Universität Augsburg
Fakultät für Angewandte
Informatik

Masterarbeit
im Studiengang Master Informatik

Stillstand, Divergenz und modale Spezifikation

Universität Augsburg
Fakultät für Angewandte Informatik
Theorie verteilter Systeme

Aufgabensteller: Prof. Dr. Walter Vogler

von: Ayleen Schinko
Stand: 11. Oktober 2017

Inhaltsverzeichnis

Einleitung

Interfaces werden oft verwendet um komplexe Systeme zu entwerfen. Dadurch kann bereits während des Designs überprüft werden, wie gut die einzelnen Komponenten zusammenpassen. Durch Interfaces können insbesondere auch nebenläufige Systeme komponentenweise modelliert werden. Einige der Theorien über Interfaces basieren auf den Interface Automaten (IA) aus [dAH05]. Dort ist eine Parallelkomposition auf LTS (Labelled Transition Systems) mit Inputs und Outputs der charakterisierende Punkt. Wenn ein unerwarteter Input empfangen wird, wird dieser als Fehler aufgefasst, d.h. es kommt zu einem Fehler in der Kommunikation der Systeme. In so genannten pessimistischen Ansätzen wie in [BMSH10] ist die Komposition zweier Systemen nicht definiert, wenn ein solcher Fehler durch ihre Kommunikation entstehen würde. In optimistischen Ansätzen, wie z.B. in [LV13] und [BFLV16], wird ein Kommunikationsfehler als akzeptabel angesehen, solange eine Umgebung verhindert kann, dass der Fehler erreicht wird. In dieser Arbeit kommt die soeben beschriebene optimistische Sichtweise zum Einsatz.
Interface Automaten wurden schon in mehreren Veröffentlichungen mit Modalen Transitions Systemen (MTS) [Lar89] kombiniert. Die Kombination die hier als Grundlage dient sind die Modalen Interface Automaten (MIA) aus [LV13] und [BFLV16].
Diese Kombination aus IA und MTS ergibt eine modale Spezifikations-Form. Die Modalitäten geben einem dabei die Freiheit Forderungen an potentielle Implementierungen zu stellen. Die Forderungen können die Spezifikation von Verhalten, das zwingendermaßen umgesetzt werden muss, und den erlaubten Spielraum betreffen. Die hier verwendeten modalen Spezifikationen sind spezielle Transitionssysteme. Must-Transitionen stellen dabei in einer Spezifikation das erzwungene und may-Transitionen das erlaubte Verhalten dar.
Interfaces spezifizieren durch nicht vorhandene Transitionen auch Anforderungen an die Umgebung. Falls ein System an einem Zustand eine Input-Transition nicht umsetzt, stellt dies die Forderung an die Umgebung, dass dieser Input in diesem Zustand nicht empfangen werden sollte. Ansonsten tritt wie bereits beschrieben in einer Kommunikation zwischen der Komponente und ihrer Umgebung ein Fehler auf. Implementierungen können jedoch auch bereits Fehler enthalten. In diesem Fall wird die Anforderung an die Umgebung gestellt, dass dieser Zustand nicht erreicht werden sollte.
Da diese Arbeit die Ideen eines optimistischen Ansatzes bezüglich der Relevanz von Fehlern aufgreift, wird davon ausgegangen, dass die Kommunikation mit einer hilfreichen Umgebung bzw. einem hilfreichen User statt findet. Dadurch sind Systemen mit Fehler nicht automatisch schlecht, sondern man kann optimistisch davon ausgehen, dass Fehler erst ein Problem sind, wenn sie durch lokal kontrollierte Aktionen erreicht werden können. Sobald ein Fehler jedoch durch lokal kontrollierte Aktionen erreichbar ist,

kann selbst eine hilfreiche Umgebung nicht mehr verhindern, dass der Fehler auftritt. Im Gegensatz zu den MIAs aus [LV13] und [BFLV16] sollen die Zustände, von denen aus ein Fehler von einer hilfreichen Umgebung nicht mehr verhindert werden kann, nicht aus der Parallelkomposition entfernt werden. Somit können auch nach der Komposition noch Rückschlüsse auf die Quellen der Fehler gezogen werden. Dies ermöglicht es Spezifikationen gezielt verbessern zu können, falls dies gewünscht ist.
Die MIAs enthalten, laut ihrer Definition, disjunktive must-Transitionen. Dies sind Transitionen die von einem Zustand mit einer Aktion zu einer Menge von alternativen Zielzuständen führen. Falls diese Transition ausgeführt wird, wird ein beliebiger Zustand aus der Zielzustands-Menge erreicht. Diese Art von must-Transitionen werden in dieser Arbeit jedoch nicht betrachtet. Die must- und may-Transitionen sollen hier die gleiche Form besitzen, so dass via einer Transition von einem Zustand immer nur ein Zustand erreicht wird. Dieser Unterschied bezüglich der must-Transitionen wird vor allem im Vergleich der hier verwendeten Transitionssysteme mit den MIAs am Ende des Kapitels 2.4 relevant.
Auf Basis spezieller Testsysteme wie in [BV15b] und [BSV17] wird hier in jedem Kapitel eine beobachtbare Präkongruenz für die unterschiedlichen Fehler-Arten nachgewiesen. Für Kommunikationsfehler wird analog zu [Sch16] zusätzlich begründet, dass die Präkongruenz die gröbste bezüglich einer optimistische Basisrelation.
Die unterschiedlichen Fehler-Arten werden in den Systemen auf Basis von Trace-Mengen analysiert. Dieser Ansatz orientiert an [BV15a] und [Sch16].
Die hier betrachteten Fehler-Arten sind am Anfang nur die bereits oben beschriebenen Kommunikationsfehler, auch genannt Fehler. Stillstand ist die erste Erweiterung in der Betrachtung des Fehlverhaltens von Systemen. Eine Komponente hat einen Stillstand erreicht, wenn es keinen Fortschritt mehr gibt ohne zutun von außen. Es handelt sich dabei also um eine Art Deadlock. Die dritte Fehler-Art, die betrachtet wird, ist Divergenz. Dabei kann das jeweilige System zwar immer etwas tun, jedoch ist eine Kommunikation nach außen hin nicht mehr sichergestellt. Divergenz kann somit auch als Art Livelock aufgefasst werden. Die betrachteten Fehler-Arten entsprechen denen aus [Sch16]. Jedoch wurden dort Interface Automaten mit Fehler-Zuständen betrachtet. Somit lassen sich die Definitionen aus [Sch16] nicht einfach übernehmen, sondern können nur als Grundideen für die Erweiterung auf modale Systeme verwendet werden. Die Ideen, die sich daraus ergeben, werden in den einzelnen Kapitel überprüft.

1 Grundlagen

1.1 Definitionen

Das Definitions-Kapitel wurde auf Grundlage von [BV15b] und [Sch16] zusammengestellt und ist teilweise von Ideen aus [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 MEIOs 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 hingegen sind 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 Menge der Zustände (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 *Ablaufes* $p \overset{\alpha_1}{\dashrightarrow} p_1 \overset{\alpha_2}{\dashrightarrow} \dots p_{n-1} \overset{\alpha_n}{\dashrightarrow} p'$ ($p \overset{\alpha_1}{\dashrightarrow} p_1 \overset{\alpha_2}{\dashrightarrow} \dots p_{n-1} \overset{\alpha_n}{\dashrightarrow}$) 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 sollen $p \overset{a}{\not\longrightarrow}$ und $p \overset{a}{\not\dashrightarrow}$ für $\neg\exists p' : p \overset{a}{\longrightarrow} p'$ bzw. $\neg\exists p' : p \overset{a}{\dashrightarrow} p'$ 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 (gestrichelter) Pfeil gezeichnet. 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:*

- $I = (I_1 \cup I_2) \backslash (O_1 \cup O_2)$,
- $O = (O_1 \cup O_2)$,
- $$\begin{aligned}\longrightarrow_{12} = &\left\{((p_1,p_2),\alpha,(p_1',p_2)) \mid p_1 \overset{\alpha}{\longrightarrow}_1 p_1', \alpha \in \Sigma_\tau \backslash \mathrm{Synch}(P_1,P_2)\right\} \\ &\cup \left\{((p_1,p_2),\alpha,(p_1,p_2')) \mid p_2 \overset{\alpha}{\longrightarrow}_2 p_2', \alpha \in \Sigma_\tau \backslash \mathrm{Synch}(P_1,P_2)\right\} \\ &\cup \left\{((p_1,p_2),\alpha,(p_1',p_2')) \mid p_1 \overset{\alpha}{\longrightarrow}_1 p_1', p_2 \overset{\alpha}{\longrightarrow}_2 p_2', \alpha \in \mathrm{Synch}(P_1,P_2)\right\},\end{aligned}$$
- $$\begin{aligned}\dashrightarrow_{12} = &\left\{((p_1,p_2),\alpha,(p_1',p_2)) \mid p_1 \overset{\alpha}{\dashrightarrow}_1 p_1', \alpha \in \Sigma_\tau \backslash \mathrm{Synch}(P_1,P_2)\right\} \\ &\cup \left\{((p_1,p_2),\alpha,(p_1,p_2')) \mid p_2 \overset{\alpha}{\dashrightarrow}_2 p_2', \alpha \in \Sigma_\tau \backslash \mathrm{Synch}(P_1,P_2)\right\} \\ &\cup \left\{((p_1,p_2),\alpha,(p_1',p_2')) \mid p_1 \overset{\alpha}{\dashrightarrow}_1 p_1', p_2 \overset{\alpha}{\dashrightarrow}_2 p_2', \alpha \in \mathrm{Synch}(P_1,P_2)\right\},\end{aligned}$$
- $$\left.\begin{aligned}E = &(P_1 \times E_2) \cup (E_1 \times P_2) \\ &\cup \left\{(p_1,p_2) \mid \exists a \in O_1 \cap I_2 : p_1 \overset{a}{\dashrightarrow}_1 \wedge p_2 \overset{a}{\not\dashrightarrow}_2\right\} \\ &\cup \left\{(p_1,p_2) \mid \exists a \in I_1 \cap O_2 : p_1 \overset{a}{\not\dashrightarrow}_1 \wedge p_2 \overset{a}{\dashrightarrow}_2\right\}\end{aligned}\right\} \begin{array}{l}\text{geerbte Fehler} \\ \\ \text{neue Kommunikationsfehler.}\end{array}$$

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 sicherstellt (keine must-Transition vorhanden). Es muss also in der Kommunikation möglichen Implementierungen nicht wirklich zu diesem Fehler kommen, da die Output-Transition nicht zwingendermaßen implementiert werden muss und die may-Input-Transition trotzdem umgesetzt 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 zu synchronisierenden Input sicherstellt, der synchronisiert wird, wird dieser Input in der Parallelkomposition nicht mehr garantiert. Es kann also in der Kommunikation mit einem weiteren MEIO an dieser Stelle 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* $q \xrightarrow{\alpha}_Q q'$*, gibt es eine Transition* $p \xrightarrow{\alpha}_P p'$ *für ein* p' *mit* $p' \mathcal{R} q'$*,*
3. *falls* $p \overset{\alpha}{\dashrightarrow}_P p'$*, gibt es eine Transition* $q \overset{\alpha}{\dashrightarrow}_Q q'$ *für ein* q' *mit* $p' \mathcal{R} q'$*.*

Für die Vereinigung aller dieser Simulations-Relationen wird $\sqsubseteq_{\text{as}}$ *geschrieben und* $\sqsubseteq_{\text{as}}$ *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 *bzw. 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*. 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 $q \notin E_Q$ $p \notin E_P$ folgt, gilt auch die Implikation $p \in E_P \Rightarrow q \in E_Q$.
Für LTS, 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. Man muss dazu einen Äquivalenz Begriff wie z.B. in [Mil89] verwenden.

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, das 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 keinen Fehler enthält, aber potentiell zusätzliches Verhalten aufweist. Hierzu sollte auch das Beispiel aus Abbildung 1.1 beachtet werden. Die Implementierung P ist eine Verfeinerung der Implementierung Q und zwischen ihnen erfüllt die as-Verfeinerungs-Relation $\mathcal{R} = \{(p_0, q_0)\}$ die Definition 1.3. P verfeinert den Fehler-Zustand von Q nicht mit einem Fehler sondern durch die o Schlinge.
Falls eine zusammenhängende Implementierung, in der jeder Zustand vom Startzustand aus erreichbar ist, bereits frei von Fehler-Zuständen ist, entspricht sie einem LTS und kann somit nur noch durch äquivalente Implementierungen „verfeinert“ werden.

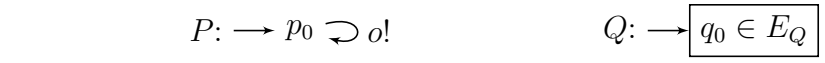

Abbildung 1.1: Beispiel für die Verfeinerung von Implementierungen

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 alle $i \in I$, $\omega \in O \cup \{\tau\}$ gilt:*

1. *$p \notin E_P$,*
2. *falls $q \xrightarrow{i}_Q q'$, gibt es eine Transition $p \xrightarrow{i}_P \overset{\varepsilon}{\Longrightarrow}_P p'$ für ein p' mit $p' \mathcal{R} q'$,*
3. *falls $q \xrightarrow{\omega}_Q q'$, gibt es eine Transition $p \overset{\hat{\omega}}{\Longrightarrow}_P p'$ für ein p' mit $p' \mathcal{R} q'$,*
4. *falls $p \overset{i}{\dashrightarrow}_P p'$, gibt es eine Transition $q \overset{i}{\dashrightarrow}_Q = \overset{\varepsilon}{\Longrightarrow}_Q q'$ für ein q' mit $p' \mathcal{R} q'$,*
5. *falls $p \overset{\omega}{\dashrightarrow}_P p'$, 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 den Begriff schwache as-Verfeinerungen *definieren.*
Für ein MEIO Q und eine Implementierung P mit $P \sqsubseteq_{\text{w-as}} Q$, ist P eine w-as-Implementierung *von Q. Es wird* w-as-impl(Q) *für die Menge aller w-as-Implementierungen von Q verwendet.*

Die schwache Simulation erlaubt interne Aktionen des MEIOs, 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 dann 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 jedoch 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:* $W_1 \| W_2 := \bigcup \{w_1 \| w_2 \mid w_1 \in W_1 \wedge w_2 \in W_2\}$.

Die Wörter 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 Inputs und Outputs. Es existiert in einer Parallelkomposition eine Transitionsfolge vom Startzustand aus, die mit den Aktionen aus $w \in w_1 \| w_2$ beschriftet ist. Ein Trace aus der Menge $w_1 \| w_2$ der Parallelkomposition kann also auch als Wort aufgefasst werden, das verarbeitet wird während des Ablaufs des Systems.

Die Definitionen der Funktionen prune und cont zum Abschneiden und Verlängern von Traces können ebenso wie die vorangegangene 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 Sprache wie sie in [Sch16] für EIOs definiert ist. 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 seiner möglichen as-Implementierungen.
Von der Sprache einer as-Verfeinerung eines MEIOs kann man im Allgemeinen nur wenig

Rückschlüsse auf die ursprüngliche Sprache ziehen, da man nicht weiß, welche Fehler-Zustände in die 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.

Im folgenden soll ein Operator definiert werden, um Aktionen zu verbergen. Dieses Vorgehen wird Hiding genannt und orientiert sich an den Definitionen aus [BFLV16] und [Sch16]. Da die Outputs vom jeweiligen System kontrolliert sind, werden diese durch das Verbergen als interne Aktionen, nach außen nicht mehr sichtbar ausgeführt. Inputs hingeben werden ausgeführt, wenn die Umgebung das passende Signal sendet. Wenn man Inputs verbergen würde, schließt man dadurch den Kommunikationskanal und blockiert die Kommunikation über die entsprechenden Aktionen mit anderen Systemen. Das Verbergen von Inputs verbietet die entsprechend beschrifteten Transitionen, ähnlich wie bei der Anwendung von Restriktionen in CCS z.B. in [Mil89]. Das Verbergen von Inputs wird in [BFLV16] auch Restriktion genannt. Dies soll hier jedoch nicht betrachtet werden. Der Hiding-Operator wird hier somit nur für Output-Aktionen definiert wie z.B. in [Sch16].

Definition 1.8 (*Hiding-Operator*). *Für einen MEIO* $P = (P, I_P, O_P, \longrightarrow_P, \dashrightarrow_P, p_{0_P}, E_P)$ *und eine Menge* $X \subseteq O$ *ist* P/X*, mit dem Hiding-Operator* $\cdot/\cdot$*, definiert als der MEIO* P/X *mit:*

- $P/X = P$, $I_{P/X} = I_P$, $p_{0_{P/X}} = p_{0_P}$ *und* $E_{P/X} = E_P$,
- $O_{P/X} = O_P \backslash X$,
- $\longrightarrow_{P/X} = (\longrightarrow_P \cup \{(p, \tau, p') \mid (p, o, p') \in \longrightarrow_P, o \in X\}) \backslash \{(p, o, p') \mid (p, o, p') \in \longrightarrow_P, o \in X\}$,
- $\dashrightarrow_{P/X} = (\dashrightarrow_P \cup \{(p, \tau, p') \mid (p, o, p') \in \dashrightarrow_P, o \in X\}) \backslash \{(p, o, p') \mid (p, o, p') \in \dashrightarrow_P, o \in X\}$.

Die Transitionen, die mit einem Output beschriftet sind, der in der Menge X der zu verbergenden Aktionen enthalten ist, werden durch das Hiding zu internen Transitionen. Dieses Vorgehen wird im weiteren Verlauf auf als Internalisieren von Aktionen bezeichnet. Die Transitionen sind also weiterhin lokal kontrolliert ausführbar. Auch für schwache must- und may-Transitionen ändert sich nur die Beschriftung, jedoch nicht die Ausführbarkeit.
Häufig werden die synchronisierten Aktionen in einer Parallelkomposition nicht beibehalten, sondern verborgen. Dies wurde für EIO z.B. in [BV15a] praktiziert. Man kann diese Art der Parallelkomposition durch den Hiding-Operator versuchen nachzumachen. Dadurch kann gezeigt werden, dass die späteren Ergebnisse auch für eine Parallelkomposition mit Internalisierung gelten. Der Hiding-Operator ist jedoch nur für Output-

Aktionen definiert. Es können also nur die Outputs, die durch die Synchronisation entstehen, verborgen werden. Jedoch sind Inputs Kommunikationskanäle zu weiteren Systemen. Wenn ein Input durch Synchronisation entsteht, hatten beide Komponenten die Möglichkeit über diesen Input zu kommunizieren. Die beiden Systeme haben in der Parallelkomposition jedoch nicht über diesen Input kommuniziert. Die Kommunikation mit einem weiteren System sollte also auch nicht durch das Verbergen des Inputs unterbunden werden. Die Inputs sollten also auch in einer Definition einer Parallelkomposition mit Internalisierung nur synchronisiert und nicht restringiert werden.

Definition 1.9 (*Parallelkomposition mit Internalisierung*). *Seien P_1 und P_2 komponierbare MEIOs, dann ist die Parallelkomposition mit Internalisierung definiert als $P_1|P_2 := P_{12}/(\text{Synch}(P_1, P_2) \cap O_{12})$.*

1.2 Allgemeine Folgerungen

Proposition 1.10 (*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, muss 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 1.11 (*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, das in L_{12} enthalten ist, hat 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 1.12 (*w-as-Verfeinerung und Parallelkomposition*). *Seien P_1 und P_2 komponierbar MEIOs und P_1' eine schwache as-Verfeinerung von P_1 aufgrund der schwachen as-Verfeinerungs-Relation $\mathcal{R}_1$. Dann gelten die Aussagen 2. bis 5. aus der Definition 1.4 für die Relation $\mathcal{R}_{12} = \{((p_1', p_2), (p_1, p_2)) \mid (p_1', p_1) \in \mathcal{R}_1, p_2 \in P_2\}$.*

Beweis. Für alle folgenden Fälle sei $((p_1', p_2), (p_1, p_2))$ ein beliebig gewähltes Element aus $\mathcal{R}_{12}$ mit $(p_1, p_2) \notin E_{12}$.

2. Aus der Definition der schwachen alternierenden Simulation in 1.4 folgt, dass für diesen Punkt folgendes zu zeigen ist: $(p_1, p_2) \xrightarrow{i}_{12} (q_1, q_2)$ in P_{12} impliziert $(p_1', p_2) \xrightarrow{i}_{P_1' \| P_2} \overset{\varepsilon}{\Longrightarrow}_{P_1' \| P_2} (q_1', q_2)$ in $P_1' \| P_2$ 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. Die Gültigkeit der geforderten Transitionsfolge und das geforderten Elements der Relation $\mathcal{R}_{12}$ folgen 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 $\left(p_1 \xrightarrow{i}_1 q_1\right)$. Mit der Relation $\mathcal{R}_1$ und 1.4.2 folgt, dass in P_1' i als schwache Transition in der Form $p_1' \xrightarrow{i}_{P_1'} \overset{\varepsilon}{\Longrightarrow}_{P_1'} q_1'$ 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 $(p_1', p_2) \xrightarrow{i}_{P_1' \| P_2} \overset{\varepsilon}{\Longrightarrow}_{P_1' \| P_2} (q_1', q_2)$. 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 $p_1' \xrightarrow{i}_{P_1'} \overset{\varepsilon}{\Longrightarrow}_{P_1'} q_1'$ 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 zu 2. Fall 1. 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 $p_1' \xRightarrow{\hat{\omega}}_{P_1'} q_1'$ mit $(q_1', q_1) \in \mathcal{R}_1$ gilt. In der Komposition folgt dann $(p_1', p_2) \xRightarrow{\hat{\omega}}_{P_1'\|P_2} (q_1', q_2)$. 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 $\omega \in I_1$ $p_1' \xrightarrow{\omega}_{P_1'} \xRightarrow{\varepsilon}_{P_1'} q_1'$ 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 $p_1' \xRightarrow{\omega}_{P_1'} q_1'$ 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 sich also die Transitionsfolge $(p_1', p_2) \xRightarrow{\hat{\omega}}_{P_1'\|P_2} (q_1', q_2)$ und das Tupel $((q_1', q_2), (q_1, q_2))$ in der Relation $\mathcal{R}_{12}$.

4. $(p_1', p_2) \xrightarrow{i}_{P_1'\|P_2} (q_1', q_2)$ impliziert $(p_1, p_2) \xrightarrow{i}_{12} \xRightarrow{\varepsilon}_{12} (q_1, q_2)$ 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 $p_1' \xrightarrow{i}_1 q_1'$ gilt. Mit der Relation $\mathcal{R}_1$ und 1.4.4

folgt, dass in P_1 i als schwache Transition der Form $p_1 \overset{i}{\dashrightarrow}_1 = \overset{\varepsilon}{\Rightarrow}_1 q_1$ ausführbar sein muss 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 $(p_1, p_2) \overset{i}{\dashrightarrow}_{12} = \overset{\varepsilon}{\Rightarrow}_{12} (q_1, q_2)$ 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, 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 interne 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 1.13 (*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 1.12 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 1.12 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 $p_2 \overset{a}{\not\rightarrow}_2$ 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 1.2 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 $\mathcal{R}_{12}$.

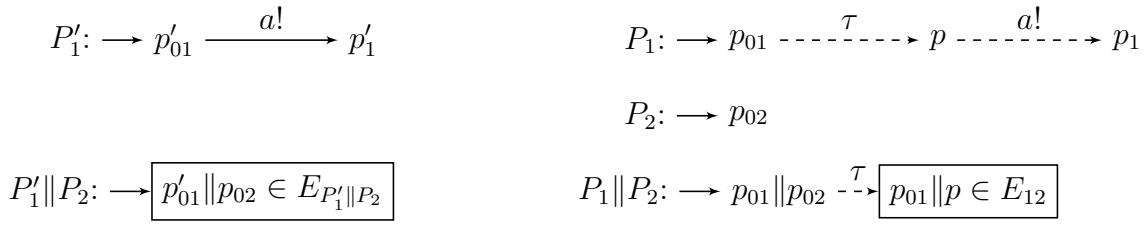

Abbildung 1.2: Gegenbeispiel für 1. von $\mathcal{R}_{12}$ bzgl. Definition 1.4 mit $a \in O_1 \cap I_2$

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 das in Proposition 1.13 beschriebene 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 1.14 (*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 1.12 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 1.12 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 zu Proposition 1.13 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 in beiden Fällen auch $(p_1, p_2) \in E_{12}$ gelten, was ein Widerspruch zur Voraussetzung wäre. (p_1', p_2') kann also weder ein geerbter noch ein neuer Fehler 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 1.12 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. $\mathcal{R}_{12}$ erfüllt die Forderungen für die starke as-Verfeinerungs-Relation dieses Punktes.

