CARL VON OSSIETZKY universität OLDENBURG
Fachbereich Informatik

Diplomarbeit

Fehlerbaumanalyse und Model-Checking

Vorgelegt von	:	Andreas Schäfer
Betreuer	:	Prof. Dr. Ernst-Rüdiger Olderog
Zweitprüfer	:	Dr. Henning Dierks

Oldenburg, den 5. Dezember 2001

Zusammenfassung

In den Ingenieurswissenschaften wird die Fehlerbaumanalyse als Methode eingesetzt, um die Sicherheit von technischen Systemen zu beurteilen. Sie beruht nicht auf formalen Modellen und Beweisen, sondern auf der Einschätzung der Analysten. Es gibt deshalb verschiedene Ansätze, eine formale Semantik für Fehlerbäume zu definieren, um die Sicherheit der Analyse und damit auch des technischen Systems zu erhöhen. In dieser Arbeit soll untersucht werden, wie eine Semantik, die dem Fehlerbaum Formeln aus dem Duration Calculus zuordnet, auf das operationale Modell der Phasenautomaten übertragen und mit einem Model-Checker für Phasenautomaten überprüft werden kann. Die Ergebnisse werden anhand von Fallbeispielen demonstriert.

Danksagungen

Bei allen Personen, die zur Entstehung dieser Arbeit beigetragen haben, möchte ich mich vorweg bedanken.
Zuerst danke ich Prof. Dr. Ernst-Rüdiger Olderog für die intensive Betreuung und die wichtigen Ratschläge und Hilfen.
Für jede Menge Tipps zu Realzeitsystemen, Moby/DC und Moby/PLC danke ich Dr. Henning Dierks und Dr. Josef Tapken.
Dank gebührt auch meinen Freunden für ihre Unterstützung und ihr Verständnis, darunter besonders auch Monika Willenborg dafür, dass sie die ganze Arbeit korrektur gelesen hat.
Ganz besonders danke ich auch meinem Vater Werner Karl Schäfer für seine Unterstützung während des gesamten Studiums.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

In vielen sicherheitskritischen Bereichen wie Verkehrsmitteln, Kernkraftwerken und medizintechnischen Anlagen können Fehlfunktionen der technischen Systeme zu hohen Geldverlusten, zu Verletzung und sogar zum Tod von Menschen führen. Aus diesem Grund muss bei der Entwicklung dieser Anlagen sehr viel Wert auf die Sicherheitsanalyse gelegt werden. Ein einfaches modellhaftes sicherheitskritisches System ist das Fallbeispiel der verteilten Weichen- und Signalsteuerung.

1.1 Fallbeispiel „Verteilte Weichen- und Signalsteuerung“

Das Fallbeispiel verteilte Weichen- und Signalsteuerung ist die Abschlussaufgabe des Fortgeschrittenenpraktikums Realzeitsysteme an der Universität Oldenburg aus dem Wintersemester 2000/2001 ([DPS+01]). Die Aufgabe bestand darin, auf dem in Abbildung 1.2 dargestellten Gleissystem eine sichere Weichen- und Signalsteuerung zu implementieren. Auf diesem Gleissystem fahren zwei Züge, die auf dem äußeren Kreis nur rechts herum und auf dem inneren Kreis nur links herum fahren dürfen. Nur der gemeinsame kritische Abschnitt – mit B1 bezeichnet – darf in beide Richtungen befahren werden. Wenn sich ein Zug im kritischen Abschnitt befindet, darf er die Fahrtrichtung ändern, sofern dadurch kein Zusammenstoß mit einem anderen Zug ausgelöst wird. Auf diesem Abschnitt wird “auf Sicht” gefahren, d.h. der Zug ist selbst dafür verantwortlich, dass es durch einen Wechsel der Fahrtrichtung nicht zu einem Zusammenstoß kommt. Es ist erlaubt, dass sich beide Züge in dem gleichen Abschnitt befinden, das Steuerungssystem muss sogar ausdrücklich erlauben, dass zwei Züge aus der gleichen Richtung hintereinander in den kritischen Bereich einfahren (“Folgefahrt”). Die Einfahrt in den kritischen Abschnitt wird über zwei Signale ($L1, L2$) geregelt, die durch eine verteilte Steuerung angesteuert werden. Sie besteht aus zwei kommunizierenden Controllern, die jeweils über drei Sensoren (mit S bezeichnet) vorbeifahrende Züge registrieren können, sowie die zugeordnete Weiche und das zugeordnete Signal steuern.

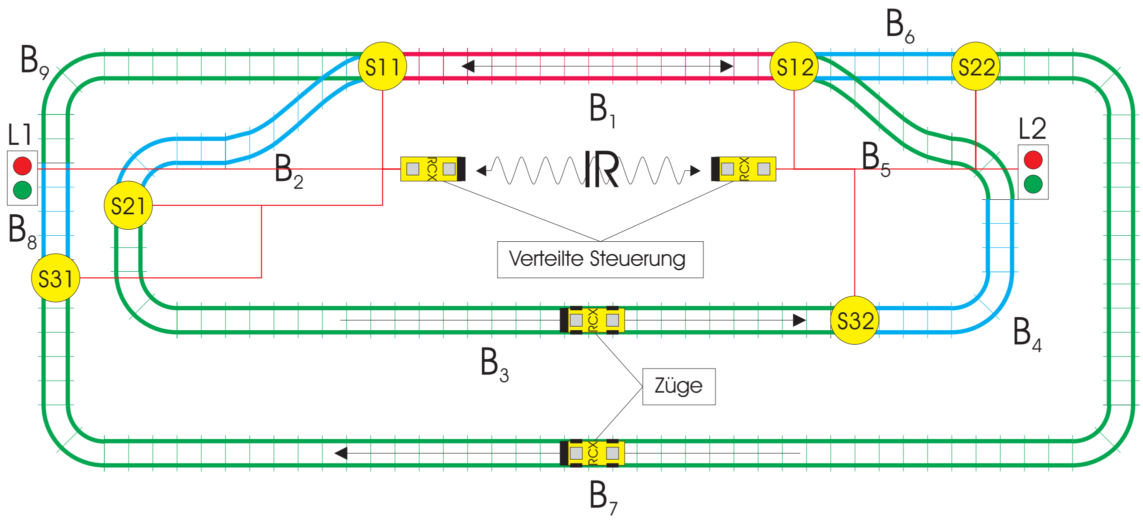


Abbildung 1.1: Gleissystem zum Fallbeispiel Zugsteuerung

Abbildung 1.2: Realisierung mit Lego-Mindstorms

Die einzelnen Sensoren haben folgende Funktionen:

- $S11$ und $S12$ registrieren Züge, die in den kritischen Bereich einfahren, sie werden *nicht* durch herausfahrende Züge aktiviert.
- $S31$ und $S32$ registrieren Züge, die in den kritischen Bereich einfahren wollen.
- $S21$ und $S22$ registrieren die Ausfahrt aus dem kritischen Bereich.

Die Züge müssen selbst auf die Signale achten und ggf. warten.
Im Fortgeschrittenenpraktikum wurde das Fallbeispiel mit Lego-Mindstorm-Technik realisiert (Abbildung 1.2). Die Controller-Steuerung wurde mit Moby/plc entworfen, automatisch generiert und verifiziert. Die Züge wurden direkt in C programmiert.

1.2 Sicherheitsanalyse

Um in einer Gefährdungsanalyse zu untersuchen, unter welchen Ümständen gefährliche Situationen eintreten können, kann die Fehlerbaumanalyse eingesetzt werden, in der schrittweise jede Störung auf Störungen von Teilkomponenten zurückgeführt wird. Dabei werden keine formalen Modelle und keine Beweise benutzt, sondern es wird nur „plausibel gemacht“, dass keine Ursachen für Störungen übersehen worden sind und der Baum „vollständig“ ist.
Es gibt verschiedene Ansätze die Fehlerbaumanalse mit formalen Methoden, wie dem präpositionalen modalen μ-Kalkül ([BA93]), stochastischen Petri-Netzen [Buc00], der schwächsten Vorbedingung [CM93] oder dem Duration Calculus ([Han96, RST00, RST01]) zu verbinden. Nach [RST01] sollen dadurch sowohl die Fehlerbaumanalyse als auch die formalen Methoden profitieren.

- Durch die Fehlerbaumanalyse kann die Suche nach Anforderungen systematisiert werden. Es besteht auch die Möglichkeit, die Auswirkung des Ausfalls einer Teilkomponente auf das Gesamtsystem zu untersuchen und so auch redundante Systeme zu berücksichtigen.
- Die Fehlerbaumanalyse gewinnt durch exakte Beurteilungen des Systemverhaltens.

Die Gefährdungsanalyse ist ein Teil der Sicherheitsanalyse, die u.a. auch die Folgenanalyse beinhaltet, die die Folgen einer Gefährdung untersucht. Eine Vorgehensmodell für eine Sicherheitsanalyse im Eisenbahnbereich wird z.B. in den neuen CENELEC Normen – besonders EN 50126 – empfohlen [Bra01]. Den Bahnbetreibern fallen dabei die Aufgaben zu,

- das System zu definieren,

- die Gefährdungen (Zug überfährt rotes Signal) zu identifizieren,
- die Folgen einer Gefährdung (Zusammenstoß von zwei Zügen) zu analysieren und
- und die Gefährdungen quantitativ zu bewerten.

Die Schnittstelle zu den Herstellern bilden die identifizierten und quantitativ bewerteten Gefährdungen.
Die Hersteller ihrerseits realisieren das System und führen für diese Realisierung eine Gefährdungsanalyse durch. Für den FunkFahrBetrieb der Bahn wird das Vorgehen bei der Fehlerbaumanalyse in [Ren01] dargestellt. Die gesamte Analyse muss danach durch die Eisenbahnbehörden überprüft werden.

1.3 Ziele und Gliederung der Arbeit

In dieser Arbeit wird die an der Universität Augsburg entwickelte Fehlerbaumsemantik [RST01] benutzt und untersucht, wie die Semantik mit Hilfe des von Josef Tapken [Tap01] eingeführten Formalismus der Phasenautomaten durch Model-Checking überprüft werden kann.
In Kapitel 2 werden die verwendeten Logiken, der Duration Calculus und zwei Erweiterungen davon, kurz vorgestellt und einige Lemmata bewiesen, die später verwendet werden.
In Kapitel 3 werden Phasenautomaten definiert. Dieses operationale Modell soll dann für das Model-Checking der Fehlerbäume verwendet werden. Deshalb wird zuerst gezeigt, dass Phasenautomaten äquivalent zu Realzeitautomaten und deshalb auch nicht im Allgemeinen komplementierbar sind. Für eine Teilklasse kann aber eine Komplementkonstruktion angegeben werden.
In Kapitel 4 werden Fehlerbäume zuerst informell und danach auch formal definiert, indem einem Fehlerbaum Formeln im Duration Calculus zugeordnet werden und gezeigt wird, wie ein Model-Checking mit Hilfe von Phasenautomaten möglich ist.
Diese Methode wird in Kapitel 5 an Fallbeispielen demonstriert.

2 Grundlegende Definitionen

2.1 Notationen

Definition 2.1.1 (Gleichheit von Funktionen). Zur Vereinfachung wird die folgende Notation aus [Tap01] benutzt. Seien A, B zwei Mengen und $g, g' : A \to B$ zwei Funktionen und $a \in A$. Dann sei $=_{\setminus a}$ als Gleichheit bis auf a definiert durch

$$g =_{\setminus a} g' \quad \text{gdw.} \quad \forall a' \in A \setminus \{a\} : g(a') = g'(a').$$

2.2 Duration Calculus

Der *Duration Calculus* (DC abgekürzt) ist eine temporale Logik zur Beschreibung von Realzeitsystemen und wurde in [ZHR91] vorgestellt. Sie erlaubt Aussagen darüber, wie lange ein Zustand andauert. Dabei wird die Zeit durch die Menge der nicht-negativen reellen Zahlen modelliert, d.h.

$$\mathbf{time} \stackrel{df}{=} \mathbb{R}_{\geq 0}$$

In dem Zusammenhang mit Zeitangaben wird die Abkürzung ZE für Zeiteinheiten verwendet. Der Zustand eines Systems zu einem bestimmten Zeitpunkt wird im DC durch sogenannte *Observablen* beschrieben. Das sind Variablen, deren Werte sich im Verlauf der Zeit ändern. Eine Interpretation $\mathcal{I}$ ordnet dann jeder Observablen X eine Funktion

$$\mathcal{I}(X) : \mathbf{time} \to \mathrm{ran}(X)$$

zu, die die Werte der Observablen zu allen Zeitpunkten beschreibt. Dabei wird verlangt, dass $\mathrm{ran}(X)$ endlich ist und die Funktion $\mathcal{I}(X)$ nur endliche Variabilität besitzt, d.h. dass in jedem endlichen Intervall nur endlich viele Unstetigkeitsstellen existieren. Der Fall, dass die Funktion einen Wert nur auf einem Punktintervall annimmt, wird ebenfalls ausgeschlossen. Die Menge der Observablen wird mit *Obs* bezeichnet. Zusätzlich können im DC globale Variablen benutzt werden, deren Wert sich im Zeitablauf nicht ändern. Die Werte dieser Variablen werden durch eine Belegung

$$\mathcal{V} : Vars \to \mathbb{R}$$

bestimmt, wobei die Menge der Variablen mit $Vars$ bezeichnet wird. Für die Beschreibung von Operationen werden Funktionssymbole f und Prädikatssymbole p benutzt, denen wie üblich Funktionen

$$\begin{aligned} \hat{f} : \mathbb{R}^n &\rightarrow \mathbb{R} \\ \hat{p} : \mathbb{R}^n &\rightarrow \{0, 1\} \end{aligned}$$

zugeordnet werden.
Das Verhalten des Systems aus dem Fallbeispiel Weichensteuerung kann z.B. durch die folgenden Observablen beschrieben werden.

- $ZugPos_A$ $\mathrm{ran}(ZugPos_A) = \{B_1, \ldots B_9\}$ Position von $\mathbf{Zug}_A$
- $ZugPos_B$ $\mathrm{ran}(ZugPos_B) = \{B_1, \ldots B_9\}$ Position von $\mathbf{Zug}_B$
- $ZugDir_A$ $\mathrm{ran}(ZugDir_A) = \{vor, rueck, stop\}$ Fahrtrichtung von $\mathbf{Zug}_A$
- $ZugDir_B$ $\mathrm{ran}(ZugDir_B) = \{vor, rueck, stop\}$ Fahrtrichtung von $\mathbf{Zug}_B$
- $Signal_1$ $\mathrm{ran}(Signal_1) = \{rot, gruen\}$ Status des linken Signals
- $Signal_2$ $\mathrm{ran}(Signal_2) = \{rot, gruen\}$ Status des rechten Signals
- $Sensor_{xy}$ $\mathrm{ran}(Sensor_{xy}) = \{aktiv, deaktiv\}$ Eine Observable für jeden Sensor

2.2.1 Zustandszusicherungen

Zustandszusicherungen (auch Zustandsausdrücke) sind Aussagen über den Zustand des betrachteten Systems zu einem bestimmten Zeitpunkt, zum Beispiel, dass $Signal_1$ grün ist.

Definition 2.2.1 (Syntax von Zustandszusicherungen). Zustandszusicherungen werden mit π bezeichnet und durch folgende Grammatik erzeugt

$$\pi ::= 0 \,|\, 1 \,|\, X = d \,|\, \neg\pi_1 \,|\, \pi_1 \wedge \pi_2.$$

Dabei ist X eine Observable, d ein Wert aus $\mathrm{ran}(X)$ und 0 und 1 stellen die boolschen Konstanten true und false dar. Die anderen boolschen Verknüpfungen werden als Abkürzungen definiert. Für eine Menge Obs von Observablen wird die Menge aller Zustandszusicherungen über diesen Observablen mit $SA(Obs)$ bezeichnet. Falls die Observablen aus dem Zusammenhang erkennbar sind, wird die Menge Obs nicht explizit angegeben und es wird anstatt $X = d$ einfach d geschrieben, falls die zu d gehörende Observable klar ist.

Definition 2.2.2 (Semantik von Zustandszusicherungen). Sei π eine Zustandszusicherung. Dann ist die Semantik von π unter einer Interpretation $\mathcal{I}$ eine Funktion

$$\mathcal{I}[\![\pi]\!] : \mathbf{time} \rightarrow \{0, 1\},$$

die wie folgt induktiv definiert ist:

- $\mathcal{I}[\![0]\!](t) \stackrel{df}{=} 0$
- $\mathcal{I}[\![1]\!](t) \stackrel{df}{=} 1$
- $\mathcal{I}[\![X = d]\!](t) \stackrel{df}{=} \begin{cases} 1 & \text{falls } \mathcal{I}(X)(t) = d \\ 0 & \text{sonst} \end{cases}$
- $\mathcal{I}[\![\neg\pi]\!](t) \stackrel{df}{=} 1 - \mathcal{I}[\![\pi]\!](t)$
- $\mathcal{I}[\![\pi_1 \wedge \pi_2]\!](t) \stackrel{df}{=} \mathcal{I}[\![\pi_1]\!](t) \cdot \mathcal{I}[\![\pi_2]\!](t)$

2.2.2 DC-Terme

Die Zeit, die ein Zustand in einem vorgegebenen Intervall eine Zustandszusicherung erfüllt, kann durch Integration gemessen werden. DC-Terme beschreiben diese Zeit.

Definition 2.2.3 (Syntax von DC-Termen). DC-Terme werden mit θ bezeichnet und durch folgende Grammatik erzeugt:

$$\theta ::= x \mid \ell \mid \int \pi \mid f(\theta_1, \dots, \theta_n)$$

Hier ist $x \in Vars$ eine globale Variable, ℓ ein Symbol für die Länge des Intervalls, π eine Zustandszusicherung und f ein Funktionsymbol.

Definition 2.2.4 (Semantik von DC-Termen). Die Semantik eines DC-Terms θ unter einer Interpretation $\mathcal{I}$ und einer Belegung $\mathcal{V}$ ist für ein Intervall $[b, e] \subseteq$ **time** definiert durch

- $\mathcal{I}[\![x]\!](\mathcal{V}, [b, e]) \stackrel{df}{=} \mathcal{V}(x)$
- $\mathcal{I}[\![\ell]\!](\mathcal{V}, [b, e]) \stackrel{df}{=} e - b$
- $\mathcal{I}[\![\int \pi]\!](\mathcal{V}, [b, e]) \stackrel{df}{=} \int_b^e \mathcal{I}[\![\pi]\!](t)dt$
- $\mathcal{I}[\![f(\theta_1, \dots \theta_n)]\!](\mathcal{V}, [b, e]) \stackrel{df}{=} \hat{f}(\mathcal{I}[\![\theta_1]\!](\mathcal{V}, [b, e]), \dots (\mathcal{I}[\![\theta_n]\!](\mathcal{V}, [b, e]))$

2.2.3 DC-Formeln

Mit Hilfe der DC-Terme können dann auch DC-Formeln definiert werden.

Definition 2.2.5 (Syntax von DC-Formeln). DC-Formeln werden mit D bezeichnet und durch folgende Grammatik erzeugt.

$$D ::= \mathsf{true} \mid p(\theta_1, \ldots \theta_n) \mid \neg D \mid D_1 \wedge D_2 \mid \exists x : D \mid D_1 ; D_2$$

Dabei sind $\theta_1, \ldots, \theta_n$ DC-Terme, p ein Prädikatssymbol der Stelligkeit n und $x \in Vars$ eine Variable. Der neue Operator ; heißt Chop-Operator, die anderen boolschen Verknüpfungen werden wieder als Abkürzungen definiert.

Definition 2.2.6 (Semantik von DC-Formeln). Die Erfüllbarkeitsrelation $\models_{DC}$ einer DC-Formel ist induktiv definiert. Sei $\mathcal{I}$ eine Interpretation, $\mathcal{V}$ eine Belegung und $[b, e] \subseteq \mathbf{time}$ ein Intervall. Dann gilt:

- $\mathcal{I}, \mathcal{V}, [b, e] \models_{DC} \mathsf{true}$ immer
- $\mathcal{I}, \mathcal{V}, [b, e] \models_{DC} p(\theta_1, \ldots, \theta_n)$ gdw. $\hat{p}(\mathcal{I}[\![\theta_1]\!](\mathcal{V}, [b, e]), \ldots, \mathcal{I}[\![\theta_n]\!](\mathcal{V}[b, e])$
- $\mathcal{I}, \mathcal{V}, [b, e] \models_{DC} \neg D_1$ gdw. $\mathcal{I}, \mathcal{V}, [b, e] \not\models_{DC} D_1$
- $\mathcal{I}, \mathcal{V}, [b, e] \models_{DC} D_1 \wedge D_2$ gdw. $\mathcal{I}, \mathcal{V}, [b, e] \models_{DC} D_1$ und $\mathcal{I}, \mathcal{V}, [b, e] \models_{DC} D_2$
- $\mathcal{I}, \mathcal{V}, [b, e] \models_{DC} \exists x : D$ gdw. Es existiert eine Belegung $\mathcal{V}'$ mit $\mathcal{V} =_{\backslash x} \mathcal{V}'$ und $\mathcal{I}, \mathcal{V}', [b, e] \models_{DC} D$
- $\mathcal{I}, \mathcal{V}, [b, e] \models_{DC} D_1 ; D_2$ gdw. ein $m \in [b, e]$ existiert mit
  - $\mathcal{I}, \mathcal{V}, [b, m] \models_{DC} D_1$ und
  - $\mathcal{I}, \mathcal{V}, [m, e] \models_{DC} D_2$.

Da es für Realzeitsysteme sinnvoll ist, Abläufe vom Zeitpunkt 0 an zu betrachten, wird definiert, dass eine Interpretation $\mathcal{I}$ und eine Belegung $\mathcal{V}$ eine DC-Formel D erfüllen, falls die Formel für alle bei 0 beginnenden Intervalle erfüllt ist, d.h.

$$\mathcal{I}, \mathcal{V} \models_{DC} D \quad \text{gdw.} \quad \text{für alle } t \in \mathbf{time} \text{ gilt } \mathcal{I}, \mathcal{V}, [0, t] \models_{DC} D$$

Eine DC-Formel D heißt *erfüllbar*, falls ein Paar $(\mathcal{I}, \mathcal{V})$ existiert mit $\mathcal{I}, \mathcal{V} \models_{DC} D$. Wenn alle Paare $(\mathcal{I}, \mathcal{V})$ eine Formel D erfüllen, heißt D *gültig* oder *Tautologie*:

$$\models_{DC} D$$

Die Semantik einer DC-Formel ist die Menge aller Paare $(\mathcal{I}, \mathcal{V})$ aus Interpretationen und Belegungen, die die Formel erfüllen.

$$[\![D]\!] \stackrel{df}{=} \{(\mathcal{I}, \mathcal{V}) | (\mathcal{I}, \mathcal{V}) \models_{DC} D\}$$

2.2.4 Abkürzungen

Folgende Abkürzungen werden benutzt:

- $b \sim_1 \ell \sim_2 e \stackrel{df}{=} b \sim_1 \ell \wedge \ell \sim_2 e, \quad \sim_i \in \{<, \leq, >, \geq\}$
- $\lceil\rceil \stackrel{df}{=} \ell = 0$ (Punktintervall)
- $\lceil\pi\rceil \stackrel{df}{=} (\int \pi = \ell) \wedge (\ell > 0)$ (everywhere)
- $\lceil\pi\rceil^* \stackrel{df}{=} (\int \pi = \ell)$
- $\lceil\pi\rceil^t \stackrel{df}{=} (\int \pi = \ell) \wedge (\ell = t)$
- $\lceil\pi\rceil^{\sim t} \stackrel{df}{=} (\int \pi = \ell) \wedge (\ell \sim t), \quad \sim \in \{<, \leq, >, \geq\}$
- $\Diamond D \stackrel{df}{=} \mathsf{true}; D; \mathsf{true}$ (sometimes)
- $\Box D \stackrel{df}{=} \neg\Diamond\neg D$ (always)
- $D \longrightarrow \lceil\pi\rceil \stackrel{df}{=} \neg\Diamond(D; \lceil\neg\pi\rceil)$ (followed-by)
- $D \xrightarrow{t} \lceil\pi\rceil \stackrel{df}{=} (D \wedge \ell = t) \longrightarrow \lceil\pi\rceil$ (times leads-to)
- $D \xrightarrow{\leq t} \lceil\pi\rceil \stackrel{df}{=} (D \wedge \ell \leq t) \longrightarrow \lceil\pi\rceil$ (timed up-to)

Mit dieser Definition kann der Fall, dass sich zwei Züge gleichzeitig in Abschnitt B_1 befinden durch die Formel

$$\Diamond\lceil ZugPos_A = B_1 \wedge ZugPos_B = B_1\rceil$$

beschrieben werden. In dem betrachteten Beispiel ist dieser Fall selbst noch kein Fehlerfall. Es soll erlaubt sein, dass zwei Züge direkt nacheinander in gleicher Richtung durch den kritischen Bereich fahren.

2.2.5 S-Formeln

Für die Definition des verwendeten Automaten-Modells muss der Duration Calculus um Quantifizierungen über Observablen erweitert werden. Dazu wird in [Tap01] der Begriff der Spezifikations-Formeln (kurz S-Formeln) aus [Gra97] leicht verändert übernommen.

Definition 2.2.7 (Syntax von S-Formeln). Für S-Formeln wird der Buchstabe F verwendet und sie werden durch folgende Grammatik erzeugt:

$$F ::= [D] \,|\, \neg F \,|\, F_1 \wedge F_2 \,|\, \exists X : F \,|\, \exists x : F$$

wobei D eine DC-Formel, X eine Observable und x eine Variable ist. Andere Quantoren und boolschen Verknüpfungen können wieder als Abkürzungen definiert werden. Der $[\cdot]$-Operator quantifiziert über alle bei Null beginnenden Intervalle.

Definition 2.2.8 (Semantik von S-Formeln). Die Erfüllbarkeitsrelation von S-Formeln für eine Interpretation $\mathcal{I}$ und eine Belegung $\mathcal{V}$ ist ähnlich wie bei den DC-Formeln definiert.

- $\mathcal{I}, \mathcal{V} \models [D]$ gdw. $\mathcal{I}, \mathcal{V} \models_{DC} D$
- $\mathcal{I}, \mathcal{V} \models \neg F$ gdw. $\mathcal{I}, \mathcal{V} \not\models F$
- $\mathcal{I}, \mathcal{V} \models F_1 \wedge F_2$ gdw. $\mathcal{I}, \mathcal{V} \models F_1$ und $\mathcal{I}, \mathcal{V} \models F_2$
- $\mathcal{I}, \mathcal{V} \models \exists X : F$ gdw. Es existiert eine Interpretation $\mathcal{I}'$ mit $\mathcal{I} =_{\setminus X} \mathcal{I}'$ und $\mathcal{I}', \mathcal{V} \models F$
- $\mathcal{I}, \mathcal{V} \models \exists x : F$ gdw. Es existiert eine Belegung $\mathcal{V}'$ mit $\mathcal{V} =_{\setminus x} \mathcal{V}'$ und $\mathcal{I}, \mathcal{V}' \models F$

Die Semantik $[\![\cdot]\!]$ ist dann auch definiert durch

$$[\![F]\!] \stackrel{df}{=} \{(\mathcal{I}, \mathcal{V}) | (\mathcal{I}, \mathcal{V}) \models F\}$$

2.2.6 Duration Calculus with Liveness

Da für die formale Semantik von Fehlerbäumen ein "echter" Eventually-Operator benötigt wird, wird der Duration Calculus with Liveness (DCL) ([Ska94]) verwendet, der zwei zusätzliche Chop-Operatoren einführt, aber sonst die Semantik unverändert lässt.

Definition 2.2.9 (Syntax von DCL-Formeln). DCL-Formeln werden mit A bezeichnet und durch folgende Grammatik erzeugt:

$$A ::= \mathsf{true} \,|\, p(\theta_1, \ldots \theta_n) \,|\, \neg A \,|\, A_1 \wedge A_n \,|\, \exists x : A \,|\, A_1 ; A_2 | A_1 \lhd A_2 | A_1 \rhd A_2$$

Definition 2.2.10 (Sematik von DCL-Formeln). Die Semantik von DCL-Formeln entspricht der von DC-Formeln, es müssen nur die neuen Chop-Operatoren definiert werden. Aus diesem Grund wird die erweiterte Erfüllbarkeitsrelation des DCL ebenfalls mit $\models_{DC}$ bezeichnet.

- $\mathcal{I}, \mathcal{V}, [b,e] \models_{DC} A_1 \lhd A_2$ gdw. ein $k \leq b$ existiert mit
  - $\mathcal{I}, \mathcal{V}, [k,b] \models_{DC} A_1$ und
  - $\mathcal{I}, \mathcal{V}, [k,e] \models_{DC} A_2$.

- $\mathcal{I}, \mathcal{V}, [b,e] \models_{DC} A_1 \rhd A_2$ gdw. ein $k \geq e$ existiert mit
  - $\mathcal{I}, \mathcal{V}, [b,k] \models_{DC} A_1$ und
  - $\mathcal{I}, \mathcal{V}, [e,k] \models_{DC} A_2$.

Mit diesen Operatoren werden in [Ska94] folgende Abkürzungen definiert:

- $\Diamond_L A \overset{df}{=} (\mathsf{true}; A; \mathsf{true}) \rhd \mathsf{true}$ (echtes eventually)
- $\Box_L A \overset{df}{=} \neg \Diamond_L \neg A$ (echtes always)

Hier steht das "L" für Liveness. In [Ska94] werden statt der Symbole $\Diamond_L$ und $\Box_L$ die Symbole $\Box$ und $\Diamond$ benutzt und die Standard-DC Operatoren mit $\Diamond_D$ und $\Box_D$ bezeichnet. Außerdem wird hier folgende Abkürzung benutzt:

- $\Box_{L0} A \overset{df}{=} \neg(\neg A \rhd \mathsf{true}) \wedge \lceil\rceil$ (für alle bei Null beginnenden Intervalle)

Für DCL-Formeln sind folgende Lemmata hilfreich:

Lemma 2.2.11 ($\rhd$true-Ergänzung). Sei π ein Zustandsausdruck und $b \in$ **time** ein Zeitpunkt. Dann gilt:

$$\exists\, e \in \textbf{time}\ \mathcal{I}, \mathcal{V},\ [b,e] \models_{DC} \lceil \pi \rceil \iff \mathcal{I}, \mathcal{V},\ [b,b] \models_{DC} (\lceil \pi \rceil \rhd \mathsf{true}) \wedge \lceil\rceil$$

Der Beweis folgt direkt aus der Definition des $\rhd$-Operators.

