auch für p'_n eine unendliche Folge an τs ausführbar. Es gilt also $p'_n \in Div_{P'}$ und somit $w \in StDT(P')$. Insgesamt folgt also für diese P' $StDT(P) = StDT(P')$.

2. Dieser Punkt entspricht 1. bis auf die Anwendung der prune-Funktion auf beiden Seiten des Inklusions-Symbols. Da prune monoton ist, folgt dieser Punkt direkt auf dem letzten.

□

Da die Stille-Traces mit den Fehler- und Divergenz-Traces geflutet werden sollen, kann die Stillstands-Semantik nicht aus dem letzten Kapitel übernommen werden. Auch die geflutete Sprache aus dem Fehler-Kapitel kann nicht beibehalten werden. Nur die Fehler-Traces ET können ohne Veränderung auch in diesem Kapitel verwendet werden. Jedoch werden diese Traces im weiteren Verlauf nur innerhalb der größeren Trace-Menge EDT relevant sein.

Definition 4.5 (*Kommunikationsfehler-, Stillstands- und Divergenz-Semantik*). *Sei P ein MEIO.*

- *Die Menge der* Divergenz-Traces *von P ist $DT(P) := \text{cont}(PrDT(P))$.*
- *Die Menge der* Fehler-Divergenz-Traces *von P ist $EDT(P) := ET(P) \cup DT(P)$.*
- *Die Menge der* Fehler-Divergenz-gefluteten Stille-Traces *von P ist $QDT(P) := StQT(P) \cup EDT(P)$.*
- *Die Menge der* Fehler-Divergenz-gefluteten Sprache *von P ist $EDL(P) := L(P) \cup EDT(P)$.*

Für zwei MEIOs P_1, P_2 mit der gleichen Signatur schreibt man $P_1 \sqsubseteq_{Div} P_2$, wenn $EDT_1 \subseteq EDT_2, QDT_1 \subseteq QDT_2$ und $EDL_1 \subseteq EDL_2$ gilt.

Proposition 4.6 (*Kommunikationsfehler-, Stillstands-, Divergenz-Semantik und Implementierungen*). *Sie P ein MEIO.*

1. *Für die Menge der Divergenz-Traces von P gilt* $DT(P) \subseteq \bigcup_{P' \in \text{as-impl}(P)} DT(P')$.
2. *Für die Menge der Fehler-Divergenz-Traces von P gilt die folgende Gleichheit* $EDT(P) = \bigcup_{P' \in \text{as-impl}(P)} EDT(P')$.
3. *Für die Menge der Fehler-Divergenz-gefluteten Stille-Traces von P gilt* $QDT(P) = \bigcup_{P' \in \text{as-impl}(P)} QDT(P')$.
4. *Für die Menge der Fehler-Divergenz-gefluteten Sprache von P gilt* $EDL(P) = \bigcup_{P' \in \text{as-impl}(P)} EDL(P')$.

Beweis.
1.:
Es gilt bereits $PrDT(P) \subseteq \bigcup_{P' \in \text{as-impl}(P)} PrDT(P')$, wegen 2. der Proposition 4.4. Aus der Monotonie von cont folgt also auch die hier geforderte Inklusion.

2. „$\subseteq$“:

$$\begin{aligned}
EDT(P) &\overset{4.5}{=} ET(P) \cup DT(P) \\
&\overset{2.7.1}{=} \left(\bigcup_{P' \in \text{as-impl}(P)} ET(P') \right) \cup DT(P) \\
&\overset{1.}{\subseteq} \left(\bigcup_{P' \in \text{as-impl}(P)} ET(P') \right) \cup \left(\bigcup_{P' \in \text{as-impl}(P)} DT(P') \right) \\
&= \bigcup_{P' \in \text{as-impl}(P)} ET(P') \cup DT(P') \\
&\overset{4.5}{=} \bigcup_{P' \in \text{as-impl}(P)} EDT(P').
\end{aligned}$$

2. „$\supseteq$“:
Für ein präfix-minimales w aus der Menge $EDT(P')$ einer as-Implementierung P' von P wird für diese Inklusion gezeigt, dass auch $w \in EDT(P)$ gilt. Es genügt ein präfix-minimales Element, da die EDT-Mengen unter cont abgeschlossen sind. Falls das w in $ET(P')$ enthalten ist, folgt $w \in ET(P) \subseteq EDT(P)$ aufgrund des ersten Punktes der Proposition 2.7. Es ist also nur noch der Fall zu betrachten, in dem $w \in DT(P') \backslash ET(P')$ gilt. Es gibt also einen Trace der Form $p'_0 \overset{\alpha_1}{\dashrightarrow}_{P'} p'_1 \overset{\alpha_2}{\dashrightarrow}_{P'} \dots p'_{n-1} \overset{\alpha_n}{\dashrightarrow}_{P'} p'_n \in Div_{P'}$ in P', wobei $\alpha_1 \alpha_2 \dots \alpha_n$ dem Wort wv bis auf interne Aktionen entspricht mit $v \in O^*$. Keiner der Zustände p'_j mit $0 \leq j \leq n$ ist in $E_{P'}$ enthalten. Da P' eine as-Implementierung von P ist, muss es eine as-Verfeinerungs-Relationen $\mathcal{R}$ zwischen P' und P geben mit $(p'_0, p_0) \in \mathcal{R}$. Solange in P kein Fehler-Zustand erreicht wird, fordert 1.3.3, dass der Trace

aus P' in P ebenfalls möglich ist. Falls ein Fehler-Zustand in P erreicht wird, ist ein Präfix von wv in $StET(P)$ enthalten und mit $w = \text{prune}(wv)$ gilt somit $w \in ET(P) \subseteq EDT(P)$. Es gibt den Trace $p_0 \xrightarrow{\alpha_1}_P p_1 \xrightarrow{\alpha_2}_P \ldots p_{n-1} \xrightarrow{\alpha_n}_P p_n$ mit $p'_j \mathcal{R} p_j$, falls keiner der Zustände p_j in E_P enthalten ist für $j \in \{0, 1, \ldots, n\}$. Für p'_n in ein Transitionsfolge aus unendlichen vielen τs ausführbar. Es gibt also entweder eine Schleife, die via τs immer wieder durchlaufen werden kann, oder einen Weg mit unendlichen vielen Zuständen, die durch τ-Transitionen verbunden sind. Falls innerhalb der Schleife oder auf dem Weg ein Zustand in $E_{P'}$ enthalten ist, fordert $\mathcal{R}$ mit 1.3.1, dass der erste solche Zustand in Relation mit einem Zustand aus E_P steht und somit $w \in ET_P \subseteq EDT_P$ gilt. Falls alle Zustände, die auf dem der unendlichen τ-Folge von p'_n in P' aus liegen nicht in $E_{P'}$ enthalten sind, fordert 1.3.3 auch deren Ausführbarkeit von p_n aus in P, bis entweder ein Zustand in E_P erreicht wird, oder ebenfalls eine unendliche τ-Folge in P von p_n aus ausführbar ist. Im ersten Fall gilt wieder $w \in ET_P \subseteq EDT_P$. Ansonsten gilt $p_n \in Div_P$ und somit $wv \in StDT(P)$. Mit $w = \text{prune}(wv)$ folgt $w \in PrDT(P) \subseteq EDT(P)$.

3. „$\subseteq$“:

$$
\begin{aligned}
QDT(P) &\overset{4.5}{=} StQT(P) \cup EDT(P) \\
&\overset{3.4}{\subseteq} \left(\bigcup_{P' \in \text{as-impl}(P)} StQT(P') \right) \cup EDT(P) \\
&\overset{2.}{=} \left(\bigcup_{P' \in \text{as-impl}(P)} StQT(P') \right) \cup \left(\bigcup_{P' \in \text{as-impl}(P)} EDT(P') \right) \\
&= \bigcup_{P' \in \text{as-impl}(P)} StQT(P') \cup EDT(P') \\
&\overset{4.5}{=} \bigcup_{P' \in \text{as-impl}(P)} QDT(P').
\end{aligned}
$$

3. „$\supseteq$“:
Dieser Beweis verläuft analog zu dem Beweis von „$\supseteq$“ der Proposition 3.6, man muss nur die ET-Mengen durch EDT-Mengen ersetzten und für den Fall $w \in EDT(P')$ folgt $w \in EDT(P)$ wegen des zweiten Punktes dieser Proposition und nicht wegen Proposition 2.7.