3. Hierfür werden die Punkte 3. und 4. aus dem Beweis des Lemmas 1.12 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 für $\mathcal{R}_{12}$.

□

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 1.15 (*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 1.14. 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, so dass $(P_1' \| P_2') \in \text{as-impl}(P_1 \| P_2)$ gilt. □

Für schwache as-Implementierungen kann es kein analoges Korollar zu 1.15 gegen, da bereits die Verfeinerung im allgemeinen scheitert (Proposition 1.13); man beachte auch die Diskussion des Beispiels 1.2. 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 1.14 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 1.3 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.

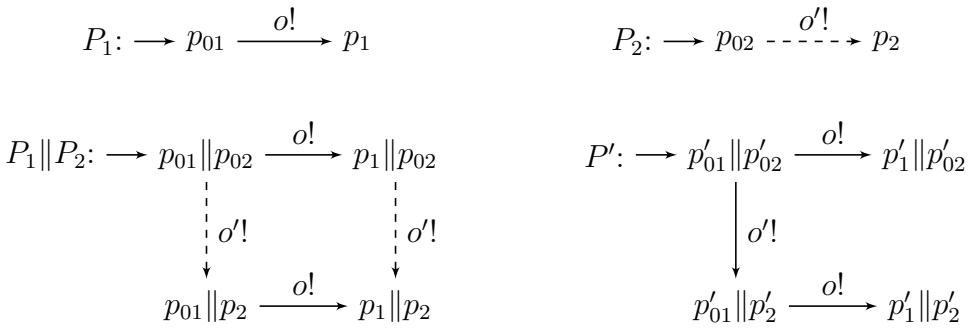

Abbildung 1.3: Gegenbeispiel für Umkehrung von Lemma 1.14

Ein neuer Fehler in einer Parallelkomposition zweier MEIOs muss in einer Implementierung (as oder w-as) dieser Parallelkomposition nicht auftauchen, auch nicht in der Parallelkomposition von Implementierungen der einzelnen Komponenten. Dies liegt daran, dass für den Input nur 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.

Lemma 1.12 und Korollar 1.14 lassen die Vermutung zu, dass es einen Zusammenhang zwischen den beiden Verfeinerungs-Relation $\sqsubseteq_{\text{w-as}}$ und $\sqsubseteq_{\text{as}}$ gibt. Durch Proposition 1.13 fällt jedoch schon auf, dass es Stellen gibt, an denen sich die Relationen unterscheiden. Durch die Definitionen wird klar, das der Unterschied der beiden Relationen in den τ-Transitionen liegt. Die folgende Aussage, dass jede starke as-Verfeinerungs-Relation auch eine schwache ist, die Umkehrung jedoch nicht gilt sollte somit nicht überraschen.

Lemma 1.16 (*Zusammenhang der Verfeinerungs-Relationen*). *Jede starke as-Verfeinerungs-Relation ist auch eine schwache as-Verfeinerungs-Relation. Jedoch ist eine schwache as-Verfeinerungs-Relation im allgemeinen keine starke as-Verfeinerungs-Relation. Es gilt also: $P \sqsubseteq_{\text{as}} Q \Rightarrow P \sqsubseteq_{\text{w-as}} Q$ und im allgemeinen $P \sqsubseteq_{\text{as}} Q \not\Leftarrow P \sqsubseteq_{\text{w-as}} Q$.*

Beweis.
„⇒“:

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 $\alpha \in \Sigma_\tau$ gilt mit 1.3.2, dass $q \xrightarrow{\alpha}_Q q'$ $p \xrightarrow{\alpha}_P p'$ impliziert 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.

„$\not\Leftarrow$“:
Im Abbildung 1.4 wird ein Gegenbeispiel veranschaulicht 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 sind 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 1.4: Gegenbeispiel zu $\sqsubseteq_{\text{as}} \Leftarrow \sqsubseteq_{\text{w-as}}$

□

2 Verfeinerungen für Kommunikationsfehler-Freiheit

2.1 Gröbster Präkongruenz-Ansatz

Dieses Kapitel hat das Ziel die Präkongruenz für Error auf 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 $p_0 \overset{w}{\Rightarrow}_P p \in E$. Ein MEIO P ist* lokal fehler-frei*, wenn $\neg\exists w \in O^* : p_0 \overset{w}{\Rightarrow}_P p \in E$.*
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. $\sqsubseteq_E^{\mathrm{B}}$ 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*:* $StET(P) := \left\{w \in \Sigma^* \mid p_0 \overset{w}{\Rightarrow}_P p \in E\right\}$,
- gekürzte Fehler-Traces*:* $PrET(P) := \{\mathrm{prune}(w) \mid w \in StET(P)\}$,
- Input-kritische-Traces*:* $MIT(P) := \left\{wa \in \Sigma^* \mid p_0 \overset{w}{\Rightarrow}_P p \wedge a \in I \wedge p \overset{a}{\not\rightarrow}_P\right\}$.

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 beschreibt eine Teilmengen-Beziehung zwischen den Traces eines MEIOs und den Traces seiner as-Implementierungen. Die umgekehrte Teilmengen-Beziehung 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 die 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 1.10 alle Transitionen von P. Das P' wird hier jedoch im Gegensatz zu Beweis von 1.10 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 1.3.3 als Elemente der as-Verfeinerungs-Relation $\mathcal{R}$ gefordert werden, sind bereits durch die Identitäts-Relation garantiert, wie im Beweis von 1.10. 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, dass alle strikten Fehler-Traces von P in der Vereinigung aller strikten Fehler-Traces der as-Implementierungen von P enthalten sind. Wenn auf alle Wörter in beiden Mengen die prune-Funktion angewendet wird, gilt die Inklusion der daraus entstanden Mengen weiterhin. Dies entspricht der Behauptung des aktuell betrachteten 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 w 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 zusätzlich 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 die Inklusionen $ET_1 \subseteq ET_2$ und $EL_1 \subseteq EL_2$ gelten.

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

Im weiteren Verlauf wird immer wieder eine Aussage den Zusammenhang von Abläufen in as-Verfeinerungen bzw. schwachen as-Verfeinerungen und ihren Spezifikationen benötigt. Um nicht immer wieder ähnliche Argumentationen zu benötigen, wird diese Aussage nun hier allgemein zunächst für schwache as-Verfeinerungen bewiesen. Die analoge Aussage für starke as-Verfeinerungen folgt dann mit Lemma 1.16.

Lemma 2.6 (*Abläufe in schwachen as-Verfeinerungen und Spezifikationen*). *Sei P eine schwache as-Verfeinerung eines MEIOs Q und $\mathcal{R}$ die schwache 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 $q_0 \overset{\widehat{\alpha_1}}{\Longrightarrow}_Q q_1 \overset{\widehat{\alpha_2}}{\Longrightarrow}_Q \dots q_{n-1} \overset{\widehat{\alpha_n}}{\Longrightarrow}_Q q_n$ mit $p_n \mathcal{R} q_n$ in Q gematched werden oder $\exists k : 0 \leq k < n$, so dass $q_0 \overset{\widehat{\alpha_1}}{\Longrightarrow}_Q q_1 \overset{\widehat{\alpha_2}}{\Longrightarrow}_Q \dots q_{k-1} \overset{\widehat{\alpha_k}}{\Longrightarrow}_Q q_k$ ein Ablauf in Q ist für den $q_k \mathcal{R} q_k$ und $q_k \in E_Q$ gilt. Für $w = (\alpha_1 \alpha_2 \dots \alpha_n)|_\Sigma$ gilt $w \in L(P)$ und es folgt $w \in L(Q)$ oder $w \in \text{cont}(StET(Q)) \subseteq ET(Q)$.*

Beweis. Die Existenz des Ablaufes in Q soll induktiv über die einzelnen Transitionen des Ablaufes aus P bewiesen werden.
Für $n = 0$ gibt es keine Transitionen, die zuvor ausgeführt werden müssen, und $p_0 \mathcal{R} q_0$ folgt direkt aus der Voraussetzung, das $\mathcal{R}$ eine schwache as-Verfeinerungs-Relation zwischen P und Q ist.
Die Aussage des Lemmas über die Abläufe gelte für ein j mit $0 \leq j < n$.

- Fall 1 ($\exists 0 \leq k < j : q_k \in E_Q \wedge p_k \mathcal{R} q_k$): Es gilt auch $k < j + 1$, somit ist die Behauptung auch für $j + 1$ erfüllt.
- Fall 2 ($p_j \mathcal{R} q_j \wedge q_j \in E_Q$): Dann ist j das $k < j + 1$, für das q_k ein Fehler-Zustand ist.

- Fall 3 ($p_j \mathcal{R} q_j \wedge q_j \notin E_Q \wedge \neg\exists k : 0 \leq k < j \wedge q_k \in E_Q$): In der Verfeinerung P existiert die Transition $p_j \overset{\alpha_{j+1}}{\dashrightarrow}_P p_{j+1}$. Das α_{j+1} kann ein Input oder eine lokale Aktion sein. Falls $\alpha_{j+1} \in I$ gilt, folgt mit Definition 1.4.4 die Transition $q_j \overset{\alpha_{j+1}}{\dashrightarrow}_Q \overset{\varepsilon}{\Rightarrow}_Q q_{j+1}$ in Q für ein q_{j+1} mit $(p_{j+1}, q_{j+1}) \in \mathcal{R}$. Da schwach-nachlaufende may-Transitionen eine spezielle Form von schwachen may-Transitionen sind, gilt also auch $q_j \overset{\widehat{\alpha_{j+1}}}{\Longrightarrow}_Q q_{j+1}$. α_{j+1} kann jedoch auch eine lokale Aktion sein. Mit Definition 1.4.4 folgt dann die Existenz der Transition $q_j \overset{\widehat{\alpha_{j+1}}}{\Longrightarrow}_Q q_{j+1}$ in der Spezifikation Q mit $p_{j+1} \mathcal{R} q_{j+1}$.

Für jedes w aus $L(P)$ gibt es einen Ablauf wie er in diesem Lemma vorausgesetzt wurde, so dass $w = (\alpha_1 \alpha_2 \dots \alpha_n)|_\Sigma$ gilt. Im Fall 1 bzw. Fall 2 wurden bis q_k bzw. q_j die Aktionen α in ihrer Reihenfolge schwach ausgeführt. Es gibt also ein Präfix v von w, dass in Q zu einem Fehler-Zustand führt und somit $v \in StET(Q)$ erfüllt. w ist eine Fortsetzung von v und somit gilt $w \in \text{cont}(StET(Q)) \subseteq ET(Q)$. Falls der Fall 3 für $j = n-1$ angewendet werden kann, folgt daraus $p_n \mathcal{R} q_n$ und $w = (\alpha_1 \alpha_2 \dots \alpha_n)|_\Sigma$ ist die Folge der sichtbaren Transitionsbeschriftungen, die vom Startzustand q_0 zum Zustand q_n in Q führen. Das Wort w ist also in Q ausführbar und somit in der Sprache $L(Q)$ enthalten. □

Man könnte durch das gerade eben bewiesene Lemma die gleiche Aussage für starke as-Verfeinerungen zeigen, in dem man Lemma 1.16 anwendet. Die Aussage wäre zunächst auch stark genug, jedoch sobald Divergenz mit betrachtet wird, kann es durch das schwache Transitions-Matching zu Problemen kommen, da dadurch zusätzliches Divergenz-Verhalten möglich ist. Das nächste Lemma ist somit nur ähnlich zum vorangegangenen, hat jedoch eine etwas stärkere Aussage.

Lemma 2.7 (*Abläufe in as-Verfeinerungen und Spezifikationen*). *Sei P eine as-Verfeinerung eines MEIOs Q 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 $q_0 \overset{\alpha_1}{\dashrightarrow}_Q q_1 \overset{\alpha_2}{\dashrightarrow}_Q \dots q_{n-1} \overset{\alpha_n}{\dashrightarrow}_Q q_n$ mit $p_n \mathcal{R} q_n$ in Q gematched werden oder $\exists k : 0 \leq k < n$, so dass $q_0 \overset{\alpha_1}{\dashrightarrow}_Q q_1 \overset{\alpha_2}{\dashrightarrow}_Q \dots q_{k-1} \overset{\alpha_k}{\dashrightarrow}_Q q_k$ ein Ablauf in Q ist für den $q_k \mathcal{R} q_k$ und $q_k \in E_Q$ gilt. Für $w = (\alpha_1 \alpha_2 \dots \alpha_n)|_\Sigma$ gilt $w \in L(P)$ und es folgt $w \in L(Q)$ oder $w \in \text{cont}(StET(Q)) \subseteq ET(Q)$.*

Beweis. Aus $w \in L(P)$ folgt $w \in L(Q)$ oder $w \in \text{cont}(StET(Q)) \subseteq ET(Q)$ ist eine Aussage, die genau so auch in Lemma 2.6 für schwache as-Verfeinerungen galt. Jede starke as-Verfeinerung ist auch eine schwache (Lemma 1.16), somit folgt diese Aussage direkt aus Lemma 2.6.
Aus dem vorausgesetzten Ablauf für P würde mit den Lemmata 2.6 und 1.16 nur der Ablauf $q_0 \overset{\widehat{\alpha_1}}{\Longrightarrow}_Q q_1 \overset{\widehat{\alpha_2}}{\Longrightarrow}_Q \dots q_{n-1} \overset{\widehat{\alpha_n}}{\Longrightarrow}_Q q_n$ in Q folgen. In diesem Lemma sollen die Transitionen jedoch ohne zusätzliche oder entfallende τs möglich sein. Die Induktion aus dem Beweis von 2.6 kann jedoch entsprechend verstärkt werden. Man verwendet für das j mit $0 \leq j < n$ die stärkere Aussage dieses Lemmas. Die Beweise der Fälle 1 und 2 können unverändert übernommen werden. Für Fall 3 entfällt die Unterscheidung, ob α_{j+1} ein Input oder eine lokale Aktion ist. Mit Definition 1.3.3 kann begründet werden,

dass die α_{j+1}-Transition direkt zum Zustand q_{j+1} führt. Es gilt also in jedem Schritt der Induktion, der Fall 3 betrifft, $q_j \overset{\alpha_{j+1}}{\dashrightarrow}_Q q_{j+1}$. Es folgt also der Ablauf, der hier gezeigt werden sollte, falls kein Fehler-Zustand in Q angetroffen wird. □

Man kann das in Lemma 2.7 angewendete Vorgehen auch auf unendliche Ablaufe anwenden. Die Aussage gilt für alle Längen n von Abläufen. Falls ein Fehler-Zustand erreicht wird für ein k, hat man einen endlichen Fehler-Trace. Wenn kein Fehler-Zustand erreicht wird, kann jedes beliebiges Anfangsstück des unendlichen Ablaufes aus P durch einen Ablauf in Q gematched werden. Durch wiederholte Anwendung von Fall 3, kann dieser Ablauf schrittweise verlängert werden. Es muss also auch in Q einen analogen unendlichen Ablauf zu dem aus P geben.
Dieser Punkt wird in einem späteren Kapitel vor allem für unendliche τ-Abläufe bezüglich Divergenz relevant.

Korollar 2.8 (*unendliche Abläufe in as-Verfeinerungen und Spezifikationen*). *Sei P eine as-Verfeinerung eines MEIOs Q und $\mathcal{R}$ die as-Verfeinerungs-Relation zwischen den beiden Transitionssystemen. Ein unendlicher Ablauf $p_0 \overset{\alpha_1}{\dashrightarrow}_P p_2 \overset{\alpha_2}{\dashrightarrow}_P \ldots$ aus P kann durch einen unendlichen Ablauf $q_0 \overset{\alpha_1}{\dashrightarrow}_Q q_1 \overset{\alpha_2}{\dashrightarrow}_Q \ldots$ in Q zu gematched werden oder es gibt ein endliche Anfangsstück des Ablaufes, das in Q einen Fehler-Zustand führt.*

Aus den Propositionen für die Sprache und die in der Menge ET relevanten 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 zeigt, für die Fehler-Traces und die Fehler-geflutete Sprache Gleichheit und nicht nur die Inklusion.

Proposition 2.9 (*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. „⊆“:

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

$$\overset{\substack{\text{cont}\\ \text{monoton}}}{=} \bigcup_{P' \in \text{as-impl}(P)} \text{cont}(PrET(P')) \cup \text{cont}(MIT(P'))$$
$$\overset{2.5}{=} \bigcup_{P' \in \text{as-impl}(P)} ET(P').$$

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. In P' ist w entweder vollständig oder bis auf den letzten Buchstaben 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 das Lemma 2.7 angewendet werden, wobei vorausgesetzt werden kann, dass der Ablauf in P' zu einem Zustand p'_n führt, der in $E_{P'}$ enthalten ist. Falls ein p_k mit $0 \leq k < n$ 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_k mit $k < n$ $p_k \notin E_P$ gilt, kann $p'_n \mathcal{R} p_n$ gefolgert werden durch Lemma 2.7. 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 dafür das Lemma 2.7 angewendet werden, wobei das erreicht p'_n den Input a nicht als ausgehenden must-Transition besitzen soll. Falls es ein $p_k \in E_P$ mit $0 \leq k \leq n$ gibt, gilt mit Lemma 2.7 $v \in ET(P)$ 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 wäre jedoch ein Widerspruch zur Wahl des Zustandes p'_n. Es gilt also $p_n \overset{a}{\not\rightarrow}_P$ und $va \in MIT(P)$. Daraus ergibt sich direkt $w \in ET(P)$.

2. „$\subseteq$“:

$$EL(P) \overset{2.5}{=} L(P) \cup ET(P)$$
$$\overset{1.10}{\subseteq} \left(\bigcup_{P' \in \text{as-impl}(P)} L(P') \right) \cup ET(P)$$

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

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 mit Lemma 2.7 $w \in ET(P) \subseteq EL(P)$. Falls w ausführbar ist in P gilt $w \in L(P) \subseteq EL(P)$. □

Korollar 2.10 (*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 2.9.1 mindestens ein $P' \in \text{as-impl}(P)$ 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-Implementierung 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 2.9.1 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-Implementierung 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 2.11 (*Kommunikationsfehler-Semantik für Parallelkompositionen*). *Für zwei komponierbare MEIOs P_1, P_2 und ihre Komposition P_{12} gilt:*

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

Beweis.
1. „$\subseteq$“:
Da beide Seiten der Gleichung unter der Fortsetzung cont-Funktion 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 $w = va$ gibt mit $(p_{01}, p_{02}) \stackrel{v}{\Rightarrow}_{12} (p_1, p_2) \wedge a \in I_{12} \wedge (p_1, p_2) \stackrel{a}{\not\rightarrow}_{12}$. 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 $p_{01} \stackrel{v_1}{\Rightarrow}_1 p_1 \stackrel{a}{\not\rightarrow}_1$ und $p_{02} \stackrel{v_2}{\Rightarrow}_2 p_2 \stackrel{a}{\not\rightarrow}_2$ oder $p_{02} \stackrel{v_2}{\Rightarrow}_2 p_2 \stackrel{a}{\rightarrow}_2$ mit $v \in v_1 \| v_2$. Daraus kann $v_1 a \in MIT_1 \subseteq ET_1$ und $v_2 a \in EL_2$ ($v_2 a \in MIT_2$ oder $v_2 a \in L_2$) gefolgert werden. Damit folgt $w \in (v_1 \| v_2) \cdot \{a\} = (v_1 a) \| (v_2 a) \subseteq ET_1 \| EL_2$, 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: $p_{01} \stackrel{v_1}{\Rightarrow}_1 p_1 \stackrel{a}{\not\rightarrow}_1$ und $p_{02} \stackrel{v_2}{\Rightarrow}_2 p_2$ mit $v \in v_1 \| v_2$. Daraus folgt $v_1 a \in MIT_1 \subseteq ET_1$ und $v_2 \in L_2 \subseteq EL_2$. Somit gilt $w \in (v_1 \| v_2) \cdot \{a\} \subseteq (v_1 a) \| v_2 \subseteq ET_1 \| EL_2$. Dies ist eine Teilmenge der rechten Seite der Gleichung.

- Fall 2 ($w \in PrET_{12}$): Aus der Definitionen 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 bei dem Zustands-Tupel 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 $p'_1 \overset{a}{\not\rightarrow}_1$ und $p'_2 \overset{a}{\dashrightarrow}_2$. 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: entweder endet v auf einen Input I_{12} 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 $p_{02} \overset{v'_2}{\Rightarrow}_2$ gilt. (*) Falls $p_{02} \overset{v'_2}{\Rightarrow}$ 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 $p_{01} \overset{v'_1}{\Rightarrow}_1 p_1 \overset{a}{\not\rightarrow}_1$ und es gilt $v' = v''a$.

- Fall Ia) ($a \notin \Sigma_2$): Es gilt $p_{02} \stackrel{v_2'}{=\!\Rightarrow}_2 p_2$ mit $v'' \in v_1' \| v_2'$. Dadurch erhält man $(p_{01}, p_{02}) \stackrel{v''}{=\!\Rightarrow}_{12} (p_1, p_2) \stackrel{a}{\not\rightarrow}_{12}$ 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 $p_{02} \stackrel{v_2''}{=\!\Rightarrow}_2 p_2 \stackrel{a}{\not\rightarrow}_2$ und $v'' \in v_1' \| v_2''$. a ist für P_2, ebenso wie für P_1, ein nicht sichergestellter Input. Daraus folgt, dass $(p_1, p_2) \stackrel{a}{\not\rightarrow}_{12}$ 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 $p_{02} \stackrel{v_2''}{=\!\Rightarrow}_2 p_2 \stackrel{a}{\longrightarrow}_2$ mit $v_2' = v_2''a \in L_2$, wegen (*). Da die gemeinsamen Inputs synchronisiert werden, folgt bereits aus $p_1 \stackrel{a}{\not\rightarrow}_1 (p_1, p_2) \stackrel{a}{\not\rightarrow}_{12}$. 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 $p_{02} \stackrel{v_2'}{=\!\Rightarrow}_2$. Man erhält also $p_{02} \stackrel{v_2''}{=\!\Rightarrow}_2 p_2 \stackrel{a}{\dashrightarrow}_2$ mit $v'' \in v_1' \| v_2''$. Daraus ergibt sich $(p_{01}, p_{02}) \stackrel{v''}{=\!\Rightarrow}_{12} (p_1, p_2)$ mit $p_2 \stackrel{a}{\dashrightarrow}_2$, $p_1 \stackrel{a}{\not\rightarrow}_1$, $a \in I_1$ 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 ($v_1 \in PrET_1$): $\exists u_1 \in O_1^* : p_{01} \stackrel{v_1}{=\!\Rightarrow}_1 p_1 \stackrel{u_1}{=\!\Rightarrow}_1 p_1'$ 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 für alle anderen Fälle gilt $p_{02} \stackrel{v_2'}{=\!\Rightarrow}_2 p_2$ mit $(p_{01}, p_{02}) \stackrel{v'}{=\!\Rightarrow}_{12} (p_1, p_2)$.