Lemma 2.2.12 (Beginn-Intervall). Sei A eine DCL-Formel. Dann gilt

$$\models_{DC} \Box_{L0} A \Rightarrow (A \wedge \neg\lceil\rceil) \rhd \mathsf{true}$$

Beweis

$$
\begin{aligned}
& & \mathcal{I},\mathcal{V},\ [b,e] \models_{DC} \boxed{L}_0 A \\
\{\text{Def. } \boxed{L}_0\} \Rightarrow & & \mathcal{I},\mathcal{V},\ [b,e] \models_{DC} \neg(\neg A \rhd \mathsf{true}) \wedge \lceil\rceil \\
\{\text{Def. } \lceil\rceil\} \Rightarrow & & \mathcal{I},\mathcal{V},\ [b,e] \models_{DC} \neg(\neg A \rhd \mathsf{true}) \quad \text{und}\, b = e \\
\Rightarrow & & \mathcal{I},\mathcal{V},\ [b,b] \models_{DC} \neg(\neg A \rhd \mathsf{true}) \quad \text{und}\, b = e \\
\{\text{Def. } \rhd\} \Rightarrow & \nexists\, k \geq b : & \mathcal{I},\mathcal{V},\ [b,k] \models_{DC} \neg A \quad \text{und}\, b = e \\
\{\text{Praed. Logik}\rhd\} \Rightarrow & \nexists\, k > b : & \mathcal{I},\mathcal{V},\ [b,k] \models_{DC} \neg A \quad \text{und}\, b = e \\
\{\text{Praed. Logik}\} \Rightarrow & \not\forall\, k > b : & \mathcal{I},\mathcal{V},\ [b,k] \models_{DC} \neg A \quad \text{und}\, b = e \\
\{\text{Praed. Logik}\} \Rightarrow & \exists\, k > b : & \mathcal{I},\mathcal{V},\ [b,k] \models_{DC} A \quad \text{und}\, b = e \\
\{\text{Def. } \lceil\rceil\} \Rightarrow & \exists\, k > b : & \mathcal{I},\mathcal{V},\ [b,k] \models_{DC} A \wedge \neg\lceil\rceil \quad \text{und}\, b = e \\
\{\text{Def. } \rhd\} \Rightarrow & & \mathcal{I},\mathcal{V},\ [b,b] \models_{DC} (A \wedge \neg\lceil\rceil) \rhd \mathsf{true} \quad \text{und}\, b = e \\
\Rightarrow & & \mathcal{I},\mathcal{V},\ [b,e] \models_{DC} (A \wedge \neg\lceil\rceil) \rhd \mathsf{true}
\end{aligned}
$$

□

Lemma 2.2.13 ($\vee - \rhd$-Distributiv). Es gilt:

$$\mathcal{I},\mathcal{V},\ [b,b] \models_{DC} (\lceil\pi_1 \vee \pi_2\rceil \rhd \mathsf{true}) \iff \big((\lceil\pi_1\rceil \rhd \mathsf{true}) \vee (\lceil\pi_2\rceil \rhd \mathsf{true})\big)$$

Beweis

Die $\Leftarrow$-Richtung ist klar. Für die $\Rightarrow$-Richtung gilt:

$$
\begin{aligned}
& \mathcal{I},\mathcal{V},\ [b,b] \models_{DC} \lceil\pi_1 \vee \pi_2\rceil \rhd \mathsf{true} \\
\{\text{Nach Def.}\rhd\} & \\
\{\text{existiert ein k}\} \Rightarrow & \mathcal{I},\mathcal{V},\ [b,k] \models_{DC} \lceil\pi_1 \vee \pi_2\rceil \\
\{\text{endl. Variabilität}\} \Rightarrow & \mathcal{I},\mathcal{V},\ [b,k] \models_{DC} \lceil\pi_1 \vee \pi_2\rceil \wedge \big((\lceil\pi_1\rceil;\mathsf{true}) \vee (\lceil\neg\pi_1\rceil;\mathsf{true})\big) \\
\text{1. Fall} \quad & \mathcal{I},\mathcal{V},\ [b,k] \models_{DC} \lceil\pi_1 \vee \pi_2\rceil \wedge (\lceil\pi_1\rceil;\mathsf{true}) \\
\{\text{Nach Def.};\} & \\
\{\text{existiert ein k'}\} \Rightarrow & \mathcal{I},\mathcal{V},\ [b,k'] \models_{DC} \lceil\pi_1\rceil \\
\Rightarrow & \mathcal{I},\mathcal{V},\ [b,b] \models_{DC} \lceil\pi_1\rceil \rhd \mathsf{true} \\
\text{2. Fall} \quad & \mathcal{I},\mathcal{V},\ [b,k] \models_{DC} \lceil\pi_1 \vee \pi_2\rceil \wedge (\lceil\neg\pi_1\rceil;\mathsf{true}) \\
\{\text{Nach Def.};\} & \\
\{\text{existiert ein k'}\} \Rightarrow & \mathcal{I},\mathcal{V},\ [b,k'] \models_{DC} \lceil\pi_1 \vee \pi_2\rceil \wedge \lceil\neg\pi_1\rceil \\
\Rightarrow & \mathcal{I},\mathcal{V},\ [b,k'] \models_{DC} \lceil\pi_2\rceil \\
\Rightarrow & \mathcal{I},\mathcal{V},\ [b,b] \models_{DC} \lceil\pi_2\rceil \rhd \mathsf{true}
\end{aligned}
$$

□

Lemma 2.2.14 ($\rhd$true-Ergänzung). Sei π ein Zustandsausdruck. Dann gilt:

$$(\lceil\pi\rceil;\mathsf{true}) \equiv (\lceil\pi\rceil;\mathsf{true}) \rhd \mathsf{true} \wedge \neg\lceil\rceil$$

Beweis Sei $\mathcal{I}$ eine Interpretation, $\mathcal{V}$ eine Belegung und $[b, e]$ ein Intervall. **„⇒“-Richtung**

$$\begin{aligned}
& \mathcal{I}, \mathcal{V},\ [b, e] \models_{DC} \lceil \pi \rceil ; \mathsf{true} \\
\{\text{Def. } \lceil \pi \rceil\} \Rightarrow\quad & \mathcal{I}, \mathcal{V},\ [b, e] \models_{DC} (\lceil \pi \rceil ; \mathsf{true}) \wedge \neg \lceil\rceil \\
\{\text{Def. } \} \Rightarrow\quad & \mathcal{I}, \mathcal{V},\ [b, e] \models_{DC} (\lceil \pi \rceil ; \mathsf{true}) \wedge \neg \lceil\rceil \quad \text{und} \quad \mathcal{I}, \mathcal{V},\ [e, e] \models_{DC} \mathsf{true} \\
\{\text{Def. } \rhd\} \Rightarrow\quad & \mathcal{I}, \mathcal{V},\ [b, e] \models_{DC} ((\lceil \pi \rceil ; \mathsf{true}) \rhd \mathsf{true}) \wedge \neg \lceil\rceil
\end{aligned}$$

„⇐“-Richtung

$$\begin{aligned}
& \mathcal{I}, \mathcal{V},\ [b, e] \models_{DC} ((\lceil \pi \rceil ; \mathsf{true}) \rhd \mathsf{true}) \\
\{\text{Def. } \rhd\} \Rightarrow\quad & \exists k \geq e : \mathcal{I}, \mathcal{V},\ [b, k] \models_{DC} (\lceil \pi \rceil ; \mathsf{true}) \\
\{\text{Def. } ;\} \Rightarrow\quad & \exists k, k' : b \leq k' \leq k, e \leq k,\ \mathcal{I}, \mathcal{V},\ [b, k'] \models_{DC} \lceil \pi \rceil \\
\text{1. Fall } k' \leq e & \\
\{\text{Def. } ;\} \Rightarrow\quad & \mathcal{I}, \mathcal{V},\ [b, e] \models_{DC} \lceil \pi \rceil ; \mathsf{true} \\
\text{2. Fall } k' > e & \\
\{b < e\} \Rightarrow\quad & \mathcal{I}, \mathcal{V},\ [b, e] \models_{DC} \lceil \pi \rceil \\
\Rightarrow\quad & \mathcal{I}, \mathcal{V},\ [b, e] \models_{DC} (\lceil \pi \rceil ; \mathsf{true})
\end{aligned}$$

□

Da DCL-Formeln auch für Intervalle definiert sind, kann der $[\cdot]$-Operator in S-Formeln auf DCL-Formeln genauso wie auf DC-Formeln angewendet werden. Es gelten dann folgende Beziehungen zwischen S-Formeln und DCL-Formeln.

Lemma 2.2.15. Seien A, A_1, A_2 DCL-Formeln, $\mathcal{I}$ eine Interpretation und $\mathcal{V}$ eine Belegung. Dann gilt:

1. $\mathcal{I}, \mathcal{V} \models [A] \iff \mathcal{I}, \mathcal{V},\ [0, 0] \models_{DC} \boxed{L}_0 A$

2. $\mathcal{I}, \mathcal{V},\ [b, b] \models_{DC} \Diamond_L A \iff \forall\, e \in \mathbf{time}, e \geq b : \mathcal{I}, \mathcal{V},\ [b, e] \models_{DC} \Diamond_L A$

3. $\boxed{L}_0 A_1 \wedge \boxed{L}_0 A_2 \equiv \boxed{L}_0 (A \wedge B)$

4. $\mathcal{I}, \mathcal{V},\ [b, b] \models_{DC} \boxed{L} A \iff \exists\, e \in \mathbf{time}, e \geq b : \mathcal{I}, \mathcal{V},\ [b, e] \models_{DC} \boxed{L} A$

Beweis 1. Teil:

$$
\begin{array}{rcr}
 & & \mathcal{I},\mathcal{V},\ [0,0] \models_{DC} \boxed{L}_0 A \\
\{\text{Def. } \boxed{L}_0\} & \iff & \mathcal{I},\mathcal{V},\ [0,0] \models_{DC} \neg(\neg A \rhd \mathsf{true}) \wedge \lceil\rceil \\
 & \iff & \mathcal{I},\mathcal{V},\ [0,0] \models_{DC} \neg(\neg A \rhd \mathsf{true}) \\
\{\text{Def. } \neg\} & \iff & \mathcal{I},\mathcal{V},\ [0,0] \not\models_{DC} (\neg A \rhd \mathsf{true}) \\
\{\text{Def. } \rhd\} & \iff & \nexists\, e \geq 0 : \mathcal{I},\mathcal{V},\ [0,e] \models_{DC} \neg A \\
 & \iff & \forall\, e \geq 0 : \mathcal{I},\mathcal{V},\ [0,e] \not\models_{DC} \neg A \\
\{\text{Def. } [\cdot]\} & \iff & \mathcal{I},\mathcal{V} \models [A]
\end{array}
$$

2. Teil:
Hier ist nur eine Richtung zu beweisen. Sei $e \in \mathbf{time}$, dann gilt:

$$
\begin{array}{rcr}
 & & \mathcal{I},\mathcal{V},\ [b,b] \models_{DC} \Diamond_{L} A \\
\{\text{Def. } \Diamond_{L}\} & \Rightarrow & \mathcal{I},\mathcal{V},\ [b,b] \models_{DC} (\mathsf{true}; A; \mathsf{true}) \rhd \mathsf{true} \\
\{\text{Def. } \rhd\} & \Rightarrow & \exists\, k \geq b\ :\ \mathcal{I},\mathcal{V},\ [b,k] \models_{DC} (\mathsf{true}; A; \mathsf{true}) \\
\text{1.Fall } e \leq k : & & \\
 & \Rightarrow & \exists\, k \geq e\ :\ \mathcal{I},\mathcal{V},\ [b,k] \models_{DC} (\mathsf{true}; A; \mathsf{true}) \\
\{\text{Def. } \rhd\} & \Rightarrow & \mathcal{I},\mathcal{V},\ [b,e] \models_{DC} (\mathsf{true}; A; \mathsf{true}) \rhd \mathsf{true} \\
\text{2.Fall } e > k : & & \\
 & \Rightarrow & \mathcal{I},\mathcal{V},\ [b,e] \models_{DC} (\mathsf{true}; A; \mathsf{true}) \\
 & \Rightarrow & \mathcal{I},\mathcal{V},\ [b,e] \models_{DC} (\mathsf{true}; A; \mathsf{true}) \rhd \mathsf{true}
\end{array}
$$

3. Teil:

$$
\begin{array}{rcr}
 & & \boxed{L}_0 A_1 \wedge \boxed{L}_0 A_2 \\
\{\text{Def. } \boxed{L}_0\} & \equiv & \neg(\neg A_1 \rhd \mathsf{true}) \wedge \neg(\neg A_2 \rhd \mathsf{true}) \wedge \lceil\rceil \\
\{\text{DeMorgan }\} & \equiv & \neg\,((\neg A_1 \rhd \mathsf{true}) \vee (\neg A_2 \rhd \mathsf{true})) \wedge \lceil\rceil \\
\{\text{Distrib. [Ska94]}\} & \equiv & \neg\,((\neg A_1 \vee \neg A_2) \rhd \mathsf{true}) \wedge \lceil\rceil \\
\{\text{DeMorgan}\} & \equiv & \neg\,((\neg(A_1 \wedge A_2)) \rhd \mathsf{true}) \wedge \lceil\rceil \\
\{\text{Def. } \boxed{L}_0\} & \equiv & \boxed{L}_0(A_1 \wedge A_2)
\end{array}
$$

4. Teil:

$$
\begin{array}{rcr}
 & & \mathcal{I}, \mathcal{V},\ [b,b] \models_{DC} \boxed{L}A \\
\{\text{Def.}\boxed{L}\} & \Longleftrightarrow & \mathcal{I}, \mathcal{V},\ [b,b] \models_{DC} \neg \Diamond_{L} \neg A \\
 & \Longleftrightarrow & \mathcal{I}, \mathcal{V},\ [b,b] \not\models_{DC} \Diamond_{L} \neg A \\
\{\text{Teil (2)}\} & \Longleftrightarrow & \neg(\forall e \geq b : \mathcal{I}, \mathcal{V},\ [b,e] \models_{DC} \Diamond_{L} \neg A) \\
\{\text{Def.}\boxed{L}\} & \Longleftrightarrow & \neg(\forall e \geq b : \mathcal{I}, \mathcal{V},\ [b,e] \models_{DC} \neg \boxed{L} A) \\
 & \Longleftrightarrow & \exists e \geq b : \mathcal{I}, \mathcal{V},\ [b,e] \models_{DC} \boxed{L}A
\end{array}
$$

□

2.3 Intervall-Ausdrücke

Um die Dauer von Phasen mit Intervallen vergleichen zu können, werden Intervallausdrücke definiert, mit denen Aussagen darüber getroffen werden können, ob die Dauer einer Phase innerhalb eines Zeitintervalls liegt.

Definition 2.3.1 (Intervall-Ausdrücke). Intervallausdrücke werden mit ie bezeichnet und durch folgende Grammatik erzeugt:

$$ie := \langle ex_1, ex_2 \rangle | \langle ex, \infty]$$

wobei ex_1 und ex_2 arithmetische Ausdrücke und $\langle \in \{[, (\}, \rangle \in \{),]\}$ sind. Die arithmetischen Ausdrücke können auch Variablen enthalten. Die Menge aller Intervallausdrücke wird mit IE bezeichnet.

Definition 2.3.2 (Größer/Kleiner-Relationen für Intervalle). Sei $= \langle ex_1, ex \rangle$, $\langle \in \{[, (\}, \rangle \in \{),]\}$ ein Intervallausdruck. Dann sei

$$\ell < ie \stackrel{df}{=} \begin{cases} \ell < ex_1 & \text{falls } \langle = [\\ \ell \leq ex_1 & \text{sonst} \end{cases}$$

und

$$\ell > ie \stackrel{df}{=} \begin{cases} \ell > ex_1 & \text{falls } \rangle =] \\ \ell \geq ex_1 & \text{sonst} \end{cases}$$

und $\ell > \infty \stackrel{df}{=} \mathsf{false}$.

3 Phasenautomaten

In [Tap01] werden Phasenautomaten als Zwischensprache für das Model-Checking von Duration Calculus Spezifikationen vorgestellt. Phasenautomaten besitzen eine formale Semantik im DC und mit Hilfe des Tools Moby/dc kann für eine Menge von Phasenautomaten (Netzwerk von Phasenautomaten) überprüft werden, ob sie einen gemeinsamen Ablauf besitzen.

3.1 Definition von Phasenautomaten

Phasenautomaten sind ein operationelles Modell für eine Teilmenge des Duration Calculus. Die Zustände eines Phasenautomaten sind mit DC-Phasen ($\lceil \cdot \rceil$) beschriftet. Das bedeutet, dass sich das System zu diesem Zeitpunkt in einer solchen Phase befindet. Über Uhren kann die Verweildauer in einem Zustand beschränkt werden. Dabei können jeder Uhr mehrere Zustände zugeordnet sein und jedem Zustand mehrere Uhren.

Definition 3.1.1 (Syntax von Phasenautomaten). Ein Phasenautomat $\mathcal{A}$ ist ein Tupel $\mathcal{A} = (P, E, C, cl, s, d, P_0)$, wobei

- P eine endliche Menge von Zuständen,
- $E \subseteq P \times P$ die Transitionsrelation,
- C eine endliche Menge von Uhren,
- $cl : P \to 2^C$ eine Abbildung, die jedem Zustand eine Menge von Uhren zuordnet,
- $s : P \to SA(Obs)$ eine Abbildung, die jedem Zustand einen Zustandsausdruck zuordnet,
- $d : C \to IE$ eine Abbildung, die jeder Uhr einen Intervallausdruck zuordnet, der auch Variablen enthalten kann und die Verweildauer in dem entsprechenden Uhrenbereich beschränkt und
- $P_0 \subseteq P$ eine Menge von Startzuständen ist.

Die Zustände eines Phasenautomaten werden Phasen genannt.

Definition 3.1.2 (Semantik von Phasenautomaten). Die Semantik eines Phasenautomaten $\mathcal{A}$ wird durch eine S-Formel $F(\mathcal{A})$ beschrieben, die wie folgt definiert ist.

$$D(\mathcal{A}) \stackrel{df}{=} D_{init}(\mathcal{A}) \wedge \bigwedge_{p \in P} (D^p_{suc}(\mathcal{A}) \wedge D^p_{out}(\mathcal{A})) \wedge \bigwedge_{c \in C} (D^c_{pr}(\mathcal{A}) \wedge D^c_{st}(\mathcal{A}) \wedge D^c_{ist}(\mathcal{A}))$$

$$F(\mathcal{A}) \stackrel{df}{=} \exists\, \mathsf{ph}_{\mathcal{A}}\, [D(\mathcal{A})]$$

Die in der S-Formel eingeführte Observable $\mathsf{ph}_{\mathcal{A}}$ hat als Bildbereich die Menge der Zustände des Phasenautomaten und beschreibt den Ablauf des Phasenautomaten, also welche Phase des Automaten zu welchem Zeitpunkt angenommen wird. Die einzelnen Teilformeln kodieren die Bedingungen an einen Ablauf des Phasenautomaten.

- $D_{init}(\mathcal{A}) \stackrel{df}{=} \neg(\lceil \mathsf{ph}_{\mathcal{A}} \notin P_0 \rceil; \mathsf{true})$ verlangt, dass die initiale Phase des Ablaufs $\mathsf{ph}_{\mathcal{A}}$ eine Startphase ist.

- $D^p_{suc}(\mathcal{A}) \stackrel{df}{=} \neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}} = p \rceil; \lceil \mathsf{ph}_{\mathcal{A}} \notin \widetilde{\mathrm{suc}}(p) \rceil)$ verlangt, dass nur zwei aufeinander folgende Phasen in einem Ablauf auch Nachfolger im Automaten sein müssen. Dabei ist $\widetilde{\mathrm{suc}}(p) = \{p\} \cup \{p' | p \to p' \in E\}$ die Menge aller Nachfolger von p und p selber.

- $D^p_{out}(\mathcal{A}) \stackrel{df}{=} \neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}} = p \wedge \neg s(p) \rceil)$ formalisiert, dass eine Phase p des Automaten nur angenommen werden darf, wenn zu diesem Zeitpunkt der Zustandsausdruck $s(p)$ erfüllt ist.

- $D^c_{pr}(\mathcal{A}) \stackrel{df}{=} \neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}} \in cl^{-1}(c) \rceil^{>d(c)})$ fordert, dass sich der Automat nicht länger im Uhrenbereich einer Uhr c aufhalten darf, als die obere Grenze des Intervalls $d(c)$ angibt.

- $D^c_{st}(\mathcal{A}) \stackrel{df}{=} \neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}} \notin cl^{-1}(c) \rceil; \lceil \mathsf{ph}_{\mathcal{A}} \in cl^{-1}(c) \rceil^{<d(c)}; \lceil \mathsf{ph}_{\mathcal{A}} \notin cl^{-1}(c) \rceil)$ legt fest, dass der Automat mindestens so lange Zustände aus dem Uhrenbereich von c annehmen muss, wie das Intervall angibt. Die initiale Stabilität muss extra durch die DC-Formel

- $D^c_{ist}(\mathcal{A}) \stackrel{df}{=} \neg(\lceil \mathsf{ph}_{\mathcal{A}} \in cl^{-1}(c) \rceil^{<d(c)}; \lceil \mathsf{ph}_{\mathcal{A}} \notin cl^{-1}(c) \rceil)$ beschrieben werden.

Die Semantik eines Phasenautomaten ist die Menge aller Paare $(\mathcal{I}, \mathcal{V})$, die die zugehörige S-Formel erfüllen. Zwei Phasenautomaten mit gleicher Semantik heißen *äquivalent*.

$$[\![\mathcal{A}]\!] \stackrel{df}{=} [\![F(\mathcal{A})]\!]$$

3.1.1 Netzwerke von Phasenautomaten (Nopas)

Die Parallelkomposition von Phasenautomaten $\mathcal{A}_1, \ldots, \mathcal{A}_n$ wird mit

$$\mathcal{N} = \mathcal{A}_1 \| \ldots \| \mathcal{A}_n$$

bezeichnet und Netzwerk von Phasenautomaten (Network of Phase Automata, kurz Nopa) genannt.

Definition 3.1.3 (Semantik von Nopas). Sei $\mathcal{N} = \mathcal{A}_1 \| \ldots \| \mathcal{A}_n$ ein Netzwerk von Phasenautomaten. Dann wird die Definition von D und F wie folgt erweitert:

$$\begin{aligned} D(\mathcal{A}_1 \| \ldots \| \mathcal{A}_n) &\overset{df}{=} \bigwedge_{i=1}^{n} D(\mathcal{A}_i) \\ F(\mathcal{A}_1 \| \ldots \| \mathcal{A}_n) &\overset{df}{=} \bigwedge_{i=1}^{n} F(\mathcal{A}_i). \end{aligned}$$

Die Semantik ist dann entsprechend definiert.

$$[\![\mathcal{N}]\!] \overset{df}{=} [\![F(\mathcal{N})]\!]$$

In [Tap01] wird das kartesische Produkt für Phasenautomaten definiert und gezeigt, dass der Produktautomat die gleiche Semantik wie die parallele Komposition besitzt.

3.1.2 Lebendigkeitseigenschaften

Mit der angegebenen Semantik von Phasenautomaten können auch Lebendigkeitseigenschaften formalisiert und überprüft werden. Dabei wird ein Uhrintervall $(0, \infty)$ verwendet, das eine Abkürzung für ein Intervall $(0, x]$ darstellt, wobei x eine sonst nicht verwendete Variable ist.

3.1.1 Beispiel (Lebendigkeitseigenschaften)

Der Phasenautomat in Abbildung 3.1 beschreibt die Eigenschaft „irgendwann gilt immer a“.

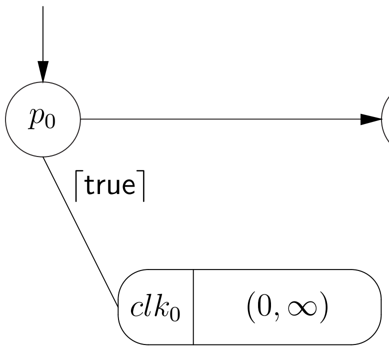


Abbildung 3.1: Phasenautomat zu „irgendwann gilt nur noch a“

3.2 Oder-Konstruktion

Zu zwei Phasenautomaten $\mathcal{A}_1$ und $\mathcal{A}_2$ kann ein Phasenautomat $\mathcal{A}_1 \vee \mathcal{A}_2$ konstruiert werden, der alle Paare $(\mathcal{I}, \mathcal{V})$ akzeptiert, die $\mathcal{A}_1$ oder $\mathcal{A}_2$ akzeptiert. Diese Konstruktion wird in Abschnitt 3.3.3 für die Konstruktion von Phasenautomaten zu regulären Ausdrücken mit Zeit und für das Model-Checking von Fehlerbäumen in Abschnitt 4.4 benötig.

Satz 3.2.1. Seien $\mathcal{A}_1 = (P_1, E_1, C_1, cl_1, s_1, d_1, P_{01})$ und $\mathcal{A}_2 = (P_2, E_2, C_2, cl_2, s_2, d_2, P_{02})$ mit $P_1 \cap P_2 = \emptyset, C_1 \cap C_2 = \emptyset$ zwei Phasenautomaten. Dann sei $\mathcal{A}_1 \vee \mathcal{A}_2$ definiert durch

$$\mathcal{A}_1 \vee \mathcal{A}_2 = (P_1 \cup P_2, E_1 \cup E_2, C_1 \cup C_2, cl, s, d, P_{01} \cup P_{02})$$

mit

$$cl(p) = \begin{cases} cl_1(p) & \text{falls}\, p \in P_1 \\ cl_2(p) & \text{falls}\, p \in P_2 \end{cases} \tag{3.2.3.1}$$

$$s(p) = \begin{cases} s_1(p) & \text{falls}\, p \in P_1 \\ s_2(p) & \text{falls}\, p \in P_2 \end{cases} \tag{3.2.3.2}$$

$$d(c) = \begin{cases} d_1(c) & \text{falls}\, c \in C_1 \\ d_2(c) & \text{falls}\, c \in C_2. \end{cases} \tag{3.2.3.3}$$

Damit gilt

$$\models F(\mathcal{A}_1 \vee \mathcal{A}_2) \iff \big(F(\mathcal{A}_1) \vee F(\mathcal{A}_2)\big) \tag{3.2.3.4}$$

Beweis Seien $\mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2}, \mathsf{ph}_{\mathcal{A}_1}, \mathsf{ph}_{\mathcal{A}_2}$ die Observablen, die die Phasen der Phasenautomaten beschreiben. Wegen

$$\begin{aligned} \operatorname{ran}(\mathsf{ph}_{\mathcal{A}_1}) &= P_1 \\ \operatorname{ran}(\mathsf{ph}_{\mathcal{A}_2}) &= P_2 \\ \operatorname{ran}(\mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2}) &= P_1 \cup P_2 \end{aligned} \tag{3.2.3.5}$$

können die Belegungen dieser Observablen so gewählt werden, dass auch die folgende Bedingung erfüllt ist:

$$\mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2}(t) = \mathsf{ph}_{\mathcal{A}_1}(t) \vee \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2}(t) = \mathsf{ph}_{\mathcal{A}_2}(t) \quad \text{für alle } t \in \mathbf{time} \tag{3.2.3.6}$$

Für den Beweis kann dann die Bedingung

$$\models_{DC} D(\mathcal{A}_1 \vee \mathcal{A}_2) \iff \big(D(\mathcal{A}_1) \vee D(\mathcal{A}_2)\big) \tag{3.2.3.7}$$

gezeigt werden. Dabei wird die Bedingung für jede Teilformel einzeln gezeigt.

$$
\begin{aligned}
& & & \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} D_{init}(\mathcal{A}_1 \vee \mathcal{A}_2) \\
\{\text{Def.}\} & & \iff & \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \neg(\lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} \notin (P_{01} \cup P_{02}) \rceil ; \mathsf{true}) \\
\{\text{Def.} \cup\} & & \iff & \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \neg(\lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} \notin P_{01} \wedge \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} \notin P_{02} \rceil ; \mathsf{true}) \\
\{(3.2.3.7)\} & & \iff & \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \neg(\lceil \neg(\mathsf{ph}_{\mathcal{A}_1} \in P_{01} \vee \mathsf{ph}_{\mathcal{A}_2} \in P_{01}) \\
& & & \qquad \wedge \neg(\mathsf{ph}_{\mathcal{A}_1} \in P_{02} \vee \mathsf{ph}_{\mathcal{A}_2} \in P_{02}) \rceil ; \mathsf{true}) \\
\{(3.2.3.5)\} & & \iff & \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \neg(\lceil \neg(\mathsf{ph}_{\mathcal{A}_1} \in P_{01} \vee \mathsf{false}) \\
& & & \qquad \wedge \neg(\mathsf{false} \vee \mathsf{ph}_{\mathcal{A}_2} \in P_{02}) \rceil ; \mathsf{true}) \\
& & \iff & \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \neg(\lceil (\mathsf{ph}_{\mathcal{A}_1} \notin P_{01}) \wedge (\mathsf{ph}_{\mathcal{A}_2} \notin P_{02}) \rceil ; \mathsf{true}) \\
\{\text{DC}\} & & \iff & \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \neg((\lceil \mathsf{ph}_{\mathcal{A}_1} \notin P_{01} \rceil ; \mathsf{true}) \wedge (\lceil \mathsf{ph}_{\mathcal{A}_2} \notin P_{02} \rceil ; \mathsf{true})) \\
\{\text{Logik}\} & & \iff & \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \neg(\lceil \mathsf{ph}_{\mathcal{A}_1} \notin P_{01} \rceil ; \mathsf{true}) \vee \neg(\lceil \mathsf{ph}_{\mathcal{A}_2} \notin P_{02} \rceil ; \mathsf{true}) \\
\{\text{Def.}\} & & \iff & \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} D_{init}(\mathcal{A}_1) \vee D_{init}(\mathcal{A}_2)
\end{aligned}
$$

Sei $\widetilde{\mathrm{suc}}_{\mathcal{A}_1 \vee \mathcal{A}_2}()$ die erweiterte Nachfolgerelation von $\mathcal{A}_1 \vee \mathcal{A}_2$ und seien $\widetilde{\mathrm{suc}}_{\mathcal{A}_1}()$ und $\widetilde{\mathrm{suc}}_{\mathcal{A}_2}()$ die erweiterten Nachfolgerelationen von $\mathcal{A}_1$ und $\mathcal{A}_2$. Da die Zustandsmengen bei der Konstruktion als disjunkt vorrausgesetzt werden, gilt nach Konstruktion

$$\widetilde{\mathrm{suc}}_{\mathcal{A}_1 \vee \mathcal{A}_2}(p) = \widetilde{\mathrm{suc}}_{\mathcal{A}_1}(p) \text{ für alle } p \in P_1 \tag{3.2.3.8}$$

$$\widetilde{\mathrm{suc}}_{\mathcal{A}_1 \vee \mathcal{A}_2}(p) = \widetilde{\mathrm{suc}}_{\mathcal{A}_2}(p) \text{ für alle } p \in P_2 \tag{3.2.3.9}$$

und damit auch

$$
\begin{aligned}
& & & \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \bigwedge_{p \in P_1 \cup P_2} D^p_{suc}(\mathcal{A}_1 \vee \mathcal{A}_2) \\
& & \iff & \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \bigwedge_{p \in P_1} D^p_{suc}(\mathcal{A}_1 \vee \mathcal{A}_2) \wedge \\
& & & \qquad \bigwedge_{p \in P_2} D^p_{suc}(\mathcal{A}_1 \vee \mathcal{A}_2) \\
\{\text{Def.}\} & & \iff & \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \bigwedge_{p \in P_1} \Big(\neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} = p \rceil ; \\
& & & \qquad \lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} \notin \widetilde{\mathrm{suc}}_{\mathcal{A}_1 \vee \mathcal{A}_2}(p) \rceil) \Big) \wedge \\
& & & \qquad \bigwedge_{p \in P_2} \Big(\neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} = p \rceil ; \\
& & & \qquad \lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} \notin \widetilde{\mathrm{suc}}_{\mathcal{A}_1 \vee \mathcal{A}_2}(p) \rceil) \Big) \\
\{(3.2.3.8), (3.2.3.9)\} & & \iff & \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \bigwedge_{p \in P_1} \Big(\neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} = p \rceil ;
\end{aligned}
$$

$$
\begin{aligned}
& && \lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} \notin \widetilde{\mathrm{suc}}_{\mathcal{A}_1}(p) \rceil) \Big) \wedge \\
& && \bigwedge_{p \in P_2} \Big(\neg \Diamond (\lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} = p \rceil ; \\
& && \lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} \notin \widetilde{\mathrm{suc}}_{\mathcal{A}_2}(p) \rceil) \Big) \\
\{(3.2.3.7)\} \quad & \iff \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} && \bigwedge_{p \in P_1} \Big(\neg \Diamond (\lceil \mathsf{ph}_{\mathcal{A}_1} = p \vee \mathsf{ph}_{\mathcal{A}_2} = p \rceil ; \\
& && \lceil \mathsf{ph}_{\mathcal{A}_1} \notin \widetilde{\mathrm{suc}}_{\mathcal{A}_1}(p) \wedge \mathsf{ph}_{\mathcal{A}_2} \notin \widetilde{\mathrm{suc}}_{\mathcal{A}_1}(p) \rceil) \Big) \wedge \\
& && \bigwedge_{p \in P_2} \Big(\neg \Diamond (\lceil \mathsf{ph}_{\mathcal{A}_1} = p \vee \mathsf{ph}_{\mathcal{A}_2} = p \rceil ; \\
& && \lceil \mathsf{ph}_{\mathcal{A}_1} \notin \widetilde{\mathrm{suc}}_{\mathcal{A}_2}(p) \wedge \mathsf{ph}_{\mathcal{A}_2} \notin \widetilde{\mathrm{suc}}_{\mathcal{A}_2}(p) \rceil) \Big) \\
\{(3.2.3.5)\} \quad & \iff \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} && \bigwedge_{p \in P_1} \Big(\neg \Diamond (\lceil \mathsf{ph}_{\mathcal{A}_1} = p \vee \mathsf{false} \rceil ; \\
& && \lceil \mathsf{ph}_{\mathcal{A}_1} \notin \widetilde{\mathrm{suc}}_{\mathcal{A}_1}(p) \wedge \mathsf{true} \rceil) \Big) \wedge \\
& && \bigwedge_{p \in P_2} \Big(\neg \Diamond (\lceil \mathsf{false} \vee \mathsf{ph}_{\mathcal{A}_2} = p \rceil ; \\
& && \lceil \mathsf{true} \wedge \mathsf{ph}_{\mathcal{A}_2} \notin \widetilde{\mathrm{suc}}_{\mathcal{A}_2}(p) \rceil) \Big) \\
& \iff \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} && \bigwedge_{p \in P_1} \Big(\neg \Diamond (\lceil \mathsf{ph}_{\mathcal{A}_1} = p \rceil ; \\
& && \lceil \mathsf{ph}_{\mathcal{A}_1} \notin \widetilde{\mathrm{suc}}_{\mathcal{A}_1}(p) \rceil) \Big) \wedge \\
& && \bigwedge_{p \in P_2} \Big(\neg \Diamond (\lceil \mathsf{ph}_{\mathcal{A}_2} = p \rceil ; \lceil \mathsf{ph}_{\mathcal{A}_2} \notin \widetilde{\mathrm{suc}}_{\mathcal{A}_2}(p) \rceil) \Big) \\
& \iff \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} && \bigwedge_{p \in P_1} D^p_{suc}(\mathcal{A}_1) \wedge \bigwedge_{p \in P_2} D^p_{suc}(\mathcal{A}_2)
\end{aligned}
$$

Für die anderen Teilformeln gelten analoge Beziehungen wie für D^p_{suc}.