4. „$\subseteq$“:

$$\begin{aligned} EDL(P) &\overset{4.5}{=} L(P) \cup EDT(P) \\ &\overset{1.8}{\subseteq} \left(\bigcup_{P' \in \text{as-impl}(P)} L(P') \right) \cup EDT(P) \\ &\overset{2.}{=} \left(\bigcup_{P' \in \text{as-impl}(P)} L(P') \right) \cup \left(\bigcup_{P' \in \text{as-impl}(P)} EDT(P') \right) \\ &= \bigcup_{P' \in \text{as-impl}(P)} L(P') \cup EDT(P') \\ &\overset{4.5}{=} \bigcup_{P' \in \text{as-impl}(P)} EDL(P'). \end{aligned}$$

4. „$\supseteq$“:
Für den Beweis dieser Inklusion kann man auf den Beweis von 2.7.2 „$\supseteq$“ zurück greifen. Es müssen wie bei 3. nur die ET-Mengen durch EDT-Mengen ersetzt werden und die Einschränkung auf die geflutete Sprache ohne die Menge EDT ist möglich wegen des zweiten Punktes der aktuellen Proposition. □

Aus der so eben bewiesenen Proposition über die Gleichheit der betrachteten Traces, lässt sich wie in den letzten beiden Kapiteln eine Aussage über die lokale Erreichbarkeit der fehlerhaften Zustände in einer Spezifikation und den zugehörigen as-Implementierungen treffen.

Korollar 4.7 (*lokale Divergenz Erreichbarkeit*).

(i) Falls in einem MEIO P ein Fehler lokal erreichbar ist, dann existiert auch eine as-Implementierung, in der ein Fehler lokal erreichbar ist.

(ii) Falls in einem MEIO P Divergenz lokal erreichbar ist, dann existiert auch eine as-Implementierung, in der Divergenz lokal erreichbar ist.

(iii) Falls ein MEIO P einen lokal erreichbaren stillen Zustand besitzt, dann existiert auch eine as-Implementierung, in der ein stiller Zustand lokal erreichbar ist.

(iv) Falls es eine as-Implementierung von P gibt, die Fehler, Stille oder Divergenz lokal erreicht, dann ist auch Fehler, Stille oder Divergenz in P lokal erreichbar.

Beweis.

(i) Dieser Punkt folgt wie in 3.7 direkt aus Korollar 2.8 (i).

(ii) Ein divergenter Zustand ist in P lokal erreichbar, wenn $\varepsilon \in DT_P$ gilt. Mit 4.6.1 folgt daraus, dass es auch mindestens eine as-Implementierung P' aus as-impl(P) geben muss, für die ε in $DT(P')$ enthalten ist. Da DT die Menge der fortgesetzten um lokale Aktionen gekürzten strikten Divergenz-Traces ist, muss es lokale Aktionen in P' geben, die zu einem divergenten Zustand führen. Es ist also auch in P' Divergenz lokal erreichbar.

(iii) Dieser Punkt folgt direkt aus Korollar 3.7 (ii).

(iv) In $P' \in$ as-impl(P) sei ein Fehler-, stillen oder Divergenz-Zustand lokal erreichbar. Es gilt dann $w \in QDT_{P'}$ für $w \in O^*$. Mit Proposition 4.6.3 gilt auch $w \in QDT_P$ Die Menge QDT setzt sich aus den Mengen ET, $StQT$ und DT zusammen. Es muss also in P ein Fehler-, stiller oder Divergenz-Zustand lokal erreichbar sein, da ein w, bestehend nur aus lokalen Aktionen, in QDT_P enthalten ist.

□

Die Relation$\sqsubseteq_{Div}$ ist somit keine Einschränkung von $\sqsubseteq_E$ so wie $\sqsubseteq_{Qui}$. Es können Systeme mit einem Fehler nicht von Systemen mit Divergenz unterschieden werden. Da die Divergenz-Test zwischen diesen Fehler-Arten auch keine Unterscheidung machen, muss eine sinnvolle Relation dies Eigenschaft auch übernehmen, so wie $\sqsubseteq_{Div}$ dies tut.

Satz 4.8 (*Kommunikationsfehler-, Stillstands- und Divergenz-Semantik für Parallelkompositionen*). *Für zwei komponierbare MEIOs P_1, P_2 und ihre Komposition P_{12} gilt:*

1. $EDT_{12} = \text{cont}(\text{prune}((EDT_1 \| EDL_2) \cup (EDL_1 \| EDT_2)))$,
2. $QDT_{12} = (QDT_1 \| QDT_2) \cup EDT_{12}$,
3. $EDL_{12} = (EDL_1 \| EDL_2) \cup EDT_{12}$.

Beweis.
1. „$\subseteq$“:
Da beide Seiten der Gleichung unter cont abgeschlossen sind, genügt es ein präfixminimales Element w zu betrachten. Es muss hier unterschieden werden, ob $w \in ET_{12}$ oder $w \in DT_{12} \backslash ET_{12}$ betrachtet wird. Im ersten Fall ist das w in der rechten Seite der Gleichung enthalten wegen des Beweises des ersten Punktes von Satz 2.9 und da $ET(P) \subseteq EDT(P)$ und $EL(P) \subseteq EDL(P)$ gilt. Deshalb wird im weiteren Verlauf dieses Beweises davon ausgegangen, dass $w \in DT_{12} \backslash ET_{12}$ gilt und es wird versuch zu zeigen, dass dieses w ebenfalls in der rechten Seite enthalten ist. Da das betrachtete w präfixminimal ist, gilt $w \in PrDT_{12} \backslash ET_{12}$. Aus der Definition 4.5 weiß man, dass ein $v \in O_{12}^*$ existiert, sodass $(p_{01}, p_{02}) \overset{w}{\Longrightarrow}_{12} (p_1, p_2) \overset{v}{\Longrightarrow}_{12} (p_1', p_2')$ gilt mit $(p_1', p_2') \in Div_{12}$. Durch die Projektion auf die Transitionssysteme P_1 und P_2 erhält man $p_{01} \overset{w_1}{\Longrightarrow}_1 p_1 \overset{v_1}{\Longrightarrow}_1 p_1'$ und $p_{02} \overset{w_2}{\Longrightarrow}_2 p_2 \overset{v_2}{\Longrightarrow}_2 p_2'$ mit $w \in w_1 \| w_2$ und $v \in v_1 \| v_2$. Aus $(p_1', p_2') \in Div_{12}$ folgt, dass oBdA $p_1' \in Div_1$ gilt, d.h. $w_1 v_1 \in StDT_1 \subseteq EDT_1$. Da $p_{02} \overset{w_2 v_2}{\Longrightarrow}_2$ gilt, erhält man $w_2 v_2 \in EDL_2$.

Somit gilt insgesamt $wv \in EDT_1 \| EDL_2$ und da $v \in O_{12}^*$, gilt $\text{prune}(wv) = w$ und w ist in der rechten Seite der Gleichung enthalten.

1. „$\supseteq$“:
Es wird ebenso wie oben nur ein präfix-minimales x betrachtet wegen des Abschlusses beider Seiten der Gleichung unter cont. Es wird also für ein beliebiges $x \in \text{prune}((EDT_1 \| EDL_2) \cup (EDL_1 \| EDT_2))$ gezeigt, dass dieses oder eines seiner Präfixe auch in EDT_{12} enthalten ist. Da das x aus der prune-Funktion entstanden ist, lässt sich ein y aus O_{12}^* finden, sodass $xy \in (EDT_1 \| EDL_2) \cup (EDL_1 \| EDT_2)$. Es wird nun noch vorausgesetzt, dass oBdA $xy \in EDT_1 \| EDL_2$ gilt, d.h. es existiert $w_1 \in EDT_1$ und $w_2 \in EDL_2$ mit $xy \in w_1 \| w_2$. TODO: erzwungenen Zeilenumbruch kontrollieren
Die folgende Argumentation läuft analog zu der im Beweis der Inklusion $ET_{12} \supseteq \text{cont}(\text{prune}((ET_1 \| EL_2) \cup (EL_1 \| ET_2)))$ aus Satz 2.9. Es muss dazu nur jeweils an den Stellen, an denen $PrET(P) \cup MIT(P)$ steht auch noch eine Vereinigung mit $PrDT(P)$ vorgenommen werden. Für Fall I und II aus dem Beweis der oben genannten Inklusion von Satz 2.9 ist jeweils kein weiterer Unterfall für v_2' notwendig da, wenn v_2' nicht ausführbar ist, bereits ein Fehler-Zustand in der Parallelkomposition entsteht. Somit ist egal, ob auch noch Divergenz vorlag. Falls v_2' ausführbar, ist nicht relevant, ob eine Divergenz-Möglichkeit bestanden hat, da diese nicht an der Ausführbarkeit ändert. Am Ende ist ein zusätzlicher Fall für $v_1 \in PrDT_1$ zu ergänzen: TODO: erzwungenen Zeilenumbruch kontrollieren