  - Fall IIa) $\Big(u_2 \in (O_1 \cap I_2)^*, c \in (O_1 \cap I_2)$, so dass u_2c Präfix von $u_1|_{I_2}$ mit $p_2 \stackrel{u_2}{\Longrightarrow}_2 p_2' \stackrel{c}{\not\rightarrow}_2\Big)$: Für das Präfix $u_1'c$ von u_1 mit $(u_1'c)|_{I_2} = u_2c$ weiß man, dass $p_1 \stackrel{u_1'}{=\!\Rightarrow}_1 p_1'' \stackrel{c}{\dashrightarrow}_1$. Somit gilt $u_1' \in u_1' \| u_2$ und $(p_1, p_2) \stackrel{u_1'}{=\!\Rightarrow}_{12} (p_1'', p_2') \in E_{12}$, da für P_2 der entsprechende Input nicht sichergestellt wird, der mit dem c Output von P_1 zu synchronisieren 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) $\Big(p_2 \stackrel{u_2}{\Longrightarrow}_2 p_2'$ mit $u_2 = u_1|_{I_2}\Big)$: Es gilt $u_1 \in u_1 \| u_2$ und $(p_1, p_2) \stackrel{u_1}{=\!\Rightarrow}_{12} (p_1', p_2') \in E_{12}$, 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:

$$(EL_1 \| EL_2) \cup ET_{12} \stackrel{2.5}{=} ((L_1 \cup ET_1) \,\|\, (L_2 \cup ET_2)) \cup ET_{12}$$

$$
\begin{aligned}
&= (L_1 \| L_2) \cup \underbrace{(L_1 \| ET_2)}_{\substack{\subseteq (EL_1 \| ET_2) \\ \overset{1.}{\subseteq} ET_{12}}} \cup \underbrace{(ET_1 \| L_2)}_{\substack{\subseteq (ET_1 \| EL_2) \\ \overset{1.}{\subseteq} ET_{12}}} \cup \underbrace{(ET_1 \| ET_2)}_{\substack{\subseteq (EL_1 \| ET_2) \\ \overset{1.}{\subseteq} ET_{12}}} \cup ET_{12} \\
&= (L_1 \| L_2) \cup ET_{12} \\
&\overset{1.11}{=} L_{12} \cup ET_{12} \\
&\overset{2.5}{=} EL_{12}.
\end{aligned}
$$

□

Korollar 2.12 (*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:

- $$
  \begin{aligned}
  ET_{31} &\overset{2.11.1}{=} \operatorname{cont}(\operatorname{prune}((ET_3 \| EL_1) \cup (EL_3 \| ET_1))) \\
  &\overset{\substack{ET_1 \subseteq ET_2 \\ \text{und} \\ EL_1 \subseteq EL_2}}{\subseteq} \operatorname{cont}(\operatorname{prune}((ET_3 \| EL_2) \cup (EL_3 \| ET_2))) \\
  &\overset{2.11.1}{=} ET_{32},
  \end{aligned}
  $$
- $$
  \begin{aligned}
  EL_{31} &\overset{2.11.2}{=} (EL_3 \| EL_1) \cup E_{31} \\
  &\overset{\substack{EL_1 \subseteq EL_2 \\ \text{und} \\ ET_{31} \subseteq ET_{32}}}{\subseteq} (EL_3 \| EL_2) \cup ET_{32} \\
  &\overset{2.11.2}{=} EL_{32}.
  \end{aligned}
  $$

□

Lemma 2.13 (*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 2.10 (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):

  - $U = \{p_0, p_1, \dots, p_{n+1}\}$,
  - $p_{0U} = p_0$,
  - $\dashrightarrow_U = \longrightarrow_U = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j \leq n\}$
    $\cup \{(p_j, x, p_{n+1}) \mid x \in I_U \backslash \{x_{j+1}\}, 0 \leq j \leq n\}$
    $\cup \{(p_{n+1}, x, p_{n+1}) \mid x \in I_U\}$,
  - $E_U = \emptyset$.

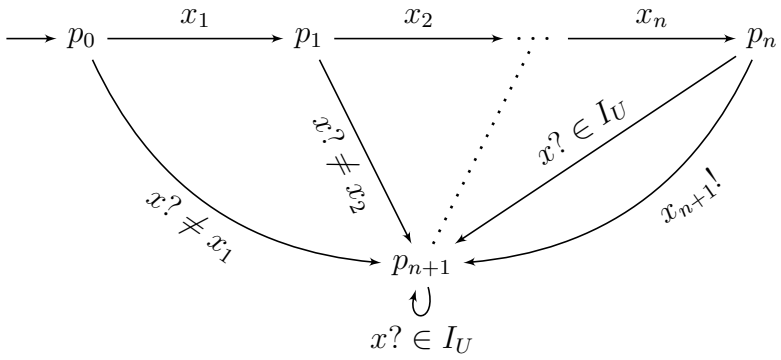

Abbildung 2.1: $x? \neq x_j$ steht für alle $x \in I_U \backslash \{x_j\}$

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

  - Fall 2a) ($w \in MIT_1$): In $U \| P_1$ erhält man $(p_0, p_{01}) \stackrel{x_1 \dots x_n}{\Longrightarrow}_{U \| P_1} (p_n, p')$ mit $p' \stackrel{x_{n+1}}{\not\longrightarrow}_1$ und $p_n \stackrel{x_{n+1}}{\longrightarrow}_U$. Deshalb ist (p_n, p') ein Element der Menge $E_{U \| P_1}$. Da alle Aktionen aus w bis auf x_{n+1} synchronisiert werden und $I \cap I_U = \emptyset$, gilt $x_1, \dots, x_n \in O_{U \| P_1}$. 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 $(p_0, p_{01}) \overset{w}{\Longrightarrow}_{U\|P_1} (p_{n+1}, p'') \overset{v}{\Longrightarrow}_{U\|P_1} (p_{n+1}, p')$ für $v \in O^*$ und $p' \in E_1$. Daraus folgt $(p_{n+1}, p') \in E_{U\|P_1}$ und somit $wv \in StET(U\|P_1)$. 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 $v \in O^*$, folgt $v \in O^*_{U\|P_1}$. 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 2.10 (ii) geschlossen werden, dass auch in $U\|P_2$ ein Fehler-Zustand lokal erreichbar sein muss.
Der in $U\|P_2$ lokal 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 mit einem passenden Input in P_2 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 $i \leq n, p' \overset{x_{i+1}}{\not\longrightarrow}_2$ und $x_{i+1} \in O_U = I$. Durch Projektion erhält man dann $p_{02} \overset{x_1 \ldots x_i}{\Longrightarrow}_2 p' \overset{x_{i+1}}{\not\longrightarrow}_2$ 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 $x_1 \ldots x_i v$ mit $v \in I_U^* = O^*$ 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 $\mathrm{prune}(x_1 \ldots x_i v) = \mathrm{prune}(x_1 \ldots x_i) \in PrET_2 \subseteq ET_2$. Da $x_1 \ldots x_i$ 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 $x_1 \ldots x_i$ in ET_2 enthalten und somit gilt auch $w \in ET_2$.

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 ($w = x_1 \ldots x_n$ mit $n \geq 1$): Es wird ein Partner U wie folgt konstruiert (siehe dazu auch Abbildung 2.2):
  - $U = \{p_0, p_1, \ldots, p_n, p\}$,
  - $p_{0U} = p_0$,
  - $\dashrightarrow_U = \longrightarrow_U = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j < n\}$
    $\cup \{(p_j, x, p) \mid x \in I_U \backslash \{x_{j+1}\}, 0 \leq j < n\}$
    $\cup \{(p, x, p) \mid x \in I_U\}$,

– $E_U = \{p_n\}$.

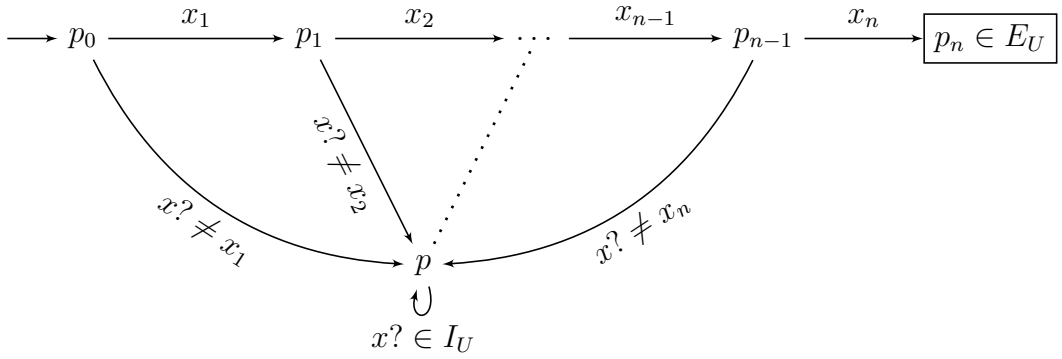

Abbildung 2.2: $x? \neq x_j$ steht für alle $x \in I_U \backslash \{x_j\}$, p_n ist der einzige Fehler-Zustand

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 2.10 (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 $p_{02} \stackrel{x_1 \ldots x_{i-1}}{\Longrightarrow}_2 p'' \stackrel{x_i}{\not\longrightarrow}_2$): Es gilt $x_1 \ldots x_i \in MIT_2$ 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 $p_{02} \stackrel{x_1 \ldots x_n}{\Longrightarrow}_2$ 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 $p_{02} \stackrel{x_1 \ldots x_i v}{\Longrightarrow}_2 p' \in E_2$ für ein $i \geq 0$ und $v \in O^*$. Somit ist $x_1 \ldots x_i v \in StET_2$ und damit $\text{prune}(x_1 \ldots x_i v) = \text{prune}(x_1 \ldots x_i) \in PrET_2 \subseteq EL_2$. 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 2.14 (*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 wegen Korollar 2.10 (i) geben, wenn P_1 und P_2 lokal erreichbare Fehler enthalten. 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 2.12 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 2.13 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.

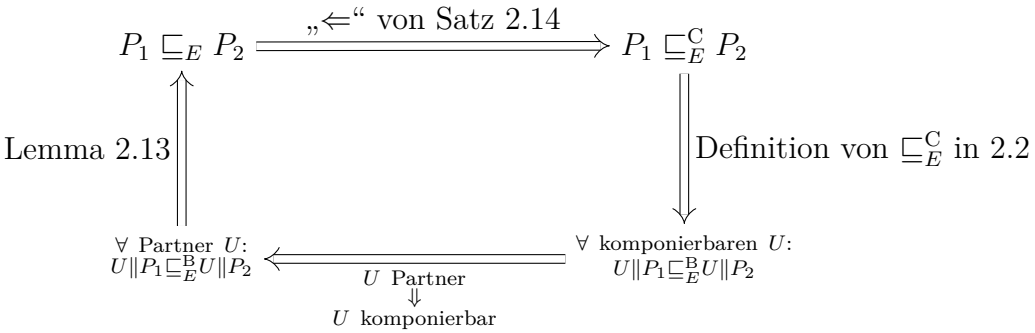

Abbildung 2.3: Folgerungskette der Fehler-Relationen

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 2.15. *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 gröbsten Präkongruenz-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 2.16 (*Test und Verfeinerung für Kommunikationsfehler*). *Sei P ein MEIO. Ein* Test *T für P ist eine zu P komponierbare Implementierung. P* as-erfüllt *T als einen Kommunikationsfehler-Test, falls $S \| T$ lokal fehler-frei ist für alle $S \in \text{as-impl}(P)$. Es wird dann $P \, sat^E_{\text{as}} \, T$ geschrieben.*
Ein MEIO P Fehler-verfeinert *P', falls sie die selbe Signaturen haben und für alle ihre Tests T: $P' \, sat^E_{\text{as}} \, T \Rightarrow P \, sat^E_{\text{as}} \, T$.*

Abgesehen von Korollar 2.10 (ii) erweisen sich Definition 2.3 bis Korollar 2.12 auch hier als nützlich.
Die Basisrelation aus dem gröbsten Präkongruenz-Ansatz gibt es in diesem Teil nicht, somit kann das Lemma 2.13 nicht in dieser Art formuliert werden. Jedoch kann hier mit Tests gearbeitet werden und unter deren zuhilfenahme ein ähnliches Lemma formuliert werden.

Lemma 2.17 (*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 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 2.9 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. Da w für P_1 präfix-minimal ist, gilt $w \in PrET_1$ oder $w \in MIT_1$ mit Proposition 2.4 kann die as-Implementierung P_1' sogar so gewählt werden, dass w für P_1' ebenfalls präfix-minimal 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 lokal fehlerfrei somit gilt $P_1 \, sat_{\text{as}}^E \, T$ nicht. Es darf also auch P_2 den Test T nicht as-erfüllen, wegen der Implikation $P_2 \, sat_{\text{as}}^E \, T \Rightarrow P_1 \, sat_{\text{as}}^E \, T$. Damit P_2 T nicht as-erfüllt muss es eine as-Implementierungen P_2' geben, die in Parallelkomposition mit T einen nicht lokal fehler-freies System ergibt. 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 2.9 folgt daraus $\varepsilon \in ET_2$.
- Fall 2 ($w = x_1 \ldots x_n x_{n+1} \in \Sigma^+$ 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):
  - $T = \{p_0, p_1, \ldots, p_{n+1}\}$,
  - $p_{0T} = p_0$,
  - $\dashrightarrow_T = \longrightarrow_T = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j \leq n\}$
    $\cup \{(p_j, x, p_{n+1}) \mid x \in I_T \backslash \{x_{j+1}\}, 0 \leq j \leq n\}$
    $\cup \{(p_{n+1}, x, p_{n+1}) \mid x \in I_T\}$,
  - $E_T = \emptyset$.

  Für das präfix-minimale w können nun zwei Fälle unterschieden werden, für die beide $\varepsilon \in PrET(P_1' \| T)$ folgen wird.

– Fall 2a) $\left(w \in MIT_{P_1'}\right)$: In $P_1'\|T$ erhält man $(p_{01}', p_0) \stackrel{x_1 \ldots x_n}{\Longrightarrow}_{P_1'\|T} (p', p_n)$ 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 $x_1, \ldots, x_n \in O_{P_1'\|T}$. Es folgt also $\varepsilon \in PrET(P_1'\|T)$.

– Fall 2b) $\left(w \in PrET_{P_1'}\right)$: In der Parallelkomposition $P_1'\|T$ erhält man die Transitionsfolge $(p_{01}, p_0) \stackrel{w}{\Longrightarrow}_{P_1'\|T} (p'', p_{n+1}) \stackrel{v}{\Longrightarrow}_{P_1'\|T} (p', p_{n+1})$ für $v \in O^*$ und $p' \in E_{P_1'}$. Daraus folgt $(p', p_{n+1}) \in E_{P_1'\|T}$ und somit $wv \in StET(P_1'\|T)$. Da alle Aktionen in w synchronisiert werden und $I \cap I_T = \emptyset$, gilt $x_1, \ldots, x_n, x_{n+1} \in O_{P_1'\|T}$ und, da $v \in O^*$, folgt $v \in O^*_{P_1'\|T}$. 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 lokal fehler-frei ist.
Der lokal erreichbar Fehler in $P_2'\|T$ kann geerbt oder neu sein. In beiden Fällen kann gezeigt werden, dass ein Präfix von w in der Menge $ET_{P_2'}$ und mit Proposition 2.9 auch in der Menge ET_2 enthalten ist.

– 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 $p_{02} \stackrel{x_1 \ldots x_i}{\Longrightarrow}_{P_2'} p' \stackrel{x_{i+1}}{\not\longrightarrow}_{P_2'}$ und damit gilt $x_1 \ldots x_{i+1} \in MIT_{P_2'} \subseteq ET_{P_2'}$. Es ist also ein Präfix von w in $ET_{P_2'}$ enthalten und mit Proposition 2.9 auch in ET_2.

– Fall 2ii) (geerbter Fehler): T hat $x_1 \ldots x_i v$ mit $v \in I_T^* = O^*$ 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 $\text{prune}(x_1 \ldots x_i v) = \text{prune}(x_1 \ldots x_i) \in PrET_{P_2'} \subseteq ET_{P_2'}$. Da $x_1 \ldots x_i$ ein Präfix von w ist, führt in diesem Fall eine Verlängerung um lokale Aktionen eines 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 $ET_{P_2'}$ enthalten und somit ist mit Proposition 2.9 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 1.10 und 2.9 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 ($w = x_1 \ldots x_n$ 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):

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

Da $p_{01} \stackrel{w}{\Longrightarrow}_{P_1'} p'$ 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) $\Big($neuer Fehler aufgrund von $x_i \in O_T$ und $p_{02} \stackrel{x_1 \ldots x_{i-1}}{\Longrightarrow}_{P_2'} p'' \stackrel{x_i}{\not\longrightarrow}_{P_2'}\Big)$: Es gilt $x_1 \ldots x_i \in MIT_{P_2'}$ und somit $w \in EL_{P_2'}$. Anzumerken ist, dass 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 2.9.
- 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 $p_{02} \stackrel{x_1 \ldots x_n}{\Longrightarrow}_{P_2'}$ gilt. In diesem Fall ist $w \in L_{P_2'} \subseteq EL_{P_2'}$. Daraus folgt mit Proposition 2.9 $w \in EL_2$.
- Fall 2d) (geerbter Fehler von P_2'): Es gilt $p_{02} \stackrel{x_1 \ldots x_i v}{\Longrightarrow}_{P_2'} p' \in E_{P_2'}$ für ein $i \geq 0$ und $v \in O^*$. Somit ist $x_1 \ldots x_i v \in StET_{P_2'}$ und damit $\text{prune}(x_1 \ldots x_i v) = \text{prune}(x_1 \ldots x_i) \in PrET_{P_2'} \subseteq EL_{P_2'}$. Mit Hilfe von Proposition 2.9 folgt $w \in EL_{P_2'} \subseteq EL_2$.

□

Satz 2.18. *Aus $P_1 \sqsubseteq_E P_2$ folgt, dass P_1 P_2 Fehler-verfeinert.*

Beweis. Für ein MEIO P gilt nach Definition $\varepsilon \in ET_P$ genau dann, wenn ein Fehler-Zustand lokal erreichbar ist in P.
Um zu zeigen, dass P_1 eine Fehler-Verfeinerung von P_2 ist, muss nachgewiesen werden, dass für alle ihre Test T $P_2\, sat^E_{\text{as}}\, T \Rightarrow P_1\, sat^E_{\text{as}}\, T$ gilt. Diese Aussage ist analog zu der Aussage, dass für alle Test T von P_1 und P_2 $\neg P_1\, sat^E_{\text{as}}\, T \Rightarrow \neg P_2\, sat^E_{\text{as}}\, T$ gilt. Es wird ein beliebiger Test T gewählt für den $\neg P_1\, sat^E_{\text{as}}\, T$ gilt. Es muss also eine as-Implementierung P_1' von P_1 geben, die in der Parallelkomposition mit T einen Fehler lokal erreicht. Es gilt also $\varepsilon \in ET_{P_1' \| T}$. Nach Satz 2.11.1 gilt die Gleichheit $ET_{P_1' \| T} =$

$\text{cont}\left(\text{prune}\left(\left(ET_{P_1'}\|EL_T\right) \cup \left(EL_{P_1'}\|ET_T\right)\right)\right)$. ε ist auf jeden Fall präfix-minimal, die cont-Funktion hat also keinen Einfluss. Mit Proposition 2.9 folgt $\varepsilon \in \text{prune}((ET_1\|EL_T) \cup (EL_1\|ET_T))$. Es gibt also ein $w \in O^*_{P_1\|T}$, dass in der Menge $ET_1\|EL_T$ oder in der Menge $EL_1\|ET_T$ enthalten ist. Es gibt also Wörter w_1 und w_T mit $w \in w_1\|w_T$, $w_1 \in ET_1$ bzw. $w_1 \in EL_1$ und $w_T \in EL_T$ bzw. $w_T \in ET_T$. Da $P_1 \sqsubseteq_E P_2$ gilt, folgt $w_1 \in ET_2$ bzw. $w_1 \in EL_2$. Es gibt aufgrund von Proposition 2.9 eine as-Implementierung P_2' von P_2 für die $w_1 \in ET_{P_2'}$ bzw. $w_1 \in ET_{P_2'}$ gilt. In Parallelkomposition mit T ist das Wort w also in der Menge $\left(ET_{P_2'}\|EL_T\right) \cup \left(EL_{P_2'}\|ET_T\right)$ enthalten. P_1 und P_2 müssen die gleiche Signatur besitzen, es gilt also auch $w \in O^*_{P_2'\|T}$ und somit $\varepsilon \in \text{prune}\left(\left(ET_{P_2'}\|EL_T\right) \cup \left(EL_{P_2'}\|ET_T\right)\right) \subseteq ET_{P_2'\|T}$. In $P_2'\|T$ ist ein Fehler lokal erreichbar und es folgt für die Spezifikation P_2 $\neg P_2\, sat^E_{\text{as}}\, T$. □

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

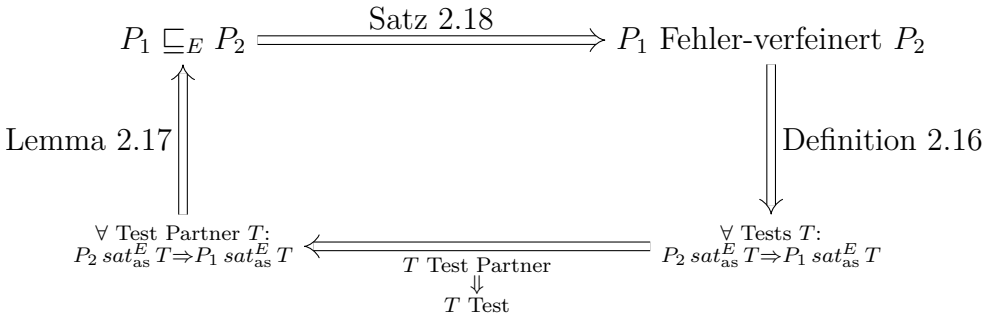

Abbildung 2.4: Folgerungskette der Testing-Verfeinerung und Fehler-Relation

Die in Abbildung 2.4 dargestellten Zusammenhänge lassen sich zu einer Äquivalenz zusammenfassen.

Korollar 2.19. *Es gilt:* $P_1 \sqsubseteq_E P_2 \Leftrightarrow P_1$ *Fehler-verfeinert* P_2.

2.3 Hiding

In diesem Kapitel soll untersucht werden, ob das Verbergen von Outputs Auswirkungen auf die Verfeinerungs-Relationen $\sqsubseteq_E$ hat.
Der relationale Zusammenhang der Basisrelation $\sqsubseteq^{\text{B}}_E$ bleibt unter der Internalisierung von Outputs erhalten. Es gilt also $P_1 \sqsubseteq^{\text{B}}_E P_2 \Rightarrow P_1/X \sqsubseteq^{\text{B}}_E P_2/X$ für alle zulässigen Aktionsmengen X. Die Begründung ist analog zu der in [Sch16]. Der Operator $\cdot/\cdot$ verändert nichts an der Fehler-Erreichbarkeit. Der Ablauf mit dem der Fehler erreicht wird, enthält nur anstatt der Outputs aus X interne Aktionen.

Zwischen den Traces von P und den Traces des Systems P/X mit verborgenen Outputs gibt es einen allgemeinen Zusammenhang.

Lemma 2.20 (*Traces unter Internalisierung*). *Für ein MEIO P gilt für alle zulässigen Aktionsmengen X: $p \stackrel{w}{\Rightarrow}_{P/X} p' \Leftrightarrow \exists w' : w'|_{\Sigma\setminus X} = w \wedge p \stackrel{w'}{\Rightarrow}_P p'$.*

Beweis. „⇒“:
Zwischen p und P' gibt es in P/X einen Trace w. Somit folgt die Existenz eines $v = \alpha_1\alpha_2\ldots\alpha_n \in \Sigma_\tau$ mit $w = \hat{v}$ und einen Ablauf der Form $p \stackrel{\alpha_1}{\dashrightarrow}_{P/X} p_1 \stackrel{\alpha_2}{\dashrightarrow}_{P/X} \ldots p_{n-1} \stackrel{\alpha_n}{\dashrightarrow}_{P/X} p'$ in P/X. Dieser Ablauf ist durch das Internalisieren von Output aus einem Ablauf aus P entstanden, da nach Definition 1.8 die Transitionen aus P/X sich auf Transitionen aus P zurückführen lassen müssen. In dem man also einige der Aktionen α_j, die interne Aktionen sind, durch Outputs aus X ersetzt erhält man ein w', für das $w'|_{\Sigma\setminus X} = w$ gilt und das die sichtbare Transitionsbeschriftung des Ablaufes zwischen p und p' in P ist.