$$
\mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \quad \bigwedge_{p \in P_1 \cup P_2} D^p_{out}(\mathcal{A}_1 \vee \mathcal{A}_2)
$$

$$
\begin{aligned}
&\iff \mathcal{I}, \mathcal{V},\ [b,e]\ \models_{DC} && \bigwedge_{p \in P_1} D^p_{out}(\mathcal{A}_1 \vee \mathcal{A}_2) \wedge \\
& && \bigwedge_{p \in P_2} D^p_{out}(\mathcal{A}_1 \vee \mathcal{A}_2) \\
\{\text{Def.}\} \quad &\iff \mathcal{I}, \mathcal{V},\ [b,e]\ \models_{DC} && \bigwedge_{p \in P_1} \Big(\neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} = p \wedge \neg s(p)\rceil\Big) \wedge \\
& && \bigwedge_{p \in P_2} \Big(\neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} = p \wedge \neg s(p)\rceil\Big) \\
\{(3.2.3.2),\ (3.2.3.7\ )\} \quad &\iff \mathcal{I}, \mathcal{V},\ [b,e]\ \models_{DC} && \bigwedge_{p \in P_1} \Big(\neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}_1} = p \wedge \neg s_1(p)\rceil\Big) \wedge \\
& && \bigwedge_{p \in P_2} \Big(\neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}_2} = p \wedge \neg s_2(p)\rceil\Big) \\
\{\text{Def.}\} \quad &\iff \mathcal{I}, \mathcal{V},\ [b,e]\ \models_{DC} && \bigwedge_{p \in P_1} D^p_{out}(\mathcal{A}_1) \wedge \\
& && \bigwedge_{p \in P_2} D^p_{out}(\mathcal{A}_2)
\end{aligned}
$$

$$
\begin{aligned}
& \mathcal{I}, \mathcal{V},\ [b,e]\ \models_{DC} && \bigwedge_{c \in C_1 \cup C_2} D^c_{pr}(\mathcal{A}_1 \vee \mathcal{A}_2) \\
&\iff \mathcal{I}, \mathcal{V},\ [b,e]\ \models_{DC} && \bigwedge_{c \in C_1} D^p_{pr}(\mathcal{A}_1 \vee \mathcal{A}_2) \wedge \\
& && \bigwedge_{c \in C_2} D^c_{pr}(\mathcal{A}_1 \vee \mathcal{A}_2) \\
\{\text{Def.}\} \quad &\iff \mathcal{I}, \mathcal{V},\ [b,e]\ \models_{DC} && \bigwedge_{c \in C_1} \Big(\neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} \in cl^{-1}\rceil^{>d(c)}\Big) \wedge \\
& && \bigwedge_{c \in C_2} \Big(\neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} \in cl^{-1}\rceil^{>d(c)}\Big) \\
\{(3.2.3.1),\ (3.2.3.3)\} \quad &\iff \mathcal{I}, \mathcal{V},\ [b,e]\ \models_{DC} && \bigwedge_{c \in C_1} \Big(\neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} \in cl_1^{-1}\rceil^{>d_1(c)}\Big) \wedge \\
& && \bigwedge_{c \in C_2} \Big(\neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}_1 \vee \mathcal{A}_2} \in cl_2^{-1}\rceil^{>d_2(c)}\Big)
\end{aligned}
$$

$$
\begin{aligned}
\{(3.2.3.7), (3.2.3.1)\} \quad &\Longleftrightarrow \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \quad \bigwedge_{c \in C_1} \left(\neg \Diamond \lceil \mathsf{ph}_{\mathcal{A}_1} \in cl_1^{-1} \rceil^{> d_1(c)} \right) \wedge \\
&\qquad\qquad\qquad\qquad\quad \bigwedge_{c \in C_2} \left(\neg \Diamond \lceil \mathsf{ph}_{\mathcal{A}_2} \in cl_2^{-1} \rceil^{> d_2(c)} \right) \\
\{\text{Def.}\} \quad &\Longleftrightarrow \mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \quad \bigwedge_{p \in P_1} D^c_{pr}(\mathcal{A}_1) \wedge \\
&\qquad\qquad\qquad\qquad\quad \bigwedge_{p \in P_2} D^c_{pr}(\mathcal{A}_2)
\end{aligned}
$$

Die Beziehung für D^c_{st} und D_{ist} folgt analog. Es gilt dann:

$$
\begin{aligned}
&\mathcal{I}, \mathcal{V} \models_{DC} D(\mathcal{A}_1 \vee \mathcal{A}_2) \\
\{\text{Def.}\} \iff \quad &\mathcal{I}, \mathcal{V}, [b,e] \models_{DC} D_{init}(\mathcal{A}_1 \vee \mathcal{A}_2) \wedge \\
&\bigwedge_{p \in P_1 \cup P_2} \left(D^p_{suc}(\mathcal{A}_1 \vee \mathcal{A}_2) \wedge D^p_{out}(\mathcal{A}_1 \vee \mathcal{A}_2) \right) \wedge \\
&\bigwedge_{c \in C_1 \cup C_2} \Big(D^c_{pr}(\mathcal{A}_1 \vee \mathcal{A}_2) \wedge \\
&\qquad D^c_{st}(\mathcal{A}_1 \vee \mathcal{A}_2) \wedge D^c_{ist}(\mathcal{A}_1 \vee \mathcal{A}_2) \Big) \\
\left\{ \begin{array}{r} \text{Bew. für} \\ \text{Teilformeln} \end{array} \right\} \iff \quad &\mathcal{I}, \mathcal{V}, [b,e] \models_{DC} \left(D_{init}(\mathcal{A}_1) \vee D_{init} \mathcal{A}_2 \right) \wedge \\
&\bigwedge_{p \in P_1} \left(D^p_{suc}(\mathcal{A}_1) \wedge D^p_{out}(\mathcal{A}_1) \right) \wedge \\
&\bigwedge_{c \in C_1} \left(D^c_{pr}(\mathcal{A}_1) \wedge D^c_{st}(\mathcal{A}_1) \wedge D^c_{ist}(\mathcal{A}_1) \right) \wedge \\
&\bigwedge_{p \in P_2} \left(D^p_{suc}(\mathcal{A}_2) \wedge D^p_{out}(\mathcal{A}_2) \right) \wedge \\
&\bigwedge_{c \in C_2} \left(D^c_{pr}(\mathcal{A}_2) \wedge D^c_{st}(\mathcal{A}_2) \wedge D^c_{ist}(\mathcal{A}_2) \right) \\
\iff \quad &\mathcal{I}, \mathcal{V}, [b,e] \models_{DC} D(\mathcal{A}_1) \vee D(\mathcal{A}_2)
\end{aligned}
$$

$\square$

3.3 Phasenautomaten und Realzeitautomaten

In [Mru00] wird gezeigt, wie Phasenautomaten in äquivalente Realzeitautomaten umgewandelt werden können. Die umgekehrte Konstruktion kann über reguläre Ausdrücke mit Zeit erfolgen. In [ACM97] werden reguläre Ausdrücke so erweitert, dass damit auch Realzeitsysteme beschrieben werden können und es wird gezeigt, dass zu jedem Realzeitautomaten auch ein regulärer Ausdruck mit Zeit konstruiert werden kann, der die gleiche Menge von Abläufen beschreibt und umgekehrt. Es genügt deshalb eine Konstruktion anzugeben, die aus einem regulären Ausdruck mit Zeit einen äquivalenten Phasenautomaten konstruiert.

3.3.1 Reguläre Ausdrücke mit Zeit

In [ACM97] wird das Verhalten eines Realzeitsystems durch eine abschnittweise konstante Funktion mit endlicher Variablität der Form

$$\xi : (0, k] \to \Sigma, k \in \mathbb{R}_+^* \cup \{\infty\}$$

beschrieben, die (endliches) Signal genannt wird. Dabei ist Σ ein endliches Alphabet, dass die Zustände des Systems repräsentiert. Das Intervall $(0, k]$ modelliert den betrachteten Zeitabschnitt. Die Zahl k heißt die Länge des Ablaufs und wird wie bei formalen Sprachen mit $|\xi|$ bezeichnet. Dieser Ansatz entspricht der Modellierung durch Interpretationen für eine Observable und passt auch zur Modellierung durch Realzeitsprachen, wie sie in [AD94] verwendet wird. Dort wird das Verhalten eines Systems durch eine Folge von Paaren (σ, τ) von Ereignissen σ und Zeitpunkten τ, zu denen das Ereignis eintritt, beschrieben. Dabei wird im Gegensatz zur Definition in [Alu99] verlangt, dass zwischen zwei Ereignissen Zeit vergeht.
Die regulären Ausdrücke mit Zeit werden zuerst für endliche Abläufe ($k \in \mathbb{R}_+^*$) definiert und anschließend auf unendliche Abläufe erweitert. Zur besseren Lesbarkeit werden Signale in der Form

$$\xi = a_1^{r_1} a_2^{r_2} \dots a_n^{r_n}$$

mit $a_i \in \Sigma$, $r_i \in \mathbb{R}$, $a_i \neq a_{i+1}$ und $\sum r_i = k$ beschrieben werden. Dabei gibt r_i an, wie lange das Signal konstant den Wert a_i annimmt.
Die Menge aller endlichen Signale für ein Alphabet Σ wird mit $\mathcal{S}(\Sigma)$ bezeichnet. Eine abschnittweise konstante Funktion

$$\xi : \mathbb{R}_{\geq 0} \to \Sigma$$

heißt unendliches Signal und die Menge aller unendlichen Signale wird mit $\mathcal{S}_\omega(\Sigma)$ bezeichnet. Wie bei formalen Sprachen, kann die Konkatenation von zwei Signalen definiert werden:

Definition 3.3.1 (Konkatenation von Signalen). Seien ξ_1 und ξ_2 zwei Signale. Dann ist die Konkatenation $\xi_1 \circ \xi_2$ definiert durch

$$\xi_1 \circ \xi_2(t) = \begin{cases} \xi_1(t) & \text{falls } t \in (0, |\xi_1|] \\ \xi_2(t - |\xi_1|) & \text{falls } t \in (|\xi_1|, |\xi_1| + |\xi_2|] \end{cases} .$$

Die Iteration kann ebenfalls wie bei formalen Sprachen definiert werden.

Definition 3.3.2 (Iteration von Konkatenationen). Sie ξ ein Signal. Dann ist die i-fache Iteration definiert durch

$$\xi^i \stackrel{df}{=} \underbrace{\xi \circ \ldots \circ \xi}_{i\text{-mal}}.$$

Diese Definitionen werden auf Mengen von Signalen verallgemeinert.

Definition 3.3.3 (Konkatenation und Iteration für Mengen von Signalen). Seien M_1 und M_2 Mengen von endlichen Signalen. Dann ist

- $$M_1 \circ M_2 \stackrel{df}{=} \{\xi_1 \circ \xi_2 | \xi_1 \in M_1 \,\text{und}\, \xi_2 \in M_2\}$$
- $$(M_1)^i \stackrel{df}{=} \{\xi^i | \xi \in M_1\}$$

Damit kann die Syntax und Semantik von regulären Ausdrücken mit Zeit definiert werden.

Definition 3.3.4 (Syntax von regulären Ausdrücken mit Zeit). Reguläre Ausdrücke mit Zeit werden mit φ bezeichnet und durch die Grammatik

$$\varphi ::= a \,|\, \varphi_1 \wedge \varphi_2 \,|\, \varphi_1 \vee \varphi_2 \,|\, \varphi_1 \cdot \varphi_2 \,|\, \varphi_1^* \,|\, \langle\varphi\rangle_I$$

erzeugt, wobei $a \in \Sigma$ und I ein Intervall mit ganzzahligen Grenzen ist. Die Menge aller regulären Ausdrücke mit Zeit über einem Alphabet Σ wird mit $\mathcal{E}(\Sigma)$ bezeichnet.

Definition 3.3.5 (Semantik von regulären Ausdrücken mit Zeit). Die Semantik eines regulären Ausdrucks mit Zeit ist eine Funktion

$$[\![\cdot]\!] : \mathcal{E}(\Sigma) \to 2^{\mathcal{S}(\Sigma)},$$

die induktiv definiert ist.

- $$[\![a]\!] = \{a^r : r \in \textbf{time}\}$$
- $$[\![\varphi_1 \wedge \varphi_2]\!] = [\![\varphi_1]\!] \cap [\![\varphi_2]\!]$$
- $$[\![\varphi_1 \vee \varphi_2]\!] = [\![\varphi_1]\!] \cup [\![\varphi_2]\!]$$
- $$[\![\varphi_1 \cdot \varphi_2]\!] = [\![\varphi_1]\!] \circ [\![\varphi_2]\!]$$
- $$[\![\varphi_1^*]\!] = \bigcup_{i=0}^{\infty} [\![\varphi_1]\!]^i$$
- $$[\![\langle\varphi\rangle_I]\!] = [\![\varphi]\!] \cap \{\xi : |\xi| \in I\}$$

3.3.2 ω-reguläre Ausdrücke mit Zeit

Die Definition der regulären Ausdrücke mit Zeit wird in [ACM97] auch auf unendliche Abläufe erweitert.

Definition 3.3.6 (Syntax von ω-regulären Ausdrücken mit Zeit). Die ω-regulären Ausdrücke mit Zeit werden mit ψ bezeichnet und durch die Grammatik

$$\psi ::= \psi_1 \wedge \psi_2 \,|\, \psi_1 \vee \psi_2 \,|\, \varphi \cdot \psi \,|\, \varphi^\omega$$

erzeugt, wobei $\varphi \in \mathcal{E}(\Sigma)$ ein regulärer Ausdruck mit Zeit. Die Menge aller ω-regulären Ausdrücke mit Zeit über einem Alphabet Σ wird mit $\mathcal{E}_\omega(\Sigma)$ bezeichnet.

Definition 3.3.7 (Semantik von ω-regulären Ausdrücken mit Zeit). Die Semantik eines ω-regulären Ausdrucks mit Zeit ist eine Funktion

$$[\![\cdot]\!] : \mathcal{E}_\omega(\Sigma) \to 2^{\mathcal{S}_\omega(\Sigma)},$$

die induktiv definiert ist.

- $$[\![\psi_1 \wedge \psi_2]\!] = [\![\psi_1]\!] \cap [\![\psi_2]\!]$$
- $$[\![\psi_1 \vee \psi_2]\!] = [\![\psi_1]\!] \cup [\![\psi_2]\!]$$
- $$[\![\varphi \cdot \psi]\!] = [\![\varphi]\!] \circ [\![\psi]\!]$$
- $$[\![\varphi^\omega]\!] = \bigcirc_{i=1}^{\infty}[\![\varphi]\!]$$

Hierbei bezeichnet $\bigcirc_{i=1}^{\infty} \xi_i$ die Konkatenation von unendlich vielen jeweils endlichen Signalen. Dadurch erhält man ein unendliches Signal unter der Annahme, dass die Summe der Längen $\sum_{i=1}^{\infty} |\xi_i|$ nicht endlich ist.

3.3.3 Konstruktion eines Phasenautomaten zu einem regulären Ausdruck mit Zeit

Da reguläre Ausdrücke mit Zeit *endliche* Abläufe beschreiben, Phasenautomaten aber für unendliche Abläufe definiert sind, wird ein endlicher Ablauf ξ zu einem unendlichen Ablauf ξ_ω erweitert, indem nach Ablauf von ξ ein spezieller neuer Funktionswert $\perp$ angenommen wird und es werden zu regulären Ausdrücken mit Zeit Phasenautomaten konstruiert, die die erweiterten Funktionen akzeptieren. Die Konstruktion erfolgt induktiv über den Aufbau der regulären Ausdrücke. Außerdem müssen für die Konstruktion von Konkatenation und *-Abschluss Zustände für den Konstruktionsprozess als Endzustände markiert werden. Diese Endzustände sollen angenommen sein, wenn der endliche Ablauf zuende ist. Danach soll der Phasenautomat in den speziellen Zustand $p_\perp$ zu einer $\lceil\perp\rceil$-Phase übergehen.
Seien φ und φ' zwei reguläre Ausdrücke mit den zugehörigen Phasenautomaten

$$\mathcal{A}_\varphi = (P, E, C, cl, s, d, P_0)$$

und

$$\mathcal{A}_{\varphi'} = (P', E', C', cl', s', d', P'_0)$$

und den Endzustandsmengen F_φ und $F_{\varphi'}$ mit $P \cap P' = \{p_\perp\}$, $C \cap C' = \emptyset$.

Automat für a Der Automat zu einem endlichen Ablauf, der die ganze Zeit den Wert a annimmt, wird in Abbildung 3.2 dargestellt und besitzt eine Phase für die endliche a-Phase und eine für die darauf folgende $\perp$-Phase, die anzeigt, dass das Ausgangssignal zu Ende ist.
$\mathcal{A}_a = (\{p0, p_\perp\}, \{(p0, p_\perp\}, \{p0^c\}, cl_a, s_a, d_a, , \{p0\}),\ F_a = p0$ mit

$$cl_a(p0) = p0^c, \quad s_a(p0) = \lceil a \rceil, \quad s_a(p_\perp) = \lceil \perp \rceil, \quad d_a(p0^c) = [0, \infty)$$

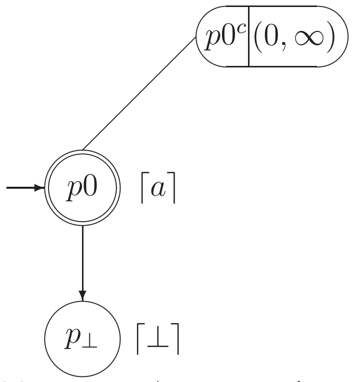


Abbildung 3.2: Automat $\mathcal{A}_a$

Automat für $\varphi \vee \varphi'$ Der Automat zu $\varphi \vee \varphi'$ ist die Vereinigung der beiden Automaten wie in der Oder-Konstruktion in Satz 3.2.1 angegeben. Die Endzustandsmenge ist die Vereinigung der beiden Endzustandsmengen.

Automat für $\varphi' \wedge \varphi$ Die Konstruktion für den Durchschnitt ist die in [Tap01] angegebene Parallelkomposition $\mathcal{A}_{\varphi' \wedge \varphi} = \mathcal{A}_\varphi \| \mathcal{A}_{\varphi'}$, $F = F_\varphi \times F_{\varphi'}$. Da bei der Konstruktion das kartesische Produkt der beiden Zustandsmengen gebildet wird, entstehen bei der Konstruktion auch Zustände der Form $(p_\perp, p')$ und $(p', p_\perp)$. Die zugeordneten Zustandsausdrücke widersprechen sich, falls der von p' nicht äquivalent zu `true` oder $\perp$ ist. Solche Zustände können dann weggelassen werden.

Automat für $\langle\varphi\rangle_I$ Um aus einem Automaten für φ einen Automaten für $\langle\varphi\rangle_I$ zu konstruieren, wird allen Zuständen außer dem Zustand $p_\perp$ eine neue Uhr mit Uhrintervall I zugeordnet. Somit darf der spezielle Wert $\perp$ nicht zu früh oder zu spät angenommen werden, der endliche Ablauf muss also eine Länge innerhalb von I besitzen.
Die Konstruktion ist in Abbildung 3.3 skizziert. Dabei bezeichnet $\mathcal{A}_\varphi \setminus \{p_\perp\}$ den Automaten $\mathcal{A}_\varphi$ aus dem die Phase $p_\perp$ entfernt wurde. Genau wird der Automat definiert durch $\mathcal{A}_{\langle\varphi\rangle_I} = (P, E, C \cup \{clk_I\}, cl_{\langle\varphi\rangle_I}, s, d_{\langle\varphi\rangle_I}, P_0)$. $F = F_\varphi$ und

$$cl_{\langle\varphi\rangle_I}(p) = \begin{cases} cl(p) \cup \{clk_I\} & \text{falls } p \neq p_\perp \\ \emptyset & \text{falls } p = p_\perp \end{cases} \qquad d_{\langle\phi\rangle_I}(clk) = \begin{cases} d(clk) & \text{falls } clk \in C \\ I & \text{falls } clk = clk_I \end{cases}$$

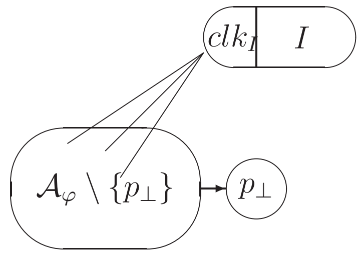


Abbildung 3.3: Automat $\mathcal{A}_{\langle\varphi\rangle_I}$

Automat für $\varphi \cdot \varphi'$ Für die Konstruktion des Phasenautomaten für die Konkatenation $\varphi \cdot \varphi'$ werden die Automaten für φ und φ' hintereinander geschaltet (Abbildung 3.4). Es werden die Endzustände des Automaten für φ mit den Anfangszuständen des Automaten für φ' verbunden und alle Kanten zu $p_\perp$, die von Zuständen aus $\mathcal{A}_\varphi$ ausgehen, entfernt.

$$\mathcal{A}_{\varphi\cdot\varphi'} = (P \cup P', E_{\varphi\cdot\varphi'}, C \cup C', cl_{\varphi'\cdot\varphi'}, s_{\varphi'\cdot\varphi'}, d_{\varphi'\cdot\varphi'}, P_0), F = F_{\varphi'}$$

mit

$$E_{\varphi\cdot\varphi'} = (E \cup E' \cup F \times P'_0) \setminus (E \times \{p_\perp\})$$

$$cl_{\varphi\cdot\varphi'}(p) = \begin{cases} cl(p) & \text{falls } p \in P \\ cl'(p) & \text{falls } p \in P' \end{cases} \qquad s_{\varphi\cdot\varphi'}(p) = \begin{cases} s(p) & \text{falls } p \in P \\ s(p) & \text{falls } p \in P' \end{cases}$$

$$d_{\varphi\cdot\varphi'}(clk) = \begin{cases} d(clk) & \text{falls } clk \in C \\ d'(clk) & \text{falls } clk \in C' \end{cases}$$

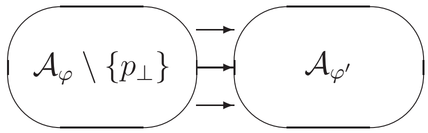


Abbildung 3.4: Automat $\mathcal{A}_{\varphi \cdot \varphi'}$

Automat für φ^* Da die Uhren in Phasenautomaten nicht explizit zurückgesetzt werden können, müssen für die Konstruktion eines Phasenautomaten für den *-Abschluss zwei isomorphe Phasenautomaten hintereinander geschaltet werden. Zu dem Automaten $\mathcal{A}_\varphi$ sei $\mathcal{A}^{(2)} = (P^{(2)}, E^{(2)}, C^{(2)}, cl^{(2)}, s^{(2)}, d^{(2)}, P_0^{(2)})$ ein isomorpher Automat in dem alle Zustände bis auf $p_\perp$ und alle Uhren umbenannt sind und $F^{(2)}$ die entsprechende Endzustandsmenge des neuen Automaten. Die beiden Automatenteile besitzen den Zustand $p_\perp$ gemeinsam und alle Zustände außer $p_\perp$ besitzen eine gemeinsame Uhr, die sicherstellt, dass der Zustand $p_\perp$ schließlich erreicht wird, also der Ablauf endlich ist. Es ist dann

$$\mathcal{A}_{\varphi^*} = (P \cup P^{(2)}, E \cup E^{(2)} \cup F_\varphi \times P_0^{(2)} \cup F^{(2)} \times P_0, C \cup C^{(2)} \cup \{clk^*\}, cl_{\varphi^*}, s_{\varphi^*}, d_{\varphi^*}, P_0)$$

mit Endzustandsmenge $F = F\varphi \cup F^{(2)}$ ein Automat für φ^*, wobei

$$cl_{\varphi^*}(p) = \begin{cases} cl(p) \cup \{clk^*\} & \text{falls } p \in P \setminus \{p_\perp\} \\ cl^{(2)}(p) \cup \{clk^*\} & \text{falls } p \in P^{(2)} \setminus \{p_\perp\} \\ \emptyset & \text{falls } p = p_\perp \end{cases}$$

$$s_{\varphi^*}(p) = \begin{cases} s(p) & \text{falls } p \in P \\ s^{(2)}(p) & \text{falls } p \in P^{(2)} \end{cases}$$

$$d_{\varphi^*}(clk) = \begin{cases} d(clk) & \text{falls } clk \in C \\ [0, \infty) & \text{falls } clk = clk^* \end{cases}$$

Die Korrektheit der Konstruktion wird durch folgenden Satz beschrieben.

Satz 3.3.8 (Korrektheit für endliche Abläufe). Sei ξ ein endliches Signal wie unter 3.3.1 definiert, X eine Observable und $\mathcal{I}$ eine Interpretation mit

$$\mathcal{I}(X)(t) = \begin{cases} \xi(t) & \text{falls } t \in (0, |\xi|] \\ \perp & \text{falls } t > |\xi| \end{cases} .$$

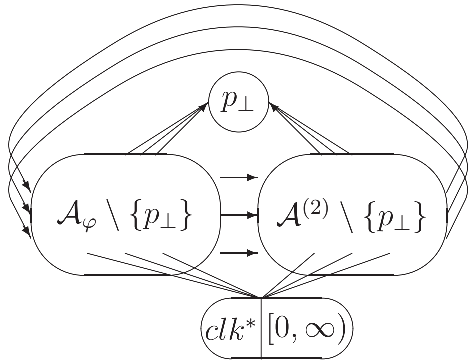


Abbildung 3.5: Automat $\mathcal{A}_{\varphi^*}$

Dann gilt $\xi \in [\![\varphi]\!]$ gdw. eine Belegung $\mathcal{V}$ existiert, so dass

$$\mathcal{I}, \mathcal{V} \models \exists\, \mathsf{ph}_{\mathcal{A}}[D(\mathcal{A}\varphi)]$$

Bemerkung 3.3.1. Aus der Konstruktion folgt, dass nur von Endzuständen aus der Zustand $p_\perp$ erreicht werden kann. Außerdem muss der Zustand $p_\perp$ irgendwann angenommen werden, da alle anderen Zustände nicht unendlich lange angenommen werden dürfen und auch Schleifen (*-Abschluss-Konstruktion) nicht unendlich oft durchlaufen werden dürfen.

Beweis über Induktion nach Aufbau der regulären Ausdrücke mit Zeit

1. Fall $\varphi = a$

„⇒“-Richtung Sei $\xi = a^r$, $r \in \mathbb{R}_+$ Wähle $\mathcal{I}'$ zu $\mathcal{I}$ mit $\mathcal{I} =_{\backslash \mathsf{ph}_{\mathcal{A}}} \mathcal{I}'$ und

$$\mathcal{I}(\mathsf{ph}_{\mathcal{A}})(t) = \begin{cases} p_0 & \text{falls } t \in [0,r] \\ p_\perp & \text{sonst.} \end{cases}$$

Ein Intervall $I = (0,\infty)$ in einer Uhrbedingung ist als Abkürzung für ein Intervall $I = (0,x)$ definiert, wobei x eine Variable ist, die sonst nicht benutzt wird. Diese Variable sei durch die Belegung $\mathcal{V}$ belegt durch

$$\mathcal{V}(x) = |\xi|.$$

Dann erfüllt das Paar $(\mathcal{I}', \mathcal{V})$

- die Initialisierungsbedingung D_{init}, da die erste Phase p_0 ein Startzustand ist,
- die Fortschrittsbedingung D^p_{suc} für alle Zustände p, da die einzige Transitition von p_0 nach $p_\perp$ erlaubt ist,

- die Bedingung D^p_{out} für alle Zustände p, da die Zustandsausdrücke jeweils erfüllt sind,
- die Fortschrittsbedingung $D^{p0^c}_{pr} = \neg\Diamond\lceil \mathsf{ph}_\mathcal{A} \in cl^{-1}(p0^c)\rceil^{>x}$ für die einzige Uhr $p0^c$, da der Zustand p_0 nicht länger als $\mathcal{V}(x) = |\xi|$ angenommen wird und
- die Stabilitätsbedingungen, da die einzige Uhrbedingung als untere Schranke den Wert 0 besitzt.

Also gilt

$$\mathcal{I}, \mathcal{V} \models \exists\, \mathsf{ph}_\mathcal{A} : [D(\mathcal{A})]$$

„⇐“-Richtung Gelte $\mathcal{I}, \mathcal{V} \models \exists \mathsf{ph}_\mathcal{A} : [D(\mathcal{A})]$. Dann gibt es ein $\mathcal{I}'$, das bis auf $\mathsf{ph}_\mathcal{A}$ mit $\mathcal{I}$ übereinstimmt, so dass $\mathcal{I}', \mathcal{V} \models_{DC} D(\mathcal{A})$. Damit gilt

- $\mathcal{I}'(\mathsf{ph}_\mathcal{A})$ ist initial p_0, weil D_{init} erfüllt ist.
- $\mathcal{I}'(\mathsf{ph}_\mathcal{A})$ geht wegen der Uhrbedingung $D^{p0^c}_{pr}$ nach höchstens $\mathcal{V}(x)$ Zeiteinheiten, etwa nach $t_\perp$ Zeiteinheiten, nach $p_\perp$ über.

Wegen D_{out} hat $\mathcal{I}$ damit die Form

$$\mathcal{I}(X)(t) = \begin{cases} \xi(t) & \text{falls } t \in (0, t_\perp] \\ \perp & \text{falls } t > \mathcal{V}(x) \end{cases} .$$

und damit $\xi = a^r$ also $\xi \in [\![a]\!]$.

2. Fall $\varphi \vee \varphi'$

Der Beweis folgt direkt aus Satz 3.2.1.

3. Fall $\varphi \wedge \varphi'$

Die Korrektheit der Produktkonstruktion wird in [Tap01] bewiesen.

4. Fall $\langle\varphi\rangle_I$

„⇒“-Richtung Sei $\xi \in [\![\langle\varphi\rangle_I]\!]$. Dann gilt nach Definition

$$\xi \in [\![\varphi]\!] \tag{3.3.3.1}$$
$$|\xi| \in I. \tag{3.3.3.2}$$

Aus (3.3.3.1) folgt mit Induktionsvoraussetzung die Existenz von passenden $\mathcal{I}', \mathcal{V}$ so dass

$$\mathcal{I}', \mathcal{V} \models_{DC} D(\mathcal{A}_\varphi).$$

Mit (3.3.3.2) folgt auch die Gültigkeit der drei Uhrbedingungen

$$\begin{aligned}
\mathcal{I}', \mathcal{V} &\models_{DC} \quad D_{pr}^{clk_I} = \neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}} \neq p_\bot \rceil^{>I}) \\
\mathcal{I}', \mathcal{V} &\models_{DC} \quad D_{st}^{clk_I} = \neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}} = p_\bot \rceil; \lceil \mathsf{ph}_{\mathcal{A}} \neq p_\bot \rceil^{<I}; \lceil \mathsf{ph}_{\mathcal{A}} = p_\bot \rceil) \\
\mathcal{I}', \mathcal{V} &\models_{DC} \quad D_{ist}^{clk_I} = \neg(\lceil \mathsf{ph}_{\mathcal{A}} \neq p_\bot \rceil^{<I}; \lceil \mathsf{ph}_{\mathcal{A}} = p_\bot \rceil)
\end{aligned}$$

und damit wegen $D(\mathcal{A}_{\langle\varphi\rangle_I}) = D(\mathcal{A}_\varphi) \wedge D_{pr}^{clk_I} \wedge D_{st}^{clk_I} \wedge D_{ist}^{clk_I}$ auch

$$\mathcal{I}', \mathcal{V} \models_{DC} D(\mathcal{A}_{\langle\varphi\rangle_I})$$

also auch

$$\mathcal{I}, \mathcal{V} \models \exists\, \mathsf{ph}_{\mathcal{A}} D(\mathcal{A}_{\langle\varphi\rangle_I}).$$

„⇐“-Richtung Gelte $\mathcal{I}, \mathcal{V} \models \exists\, \mathsf{ph}_{\mathcal{A}} D(\mathcal{A}_{\langle\varphi\rangle_I})$ und sei $\mathcal{I}'$ die zu $\mathcal{I}$ passende Interpretation, die $D(\mathcal{A}_{\langle\varphi\rangle_I})$ erfüllt. Weil

$$D(\mathcal{A}_{\langle\varphi\rangle_I}) = D(\mathcal{A}_\varphi) \wedge D_{pr}^{clk_I} \wedge D_{st}^{clk_I} \wedge D_{ist}^{clk_I},$$

gilt auch

$$\mathcal{I}', \mathcal{V} \models_{DC} D(\mathcal{A}_\varphi)$$

und damit nach Induktionsvoraussetzung auch $\xi \in [\![\varphi]\!]$. Wegen der Uhrbedingung der Uhr clk_I gilt auch $|\xi| \in I$ und damit

$$\xi \in [\![\langle\varphi\rangle_I]\!]$$

5. Fall $\varphi \cdot \varphi'$

„⇒“-Richtung Sei $\xi \in [\![\varphi \cdot \varphi']\!]$. Dann gibt es eine Zerlegung von $\xi = \xi_1 \cdot \xi_2$ mit $\xi_1 \in [\![\varphi]\!]$ und $\xi_2 \in [\![\varphi']\!]$. Nach Induktionsvoraussetzung gibt es $\mathcal{I}_1', \mathcal{V}_1, \mathcal{I}_2', \mathcal{V}_2$, so dass

$$\mathcal{I}_1', \mathcal{V}_1 \models_{DC} D(\mathcal{A}_\varphi) \tag{3.3.3.3}$$

$$\mathcal{I}_2', \mathcal{V}_2 \models_{DC} D(\mathcal{A}_\varphi') \tag{3.3.3.4}$$

gilt. Nach Bemerkung 3.3.1 ist der Automat $\mathcal{A}_\varphi$ bevor er in $p_\bot$ übergeht in einem Endzustand aus F. Also gibt es einen Lauf des Phasenautomaten $\mathcal{A}_{\varphi\cdot\varphi'}$, d.h. mit

$$\mathcal{I}'(X)(t) = \begin{cases} \xi_1(t) & \text{falls } t \in [0, |\xi_1|] \\ \xi_2(t - |\xi_1|) & \text{falls } t \in (|\xi_1|, |\xi_1| + |\xi_2|] \\ \bot & \text{sonst} \end{cases}$$

$$\mathcal{I}'(\mathsf{ph}_{\mathcal{A}})(t) = \begin{cases} \mathcal{I}_1'(\mathsf{ph}_{\mathcal{A}})(t) & \text{falls } t \in [0, |\xi_1|] \\ \mathcal{I}_2'(\mathsf{ph}_{\mathcal{A}})(t - |\xi_1|) & \text{falls } t \in (|\xi_1|, |\xi_1| + |\xi_2|] \\ \bot & \text{sonst} \end{cases}$$

und

$$\mathcal{V}(x) = \begin{cases} \mathcal{V}_1(x) & \text{falls } x \text{ Variable von } \mathcal{A}_\varphi \\ \mathcal{V}_2(x) & \text{falls } x \text{ Variable von } \mathcal{A}'_\varphi \end{cases}$$

gilt

$$\mathcal{I}', \mathcal{V} \models_{DC} D(\mathcal{A}_{\varphi\cdot\varphi'}).$$

„$\Leftarrow$"-Richtung Gelte $\mathcal{I}, \mathcal{V} \models \exists \mathsf{ph}_\mathcal{A}\, D(\mathcal{A}_{\varphi\cdot\varphi'})$ und sei $\mathcal{I}'$ die zu $\mathcal{I}$ passende Interpretation mit

$$\mathcal{I}', \mathcal{V} \models_{DC} D(\mathcal{A}_{\varphi\cdot\varphi'}).$$

Nach Bemerkung 3.3.1 wird irgendwann der Zustand $p_\perp$ erreicht, also führt der Lauf des Phasenautomaten durch Zustände aus $\mathcal{A}_{\varphi'}$ und deswegen auch durch einen Endzustand des Teilautomaten $\mathcal{A}_\varphi$. Der gesamte Lauf kann damit in zwei Teile zerlegt werden und die Induktionsvoraussetzung auf die beiden Teilläufe angewendet werden. Damit folgt, dass ξ die Form $\xi = \xi_1 \circ \xi_2$ mit $\xi_1 \in [\![\varphi]\!]$ und $\xi_2 \in [\![\varphi']\!]$ besitzt und somit $\xi \in [\![\varphi \cdot \varphi']\!]$ gilt.

