Inhaltsverzeichnis

1 Einleitung

Der Anfang dieser Arbeit orientiert sich sehr stark an [BV14]. Jedoch wird hier darauf verzichtet die Inputmengen Der Error-IO-Transitionssysteme (EIOs) als disjunkt anzunehmen und alle Definitionen und Sätze werden erst einmal ohne das verbergen der synchronisierten Aktionen betrachtet. Anschießend, wird jedoch die Auswirkung von Hiding auf unsere Struktur untersucht und somit das Verbergen nachgebildet. Diese Art der Betrachtung der EIOs wurde auch bereits in [Sch12] gewählt, jedoch wurde dieser Arbeit nicht als direkte Quelle genutzt, bis auf den Abschnitt des Hidings. Die Feststellungen in dem Definitions-Kapitel und der Kapitel über Errors stimmen somit überein, jedoch wurden alle Beweise unabhängig davon neu geführt.
Wir wählen in dieser Arbeit einen optimistischen Ansatz für die Erreichbarkeit von Error-Zuständen. Ein Error gilt hier als erreichbar, wenn er lokal erreicht werden kann, d.h. durch lokale Aktionen. Die Menge, bestehend aus der internen Aktion τ und den Outputaktionen, bezeichnen wir hier als lokale Aktionen. Alle Elemente aus dieser Menge können ohne weiteres Zutun von außen ausgeführt werden. Somit kann nicht beeinflusst werden ob diese Übergänge genommen werden oder nicht. Es besteht also die Möglichkeit, dass das EIO in einen Error-Zustände übergeht, sobald dieser lokal erreichbar ist. Diese Art der Erreichbarkeit von Zuständen wird auch in Kapitel 3 von [BV14] behandelt.
Neben dem hier betrachteten optimistischen Ansatz gibt es noch zwei weitere Ansätze in [BV14]. Einen hyper-optimistischen Ansatz, bei dem ein Zustand als erreichbar gilt, wenn er durch interne Aktionen erreicht werden kann, und einen pessimistischen Ansatz, bei dem ein Zustand als erreichbar gilt, sobald es eine Folge an Inputs und Outputs gibt, mit denen der Zustand vom Startzustand aus erreicht werden kann. Der hyper-optimistische Ansatz wird in dieser Arbeit für die Erreichbarkeit von Quiescents verwendet.
Wir versuche bei allen Ansätzen eine gröbste Präkongruenz zu finden, die in der jeweiligen Basisrelation enthalten ist und die eine Präkongruenz bezüglich der Parallelkomposition ist.

TODO: erweitern/umformulieren (bis jetzt nur Teile aus anderen Kapitel in Einleitung verschoben)

2 Definitionen und Notationen

Die Definitionen dieses Kapitels sind größtenteils aus [BV14] übernommen. Hierbei handelt es sich um die Grundlagen der Transitionssysteme, mit denen hier gearbeitet werden soll. Jedoch wurde angepasst, dass für die Parallelkomposition die Input-Aktionen der EIOs nicht disjunkt sein müssen. Dies wäre eine unnötige Einschränkung. Die nicht mit den entsprechenden Outputs synchronisierten Inputs der zu komponierenden EIOs werden als Inputs der Parallelkomposition übernommen. Zusätzlich verzichten wir hier im Gegensatz zu [BV14] auf das verbergen der synchronisierten Aktionen.
Dadurch das die synchronisierten Aktionen nicht verborgen werden, haben wir hier ein Modell, mit dem nicht nur zwei Systeme miteinander kommunizieren können sondern beliebig viele. Ein Output eines Systems ist somit ein Art Multicast. Jedes System, dass diesen Output als Input haben kann, empfängt ihn somit auch, da bei jeder Komposition der Output weitergeleitet wird an andere Systeme. Systeme, die jedoch den Output nicht als Input haben können, werden von dieser Nachricht nicht beeinträchtigt.

2.1 Error-IO-Transitionssystem

Die hier betrachteten EIOs sind Systeme, deren Übergänge mit Inputs und Outputs beschriftet sind. Jeder Übergang ist dabei mit einem Input oder einem Output versehen. Ebenfalls zulässig ist eine Transitionsbeschriftung mit τ, einer *internen*, unbeobachtbaren *Aktion*. Diese interne Aktion lässt also keine Interaktion mit der Umwelt zu. In [BV14] entsteht das τ in vielen Fällen durch das Verbergen der Inputs und Outputs, da diese in einer Komposition synchronisiert wurden. Hier werden diese Aktionen hingegen nicht verborgen. Jedoch werden wir im weiteren Verlauf noch das Hiding betrachten, in dem Outputs durch interne Aktionen ersetzt werden.

Definition 2.1 (*Error-IO-Transitionssystem*). *Ein* Error-IO-Transitionssystem (EIO) *ist ein Tupel $S = (Q, I, O, \delta, q_0, E)$, mit den Komponenten:*

- *Q – die Menge der Zustände,*
- *I, O – die disjunkten Mengen der (sichtbaren) Input- und Outputaktionen,*
- *$\delta \subseteq Q \times (I \cup O \cup \{\tau\}) \times Q$ – die Übergangsrelation,*
- *$q_0 \in Q$ – der Startzustand,*
- *$E \subseteq Q$ – die Menge der Error-Zustände.*

Die *Aktionsmenge* eines EIOs S ist $\Sigma = I \cup O$ und die *Signatur* $Sig(S) = (I, O)$.
Um in graphischen Veranschaulichungen Inputs und Outputs zu unterscheiden wird folgende Notation verwendet: $x?$ für den Input x und $x!$ für den Output x. Falls ein x ohne ? oder ! verwendet wird, steht dies für eine Aktion, bei der nicht festgelegt ist, ob sie ein Input oder ein Output ist.
Um die Komponenten der entsprechenden Transitionssystem zuzuordnen, werden für die Komponenten die gleichen Indizes wie für ihr zugehöriges System verwendet, z.B. schreiben wir I_1 für die Inputmenge des Transitionssystem S_1. Diese Notation verwenden wir später analog für die Sprachen eines Systems.
Die Elemente der Übergangsrelation δ werden wir wie folgt notieren:

- $p \xrightarrow{\alpha} q$ für $(p, \alpha, q) \in \delta$,
- $p \xrightarrow{\alpha}$ für $\exists q : (p, \alpha, q) \in \delta$,
- $p \xrightarrow{w} q$ für $p \xrightarrow{\alpha_1} p_1 \xrightarrow{\alpha_2} p_2 \ldots \xrightarrow{\alpha_n} q$ mit $w \in (\Sigma \cup \{\tau\})^*, w = \alpha_1\alpha_2 \ldots \alpha_n$,
- $p \xrightarrow{w}$ für $p \xrightarrow{\alpha_1}\xrightarrow{\alpha_2} \ldots \xrightarrow{\alpha_n}$ mit $w \in (\Sigma \cup \{\tau\})^*, w = \alpha_1\alpha_2 \ldots \alpha_n$,
- $w|_B$ steht für die Zeichenfolge, die aus w entsteht durch Löschen aller Zeichen, die nicht in $B \subseteq \Sigma$ enthalten sind, d.h. es bezeichnet die Projektion von w auf die Menge B,
- $p \overset{w}{\Rightarrow} q$ für $w \in \Sigma^*$ mit $\exists w' \in (\Sigma \cup \{\tau\})^* : w'|_\Sigma = w \wedge p \xrightarrow{w'} q$,
- $p \overset{w}{\Rightarrow}$ für $\exists q : p \overset{w}{\Rightarrow} q$.

Die *Sprache* von S ist $L(S) = \{w \in \Sigma^* \mid q_0 \overset{w}{\Rightarrow}\}$.

2.2 Parallelkomposition

Zwei EIOs sind komponierbar, wenn ihre Outputmengen disjunkt sind. Die Error-Zustände der Parallelkomposition setzten sich aus den Error-Zuständen der beiden zusammengesetzten Komponenten (geerbte Errors) und den unsynchronisierbaren Outputs (neue Errors) zusammen.
In der folgenden Definition müssen wir eine Veränderung gegenüber [BV14] vornehmen an der Menge der synchronisierten Aktionen, da nicht mehr $I_1 \cap I_2 = \emptyset$ gelten muss, werden wir diese gemeinsamen Inputs synchronisieren.