„⇐“:
Analog zur anderen Richtung gibt es auch hier ein entsprechender Ablaufs in P von p nach p', dessen sichtbare Beschriftung in diesem Fall w' ist. Es gibt also ein $v = \alpha_1\alpha_2\ldots\alpha_n \in \Sigma_\tau$ mit $w' = \hat{v}$ und einen Ablauf der Form $p \stackrel{\alpha_1}{\dashrightarrow}_P p_1 \stackrel{\alpha_2}{\dashrightarrow}_P \ldots p_{n-1} \stackrel{\alpha_n}{\dashrightarrow}_P p'$ in P. Durch die Anwendung des Hiding-Operators werden die Aktionen α_j, die in X enthalten sind, durch τs ersetzt. Die sichtbare Beschriftung des Ablaufes in P/X lautet somit nicht mehr w' sondern nur noch $w = w'|_{\Sigma\setminus X}$. □

Dieses Lemma kann nun dazu verwendet werden, zu zeigen, dass die Relation $\sqsubseteq_E$ eine Präkongruenz bezüglich des Hiding-Operators ist.

Satz 2.21 (*Fehler-Präkongruenz bzgl. Internalisierung*). *Seien P_1 und P_2 zwei MEIOs für die $P_1 \sqsubseteq_E P_2$ gilt, somit gilt auch $P_1/X \sqsubseteq_E P_2/X$ für alle zulässigen Aktionsmengen X. Daraus folge insbesondere, dass $\sqsubseteq_E$ eine Präkongruenz bezüglich $\cdot/\cdot$ ist. Es gilt für die Sprachen und Traces:*

(i) $L(P/X) = \left\{w \in (\Sigma\setminus X)^* \mid \exists w' \in L(P) : w'|_{\Sigma\setminus X} = w\right\}$,

(ii) $ET(P/X) = \left\{w \in (\Sigma\setminus X)^* \mid \exists w' \in ET(P) : w'|_{\Sigma\setminus X} = w\right\}$,

(iii) $EL(P/X) = \left\{w \in (\Sigma\setminus X)^* \mid \exists w' \in EL(P) : w'|_{\Sigma\setminus X} = w\right\}$.

Beweis. Die Hauptaussage des Satzes folgt aus den Aussagen (i) bis (iii), somit werden zunächst diese Punkte nachgewiesen.

(i) Für ein Wort w' aus der Sprache L_P gilt nach Definition 1.7 $p_0 \stackrel{w'}{\Rightarrow}_P p$ für einen Zustand p aus P. Es kann also Lemma 2.20 angewendet werden. Daraus folgt dann, der Trace $p_0 \stackrel{w}{\Rightarrow}_{P/X} p$ in P/X für $w = w'|_{\Sigma\setminus X}$. Das Wort, das in P/X ausgeführt wird, lautet also $w = w'|_{\Sigma\setminus X} \in L_{P/X}$.

Für ein Wort w aus der Sprache des Transitionssystems P/X folgt ebenfalls mit Lemma 2.20, dass es ein w' mit $w'|_{\Sigma\setminus X} = w$ gibt, dass ein Wort aus L_P ist.

(ii) Die ET-Mengen sind unter cont abgeschlossen. Es genügt also ein präfix-minimales w' aus ET_P zu betrachten. Das präfix-minimale w' kann ein Element aus $PrET_P$ oder MIT_P sein.

- Fall 1 ($w' \in PrET_P$): Da w' in P ausführbar ist, gibt es einen Trace für das w', der durch einen Trace $v \in O_P^*$ ergänzt werden kann, so dass durch $w'v$ ein Zustand aus der Menge E_P erreicht wird. Der Trace hat also die Form $p_0 \stackrel{w'v}{\Longrightarrow}_P p \in E_P$. Auf diesen Trace kann Lemma 2.20 angewendet werden. Es ergibt sich dann der Trace $p_0 \stackrel{(w'v)|_{\Sigma\setminus X}}{\Longrightarrow}_{P/X} p$, der in P/X ausführbar ist. Nach Definition 1.8 werden die Fehler-Zustände aus P in P/X übernommen. Es gilt also auch $p \in E_{P/X}$. Das Wort $(w'v)|_{\Sigma\setminus X}$ ist somit in $StET(P/X) \subseteq ET(P/X)$ enthalten. Da v nur aus lokalen Aktionen besteht, gilt auch nach der Ersetzung der Transitionsbeschriftung durch das Hiding $v|_{\Sigma\setminus X} \in (O_P\setminus X)^*$. Daraus folgt $w = w'|_{\Sigma\setminus X} = \text{prune}((w'v)|_{\Sigma\setminus X})$. Das Wort w ist also in der Menge $ET_{P/X}$ enthalten.

- Fall 2 ($w' \in MIT_P$): In diesem Fall wird durch das Präfix v' von w' ein Zustand erreicht, in dem der Input a nicht sichergestellt ist, wobei $w' = v'a$ gilt. Es gibt also einen Trace wie in Lemma 2.20 für das Präfix v, wobei $p\ p_0$ entspricht und $p' \stackrel{a}{\not\rightarrow}_P$ gilt. In P/X gibt es einen mit $v'|_{\Sigma\setminus X}$ beschrifteten Trace von p nach p', wegen Lemma 2.20. Es gilt mit Definition 1.8 auch in P/X $p' \stackrel{a}{\not\rightarrow}_{P/X}$. $w = w'|_{\Sigma\setminus X}$ ist also in der Menge $MIT_{P/X} \subseteq ET_{P/X}$ enthalten.

Für ein präfix-minimales w aus $ET_{P/X}$ kann unterschieden werden, ob $w \in PrET_{P/X}$ oder $w \in MIT_{P/X}$ gilt.

- Fall I $\left(w \in PrET_{P/X}\right)$: Analog zu Fall 1 gibt es eine Verlängerung $v \in O_{P/X}^*$, sodass wv in P/X zu einem Fehler-Zustand führt. Es gibt also ein p, so dass $p_0 \stackrel{wv}{\Longrightarrow} p \in E_{P/X}$ gilt. Mit Lemma 2.20 kann begründet werden, dass es w' und v' gibt mit $w'|_{\Sigma\setminus X} = w$ und $v'|_{\Sigma\setminus X} = v$, so dass $w'v'$ in P von p_0 nach p führt. Es muss $p \in E_P$ gelten, wegen Definition 1.8. Die Verlängerung v' kann nur aus Outputs bestehen, somit gilt wegen der prune-Funktion $w' \in PrET_P \subseteq ET_P$.

- Fall II $\left(w \in MIT_{P/X}\right)$: Das w kann in P/X ohne den letzten Input a zu einem Zustand ausgeführt werden, in dem der Input a nicht sichergestellt wird. Analog zu Fall 2 muss es wegen Lemma 2.20 eine Erweiterung w' mit $w'|_{\Sigma\setminus X} = w$ des Wortes w auf die Menge Σ geben, für dessen Präfix ohne den letzten Input a in P ebenfalls ein Zustand erreicht wird, in dem das a nicht sichergestellt ist. Somit ist w' in P ebenfalls ein Input-kritischer Trace und es gilt $w' \in ET_P$.

(iii) Die Menge EL ist die Vereinigung der Sprache L mit der Trace-Menge ET. Die Aussage dieses Punktes folgt also direkt aus den bereits nachgewiesenen Punkten (i) und (ii).

$P_1 \sqsubseteq_E P_2$ setzt die Inklusionen $ET_1 \subseteq ET_2$ und $EL_1 \subseteq EL_2$ voraus. Mit (ii) und (iii) folgt draus $ET_{P_1/X} \subseteq ET_{P_2/X}$ und $EL_{P_1/X} \subseteq EL_{P_2/X}$. Die Relation $\sqsubseteq_E$ bleibt also trotz Hiding erhalten. Somit ist $\sqsubseteq_E$ eine Präkongruenz bezüglich des Hiding-Operators $\cdot/\cdot$. □

Aus Korollar 2.12 ist bekannt, dass $\sqsubseteq_E$ ein Präkongruenz bezüglich $\cdot\|\cdot$ ist, und aus Satz 2.21, dass $\sqsubseteq_E$ auch eine Präkongruenz bezüglich $\cdot/\cdot$ ist. Die Parallelkomposition mit Internalisierung wird nach Definition 1.9 aus diesen beiden Operatoren zusammengesetzt. Somit erhält man das folgende Korollar.

Korollar 2.22 (*Fehler-Präkongruenz mit Internalisierung*). *Die Relation $\sqsubseteq_E$ ist eine Präkongruenz bezüglich $\cdot|\cdot$.*

2.4 Zusammenhänge

Satz 2.23 (*Zusammenhang der Verfeinerungs-Relationen mit der Fehler-Relation*). *Für MEIOs P und Q gilt $P \sqsubseteq_{\text{w-as}} Q \Rightarrow P \sqsubseteq_E Q$. Die Implikation in die andere Richtung gilt jedoch nicht.*

Beweis.
$P \sqsubseteq_{\text{w-as}} Q \Rightarrow P \sqsubseteq_E Q$:
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 $v \in O_P^*$, so dass das Wort wv in P einen Fehler-Zustand erreicht. Es gibt eine Transitionsfolge wie in Lemma 2.6, so dass $wv = (\alpha_1 \alpha_2 \ldots \alpha_n)|_\Sigma$ gilt und der dadurch erreichte Zustand p_n ein Zustand aus der Menge E_P ist. Falls ein Ablauf zu einem $q_k \in E_Q$ existiert wie in Lemma 2.7, folgt, dass ein Präfix von wv in $StET(Q)$ enthalten ist. Mit $w = \text{prune}(wv)$ und dem Abschluss von ET unter cont gilt dann $w \in ET_Q$. Ansonsten gibt es in Q einen Trace für das Wort wv, der einen Zustand q_n erreicht, für den $p_n \mathcal{R} q_n$ 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 $a \in I$, wobei v in P zu einem Zustand ausführbar ist, der den Input a nicht sicherstellt. Es gibt also ein Ablauf des Wortes v in P wie in Lemma 2.6, der zu einem Zustand p_n führt, der keine ausgehende must-Transition für a besitzt. Falls es einen Ablauf wie in 2.6 zu einem q_k in E_Q gibt, folgt mit dem Lemma 2.6 und dem Abschluss der Menge ET unter cont $w \in ET_Q$. Ansonsten wird durch v in Q ein Zustand q_n erreicht, für den $p_n \mathcal{R} q_n$ gilt. Da a für p_n in P keine ausgehende must-Transition sein kann, gilt mit 1.4.2 auch $q_n \overset{a}{\not\longrightarrow}$. Das 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. Es soll ein beliebiges Wort w aus der Menge $EL_P \backslash ET_P$ betrachtet werden. $EL_P \backslash ET_P$ ist eine Teilmenge der Sprache L_P. Es gibt also einen Ablauf für w in P der Form, wie sie in Lemma 2.6 vorausgesetzt wird. Falls ein q_k aus 2.6 für $0 \leq k \leq n$ 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 $q_0 \overset{\widehat{\alpha_1}}{\Longrightarrow}_Q q_1 \overset{\widehat{\alpha_2}}{\Longrightarrow}_Q \dots q_{n-1} \overset{\widehat{\alpha_n}}{\Longrightarrow}_Q q_n$ in Q mit $p_n \mathcal{R} q_n$ und somit $w \in L_Q \subseteq EL_Q$.

$P \sqsubseteq_{\text{w-as}} Q \not\Leftarrow P \sqsubseteq_E Q$:
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. P und Q sind in der Abbildung 2.5 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 die Transition $q_0 \overset{o}{\longrightarrow}_Q q_1$ in Q vorhanden ist, wird die Verfeinerung dieser in P gefordert es müsste auch eine must-Output-Transition in P geben. Es gibt jedoch keine must-Transition in P, dies stellt einen Widerspruch zur Annahme dar und es folgt, dass es keine schwache as-Verfeinerungs-Relation zwischen P und Q geben kann.

Abbildung 2.5: Gegenbeispiel zu $\sqsubseteq_{\text{w-as}} \not\Leftarrow \sqsubseteq_E$ 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 ausschließlich eingehende may-Input-Transitionen zulässt. Dabei werden die lokalen Aktionen, die zu einem Fehler führen abgeschritten. Außer dem universalen Zustand gibt es in einem MIA wie in [BFLV16] keine weiteren Fehler-Zustände. 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 2.10 (i). Dies entspricht $\varepsilon \in PrET(P)$. Das Abschneiden der lokalen Aktionen wird hier nur in der Trace Menge praktiziert, in [BFLV16] jedoch direkt auf den Transitionssystemen. Im Fall der lokalen Fehler-Erreichbarkeit bleibt dort also nur noch der Zustand e als universal Zustand übrig, für den jedes Verhalten als Verfeinerung zulässig ist, der jedoch keine ausgehenden Transitionen besitzt. Für die Fehler-Zustände ist in dem hier verwendeten Ansatz auf Trace-Ebene bzw. bezüglich der Verfeinerungs-Relationen beliebiges Verhalten möglich. In den Transitionssystemen muss es analog zu e die Transitionen für das beliebiges Verhalten im allgemeinen nicht geben. Jedoch sind ausgehende Transitionen von Zuständen aus E hier dennoch zulässig.
Da auch der Testing-Ansatz die lokale Fehler-Erreichbarkeit verwendet, existiert der Zusammenhang auch dort für die Parallelkomposition mit dem entsprechenden Test.

Jeder MEIO kann normiert werden, so dass er nur noch einen Fehler-Zustand besitzt. Im folgenden sollen nicht mehr alle MIAs wie in [BFLV16] mit den MEIOs verglichen werden, da die MIAs disjunktive must-Transitionen besitzen. Jedoch sind Modale Transitionssysteme, die syntaktisch Konsistent sind und einen universal Zustand haben, der die gleichen Voraussetzungen wie für MIAs erfüllt, ebenfalls MIAs nur statt der disjunktiven must-Transitionen, haben deren must-Transitionen auch nur einen Zustand als Ziel und keine ganze Menge. Im folgenden sollen MIAs Modale Transitionssysteme sein, die diese Voraussetzungen erfüllen. Dann gibt es zu jedem MEIO einen äquivalenten MIA.
Die Relation $\sqsubseteq$ aus [BFLV16] wurde als Grundlage für die Definition der schwachen Simulationen in 1.4 verwendet. Für normalisierte MEIOs entspricht sie also der Relation $\sqsubseteq_{\text{w-as}}$ und es gelten die bereits nachgewiesenen Zusammenhänge zu den anderen Relationen.

Definition 2.24 (*Normalform*). *Ein MEIO P ist in* Normalform *(NF), falls die Menge E_P der Fehler-Zustände nur ein einziges Element enthält $E_P = \{e\}$ und für jede Transition, für die $p \xrightarrow{\alpha} e$ mit $p \neq e$ gilt, α ein Element der Menge der Input-Aktionen I ist. Der Zustand e soll keine ausgehenden Transitionen besitzen.*

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

(i) Definiere $\overline{E_P} = \left\{p \mid \exists p' \in E_P, w \in O^* : p \stackrel{w}{\Rightarrow}_P p'\right\}$ und ersetzte E_P durch $\overline{E_P}$. Der dadurch entstehenden MEIO heißt $\overline{P}$.

(ii) Falls $p_0 \in \overline{E_P}$, ist $NF(P) = (\{p_0\}, I, O, \emptyset, \emptyset, p_0, \{p_0\})$.
Ansonsten, füge einen neuen Zustand e hinzu, der der einzige Fehler-Zustand wird. Wann immer $p \stackrel{\alpha}{\dashrightarrow}_P p' \in \overline{E_P}$ 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 $p \stackrel{\alpha}{\dashrightarrow}_{NF(P)} e$ hinzu. Entfernen die Zustände aus der Menge $\overline{E_P}$ aus dem Transitionssystem.

Schritt (i) der Normalform Konstruktion verändert nicht 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 $StET$ nach dem Schritt (i) alle Traces enthält, die direkt oder durch lokal Aktionen zu einem Fehler-Zustand des ursprünglichen MEIOs führen. Für die Menge $MIT(P)\backslash\text{cont}(PrET(P))$ ändert sich nichts durch die Anwendung des ersten Schrittes auf P, da sich an den Zuständen ohne Zusammenhang zu Fehler-Zuständen nicht ändert. $\overline{P}$ ist also äquivalent zu P bezüglich der Relation $\sqsubseteq_E$.
Im ersten Fall von Schritt (ii) gilt $ET(P) = \Sigma^* = ET(NF(P))$. Im zweiten Fall von Schritt (ii) werden Zuständen, die nicht in $\overline{E_P}$ enthalten sind und die keine Transitionen zu Zuständen in der Menge $\overline{E_P}$ besitzen, unverändert aus $\overline{P}$ in $NF(P)$ übernommen. Transitionen zwischen solchen Zuständen werden ebenfalls ohne Veränderungen in das Transitionssystem $NF(P)$ übernommen. Ein Ablauf zu einem Traces aus der Menge $MIT(P)\backslash\text{cont}(PrET(P))$ beinhaltet nur solche unveränderten Zustände und Transitionen. Somit ist jeder Trace aus $MIT(P)\backslash\text{cont}(PrET(P))$ auch in $NF(P)$ ein Input-kritische Trace und alle Traces $MIT(NF(P))\backslash\text{cont}(PrET(NF(P)))$ sind auch in P Input-kritische Traces, die nicht in $\text{cont}(PrET(P))$ enthalten sind. Da in $\sqsubseteq_E$ nur die Menge ET relevant ist, müssen Traces aus $MIT(P) \cap PrET(P)$ in $NF(P)$ keine Input-kritischen Traces sein, damit P und $NF(P)$ äquivalent sein können bezüglich $\sqsubseteq_E$. Da jeweils die erste Transition, die zu einem Zustand in der Menge $\overline{E_P}$ führt zum Zustand e umgebogen wird, sind die präfix-minimalen Traces, die in $PrET(P)$ enthalten sind auch alle in $PrET(NF(P))$ enthalten und jedes Element aus $PrET(NF(P))$ ist ein präfix-minimales Element aus $PrET(P)$. Durch das umbiegen der präfix-minimalen Elemente aus $PrET(P)$ auf den Zustand e, bei dem der letzte Input nur als may-Transition ungesetzt wird und e keine ausgehenden Transitionen besitzt, entstehen neue Input-kritische Traces. Jedoch sind diese neuen Input-kritischen Traces gleichzeitig in $MIT(NF(P))$ und in $\text{cont}(PrET(NF(P)))$ enthalten. Die Menge der Fehler-Traces ET bleibt somit im Schritt (ii) gleich. Die Sprache des MEIOs P wird durch die Konstruktion in (ii) um die Traces in $\text{cont}(PrET(P))$, die nicht präfix-minimal sind, verkleinert. Da diese Traces jedoch in der Menge ET enthalten sind, mit denen die Sprache L in EL geflutet wird, spielt

diese bezüglich der Relation $\sqsubseteq_E$ keine Rolle. Es gilt also auch $EL(P) = EL(NF(P))$. Somit ist $NF(P)$ äquivalent zu P bezüglich der Relation $\sqsubseteq_E$.

Proposition 2.25 (*Normalform*). *Jedes MEIO P ist äquivalent zu seiner Normalform $NF(P)$ bezüglich der Relation $\sqsubseteq_E$.*

Durch die Konstruktion in Definition 2.24 kann man zu jedem MEIO P einen bezüglich der Relation $\sqsubseteq_E$ äquivalenten MEIO in Normalform konstruieren. Um zu zeigen, dass es auch immer einen bezüglich $\sqsubseteq_E$ äquivalenten MIA ohne disjunktive must-Transitionen gibt, reicht es nun aus zu beweisen, dass ein MEIO in Normalform in einem MIA umgewandelt werden kann.

Satz 2.26 (*Existenz äquvialenter MIAs*). *Für jeden MEIO P gibt es einen äquivalenten MIA P' bezüglich der Relation $\sqsubseteq_E$.*

Beweis. Man kann oBdA davon ausgehen, dass P ein MEIO in Normalform ist. P besitzt also genau einen Fehler-Zustand. Dies ist der universale Zustand des MIAs. Alle eingehenden Transitionen in den Zustand e sind may-Input-Transitionen und alle must-Transitionen haben zugrundeliegende may-Transitionen. Die Menge der Zustände von P' entspricht P. Die Startzustände von P und P' sind gleich, genau so wie die Mengen der Input- und Output-Aktionen. Für die Relation $\dashrightarrow_{P'}$ gilt $\dashrightarrow_{P'} = \dashrightarrow_P$ und analog für die must-Transition. Das daraus resultierende Transitionssystem P' ist ein MIA nach der Definition in [BFLV16] bis auf die disjunktiven must-Transitionen, die hier durch must-Transitionen ohne Disjunktionen ersetzt wurden. □

Die MIAs aus [BFLV16] dürfen disjunktive must-Transitionen besitzen. Falls man jedoch die hier gemachte Einschränkung vornimmt, ist jeder MIA auch ein MEIO.

3 Verfeinerungen für Kommunikationsfehler- und Stillstand-Freiheit

In diesem Kapitel wird die Menge der betrachteten Zustandsmengen von den Kommunikations-basierten Fehlern im letzten Kapitel ergänzt um die Menge stiller 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

In den EIOs aus z.B. [Sch16] ist ein stiller Zustand, ein Zustand, der keine ausgehenden Transitionen für lokale Aktionen besitzt. Im Fall der hier betrachteten MEIOs müssen solche Zustände ebenfalls als still angesehen werden, jedoch bereits Zustände, die keine ausgehenden must-Transitionen für Outputs und τs besitzen lassen in as-Implementierungen zu, dass dort ein Stillstand entsteht. Falls also nur may-Transitionen für die lokalen Aktionen vorhanden sind, sollte der Zustand hier auch bereits still sein. Im folgenden wird beweisen, dass diese Idee richtig ist.
Zustände, die keine must-Transitionen für lokale Aktionen 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-Transition für lokale Aktionen nicht implementiert wird. So ein Zustand hat also keine must-Transitions-Möglichkeiten für lokale Aktionen, es ist also ein Deadlock-Zustand, in denen das System nichts mehr tun können muss ohne einen Input. Diese Zustände werden in dieser Arbeit als still bezeichnet, da wenn dieser Zustand erreicht wird, dass System still steht bis die Umwelt eine Aktion ausführt.

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: $Qui(P) := \left\{p \in P \mid \forall \alpha \in (O \cup \{\tau\}) : p \overset{\alpha}{\not\rightarrow}_P\right\}$.

Der Ansatz, dass stille Zustände keine Output und keine τ-Transitionen sicherstellen verallgemeinert die Definition der Stille für EIOs in z.B. [Sch16].

Ob ein stiller Zustand relevant ist für einen Transitionssystem wird wie im letzten Kapitel für die Fehler-Zustände durch ein optimistischer Ansatz der lokalen Erreichbarkeit bestimmt. Stille ist aber keine 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 unmittelbar relevant, 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*). *Sei P ein MEIO.*
P ist lokal fehler- *und* stillstand-frei*, wenn kein Fehler- und kein stiller Zustand lokal erreichbar ist.*
Ein Test *T für P ist eine zu P komponierbare Implementierung. P* as-erfüllt *T als einen Stillstands-Test, falls $S \| T$ lokal fehler- und stillstand-frei ist für alle $S \in$ as-impl(P). Es wird dann $P \, sat_{\text{as}}^{Qui} \, T$ geschrieben.*
Ein MEIO P Stille-verfeinert *P', falls sie die selbe Signatur haben und für alle ihre Tests T: $P' \, sat_{\text{as}}^{Qui} \, T \Rightarrow P \, sat_{\text{as}}^{Qui} \, T$.*

Um eine genauere Auseinandersetzung mit dem Fehlverhalten in unterschiedlichen Systemen 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*: $StQT(P) := \left\{ w \in \Sigma^* \mid p_0 \stackrel{w}{\Rightarrow}_P p \in Qui(P) \right\}$.*

Die Definition der strikten Stille-Traces befasst sich mit MEIOs im allgemeinen. Jedoch ist auch in diesem Kapitel der Zusammenhang mit 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 1.10 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.

Für alle $w \neq \varepsilon$ und, mit $w \in StQT(P)$ kann P' als die folgende as-Implementierung gewählt werden. Diese deckt auch $w = \varepsilon$ ab, wenn mindestens eine τ-Transition benötigt wird um den stillen Zustand zu erreichen.