6. Fall φ^*

Dieser Fall wird analog zu Fall 5 bewiesen.

□

3.3.4 Konstruktion eines Phasenautomaten zu einem ω-regulären Ausdruck mit Zeit

Die Konstruktion kann auf ω-reguläre Ausdrücke erweitert werden. Die einzelnen Konstruktionsschritte entsprechen im wesentlichen den Konstruktionen aus Abschnitt 3.3.3. Nur die einschränkende Uhrbedingung $(0, \infty)$ wird bei der Konstruktion der ω-Operation im Gegensatz zum *-Abschluss weggelassen. Außerdem wird der Zustand $p_\perp$ nicht mehr benötigt, da das *unendliche* Signal den Wert $\perp$ nicht annimmt. Seien

$$\mathcal{A}_{\psi_1} = (P_{\psi_1}, E_{\psi_1}, C_{\psi_1}, c_{\psi_1}l, s_{\psi_1}, d_{\psi_1}, P_{0(\psi_1)})$$

und

$$\mathcal{A}_{\psi_2} = (P_{\psi_2}, E_{\psi_2}, C_{\psi_2}, c_{\psi_2}l, s_{\psi_2}, d_{\psi_2}, P_{0(\psi_2)})$$

Phasenautomaten zu den ω-regulären Ausdrücken mit Zeit ψ_1 und ψ_2 und sei

$$\mathcal{A}_\varphi = (P_\varphi, E_\varphi, C_\varphi, c_\varphi l, s_\varphi, d_\varphi, P_{0(\varphi)})$$

ein Phasenautomat zu einem regulären Ausdruck mit Zeit φ.

Automaten zu $\psi_1 \wedge \psi_2$, zu $\psi_1 \vee \psi_2$ oder zu $\varphi \cdot \psi_1$ werden genau wie im endlichen Fall konstruiert und die Korrektheit wird analog zum endlichen Fall bewiesen.

Der Automat zu φ^ω entspricht im wesentlichen dem Automaten zu φ^*. Es wird jedoch keine zusätzliche Uhr benötigt, da die Schleife unendlich oft durchlaufen werden darf. Die Schleife wird auch tatsächlich unendlich oft durchlaufen, wie in der Definition von φ^ω gefordert, da in den Teilautomaten $\mathcal{A}_\varphi \setminus \{p_\perp\}$ und $\mathcal{A}^{(2)} \setminus \{p_\perp\}$ nach Bemerkung 3.3.1 nicht unendlich viel Zeit verstreichen darf. Damit folgt die Korrektheit der Konstruktion für den ω-Abschluss. Insgesamt gilt also der folgende Satz:

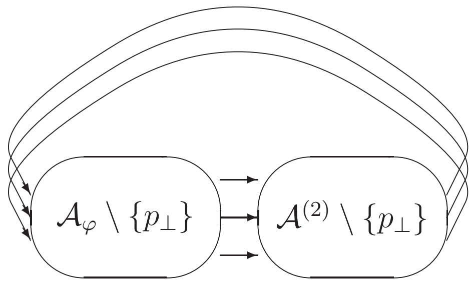


Abbildung 3.6: Automat $\mathcal{A}_{\phi^\omega}$

Satz 3.3.9 (Äquivalenz von ω-regulären Ausdrücken mit Zeit und Phasenautomaten). Für jeden ω-regulären Ausdruck mit Zeit ψ existiert ein Phasenautomat $\mathcal{A}_\psi$, der die gleichen Signale akzeptiert.

Korollar 3.3.10 (Äquivalenz von Realzeitautomaten und Phasenautomaten). Zu jedem Realzeitautomaten kann ein Phasenautomat konstruiert werden, der die gleichen Signale beschreibt und umgekehrt.

Beweis Die Konstruktion eines Realzeitautomaten zu einem Phasenautomaten ist in [Mru00] beschrieben. Für die andere Richtung kann nach [ACM97] zu einem Realzeitautomaten ein ω-regulärer Ausdruck und daraus nach Satz 3.3.9 ein Phasenautomat konstruiert werden. □

Da also Phasenautomaten und Timed Automata äquivalent sind und Timed Automata nach [AD94] nicht gegen Komplement abgeschlossen sind, ergibt sich das folgende Korollar.

3.3.1 Beispiel (Grüne Signale)

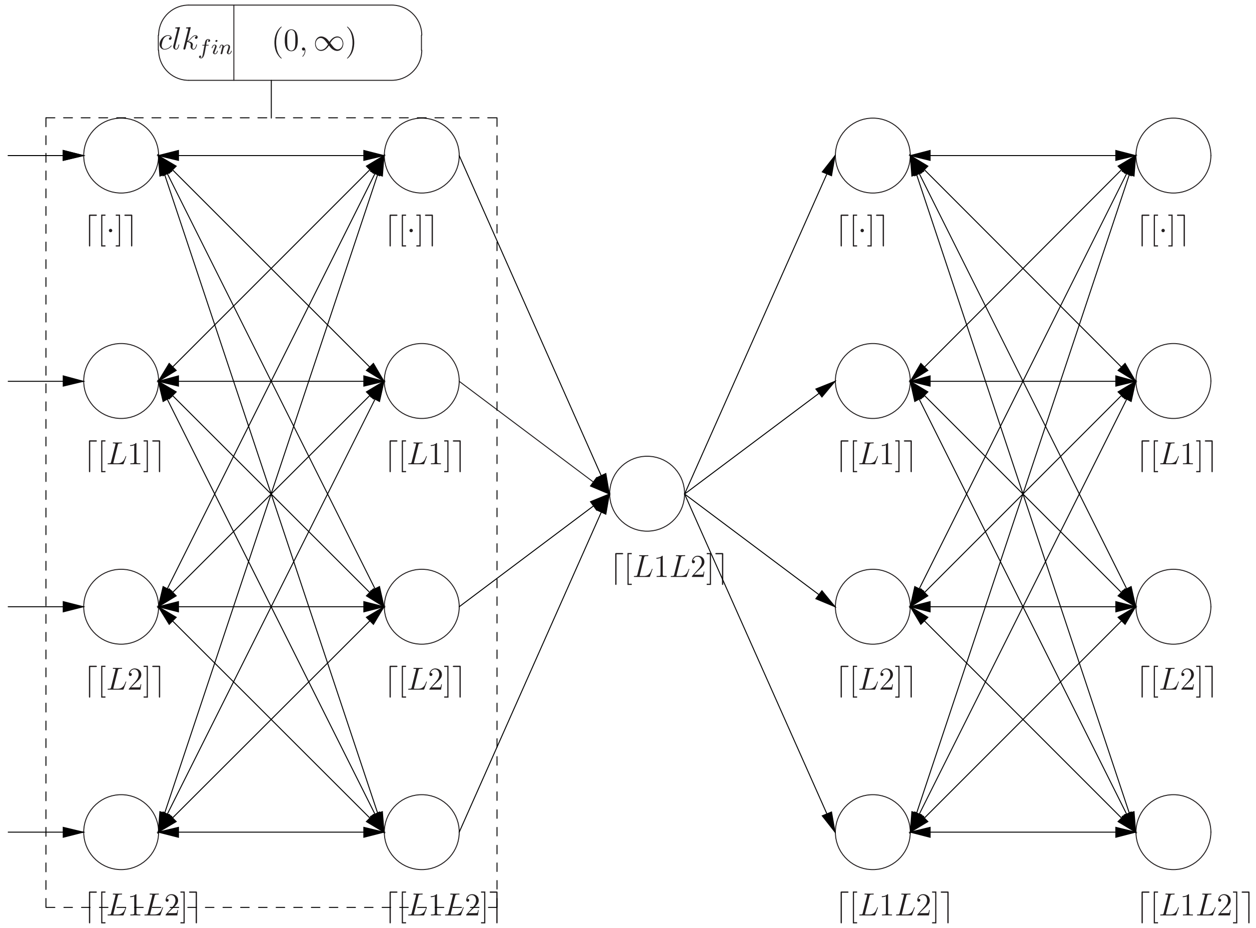
Der Fall, dass beide Signale der Weichensteuerung im Fallbeispiel gleichzeitig grün anzeigen, ist sicher ein Fehler. Der ω-reguläre Ausdruck mit Zeit

$$\psi_{sigErr} = ([\cdot] \vee [L1] \vee [L2] \vee [L1L2])^* \cdot [L1L2] \cdot ([\cdot] \vee [L1] \vee [L2] \vee [L1L2])^\omega$$

über dem Alphabet $\Sigma = \{[\cdot], [L1], [L2], [L1L2]\}$ beschreibt alle Abläufe, die genau diesen Fehlerfall enthalten. Dabei bedeuten

- $[\cdot]$ Kein Signal grün
- $[L1]$ Signal L1 grün
- $[L2]$ Signal L2 grün
- $[L1L2]$ Beide Signale grün

Wenn aus diesem Ausdruck der Phasenautomat $\mathcal{A}_{\psi_{sigErr}}$ konstruiert wird, erhält man den Automaten aus Abbildung 3.7.



Automat zu $([\cdot] \vee [L1] \vee [L2] \vee [L1L2])^*$ Automat zu $([\cdot] \vee [L1] \vee [L2] \vee [L1L2])^\omega$

Abbildung 3.7: Phasenautomat zu ψ_{sigErr}

Da also Phasenautomaten und Timed Automata äquivalent sind und Timed Automata nach [AD94] nicht gegen Komplement abgeschlossen sind, ergibt sich das folgende Korollar.

Korollar 3.3.11 (Nicht-Abgeschlossenheit gegen Komplement). Die Klasse der Phasenautomaten ist nicht gegen Komplement abgeschlossen.

3.3.5 Nicht-komplementierbarer Phasenautomat

Der Phasenautomat aus Abbildung 3.8 ist analog zum Beispiel eines Realzeitautomaten ohne Komplement aus [AD94] konstruiert und deshalb auch als Phasenautomat nicht komplementierbar. Der Phasenautomat akzeptiert alle Interpretationen, die ir-

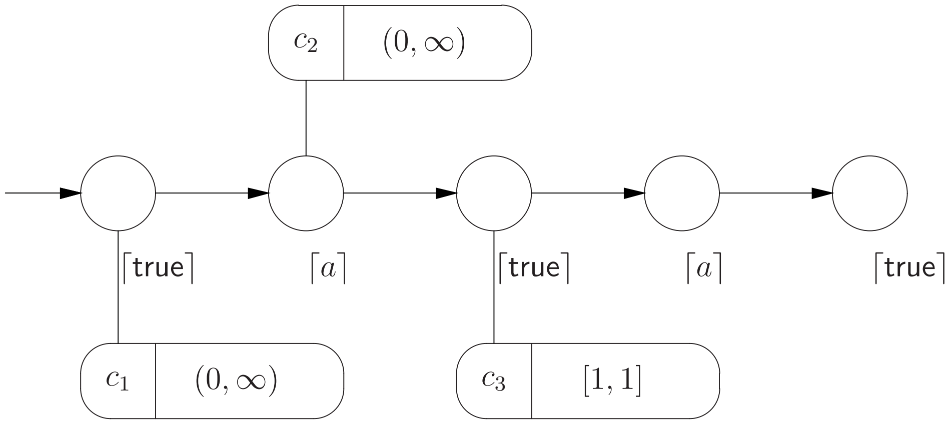


Abbildung 3.8: Nicht-komplementierbarer Phasenautomat

gendwann genau 1 ZE entfernt, zwei a-Phasen besitzen. Um dies auszuschließen muss zu jedem Zeitpunkt, an dem a erfüllt ist, eine Uhr gestartet werden, um festzustellen, ob in einer ZE wieder oder noch a gilt. Dafür sind unendlich viele Uhren notwendig.

3.4 Komplementkonstruktion für eine Teilklasse

Obwohl nicht für alle Phasenautomaten ein Komplement existiert, kann zumindest für eine Teilklasse eine Konstruktion angegeben werden.

Konstruktionsidee Die betrachtete Teilklasse sind deterministische Phasenautomaten, d.h. Phasenautomaten, die für jedes Paar $(\mathcal{I}, \mathcal{V})$, das von dem Phasenautomat akzeptiert wird, genau einen Ablauf besitzen. Deterministische Phasenautomaten können so erweitert werden, dass sie nicht mehr „steckenbleiben“ können, also alle Paare

$(\mathcal{I}, \mathcal{V})$ akzeptieren und noch deterministisch sind. Wegen des Determinismus führen alle Abläufe zu einem Paar $(\mathcal{I}, \mathcal{V})$, das auch vom Ausgangsautomaten akzeptiert wird nur durch Phasen, die auch der Ausgangsautomat besitzt. Wenn durch eine zusätzliche Uhr verhindert wird, dass ein Ablauf des erweiterten Automaten nur durch Phasen führt, die auch im Ausgangsautomaten vorkommen, erhält man einen Automaten der das Komplement akzeptiert.

3.4.1 Beispiel (Komplement des Synchronisations-Implementables)

In [Tap01] wird ein Phasenautomat zu dem Synchronisations-Implementable

$$IM_{sync} = \neg\Diamond(\lceil \pi \wedge \varphi \rceil^t ; \lceil \pi \rceil)$$

angegeben (Abbildung 3.9). Dieser Phasenautomat ist nicht deterministisch, da ein

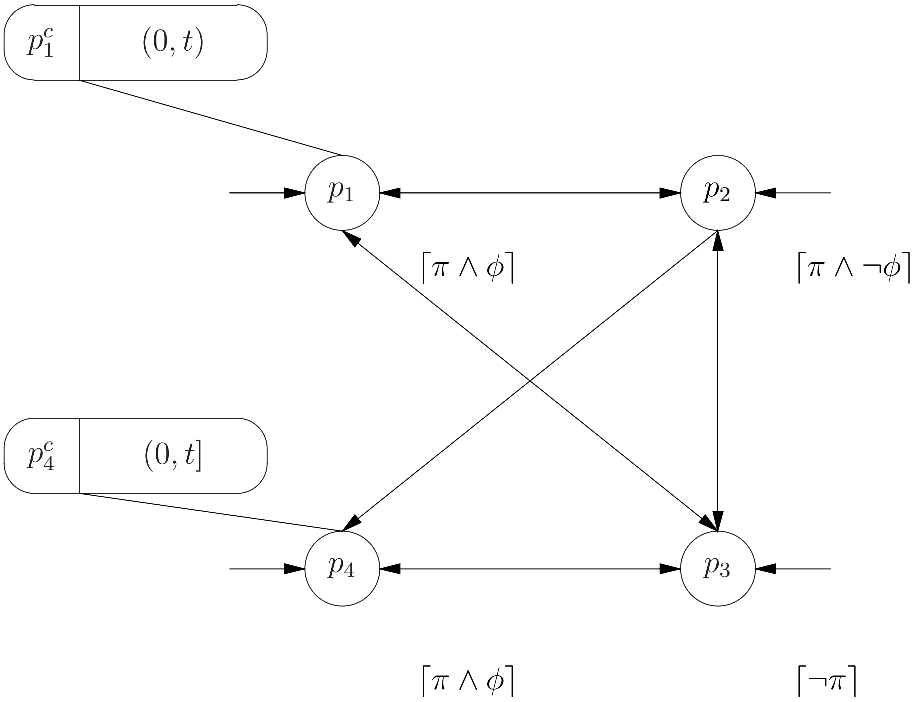


Abbildung 3.9: Phasenautomat zu $\neg\Diamond(\lceil \pi \wedge \phi \rceil^t ; \lceil \pi \rceil)$

Ablauf, der weniger als t Zeiteinheiten lang $\lceil \pi \wedge \phi \rceil$ erfüllt, sowohl durch die Phase p_1 als auch durch p_4 führen kann. Der Determinismus kann aber leicht erreicht werden (Abbildung 3.10). In Abbildung 3.11 ist der Automat vervollständigt und die Phasen des Ausgangsautomaten dürfen wegen der zusätzlichen Uhr nicht unendlich lange angenommen werden. Dieses Verfahren führt aber nicht zu einer optimalen Lösung, wie man an dem Automaten in Abbildung 3.12 sieht.

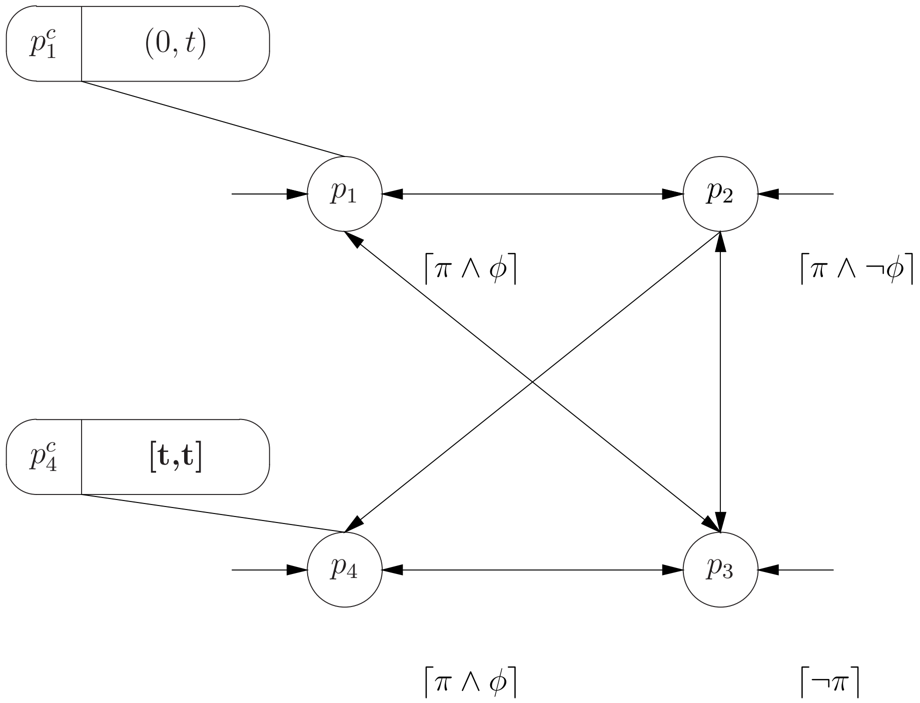


Abbildung 3.10: Deterministischer Phasenautomat zu $\neg\Diamond(\lceil \pi \wedge \phi \rceil^t ; \lceil \pi \rceil)$

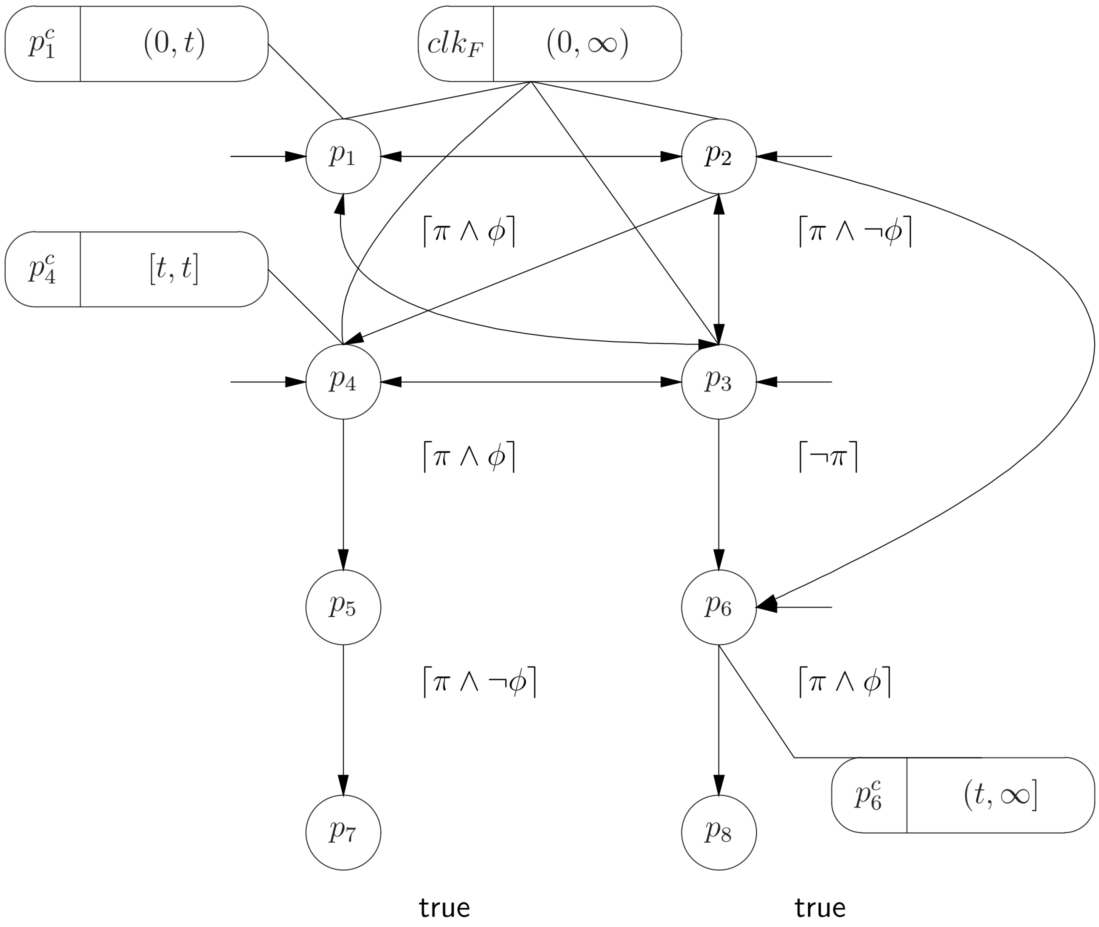


Abbildung 3.11: Deterministischer Phasenautomat zu $\neg\neg\Diamond(\lceil \pi \wedge \phi \rceil^t; \lceil \pi \rceil)$

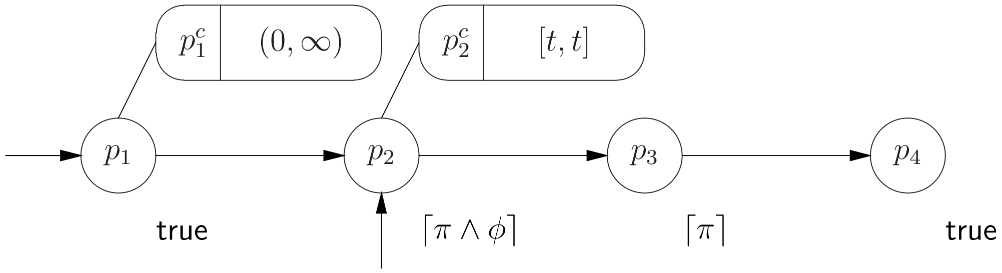


Abbildung 3.12: Einfacherer Phasenautomat zu $\Diamond(\lceil \pi \wedge \phi \rceil^t; \lceil \pi \rceil)$

3.4.1 Deterministische Phasenautomaten

Definition 3.4.1 (Deterministische Phasenautomaten). Ein Phasenautomat $\mathcal{A} = (P, E, C, cl, s, d, P_0)$ heiße deterministisch, falls folgende Bedingungen erfüllt sind.

- Jedem Zustand ist höchstens eine Uhr zugeordnet, d.h.

$$|cl(p)| \leq 1 \quad \text{für alle } p \in P. \tag{3.4.3.1}$$

- Jeder Uhr ist genau ein Zustand zugeordnet, d.h.

$$|cl^{-1}(c)| = 1 \quad \text{für alle } c \in C. \tag{3.4.3.2}$$

- Für zwei Startphasen oder zwei Phasen p, p' mit gleichem Vorgänger gilt für die Zusicherungen

$$s(p) = s(p') \,\text{oder}\, s(p) \wedge s(p') \equiv \mathsf{false}. \tag{3.4.3.3}$$

- Für alle Startphasen $p, p' \in P_0$ mit $p \neq p'$ gilt: Falls $s(p) = s(p')$, dann

$$d(cl(p)) \cap d(cl(p')) = \emptyset. \tag{3.4.3.4}$$

  Falls in einer Uhrbedingung eine Variable vorkommt, soll die Bedingung (3.4.3.4)für alle Belegungen der Variablen erfüllt sein.

- Für alle Phasen $p \in P$ und alle Nachfolger $\hat{p}$ von p gilt:

$$s(p) \wedge s(\hat{p}) \equiv \mathsf{false}. \tag{3.4.3.5}$$

- Für alle Phasen $p, p_1, p_2 \in P$, wobei p_1 und p_2 unterschiedliche Nachfolgephasen von p sind gilt: Falls $s(p_1) = s(p_2)$, dann

$$d(cl(p_1)) \cap d(cl(p_2)) = \emptyset. \tag{3.4.3.6}$$

  In diesem Fall müssen die Uhrintervalle ebenfalls unter allen Belegungen disjunkt sein.

Lemma 3.4.2 (Determinismus). Sei $\mathcal{A}$ ein deterministischer Phasenautomat und $\mathcal{I}, \mathcal{V}$ eine Interpretation und eine Belegung, die $\exists \mathsf{ph}_{\mathcal{A}}[D(\mathcal{A})]$ erfüllt, dann ist der Lauf des Phasenautomaten eindeutig bestimmt, d.h. es existiert genau ein $\mathcal{I}'$ mit $\mathcal{I} =_{\backslash \mathsf{ph}_{\mathcal{A}}} \mathcal{I}'$ und $\mathcal{I}', \mathcal{V} \models_{DC} [D(\mathcal{A})]$.

Beweis Angenommen es existieren zwei Interpretationen $\mathcal{I}_1$ und $\mathcal{I}_2$ mit $\mathcal{I} =_{\backslash \mathsf{ph}_\mathcal{A}} \mathcal{I}_1 =_{\backslash \mathsf{ph}_\mathcal{A}} \mathcal{I}_2$, $\mathcal{I}_1(\mathsf{ph}_\mathcal{A}) \neq \mathcal{I}_2(\mathsf{ph}_\mathcal{A})$ und $\mathcal{I}_1 \models_{DC} [D(\mathcal{A})]$ und $\mathcal{I}_2 \models_{DC} [D(\mathcal{A})]$.
Sei $\tau = \inf\{t : \mathcal{I}_1(\mathsf{ph}_\mathcal{A})(t) \neq \mathcal{I}_2(\mathsf{ph}_\mathcal{A})(t)\}$ und $p_1 = \mathcal{I}_1(\mathsf{ph}_\mathcal{A})(\tau)$ und $p_2 = \mathcal{I}_2(\mathsf{ph}_\mathcal{A})(\tau)$.

1. Fall $\tau = 0$. In diesem Fall sind p_1 und p_2 unterschiedliche Startphasen. Da $\mathcal{I}_1$ und $\mathcal{I}_2$ bis auf die Observable $\mathsf{ph}_\mathcal{A}$ übereinstimmen, müssen die beiden Zustandszusicherungen $s(p_1)$ und $s(p_2)$ beide erfüllt sein, wegen (3.4.3.3) gilt sogar $s(p_1) = s(p_2)$ und somit müssen die Intervalle der den Zuständen zugeordneten Uhren disjunkt sein.
   Falls in der Interpretation $\mathcal{I}_1$ die Phase p_1 wieder verlassen wird, muss wegen (3.4.3.5) und $s(p_1) = s(p_2)$ zum gleichen Zeitpunkt in $\mathcal{I}_2$ die Phase p_2 verlassen werden. Das ist ein Widerspruch zur Disjunktheit der Uhrintervalle.
   Falls in $\mathcal{I}_1$ die Phase p_1 nicht verlassen wird, folgt mit der gleichen Argumentation, dass die Phase p_2 in $\mathcal{I}_2$ nicht verlassen wird und damit ebenfalls ein Widerspruch zur Disjunktheit der Uhrintervalle.

2. Fall $\tau > 0$. Wegen der Minimalität von τ sind die Phasen in beiden Interpretationen vor dem Zeitpunkt τ gleich. Sei dann p die Phase, in der sich beide Interpretationen vor dem Zeitpunkt τ befinden. Wegen (3.4.3.5) muss in *beiden* Interpretationen die Phase p zum gleichen Zeitpunkt verlassen werden. Wie im 1. Fall müssen die beiden Interpretationen die Zusicherungen $s(p_1)$ und $s(p_2)$ erfüllen und deshalb nach (3.4.3.6) die beiden Uhrintervalle von p_1 und p_2 disjunkt sein. Der Widerspruch folgt dann wie im 1. Fall.

Also ist die Annahme falsch. □

3.4.2 Komplement für deterministische Phasenautomaten

Sei $\mathcal{A} = (P, E, C, cl, s, d, P_0)$ ein deterministischer Phasenautomat. Für den Komplement-Automaten $\bar{\mathcal{A}} = (\bar{P}, \bar{E}, \bar{C}, \bar{cl}, \bar{s}, \bar{d}, \bar{P_0})$ wird der Automat $\mathcal{A}$ zuerst zu $\mathcal{A}_\mathsf{true}$ erweitert.

1. Setze $P_\mathsf{true} \stackrel{df}{=} P$, $E_\mathsf{true} \stackrel{df}{=} E$, $C_\mathsf{true} \stackrel{df}{=} C$, $cl_\mathsf{true} \stackrel{df}{=} cl$, $s_\mathsf{true} \stackrel{df}{=} s$, $d_\mathsf{true} \stackrel{df}{=} d$, $P_{0\,\mathsf{true}} \stackrel{df}{=} P_0$

2. Hinzufügen einer Startphase, deren Zustandzusicherung erfüllt ist, falls keine andere Zusicherung einer Startphase erfüllt ist.
   Falls $\bigvee_{p \in P_0} s(p) \not\equiv \mathsf{true}$, dann füge einen Zustand p_0^{sonst} hinzu mit
   $$s_\mathsf{true}(p_0^{sonst}) \stackrel{df}{=} \neg \bigvee_{p \in P_0} s(p),$$
   einen Zustand p_0^true mit Zusicherung
   $$s_\mathsf{true}(p_0^\mathsf{true}) \stackrel{df}{=} \neg s_\mathsf{true}(p_0^{sonst})$$

und einen Übergang von p_0^{sonst} zu p_0^{true} und zurück.

3. Für jeden Zustand des Phasenautomaten wird für die Nachfolgezustände analog verfahren. Sei p ein Zustand des Phasenautomaten mit Nachfolgezuständen $p_1, \dots p_l$. Falls $\bigvee_{p_i} s(p_i) \vee s(p) \not\equiv \mathsf{true}$ gilt, füge einen Zustand p^{sonst} hinzu, mit Zusicherung
$$s_{\mathsf{true}}(p^{sonst}) \stackrel{df}{=} \neg((\bigvee_{p_i} s(p_i)) \vee s(p))$$
und einem Nachfolgezustand p^{true} wie unter Punkt 2.

4. Für jede Menge von Startphasen oder Phasen mit gleichem Vorgänger, die die gleiche Zusicherung besitzen müssen die Intervalle der zugehörigen Uhren disjunkt sein. Da die Vereinigung der Intervalle aber nicht das Intervall $(0, \infty]$ ergeben muss, kann der Automat „steckenbleiben“.
Sei $\mathbb{P} = \{P' \subseteq P | P' = P_0 \vee \exists p \in P : P' = suc(p)\}$ die Menge aller Mengen von Nachfolgezuständen und Startzuständen. Für alle Mengen $P' \in \mathbb{P}$ und alle maximal großen Teilmengen P_{eq} von P' mit gleichen Zusicherungen $s(P_{eq})$ führe folgende Konstruktion aus:

   Sei $IV' = \{d(cl(p)) | p \in P_{eq}\}$ die Menge der vorkommenden Uhrintervalle und IV eine kleinstmögliche Erweiterung von IV' mit $\bigcup_{I \in IV} I = (0, \infty]$, so dass zwei Intervalle in IV weiterhin disjunkt sind.
Für jedes Intervall $I_{sonst} \in IV \setminus IV'$ füge hinzu

   a) eine Phase $p^{I_{sonst}}$ mit Zusicherung $s_{\mathsf{true}}(p^{I_{sonst}}) = s(P_{eq})$,
   b) eine Uhr $clk^{I_{sonst}}$ mit $cl_{\mathsf{true}}(p^{I_{sonst}}) = clk^{I_{sonst}}$ und $d_{\mathsf{true}}(clk^{I_{sonst}}) = I_{sonst}$,
   c) eine Phase $p^{I_{sonst}(2)}$ mit Zusicherung $s_{\mathsf{true}}(p^{I_{sonst}(2)}) = \neg s(P_{eq})$,
   d) einen Übergang von $p^{I_{sonst}}$ zu $p^{I_{sonst}(2)}$ und
   e) einen Übergang von $p^{I_{sonst}(2)}$ zu einem true-Zustand wie unter Punkt 2.