Definition 2.2 (*Parallelkomposition*). *Zwei EIOs S_1, S_2 sind* komponierbar*, falls $O_1 \cap O_2 = \emptyset$ gilt. Die* Parallelkomposition *ist $S_1 \| S_2 = (Q, I, O, \delta, q_0, E)$ mit den Komponenten:*

- $Q = Q_1 \times Q_2$,
- $I = (I_1 \backslash O_2) \cup (I_2 \backslash O_1)$,

- $O = O_1 \cup O_2$,
- $q_0 = (q_{01}, q_{02})$,
- $$\begin{aligned}\delta =& \{((q_1,q_2),\alpha,(p_1,q_2)) \mid (q_1,\alpha,p_1) \in \delta_1, \alpha \in (\Sigma_1 \cup \{\tau\}) \backslash Synch(S_1,S_2)\} \\ & \cup \{((q_1,q_2),\alpha,(q_1,p_2)) \mid (q_2,\alpha,p_2) \in \delta_2, \alpha \in (\Sigma_2 \cup \{\tau\}) \backslash Synch(S_1,S_2)\} \\ & \cup \{((q_1,q_2),\alpha,(p_1,p_2)) \mid (q_1,\alpha,p_1) \in \delta_1, (q_2,\alpha,p_2) \in \delta_2, \alpha \in Synch(S_1,S_2)\},\end{aligned}$$
- $$\begin{aligned}E =& (Q_1 \times E_2) \cup (E_1 \times Q_2) && \text{geerbte Errors} \\ & \left.\begin{aligned} & \cup \{(q_1,q_2) \mid \exists a \in O_1 \cap I_2 : q_1 \xrightarrow{a} \wedge q_2 \not\xrightarrow{a}\} \\ & \cup \{(q_1,q_2) \mid \exists a \in I_1 \cap O_2 : q_1 \not\xrightarrow{a} \wedge q_2 \xrightarrow{a}\} \end{aligned}\right\} && \text{neue Errors.}\end{aligned}$$

Dabei werden die synchronisierten Aktionen $Synch(S_1,S_2) = (I_1 \cap O_2) \cup (O_1 \cap I_2) \cup (I_1 \cap I_2)$ *nicht versteckt, sondern als Outputs der Komposition beibehalten.*
Wir nennen S_1 einen Partner *von S_2, wenn ihre Parallelkomposition geschlossen ist, d.h. wenn sie duale Signaturen haben $Sig(S_1) = (I,O)$ und $Sig(S_2) = (O,I)$.*

Die Parallelkomposition kann nicht nur für Transitionssysteme betrachtet werden, sondern auch über Aktionsfolgen. *Traces* sind die möglichen Wege des Systems, während ein bestimmtes Wort verarbeitet wird. Dieses Wort besteht aus Inputs und Outputs, mit denen die Folge ab dem Startzustand q_0 beschriftet ist.

Definition 2.3 (*Parallelkomposition auf Traces*). *Gegeben zwei EIOs S_1 und S_2, sowie $w_1 \in \Sigma_1, w_2 \in \Sigma_2, W_1 \subseteq \Sigma_1^*, W_2 \subseteq \Sigma_2^*$:*

- $w_1 \| w_2 := \{w \in (\Sigma_1 \cup \Sigma_2)^* \mid w|_{\Sigma_1} = w_1 \wedge w|_{\Sigma_2} = w_2\}$,
- $W_1 \| W_2 := \bigcup \{w_1 \| w_2 \mid w_1 \in W_1 \wedge w_2 \in W_2\}$.

Die Semantik der späteren Kapitel basiert darauf die jeweiligen Zustände, die zu Problemen führen, mit den Traces zu betrachten, mit denen man diesem Zustände erreicht. Um dies besser umsetzten zu können, definieren wir eine *prune*-Funktion, die alle Outputs am Ende eines Traces entfernt. Zusätzlich werden Funktionen definiert, die die Traces beliebig fortsetzen.

Definition 2.4 (*Pruning- und Fortsetzungs-Funktion*). *Für ein EIO S definieren wir:*

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

Für zwei komponierbare EIOs S_1 und S_2 ist ein Ablauf ihrer Parallelkomposition $S_{12} = S_1 \| S_2$ eine Transitionsfolge der Form $(p_1,p_2) \stackrel{w}{\Rightarrow} (q_1,q_2)$ für ein $w \in \Sigma_{12}^*$. So ein Ablauf kann auf Abläufe von S_1 und S_2 projiziert werden. Diese Projektionen erfüllen $p_i \stackrel{w_i}{\Rightarrow} q_i$ mit $w|_{\Sigma_i} = w_i$ für $i = 1, 2$. Umgekehrt sind zwei Abläufe von S_1 und S_2 der Form

$p_i \overset{w_i}{\Rightarrow} q_i$ mit $w|_{\Sigma_i} = w_i$ für $i = 1, 2$, Projektionen von genau einem Ablauf in S_{12} der Form $(p_1, p_2) \overset{w}{\Rightarrow} (q_1.q_2)$. Es ist dafür nötig, dass die Abläufe der beiden Systeme und die Systeme selbst komponierbar sind. Dadurch, dass wir dann ein w wählen, dass projiziert auf die einzelnen Alphabete die jeweiligen Wörter ergibt, können wir auch sagen, dass nur ein Ablauf möglich ist. Daraus ergibt sich das folgende Lemma.

Lemma 2.5 (*Sprache der Parallelkomposition*). *Für zwei komponierbare EIOs S_1 und S_2 gilt:*

$$L_{12} := L(S_1 \| S_2) = L(S_1) \| L(S_2).$$

Dieses Lemma ist bereits in [BV14] enthalten. Hier benötigen wir jedoch keine Erwähnung der Sprache für die Parallelkomposition mit Verbergen.

2.3 Hiding

Hiding wurde bereist in [CJK13] auf Traces betrachtet. Da wir hier aber unsere Betrachtungsweise von Transitionssystemen aus starten, werden wir auch Hiding aus der Sicht dieser Systeme definieren, wie in [Sch12]. Eine ähnliche Betrachtung für Hiding bei EIOs wurde auch bereits in [Lyn96] umgesetzt. Dort werden nur Output Aktionen internalisiert, jedoch gibt es eine Menge an internen Aktionen und nicht nur eine. Das Hiding wir durch einen Internalisierungsoperator umgesetzt. Es sollen dadurch Aktionen versteckt werden können, d.h. durch τs ersetzt werden. In [CJK13] ist es in der Definition des Hidings möglich Outputs und Inputs zu verstecken. Durch das verstecken von Outputs sind diese nur nicht mehr nach Außen sichtbar. Werden jedoch Inputs versteckt, sind alle Traces, die diesen Input benötigen nicht mehr ausführbar. Sie sind dann ab dem versteckten Input nicht mehr weiterführbar. Somit werden wir in unserer Definition nur die Internalisierung von Outputs erlauben und uns deshalb an [Sch12] als Quelle halten.

Definition 2.6 (*Internalisierungsoperator*). *Für ein EIO $S = (Q, I, O, \delta, q_0.E)$ ist $S/\{x_1, x_2, \ldots, x_n\}$, mit dem* Internalisierungsoperator $\cdot/\cdot$, *definiert als $S' = (Q, I, O', \delta', q_0, E)$ mit:*

- $O' = O \backslash \{x_1, x_2, \ldots, x_n\}$
- $\delta' = (\delta \cup \{(q, \tau, q') \mid (q, x, q') \in \delta, x \in (\{x_1, x_2, \ldots, x_n\} \cap O)\}) \backslash \{(q, x, q') \mid x \in (\{x_1, x_2, \ldots, x_n\} \cap O)\}$

Die Menge hinter dem Internalisierungsoperator ist nicht auf Outputs beschränkt, so sind dort ebenfalls Inputs zulässig. Jedoch hat ein Input in dieser Menge keine Auswirkungen auf den das System auf das es angewendet wird. Da wir jedoch uns im weiteren Verlauf Gedanken um die Auswirkungen dieses Operators auf unsere Verfeinerungsrelationen machen möchten, werden wir auch Systeme betrachten müssen, die vor der Anwendung der Parallelkomposition einen Input in der Menge hinter dem Operator stehen haben.

3 Verfeinerung über Errortraces

3.1 Präkongruenz für Error

Da es in dieser Arbeit vor allem um die Erreichbarkeit und die Kommunikation zwischen EIOs geht, wurden die nächsten beiden Definitionen explizit getrennt und erweitert zu denen in [BV14]. Ebenfalls wurde die Parallelkomposition geändert, wie in [Sch12].

Definition 3.1 (*error-freie Kommunikation*). *Ein Error ist* lokal erreichbar *in einem EIO S, wenn $\exists w \in O^* : q_0 \overset{w}{\Rightarrow} q \in E$.*
Zwei EIOs S_1 und S_2 kommunizieren error-frei*, wenn in ihrer Parallelkomposition $S_1 \| S_2$ keine lokalen Errors erreicht werden können.*

Mittels der lokalen Erreichbarkeit von Errors können wir eine Verfeinerungsrelation definieren. Zusätzlich definieren wir uns bereits die gröbste Präkongruenz, die wir suchen wollen. Somit können wir überprüfen, ob eine andere Präkongruenz, die wir finden auch die Eigenschaft erfüllt die gröbste zu sein.

Definition 3.2 (*Error-Verfeinerungs-Basisrelation*). *Für EIOs S_1 und S_2 mit der gleichen Signatur schreiben wir $S_1 \sqsubseteq_E^B S_2$, wenn ein Error in S_1 nur dann lokal erreichbar ist, wenn er auch in S_2 lokal erreichbar ist. Es handelt sich dabei um eine* Basisrelation *für die* Verfeinerung *im Bezug auf* Errors*.*
$\sqsubseteq_E^C$ bezeichnet die vollständig abstrakte Präkongruenz *von $\sqsubseteq_E^B$ bezüglich $\cdot\|\cdot$, d.h. die gröbste Präkongruenz bezüglich $\cdot\|\cdot$ die in $\sqsubseteq_E^B$ enthalten ist.*

Um uns nun näher mit den Präkongruenzen auseinandersetzen zu können, müssen wir bestimmte Traces aus unser Struktur hervorheben. Die strikten Errortraces entsprechen Wege, die direkt vom Startzustand zu einem Zustand in der Menge E führen. Da Outputs Aktionen sind, die von außen nicht verhindert werden können, benötigen wir auch noch die Menge der Traces, die zu einem Zustand führen, von dem aus mit lokalen Aktionen ein Error erreicht werden kann. Zusätzlich ist auch noch die Menge der Traces interessant, für die es einen Input $a \in I$ gibt, durch den sie möglicherweise nicht fortgesetzt werden können. Diese führen zwar nicht direkt zu einem Error, jedoch in Komposition mit einem anderen Transitionssystem sind dies gefährdete Stellen. Sie führen zu einem neuen Error, sobald dieser Input für die Synchronisation fehlt.