- $P' = \{p' \mid p \in P\} \dot\cup \{p'' \mid p \in P\}$,
- Startzustand p'_0,
- $I_{P'} = I_P$ und $O_{P'} = O_P$,
- $\longrightarrow_{P'} = \dashrightarrow_{P'} = \left\{(p'_j, \alpha, p'_k) \mid p_j \overset{\alpha}{\dashrightarrow} p_k\right\}$
  $\cup \left\{(p'_j, \alpha, p''_k) \mid p_j \overset{\alpha}{\dashrightarrow} p_k\right\}$
  $\cup \left\{(p''_j, \alpha, p''_k) \mid p_j \overset{\alpha}{\longrightarrow} p_k\right\}$
- $E_{P'} = \emptyset$.

In Abbildung 3.1 ist für ein Beispiel dargestellt, wie die as-Implementierung P' aus einem MEIO P entsteht.
Als as-Verfeinerungs-Relation zwischen P und P' wird die Relation $\mathcal{R} = \left\{(p'_j, p_j) \mid p_j \in P\right\} \cup \left\{(p''_j, p_j) \mid p_j \in P\right\}$ verwendet. Es werden in P' für die Zustände mit einem Strich p' die must- und may-Transitionen zu beiden „Arten“ von Zuständen implementiert und für die Zustände mit zwei Strichen p'' nur die must-Transitionen zu den anderen Zuständen mit zwei Strichen. Die Zustände p'' sind die potentiell ruhigen Zustände, die nur die zwingend notwendigen ausgehenden Transitionen besitzen. Die Zustände mit p' hingegen sind die Zustände, die das gesamte erlaubte Verhalten umsetzten. Durch die Transitionen, die von den zweifach gestrichenen Zuständen zu den einfach gestrichenen führen, ist alles erlaubte möglich bis entschieden wird, nur noch das geforderte zu benötigen. Da die Menge der Fehler-Zustände leer ist, gilt 1.3.1 für $\mathcal{R}$. Die must-Transitionen werden für die einfach und zweifach 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 Zustände mit einem wie auch mit zwei Strichen 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 p''_j in P', da für p''_j alle ausgehenden must-Transitionen von p''_j implementiert wurden, aber keine einzige may-Transition, die keiner 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 p''_j nicht. p''_j ist in P' mit den selben Traces erreichbar wie p_j in P, da jeder Zustand in P' die selben eingehenden Transitionen hat wie der entsprechende in Relation $\mathcal{R}$ stehende Zustand aus P. Von p'_0 aus kann der Trace, der in P zu einem stillen Zustand p_j, in P' nachgemacht werden über die einfach gestrichenen Zustände. Die letzte Transition des Ablaufes muss dann zum Zustand p''_j genommen werden. 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. p'' führt.

Die oben aufgeführte as-Implementierung und die as-Verfeinerungs-Relation lassen sich für fasst alle Traces aus $StQT$ einsetzen und zu zeigen, dass es diese auch in der Vereinigung dieser Traces aller as-Implementierungen gibt. Jedoch gibt es ein Problem, wenn

der Trace in $StQT(P)$ keine Transition in P ausführen muss, um den stillen Zustand zu erreichen. Für diesen Fall muss eine andere as-Implementierung mit einer entsprechend angepassten as-Verfeinerungs-Relation verwendet werden. Die as-Implementierung P' für den Fall, dass der Startzustand p_0 in der Zustandsmenge $Qui(P)$ enthalten ist, 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 $p \overset{a}{\not\rightarrow}_P$, gilt auch in P' $p \overset{a}{\not\rightarrow}_{P'}$ für den Zustand, der mit p in Relation steht. Der Startzustand von P' ist mit ε erreichbar und ebenfalls still. Es gilt also $p_0 \in Qui(P')$ für den Startzustand von 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. □

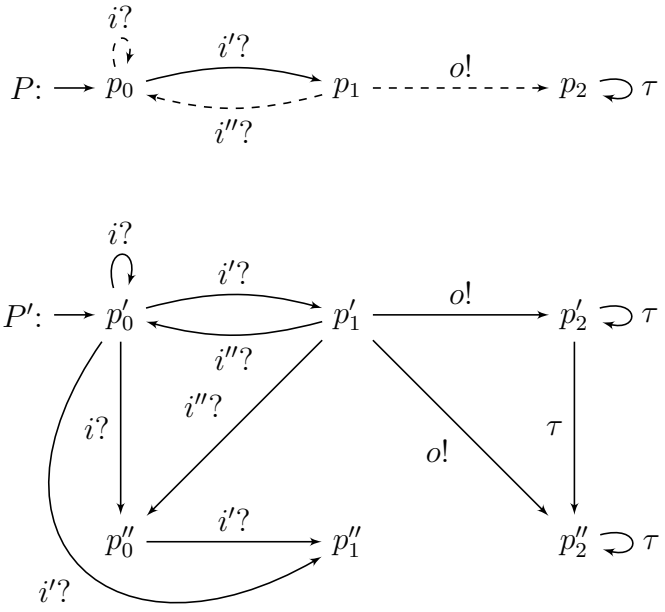

Abbildung 3.1: Beispiel zur Konstruktion aus Beweis zu Proposition 3.4

Man hätte anstatt des hier verwendeten P' mit zwei Arten von Zuständen auch den Ansatz des ausrollens für jeden Trace 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.

Durch die Fehler-Flutung kann die Inklusion aus Proposition 3.4 für QET analog zu Proposition 2.9 zur Gleichheit erweitert werden.

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.
„$\subseteq$“:

$$\begin{aligned} QET(P) &\overset{3.5}{=} StQT(P) \cup ET(P) \\ &\overset{3.4}{\subseteq} \left(\bigcup_{P' \in \text{as-impl}(P)} StET(P') \right) \cup ET(P) \\ &\overset{2.9.1}{=} \left(\bigcup_{P' \in \text{as-impl}(P)} StET(P') \right) \cup \left(\bigcup_{P' \in \text{as-impl}(P)} ET(P') \right) \\ &= \bigcup_{P' \in \text{as-impl}(P)} StQT(P') \cup ET(P') \\ &\overset{3.5}{=} \bigcup_{P' \in \text{as-impl}(P)} QET(P'). \end{aligned}$$

„$\supseteq$“:
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 2.9 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. Der mit w beschriftete Ablauf führt in P' also nur zu einem stillen Zustand p'_n und hat nichts mit Fehler-Zuständen in P' zu tun. Er hat die Form aus Lemma 2.7. Da P' eine as-Verfeinerung von P ist, gibt es auch eine as-Verfeinerungs-Relation $\mathcal{R}$, die zwischen den beiden MEIOs gilt. In P führt wegen 2.7 entweder ein Präfix von w zu einem Fehler-Zustand oder w ist in P zu einem Zustand p_n ausführbar, für den $p'_n \mathcal{R} p_n$ gilt. Falls ein Präfix von w zu einem Fehler-Zustand führt, gilt $w \in ET_P \subseteq QET_P$. Andernfalls wird 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 $\omega \in O \cup \{\tau\}$ $p'_n \overset{\omega}{\not\rightarrow}$. Da $(p'_n, p_n) \in \mathcal{R}$ gilt und beide Zustände keine Fehler-Zustände sind, muss auch $p_n \overset{\omega}{\not\rightarrow}$ 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 2.10 (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 $P' \in \text{as-impl}(P)$ 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 via 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.

   - 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_1, p_2) die Möglichkeit für eine τ-must-Transition. Dies ist jedoch durch die Voraussetzung verboten und somit kann dieser Fall nicht eintreten.

   - Fall 2 $\left(p_1 \xrightarrow{a}_1 \text{ mit } a \in O_1 \backslash \text{Synch}(P_1, P_2)\right)$: Dieser Fall kann wegen einer analogen Begründung wie in Fall 1 nicht auftreten.

- 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 in 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 nicht sicherstellt, 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} 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 2.11.

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 dessen Zugehörigkeit zur rechten Menge gezeigt. Aufgrund von Definition 3.3 weiß man, dass $(p_{01}, p_{02}) \overset{w}{\Longrightarrow}_{12} (p_1, p_2)$ gilt mit $(p_1, p_2) \in Qui_{12} \backslash E_{12}$. Durch Projektion erhält man $p_{01} \overset{w_1}{\Longrightarrow}_1 p_1$ und $p_{02} \overset{w_2}{\Longrightarrow}_2 p_2$ 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 2.11 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 $P_1 \sqsubseteq_{Qui} P_2$ für zwei MEIOs P_1 und P_2 gilt, so auch $P_{31} \sqsubseteq_{Qui} P_{32}$ für alle komponierbare Systeme 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, wegen der Definition von $\sqsubseteq_{Qui}$ in 3.5. Dies ergibt sich, wie im Beweis zu Korollar 2.12, aus der Monotonie von $\cdot\|\cdot$ auf Sprachen wie folgt:

- $P_{31} \overset{\text{Korollar 2.12 und } P_1 \sqsubseteq_E P_2}{\sqsubseteq_E} P_{32}$,
- $QET_{31} \overset{3.9.2}{=} (QET_3 \| QET_1) \cup ET_{31} \overset{ET_{31} \subseteq ET_{32} \text{ und } QET_1 \subseteq QET_2}{\subseteq} (QET_3 \| QET_2) \cup ET_{32} \overset{3.9.2}{=} QET_{32}$.

□

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 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 2.17 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 Tests 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 2.17 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 2.9, 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 den Tests T nicht 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 wegen der τ-Schlinge 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 2.9 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 3.2):
  - $T = \{p_0, p_1, \dots, p_{n+1}\}$,
  - $p_{0T} = p_0$,
  - $\dashrightarrow_T = \longrightarrow_T = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j \leq n\}$
    $\cup \{(p_j, x, p_{n+1}) \mid x \in I_T \backslash \{x_{j+1}\}, 0 \leq j \leq n\}$
    $\cup \{(p_{n+1}, x, p_{n+1}) \mid x \in I_T\}$
    $\cup \{(p_j, \tau, p_j) \mid 0 \leq j \leq n+1\}$,
  - $E_T = \emptyset$.

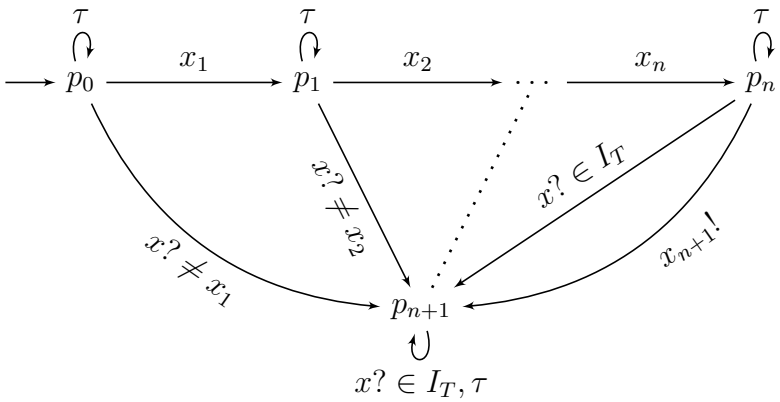

Abbildung 3.2: $x? \neq x_j$ steht für alle $x \in I_T \backslash \{x_j\}$

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 dem Beweis der selben Inklusion von Lemma 2.17. 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 2.17. 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 2.17 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 1.10 und 2.9 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 2.17 gilt auch hier, dass ε immer in EL_2 enthalten ist.
- Fall 2 ($w = x_1 \dots x_n$ 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 2.17 ab. Somit ist der Partner T dann wie folgt definiert (siehe dazu auch Abbildung 3.3):
  - $T = \{p_0, p_1, \dots, p_n, p\}$,
  - $p_{0T} = p_0$,
  - $\dashrightarrow_T = \longrightarrow_T = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j < n\}$
    $\cup \{(p_j, x, p) \mid x \in I_T \backslash \{x_{j+1}\}, 0 \leq j < n\}$
    $\cup \{(p_j, \tau, p_j) \mid 0 \leq j \leq n\}$
    $\cup \{(p, \alpha, p) \mid \alpha \in I_T \cup \{\tau\}\}$,
  - $E_T = \{p_n\}$.

  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 2.17. Und somit gilt für alle Fälle $\big($2a) bis 2d)$\big)$, 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 2.9 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.

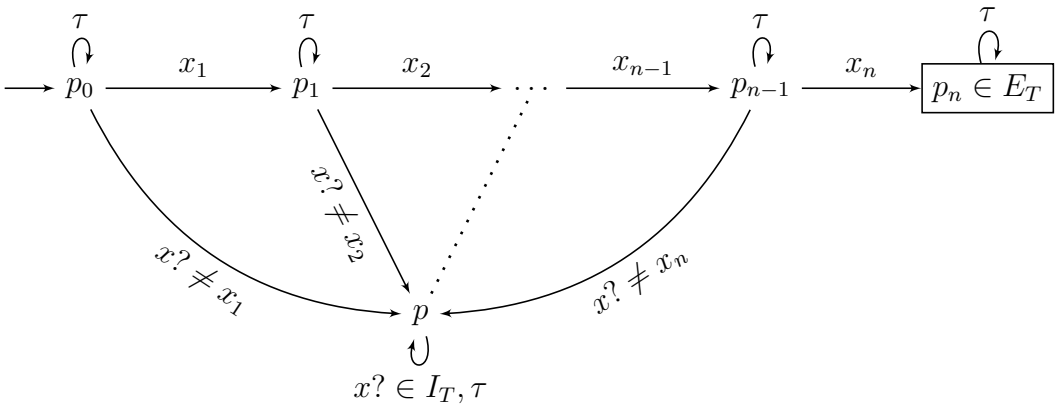

Abbildung 3.3: $x? \neq x_j$ steht für alle $x \in I_T \backslash \{x_j\}$, p_n ist der einzige Fehler-Zustand

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 $w = x_1 \dots x_n \in \Sigma^*$ mit $n \geq 0$. Es wird der folgende Partner T betrachtet (siehe auch Abbildung 3.4):

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

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 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.

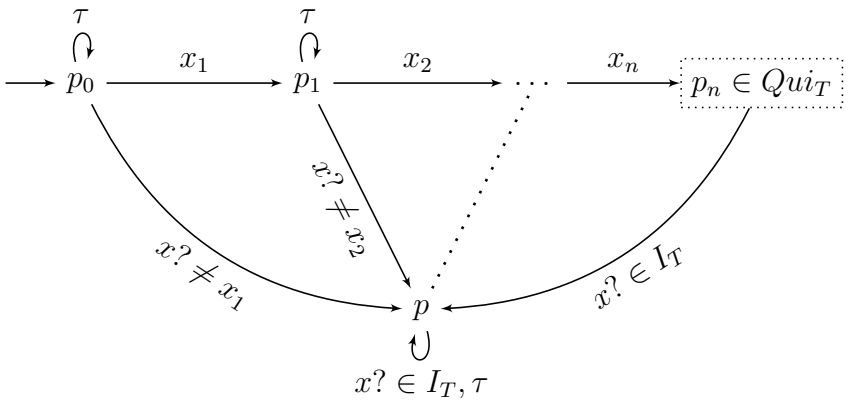

Abbildung 3.4: $x? \neq x_j$ steht für alle $x \in I_T \backslash \{x_j\}$, p_n ist der einzige stille Zustand

- 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. Von T kann kein Fehler geerbt werden und T kann hat in allen Zuständen die Möglichkeit alle Inputs entgegen zu nehmen. Die einzigen Outputs, die T ausführen kann, befinden sich auf dem Trace, auf dem T das Wort w ausführt. Es gilt, dass ein Präfix von w in $ET_{P_2'}$ enthalten ist, wegen des Beweises des ersten Punktes aus Lemma 2.17 und da T nur neue Fehler auf dem Trace w zulässt. Die Menge ET ist unter cont abgeschlossen, somit gilt $w \in ET_{P_2'} \subseteq QET_{P_2'}$. 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. *Aus $P_1 \sqsubseteq_{Qui} P_2$ folgt, dass P_1 P_2 Stille-verfeinert.*

Beweis. In einem MEIO P ist ein Fehler- oder stiller Zustand nach Definition genau dann lokal erreichbar, wenn $w \in QET_P$ für ein $w \in O_P^*$ gilt.
P_1 Stille-verfeinert P_2 genau dann, wenn für alle Tests T von P_1 und P_2 die Implikation $P_2\, sat_{\text{as}}^{Qui}\, T \Rightarrow P_1\, sat_{\text{as}}^{Qui}\, T$ gilt. Wenn man die Implikation negiert, ist für alle Tests T der beiden MEIOs $\neg P_1\, sat_{\text{as}}^{Qui}\, T \Rightarrow \neg P_2\, sat_{\text{as}}^{Qui}\, T$ zu zeigen. Es soll $\neg P_1\, sat_{\text{as}}^{Qui}\, T$ für einen beliebig gewählten Test T gelten. Es existiert also ein P_1' aus der Menge as-impl(P_1), für das $P_1' \| T$ einen Fehler oder einen stillen Zustand lokal erreicht. Für ein $w \in O_{P_1' \| T}^*$ gilt

also $w \in QET_{P_1' \| T}$. Aufgrund von Definition 3.5 kann die Menge $QET_{P_1' \| T}$ auch als die Vereinigung von $StQT_{P_1' \| T}$ und $ET_{P_1' \| T}$ geschrieben werden.

- Fall 1 $\left(w \in ET_{P_1' \| T}\right)$: Es muss auch $\varepsilon \in ET_{P_1' \| T}$ gelten, da w keinen Input aus der Menge $I_{P_1' \| T}$ enthält und somit w kein Element der Menge $\text{cont}(MIT_{P_1' \| T})$ sein kann. Es kann also der Beweis von Satz 2.18 angewendet werden, um in diesem Fall zu folgern, dass für eine as-Implementierung P_2' von P_2 ein Fehler in $P_2' \| T$ lokal erreichbar ist und somit $\neg P_2 \, sat_{\text{as}}^{Qui} \, T$ gilt.

- Fall 2 $\left(w \in StQT_{P_1' \| T} \backslash ET_{P_1' \| T}\right)$: Damit dieser Fall eintreten kann, muss das Wort w in $P_1' \| T$ einen stillen Zustand (p_1, p_T) erreichen, der nicht in $E_{P_1' \| T}$ enthalten ist. Mit Lemma 3.8.2 sind auch die Zustände p_1 und p_T in den Teilsystemen still. Es gibt also Projektionen des Wortes w auf die Teilsysteme, so dass $w \in w_1 \| w_T$, $w_1 \in StQT_{P_1'}$ und $w_T \in StQT_T$ gilt, wobei w_1 den Zustand p_1 in P_1' und w_T den Zustand p_T in T erreicht. w_1 ist in der Menge $StQT_{P_1'} \subseteq QET_{P_1'}$ enthalten. Mit Proposition 3.6 gilt also auch $w_1 \in QET_1$ und mit $P_1 \sqsubseteq_{Qui} P_2$ folgt draus, dass w_1 auch in der Menge QET_2 enthalten sein muss. Es gibt wegen Proposition 3.6 auch eine as-Implementierung P_2' von P_2, für die $w_1 \in QET_{P_2'}$ gilt. Da QET die Menge der fehler-gefluteten Stille-Trace ist, können im folgenden zwei Fälle unterschieden werden.

  - Fall 2a) $\left(w_1 \in ET_{P_2'}\right)$: Da w_T in T ausführbar ist, gilt $w_T \in L_T \subseteq EL_T$ und somit folgt mit Satz 2.11.1 $w \in ET_{P_2' \| T}$. Es ist also wie im Beweis von Satz 2.18 ein Fehler in $P_2' \| T$ lokal erreichbar. Somit folgt wie in Fall 1 $\neg P_2 \, sat_{\text{as}}^{Qui} \, T$.

  - Fall 2b) $\left(w_1 \in StQT_{P_2'}\right)$: Das Wort w_1 führt in P_2' zu einem Zustand p_2, der still ist. Wenn man also die Wörter w_1 und w_T in $P_2' \| T$ parallel zum Zustand (p_2, p_T) ausführt, ist dieser Zustand durch Lemma 3.8.1 still. Es gilt also $w \in StQT_{P_2' \| T}$ mit $w \in O^*_{P_2' \| T}$, da P_1 und P_2 die selbe Signatur haben müssen. Ein stiller Zustand ist als in der Parallelkomposition $P_2' \| T$ lokal erreichbar. Es gilt also auch in diesem Fall $\neg P_2 \, sat_{\text{as}}^{Qui} \, T$.

□

Es wurde, wie im letzten Kapitel, eine Kette an Folgerungen gezeigt, die sich zu einem Ring schließen. Dies ist in Abbildung 3.5 dargestellt.

Aus der Abbildung 3.5 folgt eine Äquivalenz, die im nächsten Korollar noch extra hervorgehoben wird.

Korollar 3.13. *Es gilt:* $P_1 \sqsubseteq_{Qui} P_2 \Leftrightarrow P_1$ *Stille-verfeinert* P_2.

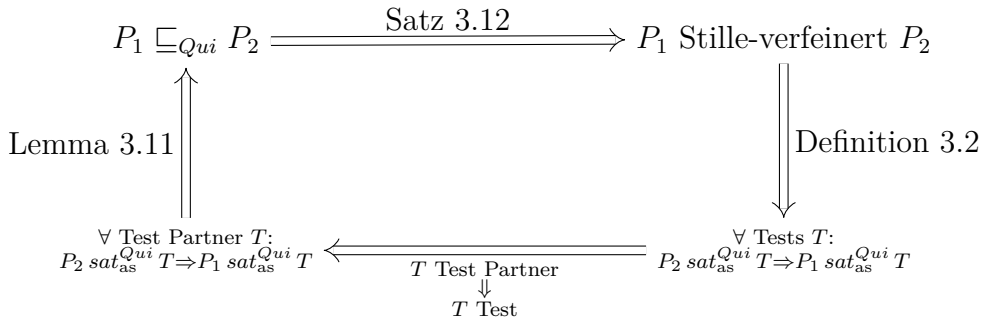

Abbildung 3.5: Folgerungskette der Testing-Verfeinerung und Stillstands-Relation

3.2 Hiding

Es soll nun auch für die Relation $\sqsubseteq_{Qui}$ die Auswirkungen des Hiding-Operators untersucht werden. Outputs, die dabei in der Menge X enthalten sind, werden in interne Aktionen umgewandelt. Die Definition der stillen Zustände verlangt, dass keine ausgehenden must-Transitionen für lokale Aktionen an einem Zustand existieren dürfen, damit er als still gilt. Die Menge *Qui* bleibt also unter Hiding erhalten. Da die Trace-Definitionen lokale Erreichbarkeit für die Relevanz von fehlerhaften Zuständen verwenden, sollte sich unter der Internalisierung von Aktionen aus der Menge X nicht an der Relevanz von fehlerhaften Zuständen ändern.

Satz 3.14 (*Stillstands-Präkongruenz bzgl. Internalisierung*). *Seien P_1 und P_2 zwei MEIOs für die $P_1 \sqsubseteq_{Qui} P_2$ gilt, dann folgt auch die Gültigkeit von $P_1/X \sqsubseteq_{Qui} P_2/X$. Die Relation $\sqsubseteq_{Qui}$ ist also ein Präkongruenz bezüglich $\cdot/\cdot$. Es gilt für die Sprachen und Traces:*

(i) $L(P/X) = \Big\{w \in (\Sigma\backslash X)^* \mid \exists w' \in L(P) : w'|_{\Sigma\backslash X} = w\Big\}$,

(ii) $ET(P/X) = \Big\{w \in (\Sigma\backslash X)^* \mid \exists w' \in ET(P) : w'|_{\Sigma\backslash X} = w\Big\}$,

(iii) $EL(P/X) = \Big\{w \in (\Sigma\backslash X)^* \mid \exists w' \in EL(P) : w'|_{\Sigma\backslash X} = w\Big\}$,

(iv) $StQT(P/X) = \Big\{w \in (\Sigma\backslash X)^* \mid \exists w' \in StQT(P) : w'|_{\Sigma\backslash X} = w\Big\}$,

(v) $QET(P/X) = \Big\{w \in (\Sigma\backslash X)^* \mid \exists w' \in QET(P) : w'|_{\Sigma\backslash X} = w\Big\}$.