Lemma 3.4.3. Der so konstruierte Automat $\mathcal{A}_{\mathsf{true}}$ ist deterministisch und
$$F(\mathcal{A}_{\mathsf{true}}) \equiv \mathsf{true}.$$

Beweis

Determinismus Der Phasenautomat $\mathcal{A}$ ist deterministisch. Die durch die Konstruktion hinzugefügten Phasen erfüllen ebenfalls die geforderten Bedingungen.

Bedingung $F(\mathcal{A}_{\mathsf{true}}) \equiv$ **true** Sei $\mathcal{I}$ eine beliebige Interpretation über der Menge der betrachteten Observablen und $\mathcal{V}$ eine Belegung der Variablen. Dann kann $\mathcal{I}'$ mit $\mathcal{I} =_{\backslash \mathsf{ph}_{\mathcal{A}}} \mathcal{I}'$ und $\mathcal{I}', \mathcal{V} \models_{DC} D(\mathcal{A}_{\mathsf{true}})$ induktiv über die Anzahl der Veränderungen konstruiert werden. Sei $t_0, t_1, \ldots$ die Folge von Zeitpunkten zu denen sich die Werte von $\mathcal{I}$ ändern, d.h. die Unstetigkeitsstellen von $\mathcal{I}$. Sei außerdem $\mathrm{dur}_{\mathcal{I}}$ als Funktion definiert, die misst, wie lange die Belegung der Observablen einer Interpretation $\mathcal{I}$ noch konstant bleibt:

$$\mathrm{dur}_{\mathcal{I}} : \begin{cases} \textbf{time} & \to \textbf{time} \\ t & \mapsto \tau - t \quad \text{mit } \tau = \min\{t_n | t_n > t\} \end{cases} \tag{3.4.3.7}$$

Dann wird der Lauf $\mathcal{I}'(\mathsf{ph}_{\mathcal{A}})$ induktiv konstruiert.

Induktionsanfang $t_0 = 0$

1. Fall $\mathcal{I}$ erfüllt zum Zeitpunkt 0 keine Zusicherung aus P_0. Dann erfüllt $\mathcal{I}$ die Zusicherung von p_0^{sonst}. Setze

$$\mathcal{I}'(\mathsf{ph}_{\mathcal{A}})(t) \stackrel{df}{=} \begin{cases} p_0^{sonst} & \text{falls } t < t_1 \\ p_0^{true} & \text{sonst} \end{cases}$$

Dies ist offensichtlich ein gültiger Lauf des Phasenautomaten und das Paar $(\mathcal{I}', \mathcal{V})$ erfüllt die Formel $D(\mathcal{A}_{\mathsf{true}})$.

2. Fall $\mathcal{I}$ erfüllt eine Zusicherung aus P_0. Wegen der Disjunktheit der Uhrintervalle in $\mathcal{A}_{true}$ gibt es genau eine Zustand p_0, so dass unter der Belegung $\mathcal{V}$ gilt $\mathrm{dur}_{\mathcal{I}}(0) \in d(cl(p_0))$. Die Belegung muss mit berücksichtigt werden, da die Intervalle Variablen enthalten können. Setze dann

$$\mathcal{I}'(\mathsf{ph}_{\mathcal{A}})(t) \stackrel{df}{=} \begin{cases} p_0 & \text{falls } t < \mathrm{dur}_{\mathcal{I}}(0) \\ \text{undef.} & \text{sonst} \end{cases}$$

Damit gilt dann

- $$\mathcal{I}', \mathcal{V}, [0, t_1) \models_{DC} \neg(\lceil \mathsf{ph}_{\mathcal{A}} \notin P_{0\,\mathsf{true}} \rceil ; \mathsf{true}) \qquad (= D_{init}(\mathcal{A}_{\mathsf{true}}))$$

  weil p_0 eine Startphase ist.

- Für alle Phasen p gilt:

  $$\mathcal{I}', \mathcal{V}, [0, t_1) \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}} = p \rceil ; \lceil \mathsf{ph}_{\mathcal{A}} \notin \widetilde{\mathrm{suc}}(p) \rceil) \qquad (= D_{suc}^{p}(\mathcal{A}_{\mathsf{true}}))$$

  weil $\mathcal{I}'(\mathsf{ph}_{\mathcal{A}})$ auf $[0, t_1)$ konstant ist.

- Für alle Phasen p gilt:

$$\mathcal{I}', \mathcal{V}, [0, t_1) \models_{DC} \neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}} = p \wedge \neg s(p) \rceil \qquad (= D^p_{out}(\mathcal{A}_{\mathsf{true}}))$$

nach Definition von $\mathcal{I}'(\mathsf{ph}_{\mathcal{A}})$

- Für alle Uhren c gilt:

$$\mathcal{I}', \mathcal{V}', [0, t_1) \models_{DC} \neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}} \in cl^{-1}(c) \rceil^{>d(c)} \qquad (= D^c_{pr}(\mathcal{A}_{\mathsf{true}}))$$

nach Definition von $\mathcal{I}'(\mathsf{ph}_{\mathcal{A}})$.

- Für alle Uhren c gilt:

$$\mathcal{I}', \mathcal{V}', [0, t_1) \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}} \notin cl^{-1}(c) \rceil; \lceil \mathsf{ph}_{\mathcal{A}} \in cl^{-1}(c) \rceil^{<d(c)}; \lceil \mathsf{ph}_{\mathcal{A}} \notin cl^{-1}(c) \rceil) \qquad (= D^c_{st}(\mathcal{A}_{\mathsf{true}}))$$

- Für alle Uhren c gilt:

$$\mathcal{I}', \mathcal{V}', [0, t_1) \models_{DC} \neg(\lceil \mathsf{ph}_{\mathcal{A}} \notin cl^{-1}(c) \rceil^{<d(c)}; \lceil \mathsf{ph}_{\mathcal{A}} \notin cl^{-1}(c) \rceil) \qquad (= D^c_{ist}(\mathcal{A}_{\mathsf{true}}))$$

und deshalbt auch $\mathcal{I}', \mathcal{V}, [0, t_1) \models_{DC} D(\mathcal{A}_{\mathsf{true}})$.

Induktionsschritt Sei $\mathcal{I}'$ bereits für den Bereich t_0 bis t_n konstruiert, so dass

$$\mathcal{I}', \mathcal{V}, [0, t_n) \models_{DC} D(\mathcal{A}_{\mathsf{true}}).$$

Sei $p \overset{df}{=} \mathcal{I}'(\mathsf{ph}_{\mathcal{A}})(t_{n-1})$.

1. Fall $\mathcal{I}$ erfüllt zum Zeitpunkt t_n keine Zusicherung aus suc(p) im Automaten $\mathcal{A}$. Dann erfüllt $\mathcal{I}$ die Zusicherung von p_0^{sonst} im Automaten $\mathcal{A}_{\mathsf{true}}$. Setze

$$\hat{\mathcal{I}}'(\mathsf{ph}_{\mathcal{A}})(t) = \begin{cases} \mathcal{I}'(\mathsf{ph}_{\mathcal{A}})(t) & \text{falls } t < t_n \\ p^{sonst} & \text{falls } t_n \leq t < t_{n+1} \\ p^{true} & \text{sonst} \end{cases}$$

Da $\hat{\mathcal{I}}'$ eine Erweiterung von $\mathcal{I}'$ ist, wird $\hat{\mathcal{I}}'$ wieder $\mathcal{I}'$ genannt. Dann erfüllt $(\mathcal{I}', \mathcal{V})$ die Formel $D(\mathcal{A}_{\mathsf{true}})$, wie man genau wie im Fall des Induktionsanfangs sieht.

2.Fall $\mathcal{I}$ erfüllt zum Zeitpunkt t_n eine Zusicherung aus suc(p) in $\mathcal{A}$. Analog zum Induktionsanfang gibt es genau eine Zustand p', so dass unter der Belegung $\mathcal{V}$ gilt $\mathrm{dur}_{\mathcal{I}}(t_n) \in d(cl(p'))$. Setze dann

$$\hat{\mathcal{I}}'(\mathsf{ph}_{\mathcal{A}})(t) = \begin{cases} \mathcal{I}'(\mathsf{ph}_{\mathcal{A}})(t) & \text{falls } t < t_n \\ p' & \text{falls } t_n \leq t < t_{n+1} \\ \text{undef.} & \text{sonst} \end{cases}$$

Die Unterscheidung zwischen $\hat{\mathcal{I}}'$ und $\mathcal{I}'$ wird aufgegeben und es gilt dann

$$\mathcal{I}', \mathcal{V},\ [0, t_{n+1}) \models_{DC} D(\mathcal{A}_{\mathsf{true}}).$$

Insgesamt gilt damit $\mathcal{I}', \mathcal{V} \models_{DC} D(\mathcal{A}_{\mathsf{true}})$. □

Satz 3.4.4 (Komplementkonstruktion). Sei $\mathcal{A} = (P, E, C, cl, s, d, P_0)$ ein deterministischer Phasenautomat und $\mathcal{A}_{\mathsf{true}} = (P_{\mathsf{true}}, E_{\mathsf{true}}, C_{\mathsf{true}}, cl_{\mathsf{true}}, s_{\mathsf{true}}, d_{\mathsf{true}}, P_{0\,\mathsf{true}})$ die wie angegeben konstruierte Erweiterung. Sei

$$\bar{\mathcal{A}} \stackrel{df}{=} (P_{\mathsf{true}}, E_{\mathsf{true}}, C_{\mathsf{true}} \cup \{clk_{fin}\}, \bar{cl}, s_{\mathsf{true}}, \bar{d}, P_{0\,\mathsf{true}})$$

mit

$$\bar{cl}(p) \stackrel{df}{=} \begin{cases} cl_{\mathsf{true}}(p) & \text{falls } p \notin P \\ cl_{\mathsf{true}}(p) \cup \{clk_{fin}\} & \text{falls } p \in P \end{cases}$$

und

$$\bar{d}(clk) \stackrel{df}{=} \begin{cases} d_{\mathsf{true}}(clk) & \text{falls } clk \neq clk_{fin} \\ (0, \infty) & \text{falls } clk = clk_{fin}. \end{cases}.$$

Sei x_c die für das Uhrintervall zusätzlich eingeführte Variable, die sonst nicht vorkommt. Dann gilt

$$\models F(\mathcal{A}) \iff \neg(\exists\, x_c : F(\bar{\mathcal{A}}))$$

Beweis „⇒"-Richtung Sei $(\mathcal{I}, \mathcal{V})$ ein Paar, dass $F(\mathcal{A})$ erfüllt. Nach Lemma 3.4.2 gibt es genau ein $\mathcal{I}'$ mit $\mathcal{I} =_{\backslash \mathsf{ph}_{\mathcal{A}}} \mathcal{I}'$ und $\mathcal{I}', \mathcal{V} \models [D(\mathcal{A})]$. Da jeder Lauf von $\mathcal{A}$ auch ein Lauf von $\mathcal{A}_{\mathsf{true}}$ ist und $\mathcal{A}_{\mathsf{true}}$ deterministisch ist, ist $\mathcal{I}'$ auch die einzige bis auf $\mathsf{ph}_{\mathcal{A}}$ mit $\mathcal{I}$ identische Interpretation, die $[D(\mathcal{A}_{\mathsf{true}})]$ erfüllt. Außerdem werden nur Zustände aus P angenommen. Wegen der zusätzlichen Uhrbedingung in $\bar{\mathcal{A}}$ gilt dann $(\mathcal{I}', \mathcal{V}) \not\models [D(\bar{\mathcal{A}})]$. Wenn es eine andere Interpretation $\mathcal{I}''$ gäbe mit $\mathcal{I} =_{\backslash \mathsf{ph}_{\mathcal{A}}} \mathcal{I}''$ und $(\mathcal{I}'', \mathcal{V}) \models [D(\bar{\mathcal{A}})]$, dann würde diese Interpretation auch $[D(\mathcal{A}_{\mathsf{true}})]$ erfüllen. Das ist wegen des Determinismus von $\mathcal{A}_{\mathsf{true}}$ ausgeschlossen, also gilt

$$(\mathcal{I}, \mathcal{V}) \not\models F(\mathcal{A}).$$

„⇐"-Richtung Sei $(\mathcal{I}', \mathcal{V}')$ ein Paar, dass nicht $F(\mathcal{A})$ erfüllt. Nach Lemma 3.4.3 gibt es aber eine Interpretation $\mathcal{I}'$ mit $\mathcal{I} =_{\backslash \mathsf{ph}_{\mathcal{A}}} \mathcal{I}'$ und $\mathcal{I}', \mathcal{V} \models [D(\mathcal{A}_{\mathsf{true}})]$. Dann gibt es einen Zeitpunkt t, so dass $\mathcal{I}'$ danach eine Zeit lang Phasen aus $P_{\mathsf{true}} \setminus P$ annimmt. Da nach Konstruktion kein Weg von einer Phase aus $P_{\mathsf{true}} \setminus P$ zurück zu einer Phase aus P existiert, werden nach diesem Zeitpunkt t gar keine Phasen aus P mehr angenommen. Setze dann $\mathcal{V}'(x_c) \stackrel{df}{=} t$ und $\mathcal{V}'(x) \stackrel{df}{=} \mathcal{V}(x)$ für alle anderen Variablen. Damit gilt auch $\mathcal{I}', \mathcal{V}' \models [D(\bar{\mathcal{A}})]$, weil sich $\mathcal{A}_{\mathsf{true}}$ und $\bar{\mathcal{A}}$ nur in der Uhrbedingung unterscheiden, die fordert, dass irgendwann nur noch Phasen aus $P_{\mathsf{true}} \setminus P$ angenommen werden. Also gilt

$$\mathcal{I}, \mathcal{V} \models \exists x_c : F(\bar{\mathcal{A}})$$

□

3.4.3 Intervalle der Form $(0, \infty)$

Da formal Intervalle der Form $(0, \infty)$ eine Abkürzung für $(0, x]$ darstellen, wobei x eine Variable ist, die sonst nicht benutzt wird, stimmt das Ergebnis der angegebenen Konstruktion nicht mit der Intuition überein.

3.4.2 Beispiel (Probleme bei $(0, \infty)$-Intervallen)

Formal entspricht der Automat in Abbildung 3.13 mit dem Intervall $(0, \infty)$ dem Automaten in Abbildung 3.14. Die Vervollständigung sieht man in Abbildung 3.15,

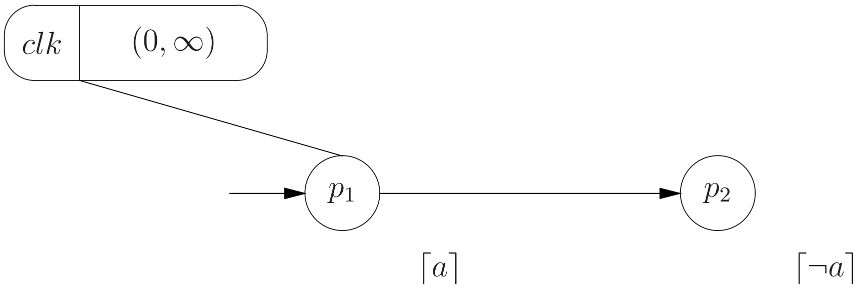


Abbildung 3.13: Phasenautomat $\mathcal{A}$

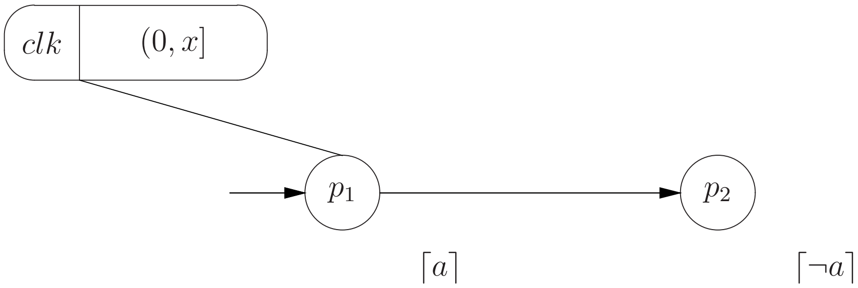


Abbildung 3.14: Phasenautomat $\mathcal{A}_{(0,\infty)}$

den Automaten zum formalen Komplement in Abbildung 3.16. In dem Komplement-Automaten können die beiden Zustände p_1 und p_2 offensichtlich entfernt werden. Der Automat $\mathcal{A}_{(0,\infty)}$ akzeptiert alle Paare $(\mathcal{I}, \mathcal{V})$ die mit einer a Phase starten, die kürzer als die Belegung von x ist und danach in eine $\neg a$ Phase übergehen. Dementsprechend akzeptiert $\bar{\mathcal{A}}_{(0,\infty)}$ alle Paare $(\mathcal{I}, \mathcal{V})$ die mit einer $\neg a$ Phase beginnen oder mit einer a-Phase, die länger als die Belegung von x ist. Das entspricht nicht der intuitiven Idee des Komplements. Man würde dagegen erwarten, dass alle Interpretationen akzeptiert werden, die mit $\neg a$ beginnen oder eine unendlich lange a Phase haben. Soll so ein

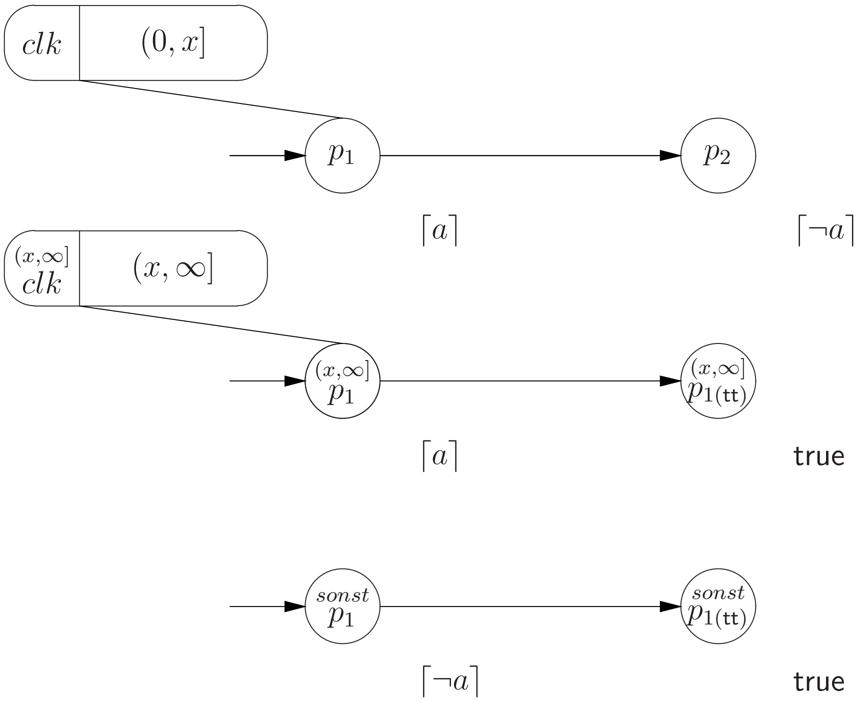


Abbildung 3.15: Phasenautomat nach Vervollständigung $\mathcal{A}_{(0,\infty)(\mathsf{true})}$

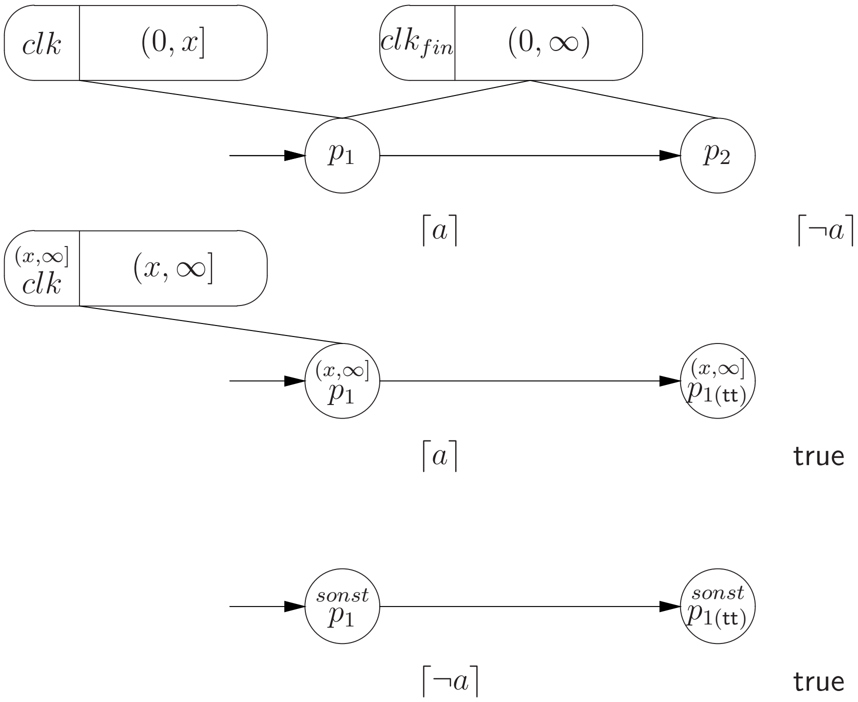


Abbildung 3.16: Komplement-Automat $\bar{\mathcal{A}}_{(0,\infty)}$

Automat konstruiert werden, so muss ein Intervall (a, ∞) mit einem Intervall $[\infty, \infty]$ ergänzt werden. Eine solche Uhrbedingung bedeutet, dass der Zustand nicht verlassen werden darf, also keine Nachfolger besitzt. Der entsprechende Automat ist $\bar{\mathcal{A}}^{(2)}{}_{(0,\infty)}$ in Abbildung 3.17.

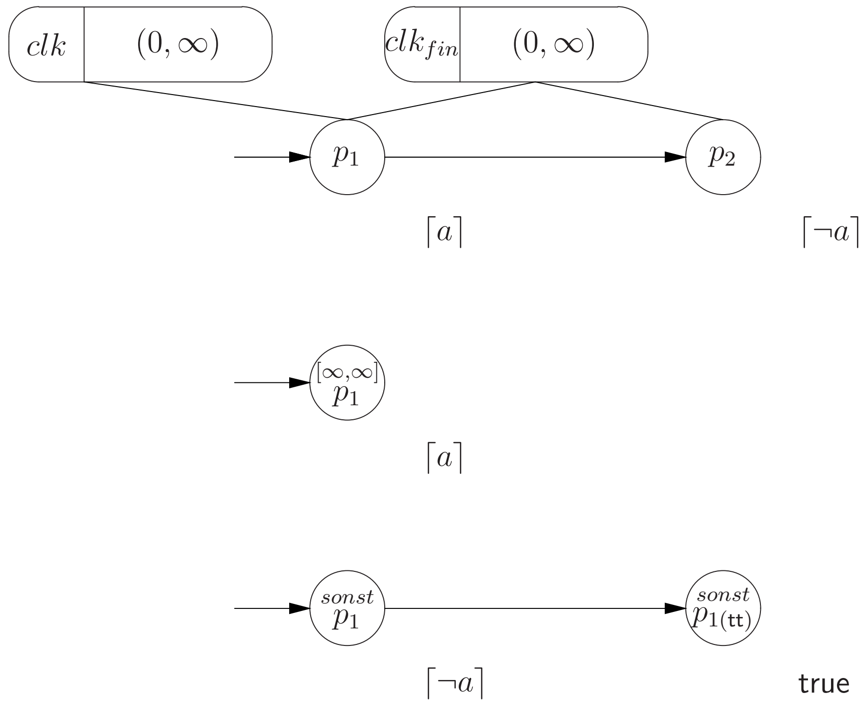


Abbildung 3.17: Alternatives Komplement $\bar{\mathcal{A}}^{(2)}{}_{(0,\infty)}$

3.4.4 Vereinfachungen

In [Tap01] wird gezeigt, dass in Phasenautomaten Phasen entfernt werden können, falls sie durch keine Transition verlassen werden können und eine gemeinsame Uhr mit einer oberen Schranke die echt kleiner als unendlich ist besitzen. Für die durch die Komplementkonstruktion entstandenen Automaten ist diese Methode interessant, da durch die neu eingeführte Uhr gerade solche Fälle entstehen, wie man in Abbildung 3.16 sieht. Hier können die Phasen p_1 und p_2 entfernt werden.

3.5 Moby/DC

Die Entwicklungsumgebung MOBY/DC wurde von J. Tapken ([Tap01]) entwickelt und gehört zu den MOBY[1]-Umgebungen. Sie enthält einen graphischen Editor und den Model-Checker MC4DC für Phasenautomaten. Außerdem existieren Export-Funktionen, um andere Model-Checker wie KRONOS und CMC benutzen zu können. Der Model-Checker prüft, ob ein Netzwerk von Phasenautomaten keinen Ablauf hat, d.h. ob für die Phasenautomaten $\mathcal{A}_1, \ldots, \mathcal{A}_n$ gilt:

$$\models \neg F(\mathcal{A}_1 \parallel \ldots \parallel \mathcal{A}_n)$$

Abbildung 3.18 zeigt die Darstellung des Phasenautomaten aus Abbildung 3.13 im

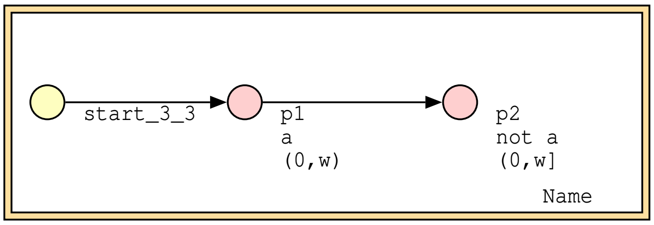


Abbildung 3.18: Moby/DC-Darstellung des Phasenautomaten aus Abbildung 3.13

graphischen Editor von MOBY/DC. Die Phasen sind jeweils mit Name, Zustandsausdruck und Intervall beschriftet. Jeder Phase muss in MOBY/DC genau eine Uhr zugeordnet werden. In [Tap01] wird gezeigt, dass das keine Einschränkung ist, da jeder Phasenautomat in ein Netzwerk von Phasenautomaten übersetzt werden kann, in dem jeder Phasenautomat die Bedingung erfüllt. Außerdem wird der Startzustand nicht durch einen Pfeil ohne Ursprungsknoten markiert, sondern durch einen Pfeil, der von einem mit `start_` beschrifteten Knoten ausgeht. Dieser Start-Knoten ist keine Phase (kein Zustand) des Phasenautomaten.

[1] **MO**delling of distri**B**uted s**Y**stems

4 Fehlerbaumanalyse

Nach [IEC93] ist ein Fehlerbaum (in der Norm Störungsbaum genannt) eine

> „geordnete graphische Darstellung der Bedingungen und Einflüsse, die zum festgelegten unerwünschten Ereignis (hier immer Hauptereignis genannt) führen oder beitragen.“

4.1 Klassische Fehlerbaumanalyse

Die Technik der Fehlerbaumanalyse (engl. Fault Tree Analysis, FTA) wurde 1961 von H.A. Watson an den Bell Laboratories für die Analyse des Startsystems der Minuteman Raketen entwickelt ([Lev95]). Es wird dabei von einem unerwünschten Ereignis (Top-Event) ausgegangen und dann werden schrittweise die Gründe für dieses Ereignis analysiert. Die dabei auftretenden Zwischenereignisse werden durch *Gatter* mit den übergeordneten Ereignissen in Beziehung gesetzt. Auf diese Weise entsteht eine Baumstruktur, in der auf jeder Ebene das gleiche Ereignis in unterschiedlichem Detaillierungsgrad dargestellt wird, da in jedem Analyseschritt eine Verfeinerung des übergeordneten Fehlers stattfindet. Die üblicherweise verwendeten Gatter sind in Tabelle 4.1 dargestellt. Die Norm [IEC93] empfiehlt zwar Symbole für die Gatter stellt jedoch auch die Nutzung älterer Symbole oder den Einsatz ganz neuer, eigener Symbole frei, sofern die Bedeutung im Kontext erklärt ist. Der Baum mit Gattern nach DIN wird meist in horizontaler Form gezeichnet, während bei Verwendung der älteren Symbole der Baum vertikal dargestellt wird. Die in einem Fehlerbaum vorkommenden Ereignisse können in verschiedene Typen unterteilt werden und durch unterschiedliche Symbole dargestellt werden. Die verschiedenen Symbole sind in Tabelle 4.1 dargestellt. Wenn die Grundereignisse mit Eintrittswahrscheinlichkeiten versehen werden, kann die Eintrittswahrscheinlichkeit des Top-Ereignisses daraus berechnet werden.

4.2 Fehlerbaumerstellung

Die Fehlerbaumanalyse wird in der Regel von einem Einzelnen oder einer Gruppe von Analytikern durchgeführt, die genaue Kenntnisse über das zu untersuchende System besitzen. Diese Kenntnisse können von Schaltplänen, Funktionsbeschreibungen

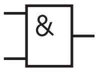
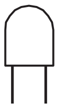
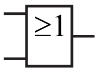
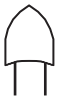
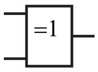
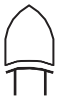
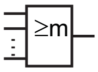
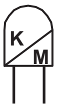
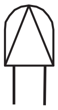
Symbol nach DIN IEC 61025	**Alte (gebräuchliche) Form**	**Beschreibung**
		UND-Gatter Beide Eingangsereignisse müssen eintreten, damit das Ausgangsereignis eintritt.
		ODER-Gatter Ein Eingangsereignis muss eintreten, damit das Ausgangsereignis eintritt.
		XOR-Gatter Genau ein Eingangsereignis muss eintreten, damit das Ausgangsereignis eintritt.
		K-von-M-Gatter K von M Eingangsereignisse müssen eintreten, damit das Ausgangsereignis eintritt.
—		Prioritäts-UND-Gatter Die Eingangsereignisse müssen von links nach rechts zeitlich nacheinander auftreten, damit das Ausgangsereignis eintritt.

Tabelle 4.1: Gatter-Symbole in der Fehlerbaumanalyse

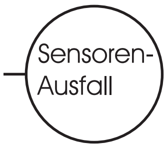
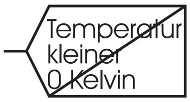
Symbol nach DIN	**Beschreibung**
	Zwischenereignis (teilw. auch als Symbol für Grundereignisse benutzt).
	Grundereignis - kann nicht weiter zerlegt werden.
	Ereignis, das nicht weiter untersucht wird, weil dazu keine Informationen vorliegen oder weil es als unbedeutend eingeschätzt wird.
	Ereignis, das in einer anderen Fehlerbaumanalyse untersucht wird.
	Ereignis, das niemals eintreten kann.
	Ereignis, das im normalen Betrieb gewöhnlich eintritt.
	Ereignis ist an anderer Stelle weiter untersucht. (Transfer-Out)
	Gegenstück zum Transfer-Out (Transfer-In). An dieser Stelle wird das Ereignis weiter untersucht.

Tabelle 4.2: Event-Symbole in der Fehlerbaumanalyse

und Gesprächen mit Fachleuten stammen. Es existieren einige Programme, die die Fehlerbaumanalyse unterstützen. Sie beschränken sich aber dabei meist auf die Verwaltung und Auswertung von Fehlerbäumen und Berechnung von Fehlerwahrscheinlichkeiten ([Dug00]). Es existieren aber Ansätze zur automatischen Fehlerbaumerstellung ([Bec83]), bei der ausgehend von Typbibliotheken für einzelne Komponenten wie Pumpen und Ventilen, Fehlerbäume für komplexere Systeme an Hand von Funktionsbeschreibungen synthetisiert werden können. Auch bei diesem Ansatz müssen Fehlerbäume von Analytikern manuell überprüft und ergänzt werden. In [MW90, Bec83, VGRH81, IEC93] werden die Vorgehensweisen für die Fehlerbaumerstellung beschrieben:

System-Informationen sammeln Im ersten Schritt müssen Informationen über das zu analysierende System zusammengetragen werden. Dazu gehören nach [IEC93] Informationen über

- das Entwicklungsziel,
- die Grenzen und Schnittstellen des Systems,
- mechanischen Aufbau,
- die Betriebsarten des Systems und deren Wirkungsweise und
- Umgebungsbedingungen (auch menschliche Einflüsse).

Insbesondere muss festgelegt werden, bis auf welche Ebene die Analyse erfolgen soll, d.h. wie detailliert das System beschrieben wird. Z.B. wird es meistens ausreichen, für einen Transistor nur zwischen Funktionieren und Nicht-Funktionieren zu unterscheiden, ohne die physikalischen Grundlagen von Transistoren zu betrachten. In dieser Arbeit wird davon ausgegangen, dass das zu untersuchende System bereits formal durch Phasenautomaten modelliert ist.

Auswahl von Top-Ereignissen Im nächsten Schritt müssen die unerwünschten Ereignisse definiert werden, die durch Fehlerbaumanalyse ausgeschlossen werden sollen. Dazu müssen möglichst *alle* unerwünschten Ereignisse betrachtet werden. Die Menge der Top-Ereignisse gehört damit zur Spezifikation des Systems. In der Regel werden zwei Arten von Fehlerereignissen unterschieden.

- **Operationale-Fehler** sind Fehler, die auftreten, wenn eine Komponente zwar für sich betrachtet korrekt funktioniert, aber die Funktion an einem falschen Ort oder zu einem falschen Zeitpunkt durch falsche Steuerung geschieht. Ein grünes Signal ist für sich noch kein Fehlerereignis, aber ein grünes Signal, dass die Einfahrt in einen kritischen Bereich erlaubt, in dem sich ein entgegenkommender Zug befindet, schon.

- **Komponenten-Fehler** sind Fehlfunktionen einer Komponente. Dabei wird noch zwischen *primären* und *sekundären* Fehlern unterschieden. Primäre Fehler treten unter Umweltbedingungen auf, für die die Komponente entwickelt worden ist, sekundäre Fehler unter Umweltbedingungen für die sie nicht entwickelt worden ist. Datenverlust auf Disketten, die in einer Diskettenbox auf dem Schreibtisch aufbewahrt werden ist ein primärer Fehler, während Datenverlust bei Aufbewahrung auf einem großen Magneten ein sekundärer Fehler ist.