- Fall III ($v_1 \in PrDT_1$): Es existiert ein u_1 aus O_1^*, sodass $p_{01} \overset{v_1}{\Rightarrow}_1 p_1 \overset{u_1}{\Rightarrow}_1 p_1'$ mit $p_1' \in Div_1$ gilt. Da es hier keine disjunkten Inputmengen gibt kann das a, auf das v_1 im Fall $v_1 \neq \varepsilon$ endet, ebenfalls der letzte Buchstabe von v_2 sein. Im Fall von $v_2 \in MIT_2$ kann somit $a = b$ gelten und damit wäre $v_2 = v_2'$. Dieser Fall verläuft jedoch analog zu Fall Ic) aus dem Beweis der oben genannten Inklusion von Satz 2.9 und wird somit hier nicht weiter betrachtet. Deshalb gilt für alle im folgenden betrachteten Fälle $p_{02} \overset{v_2'}{\Rightarrow}_2 p_2$ mit $(p_{01}, p_{02}) \overset{v'}{\Rightarrow}_{12}$.

  - Fall IIIa) $\Big(u_2 \in (O_1 \cap I_2)^*, c \in (O_1 \cap I_2)$, sodass $u_2 c$ ein Präfix von $u_1|_{I_2}$ mit $p_2 \overset{u_2}{\Longrightarrow}_2 p_2' \overset{c}{\not\rightarrow}_2\Big)$: Für ein Präfix von $u_1' c$ von u_1 mit $(u_1' c)|_{I_2} = u_2 c$ weiß man, dass $p_1 \overset{u_1'}{\Rightarrow}_1 p_1'' \overset{c}{\dashrightarrow}_1$. Somit gilt $u_1' \in u_1' \| u_2$ und $(p_1, p_2) \overset{u_1'}{\Rightarrow}_{12} (p_1'', p_2') \in E_{12}$, da für P_2 der entsprechende Input fehlt, der mit dem Output von c 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 IIIb) $\Big(p_2 \overset{u_2}{\Longrightarrow}_2 p_2'$ mit $u_2 = u_1|_{I_2}\Big)$: Somit ist $u_1 \in u_1 \| u_2$ und $(p_1, p_2) \overset{u_1}{\Rightarrow}_{12} (p_1', p_2') \in Div_{12}$, da $p_1 \in Div_1$. P_{12} hat also die Divergenz von P_1 geerbt. Es wird nun $v := \text{prune}(v' u_1) \in PrDT_{12}$ gewählt, das wiederum ein Präfix von v' ist.

2. „$\subseteq$“:
Diese Inklusionsrichtung kann analog zum Beweis derselben Inklusionsrichtung des zweiten Punktes von Satz 3.9 gezeigt werden. Es muss dabei nur in der Argumentation

die Menge ET_{12} durch die Menge EDT_{12} und die Mengen $QET(P)$ durch die Mengen $QDT(P)$ für die entsprechenden Transitionssysteme P ersetzt werden. Dadurch kann ebenso gefolgert werden, dass im Fall $w \in StQT_{12} \backslash EDT_{12}$ der erreichte Zustand (p_1, p_2) kein Fehler sein kann, da $ET_{12} \subseteq EDT_{12}$ gilt und somit lässt sich auch hier der zweite Punkt von Lemma 3.8 anwenden.

2. „$\supseteq$“:
Es muss wieder danach unterschieden werden, aus welcher Menge das betrachtete Element stammt. Falls w ein Element von EDT_{12} ist, folgt die Zugehörigkeit zur linken Seite der Gleichung direkt. Somit wird für den weiteren Verlauf dieses Beweises davon ausgegangen, dass $w \in QDT_1 \| QDT_2$ gilt. Für dieses w soll dann gezeigt werden, dass es auch in QDT_{12} enthalten ist. Da $QDT_i = StQT_i \cup EDT_i$ gilt, existierten für w_1 und w_2 mit $w \in w_1 \| w_2$ unterschiedliche Möglichkeiten:

- Fall 1 ($w_1 \in EDT_1 \vee w_2 \in EDT_2$): OBdA gilt $w_1 \in EDT_1$. Es kann nun $w_2 \in StQT_2 \subseteq L_2$ gelten oder $w_2 \in EDT_2 \subseteq EDL_2$ und somit gilt auf jeden Fall $w_2 \in EDL_2$. Daraus kann mit dem ersten Punkt dieses Satzes gefolgert werden, dass $w \in EDT_{12}$ gilt und somit w in der linken Seite der Gleichung enthalten ist.
- Fall 2 ($w_1 \in StQT_1 \backslash EDT_1 \wedge w_2 \in StQT_2 \backslash EDT_2$): Dieser Fall läuft analog zu Fall 2 derselben Inklusionsrichtung des Beweises von Satz 3.9. Hierfür muss die Menge QET_{12} durch QDT_{12} ersetzt werden.

3.:
Durch die Definition 4.5 ist klar, dass $L_i \subseteq EDL_i$ und $EDT_i \subseteq EDL_i$ gilt. Die Argumentation wird von der rechten Seite der Gleichung aus begonnen:

$$
\begin{aligned}
(EDL_1 \| EDL_2) \cup EDT_{12} &\overset{4.5}{=} ((L_1 \cup EDT_1) \,\|\, (L_2 \cup EDT_2)) \cup EDT_{12} \\
&= (L_1 \| L_2) \cup \underbrace{(L_1 \| EDT_2)}_{\substack{\subseteq (EDL_1 \| EDT_2) \\ \overset{1.}{\subseteq} EDT_{12}}} \cup \underbrace{(EDT_1 \| L_2)}_{\substack{\subseteq (EDT_1 \| EDL_2) \\ \overset{1.}{\subseteq} EDT_{12}}} \\
&\quad \cup \underbrace{(EDT_1 \| EDT_2)}_{\substack{\subseteq (EDL_1 \| EDT_2) \\ \overset{1.}{\subseteq} EDT_{12}}} \cup EDT_{12} \\
&= (L_1 \| L_2) \cup EDT_{12} \\
&\overset{1.9}{=} L_{12} \cup EDT_{12} \\
&\overset{4.5}{=} EDL_{12}.
\end{aligned}
$$

□

Analog wie in den beiden vorangegangenen Kapitel, ergibt sich aus diesem Satz als direkte Folgerung, dass es sich bei der Relation $\sqsubseteq_{Div}$ um eine Präkongruenz handelt.

Korollar 4.9 (*Divergenz-Präkongruenz*). *Die Relation $\sqsubseteq_{Div}$ ist eine Präkongruenz bezüglich $\cdot \| \cdot$.*

Beweis. Um zu zeigen, dass es sich bei $\sqsubseteq_{Div}$ um eine Präkongruenz handelt, muss nachgewiesen werden, dass aus $P_1 \sqsubseteq_{Div} P_2$ auch $P_{31} \sqsubseteq_{Div} P_{32}$ für jedes komponierbare System P_3 folgt. D.h. es ist zu zeigen, dass aus $EDT_1 \subseteq EDT_2, QDT_1 \subseteq QDT_2$ und $EDL_1 \subseteq EDL_2$, sowohl $EDT_{31} \subseteq EDT_{32}, QDT_{31} \subseteq QDT_{32}$ als auch $EDL_{31} \subseteq EDL_{32}$ folgt. Dies ergibt sich, wie in den Beweisen zu den Korollaren 2.10 und 3.10, aus der Monotonie von cont, prune und $\cdot\|\cdot$ auf Sprachen wie folgt:

- $EDT_{31} \overset{4.8.1}{=} \text{cont}\,(\text{prune}\,((EDT_3\|EDL_1) \cup (EDL_3\|EDT_1)))$
  $\overset{\substack{EDT_1\subseteq EDT_2 \\ \text{und} \\ EDL_1\subseteq EDL_2}}{\subseteq} \text{cont}\,(\text{prune}\,((EDT_3\|EDL_2) \cup (EDL_3\|EDT_2)))$
  $\overset{4.8.1}{=} EDT_{32},$