Definition 3.3 (*Errortraces*). *Für ein EIO S definieren wir:*

- strikte Errortraces*: $StET(S) = \{w \in \Sigma^* \mid q_0 \overset{w}{\Rightarrow} q \in E\}$,*
- gekürzte Errortraces*: $PrET(S) = \{prune(w) \mid w \in StET(S)\}$,*
- fehlende Input-Traces*: $MIT(S) = \{wa \in \Sigma^* \mid q_0 \overset{w}{\Rightarrow} q \wedge a \in I \wedge q \overset{a}{\not\rightarrow}\}$.*

In der folgenden Definition halten wir fest, was wir als Errortraces auffassen. Diese Menge ist dadurch, dass sie die fortgesetzten Traces aus $PrET$ enthält deutlich allgemeiner wie die Menge $StET$. Zusätzlich definieren wir auch noch die geflutete Sprache, in der wir die Informationen aus der Sprache und den Errortraces vereinen und somit bei der Inklusion dann nicht mehr explizit unterscheiden.

Definition 3.4 (*Error-Semantik*). *Sei S ein EIO.*

- *Die Menge der* Errortraces *von S ist $ET(S) := cont(PrET(S)) \cup cont(MIT(S))$.*
- *Die* error-geflutete Sprache *von S ist $EL(S) := L(S) \cup ET(S)$.*

Für zwei EIOs S_1, S_2 mit der gleichen Signatur schreiben wir $S_1 \sqsubseteq_E S_2$, wenn $ET(S_1) \subseteq ET(S_2)$ und $EL(S_1) \subseteq EL(S_2)$ gilt.

Der folgende Satz wurde in [BV14] nur für die Parallelkomposition mit verborgenen synchronisierten Aktionen formuliert, jedoch entspricht er dem analogen Satz aus [Sch12]. Da der Beweis jedoch ohne Beachtung von [Sch12] neu geführt wurde, werden hier eher auf die Erwähnung der Unterschiede zu [BV14] wert legen.

Satz 3.5 (*Error-Semanik für Parallelkompositionen*). *Für zwei komponierbare EIOs S_1, S_2 und $S_{12} = S_1 \| S_2$, gilt:*

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

Beweis.

1. „$\subseteq$“:

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

- Fall 1 ($w \in MIT_{12}$): Aus der Definition von MIT folgt, dass es eine Aufteilung $w = xa$ gibt mit $(q_{01}, q_{02}) \overset{x}{\Rightarrow} (q_1, q_2) \wedge a \in I_{12} \wedge (q_1, q_2) \overset{a}{\not\rightarrow}$. Da $I_{12} \overset{2.2}{=} (I_1 \backslash O_2) \cup (I_2 \backslash O_1) = (I_1 \cup I_2) \backslash (O_1 \cup O_2)$ ist $a \in (I_1 \cup I_2)$ und $a \notin (O_1 \cup O_2)$. Wir unterscheiden, ob $a \in (I_1 \cap I_2)$ oder $a \in (I_1 \cup I_2) \backslash (I_1 \cap I_2)$ ist. Diese Unterscheidung ist in [BV14] nicht nötig, da dort $I_1 \cap I_2 = \emptyset$ gilt, somit gibt es dort nur den Fall 1b).
  - Fall 1a) ($a \in (I_1 \cap I_2)$): Nun können wir den Ablauf der Komposition auf die Transitionssysteme projizieren und erhalten dann oBdA $q_{01} \overset{x_1}{\Rightarrow} q_1 \overset{a}{\not\rightarrow}$ und $q_{02} \overset{x_2}{\Rightarrow} q_2 \overset{a}{\not\rightarrow}$ oder $q_{02} \overset{x_2}{\Rightarrow} q_2 \overset{a}{\rightarrow}$ mit $x \in x_1 \| x_2$. Daraus können wir $x_1 a \in cont(MIT_1) \subseteq ET_1 \subseteq EL_1$ und $x_2 a \in EL_2$ ($x_2 a \in MIT_2$ oder $x_2 a \in L_2$) folgern. Damit folgt $w \in (x_1 \| x_2) \cdot \{a\} \subseteq (x_1 a) \| (x_2 a) \subseteq ET_1 \| EL_2$, und somit ist w in der rechten Seite der Gleichung enthalten.

  - Fall 1b) ($a \in (I_1 \cup I_2) \backslash (I_1 \cap I_2)$): OBdA gilt $a \in I_1$. Durch Projektion erhalten wir: $q_{01} \stackrel{x_1}{\Rightarrow} q_1 \stackrel{a}{\not\rightarrow}$ und $q_{02} \stackrel{x_2}{\Rightarrow} q_2$ mit $x \in x_1 \| x_2$. Daraus folgt $x_1 a \in cont(MIT_1) \subseteq ET_1$ und $x_2 \in L_2 \subseteq EL_2$. Somit gilt $w \in (x_1 \| x_2) \cdot \{a\} \subseteq (x_1 a) \| x_2 \subseteq ET_1 \| EL_2$. Dies ist eine Teilmenge der rechten Seite der Gleichung.

- Fall 2 ($w \in PrET_{12}$): Durch die Definitionen von $PrET$ und $prune$ wissen wir, dass es ein $v \in O_{12}^*$ gibt, so dass $(q_{01}, q_{02}) \stackrel{w}{\Rightarrow} (q_1, q_2) \stackrel{v}{\Rightarrow} (q_1', q_2')$ gilt mit $(q_1', q_2') \in E_{12}$ und $w = prune(wv)$. Durch Projektion erhalten wir $q_{01} \stackrel{w_1}{\Rightarrow} q_1 \stackrel{v_1}{\Rightarrow} q_1'$ und $q_{02} \stackrel{w_2}{\Rightarrow} q_2 \stackrel{v_2}{\Rightarrow} q_2'$ mit $w \in w_1 \| w_2$ und $v \in v_1 \| v_2$. Aus $(q_1', q_2') \in E_{12}$ folgt, dass es sich entweder um einen geerbten oder einen neuen Error handelt. Bei einem geerbten wäre bereits einer der beiden Zustände ein Error-Zustand gewesen. Der neue Error hingegen wäre durch die fehlende Möglichkeit entstanden eine synchronisierte Aktion auszuführen.
  - Fall 2a) (geerbter Error): OBdA $q_1' \in E_1$. Daraus folgt $w_1 v_1 \in StET_1 \subseteq cont(PrET_1) \subseteq ET_1$. Da gilt $q_{02} \stackrel{w_2 v_2}{\Rightarrow}$, erhalten wir $w_2 v_2 \in L_2 \subseteq EL_2$. Dadurch ergibt sich $wv \in ET_1 \| EL_2$ mit $w = prune(wv)$ und somit ist w in der rechten Seite der Gleichung enthalten.
  - Fall 2b) (neuer Error): OBdA $a \in I_1 \cap O_2$ mit $q_1' \stackrel{a}{\not\rightarrow} \wedge q_2' \stackrel{a}{\rightarrow}$. 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 $a \in O_2 \subseteq O_{12}$ gilt $w = prune(wva)$ und somit ist w in der rechten Seite der Gleichung enthalten.

1. „$\supseteq$“:

Wegen der Abgeschlossenheit beider Seiten der Gleichung gegenüber *cont* betrachten wir auch in diesem Fall nur ein präfix-minimales Element $x \in prune((ET_1 \| EL_2) \cup (EL_1 \| ET_2))$. 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 gehen wir davon aus, dass $xy \in ET_1 \| EL_2$ gilt, d.h. es gibt $w_1 \in ET_1$ und $w_2 \in EL_2$ mit $xy \in w_1 \| w_2$. In dem Punkt, dass wir das präfix-minimale Element noch mit Outputs fortsetzten können, unterscheidet sich dieser Beweis von dem in [Sch12]. Dort wird nicht weiter darauf eingegangen, dass die *prune*-Funktion hier noch zur Anwendung kommt, da wir jedoch später nur Präfixe von x betrachten werden, ist dieser Unterschied irrelevant.
Im Folgenden werden wir für alle Fälle von xy zeigen, dass es ein $v \in PrET(S_1 \| S_2) \cup MIT(S_1 \| S_2)$ gibt, das ein Präfix von xy ist und v entweder auf einen Input aus I_{12} endet oder $v = \varepsilon$. Da v entweder leer ist oder auf einen Input endet, muss v ein Präfix von x sein. ε ist Präfix von jedem Wort und sobald v mindestens einen Buchstaben enthält, muss das Ende von v vor dem Anfang von $y \in O_{12}^*$ liegen. Dadurch hat x ein Präfix in $PrET(S_1 \| S_2) \cup MIT(S_1 \| S_2)$, damit ist x in der Fortsetzung dieser Menge enthalten und somit gilt $x \in ET_{12}$.
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. Wir gehen nun davon aus, dass entweder $v_2 = w_2 \in L_2$ gilt

oder die letzte Aktion von v_1 findet vor oder gleichzeitig mit der letzten Aktion von v_2 statt. Ansonsten endet $v_2 \in PrET_2 \cup MIT_2$ vor v_1 und somit ist dieser Fall analog zu v_1 endet vor v_2.

- Fall 1 ($v_1 = \varepsilon$): Dadurch dass $\varepsilon \in PrET_1 \cup MIT_1$, ist bereits in S_1 ein Error lokal erreichbar. $\varepsilon \in MIT_1$ ist nicht möglich, da jedes Element aus MIT nach Definition mindestens die Länge 1 haben muss. Wir wähle $v_2' = v' = \varepsilon$, somit 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 $v_2 \in MIT_2$ können wir durch die gleiche Argumentation ebenfalls schließen, dass v_2' ein Präfix von v_2 ist. Wir wissen zusätzlich, dass v_2 auf $b \in I_2$ endet, jedoch muss nicht mehr wie in [BV14] $b \neq a$ gelten. Wir können also keine Aussage mehr darüber treffen, ob es sich um ein echtes Präfix handelt.