Fehlerbaum-Konstruktion Für die Konstruktion des Fehlerbaums zu einem Top-Ereignis werden in [MW90] folgende Schritte vorgeschlagen:

1. Bestimme die direkten Ursachen des Top-Ereignisses.
2. Für jedes Ereignis bestimme die direkten Ursachen.
3. Wiederhole Schritt 2 bis alle Ereignisse bis zur vorgegebenen Betrachtungsebene beschrieben sind.

4.3 Formale Methoden für Fehlerbäume

4.3.1 Definition von Fehlerbäumen

Bevor die Semantik von Fehlerbäumen definiert werden kann, werden die Begriffe Ereignis, Gatter und Fehlerbaum formal definiert. Während in [Buc00] Fehlerbäume als gerichtete Graphen definiert werden, wird hier eine induktive Definition als Termstruktur verwendet, um auch die Semantik induktiv definieren zu können.

Definition 4.3.1 (Gatter). Für Gatter wird der Buchstabe G verwendet und folgende Gatter stehen zur Verfügung:

$$Gatter \stackrel{df}{=} \{\mathsf{AND}, \mathsf{OR}, \mathsf{AND\text{-}CC}, \mathsf{OR\text{-}CC}, \mathsf{AND\text{-}CC\text{-}Syn}, \mathsf{DIR}\}.$$

Neben den üblichen Gattern AND und OR, die keine zeitlichen Zusammenhänge zwischen den Eingangs- und Ausgangsereignissen beschreiben werden in [RST01, RST00] auch Gatter für Ursache-Wirkung- (CC für Cause-Consequence) und synchrone Ursache-Wirkung-Beziehungen (AND-CC-Syn) zur Verfügung gestellt, die in dieser Arbeit ebenfalls betrachtet werden. Das Gatter DIR wird in der graphischen Darstellung nur durch eine Verbindungslinie dargestellt und bedeutet nur eine Umformulierung des Ereignisses.

Definition 4.3.2 (Ereignis). Ein Ereignis E ist ein Paar

$$E = (Typ, A_E),$$

wobei $Typ \in \{Z, G, N\}$ angibt, ob es sich um ein Zwischen-, Grund- oder nicht weiter untersuchtes Ereignis handelt und A_E eine DCL-Formel ist, die das Ereignis beschreibt.

Definition 4.3.3 (Fehlerbaum). Fehlerbäume werden mit dem Buchstaben T (für tree) bezeichnet und die Menge $FaultTrees$ der Fehlerbäume wird induktiv definiert.

- Ereignisse der Form $E = (G, A_E)$ und $E = (N, A_E)$ sind Elemente der Menge $FaultTrees$, d.h. Grund- und nicht weiter untersuchte Ereignisse sind Fehlerbäume mit Top-Ereignis E.

- Für eine nicht-leere endliche Menge von Fehlerbäumen $\mathbb{T} \subseteq FaultTrees$, ein Ereignis $E = (Z, A_E)$ und ein Gatter G, ist auch $(E, G, \mathbb{T})$ ein Fehlerbaum mit Top-Ereignis E. Fehlerbäume können durch ein Gatter und ein Zwischenereignis zu neuen Fehlerbäumen kombiniert werden. Dabei erlaubt die Definition mehr als zwei Eingänge in Gatter, in der folgenden graphischen Darstellung wird der Einfachheit halber immer von ein oder zwei Eingängen ausgegangen. Es genügt eine Menge $\mathbb{T}$ von Fehlerbäumen anstatt einer Folge zu betrachten, da alle verwendeten Operationen assoziativ sein werden.

Außer den so definierten Elementen, gibt es keine weiteren Fehlerbäume.

4.3.2 Formale Semantik

In [RST00, RST01] wird die Semantik eines Fehlerbaums durch DC-Formeln beschrieben, die jedem Gatter zugeordnet werden. Diese an der Universität Augsburg entwickelte Semantik betrachtet zwei Aspekte: *Korrektheit* und *Vollständigkeit*.

Korrektheit (hinreichende Bedingungen) Die Idee besteht darin, dass die Unterereignisse tatsächlich zum Eintreten des Ereignisses beitragen, also keine überflüssigen Unterereignisse angegeben werden, die nichts mit dem Ereignis zu tun haben.

Vollständigkeit (notwendige Bedingungen) Entgegengesetzt ist eine Zerlegung vollständig, falls keine Unterereignisse im Fehlerbaum vergessen worden sind.

Hier werden die DC-Formeln der Augsburger Semantik übernommen und leicht verändert, um Model-Checking mit Phasenautomaten zu ermöglichen. Die Aspekte Korrektheit und Vollständigkeit werden hier für den ganzen Fehlerbaum durch zwei DCL-Formeln A_K und A_V beschrieben, die induktiv über den Aufbau von Fehlerbäumen definiert werden. Dabei ist die Vollständigkeit sehr viel wichtiger. Falls eine Zerlegung vollständig ist und die Unterereignisse ausgeschlossen werden können, kann auch das Ereignis selber ausgeschlossen werden. Für UND-Zerlegung reicht es in diesem Fall sogar aus, nur ein Unterereignis ausschließen zu können. In den Fallbeispielen wird sich außerdem zeigen, dass die Korrektheitsbedingung in vielen Fällen nicht erfüllt ist.

Definition 4.3.4 (Korrektheit und Vollständigkeit). Sei T ein Fehlerbaum. Dann sind die DCL-Formeln $A_K(T)$ und $A_V(T)$ wie folgt definiert.

1. Fall T ist ein Grund- oder nicht weiter untersuchtes Ereignis, d.h. $T = (Z, A)$ oder $T = (N, A)$. Dann ist

$$A_K(T) \overset{df}{=} \mathsf{true}$$

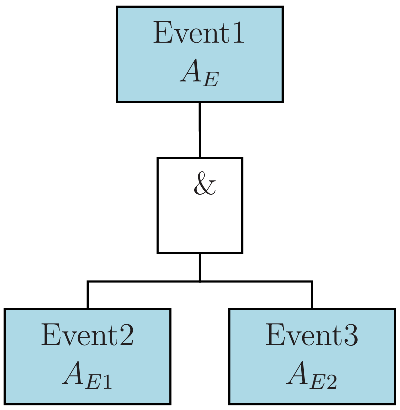
und

$$A_V(T) \overset{df}{=} \mathsf{true}.$$

Ein Fehlerbaum, der noch keine Zerlegungen enthält ist also in jedem Fall korrekt und vollständig.

2. Fall $T = (E, G, \mathbb{T})$ mit $E = (Z, A_E)$. Sei $\mathbb{T} = \{T_1, \dots, T_n\}$ und seien $A_{E1}, \dots, A_{En}$ die DCL-Formeln der Top-Ereignisse der untergeordneten Fehlerbäume $T_1, \dots, T_n$. Dann sind die Korrektheits- und Vollständigkeitsformeln in Abhängigkeit des Gattertyps definiert.

AND-Gatter



Intuitiv bedeutet eine Und-Zerlegung eines Ereignisses, dass alle Eingangsereignisse eintreten müssen, damit das Ausgangsereignis eintritt. Dabei wird keine zeitliche Relation zwischen den verschiedenen Ereignissen hergestellt.

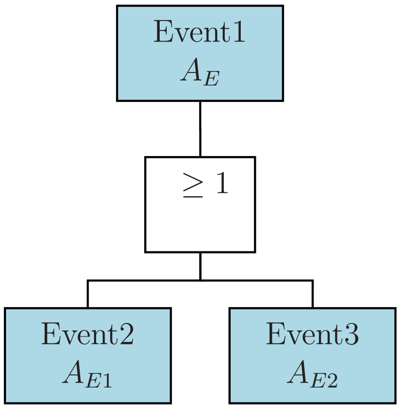
Korrektheit Eine Zerlegung eines Ereignisses mit Hilfe eines UND-Gatters in Unterereignisse ist *korrekt*, falls alle Unterereignisse gemeinsam in jedem Fall das Ereignis implizieren. Dies muss zusätzlich zu den Korrektheitbedingungen für die Teilbäume gezeigt werden, d.h.

$$A_K(T) \overset{df}{=} \underbrace{\left(\boxed{L}_0 (\bigwedge_{i=1}^{n} A_{Ei}) \Rightarrow \boxed{L}_0 A_E \right)}_{\text{zusätzliche Beweisverpflichtung}} \wedge \bigwedge_{i=1}^{n} A_K(T_i)$$

Vollständigkeit Die Zerlegung ist vollständig, falls das Ereignis nur dann auftritt, wenn auch die Unterereignisse eintreten, d.h.

$$A_V(T) \overset{df}{=} \underbrace{\left(\boxed{L}_0 A_E \Rightarrow (\bigwedge_{i=1}^{n} \boxed{L}_0 A_{Ei}) \right)}_{\text{zusätzliche Beweisverpflichtung}} \wedge \bigwedge_{i=1}^{n} A_V(T_i)$$

OR-Gatter



Entsprechend wie bei den AND-Gattern werden Korrektheits- und Vollständigkeitsformeln für OR-Gatter definiert.

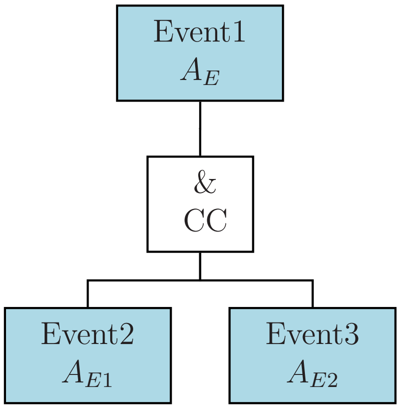
Korrektheit Eine Zerlegung eines Ereignisses mit Hilfe eines ODER-Gatters in Unterereignisse ist *korrekt*, falls ein Unterereignis in jedem Fall das Ereignis impliziert.

$$A_K(T) \stackrel{df}{=} \Big(\underbrace{(\bigvee_{i=1}^{n} \boxed{L}_0 A_{Ei}) \Rightarrow \boxed{L}_0 A_E}_{\text{zusätzliche Beweisverpflichtung}} \Big) \wedge \bigwedge_{i=1}^{n} A_K(T_i)$$

Vollständigkeit Die Zerlegung ist vollständig, falls das Ereignis nur dann auftritt, wenn mindestens ein Unterereignis eintritt.

$$A_V(T) \stackrel{df}{=} \Big(\underbrace{\boxed{L}_0 A_E \Rightarrow (\bigvee_{i=1}^{n} \boxed{L}_0 A_{Ei})}_{\text{zusätzliche. Beweisverpflichtung}} \Big) \wedge \bigwedge_{i=1}^{n} A_V(T_i)$$

AND-CC-Gatter



Durch Ursache-Wirkungs-UND-Gatter können zeitliche Beziehungen zwischen den Eingangsereignissen und den Ausgangsereignissen beschrieben werden.

Korrektheit Für die Korrektheit dieser Beziehung ist folgendes wichtig: Falls die Eingangsereignisse $E1, \ldots, En$ irgendwann eintreten, dann tritt auch das Ausgangsereignis E irgendwann ein. Da nur Korrektheit betrachtet wird, muss das Ausgangsereignis nicht *nach* den Eingangsereignissen eintreten, da das Ausgangsereignis auch durch andere Ursachen ausgelöst worden sein kann. Entscheidend ist nur, dass das Ausgangsereignis überhaupt eintritt.

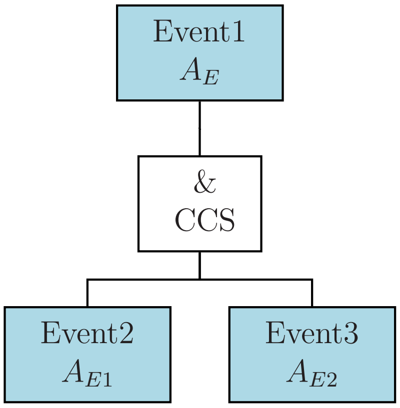
$$A_K(T) \stackrel{df}{=} \Big(\underbrace{(\bigwedge_{i=1}^{n} \Diamond_L A_{Ei}) \Rightarrow \Diamond_L A_E}_{\text{zusätzliche Beweisverpflichtung}} \Big) \wedge \bigwedge_{i=1}^{n} A_K(T_i)$$

Vollständigkeit Die Zerlegung ist vollständig, falls das Ereignis nur dann auftritt, wenn vorher alle Untereignisse eingetreten sind.

$$A_V(T) \overset{df}{=} \underbrace{\big(\, \boxed{L}_0 \neg(\neg(\bigwedge_{i=1}^{n} \Diamond A_{Ei}); A_E)\,\big)}_{\text{zusätzliche Beweisverpflichtung}} \wedge \bigwedge_{i=1}^{n} A_V(T_i)$$

AND-CC-Syn-Gatter

Das synchrone Ursache-Wirkungs-UND-Gatter stellt auch eine Beziehung zwischen den Eingangsereignissen selbst her, indem es fordert, dass die Eingangsereignisse $E_1, \ldots, E_n$ gleichzeitig eintreten müssen, damit das Ausgangsereignis eintritt.



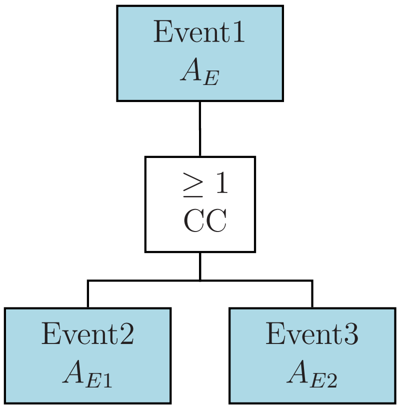
Korrektheit Die Korrektheitsformel für AND-CC-Syn-Gatter fordert zusätzlich zu denen für AND-CC-Gatter, das gleichzeitige Auftreten der Eingangsereignisse.

$$A_K(T) \overset{df}{=} \underbrace{\big(\, (\Diamond_{L}(\bigwedge_{i=1}^{n} A_{Ei})) \Rightarrow \Diamond_{L} A_E \,\big)}_{\text{zusätzliche Beweisverpflichtung}} \wedge \bigwedge_{i=1}^{n} A_K(T_i)$$

Vollständigkeit Die Zerlegung ist vollständig, falls das Ereignis nur dann auftritt, wenn vorher alle Untereignisse gleichzeitig eingetreten sind.

$$A_V(T) \overset{df}{=} \underbrace{\big(\, \boxed{L}_0 \neg(\neg\Diamond(\bigwedge_{i=1}^{n} A_{Ei}); A_E)\,\big)}_{\text{zusätzliche Beweisverpflichtung}} \wedge \bigwedge_{i=1}^{n} A_V(T_i)$$

OR-CC-Gatter



Das asynchrone Ursache-Wirkungs-ODER-Gatter ist das Pendant zum AND-CC-Gatter. Eine synchrone Version wie bei den UND-Gattern kann es nicht geben.

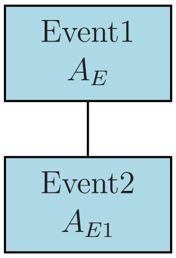
Korrektheit

$$A_K(T) \overset{df}{=} \underbrace{\big(\, (\bigvee_{i=1}^{n} \Diamond_{L} A_{Ei}) \Rightarrow \Diamond_{L} A_E \,\big)}_{\text{zusätzliche Beweisverpflichtung}} \wedge \bigwedge_{i=1}^{n} A_K(T_i)$$

Vollständigkeit Die Zerlegung ist vollständig, falls das Ereignis nur dann auftritt, wenn vorher mindestens ein Untereregnis eingetreten sind.

$$A_V(T) \overset{df}{=} \underbrace{\big(\, \boxed{L}_0 \neg(\neg(\bigvee_{i=1}^{n} \Diamond A_{Ei}); A_E)\,\big)}_{\text{zusätzliche Beweisverpflichtung}} \wedge \bigwedge_{i=1}^{n} A_V(T_i)$$

DIR-Gatter



Auch für ein "Umschreiben" eines Ereignisses, muss eine formale Korrektheits- und Vollständigkeitsbedingung angegeben werden.

Korrektheit

$$A_K(T) \overset{df}{=} \big(\underbrace{\boxed{L}_0 A_{E1} \Rightarrow \boxed{L}_0 A_E}_{\text{zusätzliche Beweisverpflichtung}} \big) \wedge A_K(T_1)$$

Vollständigkeit

$$A_V(T) \overset{df}{=} \big(\underbrace{\boxed{L}_0 A_E \Rightarrow \boxed{L}_0 A_{E1}}_{\text{zusätzliche Beweisverpflichtung}} \big) \wedge A_V(T_1)$$

Bemerkung 4.3.1. Der $\boxed{L}_0$-Operator wird jeweils eingefügt, um Model-Checking mit Phasenautomaten zu ermöglichen. Falls die den Ereignissen zugeordneten Formeln die Form

$$\Diamond_L A$$

besitzten, spielt der $\boxed{L}_0$-Operator keine Rolle. Denn es gilt nach Lemma 2.2.15 (1) und (2) für DCL-Formeln A, A_1, A_2 folgende Beziehung:

$$\mathcal{I}, \mathcal{V}, [0,0] \models_{DC} \boxed{L}_0 \Diamond_L A \iff \mathcal{I}, \mathcal{V} \models_{DC} \Diamond_L A$$

und damit nach Definition des $\boxed{L}_0$-Operators auch

$$\models_{DC} \Big((\boxed{L}_0 \Diamond_L A_1) \Rightarrow (\boxed{L}_0 \Diamond_L A_2) \Big) \iff \Big((\boxed{L}_0 \Diamond_L A_1) \Rightarrow (\boxed{L}_0 \Diamond_L A_2) \Big)$$

Modellierung des Systems

Damit formale Aussagen über ein System getroffen werden können, muss das System formal modelliert sein. Die Modellierung kann durch eine DCL-Formel $A_{\mathcal{M}}$ oder äquivalent durch einen Phasenautomaten $\mathcal{M}$ mit $\models F(\mathcal{M}) \iff [A_{\mathcal{M}}]$ erfolgen.

Definition 4.3.5 (Semantik von Fehlerbäumen). Mit der Definition der Korrektheits- und Vollständigkeitsformeln kann auch die Korrektheits- und Vollständigkeitssemantik von Fehlerbäumen definiert werden. Die Menge der DCL-Formeln, die ein System modellieren, für die ein Fehlerbaum korrekt ist, wird als Korrektheitssemantik $[\![\cdot]\!]_K$, die Menge für die ein Fehlerbaum vollständig ist als Vollständigkeitssemantik $[\![\cdot]\!]_V$ und die Menge für die ein Fehlerbaum vollständig und korrekt ist als Semantik $[\![\cdot]\!]$ bezeichnet.

$$[\![T]\!]_K \stackrel{df}{=} \{A_{\mathcal{M}} | \models_{DC} \boxed{L}_0 A_{\mathcal{M}} \Rightarrow A_K(T)\}$$
$$[\![T]\!]_V \stackrel{df}{=} \{A_{\mathcal{M}} | \models_{DC} \boxed{L}_0 A_{\mathcal{M}} \Rightarrow A_V(T)\}$$
$$[\![T]\!] \stackrel{df}{=} \{A_{\mathcal{M}} | \models_{DC} \boxed{L}_0 A_{\mathcal{M}} \Rightarrow (A_K(T) \wedge A_V(T))\}$$

Die Vollständigkeitssemantik ist dabei besonders für die Gefährdungsanalyse interessant, da dabei untersucht wird, wie eine bestimmte Störung ausgeschlossen werden kann. Dafür ist es wichtig, dass keine Ereignisse vergessen worden sind. Für die Folgenanalyse ist die Korrektheitssemantik interessant, da hier untersucht wird, welche Ereignisse zusammenkommen müssen, damit ein Ereignis auch tatsächlich eintritt (vgl. Abschnitt 1.2).

4.4 Model-Checking mit Moby/DC

Mit Hilfe des Tools Moby/DC soll für eine Modellierung und einen Fehlerbaum überprüft werden, ob der Fehlerbaum im Sinne von Definition 4.3.5 korrekt und vollständig ist. Das ist genau dann der Fall, wenn für jedes vorkommende Gatter die zusätzliche Beweisverpflichtung für eine vorgegebene Modellierung bewiesen werden kann, also wenn die Beweisverpflichtung eine Folgerung von $\boxed{L}_0 A_{\mathcal{M}}$ ist.

4.4.1 Model-Checking für Gatter ohne Ursache-Wirkung-Beziehung

Um AND- und OR-Gatter ohne Ursache-Wirkung-Beziehung mit Moby/DC zu modelchecken, muss nur vorausgesetzt werden, dass zu den DCL-Formeln entsprechende Phasenautomaten und deren Komplemente existieren. Es ist dabei möglich, dass bei der Konstruktion der Phasenautomaten Uhrintervalle der Form $(0, \infty)$ verwendet werden. Diese Intervalle sind nach Definition äquivalent zu Intervallen $(0, x]$, wobei x eine frische sonst nicht verwendete Variable ist. Dadurch kommen in den Phasenautomaten Variablen vor, über die in den zugrunde liegenden DCL-Formeln keine Aussagen gemacht werden, so dass eine Beziehung der Form

$$\models F(\mathcal{A}_A) \iff [A] \qquad (4.4.4.1)$$

in diesen Fällen nicht gezeigt werden kann. Aus diesem Grund wird nur die folgende schwächere Bedingung gefordert.

Voraussetzungen Seien $A_{E1}, \ldots, A_{En}$ und A_E die zu einem Gatter gehörenden DCL-Formeln. Dazu seien $\mathcal{A}_{A_{E1}}, \ldots, \mathcal{A}_{A_{En}}$ und $\mathcal{A}_{A_E}$ Phasenautomaten. Zu jedem dieser Phasenautomaten $\mathcal{A}_x$ sei V_x die Menge der Variablen, die nur in diesem Phasenautomaten verwendet werden, also z.B. durch Intervalle der Form $(0, \infty)$ entstehen. Vorausgesetzt werden dann folgenden Bedingungen für die Phasenautomaten:

$$(\mathcal{I}, \mathcal{V}) \models [A_x] \quad \text{gdw.} \quad \exists\, \mathcal{V}' : \mathcal{V}' =_{\setminus V_x} \mathcal{V} \text{ und } (\mathcal{I}, \mathcal{V}') \models F(\mathcal{A}_x) \tag{4.4.4.2}$$

Außerdem existieren Komplementautomaten $\overline{\mathcal{A}_{A_{E1}}}, \ldots, \overline{\mathcal{A}_{A_{En}}}, \overline{\mathcal{A}_{A_E}}$ mit

$$(\mathcal{I}, \mathcal{V}) \models \neg[A_x] \quad \text{gdw.} \quad \exists\, \mathcal{V}' : \mathcal{V}' =_{\setminus V_x} \mathcal{V} \text{ und } (\mathcal{I}, \mathcal{V}') \models F(\overline{\mathcal{A}_x}). \tag{4.4.4.3}$$

Falls eine Beziehung wie in (4.4.4.1) gezeigt werden kann und das Komplement wie in Abschnitt 3.4.2 beschrieben, konstruiert werden kann, sind die Bedingungen (4.4.4.2) und (4.4.4.3) ebenfalls erfüllt.

UND-Gatter

Korrektheitsbedingung Die Bedingung $\models_{DC} \boxed{L}_0 A_{\mathcal{M}} \Rightarrow \big((\bigwedge_{i=1}^{n} \boxed{L}_0 A_{Ei}) \Rightarrow \boxed{L}_0 A_E\big)$ kann dann als Leerheitstest von $\mathcal{M} \| \mathcal{A}_{A_{E1}} \| \ldots \| \mathcal{A}_{A_{En}} \| \overline{\mathcal{A}_{A_E}}$ überprüft werden:
Dazu wird zuerst gezeigt, dass die u.U. durch Intervalle $(0, \infty)$ neu eingeführten Variablen keine Auswirkungen auf das Model-Checking haben. Es gilt:

$$\begin{aligned} &\exists(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models F(\mathcal{M}) \wedge F(\mathcal{A}_{A_{E1}}) \wedge \ldots \wedge F(\mathcal{A}_{A_{En}}) \wedge F(\overline{\mathcal{A}_{A_E}}) \\ \Longleftrightarrow\ &\exists(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models [A_M] \wedge [A_{E1}] \wedge \ldots \wedge [A_{En}] \wedge \neg[A_E] \end{aligned} \tag{4.4.4.4}$$

Beweis „⇒“-Richtung

$$\begin{aligned} & (\mathcal{I}, \mathcal{V}) \models F(\mathcal{M}) \wedge F(\mathcal{A}_{A_{E1}}) \wedge \ldots \wedge F(\mathcal{A}_{A_{En}}) \wedge F(\overline{\mathcal{A}_{A_E}}) \\ \{(4.4.4.2)(4.4.4.3)\} \Rightarrow\quad & (\mathcal{I}, \mathcal{V}) \models [A_M] \wedge [A_{E1}] \wedge \ldots \wedge [A_{En}] \wedge \neg[A_E] \end{aligned}$$

„⇐“-Richtung Gelte $(\mathcal{I}, \mathcal{V}) \models [A_M] \wedge [A_{E1}] \wedge \ldots \wedge [A_{En}] \wedge \neg[A_E]$. Dann existieren nach Voraussetzung (4.4.4.2) und (4.4.4.3) Belegungen $\mathcal{V}_{\mathcal{M}}, \mathcal{V}_1, \ldots \mathcal{V}_n, \mathcal{V}_E$, so dass gilt

$$\begin{aligned} (\mathcal{I}, \mathcal{V}_{\mathcal{M}}) &\models F(\mathcal{M}) \\ (\mathcal{I}, \mathcal{V}_1) &\models F(\mathcal{A}_{A_{E1}}) \\ &\ldots \\ (\mathcal{I}, \mathcal{V}_n) &\models F(\mathcal{A}_{A_{E1}}) \\ (\mathcal{I}, \mathcal{V}_E) &\models F(\overline{\mathcal{A}_{A_E}}) \end{aligned}$$

und die sich nur in den Werten der Variablen unterscheiden, die nur in den zugehörigen Phasenautomaten vorkommen. Damit kann eine Belegung $\mathcal{V}'$ wie folgt konstruiert werden:

$$\mathcal{V}'(x) \stackrel{df}{=} \begin{cases} \mathcal{V}_{\mathcal{M}}(x) & \text{falls } x \in V_{\mathcal{M}} \\ \mathcal{V}_1(x) & \text{falls } x \in V_1 \\ \dots & \\ \mathcal{V}_n(x) & \text{falls } x \in V_n \\ \mathcal{V}_E(x) & \text{falls } x \in V_E \\ \mathcal{V}(x) & \text{sonst} \end{cases}$$

Für das Paar $(\mathcal{I}, \mathcal{V}')$ gilt dann:

$$(\mathcal{I}, \mathcal{V}') \models F(\mathcal{M}) \wedge F(\mathcal{A}_{A_{E1}}) \wedge \ldots \wedge F(\mathcal{A}_{A_{En}}) \wedge F(\overline{\mathcal{A}_{A_E}})$$

$\square$

Damit kann die eigentliche Behauptung bewiesen werden.

Beweis

$$\begin{array}{rcl}
\left\{\begin{array}{r}\text{Mit M}\textsc{oby/dc}\\ \text{überprüfbar}\end{array}\right\} & & \models \neg F(\mathcal{M} \| \mathcal{A}_{A_{E1}} \| \ldots \| \mathcal{A}_{A_{En}} \| \overline{\mathcal{A}_{A_E}}) \\
\{\text{Def. Gültigkeit}\} & \iff & \neg\exists(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models F(\mathcal{M} \| \mathcal{A}_{A_{E1}} \| \ldots \| \mathcal{A}_{A_{En}} \| \overline{\mathcal{A}_{A_E}}) \\
\{\text{Def. } \|\} & \iff & \neg\exists(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models F(\mathcal{M}) \wedge F(\mathcal{A}_{A_{E1}}) \wedge \ldots \wedge F(\mathcal{A}_{A_{En}}) \\
& & \qquad \wedge F(\overline{\mathcal{A}_{A_E}}) \\
\{(4.4.4.4)\} & \iff & \neg\exists(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models [A_M] \wedge [A_{E1}] \wedge \ldots \wedge [A_{En}] \wedge \neg[\mathcal{A}_{A_E}] \\
& \iff & \forall(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models \neg\big([A_M] \wedge [A_{E1}] \wedge \ldots \wedge [A_{En}] \wedge \neg[\mathcal{A}_{A_E}]\big) \\
& \iff & \forall(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models \neg[A_M] \vee \neg\big([A_{E1}] \wedge \ldots \wedge [A_{En}] \wedge \neg[\mathcal{A}_{A_E}]\big) \\
& \iff & \forall(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models [A_M] \Rightarrow \neg\big([A_{E1}] \wedge \ldots \wedge [A_{En}] \wedge \neg[\mathcal{A}_{A_E}]\big) \\
& \iff & \forall(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models [A_M] \Rightarrow \big(([A_{E1}] \wedge \ldots \wedge [A_{En}]) \Rightarrow [\mathcal{A}_{A_E}]\big) \\
\{\text{Lemma 2.2.15 (1)}\} & \iff & \forall(\mathcal{I}, \mathcal{V}) : \\
& & (\mathcal{I}, \mathcal{V}),\ [0,0] \models_{DC} \big(\boxed{L}_0 A_{\mathcal{M}} \wedge \boxed{L}_0 A_{E1} \wedge \ldots \wedge \boxed{L}_0 A_{En}\big) \\
& & \qquad \Rightarrow \boxed{L}_0 A_{En}.
\end{array}$$

$$
\begin{array}{rrl}
\{\text{Für } t > 0 \text{ ist}\} & & \forall(\mathcal{I},\mathcal{V}), \forall t : \\
\{\text{die Voraus-}\} & \Longleftrightarrow & (\mathcal{I},\mathcal{V}),\ [0,t] \models_{DC} \left(\boxed{L}_0 A_{\mathcal{M}} \wedge \boxed{L}_0 A_{E1} \wedge \ldots \wedge \boxed{L}_0 A_{En}\right) \\
\{\text{setzung nach }\} & & \qquad \Rightarrow \boxed{L}_0 A_{En} \\
\{\text{Def. } \boxed{L}_0 \text{ falsch}\} & & \\
 & \Longleftrightarrow & \models_{DC} \boxed{L}_0 A_{\mathcal{M}} \Rightarrow ((\boxed{L}_0 A_{E1} \wedge \ldots \wedge \boxed{L}_0 A_{En}) \\
 & & \qquad \Rightarrow \boxed{L}_0 A_E)
\end{array}
$$

□

Vollständigkeitsbedingung Die Bedingung $\models_{DC} \boxed{L}_0 A_{\mathcal{M}} \Rightarrow \left((\boxed{L}_0 A_E \Rightarrow (\bigwedge_{i=1}^{n} \boxed{L}_0 A_{Ei})\right)$ kann als Leerheitstest von $\mathcal{M} \| (\overline{\mathcal{A}_{A_{E1}}} \vee \ldots \vee \overline{\mathcal{A}_{A_{En}}}) \| \mathcal{A}_{A_E}$ überprüft werden:
Dazu wird wie im Korrektheitsfall eine Hilfsaussage bewiesen:

$$
\begin{aligned}
& \exists(\mathcal{I},\mathcal{V}) : (\mathcal{I},\mathcal{V}) \models F(\mathcal{M}) \wedge \left(F(\overline{\mathcal{A}_{A_{E1}}}) \vee \ldots \vee F(\overline{\mathcal{A}_{A_{En}}})\right) \wedge F(\mathcal{A}_{A_E}) \\
\Longleftrightarrow\ & \exists(\mathcal{I},\mathcal{V}) : (\mathcal{I},\mathcal{V}) \models [A_{\mathcal{M}}] \wedge \left(\neg[A_{E1}] \vee \ldots \vee \neg[A_{En}]\right) \wedge [A_E] \qquad (4.4.4.5)
\end{aligned}
$$

Beweis „⇒“-Richtung

$$
\begin{array}{rl}
 & (\mathcal{I},\mathcal{V}) \models F(\mathcal{M}) \wedge \left(F(\overline{\mathcal{A}_{A_{E1}}}) \vee \ldots \vee F(\overline{\mathcal{A}_{A_{En}}})\right) \wedge F(\mathcal{A}_{A_E}) \\
\{(4.4.4.2)(4.4.4.3)\} \Rightarrow & (\mathcal{I},\mathcal{V}) \models [A_{\mathcal{M}}] \wedge \left(\neg[A_{E1}] \vee \ldots \vee \neg[A_{En}]\right) \wedge [A_E]
\end{array}
$$

„⇐“-Richtung Gelte $(\mathcal{I},\mathcal{V}) \models [A_{\mathcal{M}}] \wedge \left(\neg[A_{E1}] \vee \ldots \vee \neg[A_{En}]\right) \wedge [A_E]$. Dann erfüllt $(\mathcal{I},\mathcal{V})$ die Formeln $[A_{\mathcal{M}}]$, $[A_E]$ und mindestens eine der Formeln $\neg[A_{E1}] \ldots \neg[A_{En}]$. O.B.d.A sei $[A_{E1}]$ die erfüllte Formel. Dann gibt es nach Voraussetzung (4.4.4.2) und (4.4.4.3) Belegungen $\mathcal{V}_{\mathcal{M}}, \mathcal{V}_1, \mathcal{V}_E$, so dass gilt

$$
\begin{aligned}
(\mathcal{I},\mathcal{V}_{\mathcal{M}}) &\models F(\mathcal{M}) \\
(\mathcal{I},\mathcal{V}_1) &\models F(\overline{\mathcal{A}_{A_{E1}}}) \\
(\mathcal{I},\mathcal{V}_E) &\models F(\mathcal{A}_{A_E})
\end{aligned}
$$

und die sich nur in den Werten der Variablen unterscheiden, die nur in den zugehörigen Phasenautomaten vorkommen. Damit kann eine Belegung $\mathcal{V}'$ wie folgt konstruiert werden:

$$
\mathcal{V}'(x) \overset{df}{=} \begin{cases} \mathcal{V}_{\mathcal{M}}(x) & \text{falls } x \in V_{\mathcal{M}} \\ \mathcal{V}_1(x) & \text{falls } x \in V_1 \\ \mathcal{V}_E(x) & \text{falls } x \in V_E \\ \mathcal{V}(x) & \text{sonst} \end{cases}
$$