- $QDT_{31} \overset{4.8.2}{=} (QDT_3\|QDT_1) \cup EDT_{31}$
  $\overset{\substack{EDT_{31}\subseteq EDT_{32}, \\ \text{und} \\ QDT_1\subseteq QDT_2}}{\subseteq} (QDT_3\|QDT_2) \cup EDT_{32}$
  $\overset{4.8.2}{=} QDT_{32}.$

- $EDL_{31} \overset{4.8.3}{=} (EDL_3\|EDL_1) \cup EDT_{31}$
  $\overset{\substack{EDT_{31}\subseteq EDT_{32}, \\ \text{und} \\ EDL_1\subseteq EDL_2}}{\subseteq} (EDL_3\|EDL_2) \cup EDT_{32}$
  $\overset{4.8.3}{=} EDL_{32}.$

□

Im nächsten Lemma soll eine Verfeinerung bezüglich guter Kommunikation betrachtet werden. Die Vorgaben für gute Kommunikation gibt hierbei die Definition der Tests und die daraus resultierende Verfeinerung in 4.2 vor. Es muss in diesem Lemma eine Veränderung zu den analogen Lemmata aus den vorangegangenen Kapitel vorgenommen werden. Die Einschränkung der Tests T auf Partner, kann nicht mehr beibehalten werden, da die Strategie zur Vermeidung von Stille im Beweis aus dem letzten Kapitel hier zu Divergenz führen würde. Somit werden für die Stillstands-Vermeidung in diesem Kapitel Aktionen außerhalb der Menge Synch benötigt, die nicht die interne Aktionen τ sind. Jedoch müssen trotzdem nicht alle Tests T betrachtet werden. Es kann eine Einschränkung gemacht werden, sodass T fast ein Partner ist. Zur Vereinfachung von umständlichen Formulierungen im Folgenden wird hierfür nun ein neuer Begriff definiert. Der jedoch auch bereits so in z.B. [Sch16] analog für EIOs verwendet und definiert wurde.

Definition 4.10 (ω-Partner). *Ein MEIO P_1 ist ein ω-Partner von einem MEIO P_2, wenn $I_1 = O_2$ und $O_1 = I_2 \cup \{\omega\}$ mit $\omega \notin I_2 \cup O_2$ gilt.*

Ein ω-Partner P_1 von P_2 unterscheidet sich von einem Partner von P_2 nur um den Output ω, der nicht in der Menge Synch(P_1, P_2) enthalten ist.

Lemma 4.11 (*Testing-Verfeinerung mit Divergenz*). *Gegeben sind zwei MEIOs P_1 und P_2 mit der gleichen Signatur. Wenn für alle Tests T, die ω-Partner von P_1 bzw. P_2 sind, $P_2\, sat_{\text{as}}^{Div}\, T \Rightarrow P_1\, sat_{\text{as}}^{Div}\, T$ gilt, dann folgt daraus die Gültigkeit von $P_1 \sqsubseteq_{Div} P_2$.*

Beweis. Da P_1 und P_2 die gleiche Signatur haben, definiert man $I := I_1 = I_2$ und $O := O_1 = O_2$. Für jeden ω-Partner Test T gilt $I_T = O$ und $O_T = I \cup \{\omega\}$ mit $\omega \notin I \cup O$.
Um zu zeigen, dass die Relation $P_1 \sqsubseteq_{Div} P_2$ gilt, müssen die folgenden Punkte nachgewiesen werden:

- $EDT_1 \subseteq EDT_2$,
- $QDT_1 \subseteq QDT_2$,
- $EDL_1 \subseteq EDL_2$.

In den Lemmata 2.15 und 3.11 wurde bereits etwas Ähnliches gezeigt. Jedoch kann daraus aufgrund der unterschiedlichen Implikationen, die vorausgesetzt werden, nichts über dieses Lemma und dessen Gültigkeit ausgesagt werden. Es kann in diesem Lemma, ebenso wie im Lemma 3.11, aus der lokalen Erreichbarkeit eines Fehlers in einer Parallelkomposition einer as-Implementierung von P_1 mit einem Test T und der Implikation $P_2\, sat_{\text{as}}^{Div}\, T \Rightarrow P_1\, sat_{\text{as}}^{Div}\, T$ nur geschlossen werden, dass es in einer Parallelkomposition einer as-Implementierung von P_2 mit T auch einen lokal erreichbaren fehlerhaften Zustand geben muss, jedoch kann die Fehlerhaftigkeit hier ein Fehler, Stille oder Divergenz sein. Analog verhält es sich, wenn in der Parallelkomposition einer as-Implementierung von P_1 mit einem Test T ein Divergenz-Zustand oder stiller Zustand lokal erreichbar ist. Es kann nur geschlossen werden, dass P_2 den Test T nicht erfüllen darf. Die nicht Erfüllung kann jedoch auf einem beliebigen Fehlverhalten des MEIOs basieren.

Als Erstes wird der erste Beweispunkt gezeigt, also die Inklusion $EDT_1 \subseteq EDT_2$.
Es wird für ein präfix-minimales w aus EDT_1 gezeigt, dass dieses w oder eines seiner Präfixe in EDT_2 enthalten ist. Diese Möglichkeit bietet sich, da beide Mengen unter cont abgeschlossen sind. Wegen Proposition 4.6.2 ist w auch ein präfix-minimales Elemente der Menge $EDT_{P_1'}$ einer as-Implementierung P_1' von P_1.

- Fall 1 ($w = \varepsilon$): Es handelt sich um einen lokal erreichbaren Fehler oder um lokale erreichbare Divergenz in P_1'. Für T wird ein Transitionssysteme verwendet, das nur aus dem Startzustand und einer must-Schleife für alle Inputs $x \in I_T$ und einer must-Schleife für ω besteht. Somit kann P_1' im Prinzip die gleichen Fehler- und Divergenz-Zustände wie $P_1' \| T$ lokal erreichen. P_1 erfüllt den Test T nicht. P_2 darf T somit auch nicht erfüllen. Es muss eine as-Implementierung P_2' von P_2 existieren, für die $P_2' \| T$ einen fehlerhaften Zustand lokal erreicht. Durch die Struktur von T ist in einer Parallelkomposition mit T kein stiller Zustand möglich. Der fehlerhafte Zustand, der in $P_2' \| T$ lokal erreichbar ist, muss also ein Fehler- oder

Divergenz-Zustand sein. Da von T kein Fehler und keine Divergenz geerbt werden kann und durch die Input-Schleife auch kein neuer Fehler entstehen kann, muss der fehlerhafte Zustand von P_2' geerbt sein. Somit muss in P_2' ein Fehler- oder Divergenz-Zustand lokal erreichbar sein. Da $EDT(P) = ET(P) \cup DT(P)$ gilt, folgt $w \in EDT_{P_2'}$ und mit 4.6.2 $w \in EDT_2$.

- Fall 2 ($w = x_1 \dots x_n x_{n+1} \in \Sigma^+$ mit $n \geq 0$ und $x_{n+1} \in I$): Es wird der folgende ω-Partner T betrachtet (siehe auch Abbildung 4.1):
  - $T = \{p_0, p_1, \dots, p_{n+1}\}$,
  - $p_{0T} = p_0$,
  - $\dashrightarrow_T = \longrightarrow_T = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j \leq n\}$
    $\cup \{(p_j, x, p_{n+1}) \mid x \in I_T \backslash \{x_{j+1}\}, 0 \leq j \leq n\}$
    $\cup \{(p_{n+1}, x, p_{n+1}) \mid x \in I_T\}$
    $\cup \{(p_j, \omega, p_{n+1}) \mid 0 \leq j \leq n+1\}$,
  - $E_T = \emptyset$.