Beweis. Zunächst sollen hier die Aussagen (i) bis (v) beweisen werden. Die Punkte (i) bis (iii) wurden bereits in Satz 2.21 entsprechend nachgewiesen. Es müssen also nur noch (iv) und (v) gezeigt werden.

(iv) Jeder Trace aus *StQT* ist vom Startzustand aus in dem jeweiligen Transitionssystem ausführbar und es wird dadurch ein Zustand p erreicht, der keine ausgehenden must-Transitionen für lokale Aktionen besitzt. Der Hiding-Operator kann

keine Modalitäten von Transitionen verändern. Ein stiller Zustand in P bzw. P/X muss also auch in P/X bzw. P still sein. Die entsprechenden Zusammenhänge der Traces in den Systemen folgen analog zum Beweis von Satz 2.21 aus Lemma 2.20.

(v) Die Menge QET ist die Vereinigung der Trace-Mengen ET und $StQT$. Aus (ii) und (iv) folgt also auch die Aussage diese Punktes.

Aus $P_1 \sqsubseteq_E P_2$ folgt mit Satz 2.21 $P_1/X \sqsubseteq_E P_2/X$. Um die analoge Folgerung für $\sqsubseteq_{Qui}$ zu begründen, muss wegen Definition 3.5 also nur noch nachgewiesen werden, dass $QET_{P_1/X} \subseteq QET_{P_2/X}$ unter der Voraussetzung der Inklusion $QET_1 \subseteq QET_2$ gilt. Dies folgt aus der Aussage des Punktes (iv).
Die Relation $\sqsubseteq_{Qui}$ bleibt also unter der Anwendung des Hiding-Operators erhalten und ist somit bezüglich diesem eine Präkongruenz. □

In Definition 1.9 wurde mit Hilfe des Hiding-Operators aus der Parallelkomposition ohne Verbergen die Parallelkomposition mit Internalisierung der synchronisierten Aktionen nachgebildet. Die Präkongruenz-Eigenschaft bezüglich $\sqsubseteq_{Qui}$ der Parallelkomposition $\cdot|\cdot$ kann aus den Präkongruenz-Eigenschaften von $\cdot\|\cdot$ und $\cdot/\cdot$ bezüglich der Relation $\sqsubseteq_{Qui}$ aus dem Korollar 3.10 und dem Satz 3.14 geschlossen werden.

Korollar 3.15 (*Stillstands-Präkongruenz mit Internalisierung*). *Die Relation $\sqsubseteq_{Qui}$ ist eine Präkongruenz bezüglich $\cdot|\cdot$.*

3.3 Zusammenhänge

Satz 3.16 (*Zusammenhang der Verfeinerungs-Relationenen mit der Stillstands-Relation*). *Für MEIOs P und Q gilt $P \sqsubseteq_{\text{as}} Q \Rightarrow P \sqsubseteq_{Qui} Q \Rightarrow P \sqsubseteq_E Q$. Die Implikationen in die andere Richtung gelten jedoch nicht. Die Relationen $\sqsubseteq_{\text{w-as}}$ und $\sqsubseteq_{Qui}$ sind unvergleichbar.*

Beweis.
$P \sqsubseteq_{\text{as}} Q \Rightarrow P \sqsubseteq_{Qui} Q$:
Im Beweis des Satzes 2.23 wurde bereits bewiesen, dass eine schwache as-Verfeinerungs-Relation $\mathcal{R}$ zwischen P und Q die Eigenschaften der Fehler-Relation $\sqsubseteq_E$ erfüllt. Nach Lemma 1.16 ist jede starke as-Verfeinerungs-Relation auch eine schwache. Es fehlt also für diese Implikation 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. Die as-Verfeinerungs-Relation $\mathcal{R}$ zwischen P und Q ist in diesem Fall als stark anzunehmen. Für ein Wort w aus $StQT_P \backslash ET_P$ gibt es einen ausführbaren Trace der Form wie in Lemma 2.7, der mit p_n einen stillen Zustand erreicht. Es gilt $w \in \text{cont}(StET_Q) \subseteq ET_Q \subseteq QET_Q$, falls ein Fehler-Zustand in Q auf einem Präfix-Trace von w auftritt, wegen Lemma 2.7. Im folgenden wird davon ausgegangen, das w in Q ohne das erreichen eines Fehler-Zustandes ausführbar ist. Wegen 2.7 gibt es also einen Ablauf für w in Q, der in

einem Zustand q_n mit $p_n \mathcal{R} q_n$ endet. p_n ist für P ein stiller Zustand, es gilt also für alle $\omega \in (O \cup \{\tau\})$ $p_n \overset{\omega}{\not\rightarrow}_P$. Da $\mathcal{R}$ eine starke as-Verfeinerungs-Relation ist, muss wegen 1.3.2 auch $q_n \overset{\omega}{\not\rightarrow}_Q$ 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 $P \sqsubseteq_E Q$.

$P \sqsubseteq_{\text{w-as}} Q \not\Rightarrow P \sqsubseteq_{Qui} Q$:
Diese Implikation scheitert dran, dass für stille Zustände keine τ-must-Transitionen zulässig sind und $\sqsubseteq_{Qui}$ keine Divergenz mit betrachtet. Ein entsprechenden Gegenbeispiel ist in Abbildung 3.6 dargestellt. Da hier nur die strikten Stille-Trace das Problem erzeugen, soll $I = \emptyset$ gelten, damit die MIT-Mengen für beide Systeme leer sind. $\mathcal{R} = \{(p_0, q_0)\}$ ist eine schwache as-Verfeinerungs-Relation zwischen P und Q. Beides Transitionssysteme enthalten keine Fehler-Zustände, P besitzt keine Transitionen und Q nur eine Transition für eine interne Aktion, somit sind die Punkt 1.,2.,4. und 5. der Definition 1.4 für $\mathcal{R}$ sicher erfüllt. Der dritte Punkt von 1.4 fordert, dass die Transition $q_0 \xrightarrow{\tau}_Q q_0$ in P schwach gematched wird. Da $\hat{\tau}$ jedoch ε entspricht, muss es keine echte interne Transition in P für p_0 geben. Die Definition 1.4.3 ist also ebenfalls durch das bereits enthaltene Tupel (p_0, q_0) erfüllt.
Für P ist ε ein strikter Stille-Trace. Die Menge $StQT(Q) = QET(Q)$ ist jedoch leer. Die Inklusion $QET(P) \subseteq QET(Q)$, die für $P \sqsubseteq_{Qui} Q$ gelten müsste ist also nicht erfüllt.

Abbildung 3.6: Gegenbeispiel zu $\sqsubseteq_{\text{w-as}} \Rightarrow \sqsubseteq_{Qui}$ mit $I_P = I_Q = \emptyset$

$P \sqsubseteq_{\text{w-as}} Q \not\Leftarrow P \sqsubseteq_{Qui} Q$:
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 3.7 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 schwache as-Verfeinerungs-Relation zwischen P und Q sein.

Abbildung 3.7: Gegenbeispiel zu $\sqsubseteq_{\text{w-as}} \Leftarrow \sqsubseteq_{Qui}$

$P \sqsubseteq_{\text{as}} Q \not\Leftarrow P \sqsubseteq_{Qui} Q$:
Falls diese Implikation gelten würde, würde jedes Paar von MEIOs, dass in der Relation $\sqsubseteq_{Qui}$ steht auch in der Relation $\sqsubseteq_{\text{as}}$ stehen. Mit Lemma 1.16 würde draus folgen, dass das MEIO Paar auch in der Relation $\sqsubseteq_{\text{w-as}}$ stehen muss Dies stellt jedoch ein Widerspruch zur Unvergleichbarkeit von $\sqsubseteq_{Qui}$ und $\sqsubseteq_{\text{w-as}}$ dar.

$P \sqsubseteq_{Qui} Q \not\Leftarrow P \sqsubseteq_{E} Q$:
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 3.8 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 strikten 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 3.8: Gegenbeispiel zu $\sqsubseteq_{Qui} \Leftarrow \sqsubseteq_{E}$ mit $I_P = I_Q = \emptyset$

□

Alternativ wäre es auch möglich gewesen eine andere Betrachtung zu wählen, die für stille Zustände zunächst nur fordert, dass keine must-Outputs möglich sein dürfen. 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. Die Menge der stillen Zustände würde in der alternativen Betrachtung durch $\left\{p \in P \mid \forall a \in O : p \overset{a}{\Rightarrow}_P\right\}$ beschrieben werden. Diese Menge ist in der hier betrachteten Menge der stillen Zustände $Qui(P)$ enthalten. Zusätzlich zu den in dieser Arbeit betrachteten Verklemmungen der Art Deadlock, lässt die Betrachtungsweise mit den zugelassenen τ-must-Transitionen auch Verklemmungen der Art Livelock zu, 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. 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. Es ist unklar zu welchen Konsequenzen dies führen kann. Vor allem im Bezug auf Hiding ist es schwierig, da unterschiedliche Transitionen in einem System zu unterschiedlichem Verhalten führen, obwohl sie von der Transitionsbeschriftung her nicht zu unterscheiden sind. Hierzu ist auch das Beispiel in Abbildung 3.9 zu beachten. Die Sprachen beider MEIOs sind gleich und auch die strikten stille Traces stimmen überein, da jeweils durch beliebig viele os (mindestens eins), die von einem o' gefolgt werden eine ruhiger Zustand erreicht wird. Bei P ist dies immer möglich. Falls aber in Q die erste o Transition nach rechts ausgeführt wird, kann kein stiller Zustand mehr erreicht werden. Die MEIOs können also auf Trace-Ebenen nicht von einander unterschieden werden, jedoch können durch lokal Entscheidungen sehr verschiedenen Ergebnisse entstehen. Falls man das o durch Hiding in ein τ umwandelt, wäre p_0 und p_1 sicher nicht still. q_{12} hingegen würde in der alternativen Betrachtung still werden. Die davor über die Trace-Mengen nicht unterscheidbaren Systeme würden durch Hiding dann einen Unterschied in den Trace-Mengen aufweisen.

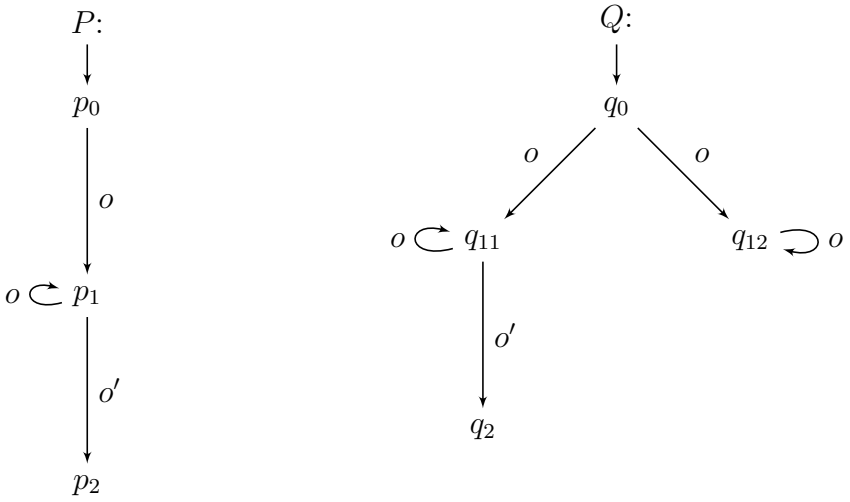

Abbildung 3.9: Beispiel für Probleme bei Trace-Betrachtung

Im nächsten Kapitel, in dem Zustände mit Divergenz betrachtet werden, heben sich die Unterschiede in den Trace-Mengen, die die beiden unterschiedlichen Ansätze hier erzeugen würden auf. Da die Zustände die eine unendliche Folge an τs ausführen können als divergent betrachtet werden und die Stille als nicht so schlimm angesehen wird und auf Trace-Ebene somit geflutet wird.

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

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

4.1 Testing-Ansatz

Für die EIO aus z.B. [Sch16] sind divergente Zustände Zustände, die eine unendliche Folge an τs ausführen können. Da die Definition für Implementierung hier das gleiche liefern soll, müssen alle Zustände, die eine unendliche Folge an τs sicherstellen müssen auf jeden Fall divergent sein. Jedoch lassen bereits unendliche Folgen von τ-may-Transitionen Divergenz in einer ihrer as-Implementierungen zu. Somit erscheint es sinnvoll Zustände bereits als divergent anzusehen, wenn sie die Möglichkeit für eine unendliche Folge von τs via may-Transitionen besitzen. Die Korrektheit dieser Idee wird im folgenden nachgewiesen.

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 einen Kreis von Zuständen, die via τ-Transitionen verbunden sind, von einem durch interne Aktionen erreichbaren Zustand ausführbar sein oder durch einen unendlichen 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.
Für die Relevant eines Divergenz-Zustandes in einem Transitionssystem wieder wieder der optimistische Ansatz der lokalen Erreichbarkeit verwendet. Auf Implementierungen, die den EIOs in z.B. [Sch16] entsprechen, ist Divergenz nicht mehr verhinderbar, sobald ein divergenter Zustand lokal erreichbar ist. Somit wird sich auch hier

herausstellen, dass Divergenz als ähnlich „schlimm“ zu bewerten ist wie ein Fehler-Zustand.

Definition 4.2 (*Test und Verfeinerung für Divergenz*). *Sei P ein MEIO.*
P ist lokal fehler-, stillstand- *und* divergenz-frei*, wenn kein Fehler-, stiller oder Divergenz-Zustand lokal erreichbar ist.*
Ein Test *T für P ist eine zu P komponierbare Implementierung. P* as-erfüllt *T als einen Divergenz-Test, falls $S \| T$ lokal fehler-, stillstand- und divergenz-frei ist für alle $S \in$ as-impl(P). Es wird dann $P \, sat_{\text{as}}^{Div} \, T$ geschrieben.*
Ein MEIO P Divergenz-verfeinert *P', falls sie die selbe Signatur haben und für alle ihre Tests T: $P' \, sat_{\text{as}}^{Div} \, T \Rightarrow P \, sat_{\text{as}}^{Div} \, T$.*

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 wohl als ähnlich „schlimmes“ Fehlverhalten anzusehen ist wie Fehler. Da das Divergieren eines Systems nicht mehr verhinderbar ist, sobald ein divergenter Zustand lokal erreichbar ist, kommt für die Divergenz-Traces wieder die prune und letztlich auch die cont-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 später semantisch 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*: $StDT(P) := \left\{ w \in \Sigma^* \mid p_0 \stackrel{w}{\Rightarrow}_P p \in Div(P) \right\}$,*
- 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 1.10 begründet erfüllt $\mathcal{R}$ alle Punkte der Definition 1.3. Es wird ein w aus $StDT(P)$ betrachtet. Es gibt also einen unendliche Ablauf in P auf einem endlichen Anfangsstück dieses Ablaufes wird das Wort w ausgeführt, danach werden nur noch τ-Transitionen ausgeführt. Es gilt also $\exists w' \in \Sigma_\tau^*, \exists \alpha_1, \alpha_2, \ldots, \alpha_n, \exists p_1, p_2, \ldots, p_n : \hat{w'} = w \wedge w' = \alpha_1\alpha_2 \ldots \alpha_n \wedge p_0 \overset{\alpha_1}{\dashrightarrow}_P p_1 \overset{\alpha_2}{\dashrightarrow}_P \ldots p_{n-1} \overset{\alpha_n}{\dashrightarrow}_P p_n \in Div_P$. Nach p_n kann der Trace durch eine unendliche τ-Folge fortgesetzt werden. Die Identitäts-Relation $\mathcal{R}$ setzt die Zustände des Traces mit den analogen Zuständen aus P' in Relation. Mit der Implementierung aller Transitionen aus P in P' ergibt sich der selbe Trace in P'. Es gilt also $p'_0 \overset{\alpha_1}{\dashrightarrow}_{P'} p'_1 \overset{\alpha_2}{\dashrightarrow}_{P'} \ldots p'_{n-1} \overset{\alpha_n}{\dashrightarrow}_{P'} p'_n$ mit $(p'_j, p_j) \in \mathcal{R}$ für $0 \leq j \leq n$. Da alle Transitionen von P in P' übernommen wurden hat p'_n ebenso wie p_n die Möglichkeit eine unendliche Folge an τs auszuführen. P' kann also den analogen unendliche Ablauf zu P ausführen. Es gilt also $p'_n \in Div_{P'}$ und somit $w \in StDT(P')$. Insgesamt folgt also für dieses 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 aus 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*). *Sei 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 Proposition 4.4.2. Aus der Monotonie von cont folgt also auch die hier geforderte Inklusion.

2. „$\subseteq$“:

$$\begin{aligned} EDT(P) &\overset{4.5}{=} ET(P) \cup DT(P) \\ &\overset{2.9.1}{=} \left(\bigcup_{P' \in \text{as-impl}(P)} ET(P') \right) \cup DT(P) \\ &\overset{1.}{\subseteq} \left(\bigcup_{P' \in \text{as-impl}(P)} ET(P') \right) \cup \left(\bigcup_{P' \in \text{as-impl}(P)} DT(P') \right) \\ &= \bigcup_{P' \in \text{as-impl}(P)} ET(P') \cup DT(P') \\ &\overset{4.5}{=} \bigcup_{P' \in \text{as-impl}(P)} EDT(P'). \end{aligned}$$

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 2.9. Es ist also nur noch der Fall zu betrachten, in dem $w \in DT(P') \backslash ET(P')$ gilt. Es gibt also einen unendlichen Ablauf analog zu dem in Korollar 2.8 für $wv \in StDT'_P$ mit $v \in O^*$, wobei nur endlich viele Transitionen benötigt werden bis die letzte sichtbare Aktion aus wv ausgeführt wird und danach unendlich viele τ-Transitionen folgen. Zwischen P' und P soll $\mathcal{R}$ als as-Verfeinerungs-Relationen gelten. Falls in dem matchenden Ablauf in P ein Fehler-Zustand erreicht wird, muss nach Korollar 2.8 ein Präfix von wv in $StET_P$ enthalten sein. Mit $w = \text{prune}(wv)$ folgt somit $w \in ET_P \subseteq EDT_P$. Falls in P jedoch ein mit wv beschrifteter Trace ausführbar ist und kein mit wv beschrifteter Trace oder ein Präfix davon einen Fehler-Zustand erreicht, muss es ein p_j in P geben, dass durch einen wv-Trace in P erreicht wird und das mit einem p'_j in der Relation $\mathcal{R}$ steht, dass in dem unendlichen Ablauf von P' nach der letzten sichtbaren Aktion auftritt. p'_j ist also ein divergenter Zustand für P' für den es einen unendlichen Ablauf gibt, der nur τ-Transitionen enthält und es gilt $p'_j \mathcal{R} p_j$. Mit Korollar 2.8 muss P diesen unendlichen Ablauf von τs von p_j aus matchen können. Es gilt also $p_j \in Div_P$ 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 2.9.

4. „$\subseteq$“:

$$\begin{aligned}
EDL(P) &\overset{4.5}{=} L(P) \cup EDT(P) \\
&\overset{1.10}{\subseteq} \left(\bigcup_{P' \in \text{as-impl}(P)} L(P') \right) \cup EDT(P)
\end{aligned}$$

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

4. „$\supseteq$“:
Für den Beweis dieser Inklusion kann man auf den Beweis von 2.9.2 „$\supseteq$“ 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. □

Es fällt auf, dass für Proposition 4.6.1 nicht die Gleichheit sondern nur eine Inklusions-Richtung behauptet und bewiesen wird. Dies liegt daran, dass die andere Richtung im allgemeinen nicht gilt. Die Menge DT ist zwar unter cont abgeschlossen, jedoch werden darin keine Fehler-Traces betrachtet. Es kann also das bereits erwähnte Problem auftreten, dass die Spezifikation einen Fehler besitzt und die as-Implementierung kann sobald sie einen Zustand mit diesem Fehler-Zustand in Relation setzt beliebiges tun kann. Ein Gegenbeispiel, in dem dieses Problem verdeutlicht wird, ist in Abbildung 4.1 dargestellt. Eine as-Verfeinerungs-Relation zwischen P' und P ist $\mathcal{R} = \{(p'_0, p_0)\}$. Die Definition 1.3 stellt an $\mathcal{R}$ keine Forderungen, da $p_0 \in E_P$ gilt. P' ist also eine as-Implementierung von P. Die Menge $DT(P)$ ist leer, jedoch entspricht die Menge $DT(P')$ der Menge Σ^*.

Abbildung 4.1: Gegenbeispiel zu $\bigcup_{P' \in \text{as-impl}(P)} DT(P') \subseteq DT(P)$

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 2.10 (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-, stiller 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 keine Einschränkung von $\sqsubseteq_E$ so wie $\sqsubseteq_{Qui}$. Diese Tatsache wird später im Satz 4.21 durch die Beispiele, wieso diese Relationen unvergleichbar sind noch einmal verdeutlicht. 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. $EDL_{12} = (EDL_1 \| EDL_2) \cup EDT_{12}$.

Man könnte diesen Satz analog zu den Sätzen 2.11 und 3.9 durch die Mengen-Inklusionen in beide Richtungen beweisen. Alternativ kann man jedoch alle beteiligen MEIOs Normieren, so dass sie divergenz-trivial sind. Divergenz-trivial ist ein MEIO,der keinen divergenten Zustand mehr enthält, der nicht auch in der Menge der Fehler-Zustände enthalten ist. Für einen MEIO, für den $Div \subseteq E$, gilt $EDT = ET$, $QDT = QET$ und $EDL = EL$.

Es wäre also der Satz 3.9 anwendbar für die Traces der Parallelkomposition. Dann wäre nur noch zu zeigen, dass die Parallelkomposition von zwei in dieser Art normierten MEIOs die Normierung der Parallelkomposition ihrer unnormierten MEIO-Versionen ist.

Definition 4.9 (*Normierung zur Divergenz-Trivialität*). *Ein MEIO P ist* divergenz-trivial*, wenn $Div_P \subseteq E_P$ gilt.*
Die divergenz-trivial Form von P ist ein MEIO $DF(P)$, das man aus P erhält, in dem man die Menge der Fehler-Zustände $E_{DF(P)}$ als die Menge $E_P \cup Div_P$ definiert.

Die Menge MIT_P, $StQT_P$ und L_P eines MEIO P bleiben durch die Konstruktion in 4.9 zur divergenz-trivialen Form $DF(P)$ unverändert. Nur die Menge cont($PrET_P$) wird zur Menge cont($PrET_{DF(P)}$) erweitert, in dem eine Vereinigung mit DT_P vorgenommen wird. Semantisch ist jedoch nur die Trace-Menge EDT relevant, die bereits die Vereinigung der Mengen ET und DT ist. Die Mengen QDT und EDL werden für beide MEIOs mit der Trace Menge EDT geflutet, somit gilt für alle in der Semantik der Relation $\sqsubseteq_{Div}$ relevanten Traces die Gleichheit zwischen den Traces des MEIOs P und seiner divergenz-trivialen Form $DF(P)$.

Proposition 4.10 (*Divergenz-Trivialität*). *Jedes MEIO P ist äquivalent zu seiner divergenz-trivialen Form $DF(P)$ bezüglich der Relation $\sqsubseteq_{Div}$.*

Wie schon oben erwähnt, sollen die Trace aus Satz 4.8 auf die des Satz 3.9 zurück geführt werden durch die Normierung Systeme in divergenz-triviale Form. Dazu müssen wir zeigen, dass die Relationen $\sqsubseteq_{Div}$ und $\sqsubseteq_{Qui}$ für divergenz-triviale Systeme übereinstimmen.