Für das Paar $(\mathcal{I}, \mathcal{V}')$ gilt dann:

$$(\mathcal{I}, \mathcal{V}') \models F(\mathcal{M}) \wedge F(\overline{\mathcal{A}_{A_{E1}}}) \wedge F(\mathcal{A}_{A_E})$$

und damit auch

$$(\mathcal{I}, \mathcal{V}') \models F(\mathcal{M}) \wedge \Big(F(\overline{\mathcal{A}_{A_{E1}}}) \vee \ldots \vee F(\overline{\mathcal{A}_{A_{En}}})\Big) \wedge F(\mathcal{A}_{A_E})$$

□

Mit diesem Hilfssatz wird wie im Korrektheitsfall die eigentliche Behauptung bewiesen:

Beweis

$$\begin{array}{rcl}
\left\{\begin{array}{r}\text{Mit MOBY/DC}\\ \text{überprüfbar}\end{array}\right\} & & \models \neg F(\mathcal{M} \| (\overline{\mathcal{A}_{A_{E1}}} \vee \ldots \vee \overline{\mathcal{A}_{A_{En}}}) \| \mathcal{A}_{A_E}) \\
\{\text{Def. Gültigkeit}\} & \iff & \neg\exists(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models F(\mathcal{M} \| (\overline{\mathcal{A}_{A_{E1}}} \vee \ldots \vee \overline{\mathcal{A}_{A_{En}}}) \| \mathcal{A}_{A_E}) \\
\{\text{Def. } \|\} & \iff & \neg\exists(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models F(\mathcal{M}) \wedge F(\overline{\mathcal{A}_{A_{E1}}} \vee \ldots \vee \overline{\mathcal{A}_{A_{En}}}) \wedge F(\mathcal{A}_{A_E}) \\
\{\text{Satz 3.2.1}\} & \iff & \neg\exists(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models F(\mathcal{M}) \wedge \big(F(\overline{\mathcal{A}_{A_{E1}}}) \vee \ldots \vee F(\overline{\mathcal{A}_{A_{En}}})\big) \\
& & \wedge\, F(\mathcal{A}_{A_E}) \\
\{(4.4.4.5)\} & \iff & \neg\exists(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models [A_{\mathcal{M}}] \wedge \big(\neg[A_{E1}] \vee \ldots \vee \neg[A_{En}]\big) \wedge [A_E] \\
& \iff & \forall(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models \neg[A_{\mathcal{M}}] \vee \neg\big(\neg[A_{E1}] \vee \ldots \vee \neg[A_{En}]\big) \vee \\
& & \neg[A_E] \\
& \iff & \forall(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models [A_{\mathcal{M}}] \Rightarrow \big([A_E] \Rightarrow ([A_{E1}] \wedge \ldots \wedge [A_{En}])\big) \\
\left\{\begin{array}{r}\text{Analog zur}\\ \text{Korrektheit}\end{array}\right\} & \iff & \models_{DC} \boxed{L}_0 A_{\mathcal{M}} \Rightarrow \\
& & \Big(\boxed{L}_0 A_E \Rightarrow (\boxed{L}_0 A_{E1} \wedge \ldots \wedge \boxed{L}_0 A_{En})\Big)
\end{array}$$

□

OR-Gatter

Korrektheitsbedingung Die Bedingung $\models_{DC} A_{\mathcal{M}} \Rightarrow \big((\bigvee_{i=1}^{n} \boxed{L}_0 A_{Ei}) \Rightarrow \boxed{L}_0 A_E\big)$ kann als Leerheitstest von $\mathcal{M} \| \big(\mathcal{A}_{A_{E1}} \vee \ldots \vee \mathcal{A}_{A_{En}}\big) \| \overline{\mathcal{A}_{A_E}}$ überprüft werden.
Der benutzte Hilfssatz lautet:

$$\begin{aligned}
&\exists(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models F(\mathcal{M}) \wedge \Big(F(\mathcal{A}_{A_{E1}}) \vee \ldots \vee F(\mathcal{A}_{A_{En}})\Big) \wedge F(\overline{\mathcal{A}_{A_E}}) \\
\iff &\exists(\mathcal{I}, \mathcal{V}) : (\mathcal{I}, \mathcal{V}) \models [A_{\mathcal{M}}] \wedge \big([A_{E1}] \vee \ldots \vee [A_{En}]\big) \wedge \neg[A_E] \qquad (4.4.4.6)
\end{aligned}$$

und kann analog zum Hilfssatz für den Beweis des Vollständigkeitstests für AND-Gatter bewiesen werden.
Die eigentliche Aussage kann damit wie folgt bewiesen werden:

Beweis

$$\begin{aligned}
&\left\{\begin{array}{r}\text{Mit MOBY/DC}\\ \text{überprüfbar}\end{array}\right\} && \models \neg F(\mathcal{M} \| (\mathcal{A}_{A_{E1}} \vee \ldots \vee \mathcal{A}_{A_{En}}) \| \overline{\mathcal{A}_{A_E}})\\
&\{\text{Def. Gültigkeit}\} \iff && \neg\exists(\mathcal{I},\mathcal{V}) : (\mathcal{I},\mathcal{V}) \models F(\mathcal{M} \| (\mathcal{A}_{A_{E1}} \vee \ldots \vee \mathcal{A}_{A_{En}}) \| \overline{\mathcal{A}_{A_E}})\\
&\{\text{Def. } \|\} \iff && \neg\exists(\mathcal{I},\mathcal{V}) : (\mathcal{I},\mathcal{V}) \models F(\mathcal{M}) \wedge F(\mathcal{A}_{A_{E1}} \vee \ldots \vee \mathcal{A}_{A_{En}}) \wedge F(\overline{\mathcal{A}_{A_E}})\\
&\{\text{Satz 3.2.1}\} \iff && \neg\exists(\mathcal{I},\mathcal{V}) : (\mathcal{I},\mathcal{V}) \models F(\mathcal{M}) \wedge \big(F(\mathcal{A}_{A_{E1}}) \vee \ldots \vee F(\mathcal{A}_{A_{En}})\big)\\
& && \qquad \wedge F(\overline{\mathcal{A}_{A_E}})\\
&\{(4.4.4.6)\} \iff && \neg\exists(\mathcal{I},\mathcal{V}) : (\mathcal{I},\mathcal{V}) \models [A_{\mathcal{M}}] \wedge \big([A_{E1}] \vee \ldots \vee [A_{En}]\big) \wedge \neg[A_E]\\
&\iff && \forall(\mathcal{I},\mathcal{V}) : (\mathcal{I},\mathcal{V}) \models \neg[A_{\mathcal{M}}] \vee \neg\big([A_{E1}] \vee \ldots \vee [A_{En}]\big) \vee\\
& && \qquad [A_E]\\
&\iff && \forall(\mathcal{I},\mathcal{V}) : (\mathcal{I},\mathcal{V}) \models [A_{\mathcal{M}}] \Rightarrow \big(([A_{E1}] \vee \ldots \vee [A_{En}]) \Rightarrow [A_E]\big)\\
&\left\{\begin{array}{r}\text{Analog zum}\\ \text{Und-Gatter}\end{array}\right\} \iff && \models_{DC} \boxed{L}_0 A_{\mathcal{M}} \Rightarrow\\
& && \left((\boxed{L}_0 A_{E1} \vee \ldots \vee \boxed{L}_0 A_{En}) \Rightarrow \boxed{L}_0 A_E\right)
\end{aligned}$$

$\square$

Vollständigkeitsbedingung Die Bedingung $\models_{DC} \boxed{L}_0 A_{\mathcal{M}} \Rightarrow \big(\boxed{L}_0 A_E \Rightarrow (\bigvee_{i=1}^{n} \boxed{L}_0 A_{Ei})\big)$ kann als Leerheitstest von $\mathcal{M} \| (\overline{\mathcal{A}_{A_{E1}}} \| \ldots \| \overline{\mathcal{A}_{A_{En}}}) \| \mathcal{A}_{A_E}$ überprüft werden.
Als Hilfssatz wird

$$\begin{aligned}
&\exists(\mathcal{I},\mathcal{V}) : (\mathcal{I},\mathcal{V}) \models F(\mathcal{M}) \wedge F(\overline{\mathcal{A}_{A_{E1}}}) \wedge \ldots \wedge F(\overline{\mathcal{A}_{A_{En}}}) \wedge F(\mathcal{A}_{A_E})\\
\iff &\exists(\mathcal{I},\mathcal{V}) : (\mathcal{I},\mathcal{V}) \models [A_M] \wedge \neg[A_{E1}] \wedge \ldots \wedge \neg[A_{En}] \wedge [A_E] \qquad (4.4.4.7)
\end{aligned}$$

verwendet, der Analog zu (4.4.4.4) bewiesen wird. Damit kann die Aussage wie folgt bewiesen werden:

Beweis

$$
\begin{array}{rcl}
\left\{\begin{array}{r}\text{Mit MOBY/DC}\\ \text{überprüfbar}\end{array}\right\} & & \models \neg F(\mathcal{M}\|(\overline{\mathcal{A}_{A_{E1}}}\|\ldots\|\overline{\mathcal{A}_{A_{En}}})\|\mathcal{A}_{A_E})\\
\{\text{Def. Gültigkeit}\} & \iff & \neg\exists(\mathcal{I},\mathcal{V}):(\mathcal{I},\mathcal{V})\models F(\mathcal{M}\|(\overline{\mathcal{A}_{A_{E1}}}\|\ldots\|\overline{\mathcal{A}_{A_{En}}})\|\mathcal{A}_{A_E})\\
\{\text{Def. }\|\} & \iff & \neg\exists(\mathcal{I},\mathcal{V}):(\mathcal{I},\mathcal{V})\models F(\mathcal{M})\wedge F(\overline{\mathcal{A}_{A_{E1}}})\wedge\ldots\wedge F(\overline{\mathcal{A}_{A_{En}}})\\
 & & \quad\wedge F(\mathcal{A}_{A_E})\\
\{(4.4.4.7)\} & \iff & \neg\exists(\mathcal{I},\mathcal{V}):(\mathcal{I},\mathcal{V})\models [A_{\mathcal{M}}]\wedge\neg[A_{E1}]\wedge\ldots\wedge\neg[A_{En}]\big)\wedge[A_E]\\
 & \iff & \forall(\mathcal{I},\mathcal{V}):(\mathcal{I},\mathcal{V})\models \neg[A_{\mathcal{M}}]\vee[A_{E1}]\vee\ldots\vee[A_{En}]\vee\\
 & & \quad\neg[A_E]\\
 & \iff & \forall(\mathcal{I},\mathcal{V}):(\mathcal{I},\mathcal{V})\models [A_{\mathcal{M}}]\Rightarrow\big([A_E]\Rightarrow([A_{E1}]\vee\ldots\vee[A_{En}])\big)\\
\left\{\begin{array}{r}\text{Analog zum}\\ \text{Und-Gatter}\end{array}\right\} & \iff & \models_{DC}\boxed{L}_0 A_{\mathcal{M}}\Rightarrow\\
 & & \Big(\boxed{L}_0 A_E\Rightarrow(\boxed{L}_0 A_{E1}\vee\ldots\vee\boxed{L}_0 A_{En})\Big)
\end{array}
$$

□

4.4.2 Pattern für Ereignisse

Die möglichen Formeln, die zur Beschreibung eines Ereignisses in einem Fehlerbaum dienen, müssen für die Gatter ohne zeitliche Beziehung nur in einen äquivalenten und komplementierbaren Phasenautomaten umgewandelt werden können. Für die meisten Fehlerbäume reichen aber schon wenige Formelpattern aus, um vorkommende Ereignisse zu beschreiben. Nach [Gór94] und [Bec83] kommen folgende Ereignisse vor:

- Das System nimmt einen bestimmten Zustand an.
- Das System nimmt einen Zustand an und dieser Zustand wird nicht mehr verlassen, wenn er einmal angenommen wurde.
- Eine Sequenz von Zuständen wird durchlaufen.

Außerdem kann es für einen Zustand wichtig sein, wie lange dieser Zustand angenommen wird. Zum Beispiel kann "Die Pumpe arbeitet länger als 60 Zeiteinheiten" ein mögliches Ereignis in einem Fehlerbaum sein. Mit Zustandsausdrücken wird der Zustand eines Systems beschreiben. Somit können die Ereignisse durch folgende DCL-Formeln beschrieben werden.

- $\diamond_L(\lceil\pi\rceil\wedge b\leq\ell\leq e)$ (Erreichbarer Zustand)

- $\Diamond \boxed{L} \lceil \pi \rceil \wedge \neg \Diamond (\lceil \pi \rceil ; \lceil \neg \pi \rceil)$ (Endzustand)

- $\Diamond (\lceil \pi_1 \rceil \wedge b_1 \leq \ell \leq e_1; \ldots ; \lceil \pi_n \rceil \wedge b_n \leq \ell \leq e_n)$ oder
  $\lceil \pi_1 \rceil \wedge b_1 \leq \ell \leq e_1; \ldots ; \lceil \pi_n \rceil \wedge b_n \leq \ell \leq e_n$ (Sequenz)

Statt $\leq$ kann natürlich auch $<$ als Relation verwendet werden. Die Konstruktion von Phasenautomaten zu diesen Pattern ist sehr einfach. Diese Automaten sind aber nicht deterministisch, so dass der Komplementautomat nicht mit der beschriebenen Konstruktion hergestellt werden kann. Aus diesem Grund werden speziell für diese Pattern Phasenautomaten *und* deren Komplemente angegeben.

1. Pattern (Erreichbarer Zustand)

Der Automat $\mathcal{A}_Z$ zu $\Diamond \lceil \pi \rceil \wedge b \leq \ell \wedge \ell \leq e$ hat die in Abbildung 4.1 dargestellte Form. Sei x_{cwait} die neue, sonst nicht benutzte Variable für das Intervall $(0, \infty)$ zur Uhr c_{wait}.

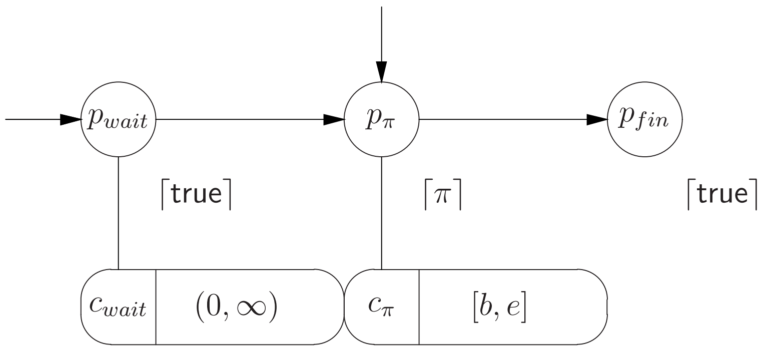


Abbildung 4.1: Automat $\mathcal{A}_Z$ zum 1. Pattern (Zustand)

Der Automat $\mathcal{A}_Z$ erfüllt dann

$$\models (\exists\, x_{cwait} : F(\mathcal{A}_Z)) \iff [\Diamond \lceil \pi \rceil \wedge b \leq \ell \wedge \ell \leq e]$$

Beweis „⇒"-Richtung

$$
\begin{array}{rrl}
 & & (\mathcal{I},\mathcal{V}) \models F(\mathcal{A}_Z) \\
\left\{\begin{array}{r}\text{Für ein geeignetes} \\ \mathcal{I}' =_{\backslash \mathsf{ph}_{\mathcal{A}}} \mathcal{I}\end{array}\right\} & \Rightarrow & \mathcal{I}',\mathcal{V} \models [D(\mathcal{A}_Z)] \\
 & \Rightarrow & \mathcal{I}',\mathcal{V} \models [D_{init}(\mathcal{A}_Z)] \\
\{\text{Lemma 2.2.15 (1)}\} & \Rightarrow & \mathcal{I}',\mathcal{V},\ [0,0] \models_{DC} \boxed{L}_0 D_{init}(\mathcal{A}_Z) \\
 & \Rightarrow & \mathcal{I}',\mathcal{V},\ [0,0] \models_{DC} \boxed{L}_0 \neg(\lceil \mathsf{ph}_{\mathcal{A}} \notin \{p_{wait}, p_\pi\}\rceil ; \mathsf{true}) \\
\{\text{endl. Variabilität}\} & \Rightarrow & \mathcal{I}',\mathcal{V},\ [0,0] \models_{DC} \boxed{L}_0((\lceil \mathsf{ph}_{\mathcal{A}} \in \{p_{wait}, p_\pi\}\rceil ; \mathsf{true}) \vee \lceil\rceil) \\
\{\text{Lemma 2.2.12}\} & \Rightarrow & \mathcal{I}',\mathcal{V},\ [0,0] \models_{DC} (\lceil \mathsf{ph}_{\mathcal{A}} = p_{wait} \vee \mathsf{ph}_{\mathcal{A}} = p_\pi\rceil ; \mathsf{true}) \rhd \mathsf{true} \\
 & \Rightarrow & \mathcal{I}',\mathcal{V},\ [0,0] \models_{DC} (\lceil \mathsf{ph}_{\mathcal{A}} = p_{wait} \vee \mathsf{ph}_{\mathcal{A}} = p_\pi\rceil) \rhd \mathsf{true} \\
\{\text{Lemma 2.2.13}\} & \Rightarrow & \mathcal{I}',\mathcal{V},\ [0,0] \models_{DC} (\lceil \mathsf{ph}_{\mathcal{A}} = p_{wait}\rceil \rhd \mathsf{true}) \vee \\
 & & \qquad (\lceil \mathsf{ph}_{\mathcal{A}} = p_\pi\rceil \rhd \mathsf{true})
\end{array}
$$

Für die erste Alternative Bedingung $\mathcal{I}',\mathcal{V},\ [0,0] \models_{DC} (\lceil \mathsf{ph}_{\mathcal{A}} = p_{wait}\rceil \rhd \mathsf{true})$ gilt:

$$
\begin{array}{rrl}
 & & \mathcal{I}',\mathcal{V},\ [0,0] \models_{DC} (\lceil \mathsf{ph}_{\mathcal{A}} = p_{wait}\rceil \rhd \mathsf{true}) \\
\{D_{pr}^{c_{wait}}(\mathcal{A})\} & \Rightarrow & \mathcal{I}',\mathcal{V},\ [0,0] \models_{DC} (\lceil \mathsf{ph}_{\mathcal{A}} = p_{wait}\rceil ; \lceil \mathsf{ph}_{\mathcal{A}} \neq p_{wait}\rceil) \rhd \mathsf{true} \\
 & \Rightarrow & \mathcal{I}',\mathcal{V},\ [0,0] \models_{DC} (\lceil \mathsf{ph}_{\mathcal{A}} = p_{wait}\rceil ; \lceil \mathsf{ph}_{\mathcal{A}} = p_\pi\rceil) \rhd \mathsf{true}
\end{array}
$$

Daraus folgt in beiden Fällen:

$$
\begin{array}{rrl}
 & & \mathcal{I}',\mathcal{V},\ [0,0] \models_{DC} \Diamond_L \lceil \mathsf{ph}_{\mathcal{A}} = p_\pi\rceil \\
\{D_{pr}^{c_\pi}(\mathcal{A}), D_{st}^{c_\pi}(\mathcal{A}), D_{ist}^{c_\pi}(\mathcal{A})\} & \Rightarrow & \mathcal{I}',\mathcal{V},\ [0,0] \models_{DC} \Diamond_L(\lceil \mathsf{ph}_{\mathcal{A}} = p_\pi\rceil \wedge b \leq \ell \wedge \ell \leq e) \\
\{D_{out}^{p_\pi}(\mathcal{A})\} & \Rightarrow & \mathcal{I}',\mathcal{V},\ [0,0] \models_{DC} \Diamond_L(\lceil \pi\rceil \wedge b \leq \ell \wedge \ell \leq e) \\
\{\mathcal{I} =_{\backslash \mathsf{ph}_{\mathcal{A}}} \mathcal{I}'\} & \Rightarrow & \mathcal{I},\mathcal{V},\ [0,0] \models_{DC} \Diamond_L(\lceil \pi\rceil \wedge b \leq \ell \wedge \ell \leq e) \\
\{\text{Lemma 2.2.15 (2) }\} & \Rightarrow & \mathcal{I},\mathcal{V} \models_{DC} \Diamond_L(\lceil \pi\rceil \wedge b \leq \ell \wedge \ell \leq e) \\
 & \Rightarrow & \mathcal{I},\mathcal{V} \models [\Diamond_L(\lceil \pi\rceil \wedge b \leq \ell \wedge \ell \leq e)]
\end{array}
$$

„⇐"-Richtung Gelte $(\mathcal{I},\mathcal{V}) \models [\Diamond_L(\lceil \pi\rceil \wedge b \leq \ell \wedge \ell \leq e)]$. Nach Definition von $\Diamond_L$ gibt es dann b', e', so dass gilt:

$$
\mathcal{I},\mathcal{V},\ [b',e'] \models_{DC} \lceil \pi\rceil \wedge b \leq (e'-b') \wedge (e'-b') \leq e. \tag{4.4.4.8}
$$

Wegen $\lceil\pi\rceil = \left(\int \pi = \ell \wedge \ell > 0\right)$ gilt auch $b' < e'$.
Sei $\mathcal{I}'$ eine Interpretation mit $\mathcal{I}' =_{\backslash \mathsf{ph}_\mathcal{A}} \mathcal{I}$ und

$$\mathcal{I}'(\mathsf{ph}_\mathcal{A})(t) \overset{df}{=} \begin{cases} p_{wait} & \text{falls } t < b' \\ p_\pi & \text{falls } b' \leq t \leq e' \\ p_{fin} & \text{falls } e' < t \end{cases}$$

sowie $\mathcal{V}'$ eine Belegung mit $\mathcal{V}' =_{\backslash x_{cwait}} \mathcal{V}$ und $\mathcal{V}(x_{cwait}) \overset{df}{=} b'$. Mit dieser Definition gilt:

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg(\lceil \mathsf{ph}_\mathcal{A} \notin \{p_{wait}, p_\pi\}\rceil; \mathsf{true})$ $\quad(= D_{init}(\mathcal{A}_Z))$
- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_\mathcal{A} = p_{wait}\rceil; \lceil \mathsf{ph}_\mathcal{A} = p_{fin}\rceil)$ $\quad(= D_{suc}^{p_{wait}}(\mathcal{A}_Z))$

  wegen $b' \neq e'$ folgen p_{wait} und p_{fin} nicht direkt aufeinander.

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_\mathcal{A} = p_\pi\rceil; \lceil \mathsf{ph}_\mathcal{A} = p_{wait}\rceil)$ $\quad(= D_{suc}^{p_\pi}(\mathcal{A}_Z))$
- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_\mathcal{A} = p_{fin}\rceil; \lceil \mathsf{ph}_\mathcal{A} = p_{wait} \vee \mathsf{ph}_\mathcal{A} = p_\pi\rceil)$ $\quad(= D_{suc}^{p_{fin}}(\mathcal{A}_Z))$
- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond\lceil \mathsf{ph}_\mathcal{A} = p_\pi \wedge \neg\pi\rceil$ $\quad(= D_{out}^{p_\pi}(\mathcal{A}_Z))$

  nach (4.4.4.8). Die anderen Teilformeln $D_{out}^{p_{wait}}(\mathcal{A}_Z) =, D_{out}^{p_{fin}}(\mathcal{A}_Z)$ sind wegen des Zustandsausdrucks true immer erfüllt.

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond\lceil \mathsf{ph}_\mathcal{A} = p_{wait}\rceil^{>x_{cwait}}$ $\quad(= D_{pr}^{c_{wait}}(\mathcal{A}_Z))$
- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond\lceil \mathsf{ph}_\mathcal{A} = p_\pi\rceil^{>e}$ $\quad(= D_{pr}^{c_\pi}(\mathcal{A}_Z))$
- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_\mathcal{A} \neq p_\pi\rceil; \lceil \mathsf{ph}_\mathcal{A} = p_\pi\rceil^{<b}; \lceil \mathsf{ph}_\mathcal{A} \neq p_\pi\rceil)$ $\quad(= D_{st}^{c_\pi}(\mathcal{A}_Z))$
- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg(\lceil \mathsf{ph}_\mathcal{A} = p_\pi\rceil^{<b}; \lceil \mathsf{ph}_\mathcal{A} \neq p_\pi\rceil)$ $\quad(= D_{ist}^{c_\pi}(\mathcal{A}_Z))$

Dann gilt nach Definition von $D(\mathcal{A}_Z)$ auch $\mathcal{I}', \mathcal{V}' \models_{DC} D(\mathcal{A}_Z)$ und damit auch $\mathcal{I}, \mathcal{V}' \models F(\mathcal{A}_Z)$ und

$$\mathcal{I}, \mathcal{V} \models \exists x_{cwait} : F(\mathcal{A}_Z)$$

□

1. Pattern (Komplement)

Der Automat $\overline{\mathcal{A}_Z}$ zu $\neg[\diamondsuit\lceil\pi\rceil \wedge b \leq \ell \wedge \ell \leq e]$ ist in Abbildung 4.2 dargestellt. Dabei wird davon ausgegangen, dass $0 < b$ und $e < \infty$ gilt. Falls $b = 0$ gilt, fällt der Zustand $p_{\pi<}$ weg, da dann das Uhrintervall von c_1 die Form $(0, 0)$ hätte. Das kann aber nicht erfüllt werden, da Phasenautomaten in jeder Phase Zeit verbrauchen. Entsprechend fällt für $e = \infty$ der Zustand $p_{\pi>}$ weg, da ein Intervall $(\infty, \infty]$ nicht existiert. Es gilt für diesen Automaten:

$$\models F(\overline{\mathcal{A}_Z}) \iff \neg[\diamondsuit\lceil\pi\rceil \wedge b \leq \ell \wedge \ell \leq e]$$

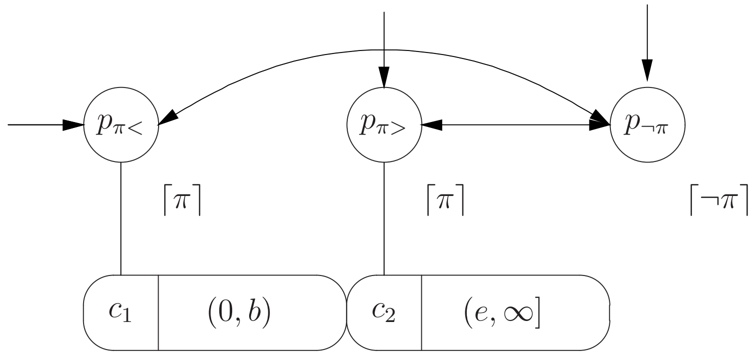


Abbildung 4.2: Komplement-Automat $\overline{\mathcal{A}_Z}$ zu $\mathcal{A}_Z$

Beweis „⇒“-Richtung Der Beweis erfolgt durch Widerspruch. Es existiert ein Lauf des Phasenautomaten, d.h. es gibt ein $\mathcal{I}'$ mit $\mathcal{I}' =_{\setminus \mathsf{ph}_{\mathcal{A}}} \mathcal{I}$ und

$$\mathcal{I}', \mathcal{V} \models_{DC} D(\overline{\mathcal{A}_Z}) \tag{4.4.4.9}$$

Annahme: Es gilt auch $\mathcal{I}, \mathcal{V} \models [\Diamond(\lceil \pi \rceil \wedge b \leq \ell \wedge \ell \leq e)$. Dann folgt:

$$\begin{array}{rrl}
 & & \mathcal{I}, \mathcal{V} \models [\Diamond(\lceil \pi \rceil \wedge b \leq \ell \wedge \ell \leq e)] \\
\{\text{Def. } [\cdot]\} & \Rightarrow & \mathcal{I}, \mathcal{V} \models_{DC} \Diamond(\lceil \pi \rceil \wedge b \leq \ell \wedge \ell \leq e) \\
\{\text{Def. Gültigkeit}\} & \Rightarrow & \mathcal{I}, \mathcal{V}, [0,0] \models_{DC} \Diamond(\lceil \pi \rceil \wedge b \leq \ell \wedge \ell \leq e) \\
\left\{\begin{array}{r}\text{Nach Def.} \Diamond \\ \text{existieren } b', e' :\end{array}\right\} & & \\
 & \Rightarrow & \mathcal{I}, \mathcal{V}, [b', e'] \models_{DC} \lceil \pi \rceil \wedge b \leq \ell \wedge \ell \leq e \\
\left\{\begin{array}{r}\text{Nach Def.} \mathcal{I}' \\ \text{weil } \mathsf{ph}_{\mathcal{A}} \text{ in } \pi \\ \text{nicht vorkommt}\end{array}\right\} & & \\
 & \Rightarrow & \mathcal{I}', \mathcal{V}, [b', e'] \models_{DC} \lceil \pi \rceil \wedge b \leq \ell \wedge \ell \leq e \\
\{4.4.4.9\} & \Rightarrow & \mathcal{I}', \mathcal{V}, [b', e'] \models_{DC} (\lceil p_{\pi<} \vee p_{\pi>} \rceil) \wedge b \leq \ell \wedge \ell \leq e \\
\left\{\begin{array}{r}\text{Keine Transition} \\ \text{zwischen } p_{\pi<} \text{ und } p_{\pi>}\end{array}\right\} & \Rightarrow & \mathcal{I}', \mathcal{V}, [b', e'] \models_{DC} (\lceil p_{\pi<} \rceil \vee \lceil p_{\pi>} \rceil) \wedge b \leq \ell \wedge \ell \leq e \\
\{\text{1. Fall}\} & \Rightarrow & \mathcal{I}', \mathcal{V}, [b', e'] \models_{DC} \lceil p_{\pi<} \rceil \wedge b \leq \ell \wedge \ell \leq e \\
 & \Rightarrow & \qquad \text{Widerspruch zur Uhrbedingung } D_{pr}^{c_1} \\
\{\text{2. Fall}\} & \Rightarrow & \mathcal{I}', \mathcal{V}, [b', e'] \models_{DC} \lceil p_{\pi>} \rceil \wedge b \leq \ell \wedge \ell \leq e \\
 & \Rightarrow & \qquad \text{Widerspruch zu den Uhrbedingungen} \\
 & & \qquad D_{st}^{c_2}, D_{ist}^{c_2}
\end{array}$$

„⇐“-Richtung Gelte $\mathcal{I}, \mathcal{V} \models \neg[\Diamond_L \lceil \pi \rceil \land b \leq \ell \land \ell \leq e]$. Dann gilt auch

$$\begin{aligned}
& & \mathcal{I}, \mathcal{V} &\models \neg[\Diamond_L \lceil \pi \rceil \land b \leq \ell \land \ell \leq e] \\
& \Rightarrow & \neg\forall t : \mathcal{I}, \mathcal{V}, [0,t] &\models_{DC} \Diamond_L \lceil \pi \rceil \land b \leq \ell \land \ell \leq e \\
\{\text{Lemma 2.2.15 (2)}\} & \Rightarrow & \mathcal{I}, \mathcal{V}, [0,0] &\not\models_{DC} \Diamond_L \lceil \pi \rceil \land b \leq \ell \land \ell \leq e \\
& \Rightarrow & \mathcal{I}, \mathcal{V}, [0,0] &\models_{DC} \neg \Diamond_L \lceil \pi \rceil \land b \leq \ell \land \ell \leq e \qquad (4.4.4.10)
\end{aligned}$$

Da immer endliche Variabilität vorausgesetzt wird, kann induktiv eine Interpretation $\mathcal{I}'$ zu $\mathcal{I}$ konstruiert werden, die $D(\overline{\mathcal{A}_Z})$ erfüllt. Setze $\mathcal{I}'(X) = \mathcal{I}(X)$ für alle Observablen $X \neq \mathsf{ph}_{\mathcal{A}}$. Sei außerdem $\mathrm{dur}_{\mathcal{I}}$ wie in Gleichung 3.4.3.7 als Funktion, die zu jedem Zeitpunkt angibt, wie lange $\mathcal{I}$ noch konstant bleibt und sei $t_0, t_1, \ldots$ als Sequenz von Zeitpunkten definiert, zu denen sich $\mathcal{I}$ ändert. Dann wird $\mathcal{I}'(\mathsf{ph}_{\mathcal{A}})(t)$ induktiv über die Anzahl der Änderungen konstruiert.

Induktionsanfang $t_0 = 0$ Setze:

$$\mathcal{I}'(\mathsf{ph}_{\mathcal{A}})(t) \overset{df}{=} \begin{cases} p_{\pi<} & \text{falls } \mathcal{I}, \mathcal{V},\ [0,t_1] \models_{DC} \lceil \pi \rceil \\ & \quad \text{und } \mathrm{dur}_{\mathcal{I}}(0) < b \text{ und } t_0 \leq t < t_1 \\ p_{\pi>} & \text{falls } \mathcal{I}, \mathcal{V},\ [0,t_1] \models_{DC} \lceil \pi \rceil \\ & \quad \text{und } e < \mathrm{dur}_{\mathcal{I}}(0) \text{ und } t_0 \leq t < t_1 \\ p_{\neg\pi} & \text{falls } \mathcal{I}, \mathcal{V},\ [0,t_1] \models_{DC} \lceil \neg\pi \rceil \\ & \quad \text{und } t_0 \leq t < t_1 \\ \text{undef.} & \text{sonst} \end{cases}$$

Da der Fall $\mathcal{I}[\![\pi]\!](0) = 1$ und $b \leq \mathrm{dur}_{\mathcal{I}}(0) \leq e$ nach (4.4.4.10) nicht vorkommen kann, ist $\mathcal{I}'(\mathsf{ph}_{\mathcal{A}})$ für alle $t \in [0, t_1)$ definiert und es gilt

- $\mathcal{I}', \mathcal{V}, [0, t_1) \models_{DC} \neg(\lceil \mathsf{ph}_{\mathcal{A}} \notin \{p_{\pi<}, p_{\pi>}, p_{\neg\pi}\} \rceil; \mathsf{true})$ $\qquad (= D_{init}(\overline{\mathcal{A}_Z}))$

  weil die Bedingung immer erfüllt ist.