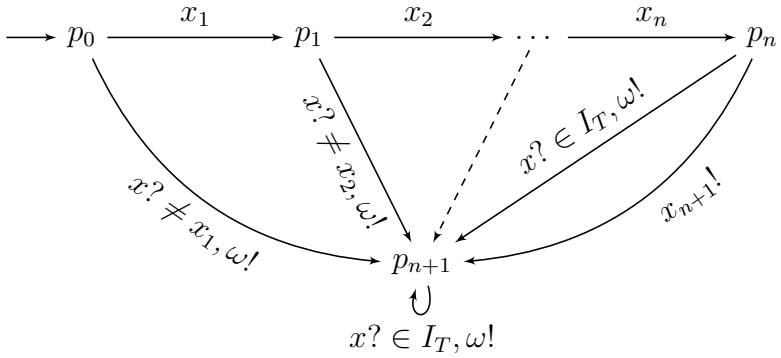

Abbildung 4.1: $x? \neq x_j$ steht für alle $x \in I_T \backslash \{x_j\}$

Die Mengen der Divergenz- und stillen Zustände des hier betrachteten Ts sind leer. Da im Vergleich zum Transitionssystem in Abbildung 2.1 nur die ω-Transitionen zu p_{n+1} ergänzt und die Mengen unbenannt wurden, ändert sich nichts an dem Fall 2a) im ersten Punkt des Beweises von Lemma 2.15. Im Fall 2b) muss die Menge O^* durch $(O \cup \{\omega\})^*$ ersetzt werden. Die Begründungen, wieso in den beiden Fällen $\varepsilon \in PrET(P_1' \| T)$ für ein $P_1' \in \text{as-impl}(P_1)$ gilt, bleibt also analog zum Beweis von Lemma 2.15. Da nun aber auch Divergenz betrachtet wird, muss ein weiterer Fall ergänzt werden:

- Fall 2c) ($w \in PrDT_{P_1'}$): In $P_1' \| T$ erhält man $(p_{01}, p_0) \stackrel{w}{\Longrightarrow} (p'', p_{n+1}) \stackrel{u}{\Longrightarrow} (p', p_{n+1})$ für $u \in (O \cup \{\omega\})^*$ und $p' \in Div_1$. Daraus folgt $(p', p_{n+1}) \in Div_{P_1' \| T}$ und somit $wu \in StDT(P_1' \| F)$. Da alle Aktionen aus w synchronisiert werden und $I_T \cap I = \emptyset$ gilt $x_1, \dots, x_n, x_{n+1} \in O_{P_1' \| T}$. Da zusätzlich u in $(O \cup \{\omega\})^*$ enthalten ist, folgt $u \in O^*_{P_1' \| T}$. Somit ergibt sich $\varepsilon \in PrDT(P_1' \| T)$.

Da ε in $PrET(P_1'\|T) \cup PrDT(P_1'\|T)$ enthalten ist, ist ein Fehler oder Divergenz lokal erreichbar in $P_1'\|T$. Mit der Implikation $P_2\, sat_{\text{as}}^{Div}\, T \Rightarrow P_1\, sat_{\text{as}}^{Div}\, T$ kann geschlossen werden, dass in der Parallelkomposition einer as-Implementierung P_2' von P_2 mit dem Test T ein fehlerhafter Zustand lokal erreichbar sein muss. Durch die ω-Transitionen an den Zuständen von T kann es in Komposition mit T keine stillen Zustände geben. Die Fehlerhaftigkeit muss also ein Fehler oder Divergenz sein.

- Fall 2i) ($\varepsilon \in ET(P_1'\|T)$ wegen neuem Fehler): Da jeder Zustand von T alle Inputs $x \in I_T = O$ zulässt, muss ein lokal erreichbarer Fehler-Zustand in diesem Fall der Form sein, dass ein Output $a \in O_T\backslash\{\omega\}$ von T möglich ist, der nicht mit einem passenden Input aus P_2' synchronisiert werden kann. Durch die Konstruktion von T ist in p_{n+1} kein Output außer ω möglich. Ein neuer Fehler muss also die Form (p', p_j) haben mit $j \leq n$, $p' \overset{x_{i+1}}{\not\rightarrow}_{P_2'}$ und $x_{i+1} \in O_T\backslash\{\omega\}$. Durch Projektion erhält man dann $p_{02} \overset{x_1 \ldots x_i}{\Longrightarrow}_{P_2'} p' \overset{x_{i+1}}{\not\rightarrow}_{P_2'}$ und damit gilt $x_1 \ldots x_{i+1} \in MIT_{P_2'} \subseteq ET_{P_2'}$. Somit ist ein Präfix von w in $EDT_{P_2'}$ enthalten. Wegen des Abschlusses unter cont und wegen Proposition 4.6.2 gilt $w \in EDT_2$.

- Fall 2ii) ($\varepsilon \in ET(P_2'\|T)$ wegen geerbtem Fehler): T hat $x_1 \ldots x_i u$ ausgeführt mit $u \in (O \cup \{\omega\})^*$ und ebenso hat P_2' den Weg $x_1 \ldots x_i u|_{\Sigma_2}$ ausgeführt. Durch dies hat P_2' einen Zustand aus $E_{P_2'}$ erreicht, da von T kein Fehler geerbt werden kann. Es gilt dann $\text{prune}(x_1 \ldots x_i u|_{\Sigma_2}) = \text{prune}(x_1 \ldots x_i) \in PrET_{P_2'} \subseteq ET_{P_2'}$. Da $x_1 \ldots x_i$ 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 $x_1 \ldots x_i$ in $EDT_{P_2'}$ enthalten und mit 4.6.2 ist ein Präfix von w in EDT_2 enthalten.

- Fall 2iii) ($\varepsilon \in DT(P_2'\|T)\backslash ET(P_2'\|T)$): Da T nicht unendlich viele Zustände hat und auch keine τ-Schleifen besitzt, kann das Divergenz-Verhalten nur von P_2' geerbt sein. T hat $x_1 \ldots x_i u$ ausgeführt mit $u \in (O \cup \{\omega\})^*$ und ebenso hat P_2' den Weg $x_1 \ldots x_i u|_{\Sigma_2}$ ausgeführt. Durch dies hat P_2' einen Zustand aus $Div_{P_2'}$ erreicht. Es gilt dann $\text{prune}(x_1 \ldots x_i u|_{\Sigma_2}) = \text{prune}(x_1 \ldots x_i) \in PrDT_{P_2'} \subseteq DT_{P_2'}$, da $u|_{\Sigma_2}$ in O^* enthalten ist. Da $x_1 \ldots x_i$ ein Präfix von w ist, führt in diesem Fall eine Verlängerung um lokale Aktionen von einem Präfix von w zu einem divergenten Zustand. Da DT die Menge aller Verlängerungen von gekürzten Divergenz-Traces ist und $DT_{P_2'} \subseteq EDT_{P_2'}$ gilt, ist in diesem Fall das Präfix $x_1 \ldots x_i$ von w in $EDT_{P_2'}$ enthalten. Mit 4.6.2 folgt daraus, dass ein Präfix von w in EDT_2 enthalten ist.

Als nächstes wird nun der zweite Beweispunkt gezeigt, d.h. die Inklusion $QDT_1 \subseteq QDT_2$. Diese Inklusion kann jedoch noch, analog zum Beweis der Inklusion der Fehler-gefluteten Sprache aus dem Fehler-Kapitel, weiter eingeschränkt werden. Da bereits bekannt ist, das $EDT_1 \subseteq EDT_2$ gilt, muss nur noch $StQT_1\backslash EDT_1 \subseteq QDT_2$ gezeigt werden.
Es wird ein $w \in StQT_1\backslash EDT_1$ gewählt und gezeigt, dass dieses auch in QDT_2 enthalten

ist. Das w ist aufgrund der Propositionen 3.4 und 4.6.2 auch für eine as-Implementierung P_1' von P_1 in $StQT_{P_1'} \backslash EDT_{P_1'}$ enthalten.
Durch die Wahl des ws wird in P_1' durch das Wort w ein stiller Zustand erreicht. Dies hat nur Auswirkungen auf die Parallelkomposition $P_1' \| T$, wenn in T ebenfalls ein stiller Zustand durch w erreicht wird.
Das betrachtete w hat die Form $w = x_1 \dots x_n \in \Sigma^*$ mit $n \geq 0$. Es wird der folgende ω-Partner Test T betrachtet (siehe auch Abbildung 4.2):

- $T = \{p_0, p_1, \dots, p_n, p\}$,
- $p_{0T} = p_0$,
- $\dashrightarrow_T = \longrightarrow_T = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j < n\}$
  $\cup \{(p_j, x, p) \mid x \in (I_T \cup \{\omega\}) \backslash \{x_{j+1}\}, 0 \leq j < n\}$
  $\cup \{(p_n, x, p) \mid x \in I_T\}$
  $\cup \{(p, x, p) \mid x \in I_T \cup \{\omega\}\}$,
- $E_T = \emptyset$.