Lemma 4.11 (*Relationen unter Divergenz-Trivialität*). *Für zwei MEIOs P und Q in divergenz-trivialer Form gilt: $P \sqsubseteq_{Div} Q \Leftrightarrow P \sqsubseteq_{Qui} Q$.*

Beweis. Da für einen MEIO P in divergenz-trivialer Form $Div_P \subseteq E_P$ gilt, ist jeder strikte Divergenz-Trace von P auch ein strikter Fehler-Trace von P. Auf beide Trace-Mengen wird die Kürzungs-Funktion prune und danach die Verlängerungs-Funktion cont angewendet. In EDT_P führt die Erweiterung der Menge ET_P um die Menge DT_P somit zu keiner Vergrößerung. Es gilt also $EDT_P = ET_P$. Die Menge $StQT_P$ und L_P werden im Fall $\sqsubseteq_{Div}$ mit der Menge EDT_P und im Fall $\sqsubseteq_{Qui}$ mit der Menge ET_P geflutet. Da die Mengen mit denen geflutet wird gleich sind, gilt auch $QDT_P = QET_P$ und $EDL_P = EL_P$. Die Mengen Gleichheiten gelten analog auch für Q. Da die Mengen für die Inklusionen, die die Relationen $\sqsubseteq_{Div}$ und $\sqsubseteq_{Qui}$ fordern, gleich sind, sind auch die Relationen äquivalent für MEIOs in divergenz-trivialer Form. □

Lemma 4.12 (*Divergenz-Trivialität und Parallelkomposition*). *Die Parallelkomposition $P_{12} = P_1 \| P_2$ zweier MEIOs P_1 und P_2 in divergenz-trivialer Form ist ebenfalls in divergenz-trivialer Form.*

Beweis. Es gilt $Div_j \subseteq E_j$ für beide $j \in \{1, 2\}$. In einer Parallelkomposition entsteht ein Divergenz-Zustand nur, wenn mindestens eine der beiden Komponenten bereits divergent war für den Zustand, der an der Komposition des neuen Zustandes beteiligt ist. Div_{12} entspricht also der Menge $(Div_1 \times P_2) \cup (P_1 \times Div_2)$. Da geerbte Fehler-Zustände in einer Parallelkomposition auch nur von einem der zu komponierenden Zustände geerbt werden müssen, gilt eine analoge Formel für die Menge der Fehler-Zustände. Es folgt also $Div_{12} \subseteq E_{12}$. P_{12} ist also auch in divergenz-trivialer Form. □

Lemma 4.13 (*Parallelkomposition und Divergenz-Trivialität*). *Für komponierbare MEIOs P_1 und P_2 gilt $DF(P_1 \| P_2) = DF(P_1) \| DF(P_2)$.*

Beweis. Die Normierung auf eine divergenz-triviale Form ändert nichts an den Transitionen des Systems. Stille Zustände und Zustände, die einen Input nicht sicher stellen bleiben also unverändert erhalten. Bezüglich dieser Art des Fehlverhaltens der Systeme kommutiert die Parallelkomposition mit der Normierungs-Konstruktion aus Definition 4.9. Man muss also nur noch vergleichen, dass die Fehler-Zustände in $DF(P_1 \| P_2)$ und $DF(P_1) \| DF(P_2)$ übereinstimmen, da in beiden Systemen die Divergenz-Zustände gleichzeitig auch Fehler-Zustände sind wegen Lemma 4.12.

$E_{DF(P_1 \| P_2)} \subseteq E_{DF(P_1) \| DF(P_2)}$:
Sei (p_1, p_2) ein beliebiger Fehler-Zustand aus der Menge $E_{DF(P_1 \| P_2)}$. Dieser Zustand kann in P_{12} ein geerbter, neuer Fehler oder divergenter Zustand sein.

- Fall 1 ((p_1, p_2) ist ein geerbter Fehler in P_{12}): OBdA ist p_1 in E_1 enthalten. Mit der Konstruktion aus Definition 4.9 gilt $p_1 \in E_{DF(P_1)}$. In der Parallelkomposition $DF(P_1) \| DF(P_2)$ ist (p_1, p_2) ebenfalls ein geerbter Fehler.

- Fall 2 ((p_1, p_2) ist ein neuer Fehler in P_{12}): OBdA stellt p_1 einen Input nicht via must-Transition sicher, den p_2 in P_2 also Output besitzt. Da durch die Normierung auf $DF(P_1)$ und $DF(P_2)$ die Transitionen unverändert bleiben. Kommt es zwischen p_1 und p_2 auch in der Parallelkomposition von $DF(P_1)$ und $DF(P_2)$ zu einem neuen Fehler.

- Fall 3 ($(p_1, p_2) \in Div_{12}$): Damit (p_1, p_2) ein divergenter Zustand ist, muss oBdA p_1 in P_1 ein divergenter Zustand sein. Durch die Konstruktion der divergenz-trivialen Form, wird p_1 in die Menge $E_{DF(P_1)}$ eingefügt. Der Zustand (p_1, p_2) ist somit in $DF(P_1) \| DF(P_2)$ ein geerbter Fehler.

$E_{DF(P_1) \| DF(P_2)} \subseteq E_{DF(P_1 \| P_2)}$:
Wähle einen beliebigen Zustand (p_1, p_2) aus der Menge $E_{DF(P_1) \| DF(P_2)}$. Es kann sich bei (p_1, p_2) in $DF(P_1) \| DF(P_2)$ um einen geerbten oder neuen Fehler handeln.

- Fall 1 ((p_1, p_2) geerbter Fehler in $DF(P_1) \| DF(P_2)$): Der Fehler ist oBdA von P_1 geerbt worden. Es gilt also $p_1 \in E_{DF(P_1)}$. p_1 kann erst durch die Normierung zu einem Fehler-Zustand geworden sein oder bereits davor in E_1 enthalten gewesen sein.

  - Fall 1a) ($p_1 \in E_1$): In P_{12} ist (p_1, p_2) ein geerbter Fehler. Da $E_{12} \subseteq E_{DF(P_1 \| P_2)}$ gilt, ist (p_1, p_2) auch in $E_{DF(P_1 \| P_2)}$ enthalten.

  - Fall 1b) ($p_1 \in Div_1$): Der Zustand (p_1, p_2) kann ebenso wie p_1 ein unendliche Folge von τs ausführen. Es gilt also $(p_1, p_2) \in Div_{12} \subseteq E_{DF(P_1 \| P_2)}$.

- Fall 2 ((p_1, p_2) neuer Fehler in $DF(P_1) \| DF(P_2)$): OBdA gilt $p_1 \stackrel{a}{\not\rightarrow}_{DF(P_1)}$ und $p_2 \stackrel{a}{\dashrightarrow}_{DF(P_2)}$ für ein a aus $I_1 \cap O_2$. Da die Konstruktion aus Definition 4.9 die Transitionen nicht verändern kann, muss p_1 in P_1 bereits a nicht sichergestellt haben und p_2 muss auch in P_2 den Output a ausführen können. In P_{12} ist (p_1, p_2) ebenfalls ein neuer Fehler und es gilt mit der Argumentation von Fall 1a) $(p_1, p_2) \in E_{DF(P_1 \| P_2)}$.

□

Beweis zu Satz 4.8.
Durch Proposition 4.10 kann man jeden MEIO P so normieren, dass er eine divergenz-triviale Form besitzt und das Ergebnis der Normierung äquivalent zu P ist bezüglich der Relation $\sqsubseteq_{Div}$. Für einen MEIO P in divergenz-trivialer Form gilt jedoch $EDT_P = ET_P$, $QDT_P = QET_P$ und $EDL_P = EL_P$. Somit lassen sich die hier zu beweisende Aussagen auf die des Satzes 3.9 reduzieren.

1. $$\begin{aligned} EDT_{12} &\stackrel{4.10}{=} EDT_{DF(P_{12})} \\ &\stackrel{4.11}{=} ET_{DF(P_{12})} \\ &\stackrel{4.13}{=} ET_{DF(P_1) \| DF(P_2)} \\ &\stackrel{3.9.1}{=} \text{cont}(\text{prune}((ET_{DF(P_1)} \| EL_{DF(P_2)}) \cup (EL_{DF(P_1)} \| ET_{DF(P_2)}))) \\ &\stackrel{4.11}{=} \text{cont}(\text{prune}((EDT_{DF(P_1)} \| EDL_{DF(P_2)}) \cup (EDL_{DF(P_1)} \| EDT_{DF(P_2)}))) \\ &\stackrel{4.10}{=} \text{cont}(\text{prune}((EDT_1 \| EDL_2) \cup (EDL_1 \| EDT_2))), \end{aligned}$$

2. $$\begin{aligned} QDT_{12} &\stackrel{4.10}{=} QDT_{DF(P_{12})} \\ &\stackrel{4.11}{=} QET_{DF(P_{12})} \\ &\stackrel{4.13}{=} QET_{DF(P_1) \| DF(P_2)} \\ &\stackrel{3.9.2}{=} (QET_{DF(P_1)} \| QET_{DF(P_2)}) \cup ET_{DF(P_1) \| DF(P_2)} \\ &\stackrel{4.13}{=} (QET_{DF(P_1)} \| QET_{DF(P_2)}) \cup EDT_{DF(P_{12})} \\ &\stackrel{4.11}{=} (QDT_{DF(P_1)} \| QDT_{DF(P_2)}) \cup EDT_{DF(P_{12})} \\ &\stackrel{4.10}{=} (QDT_1 \| QDT_2) \cup EDT_{12}, \end{aligned}$$

3. $EDL_{12} \overset{4.10}{=} EDL_{DF(P_{12})}$
$\overset{4.11}{=} EL_{DF(P_{12})}$
$\overset{4.13}{=} EL_{DF(P_1)\|DF(P_2)}$
$\overset{3.9.3}{=} (EL_{DF(P_1)} \| EL_{DF(P_2)}) \cup ET_{DF(P_1)\|DF(P_2)}$
$\overset{4.13}{=} (EL_{DF(P_1)} \| EL_{DF(P_2)}) \cup EDT_{DF(P_{12})}$
$\overset{4.11}{=} (EL_1 \| EL_2) \cup EDT_{DF(P_{12})}$
$\overset{4.10}{=} (EL_1 \| EL_2) \cup EDT_{12}$.

□

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. Der Beweis dafür ist analog zu den Beweisen der Korollare 2.12 und 3.10 und soll deshalb hier nicht wiederholt werden.

Korollar 4.14 (*Divergenz-Präkongruenz*). *Die Relation $\sqsubseteq_{Div}$ ist eine Präkongruenz bezüglich $\cdot\|\cdot$.*

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 wird 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, so dass T fast ein Partner ist. Zur Vereinfachung von umständlichen Formulierungen im Folgenden wird hierfür nun ein neuer Begriff definiert. Der jedoch bereits als ω-Partner in z.B. [Sch16] analog für EIOs verwendet und definiert wurde.

Definition 4.15 (χ-*Partner*). *Ein MEIO P_1 ist ein χ-Partner von einem MEIO P_2, wenn $I_1 = O_2$ und $O_1 = I_2 \cup \{\chi\}$ mit $\chi \notin I_2 \cup O_2$ gilt.*

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

Lemma 4.16 (*Testing-Verfeinerung mit Divergenz*). *Gegeben sind zwei MEIOs P_1 und P_2 mit der gleichen Signatur. Wenn für alle Tests T, die χ-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 χ-Partner T gilt $I_T = O$ und $O_T = I \cup \{\chi\}$ mit $\chi \notin I \cup O$.

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 2.17 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 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- 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 Element 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 χ 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 $\varepsilon = w \in EDT_{P_2'}$ 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 χ-Partner T betrachtet (siehe auch Abbildung 4.2):
  - $T = \{p_0, p_1, \dots, p_{n+1}\}$,

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

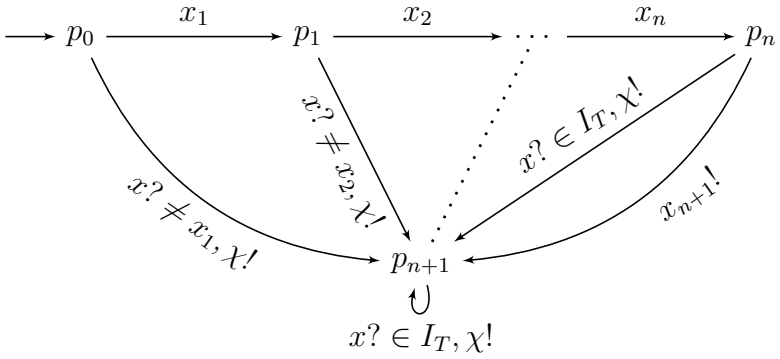

Abbildung 4.2: $x? \neq x_j$ steht für alle $x \in I_T \backslash \{x_j\}$

Die Mengen der Divergenz- und stillen Zustände des hier betrachteten Ts sind leer. Da im Vergleich zum Transitionssystem in Abbildung 2.1 nur die χ-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 2.17. Im Fall 2b) kann die Menge O^* beibehalten werden, obwohl die Parallelkomposition mit T auch χ also Output zulassen würde. Die Einschränkung, dass v keine χ Transitionen enthalten darf, verhindert, dass T unsynchronisierte Aktionen ausführt. T kann jedoch trotzdem immer alle Aktionen, die P_1' ausführen möchte synchronisieren. 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 2.17. Da nun aber auch Divergenz betrachtet wird, muss ein weiterer Fall ergänzt werden:

- Fall 2c) $\left(w \in PrDT_{P_1'}\right)$: In $P_1' \| T$ erhält man $(p_{01}, p_0) \stackrel{w}{\Longrightarrow} (p'', p_{n+1}) \stackrel{v}{\Longrightarrow} (p', p_{n+1})$ für $v \in O^*$ und $p' \in Div_1$. Die Einschränkung, dass v keine Aktionen χ enthalten darf, ist mit der gleichen Begründung wie oben möglich. Daraus folgt $(p', p_{n+1}) \in Div_{P_1' \| T}$ und somit $wv \in StDT(P_1' \| T)$. Da alle Aktionen aus w synchronisiert werden und $I_T \cap I = \emptyset$ gilt $x_1, \ldots, x_n, x_{n+1} \in O_{P_1' \| T}$. Da zusätzlich v in O^* enthalten ist, folgt $v \in O^*_{P_1' \| T}$. 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 χ-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 $a \in O_T\backslash\{\chi\}$ 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 χ 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 $x_{i+1} \in O_T\backslash\{\chi\}$. 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 $x_1 \ldots x_i v$ ausgeführt mit $v \in (O \cup \{\chi\})^*$ und ebenso hat P_2' den Weg $x_1 \ldots x_i v|_{\Sigma_2}$ ausgeführt. Durch dies hat P_2' einen Zustand aus $E_{P_2'}$ erreicht, da von T kein Fehler geerbt werden kann. Es gilt dann $\text{prune}(x_1 \ldots x_i v|_{\Sigma_2}) = \text{prune}(x_1 \ldots x_i) \in PrET_{P_2'} \subseteq ET_{P_2'}$. 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) ($\varepsilon \in DT(P_2'\|T)$): Da T keine τ-Transitionen besitzt, kann das Divergenz-Verhalten nur von P_2' geerbt sein. T hat $x_1 \ldots x_i v$ ausgeführt mit $v \in (O \cup \{\chi\})^*$ und ebenso hat P_2' den Weg $x_1 \ldots x_i v|_{\Sigma_2}$ ausgeführt. Durch dies hat P_2' einen Zustand aus $Div_{P_2'}$ erreicht. Es gilt dann $\text{prune}(x_1 \ldots x_i v|_{\Sigma_2}) = \text{prune}(x_1 \ldots x_i) \in PrDT_{P_2'} \subseteq DT_{P_2'}$, da $v|_{\Sigma_2}$ 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 χ-Partner Test T betrachtet (siehe auch Abbildung 4.3):

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

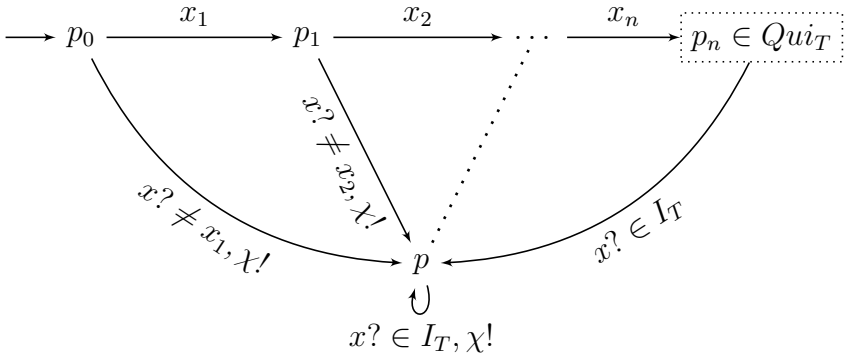

Abbildung 4.3: $x? \neq x_j$ steht für alle $x \in I_T \backslash \{x_j\}$, p_n ist der einzige stille Zustand

Falls das betrachtete w dem leeren Wort ε entspricht, reduziert sich der χ-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 einzige 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. 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 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 2.17 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 1.10 und 4.6.

- Fall 1 ($w = \varepsilon$): Analog zu den Lemmata 2.17 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 χ-Partner Tests T weicht nur durch die χ-Transitionen vom Transitionssystem aus dem Beweis der Inklusion der Fehler-gefluteten Sprache EL aus Lemma 2.17 ab. Der Test T ist dann wie folgt definiert (siehe dazu auch Abbildung 4.4):
  - $T = \{p_0, p_1, \dots, p_n, p\}$,
  - $p_{0T} = p_0$,
  - $\dashrightarrow_T = \Longrightarrow_T = \{(p_j, x_{j+1}, p_{j+1}) \mid 0 \leq j < n\}$
    $\cup \{(p_j, x, p) \mid x \in I_T\backslash\{x_{j+1}\}, 0 \leq j < n\}$
    $\cup \{(p_j, \chi, p) \mid 0 \leq j \leq n\}$
    $\cup \{(p, x, p) \mid x \in I_T \cup \{\chi\}\}$,
  - $E_T = \{p_n\}$.

  Durch die χ-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

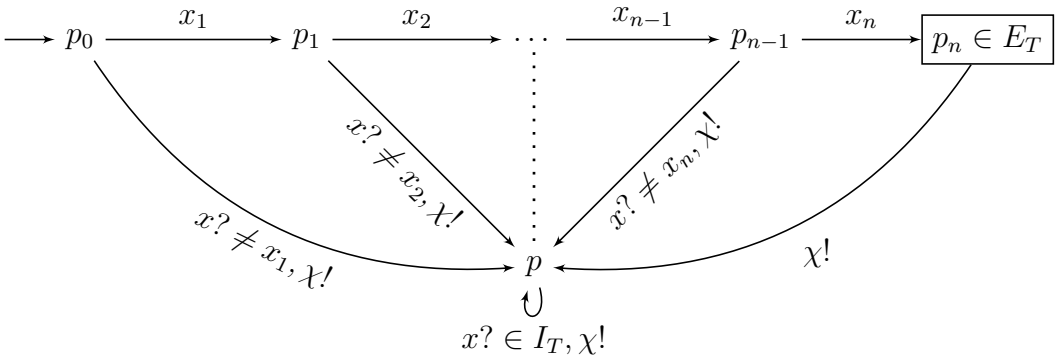

Abbildung 4.4: $x? \neq x_j$ steht für alle $x \in I_T \backslash \{x_j\}$, p_n ist der einzige Fehler-Zustand

Fehler hat. Es muss also auch eine as-Implementierung P_2' von P_2 geben, so dass $P_2' \| T$ einen lokal erreichbaren fehlerhaften Zustand hat. Wie oben bereits erwähnt, kommt Stille als Fehlerhaftigkeit nicht in Frage.

- Fall 2a) $\Big($neuer Fehler aufgrund von $x_j \in O_T \backslash \{\chi\}$ und $p_{02} \overset{x_1 \ldots x_{j-1}}{\Longrightarrow} p_2' \overset{x_j}{\not\longrightarrow}\Big)$: 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 2.17 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 $p_{02} \overset{x_1 \ldots x_j v}{\Longrightarrow}_{P_2'} p' \in Div_{P_2'}$ für $j \geq 0$ und $v \in O^*$. Somit ist $x_1 \ldots x_j v \in StDT_{P_2'}$ und damit $\text{prune}(x_1 \ldots x_j v) = \text{prune}(x_1 \ldots x_j) \in PrDT_{P_2'} \subseteq EDT_{P_2'}$. Also folgt mit 4.6.2, dass w in $EDT_2 \subseteq EDL_2$ enthalten ist, da DT unter cont abgeschlossen ist.

□

Satz 4.17. *Aus $P_1 \sqsubseteq_{Div} P_2$ folgt, dass P_1 P_2 Divergenz-verfeinert.*

Beweis. Ein Fehler-, stiller oder Divergenz-Zustand ist nach Definition genau dann in einem MEIO P lokal erreichbar, wenn $w \in QDT_P$ für ein $w \in O_P^*$ gilt.
Um nachzuweisen, dass P_1 P_2 Divergenz-verfeinert, muss bewiesen werden, dass für alle Tests T von P_1 bzw. P_2 $P_2\, sat_{\text{as}}^{Div}\, T \Rightarrow P_1\, sat_{\text{as}}^{Div}\, T$ gilt. Anstatt diese Aussage direkt zu beweisen, wird hier die äquivalente Aussage $\neg P_1\, sat_{\text{as}}^{Div}\, T \Rightarrow \neg P_2\, sat_{\text{as}}^{Div}\, T$ für alle Test T von P_1 und P_2 nachgewiesen. Sei T ein beliebiger Test T von P_1, für den $\neg P_1\, sat_{\text{as}}^{Div}\, T$ gilt. Für eine as-Implementierung P_1' von P_1 wird also ein Fehler-, stiller oder Divergenz-Zustand in $P_1' \| T$ lokal erreicht. Es muss ein $w \in O_{P_1' \| T}^*$ geben, das in der Menge $QDT_{P_1' \| T}$

enthalten ist. Ein Element der Menge $QDT_{P_1'\|T}$ kann in dieser enthalten sein, wenn es nach Definition 4.5 Element einer der zwei folgenden Mengen ist: $EDT_{P_1'\|T}$ oder $StQT_{P_1'\|T}$.

- Fall 1 ($w \in EDT_{P_1'\|T}$): Da w nur aus Outputs besteht, muss auch $\varepsilon \in EDT_{P_1'\|T}$ gelten. Mit Satz 4.8.1 folgt $\varepsilon \in \text{cont}(\text{prune}((EDT_{P_1'}\|EDL_T) \cup (EDL_{P_1'}\|EDT_T)))$. Es folgt $\varepsilon \in \text{prune}((EDT_1\|EDL_T) \cup (EDL_1\|EDT_T))$ durch Proposition 4.6 und die Präfix-Minimalität von ε. Eine Verlängerung $w' \in O^*_{P_1\|T}$ von ε um lokale Aktionen muss in der Menge $EDT_1\|EDL_T$ oder in der Menge $EDL_1\|EDT_T$ enthalten sein. Es muss also Projektionen des Wortes w auf die einzelnen Komponenten geben, so dass $w \in w_1\|w_T$ mit $w_1 \in EDT_1$ bzw. $w_1 \in EDL_1$ und $w_T \in EDL_T$ bzw. $w_T \in EDT_T$ gilt. Mit der Voraussetzung $P_1 \sqsubseteq_{Div} P_2$ folgt $w_1 \in EDT_2$ bzw. $w_1 \in EDL_2$. Wegen Proposition 4.6 muss es ein P_2' aus as-impl(P_2) geben, für dass $w_1 \in EDT_{P_2'}$ bzw. $w_1 \in EDL_{P_2'}$ gilt. In der Parallelkomposition mit T gilt also $w' \in (EDT_{P_2'}\|EDL_T) \cup (EDL_{P_2'}\|EDT_T)$. Da P_1 und P_2 die gleiche Signatur haben gilt $w' \in O^*_{P_2'\|T}$ und somit $\varepsilon \in \text{prune}((EDT_{P_2'}\|EDL_T) \cup (EDL_{P_2'}\|EDT_T)) \subseteq EDT_{P_2'\|T}$, wegen Satz 4.8.1. In der Parallelkomposition $P_2'\|T$ ist ein Fehler- oder Divergenz-Zustand lokal erreichbar. Für die Spezifikation P_2 von P_2' folgt somit also $\neg P_2\, sat^{Div}_{\text{as}}\, T$.