In allen Fällen erhalten wir $v_2' = v'|_{\Sigma_2}$ ist ein Präfix von v_2 und $v' \in v_1 \| v_2'$ ist ein Präfix von xy. Da nicht mehr $b \neq a$ gelten muss, können wir nicht mehr für alle Fälle $q_{02} \overset{v_2'}{\Rightarrow}$ folgern, wie das in [BV14] möglich war.

- Fall I ($v_1 \in MIT_1$ und $v_1 \neq \varepsilon$): Es gibt $q_{01} \overset{v_1'}{\Rightarrow} q_1 \overset{a}{\not\to}$ und sei $v' = v''a$. Bei der folgenden Fallunterscheidung müssen wir bezüglich [BV14] zwei weitere Fälle (Ib) und Ic)) einfügen, da es zulässig ist, das a sowohl in I_1 wie auch in I_2 enthalten ist.
  - Fall Ia) ($a \notin \Sigma_2$): Es gilt $q_{02} \overset{v_2'}{\Rightarrow} q_2$ mit $v'' \in v_1' \| v_2'$, da v_2' und v_2 nicht auf a enden können. Dadurch erhalten wir $(q_{01}, q_{02}) \overset{v''}{\Rightarrow} (q_1, q_2) \overset{a}{\not\to}$ mit $a \in I_{12}$. Somit können wir wählen $v := v''a = v' \in MIT_{12}$.
  - Fall Ib) ($a \in I_2$ und $v_2' \in MIT_2$): Es gilt $v_2' = v_2''a$ mit $q_{02} \overset{v_2''}{\Rightarrow} q_2 \overset{a}{\not\to}$ und $v'' \in v_1' \| v_2''$. a ist für S_2, ebenso wie für S_1, ein fehlender Input. Wir können somit schließen, dass $(q_1, q_2) \overset{a}{\not\to}$ gilt. Wir wählen $v := v''a = v' \in MIT_{12}$.
  - Fall Ic) ($a \in I_2$ und $v_2' \in L_2$): Es gilt $q_{02} \overset{v_2''}{\Rightarrow} q_2 \overset{a}{\to}$ mit $v_2' = v_2''a$. Da jedoch a in $Synch(S_1, S_2)$ liegt, da die Menge der synchronisierten Aktionen bezüglich [BV14] erweitert wurde, somit reicht schon, dass $q_1 \overset{a}{\not\to}$ gilt, um folgendes schließen zu können $(q_1, q_2) \overset{a}{\not\to}$. Somit können wir hier $v := v''a = v' \in MIT_{12}$ wählen.
  - Fall Id) ($a \in O_2$): Es gilt $v_2' = v_2''a$ und $q_{02} \overset{v_2'}{\Rightarrow}$, da der Input b von v_2 hier nicht dem a entsprechen kann. Wir erhalten also $q_{02} \overset{v_2''}{\Rightarrow} q_2 \overset{a}{\to}$ mit $v'' \in v_1' \| v_2''$.

Daraus ergibt sich $(q_{01}, q_{02}) \overset{v''}{\Rightarrow} (q_1, q_2)$ mit $q_1 \overset{a}{\not\rightarrow}, a \in I_1, q_2 \overset{a}{\rightarrow}, a \in O_2$, somit gilt $(q_1, q_2) \in E_{12}$. Wir wählen $v := prune(v'') \in PrET_{12}$.

- Fall II ($v_1 \in PrET_1$): $\exists u_1 \in O_1^* : q_{01} \overset{v_1}{\Rightarrow} q_1 \overset{u_1}{\Rightarrow} q_1'$ mit $q_1' \in E_1$. Da wir hier keine disjunkten Inputmengen, wie in [BV14], haben, 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 und somit wäre $v_2 = v_2'$. Dieser Fall verläuft jedoch analog zu Fall Ic) und wird somit hier nicht weiter betrachtet. Es gilt somit für alle anderen Fälle hier $q_{02} \overset{v_2'}{\Rightarrow} q_2$ mit $(q_{01}, q_{02}) \overset{v'}{\Rightarrow} (q_1, q_2)$.
  - Fall IIa) ($u_2 \in (O_1 \cap I_2)^*, c \in (O_1 \cap I_2)$, sodass $u_2 c$ Präfix von $u_1|_{I_2}$ mit $q_2 \overset{u_2}{\Rightarrow} q_2' \overset{c}{\not\rightarrow}$): Für das Präfix $u_1'c$ von u_1 mit $u_1'c|_{I_2} = u_2 c$ wissen wir, dass $q_1 \overset{u_1'}{\Rightarrow} q_1'' \overset{c}{\rightarrow}$. Somit gilt $u_1' \in u_1' \| u_2$ und $(q_1, q_2) \overset{u_1'}{\Rightarrow} (q_1'', q_2') \in E_{12}$, da für S_2 der entsprechende Input fehlt, der mit dem c Output von S_1 zu koppeln wäre. Es handelt sich also um einen neuen Error. Wir wählen $v := prune(v'u_1') \in PrET(S_1 \| S_2)$, dies ist ein Präfix von v', da $u_1 \in O_1^*$.
  - Fall IIb) ($q_2 \overset{u_2}{\Rightarrow} q_2'$ mit $u_2 = u_1|_{I_2}$): Somit ist $u_1 \in u_1 \| u_2$ und $(q_1, q_2) \overset{u_1}{\Rightarrow} (q_1', q_2') \in E_{12}$, da $q_1' \in E_1$ und somit handelt es sich um einen geerbten Error. Wir wählen nun $v := prune(v'u_1) \in PrET(S_1 \| S_2)$, das wiederum ein Präfix von v' ist.

2.:
Der Beweis für diesen Punkt musste bezüglich [BV14] nicht verändert werden, bis auf die Ersetzung der Zeichen der Parallelkomposition.
Es ist durch die Definition klar, dass gilt $L_i \subseteq EL_i$ und $ET_i \subseteq EL_i$. Wir beginnen mit der Argumentation von der rechten Seite der Gleichung aus:

$$
\begin{aligned}
&(EL_1 \| EL_2) \cup ET_{12} \\
&\overset{3.4}{=} (L_1 \cup ET_1) \| (L_2 \cup ET_2) \cup ET_{12} \\
&= \underbrace{(L_1 \| ET_2)}_{\substack{\subseteq (EL_1 \| ET_2) \\ \overset{1.}{\subseteq} ET_{12}}} \cup \underbrace{(ET_1 \| L_2)}_{\substack{\subseteq (ET_1 \| EL_2) \\ \overset{1.}{\subseteq} ET_{12}}} \cup (L_1 \| L_2) \cup \underbrace{(ET_1 \| ET_2)}_{\substack{\subseteq (EL_1 \| ET_2) \\ \overset{1.}{\subseteq} ET_{12}}} \cup ET_{12} \\
&= (L_1 \| L_2) \cup ET_{12} \\
&\overset{2.5}{=} L_{12} \cup ET_{12} \\
&\overset{3.4}{=} EL_{12}
\end{aligned}
$$

□

Die folgende Proposition wurde hier noch explizit mit Beweis eingefügt im Gegensatz zu den Ausführungen in [BV14], in denen diese Präkongruenz nur als Folgerung aus dem letzten Satz erwähnt wird. Die Feststellung, dass es sich um eine Präkongruenz handelt, ist wichtig, da wir dann die erste Eigenschaft erfüllt haben um eine Präkongruenz zu finden, die äquivalent ist zu der vollständig abstrakten Präkongruenz $\sqsubseteq_E^C$.

Proposition 3.6 (*Präkongruenz*). *$\sqsubseteq_E$ ist eine Präkongruenz bezüglich $\cdot\|\cdot$.*

Beweis. Es muss gezeigt werden: Wenn $S_1 \sqsubseteq_E S_2$ gilt, dann für jedes komponierbare S_3 auch $S_3\|S_1 \sqsubseteq_E S_3\|S_1$. D.h. es ist zu zeigen, dass aus $ET_1 \subseteq ET_2$ und $EL_1 \subseteq EL_2$ folgt, $ET(S_3\|S_1) \subseteq ET(S_3\|S_2)$ und $EL(S_3\|S_1) \subseteq EL(S_3\|S_2)$. Dies ergibt sich aus der Monotonie von *cont*, *prune* und $\cdot\|\cdot$ auf Sprachen wie folgt:

- $$\begin{aligned} ET(S_3\|S_1) &\overset{3.5\ 1.}{=} cont(prune((ET_3\|EL_1) \cup (EL_3\|ET_1))) \\ &\overset{\substack{ET_1 \subseteq ET_2 \\ \text{und} \\ EL_1 \subseteq EL_2}}{\subseteq} cont(prune((ET_3\|EL_2) \cup (EL_3\|ET_2))) \\ &\overset{3.5\ 1.}{=} ET(S_3\|S_2) \end{aligned}$$

- $$\begin{aligned} EL(S_3\|S_1) &\overset{3.5\ 2.}{=} (EL_3\|EL_1) \cup E_{31} \\ &\overset{\substack{EL_1 \subseteq EL_2 \\ \text{und} \\ ET_{31} \subseteq ET_{32}}}{\subseteq} (EL_3\|EL_2) \cup ET_{32} \\ &\overset{3.5\ 2.}{=} EL(S_3\|S_2) \end{aligned}$$

□

In [BV14] wurde auch die Verfeinerung von EIOs als Relation betrachtet mit Spezifikation und Implementierung. Hier soll ebenfalls eine Verfeinerungsrelation über EIOs betrachtet werden, jedoch sollen die synchronisierten Aktionen nicht verborgen werden. Dadurch ändern sich natürlich auch Teile des Beweises, vor allem muss statt mit $StET$ mit der Menge $PrET$ argumentiert werden. Dieses Lemma existiert in dieser Form nicht in [Sch12], da es dort mit der Aussage von Satz 3.8 kombiniert wurde. Jedoch ist die Aussage diese Lemmas trotzdem Teil dessen, was gezeigt wird und somit finden sich die Teile dieses Beweises auch dort wieder.