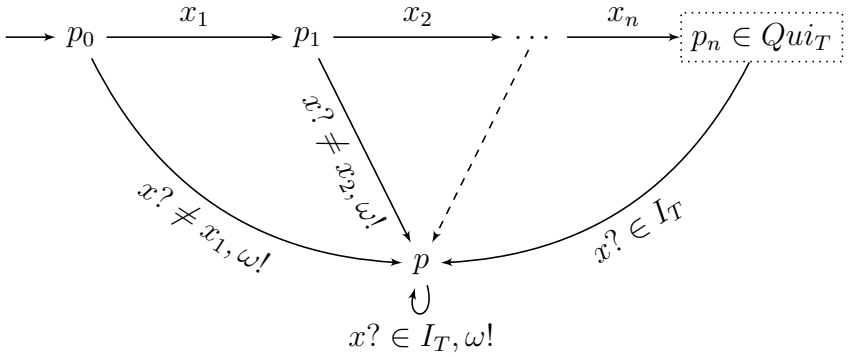

Abbildung 4.2: $x? \neq x_j$ steht für alle $x \in I_T \backslash \{x_j\}$, p_n ist der einzige stille Zustand

Falls das betrachtete w dem leeren Wort ε entspricht, reduziert sich der ω-Partner Test T auf den Zustand $p_n = p_0$ und den Zustand p. Es ist also in diesem Fall der Startzustand gleich dem stillen Zustand.
Allgemein ist der Zustand p_n aus T der einzige stille Zustand in T. Es gilt wegen des ersten Punkte von Lemma 3.8, dass auch in der Parallelkomposition $P_1' \| T$ ein stiller Zustand mit w erreicht wird. Da es sich bei allen in w befindlichen Aktionen um synchronisieren Aktionen handelt und $I_T \cap I = \emptyset$ gilt, folgt $w \in O_{P_1' \| T}$ und $w \in StQT(P_1' \| T)$. Es kann also in der Parallelkomposition durch w ein stiller Zustand lokal erreicht werden. Da $w \notin EDT_{P_1'}$, kann auf dem Weg, der mit w im Transitionssystem P_1' zurück gelegt wird, kein Fehler- oder Divergenz-Zustand lokal erreicht werden. Es kann weder von P_1' noch von T ein Fehler oder Divergenz auf diesem Weg geerbt werden oder neu entstehen. Da ein stiller Zustand in $P_1' \| T$ lokal erreichbar ist, erfüllt P_1 den Test T nicht. Es muss also auch eine as-Implementierung P_2' von P_2 geben, für die $P_2' \| T$ einen lokal erreichbaren

fehlerhaften Zustand besitzt. Hier kann jedoch zunächst keine Aussage darüber getroffen werden, ob das w in der Parallelkomposition $P_2' \| T$ ausführbar ist und ob es sich bei der Fehlerhaftigkeit um Fehler, Stille oder Divergenz handelt.

- Fall a) ($\varepsilon \in ET(P_2' \| T)$): Der lokal erreichbare fehlerhafte Zustand ist ein Fehler. Das w muss somit nicht ausführbar sein. Der Fehler kann sowohl von P_2' geerbt sein, wie durch fehlende Synchronisations-Sicherstellung als neuer Fehler in der Parallelkomposition entstanden sein. Da nur auf dem Trace w in T Synchronisations-Probleme auftreten können und wegen den Fällen 2i) und 2ii) des ersten Punktes dieses Beweises ist ein Präfix von w in $EDT_{P_2'}$ enthalten. Da die Menge EDT unter cont abgeschlossen ist und 4.6.2 gilt, folgt $w \in EDT_2 \subseteq QDT_2$.

- Fall b) ($\varepsilon \in DT(P_2' \| T) \backslash ET(P_2' \| T)$): Es handelt sich bei dem lokal erreichbaren fehlerhaften Zustand um Divergenz. Die Divergenz muss von P_2' geerbt sein, da T keine Möglichkeit für eine unendliche τ-Folge hat. Es gilt also, dass bereits in P_2' ein Präfix von w in $EDT_{P_2'}$ enthalten ist, wegen Fall 2iii) des Beweises des ersten Punktes dieses Lemmas. Mit dem Abschluss unter cont und 4.6.2 folgt, dass auch $w \in EDT_2 \subseteq QDT_2$ gilt.

- Fall c) (stiller Zustand lokal erreichbar in $P_2' \| T$ und $\varepsilon \notin EDT(P_2' \| T)$): Da in T nur durch w ein stiller Zustand erreicht werden kann, muss es sich bei dem lokal erreichbaren stillen Zustand in $P_2' \| T$ um einen handeln, der mit w erreicht werden kann. Mit dem zweiten Punkt von Lemma 3.8 kann gefolgert werden, dass auch in P_2' ein stiller Zustand mit w erreichbar sein muss, da $ET(P_2' \| T)$ eine Teilmenge von $EDT(P_2' \| T)$ ist. Es gilt also $w \in StQT_{P_2'} \subseteq QDT_{P_2'}$. Mit dem dritten Punkt von Proposition 4.6 folgt daraus $w \in QDT_2$.

Nun wird mit dem letzten Punkt des Beweises begonnen. Analog wie in den Beweisen zu den Lemmata 2.15 und 3.11 ist hier aufgrund der bereits geführten Beweisteile nur noch $L_1 \backslash EDT_1 \subseteq EDL_2$ zu zeigen. Es wird also für ein beliebig gewähltes $w \in L_1 \backslash EDT_1$ gezeigt, dass es auch in EDL_2 enthalten ist. Es gilt auch $w \in L_{P_1'} \backslash EDL_{P_1'}$ für eine as-Implementierung P_1' von P_1, wegen 1.8 und 4.6.

- Fall 1 ($w = \varepsilon$): Analog zu den Lemmata 2.15 und 3.11 gilt auch hier, dass ε immer in EDL_2 enthalten ist.

- Fall 2 ($w = x_1 \dots x_n$ mit $n \geq 1$): Die Konstruktion des ω-Partner Tests T weicht nur durch die ω-Transitionen vom Transitionssystem aus dem Beweis der Inklusion der Fehler-gefluteten Sprache EL aus Lemma 2.15 ab. Der Test T ist dann wie folgt definiert (siehe dazu auch Abbildung 4.3):

  - $T = \{p_0, p_1, \dots, p_n, p\}$,

  - $p_{0T} = p_0$,

- $\dashrightarrow_T = \longrightarrow_T = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j < n\}$
  $\cup \{(p_j, x, p) \mid x \in I_T \backslash \{x_{j+1}\}, 0 \leq j < n\}$
  $\cup \{(p_j, \omega, p) \mid 0 \leq j \leq n\}$
  $\cup \{(p, x, p) \mid x \in I_T \cup \{\omega\}\},$
- $E_T = \{p_n\}$.

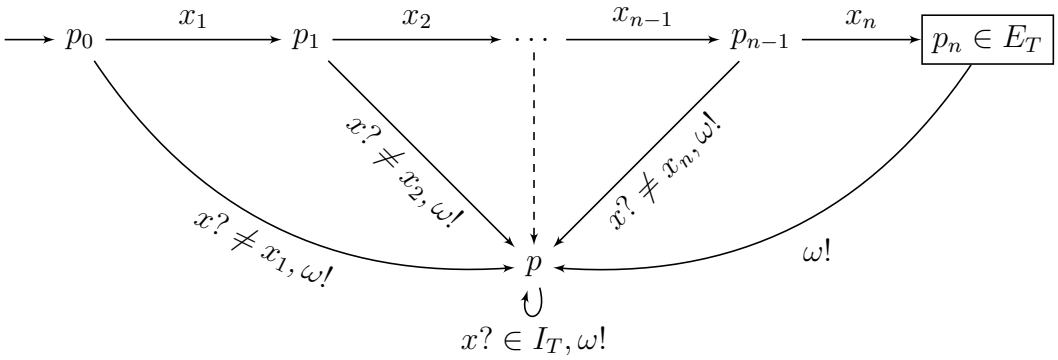

Abbildung 4.3: $x? \neq x_j$ steht für alle $x \in I_T \backslash \{x_j\}$, p_n ist der einzige Fehler-Zustand

Durch die ω-Transition an den Zuständen wird wie oben vermieden, dass es in einer Komposition mit T und auch in T selbst stille Zustände geben kann. Da $p_{01} \stackrel{w}{\Longrightarrow}_{P_1'} p_1'$ gilt, kann man schließen, dass $P_1' \| T$ einen lokal erreichbaren geerbten Fehler hat. Es muss also auch eine as-Implementierung P_2' von P_2 geben, sodass $P_2' \| T$ einen lokal erreichbaren fehlerhaften Zustand hat. Wie oben bereits erwähnt, kommt Stille als Fehlerhaftigkeit nicht in Frage.