- Fall 2 ($w \in StQT_{P_1'\|T} \backslash EDT_{P_1'\|T}$): Das Wort w muss in $P_1'\|T$ zu einem Zustand (p_1, p_T) führen, der still und keine Fehler ist. Die Zustände p_1 und p_T, die in Komposition den Zustand in $P_1\|P_T$ ergeben müssen nach Lemma 3.8.2 ebenfalls still sein. Es gibt also Wörter w_1 und w_T, so dass w_1 in P_1' den Zustand p_1 und P_T den Zustand p_T erreicht mit $w \in w_1\|w_T$. w_1 und w_T sind also in der Menge $StQT$ ihres Transitionssystems enthalten. Für w_1 gilt $w_1 \in StQT_{P_1'} \subseteq QDT_{P_1'} \subseteq QDT_1$, wegen Proposition 4.6. $w_1 \in QDT_2$ folgt durch die Voraussetzung $P_1 \sqsubseteq_{Div} P_2$. Mit Proposition 4.6 muss es eine as-Implementierung P_2' von P_2 geben, für die w_1 in der Menge $QDT_{P_2'} = EDT_{P_2'} \cup StQT_{P_2'}$ enthalten ist.

  - Fall 2a) ($w \in EDT_{P_2'}$): Für das Wort w_T gilt $w_T \in L_T \subseteq EDL_T$, da es in T ausführbar ist. Mit Satz 4.8.1 folgt $w \in EDT_{P_2'\|T}$. Da w nur aus Outputs besteht, ist ein Fehler- oder Divergenz-Zustand in $P_2'\|T$ lokal erreichbar wie in Fall 1 dieses Beweises. Es folgt $\neg P_2\, sat^{Div}_{\text{as}}\, T$.

  - Fall 2b) ($w_1 \in StQT_{P_2'}$): Durch w_1 wird in P_2' ein stiller Zustand p_2 erreicht. In der Parallelkomposition ist der Zustand (p_2, p_T), wegen Lemma 3.8.1, ebenfalls still. Der Zustand wird durch $w_1\|w_T$ erreicht. Es gilt also $w \in StQT_{P_2'\|T}$. Das w besteht auch für die Parallelkomposition $P_2'\|T$ nur aus Outputs, da P_1 und P_2 nach Voraussetzung die selbe Signatur besitzen müssen. Es ist also ein stiller Zustand lokal erreichbar in $P_2'\|T$. Somit gilt auch in diesem Fall $\neg P_2\, sat^{Div}_{\text{as}}\, T$.

□

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 4.16 T kein Partner ist, sondern ein χ-Partner. Die Folgerungskette ist in Abbildung 4.5 dargestellt.

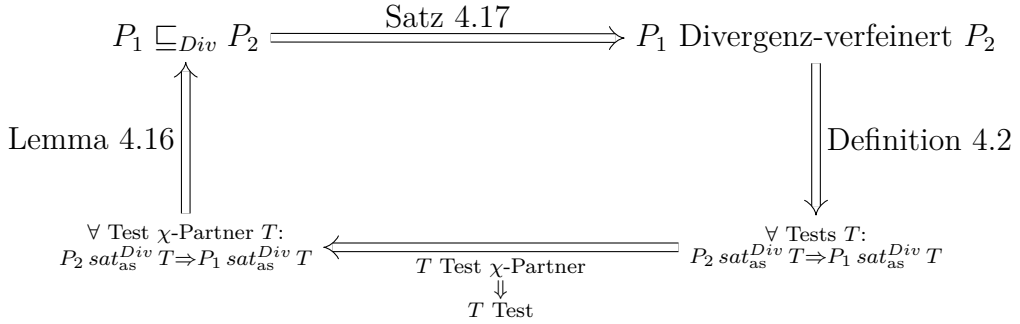

Abbildung 4.5: Folgerungskette der Testing-Verfeinerung und Divergenz-Relation

Das nächste Korollar macht die Äquivalenz, die durch die Abbildung 4.5 deutlich wird, explizit.

Korollar 4.18. *Es gilt:* $P_1 \sqsubseteq_{Div} P_2 \Leftrightarrow P_1$ *Divergenz-verfeinert* P_2.

4.2 Hiding

Der Hiding-Operator wandelt Outputs in τs um. Somit hat das Hiding auf die Divergenz-Eigenschaft im Vergleich zu den betrachteten Eigenschaften aus den beiden vorangegangen Kapiteln deutlich größere Auswirkungen. Die Menge der divergenten Zustände kann sich durch das Internalisieren vergrößern. Es kann ein Zustand divergent werden, wenn von diesem bereits lokal ein divergenter Zustand aus erreichbar war oder wenn er eine unendliche Folge von Aktionen aus $X \cup \{\tau\}$ ausführen konnte. Durch die zusätzlichen Divergenz-Zustände vergrößern sich alle Trace-Mengen, die in der Präkongruenz $\sqsubseteq_{Div}$ betrachtet werden.
Um den Zusätzlichen Aufwand der Untersuchung möglichst gering zu halten, wird dieses Teilkapitel auf endliche MEIOs beschränkt. Falls man unendlich große MEIOs zulassen würde, müsste man an anderen Stellen Endlichkeits-Voraussetzungen machen.
Die Menge X muss Teilmenge der Outputs O sein, für einen endlichen MEIO kann X ebenfalls nur endlich sein. Um eine endliche Menge von Aktionen zu internalisieren, kann man jede Aktion einzeln aus dem entsprechend MEIO entfernen. Der folgende Satz kann also darauf beschränkt werden, dass nur ein einzelnen Output verborgen werden soll. P/o soll dabei für $P/\{o\}$ stehen. Die Menge $EDT_{P/o}$ kann aus EDT_P konstruiert werden. Im allgemeinen ist die Menge $EDT_{P/o}$ jedoch größer wie die Menge $\{w \in (\Sigma\backslash\{o\})^* \mid \exists w' \in EDT(P) : w'|_{\Sigma\backslash\{o\}} = w\}$, da an Zuständen, die in P nicht divergent sind und von denen aus unendlichen Folgen von τs und os mit unendlich vielen

os möglich sind, erst durch das Hiding Divergenz entsteht. Die anderen Trace-Mengen müssen deshalb mit der neuen EDT Menge geflutet werden.

Satz 4.19 (*Divergenz-Präkongruenz bzgl. Internalisierung*). *Seien P_1 und P_2 zwei endliche MEIOs für die $P_1 \sqsubseteq_{Div} P_2$ gilt, dann folgt auch die Gültigkeit von $P_1/X \sqsubseteq_{Div} P_2/X$. Die Relation $\sqsubseteq_{Div}$ ist also ein Präkongruenz bezüglich $\cdot/\cdot$ für endliche MEIOs. Es gilt für die Sprachen und Traces:*

(i) $L(P/o) = \{w \in (\Sigma\backslash\{o\})^* \mid \exists w' \in L(P) : w'|_{\Sigma\backslash\{o\}} = w\}$,

(ii) $EDT(P/o) = \text{cont}(\text{prune}(\{w \in (\Sigma\backslash\{o\})^* \mid \exists w' : w'|_{\Sigma\backslash\{o\}} = w \wedge \forall n \geq 0 : w'o^n \in EDL(P)\}))$,

(iii) $EDL(P/o) = \{w \in (\Sigma\backslash\{o\})^* \mid \exists w' \in EDL(P) : w'|_{\Sigma\backslash\{o\}} = w\} \cup EDT(P/o)$,

(iv) $StQT(P/o) = \{w \in (\Sigma\backslash\{o\})^* \mid \exists w' \in StQT(P) : w'|_{\Sigma\backslash\{o\}} = w\}$,

(v) $QDT(P/o) = \{w \in (\Sigma\backslash\{o\})^* \mid \exists w' \in QDT(P) : w'|_{\Sigma\backslash\{o\}} = w\} \cup EDT(P/o)$.

Beweis. Die Präkongruenz-Eigenschaft lässt sich wie bei den Sätzen 2.21 und 3.14 aus den Aussagen über die Sprachen und Traces folgern. Somit sollen nun zunächst (i) bis (iv) nachgewiesen werden. Der Punkt (i) folgt aus (i) von Satz 2.21 und der Punkt (iv) aus (iv) von Satz 3.14.

(ii) „$\subseteq$“:
Beide Seiten sind abgeschlossen gegenüber prune und cont. Somit genügt es ein Element w aus $StET(P/o) \cup StDT(P/o)$ zu betrachten. Es gibt also einen mit w beschrifteten Ablauf in P/o, der zu einem Zustand p führt, der in $E_{P/o} \cup Div_{P/o}$ enthalten ist. Der selbe Zustand p kann wegen Lemma 2.20 in P durch ein w' erreicht werden mit $w'|_{\Sigma\backslash\{o\}} = w$. Falls von p aus in P ein Fehler- oder Divergenz-Zustand lokal erreichbar ist, gilt $w' \in EDT(P) \subseteq EDL(P)$. Da die Menge EDL unter cont abgeschlossen ist, gilt dann auch $w'o^n \in EDT(P) \subseteq EDL(P)$ für alle $n \in \mathbb{N}$.
Der Zustand p kann jedoch in P auch ein Zustand sein, von dem aus weder ein Fehler noch Divergenz lokal erreicht werden kann. Es muss also durch das Internalisieren des Outputs o in P/o ein Divergenz-Zustand entstanden sein. Da nur os in τs umgewandelt wurden, muss p in P eine unendliche Folge bestehend aus os und τs mit unendliche vielen os ausführen können, damit durch das Anwenden des Hiding-Operators neue Divergenz entstehen kann. Es muss also $w'o^n$ für alle $n \in \mathbb{N}$ in P ausführbar sein. Somit gilt $\forall n \geq 0 : w'o^n \in EDL(P)$.

(ii) „$\supseteq$“:
Für ein beliebiges w' für das $w'o^n \in EDL(P)$ für alle $n \in \mathbb{N}$ gilt, kann unterschieden werden, ob es ein bestimmtes $m \in \mathbb{N}$ gibt, sodass $w'o^m$ in $EDT(P)$ enthalten ist oder ob $w'o^n$ für alle $n \geq 0$ in der Sprache von P enthalten ist.

- Fall 1 ($\exists m \in \mathbb{N} : w'o^m \in EDT(P)$): Es muss ein Präfix v' von w' geben, das in $PrET(P) \cup PrDT(P)$ enthalten ist. Für $v' \in PrET(P) \subseteq ET(P)$ muss mit

Satz 2.21 (ii) $v'|_{\Sigma\setminus\{o\}}$ in $ET(P/o)$ enthalten sein. Mit einer analogen Argumentation wie in Satz 2.21 (ii) kann auch für ein v' aus $PrDT(P)$ $v'|_{\Sigma\setminus\{o\}} \subseteq DT(P/o)$ gefolgert werden. Da EDT unter der Fortsetzungs-Funktion cont abgeschlossen ist, gilt $w'|_{\Sigma\setminus\{o\}} \in EDT(P/o)$.

- Fall 2 ($\forall n \in \mathbb{N} : w'o^n \in L(P)$): Da P ein endlicher MEIO ist, muss für $n = |P|$ mindestens ein Zustand entlang des Ablaufes von o^n, der nach w' ausführbar ist, doppelt auftauchen. Der Zustand, der durch w' in P erreicht ist, ist nach dem internalisieren von o in P/o divergent. Dieser Zustand wird in P/o durch $w'|_{\Sigma\setminus\{o\}}$ erreicht, wegen Lemma 2.20. Es gilt also $w'|_{\Sigma\setminus\{o\}} \in EDT(P/o)$.

(iii) und (v) „$\subseteq$“:
Es gilt $EDL = L \cup EDT$ und $QDT = StQT \cup EDT$ für P und P/X. Somit lässt sich diese Inklusionsrichtung der Punkte analog zu den Aussagen aus Satz 2.21 und 3.14 beweisen, da der Punkt (ii) dieses Satzes bereits nachgewiesen wurde.

(iii) „$\supseteq$“:
$EDT(P/o) \subseteq EDL(P/o)$ gilt aufgrund der Definition der Menge EDL. Es sollen hier somit nur Elemente aus $\{w \in (\Sigma\setminus\{o\})^* \mid \exists w' \in EDL(P) : w'|_{\Sigma\setminus\{o\}} = w\}$ betrachtet werden. Sei w' ein beliebiges Element aus EDL_P. Nach Definition kann $w' \in L_P$ oder $w' \in EDT_P$ gelten. Für $w' \in L_P$ folgt mit (i) bereits $w = w'|_{\Sigma\setminus\{o\}} \in L_{P/o} \subseteq EDL_{P/o}$. Es wird im folgenden also davon ausgegangen, dass das w' in EDT_P enthalten ist. Es gibt also ein Präfix v' von w', dass in $PrET_P \cup PrDT_P$ enthalten ist. Mit einer analogen Begründung wie in (ii) Fall 1, folgt daraus $w'|_{\Sigma\setminus\{o\}} \in EDT_{P/o} \subseteq EDL_{P/o}$.

(v) „$\supseteq$“:
Da QDT die Vereinigung der Menge $StQT$ und EDT ist, folgt diese Inklusionsrichtung des Punktes (v) aus (iv) und der Argumentation im Beweis der selben Inklusionsrichtung des Punkts (iii).

Da P_1 und P_2 endliche sein müssen, gibt es auch nur endliche viele Outputs, die man in ihnen verbergen kann. Die Menge X muss also endlich sein. Für jedes Element o aus X kann mit den Punkten (ii) bis (v) aus $P_1 \sqsubseteq_{Div} P_2$ gefolgert werden, dass auch $P_1/o \sqsubseteq_{Div} P_2/o$ gilt. In dem man immer ein weiteren Output internalisiert. Am Ende folgt auch $P_1/X \sqsubseteq_{Div} P_2/X$ aus der Voraussetzungen $P_1 \sqsubseteq_{Div} P_2$ für endliche MEIO P_1 und P_2.
Somit bleibt die Relation $\sqsubseteq_{Div}$ unter Anwendung des Hiding-Operators erhalten und ist somit bezüglich diesem eine Präkongruenz. □

Die Parallelkomposition mit Internalisierung ist auf Basis der Parallelkomposition $\cdot\|\cdot$ und des Hiding-Operators $\cdot/\cdot$ in 1.9 definiert. Dies beiden Operatoren sind, wie in Korollar 4.14 und Satz 4.19 nachgewiesen wurde, Präkongruenzen bezüglich der Relation $\sqsubseteq_{Div}$. Somit ist auch $\cdot|\cdot$ ein Präkongruenz bezüglich dieser Relation.

Korollar 4.20 (*Divergenz-Präkongruenz mit Internalisierung*). *Die Relation $\sqsubseteq_{Div}$ ist eine Präkongruenz bezüglich $\cdot|\cdot$.*

4.3 Zusammenhänge

Satz 4.21 (*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.*
Für eine Spezifikation Q, die keinen Divergenten-Zustand erreichen kann (durch beliebige Aktionen) gilt $P \sqsubseteq_{\text{w-as}} Q \Rightarrow P \sqsubseteq_{Qui} Q$. Die umgekehrte Richtung gilt jedoch auch unter dieser Einschränkung nicht.

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 2.23 und da jede starke as-Verfeinerung auch eine schwache ist (Lemma 1.16) 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 wie in Lemma 2.7 für den der erreichte Zustand p_n in Div_P enthalten ist. Mit Lemma 2.7 muss ein Präfix von w zu einem Zustand in E_Q führen oder es gibt einen passenden Trace in Q für w. Im ersten Fall ist $w \in ET_Q \subseteq EDT_Q$. Es kann für den restlichen Beweis davon ausgegangen werden, dass das Wortes w in Q ausführbar ist. Es gibt also ein q_n, dass durch w in Q erreicht wird mit $(p_n, q_n) \in \mathcal{R}$, wegen Lemma 2.7. Von p_n ist eine unendliche Folge an τs ausführbar. Diese Folge kann wie in Korollar 2.8 als unendlicher Ablauf aufgefasst werden. Es ist also entweder ein Zustand von q_n aus mit τs erreichbar, der in E_Q enthalten ist oder von q_n aus ist ebenfalls eine unendliche Folge an τs ausführbar, wegen Korollar 2.8. 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 3.16 wurde bereits die Inklusion $StQT_P \backslash ET_P \subseteq QET_Q$ gezeigt unter der Voraussetzung, dass es eine starke as-Verfeinerungs-Relation zwischen P und Q gibt. 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 $EDL_P \backslash EDT_P \subseteq L_P$. In Satz 2.23 wurde die Inklusion $L_P \subseteq EL_Q$ unter der Voraussetzung einer schwachen as-Verfeinerungs-Relation beweisen. Es gilt also wegen der Implikation $\sqsubseteq_{\text{as}} \Rightarrow \sqsubseteq_{\text{w-as}}$ (Lemma 1.16) 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 4.6 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 somit keine Input-kritischen Traces gegen kann. Für P ist jedoch der Startzustand ein Fehler-Zustand und somit entspricht ET_P der Menge Σ^*.

Abbildung 4.6: 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 Lemma 1.16 und den Sätzen 2.23 und 3.16 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 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 4.7 verdeutliche dies. Um Input-kritische-Traces zu vermeiden wird $I = \emptyset$ vorausgesetzt. 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 schwache 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.

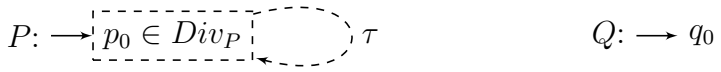

Abbildung 4.7: Gegenbeispiel zu $\sqsubseteq_{\text{w-as}} \Rightarrow \sqsubseteq_{Div}$ mit $I_P = I_Q = \emptyset$

$P \sqsubseteq_E Q \not\Rightarrow P \sqsubseteq_{Div} Q$:
Mit der Implikation $\sqsubseteq_{\text{w-as}} \Rightarrow \sqsubseteq_E$, die bereits in Satz 2.23 bewiesen wurde, kann gefolgert werden, dass das im letzten Punkt angegebenen Beispiel auch für diese Implikation anwendbar ist. Es galt $P \sqsubseteq_{\text{w-as}} Q$ somit gilt auch $P \sqsubseteq_E Q$. Die Relation $\sqsubseteq_{Div}$ hingegen ist zwischen P und Q nicht erfüllt. Es folgt also, dass die hier angegeben Implikation ebenfalls nicht gilt.

$P \sqsubseteq_{Qui} Q \not\Rightarrow P \sqsubseteq_{Div} Q$:
Das Gegenbeispiel aus Abbildung 4.7 funktioniert auch für diese Implikation jedoch mit einer abgewandelten Begründung. Unter der Voraussetzung, dass $I = \emptyset$ ist, gibt es keine Input-kritischen Traces in Q und in P. Weder Q noch P enthalten Fehler-Zustände somit gilt $ET_P = ET_Q = \emptyset$. Da die Zustände q_0 und p_0 beide keine ausgehenden must-Transitionen besitzen, sind beide still. Es gilt also $StQT_P = StQT_Q = \{\varepsilon\}$. Die Sprachen beider MEIOs bestehen nur aus dem leeren Wort. Es gelten also die notwendigen Inklusionen, damit $P \sqsubseteq_{Qui} Q$ erfüllt ist. Wie bereits begründet wurde, erfüllen P und Q jedoch nicht die Voraussetzungen für $P \sqsubseteq_{Div} Q$.

$P \sqsubseteq_{\text{w-as}} Q \Rightarrow P \sqsubseteq_{Qui} Q$ für Q erreicht keinen divergenten Zustand:
Der Beweis verläuft analog zum Beweis der Implikation $P \sqsubseteq_{\text{as}} Q \Rightarrow P \sqsubseteq_{Qui} Q$ aus Satz 3.16 bis zu dem Punkt, an dem man ein q_n in Q mit $p_n \mathcal{R} q_n$ erreicht, in dem man statt Lemma 2.7 das Lemma 2.6 anwendet und $\mathcal{R}$ eine schwache und keine starke as-Verfeinerungs-Relation ist. Für p_n als stillen Zustand gilt ebenfalls $p_n \overset{\omega}{\not\rightarrow} P$ für alle $\omega \in (O \cup \{\tau\})$. Daraus folgt mit der Definition der Relation $\Longrightarrow$ auch $p_n \overset{\hat{\omega}}{\not\Longrightarrow} P$ für alle $\omega \in O$. $\hat{\tau}$ ist für p_n auf jeden Fall via „must-Transitionen“ schwach ausführbar. Darin bestand das Problem, das das Gegenbeispiel im Satz 3.16 für diese Implikation ohne die Einschränkung für Q ausgenutzt hat. Es kann jedoch via must-Transitionen mit dem leeren Wort von p_n aus kein von p_n verschiedenen Zustand erreicht werden, da $p_n \overset{\tau}{\not\rightarrow}$ gilt. Es gilt also für alle Zustände $p \in P \backslash \{p_n\}$ $p_n \overset{\hat{\tau}}{\not\Longrightarrow} p$. Der Zustand q_n darf jedoch ausgehende τ-must-Transitionen besitzen. Jedoch kann man diesen τ-Transition so lange folgen, bis man einen Zustand q erreicht, der keine ausgehenden τ-must-Transition besitzt. Dies ist möglich, da q_n kein divergenter Zustand sein darf, da er in Q erreichbar ist. Man kann die τ-must-Transitionen aus Q schrittweise durch Definition 1.4.3 in P mit $p_n \overset{\varepsilon}{\Longrightarrow} p_n$ matchen, falls in Q durch die τ-Transitionen keine

Zustand aus E_Q erreicht wird. Falls ein Zustand auf dem Weg zu q oder q selbst in E_Q enthalten ist, gilt $w \in StET_Q \subseteq QET_Q$. Andernfalls muss mit Definition 1.4.3 $p_n \mathcal{R} q$ gelten. Der Zustand q ist in Q durch w erreichbar und es gilt $q \stackrel{\tau}{\not\rightarrow}$. Zusammen mit 1.4.3 gilt somit für alle $\omega \in (O \cup \{\tau\})$ $q \stackrel{\omega}{\not\rightarrow}$. q ist also in Qui_Q enthalten und damit folgt auch $w \in StQT_Q \subseteq QET_Q$.

$P \sqsubseteq_{\text{w-as}} Q \not\Leftarrow P \sqsubseteq_{Qui} Q$ für Q erreicht keinen divergenten Zustand:
Wenn Q eine Spezifikation ohne erreichbare Divergenz sein muss, kann das Gegenbeispiel aus Abbildung 3.7 mit einer kleinen Veränderung trotzdem verwendet werden. Die τ-Schleifen müssen durch Output-Schleifen für ein $o' \neq o$ ersetzt werden. Die Sprachen verändern sich dadurch, jedoch gilt $L(P) \subset L(Q)$ auch unter der Abwandlung. □

Literaturverzeichnis

[BFLV16] Ferenc Bujtor, Sascha Fendrich, Gerald Lüttgen, und Walter Vogler, *Nondeterministic modal interfaces*, Theor. Comput. Sci. **642** (2016), 24–53.

[BMSH10] Sebastian S. Bauer, Philip Mayer, Andreas Schroeder, und Rolf Hennicker, *On Weak Modal Compatibility, Refinement, and the MIO Workbench*, Tools and Algorithms for the Construction and Analysis of Systems, 16th International Conference, TACAS 2010, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2010, Paphos, Cyprus, March 20-28, 2010. Proceedings, 2010, pp. 175–189.

[BSV17] Ferenc Bujtor, Lev Sorokin, und Walter Vogler, *Testing Preorders for dMTS: Deadlock- and the New Deadlock-/DivergenceTesting*, ACM Trans. Embedded Comput. Syst. **16** (2017), no. 2, 41:1–41:28.

[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.

[dAH05] Luca de Alfaro und Thomas A. Henzinger, *Interface-Based Design*, pp. 83–104, Springer Netherlands, Dordrecht, 2005.

[Lar89] Kim Guldstrand Larsen, *Modal Specifications*, Automatic Verification Methods for Finite State Systems, International Workshop, Grenoble, France, June 12-14, 1989, Proceedings, 1989, pp. 232–246.

[LV13] Gerald Lüttgen und Walter Vogler, *Modal Interface Automata*, Logical Methods in Computer Science **9** (2013), no. 3.

[Mil89] Robin Milner, *Communication and Concurrency*, PHI Series in computer science, Prentice Hall, 1989.

[Sch16] Ayleen Schinko, *Kommunikationsfehler, Verklemmung und Divergenz bei Interface-Automaten*, Bachelorarbeit, Universität Augsburg, 2016.