Lemma 3.7 (*Verfeinerung mit Errors*). *Gegeben sind zwei EIOs S_1 und S_2 mit der gleichen Signatur. Wenn alle Partner EIOs U, die mit S_2 gut kommunizieren, auch mit S_1 gut kommunizieren, dann verfeinert S_1 das EIO S_2. Diese Verfeinerung entspricht der Relation $\sqsubseteq_E$ von oben: Wenn $U\|S_1 \sqsubseteq_E^B U\|S_2$ für alle Partner U, dann gilt $S_1 \sqsubseteq_E S_2$.*

Beweis. Da S_1 und S_2 die gleichen Signaturen haben, definieren wir: $I := I_1 = I_2$ und $O := O_1 = O_2$. Für jeden der Partner U gilt $I_U = O$ und $O_U = I$.
Um $S_1 \sqsubseteq_E S_2$ zu zeigen, wird nachgeprüft, dass gilt:

- $ET(S_1) \subseteq ET(S_2)$,
- $EL(S_1) \subseteq EL(S_2)$.

Wir wählen ein präfix-minimales Element $w \in ET(S_1)$ und zeigen, dass dieses w oder eines seiner Präfixe in $ET(S_2)$ enthalten ist. Dies ist möglich, da beide Mengen durch *cont* abgeschlossen sind.

- Fall 1 ($w = \varepsilon$): Es handelt sich um einen lokal erreichbaren Error in S_1. Wir nehmen für U ein Transitionssystem, das nur aus dem Startzustand und einer Schleife für alle Inputs $x \in I_U$ besteht. Somit kann S_1 die gleichen Error-Zustände lokal erreichen wie $U \| S_1$. Daraus folgt, dass auch $U \| S_2$ einen lokal erreichbaren Error-Zustand haben muss. Durch unsere Definition von U kann dieser Error nur von S_2 geerbt sein. Es muss also in S_2 ein Error-Zustand durch interne Aktionen und Outputs erreichbar sein, d.h. es gilt $\varepsilon \in PrET(S_2)$.

- Fall 2 ($w = x_1 \dots x_n x_{n+1} \in \Sigma^+$ mit $n \geq 0$ und $x_{n+1} \in I$): Wir betrachten den folgenden Partner U (siehe auch Abbildung 3.1):
  - $Q_U = \{q_0, q_1, \dots, q_{n+1}\}$
  - $q_{0U} = q_0$
  - $E_U = \emptyset$
  - $\delta_U = \{(q_i, x_{i+1}, q_{i+1}) \mid 0 \leq i \leq n\}$
    $\cup \{(q_i, x, q_{n+1}) \mid x \in I_U \backslash \{x_{i+1}\}, 0 \leq i \leq n\}$
    $\cup \{(q_{n+1}, x, q_{n+1}) \mid x \in I_U\}$

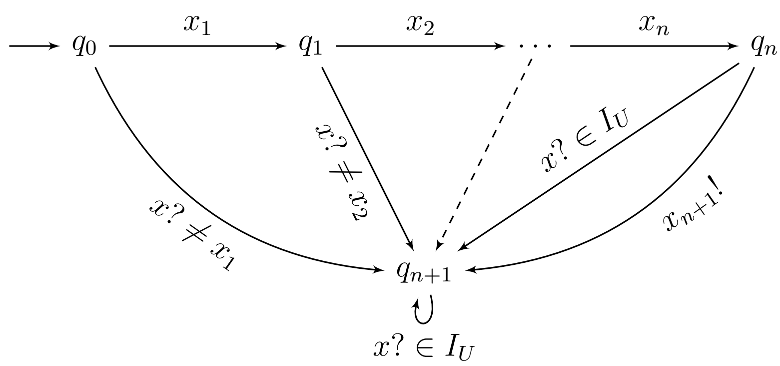

Abbildung 3.1: $x? \neq x_i$ steht für alle $x \in I_U \backslash \{x_i\}$

Wir können für w zwei Fälle unterscheiden. Beide führen zu $\varepsilon \in PrET(U \| S_1)$. Dieses Resultat unterscheidet sich von dem in [BV14], da hier die synchronisierten Aktionen als Outputs vorhanden bleiben und somit kann nicht $\varepsilon \in StET(U \| S_1)$ gelten.

- Fall 2a) ($w \in MIT(S_1)$): In $U \| S_1$ erhalten wir $(q_0, q_{01}) \overset{x_1 \dots x_n}{\Longrightarrow} (q_n, q')$ mit $q' \overset{x_{n+1}}{\not\rightarrow}$ und $q_n \overset{x_{n+1}}{\rightarrow}$. Deshalb gilt $(q_n, q') \in E_{U \| S_1}$ und $x_1 \dots x_n \in StET(U \| S_1)$. Da alle Aktionen aus w bis auf x_{n+1} synchronisiert werden gilt $x_1, \dots, x_n \in O_{U \| S_1}$. Daraus ergibt sich dann $\varepsilon \in PrET(U \| S_1)$.
- Fall 2b) ($w \in PrET(S_1)$): In $U \| S_1$ erhalten wir $(q_0, q_{01}) \overset{w}{\Rightarrow} (q_{n+1}, q'') \overset{u}{\Rightarrow} (q_{n+1}, q')$ für $u \in O^*$ und $q' \in E_1$. Daraus folgt $(q_{n+1}, q') \in E_{U \| S_1}$ und somit $wu \in StET(U \| S_1)$. Da alle Aktionen aus w synchronisiert werden gilt

$x_1, \dots, x_n, x_{n+1} \in O_{U\|S_1}$ und da $u \in O^*$ folgt $u \in O^*_{U\|S_1}$. Somit ergibt sich $\varepsilon \in PrET(U\|S_1)$.

Da wir wissen, dass $\varepsilon \in PrET(U\|S_1)$ gilt, können wir durch $U\|S_1 \sqsubseteq^B_E U\|S_2$ schließen, dass auch in $U\|S_2$ ein Error lokal erreichbar sein muss.
Dieser Error kann geerbt oder neu sein.

- Fall 2i) (neuer Error): Da jeder Zustand von U alle Inputs $x \in O = I_U$ zulässt, muss ein lokal erreichbarer Error einer sein, bei dem ein Output $a \in O_U$ von U möglich ist, der nicht mit einem passenden Input aus S_2 synchronisiert werden kann. Durch die Konstruktion von U sind in q_{n+1} keine Outputs möglich. Ein neuer Error muss also die Form (q_i, q') haben mit $i \leq n, q' \overset{x_{i+1}}{\not\rightarrow}$ und $x_{i+1} \in O_U = I$. Durch Projektion erhalten wir dann $q_{02} \overset{x_1 \dots x_i}{\Rightarrow} q' \overset{x_{i+1}}{\not\rightarrow}$ und damit gilt $x_1 \dots x_{i+1} \in MIT(S_2) \subseteq ET(S_2)$. Somit ist ein Präfix von w in $ET(S_2)$ enthalten.
- Fall 2ii) (geerbter Error): U hat $x_1 \dots x_i u$ ausgeführt mit $u \in I_U^* = O^*$ und ebenso hat S_2 diesen Weg ausgeführt. Durch dies hat S_2 einen Zustand in E_2 erreicht, da von U keine Error geerbt werden können. Es gilt dann $prune(x_1 \dots x_i u) = prune(x_1 \dots x_i) \in PrET(S_2) \subseteq ET(S_2)$. Da $x_1 \dots x_i$ ein Präfix von w ist, führt auch in diesem Fall ein Präfix von w zu einem Error.

Um uns von der zweiten Inklusion zu überzeugen reicht es aufgrund der ersten Inklusion und der Definition von EL, zu zeigen, dass $L(S_1) \backslash ET(S_1) \subseteq EL(S_2)$ gilt.
Wir nehmen uns dafür ein beliebiges $w \in L(S_1) \backslash ET(S_1)$ und zeigen, dass es in $EL(S_2)$ enthalten ist.

- Fall 1 ($w = \varepsilon$): Da ε immer in $EL(S_2)$ enthalten ist, haben wir hier nichts zu zeigen.

- Fall 2 ($w = x_1 \dots x_n$ mit $n \geq 1$): Wir konstruieren einen Partner U wie folgt (siehe dazu auch Abbildung 3.2):
  - $Q_U = \{q, q_0, q_1, \dots, q_n\}$
  - $q_{0U} = q_0$
  - $E_U = q_n$
  - $$\delta_U = \{(q_i, x_{i+1}, q_{i+1}) \mid 0 \leq i < n\} \cup \{(q_i, x, q) \mid x \in I_U \backslash \{x_{i+1}\}, 0 \leq i \leq n\} \cup \{(q, x, q) \mid x \in I_U\}$$

  Da $q_{01} \overset{w}{\Rightarrow} q'$ gilt, wissen wir, dass $U\|S_1$ einen lokal erreichbaren Error hat. Somit muss $U\|S_2$ ebenfalls einen lokal erreichbaren Error haben.