- Fall 2a) $\Big($neuer Fehler aufgrund von $x_i \in O_T \backslash \{\omega\}$ und $p_{02} \stackrel{x_1 \ldots x_{j-1}}{\Longrightarrow} p_2' \stackrel{x_j}{\not\longrightarrow}\Big)$: Es gilt $x_1 \ldots x_j \in MIT_{P_2'}$ und somit $w \in EDL_{P_2'}$ Anzumerken ist, dass nur auf diesem Weg Outputs von T aus der Menge $\text{Synch}(P_2', T)$ möglich sind, deshalb gibt es keine anderen Outputs von T, die zu einem neuen Fehler führen könnten. Wegen 4.6 gilt $w \in EDL_2$.

Die restlichen Fälle sind analog zu Lemma 2.15 möglich. Somit gilt für alle Fälle (2a) bis 2d)), dass w in EDL_2 enthalten ist, da $EL_2 \subseteq EDL_2$ gilt.

- Fall 2e) (Divergenz und kein neuer Fehler): Da T keine Möglichkeit hat zu divergieren, muss diese Möglichkeit von P_2' geerbt sein. Es gilt dann $p_{02} \stackrel{x_1 \ldots x_j u}{\Longrightarrow}_{P_2'} p' \in Div_{P_2'}$ für $j \geq 0$ und $u \in O^*$. Somit ist $x_1 \ldots x_j u \in StDT_{P_2'}$ und damit $\text{prune}(x_1 \ldots x_j u) = \text{prune}(x_1 \ldots x_j) \in PrDT_{P_2'} \subseteq EDT_{P_2'}$. Also folgt mit 4.6.2, dass w in $EDT_2 \subseteq EDL_2$ enthalten ist, da DT unter cont abgeschlossen ist.

□

Satz 4.12. *Falls $P_1 \sqsubseteq_{Div} P_2$ gilt folgt daraus auch, dass P_1 P_2 Divergenz-verfeinert.*

Beweis. Nach Definition gilt $w \in QDT(P)$ mit $w \in O^*$ genau dann, wenn in P ein Fehler-, stiller oder Divergenz-Zustand lokal erreichbar ist. $P_1 \sqsubseteq_{Div} P_2$ impliziert, dass $w \in QDT_2$ gilt, wenn $w \in QDT_1$ gilt. Somit ist ein Fehler-, stiller oder Divergenz-Zustand nur dann in P_1 lokal erreichbar, wenn auch ein solcher in P_2 lokal erreichbar ist. Wegen Korollar 4.7 gibt es dann auch as-Implementierungen P_1' und P_2' von P_1 bzw. P_2, die analoge fehlerhafte Zustände lokal erreichen. Die Parallelkomposition $P_j' \| T$ für $j \in \{1, 2\}$ und einen Test T haben dann ebenfalls einen lokal erreichbaren Fehler-, stille oder Divergenz-Zustand. Der fehlerhafte Zustand einer as-Implementierung P_1' tritt in der Parallelkomposition mit T auf jeden Fall auf, wenn es sich um einen Fehler oder Divergenz handelt. Bei Stille zeigt sich das Fehlverhalten nur bei Tests T, die einen analogen stillen Zustand enthalten. Mit der Relation $\sqsubseteq_{Div}$ gibt es auch in P_2 und somit auch in einer as-Implementierung P_2' davon einen lokal erreichbaren fehlerhaften Zustand, wenn es in P_1 einen solchen gibt. In Parallelkompositionen von P_2' mit Tests T zeigt sich das Fehlverhalten ebenfalls. Falls ein Fehler oder Divergenz lokal erreichbar ist, gilt $\varepsilon \in EDT_{P_j'}$. Für alle Tests T ist ε ein Element der Sprache es gilt also $\varepsilon \in EDL_T$. Mit Satz 4.8.1 folgt auch $\varepsilon \in EDT_{P_j' \| T}$. Falls es sich in P_2 und somit auch in P_2' nur um Stille handelt und kein Fehler oder Divergenz lokal erreichbar ist, dann muss auch bereits in P_1 bzw. P_1' Stille das einzige Fehlverhalten gewesen sein, dass lokal erreichbar war. $\sqsubseteq_{Div}$ hätte ansonsten auch andere fehlerhafte Zustände in P_2 gefordert. Es ist also die Argumentation aus Satz 3.12 anwendbar. Es gilt also $\neg P_1 \, sat_{\text{as}}^{Div} \, T \Rightarrow \neg P_2 \, sat_{\text{as}}^{Div} \, T$ für alle Test T. Daraus ergibt sich für alle Test T die Implikation $P_2 \, sat_{\text{as}}^{Div} \, T \Rightarrow P_2 \, sat_{\text{as}}^{Div} \, T$ und somit Divergenz-verfeinert $P_1 \; P_2$. □

Es wurde, wie in den letzten beiden Kapiteln, eine Kette an Folgerungen gezeigt, die sich zu einem Ring schließt. Jedoch ändert sich an der Begründung für einen der Folgepfeile etwas, da in Lemma 4.11 T kein Partner ist, sondern ein ω-Partner. Die Folgerungskette ist in Abbildung 4.4 dargestellt.

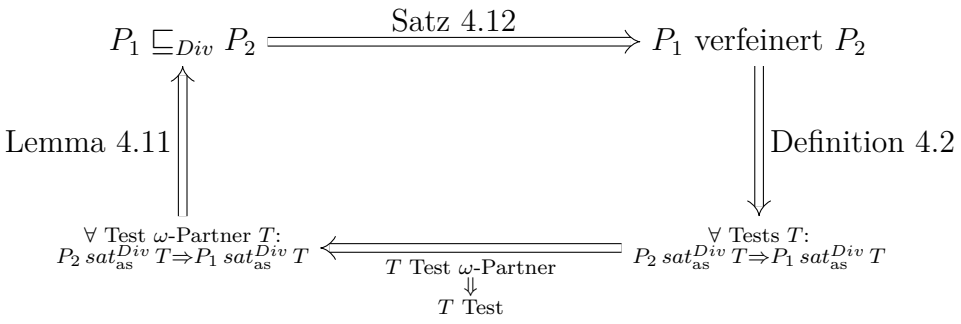

Abbildung 4.4: Folgerungskette der Testing-Verfeinerung und Divergenz-Relation

4.2 Zusammenhänge

Satz 4.13 (*Zusammenhang der Verfeinerungs-Relationen mit der Divergenz-Relation*). *Für zwei MEIOs P und Q gilt $P \sqsubseteq_{\text{as}} Q \Rightarrow P \sqsubseteq_{Div} Q$. Aus keiner der anderen bisher erwähnten Relationen folgt $\sqsubseteq_{Div}$ und auch aus der Relation $\sqsubseteq_{Div}$ kann keine der anderen Relationen gefolgert werden.*

Beweis.
$P \sqsubseteq_{\text{as}} Q \Rightarrow P \sqsubseteq_{Div} Q$:
Um diese Implikation zu zeigen, muss nachgewiesen werden, dass eine beliebige as-Verfeinerung-Relation $\mathcal{R}$ zwischen P und Q auch die Eigenschaften der Relation $\sqsubseteq_{Div}$ erfüllt. Da $\mathcal{R}$ eine as-Verfeinerungs-Relation zwischen P und Q ist, muss $p_0 \mathcal{R} q_0$ gelten. Es sind die folgenden Punkte nachzuweisen:

- $EDT_P \subseteq EDT_Q$,
- $QDT_P \subseteq QDT_Q$,
- $EDL_P \subseteq EDL_Q$.