  - Fall 2a) (neuer Error aufgrund von $x_i \in O_U$ und $q_{02} \overset{x_1 \dots x_{i-1}}{\Rightarrow} q'' \overset{x_i}{\not\rightarrow}$): Es gilt $x_1 \dots x_i \in MIT(S_2)$ und somit $w \in EL(S_2)$. Anzumerken ist, dass nur auf diesem Weg Outputs von U möglich sind, deshalb gibt es keine anderen Outputs von U, die zu einem neuen Error führen können.

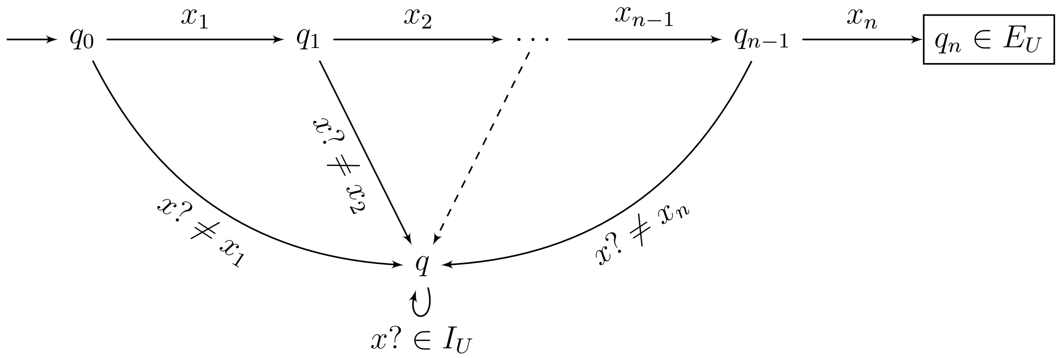

Abbildung 3.2: $x? \neq x_i$ steht für alle $x \in I_U \backslash \{x_i\}$, q_n ist der einzige Error-Zustand

- Fall 2b) (neuer Error aufgrund von $a \in O_2$): Der einzige Zustand, in dem U nicht alle Inputs erlaubt sind, ist q_n, der bereits ein Error-Zustand ist. Falls dieser Zustand erreichbar ist in $U \| S_2$, dann besitzt der komponierte EIO einen geerbten Error und es gilt $w \in L(S_2) \subseteq EL(S_2)$.
- Fall 2c) (geerbter Error von U): Da der einzige Zustand aus E_U q_n ist und alle Aktionen synchronisiert sind, ist dies nur möglich, wenn gilt $q_{02} \stackrel{x_1 \dots x_n}{\Longrightarrow}$. In diesem Fall gilt, wie im letzten, $w \in L(S_2) \subseteq EL(S_2)$.
- Fall 2d) (geerbter Error von S_2): Es gilt dann $q_{02} \stackrel{x_1 \dots x_i u}{\Longrightarrow} q' \in E_2$ für $i \geq 0$ und $u \in O^*$. Somit ist $x_1 \dots x_i u \in StET(S_2)$ und damit $prune(x_1 \dots x_i u) = prune(x_1 \dots x_i) \in PrET(S_2) \subseteq EL(S_2)$. Somit gilt $w \in EL(S_2)$.

□

Der folgende Satz sagt aus, dass $\sqsubseteq_E$ die gröbste Präkongruenz ist, die wir gesucht haben und somit mit äquivalent ist zur vollständig abstrakten Präkongruenz $\sqsubseteq_E^C$.

Satz 3.8 (*Full Abstractness für Error-Semanik*). *Seien S_1 und S_2 zwei EIOs mit derselben Signatur. Dann gilt $S_1 \sqsubseteq_E^C S_2 \Leftrightarrow S_1 \sqsubseteq_E S_2$, insbesondere ist $\sqsubseteq_E$ eine Präkongruenz.*

Beweis. Wie bereits in Proposition 3.6 festgehalten, ist $\sqsubseteq_E$ eine Präkongruenz.

„$\Leftarrow$“: Nach Definition gilt, wenn $\varepsilon \in ET(S)$, ist ein Error lokal erreichbar in S. Somit impliziert $S_1 \sqsubseteq_E S_2$, dass $\varepsilon \in ET(S_2)$ gilt, wenn $\varepsilon \in ET(S_1)$. Dadurch folgt ebenfalls, dass $S_1 \sqsubseteq_E^B S_2$ gilt. Somit folgt aus $S_1 \sqsubseteq_E S_2$ der relationale Zusammenhang $S_1 \sqsubseteq_E^C S_2$.

„$\Rightarrow$“: Durch die Definition von $\sqsubseteq_E^C$ folgt aus $S_1 \sqsubseteq_E^C S_2$, dass $U \| S_1 \sqsubseteq_E^C U \| S_2$ für alle EIOs U, die mit S_1 komponierbar sind. Somit folgt auch die Gültigkeit von $U \| S_1 \sqsubseteq_E^B U \| S_2$ für alle diese EIOs U. Mit Lemma 3.7 folgt dann $S_1 \sqsubseteq_E S_2$. □

Wir haben somit jetzt eine Kette an Folgerungen gezeigt, die sich zu einem Ring schließen. Dies ist in Abbildung 3.3 dargestellt.

Aus Satz 3.8 und Lemma 3.7 erhalten wir das folgende Korollar.

$$\begin{tikzcd}
S_1 \sqsubseteq_E S_2 \arrow[rr, Rightarrow, "\text{„}\Leftarrow\text{“ von Satz 3.8}"] & & S_1 \sqsubseteq_E^C S_2 \arrow[d, Rightarrow, "\text{„}\Rightarrow\text{“ von Satz 3.8}"] \\
\begin{array}{c}\forall \text{ Partner } U: \\ U\|S_1 \sqsubseteq_E^B U\|S_2\end{array} \arrow[u, Rightarrow, "\text{Lemma 3.7}"] & & \begin{array}{c}\forall \text{ komponierbaren } U: \\ U\|S_1 \sqsubseteq_E^B U\|S_2\end{array} \arrow[ll, Rightarrow, "\begin{array}{c}U\text{ Partner}\\ \Downarrow \\ U\text{ komponierbar}\end{array}"']
\end{tikzcd}$$

Abbildung 3.3: Folgerungskette

Korollar 3.9. *Ein EIO S_1 verfeinert einen EIO S_2 genau dann, wenn für alle EIOs U für die S_2 gut mit U kommuniziert folgt S_1 kommuniziert ebenfalls gut mit U.*
Dies lässt sich formal wie folgt ausdrücken: $S_1 \sqsubseteq_E S_2 \Leftrightarrow U\|S_1 \sqsubseteq_E^B U\|S_2$ für alle Partner U.

3.2 Hiding für Error

Wir wollen nun untersuchen, was für Auswirkungen Hiding auf unsere Verfeinerungsrelationen hat. Es werden also Outputs der Systeme internalisiert.

Proposition 3.10 (*Error Basisrelation bzgl. Internalisierungsoperator*). *Wenn $S_1 \sqsubseteq_E^B S_2$ gilt, dann folgt daraus, dass auch $(S_1/\{x_1, x_2, \ldots, x_n\}) \sqsubseteq_E^B (S_2/\{x_1, x_2, \ldots, x_n\})$ gilt.*

Beweis. Da die Definition der lokalen Erreichbarkeit auf lokalen Aktionen beruht, die aus den Outputs und der internen Aktion besteht, ändert sich durch das verbergen von Outputs nichts an der Error-Erreichbarkeit. Somit ist jeder Error, der in S_i lokal Erreichbar ist über einen Traces, der einen Outputs aus $\{x_1, x_2, \ldots, x_n\}$ enthält auch in $S_i/\{x_1, x_2, \ldots, x_n\}$ erreichbar, jedoch enthält der Traces nicht mehr diesen Output. Alle Traces, die keinen Outputs aus der Menge hinter dem Internalisierungsoperator enthalten bleiben unverändert erhalten. Es ist auch nicht möglich, dass durch das Verbergen von Outputs neue Errors entstehen. Somit folgt die Behauptung. □

Satz 3.11 (*Präkongurenz bzgl. Interalisierungsoperator*). *Seinen S_1 und S_2 zwei EIOs für die $S_1 \sqsubseteq_E S_2$ gilt, somit gilt auch $(S_1/\{x_1, x_2, \ldots, x_n\}) \sqsubseteq_E (S_2/\{x_1, x_2, \ldots, x_n\})$.*

Beweis. Da $S_1 \sqsubseteq_E S_2$ gilt, wissen wir, dass $ET_1 \subseteq ET_2$ und $EL_1 \subseteq EL_2$ gilt. Da für Elemente aus $X := \{x_1, x_2, \ldots, x_n\}$ die nicht in O enthalten sind nichts passiert bei der Internalisierung, gehen wir davon aus, das $X \subseteq O$ gilt. Wir wählen einen Traces $w = a_1a_2 \ldots a_m \in ET_1$. Solang w nicht nur aus Outputs besteht, gibt es also einen Ablauf $q_0 \overset{w'}{\Rightarrow}$ mit $w' = a_1a_2 \ldots a_k$, so dass $a_k \in I$, $k \leq m$ und $w' \in MIT_1$ oder $w' \in PrET_1$ gilt. Nach der Internalisierung bleiben von dem Ablauf nur noch die Aktionen übrig, die

nicht gleichzeitig ein Output und ein Element aus X sind. Der Traces reduziert sich also auf $w'' = w|_{\Sigma\setminus(O\cap X)}$. Diese w'' hat einen zu w' analoges Präfix, da der Input a_k erhalten beleibt. Dieses Präfix von w'' ist dann in $MIT(S_1/X)$ oder in $PrET(S_1/X)$ enthalten und somit ist w'' in $ET(S_1/X)$ enthalten. Diese Argumentation funktioniert jedoch nicht für ws, die nur aus Outputs bestehen. Da Elemente, aus $cont(MIT_1)$ mindestens einen Input enthalten müssen, muss so ein w aus $cont(PrET_1)$ sein. Somit ergibt sich ohne die $cont$-Funktion das Element $\varepsilon \in PrET_1$. Da es in diesem Wort keine Elemente aus X gibt, werden in diesem Wort auch keine Aktionen verborgen und somit gilt $\varepsilon \in PrET(S_1/X)$. Nach Voraussetzung gilt $w \in ET_2$. Nach analoger Argumentation gilt dann auch $w'' \in ET(S_2/X)$.
Somit bleibt jetzt nur noch zu zeigen, dass $L(S_1/X)\setminus ET(S_1/X) \subseteq EL(S_2/X)$ gilt. Die Argumentation, wieso nur diese Inklusion zu zeigen ist, kann dem Beweis zu Lemma 3.7 entnommen werden. Da wir bereits $L_1\setminus ET_1 \subseteq EL_2$ wissen, können wir schließen, dass alle relevanten Traces bereits in EL_2 enthalten sind und die Modifikation durch den Internalisierungsoperator wie oben keine Auswirkungen auf die Teilmengenbeziehung hat. □

In [BV14] wurden die Parallelkomposition nur mit Verbergen der synchronisierten Aktionen betrachtet. Diese Parallelkomposition können wir nun mit dem Internalisierungsoperator durch Hiding der synchronisierten Aktionen nachbilden.