Die Menge EDT ist unter cont abgeschlossen. Es reicht also für den ersten Punkt zu zeigen, dass ein beliebiges präfix-minimales w aus EDT_P auch in EDT_Q enthalten ist. Das Wort w kann ein Element aus ET_P oder DT_P sein. Für w aus ET_P folgt mit Satz 2.17 bereits $w \in ET_Q$. Es kann also im folgenden davon ausgegangen werden dass w ein präfix-minimales Elemente aus DT_P ist. Es gibt in P einen Trace der Form: $\exists w' \in \Sigma^*, \exists p_1, p_2, \ldots, p_n, \exists \alpha_1, \alpha_2, \ldots, \alpha_n : \hat{w'} = w \wedge w' = \alpha_1 \alpha_2 \ldots \alpha_n \wedge p_0 \overset{\alpha_1}{\dashrightarrow}_P p_1 \overset{\alpha_2}{\dashrightarrow}_P \ldots p_{n-1} \overset{\alpha_n}{\dashrightarrow}_P p_n \in Div_P$. Mit 1.3.3 muss dieser Trace auch in Q möglich sein oder ein Präfix von w muss zu einem Zustand in E_Q führen. Falls es einen Zustand q_j mit $q_j \in E_Q$ und $p_j \mathcal{R} p_j$ gibt für $j \in \{1, 2, \ldots, n\}$, dann gilt $w \in ET_Q \subseteq EDT_Q$. Es kann für den restlichen Beweis davon ausgegangen werden, dass der Trace in Q ausführbar ist. Es gilt also $q_0 \overset{\alpha_1}{\dashrightarrow}_Q q_1 \overset{\alpha_2}{\dashrightarrow}_Q \ldots q_{n-1} \overset{\alpha_n}{\dashrightarrow}_Q q_n$ mit $(p_j, q_j) \in \mathcal{R}$ für $0 \leq j \leq n$. Von p_n ist eine unendliche Folge an τs ausführbar. Die Definition 1.3.3 erzwingt, dass entweder ein Zustand von q_n mit τ aus erreichbar ist, der in E_Q enthalten ist oder von q_n aus ebenfalls eine unendliche Folge an τs ausführbar ist, da die Transitionen von P Transitionen in Q entsprechen müssen. Es gilt also entweder $w \in ET_Q \subseteq EDT_Q$ oder $q_n \in Div_Q$ und somit $w \in DT_Q \subseteq EDT_Q$.

Da im ersten Punkt bereits die EDT-Inklusion bewiesen wurde, reicht es für den zweiten Punkt aus zu zeigen, dass $StQT_P \backslash EDT_P \subseteq QDT_Q$ gilt. In Satz 3.13 wurde bereits die Inklusion $StQT_P \backslash ET_P \subseteq QET_Q$ gezeigt unter der Voraussetzung, dass es eine schwache as-Verfeinerungs-Relation zwischen P und Q gibt. $\mathcal{R}$ ist nicht nur eine as-Verfeinerungs-Relation sondern auch eine schwache as-Verfeinerungs-Relation, wegen Satz 2.17. Die in 3.13 bewiesene Inklusion gilt also auch hier. Es gilt $ET_P \subseteq EDT_P$ und $QET_Q \subseteq QDT_Q$. Es folgt also die zu zeigende Inklusion aus der bereits bewiesenen.

Für den letzten Punkt kann ebenfalls eine Einschränkung der zu zeigenden Inklusion vorgenommen werden. Es muss also $EDL_P \backslash EDT_P \subseteq EDL_Q$ bewiesen werden. Es gilt $L_P \subseteq EDL_P \backslash EDT_P$. In Satz 2.17 wurde die Inklusion $L_P \subseteq EL_Q$ unter der Voraussetzung einer schwachen as-Verfeinerungs-Relation beweisen. Es gilt also wegen der Implikation $\sqsubseteq_{\text{as}} \Rightarrow \sqsubseteq_{\text{w-as}}$ (Satz 2.17) und wegen der Inklusion $EL_Q \subseteq EDL_Q$ auch die hier nach zu weisende Inklusion.

$P \sqsubseteq_{Div} Q \not\Rightarrow P \sqsubseteq_E Q$:
Diese Implikation gilt nicht, da $\sqsubseteq_{Div}$ nicht zwischen Fehler und Divergenz unterscheiden kann, $\sqsubseteq_E$ hingegen nur Fehler berücksichtigt und Divergenz nicht als Fehlverhalten auffasst. Ein entsprechende Gegenbeispiel ist in Abbildung 4.5 dargestellt. Die Sprachen beider Transitionssysteme sind gleich und bestehen nur aus dem leeren Wort. Für beide Systeme besteht die Menge EDT aus allen Wörtern, die über dem Alphabet Σ möglich sind. Es gibt keine Stille-Trace. Somit gilt also $P \sqsubseteq_{Div} Q$.
Für $P \sqsubseteq_E Q$ müsste $ET_P \subseteq ET_Q$ gelten. Jedoch ist die Menge ET_Q leer, da Q keine Fehler-Zustände enthält und vorausgesetzt werden kann, das I leer ist und es keine Input-kritischen Traces gegen kann. Für P ist jedoch der Startzustand ein Fehler-Zustand und somit entspricht ET_P der Menge Σ^*. Die geforderte Inklusion gilt nicht und somit kann auch die Relation $\sqsubseteq_E$ zwischen P und Q nicht gelten.

Abbildung 4.5: Gegenbeispiel zu $\sqsubseteq_{Div} \Rightarrow \sqsubseteq_E$ mit $I_P = I_Q = \emptyset$

$P \sqsubseteq_{Div} Q \not\Rightarrow P \sqsubseteq_{Qui} Q$, $P \sqsubseteq_{Div} Q \not\Rightarrow P \sqsubseteq_{\text{w-as}} Q$ und $P \sqsubseteq_{Div} Q \not\Rightarrow P \sqsubseteq_{\text{as}} Q$:
Falls eine dieser Implikationen gelten würde, würde mit den bereits in den Sätzen 2.17 und 3.13 bewiesenen Implikationen und der Transitivität von Implikationen folgen, dass auch $P \sqsubseteq_E Q$ gilt. Dies stellt ein Widerspruch zum letzten Punkt dieses Beweises dar. Es kann also keine der Implikationen gelten.

$P \sqsubseteq_{\text{w-as}} Q \not\Rightarrow P \sqsubseteq_{Div} Q$:
Das Problem dieser Implikation beruht darauf, dass eine schwache as-Verfeinerungs-Relation es zulässt, dass τ-Transitionen schwach gematched werden. Es ist also möglich, eine unendliche τ-Folge ohne eine einzige Transition zu matchen. Das Gegenbeispiel in Abbildung 4.6 verdeutliche dies. P und Q stehen in der schwachen as-Verfeinerungs-Relation $\mathcal{R}$, die nur aus dem Tupel (p_0, q_0) besteht. Die Transitionssysteme bestehen nur aus dem Startzustände, die beide keine Fehler-Zustände sind. Q besitzt keine Transitionen und P nur eine Transition für eine interne Aktion. Es sind also die Punkt 1. bis 4. der Definition 1.4 für $\mathcal{R}$ bereits erfüllt. Der 5. Punkt fordert, die schwache Transition $q_0 \stackrel{\hat{\tau}}{\Rightarrow}_Q q$ mit $(p_0, q) \in \mathcal{R}$ für ein q aus Q. $\hat{\tau}$ entspricht ε somit ist q_0 eine zulässige Wahl für q. $\mathcal{R}$ erfüllt also auch den letzten Punkt der Definition 1.4 und ist deshalb eine as-

Verfeinerungs-Relation.
Q enthält keine Fehler- oder Divergenz-Zustände. Es gilt also $EDT_Q = \emptyset$. Für P ist jedoch der Startzustand ein divergenter Zustand. Es gilt also $\varepsilon \in StDT_P \subseteq EDT_P$. Die Inklusion $EDT_P \subseteq EDT_Q$, die für die Relation $\sqsubseteq_{Div}$ zwischen P und Q notwendig wäre, ist also nicht erfüllt.

Abbildung 4.6: Gegenbeispiel zu $\sqsubseteq_{\text{w-as}} \Rightarrow \sqsubseteq_{Div}$

$P \sqsubseteq_E Q \not\Rightarrow P \sqsubseteq_{Div} Q$ und $P \sqsubseteq_{Qui} Q \not\Rightarrow P \sqsubseteq_{Div} Q$:
Mit den Implikationen, die bereits in Satz 3.13 bewiesen wurde, kann gefolgert werden, dass das im letzten Punkt angegebenen Beispiel aus für diese Implikationen anwendbar ist. Es galt $P \sqsubseteq_{\text{w-as}} Q$ somit gilt auch $P \sqsubseteq_{Qui} Q$ und $P \sqsubseteq_E Q$. Die Relation $\sqsubseteq_{Div}$ hingegen ist zwischen P und Q nicht erfüllt. Es folgt also, dass die hier angegeben Implikationen ebenfalls nicht gelten.

□

Literaturverzeichnis

[BFLV16] Ferenc Bujtor, Sascha Fendrich, Gerald Lüttgen, und Walter Vogler, *Nondeterministic 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.