Definition 3.12 (*Parallelkomposition mit Internalisierung*). *Seinen S_1 und S_2 komponierbare EIOs, dann ist $S_1|S_2 = (S_1\|S_2)/Synch(S_1, S_2)$.*

Wir wissen aus 3.6, dass $\sqsubseteq_E$ eine Präkongruenz bezüglich $\cdot\|\cdot$ ist, und aus **??**, dass $\sqsubseteq_E$ auch eine Präkongruenz bezüglich $\cdot/\cdot$ ist. Da sich nach Definition 3.12 die Parallelkomposition mit Internalisierung nur aus diesen Operatoren zusammensetzt, erhalten wir das folgende Korollar.

Korollar 3.13 (*Präkongurenz bzgl. Präkongurenz mit Internalisierung*). *$\sqsubseteq_E$ ist eine Präkongruenz bezüglich $\cdot|\cdot$.*

4 Verfeinerung über Error- und Ruhetraces

4.1 Präkongruenz für Ruhe

In diesem Kapitel werden wir uns nicht nur um die Erreichbarkeit von Error-Zuständen kümmern, sondern auch um die Erreichbarkeit von Ruhe-Zuständen. Wir werden dabei ähnlich vorgehen wie im letzten Kapitel, jedoch halten wir uns als Quelle an [CJK13]. Darin werden ähnliche Konzepte beschrieben, jedoch aus Sicht der Traces.
Wir sehen die Zustände, die keine Outputs und keine Transitionsmöglichkeit für eine interne Aktion haben als ruhig an.

Definition 4.1 (***Ruhe***)**.** *Ein* Ruhe-Zustand *ist ein Zustand in einem EIO der keine Outputs zulässt und keine Transitions mit τ besitzt.*
Somit ist die Menge der Ruhe-Zustände in einem EIO wie folgt formal definiert: $Qui := \left\{ q \in Q \mid \forall \alpha \in (O \cup \{\tau\}) : q \overset{\alpha}{\not\to} \right\}$.

Für die Erreichbarkeit verwenden wir wie im letzten Kapitel wieder den optimistischen Ansatz der lokalen Erreichbarkeit für die Errors. Ruhe ist kein unabwendbaren Fehler, sondern durch einen Input reparierbar werden. Somit ist ein Ruhe-Zustand als nicht so schlimmer Fehler anzusehen wie ein Error und deshalb können wir unseren Erreichbarkeitsbegriff für Ruhe weiter einschränken, so dass die Ruhe-Zustände erst erreichbar sind, wenn sie durch eine interne Aktion zu erreichen sind. Also eher einen hyper-optimistischen Erreichbarkeitsbegriff.

Definition 4.2 (***error- und ruhefreie Kommunikation***)**.** *Zwei EIOs S_1 und S_2* kommunizieren gut*, wenn in ihrer Parallelkomposition $S_1 \| S_2$ keine Errors lokal und keine Ruhe-Zustände durch interne Aktionen erreichbar sind.*

Definition 4.3 (***Ruhe-Verfeinerungs-Basisrelation***)**.** *Für EIOs S_1 und S_2 mit der gleichen Signatur schreiben wir $S_1 \sqsubseteq_{Qui}^{B} S_2$, wenn ein Error oder Ruhe-Zustand in S_1 nur dann lokal bzw. durch ein τ erreichbar ist, wenn er auch in S_2 auf die gleiche Weise erreichbar ist. Diese* Basisrelation *stellt eine* Verfeinerung *bezüglich* Errors *und* Ruhe-Zustände *dar.*
$\sqsubseteq_{Qui}^{C}$ *bezeichnet die* vollständig abstrakte Präkongruenz *von $\sqsubseteq_{Qui}^{B}$ bezüglich $\cdot\|\cdot$.*

Um uns genauer mit den Präkongruenzen auseinandersetzten zu können, brauchen wir wie im letzten Kapitel die Definition von Traces auf unserer Struktur. Dadurch erhalten wir die Möglichkeit die gröbste Präkongruenz finden und definieren zu können.

Definition 4.4 (***Error- und Ruhetraces***)**.** *Sei S ein EIO und definiere:*

- *die Traces bezüglich Errors entsprechen denen aus Definition 3.3,*
- strickte Ruhetraces*:* $StQT(S) := \{w \in \Sigma^* \mid q_0 \overset{w}{\Rightarrow} q \in Qui\}$*.*

Da wir für Ruhe-Zustände keinen lokalen Erreichbarkeitsbegriff verwenden benötigen wir keine gekürzten Ruhetraces bei denen die *prune*-Funktion zur Anwendung käme.

Definition 4.5 (*Error- und Ruhe-Semantik*). *Sei S ein EIO.*

- *Die Menge der* Errortraces *ist wie in 3.4 definiert.*
- *Die Menge der* Ruhetraces *von S ist $QT(S) := StQT(S)$.*
- *Die* ruhe-geflutete Sprache *von S ist $QL(S) := L(S) \cup ET(S)$ und entspricht somit der error-gefluteten Sprache $EL(S)$ in 3.4.*

Für zwei EIOs S_1, S_2 mit der gleichen Signatur schreiben wir $S_1 \sqsubseteq_{Qui} S_2$, wenn $ET(S_1) \subseteq ET(S_2)$, $QT(S_1) \subseteq QT(S_2)$ und $QL(S_1) \subseteq QL(S_2)$ gilt.

Da QT der Menge $StQT$ entspricht, hätten wir die Definition nicht unbedingt benötigt, jedoch für eine einheitliche Notation der Kapitel ist die neue Bezeichnung sinnvoll. Ebenso bezeichnen wir hier für die bessere Unterscheidung in den Kapiteln die Sprache EL mit QL.
Da QT nur die strickten Ruhetraces enthält, also die Traces, die im Automaten vorhanden sind und zu einem Ruhe-Zustand führen, gilt $QT \subseteq L \subseteq QL$. Somit musste QT nicht mehr explizit in die Definition von QL aufgenommen werden und trotzdem ist jeder Traces, der in QT möglich ist auch in der gefluteten Sprache enthalten.

Satz 4.6 (*Error- und Ruhe-Semantik für Parallelkompositonen*). *Für zwei komponierbare EIOs S_1, S_2 und $S_{12} = S_1 \| S_2$ gilt:*

1. $ET_{12} = cont(prune((ET_1 \| QL_2) \cup (QL_1 \| ET_2)))$*,*
2. $QT_{12} = (QT_1 \| QT_2)$*,*
3. $QL_{12} = (QL_1 \| QL_2) \cup ET_{12}$*.*

Beweis.

1.:
Der Beweis diese Punktes entspricht dem Beweis von Punkt 1. im Beweis von Satz 3.5.
2. "$\subseteq$":
Wir betrachten $w \in StQT_{12}$ und versuchen dessen Zugehörigkeit zur rechten Menge zu zeigen. Aufgrund von Definition 4.4 wissen wir es gilt $(q_{01}, q_{02}) \overset{w}{\Rightarrow} (q_1, q_2)$ mit $(q_1, q_2) \in Qui_{12}$. Durch Projektion erhalten wir $q_{01} \overset{w_1}{\Rightarrow} q_1$ und $q_{02} \overset{w_2}{\Rightarrow} q_2$ mit $w \in w_1 \| w_2$. Aus $(q_1, q_2) \in Qui_{12}$ können wir folgern, dass bereits $q_1 \in Qui_1$ und $q_2 \in Qui_2$ gilt. Somit gilt $w_1 \in StQT_1 \subseteq QT_1$ und $w_2 \in StQT_2 \subseteq QT_2$. Daraus folgt dann $w \in QT_1 \| QT_2$ und somit ist w in der rechten Seiten der Gleichung enthalten.
2. "$\supseteq$":
Für diese Inklusionsrichtung betrachten wir ein Element $w \in QT_1 \| QT_2$ und zeigen, dass

es in der linken Menge enthalten ist. Da $QT_i = StQT_i$ gilt, existieren w_1 und w_2 für die gilt $q_{01} \overset{w_1}{\Rightarrow} q_1 \in Qui_1$ und $q_{02} \overset{w_2}{\Rightarrow} q_2 \in Qui_2$ mit $w \in w_1 \| w_2$. Da für die Zustände q_1 und q_2 die Zugehörigkeit zur Ruhe Menge gilt, können wir folgern, dass der aus ihnen zusammengesetzte Zustand in der Parallelkomposition ebenfalls keine Outputs zulässt. Somit gilt also für die Komposition $(q_{01}, q_{02}) \overset{w}{\Rightarrow} (q_1, q_2) \in Qui_{12}$ und dadurch ist w in der linken Seite der Gleichung enthalten.

3.:

Der Beweis diese Punktes entspricht dem Beweis von Punkt 2. im Beweis von Satz 3.5, da QL der error-gefluteten Sprache EL entspricht. □

QT_{12} ist hier in QL_{12} enthalten, da wie wir bereits oben festgestellt haben $QT_i \subseteq QL_i$ für $i = 1, 2$ gilt. Somit gilt auch $(QT_1 \| QT_2) \subseteq (QL_1 \| QL_2)$.
Die folgende Proposition ist eine direkte Folgerung aus dem letzten Satz. Jedoch ist es eine wichtige Feststellung für die weiteren Verlauf die gröbste Präkongruenz finden zu wollen.

Proposition 4.7 (*Präkongruenz*). *$\sqsubseteq_{Qui}$ ist eine Präkongruenz bezüglich $\cdot \| \cdot$.*

Beweis. Es muss gezeigt werden, wenn $S_1 \sqsubseteq_{Qui} S_2$ gilt, für jedes S_3 auch $S_3 \| S_1 \sqsubseteq_{Qui} S_3 \| S_2$ gilt. D.h. es ist zu zeigen, dass aus $ET_1 \subseteq ET_2$, $QT_1 \subseteq QT_2$ und $QL_1 \subseteq QL_2$ folgt, $ET_{31} \subseteq ET_{32}$, $QT_{31} \subseteq QT_{32}$ und $QL_{31} \subseteq QL_{32}$. Dies ergibt sich wie im Beweis zu Proposition 3.6 aus der Monotonie von *cont*, *prune* und $\cdot \| \cdot$ auf Sprachen wie folgt:

- $ET_{31} \overset{\text{Beweis 3.6 Punkt 1}}{\subseteq} ET_{32}$

- $$\begin{aligned} QT_{31} &\overset{4.6\ 2.}{=} (QT_3 \| QT_1) \\ &\overset{QT_1 \subseteq QT_2}{\subseteq} (QT_3 \| QT_2) \\ &\overset{4.6\ 2.}{=} QT_{32} \end{aligned}$$

- $QL_{31} \overset{\text{Beweis 3.6 Punkt 2}}{\subseteq} QL_{32}$

□

Lemma 4.8 (*Verfeinerung mit Ruhe-Zuständen*). *Gegeben sind zwei EIOs S_1 und S_2 mit der gleichen Signatur. Wenn alle Partner EIOs U, die mit S_2 gut kommunizieren, auch mit S_1 gut kommunizieren, dann verfeinert S_1 das EIO S_2. Diese Verfeinerung entspricht der Relation $\sqsubseteq_{Qui}$ von oben: Wenn $U \| S_1 \sqsubseteq^B_{Qui} U \| S_2$ für alle Partner U, dann gilt $S_1 \sqsubseteq_{Qui} S_2$.*

Beweis. Da wir davon ausgehen, dass S_1 und S_2 die gleiche Signatur haben, definieren wir $I := I_1 = I_2$ und $O := O_1 = O_2$. Für jeden Partner U gilt $I_U = O$ und $O_U = I$.
Um zu zeigen, dass die Relation $S_1 \sqsubseteq_{Qui} S_2$ gilt, müssen wir die folgenden Punkte nachweisen:

- $ET(S_1) \subseteq ET(S_2)$,

- $QT(S_1) \subseteq QT(S_2)$,
- $QL(S_1) \subseteq QL(S_2)$.

Der erste und der letzte Punkt wurden bereits in Lemma 3.7 gezeigt. So bleibt uns nur noch die Inklusion $QT_1 \subseteq QT_2$ zu zeigen.
Wir wählen ein $w \in QT(S_1)$ und zeigen, dass es auch in $QT(S_2)$ enthalten ist.

- Fall 1 ($w = \varepsilon$): Somit ist der Startzustand von S_1 ein Ruhe-Zustand. Sobald der Startzustand eines Partners U nicht ruhig ist, können wir nichts für S_2 schließen. Somit wählen wir für U ein Transitionssystem, dass nur aus dem Startzustand und einer Schliefe für alle Inputs $x \in I_U$ besteht. Somit ist der Startzustand von U auch ruhig. Daraus folgt, dass auch der Startzustand der Komposition $U \| S_1$ ein Ruhe-Zustand ist. Dadurch muss auch $U \| S_2$ als Startzustand einen Ruhe-Zustand haben. Da dieser aber nur entstehen kann, wenn für beide Teilsysteme bereits ε in den jeweiligen Ruhetraces liegt, somit gilt $\varepsilon \in QT(S_2)$.

- Fall 2 ($w = x_1 \ldots x_n \in \Sigma^+$ mit $n \geq 0$): Wir betrachten den Partner U, bei dem mit w ebenfalls ein Zustand erreicht wird, der ruhig ist. Somit ist in der Komposition die Komposition dieser beiden Zustände ebenfalls ein Ruhe-Zustand. Es folgt, dass ein Ruhe-Zustand auch in $U \| S_2$ durch w erreichbar sein muss und somit auch in S_2. Daraus folgt, dass gilt $w \in QT(S_2)$.

□

Satz 4.9 (*Full Abstractness für Ruhe-Semantik*). *Seien S_1 und S_2 zwei EIOs mit derselben Signatur. Dann gilt $S_1 \sqsubseteq^C_{Qui} S_2 \Leftrightarrow S_1 \sqsubseteq_{Qui} S_2$, insbesondere ist $\sqsubseteq_{Qui}$ eine Präkongruenz.*

Beweis. Wie bereits in Proposition 4.7 festgehalten, ist $\sqsubseteq_{Qui}$ eine Präkongruenz.
„$\Leftarrow$“: Nach Definition gilt, wenn $\varepsilon \in QT(S)$, ist ein Ruhe-Zustand durch ein τ erreichbar in S. Somit impliziert $S_1 \sqsubseteq_{Qui} S_2$, dass $\varepsilon \in QT_2$ gilt, wenn $\varepsilon \in QT_1$. Dadurch folgt ebenfalls, dass $S_1 \sqsubseteq^B_{Qui} S_2$ gilt. Somit folgt aus $S_1 \sqsubseteq_{Qui} S_2$ der relationale Zusammenhang $S_1 \sqsubseteq^C_{Qui} S_2$.
„$\Rightarrow$“: Durch die Definition von $\sqsubseteq^C_{Qui}$ folgt aus $S_1 \sqsubseteq^C_{Qui} S_2$, dass $U \| S_1 \sqsubseteq^C_{Qui} U \| S_2$ für alle EIOs U , die mit S_1 komponierbar sind. Somit folgt auch die Gültigkeit von $U \| S_1 \sqsubseteq^B_E U \| S_2$ für alle diese EIOs U. Mit Lemma 4.8 folgt dann $S_1 \sqsubseteq_{Qui} S_2$. □

Wir haben somit, wie im letzten Kapitel, eine Kette an Folgerungen gezeigt, die sich zu einem Ring schließen. Dies ist in Abbildung 4.1 dargestellt.
Aus Satz 4.9 und Lemma 4.8 erhalten wir das folgende Korollar.

Korollar 4.10. *Ein EIO S_1 verfeinert einen EIO S_2 genau dann, wenn für alle EIOs U für die S_2 gut mit U kommuniziert folgt S_1 kommuniziert ebenfalls gut mit U.*
Dies lässt sich formal wie folgt ausdrücken: $S_1 \sqsubseteq_{Qui} S_2 \Leftrightarrow U \| S_1 \sqsubseteq^B_{Qui} U \| S_2$ für alle Partner U.

$$
\begin{tikzcd}
S_1 \sqsubseteq_{Qui} S_2 \arrow[r, Rightarrow, "\text{„}\Leftarrow\text{“ von Satz 4.9}"] & S_1 \sqsubseteq^{C}_{Qui} S_2 \arrow[d, Rightarrow, "\text{„}\Rightarrow\text{“ von Satz 4.9}"] \\
\begin{array}{c} \forall \text{ Partner } U: \\ U\|S_1 \sqsubseteq^{B}_{Qui} U\|S_2 \end{array} \arrow[u, Rightarrow, "\text{Lemma 4.8}"] & \begin{array}{c} \forall \text{ komponierbaren } U: \\ U\|S_1 \sqsubseteq^{B}_{Qui} U\|S_2 \end{array} \arrow[l, Rightarrow, "\begin{array}{c} U \text{ Partner} \\ \Downarrow \\ U \text{ komponierbar} \end{array}"']
\end{tikzcd}
$$

Abbildung 4.1: Folgerungskette

4.2 Hiding für Ruhe

Wir wollen nun auch hier die Auswirkungen der Internalisierung von Aktionen auf unsere Verfeinerungsrelationen untersuchen.

Literaturverzeichnis

[BV14] Ferenc Bujtor und Walter Vogler, *Error-Pruning in Interface Automata*, preprint, Universität Augsburg, 2014.

[CJK13] Chris Chilton, Bengt Jonsson, und Marta Z. Kwiatkowska, *An Algebraic Theory of Interface Automata*, preprint, University of Oxford, 2013.

[Lyn96] Nancy A. Lynch, *Distributed Algorithms*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1996.

[Sch12] Christoph Franz Schlosser, *EIO-Automaten mit Parallelkomposition ohne Internalisierung*, Bachelorarbeit, Universität Augsburg, 2012.