- $\mathcal{I}', \mathcal{V}, [0, t_1) \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}} = p \rceil; \lceil \mathsf{ph}_{\mathcal{A}} \notin \widetilde{\mathrm{suc}}(p) \rceil)$ $\qquad (= D^p_{suc}(\overline{\mathcal{A}_Z}))$

  weil $\mathcal{I}'(\mathsf{ph}_{\mathcal{A}})$ auf $[0, t_1)$ konstant ist.

- $\mathcal{I}', \mathcal{V}, [0, t_1) \models_{DC} \neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}} = p \land \neg s(p) \rceil$ $\qquad (= D^p_{out}(\overline{\mathcal{A}_Z}))$

  nach Definition von $\mathcal{I}'(\mathsf{ph}_{\mathcal{A}})$

- $\mathcal{I}', \mathcal{V}', [0, t_1) \models_{DC} \neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}} = p_{\pi<} \rceil^{\geq b}$ $\qquad (= D^{c_1}_{pr}(\overline{\mathcal{A}_Z}))$

nach Definition von $\mathcal{I}'(\mathsf{ph}_\mathcal{A})$ kann $p_{\pi<}$ nur angenommen werden, wenn gilt $\mathrm{dur}_\mathcal{I}(0) < b$.

- $\mathcal{I}', \mathcal{V}', [0, t_1) \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_\mathcal{A} \neq p_{\pi>} \rceil; \lceil \mathsf{ph}_\mathcal{A} = p_{\pi>} \rceil^{\leq e}; \lceil \mathsf{ph}_\mathcal{A} \neq p_{\pi>} \rceil)$ $\quad (= D^{c_2}_{st}(\overline{\mathcal{A}_Z}))$
- $\mathcal{I}', \mathcal{V}', [0, t_1) \models_{DC} \neg(\lceil \mathsf{ph}_\mathcal{A} = p_{\pi>} \rceil^{\leq e}; \lceil \mathsf{ph}_\mathcal{A} \neq p_{\pi>} \rceil)$ $\quad (= D^{c_2}_{ist}(\overline{\mathcal{A}_Z}))$

Wegen der Form der Teilformeln, gelten die Teilformeln auch für jedes Intervall $[0, t], t < t_1$ und damit gilt für alle $t \in [0, t_1)$

$$\mathcal{I}', \mathcal{V},\ [0, t] \models_{DC} D(\overline{\mathcal{A}_Z}).$$

Induktionsschritt Sei $\mathcal{I}'$ schon für alle $t \in [t_0, t_n)$ konstruiert, so dass für alle $t \in [t_0, t_n)$ gilt

$$\mathcal{I}', \mathcal{V},\ [0, t] \models_{DC} D(\overline{\mathcal{A}_Z}).$$

Dann wird $\mathcal{I}'$ zu $\hat{\mathcal{I}}$ erweitert. Seien $\mathcal{I}'$ und $\hat{\mathcal{I}}$ für alle Observablen außer $\mathsf{ph}_\mathcal{A}$ identisch und sei

$$\hat{\mathcal{I}}(\mathsf{ph}_\mathcal{A})(t) \overset{df}{=} \begin{cases} \mathcal{I}'(\mathsf{ph}_\mathcal{A})(t) & \text{falls } t < t_n \\ p_{\pi<} & \text{falls } \mathcal{I}, \mathcal{V},\ [t_n, t_{n+1}] \models_{DC} \lceil \pi \rceil \\ & \quad \text{und } \mathrm{dur}_\mathcal{I}(t_n) < b \text{ und } t_n \leq t < t_{n+1} \\ p_{\pi>} & \text{falls } \mathcal{I}, \mathcal{V},\ [t_n, t_{n+1}] \models_{DC} \lceil \pi \rceil \\ & \quad \text{und } e < \mathrm{dur}_\mathcal{I}(t_n) \text{ und } t_n \leq t < t_{n+1} \\ p_{\neg\pi} & \text{falls } \mathcal{I}, \mathcal{V},\ [t_n, t_{n+1}] \models_{DC} \lceil \neg\pi \rceil \\ & \quad \text{und } t_n \leq t < t_{n+1} \\ undef & \text{sonst} \end{cases}$$

Mit der gleichen Argumentation wie für den Induktionsanfang ist $\hat{\mathcal{I}}(\mathsf{ph}_\mathcal{A})$ für alle $t \in [0, t_{n+1})$ definiert. Weil sich $\mathcal{I}$ bei t_n ändert, muss sich auch $\hat{\mathcal{I}}$ bei t_n ändern und in jedem Fall ist der neue Zustand ein Nachfolgezustand des alten, d.h. es gilt

$$\hat{\mathcal{I}}(\mathsf{ph}_\mathcal{A})(t_n) \in \widetilde{\mathrm{suc}}(\hat{\mathcal{I}}(\mathsf{ph}_\mathcal{A})(t_{n-1})) \quad \text{und} \tag{4.4.4.11}$$

$$\hat{\mathcal{I}}, \mathcal{V}, [t_{n-1}, t_{n+1}) \models_{DC} \lceil \mathsf{ph}_\mathcal{A} = p \rceil; \lceil \mathsf{ph}_\mathcal{A} \in \widetilde{\mathrm{suc}}(p) \rceil \tag{4.4.4.12}$$

Ab hier wird wieder $\mathcal{I}'$ statt $\hat{\mathcal{I}}$ geschrieben und für die Teilformeln von $D(\overline{\mathcal{A}_Z})$ gezeigt, dass sie durch das so konstruierte $\mathcal{I}'$ erfüllt werden.

Initiale Phase $\mathcal{I}', \mathcal{V},\ [0, t] \models_{DC} D_{init}(\overline{\mathcal{A}_Z})$, weil nach Induktionsvoraussetzung die Bedingung schon für alle $t \in [t_0, t_n)$ erfüllt ist.

Nachfolge-Relation Zu zeigen ist, dass für alle Phasen $p \in \{p_{\pi<}, p_{\pi>}, p_{\neg\pi}\}$ des Phasenautomaten gilt:

$$\begin{aligned} &\forall\, t \in [0, t_n)\ : \mathcal{I}', \mathcal{V},\ [0,t] \models_{DC} D^p_{suc} \\ \Rightarrow &\forall\, t \in [0, t_{n+1})\ : \mathcal{I}', \mathcal{V},\ [0,t] \models_{DC} D^p_{suc} \end{aligned}$$

Beweis durch Kontraposition. Sei p eine Phase.

$$\begin{aligned} & & \neg\forall\, t < t_{n+1}\ &: \mathcal{I}', \mathcal{V},\ [0,t] \models_{DC} D^p_{suc}(\overline{\mathcal{A}_Z}) \\ &\Rightarrow & \exists\, t < t_{n+1}\ &: \mathcal{I}', \mathcal{V},\ [0,t] \not\models_{DC} D^p_{suc}(\overline{\mathcal{A}_Z}) \\ &\Rightarrow & & \mathcal{I}', \mathcal{V},\ [0,t] \models_{DC} \Diamond(\lceil \mathsf{ph}_{\mathcal{A}} = p \rceil ; \lceil \mathsf{ph}_{\mathcal{A}} \notin \widetilde{\mathrm{suc}}(p) \rceil) \\ \{\text{ Def.}\Diamond\} &\Rightarrow & \exists\, 0 \le \tau_1 < \tau_2 < \tau_3 \le t \le t_{n+1}\ &: \\ & & & \mathcal{I}', \mathcal{V},\ [\tau_1, \tau_2] \models_{DC} \lceil \mathsf{ph}_{\mathcal{A}} = p \rceil \\ & & & \text{und } \mathcal{I}', \mathcal{V},\ [\tau_2, \tau_3] \models_{DC} \lceil \mathsf{ph}_{\mathcal{A}} \notin \widetilde{\mathrm{suc}}(p) \rceil \\ \{(4.4.4.12)\} &\Rightarrow & \exists\, 0 \le \tau_1 < \tau_2 < \tau_3 \le t \le t_n\ &: \\ & & & \mathcal{I}', \mathcal{V},\ [\tau_1, \tau_2] \models_{DC} \lceil \mathsf{ph}_{\mathcal{A}} = p \rceil \\ & & & \text{und } \mathcal{I}', \mathcal{V},\ [\tau_2, \tau_3] \models_{DC} \lceil \mathsf{ph}_{\mathcal{A}} \notin \widetilde{\mathrm{suc}}(p) \rceil \\ &\Rightarrow & \exists\, t < t_n\ &: \mathcal{I}', \mathcal{V},\ [0,t] \not\models_{DC} D^p_{suc}(\overline{\mathcal{A}_Z}) \\ &\Rightarrow & \neg\forall\, t < t_n\ &: \mathcal{I}', \mathcal{V},\ [0,t] \not\models_{DC} D^p_{suc}(\overline{\mathcal{A}_Z}) \end{aligned}$$

Zustandsausdrücke sind erfüllt Zu zeigen, für alle $p \in \{p_{\pi<}, p_{\pi>}, p_{\neg\pi}\}$

$$\begin{aligned} &\forall\, t \in [0, t_n) : \mathcal{I}', \mathcal{V},\ [0,t] \models_{DC} D^p_{out} \\ \Rightarrow &\forall\, t \in [0, t_{n+1}) : \mathcal{I}', \mathcal{V},\ [0,t] \models_{DC} D^p_{out} \end{aligned}$$

Beweis durch Kontraposition. Sei $p \in \{p_{\pi<}, p_{\pi>}, p_{\neg\pi}\}$

$$\begin{aligned} & & \neg\forall\, t < t_{n+1}\ &: \mathcal{I}', \mathcal{V},\ [0,t] \models_{DC} D^p_{out}(\overline{\mathcal{A}_Z}) \\ &\Rightarrow & \exists\, t < t_{n+1}\ &: \mathcal{I}', \mathcal{V},\ [0,t] \not\models_{DC} D^p_{out}(\overline{\mathcal{A}_Z}) \\ &\Rightarrow & & \mathcal{I}', \mathcal{V},\ [0,t] \models_{DC} \Diamond\lceil \mathsf{ph}_{\mathcal{A}} = p \wedge \neg s(p) \rceil \\ \{\text{ Def.}\Diamond\} &\Rightarrow & \exists\, 0 \le \tau_1 < \tau_2 \le t \le t_{n+1}\ &: \\ & & & \mathcal{I}', \mathcal{V},\ [\tau_1, \tau_2] \models_{DC} \lceil \mathsf{ph}_{\mathcal{A}} = p \wedge \neg s(p) \rceil \\ \{\text{Konstruktion}\} &\Rightarrow & \exists\, 0 \le \tau_1 < \tau_2 \le t \le t_n\ &: \\ & & & \mathcal{I}', \mathcal{V},\ [\tau_1, \tau_2] \models_{DC} \lceil \mathsf{ph}_{\mathcal{A}} = p \wedge \neg s(p) \rceil \\ &\Rightarrow & \exists\, t < t_n\ &: \mathcal{I}', \mathcal{V},\ [0,t] \not\models_{DC} D^p_{out}(\overline{\mathcal{A}_Z}) \end{aligned}$$

$$\Rightarrow \qquad \neg\forall\, t < t_n \, : \mathcal{I}', \mathcal{V},\, [0,t] \not\models_{DC} D^p_{out}(\overline{\mathcal{A}_Z})$$

Die anderen Teilformeln folgen analog. Damit gilt insgesamt:

$$\forall\, t \in [0, t_{n+1})\; \mathcal{I}', \mathcal{V},\, [0,t] \models_{DC} D(\overline{\mathcal{A}_Z})$$

Auf diese Weise kann also eine Interpretation $\mathcal{I}'$ konstruiert werden, für die gilt:

$$\mathcal{I}', \mathcal{V} \models_{DC} D(\overline{\mathcal{A}_Z})$$

Und damit gilt auch:

$$\mathcal{I}, \mathcal{V} \models F(\overline{\mathcal{A}_Z})$$

$\square$

2. Pattern

Der Automat $\mathcal{A}_E$ zu $\Diamond_L \Box_L \lceil \pi \rceil \wedge \neg \Diamond_L (\lceil \pi \rceil ; \lceil \neg\pi \rceil)$ hat die in Abbildung 4.3 dargestellte Form. Sei $x_{c\neg\pi}$ die frische Variable, die für die Uhrbedingung der Uhr $c_{\neg\pi}$ eingeführt

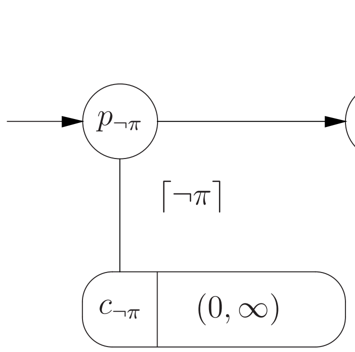


Abbildung 4.3: Automat A_E zum 2. Pattern (Endzustand)

wird. Für $\mathcal{A}_E$ gilt dann:

$$\models (\exists x_{cwait} : F(\mathcal{A}_E)) \iff [\Diamond_L \Box_L \lceil \pi \rceil \wedge \neg \Diamond_L (\lceil \pi \rceil ; \lceil \neg\pi \rceil)].$$

Beweis „⇒“-Richtung

$$
\begin{aligned}
& & \mathcal{I}, \mathcal{V} &\models F(\mathcal{A}_E) \\
\left\{\begin{array}{r}\text{Für ein geeignetes}\\ \mathcal{I}' =_{\setminus \mathsf{ph}_\mathcal{A}} \mathcal{I}\end{array}\right\} &\Rightarrow & \mathcal{I}', \mathcal{V} &\models [D(\mathcal{A}_E)] \\
&\Rightarrow & \mathcal{I}', \mathcal{V} &\models [D_{init}(\mathcal{A}_E)] \\
\{\text{Lemma 2.2.15 (1)}\} &\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} \boxed{\scriptstyle L}_0 D_{init}(\mathcal{A}_E) \\
\{\text{Def. } D_{init}(\mathcal{A}_E)\} &\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} \boxed{\scriptstyle L}_0 \neg(\lceil \mathsf{ph}_\mathcal{A} \notin \{p_{\neg\pi}, p_\pi\} \rceil; \mathsf{true}) \\
\{\text{endl. Variabilität}\} &\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} \boxed{\scriptstyle L}_0 ((\lceil \mathsf{ph}_\mathcal{A} \in \{p_{\neg\pi}, p_\pi\} \rceil; \mathsf{true}) \vee \lceil\rceil) \\
\{\text{Lemma 2.2.12}\} &\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} (\lceil \mathsf{ph}_\mathcal{A} \in \{p_{\neg\pi}, p_\pi\} \rceil; \mathsf{true}) \rhd \mathsf{true} \\
&\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} (\lceil \mathsf{ph}_\mathcal{A} = p_{\neg\pi} \vee \mathsf{ph}_\mathcal{A} = p_\pi \rceil; \mathsf{true}) \rhd \mathsf{true} \\
&\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} (\lceil \mathsf{ph}_\mathcal{A} = p_{\neg\pi} \vee \mathsf{ph}_\mathcal{A} = p_\pi \rceil) \rhd \mathsf{true} \\
\{\text{Lemma 2.2.13}\} &\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} (\lceil \mathsf{ph}_\mathcal{A} = p_{\neg\pi} \rceil \rhd \mathsf{true}) \vee \qquad (4.4.4.13) \\
& & & \quad (\lceil \mathsf{ph}_\mathcal{A} = p_\pi \rceil \rhd \mathsf{true}) \qquad (4.4.4.14)
\end{aligned}
$$

Für die erste Teilformel (4.4.4.13) gilt:

$$
\begin{aligned}
& & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} (\lceil \mathsf{ph}_\mathcal{A} = p_{\neg\pi} \rceil \rhd \mathsf{true}) \\
\{D_{pr}^{c_{\neg\pi}}(\mathcal{A}_E)\} &\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} (\lceil \mathsf{ph}_\mathcal{A} = p_{\neg\pi} \rceil; \lceil \mathsf{ph}_\mathcal{A} \neq p_{\neg\pi} \rceil) \rhd \mathsf{true} \\
&\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} (\lceil \mathsf{ph}_\mathcal{A} = p_{\neg\pi} \rceil; \lceil \mathsf{ph}_\mathcal{A} = p_\pi \rceil) \rhd \mathsf{true} \qquad (4.4.4.15)
\end{aligned}
$$

Es gilt nach Konstruktion $\widetilde{\mathrm{suc}}(p_\pi) = p_\pi$ und damit

$$\mathcal{I}', \mathcal{V} \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_\mathcal{A} = p_\pi \rceil; \lceil \mathsf{ph}_\mathcal{A} \neq p_\pi \rceil) \qquad (4.4.4.16)$$

Daraus folgt damit:

$$
\begin{aligned}
\{(4.4.4.14),\ (4.4.4.15)\ \} & & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} \Diamond_L \lceil \mathsf{ph}_\mathcal{A} = p_\pi \rceil \\
&\Rightarrow & & \\
\left\{\begin{array}{r}\text{Nach Def. } \rhd\\ \text{existieren } b, e \text{ mit}\end{array}\right\} & & \mathcal{I}', \mathcal{V},\ [b,k] &\models_{DC} \lceil \mathsf{ph}_\mathcal{A} = p_\pi \rceil \\
\{4.4.4.16\} &\Rightarrow & \forall k' \geq k\ :\ \mathcal{I}', \mathcal{V},\ [b,k'] &\models_{DC} \lceil \mathsf{ph}_\mathcal{A} = p_\pi \rceil \\
&\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} \Diamond_L \boxed{\scriptstyle L}(\lceil \mathsf{ph}_\mathcal{A} = p_\pi \rceil) \\
\{D_{out}^{p_\pi}(\mathcal{A}_E) = \neg\Diamond(\lceil p_\pi \wedge \neg\pi \rceil)\} &\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} \Diamond_L \boxed{\scriptstyle L}(\lceil \pi \rceil) \\
\left\{\begin{array}{r}\mathcal{I} =_{\setminus \mathsf{ph}_\mathcal{A}} \mathcal{I}' \text{ und}\\ \mathsf{ph}_\mathcal{A} \text{ kommt in } \pi\\ \text{nicht vor}\end{array}\right\} &\Rightarrow & \mathcal{I}, \mathcal{V},\ [0,0] &\models_{DC} \Diamond_L \boxed{\scriptstyle L}(\lceil \pi \rceil) \\
\{\text{Lemma 2.2.15 (2)}\} &\Rightarrow & \mathcal{I}, \mathcal{V} &\models_{DC} \Diamond_L \boxed{\scriptstyle L}(\lceil \pi \rceil)
\end{aligned}
$$

Außerdem gilt:

$$
\begin{array}{rcl}
 & & \mathcal{I}', \mathcal{V} \models_{DC} D^{p_\pi}_{suc}(\mathcal{A}_E) \\
\{\text{Def. } D^{p_\pi}_{suc}(\mathcal{A}_E)\} & \Rightarrow & \mathcal{I}', \mathcal{V} \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_\mathcal{A} = p_\pi \rceil ; \lceil \mathsf{ph}_\mathcal{A} \notin \widetilde{\mathrm{suc}}(p_\pi) \rceil) \\
\{\widetilde{\mathrm{suc}}(p_\pi) = p_\pi\} & \Rightarrow & \mathcal{I}', \mathcal{V} \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_\mathcal{A} = p_\pi \rceil ; \lceil \mathsf{ph}_\mathcal{A} \neq p_\pi \rceil) \\
\{D^{p_\pi}_{out}(\mathcal{A}_E)\} & \Rightarrow & \mathcal{I}', \mathcal{V} \models_{DC} \neg\Diamond(\lceil \pi \rceil ; \lceil \neg p_\pi \rceil) \\
\left\{\begin{array}{r}\text{Def. Gültigkeit} \\ \text{und Def. } \Diamond\end{array}\right\} & \Rightarrow & \neg\exists\, \tau_1, \tau_2 : \mathcal{I}', \mathcal{V}, [\tau_1, \tau_2] \models_{DC} \lceil \pi \rceil ; \lceil \neg p_\pi \rceil \\
\{\text{Def. } \Diamond_L\} & \Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] \models_{DC} \neg\Diamond_L(\lceil \pi \rceil ; \lceil \neg p_\pi \rceil) \\
\left\{\begin{array}{r}\mathcal{I} =_{\setminus \mathsf{ph}_\mathcal{A}} \mathcal{I}' \text{ und} \\ \mathsf{ph}_\mathcal{A} \text{ kommt in} \pi \\ \text{nicht vor}\end{array}\right\} & \Rightarrow & \mathcal{I}, \mathcal{V},\ [0,0] \models_{DC} \neg\Diamond_L(\lceil \pi \rceil ; \lceil \neg p_\pi \rceil) \\
\{\text{Lemma 2.2.15 (2)}\} & \Rightarrow & \mathcal{I}, \mathcal{V} \models_{DC} \neg\Diamond_L(\lceil \pi \rceil ; \lceil \neg p_\pi \rceil)
\end{array}
$$

Insgesamt gilt dann:

$$\mathcal{I}, \mathcal{V} \models [\Diamond_L \Box_L(\lceil \pi \rceil) \wedge \neg\Diamond_L(\lceil \pi \rceil ; \lceil \neg p_\pi \rceil)]$$

„⇐"-Richtung Gelte $\mathcal{I}, \mathcal{V} \models [\Diamond_L \Box_L(\lceil \pi \rceil) \wedge \neg\Diamond_L(\lceil \pi \rceil ; \lceil \neg p_\pi \rceil)]$. Sei $x_{c\neg\pi}$ die frische Uhrvariable im Intervall $(0, \infty)$ zur Uhr $c_{\neg\pi}$. Dann existiert ein Zeitpunkt t_π, so dass

$$\mathcal{I}[\![\pi]\!](t) = \begin{cases} 0 & \text{falls } t \sim t_\pi \\ 1 & \text{sonst} \end{cases} \tag{4.4.4.17}$$

gilt, wobei $\sim \in \{<, \leq\}$. Damit kann $(\mathcal{I}', \mathcal{V}')$ definiert werden durch:

- $\mathcal{I}'(X) \stackrel{df}{=} \mathcal{I}(X)$ für alle Observablen $X \neq \mathsf{ph}_\mathcal{A}$.
- $$\mathcal{I}'(\mathsf{ph}_\mathcal{A})(t) \stackrel{df}{=} \begin{cases} p_{\neg\pi} & \text{falls } \mathcal{I}[\![\pi]\!](t) = 0 \\ p_\pi & \text{sonst} \end{cases} \tag{4.4.4.18}$$
- $$\mathcal{V}'(x) \stackrel{df}{=} \begin{cases} t_\pi & \text{falls } x = x_{c\neg\pi} \\ \mathcal{V}(x) & \text{sonst.} \end{cases} \tag{4.4.4.19}$$

Mit dieser Definition gilt:

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg(\lceil \mathsf{ph}_{\mathcal{A}} \notin \{p_{\neg\pi}, p_{\pi}\} \rceil ; \mathsf{true}) \qquad (= D_{init}(\mathcal{A}_E))$

  Die Bedingung ist immer erfüllt, weil alle Zustände Startzustände sind.

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}} = p_{\neg\pi} \rceil ; \lceil \mathsf{ph}_{\mathcal{A}} \notin \{p_{\neg\pi}, p_{\pi}\} \rceil) \qquad (= D_{suc}^{p_{\neg\pi}}(\mathcal{A}_E))$

  Die Bedingung ist immer erfüllt, weil alle Zustände Nachfolger von $p_{\neg\pi}$ sind.

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}} = p_{\pi} \rceil ; \lceil \mathsf{ph}_{\mathcal{A}} \neq p_{\pi} \rceil) \qquad (= D_{suc}^{p_{\pi}}(\mathcal{A}_E))$

  wegen (4.4.4.17) und (4.4.4.18)

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}} = p_{\neg\pi} \wedge \pi \rceil \qquad (= D_{out}^{p_{\neg\pi}}(\mathcal{A}_E))$

  wegen (4.4.4.18)

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}} = p_{\pi} \wedge \neg\pi \rceil \qquad (= D_{out}^{p_{\pi}}(\mathcal{A}_E))$

  wegen (4.4.4.18)

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}} = p_{wait} \rceil^{> x_{c\neg\pi}} \qquad (= D_{pr}^{c_{\neg\pi}}(\mathcal{A}_E))$

  wegen (4.4.4.18) und (4.4.4.19)

und damit auch

$$\mathcal{I}', \mathcal{V}' \models_{DC} D(\mathcal{A}_E)$$

und

$$\mathcal{I}, \mathcal{V} \models_{DC} \exists\, x_{c\neg\pi} : F(\mathcal{A}_E).$$

□

2. Pattern (Komplement)

Der Automat $\overline{\mathcal{A}_E}$ ist in Abbildung 4.4 dargestellt. Sei $x_{c\pi}$ die Variable, die zu der Uhr c_{π} gehört. Dann gilt für den Automaten:

$$\models (\exists\, x_{c\pi} : F(\overline{\mathcal{A}_E})) \iff \neg[\Diamond_{\ell}\, \boxed{\ell}\, \lceil \pi \rceil \wedge \neg\Diamond_{\ell}(\lceil \pi \rceil ; \lceil \neg\pi \rceil)]$$

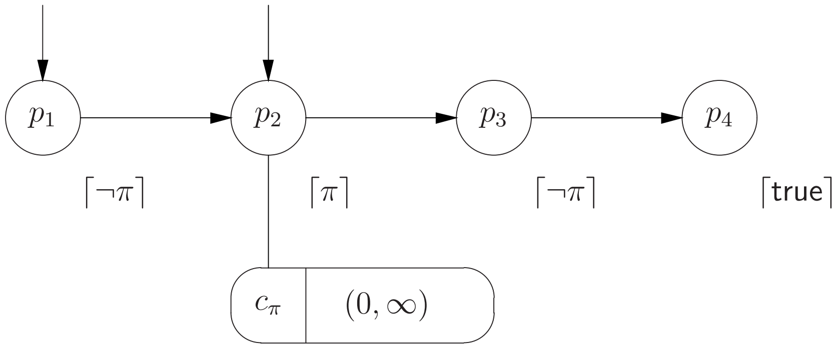


Abbildung 4.4: Automat $\overline{\mathcal{A}_E}$ (2. Pattern, Komplement)

Beweis „⇒“-Richtung Gelte

$$\mathcal{I}, \mathcal{V} \models F(\overline{\mathcal{A}_E}).$$

Dann gilt auch:

$$\begin{aligned}
& & \mathcal{I}, \mathcal{V} &\models F(\overline{\mathcal{A}_E}) \\
\left\{\begin{array}{r}\text{Für ein geeignetes} \\ \mathcal{I}' =_{\backslash \mathsf{ph}_\mathcal{A}} \mathcal{I}\end{array}\right\} &\Rightarrow & \mathcal{I}', \mathcal{V} &\models_{DC} D(\overline{\mathcal{A}_E}) \\
&\Rightarrow & \mathcal{I}', \mathcal{V} &\models_{DC} \neg(\lceil \mathsf{ph}_\mathcal{A} \notin \{p_1, p_2\} \rceil ; \mathsf{true}) \\
\left\{ \text{wie für } \mathcal{A}_E \right\} &\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} (\lceil \mathsf{ph}_\mathcal{A} = p_1 \rceil \rhd \mathsf{true}) \vee \\
& & & \quad (\lceil \mathsf{ph}_\mathcal{A} = p_2 \rceil \rhd \mathsf{true})
\end{aligned} \tag{4.4.4.20}$$

1. Fall Es gilt $\mathcal{I}', \mathcal{V},\ [0,0] \models_{DC} \boxed{L} \lceil \mathsf{ph}_\mathcal{A} = p_1 \rceil$. Dann gilt:

$$\begin{aligned}
& & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} \boxed{L} \lceil \mathsf{ph}_\mathcal{A} = p_1 \rceil \\
\left\{\begin{array}{r} D_{out}^{p_1}(\overline{\mathcal{A}_E}) \\ = \neg\Diamond\lceil \mathsf{ph}_\mathcal{A} = p_1 \wedge \pi \rceil \end{array}\right\} &\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} \boxed{L} \lceil \neg\pi \rceil \\
\{\text{Def.} \boxed{L}\} &\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} \neg \Diamond_L \lceil \neg\pi \rceil \\
&\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} \neg \Diamond_L \boxed{L} \lceil \neg\pi \rceil \\
&\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} \neg \Diamond_L \boxed{L} \lceil \neg\pi \rceil \vee \Diamond_L (\lceil \pi \rceil ; \lceil \neg\pi \rceil) \\
&\Rightarrow & \mathcal{I}', \mathcal{V},\ [0,0] &\models_{DC} \neg\big(\Diamond_L \boxed{L} \lceil \neg\pi \rceil \wedge \neg \Diamond_L (\lceil \pi \rceil ; \lceil \neg\pi \rceil)\big) \\
&\Rightarrow & \exists t : \mathcal{I}', \mathcal{V},\ [0,t] &\models_{DC} \neg\big(\Diamond_L \boxed{L} \lceil \neg\pi \rceil \wedge \neg \Diamond_L (\lceil \pi \rceil ; \lceil \neg\pi \rceil)\big) \\
&\Rightarrow & \not\exists t : \mathcal{I}', \mathcal{V},\ [0,t] &\models_{DC} \big(\Diamond_L \boxed{L} \lceil \neg\pi \rceil \wedge \neg \Diamond_L (\lceil \pi \rceil ; \lceil \neg\pi \rceil)\big)
\end{aligned}$$

$$\Rightarrow \qquad \mathcal{I}', \mathcal{V} \models \neg\big[\Diamond_L \Box_L \lceil \neg\pi \rceil \wedge \neg\Diamond_L(\lceil \pi \rceil ; \lceil \neg\pi \rceil)\big]$$

2. Fall Es gilt $\mathcal{I}', \mathcal{V}, [0,0] \models_{DC} \neg\Box_L \lceil \mathsf{ph}_{\mathcal{A}} = p_1 \rceil$. Dann folgt aus 4.4.4.20:

$$\begin{aligned}
& & & \mathcal{I}', \mathcal{V}, [0,0] \models_{DC} (\lceil \mathsf{ph}_{\mathcal{A}} = p_1 \rceil \rhd \mathsf{true}) \vee \\
& & & \qquad (\lceil \mathsf{ph}_{\mathcal{A}} = p_2 \rceil \rhd \mathsf{true}) \\
\{2.\ \text{Fall}\} & \Rightarrow & & \mathcal{I}', \mathcal{V}, [0,0] \models_{DC} ((\lceil \mathsf{ph}_{\mathcal{A}} = p_1 \rceil ; \lceil \mathsf{ph}_{\mathcal{A}} \neq p_1 \rceil) \rhd \mathsf{true}) \vee \\
& & & \qquad (\lceil \mathsf{ph}_{\mathcal{A}} = p_2 \rceil \rhd \mathsf{true}) \\
\big\{ D^{p_1}_{suc}(\overline{\mathcal{A}_E}) \big\} & \Rightarrow & & \mathcal{I}', \mathcal{V}, [0,0] \models_{DC} ((\lceil \mathsf{ph}_{\mathcal{A}} = p_1 \rceil ; \lceil \mathsf{ph}_{\mathcal{A}} = p_2 \rceil) \rhd \mathsf{true}) \vee \\
& & & \qquad (\lceil \mathsf{ph}_{\mathcal{A}} = p_2 \rceil \rhd \mathsf{true}) \\
\big\{ D^{c_\pi}_{pr}(\overline{\mathcal{A}_E}) \big\} & \Rightarrow & & \mathcal{I}', \mathcal{V}, [0,0] \models_{DC} ((\lceil \mathsf{ph}_{\mathcal{A}} = p_1 \rceil ; \lceil \mathsf{ph}_{\mathcal{A}} = p_2 \rceil ; \lceil \mathsf{ph}_{\mathcal{A}} \neq p_2 \rceil) \rhd \mathsf{true}) \vee \\
& & & \qquad ((\lceil \mathsf{ph}_{\mathcal{A}} = p_2 \rceil ; \lceil \mathsf{ph}_{\mathcal{A}} \neq p_2 \rceil) \rhd \mathsf{true}) \\
\big\{ D^{p_2}_{suc}(\overline{\mathcal{A}_E}) \big\} & \Rightarrow & & \mathcal{I}', \mathcal{V}, [0,0] \models_{DC} ((\lceil \mathsf{ph}_{\mathcal{A}} = p_1 \rceil ; \lceil \mathsf{ph}_{\mathcal{A}} = p_2 \rceil ; \lceil \mathsf{ph}_{\mathcal{A}} = p_3 \rceil) \rhd \mathsf{true}) \vee \\
& & & \qquad ((\lceil \mathsf{ph}_{\mathcal{A}} = p_2 \rceil ; \lceil \mathsf{ph}_{\mathcal{A}} \neq p_3 \rceil) \rhd \mathsf{true}) \\
\big\{ D^{p}_{out}(\overline{\mathcal{A}_E}) \big\} & \Rightarrow & & \mathcal{I}', \mathcal{V}, [0,0] \models_{DC} ((\lceil \neg\pi \rceil ; \lceil \pi \rceil ; \lceil \neg\pi \rceil) \rhd \mathsf{true}) \vee \\
& & & \qquad ((\lceil \pi \rceil ; \lceil \neg\pi \rceil) \rhd \mathsf{true}) \\
& \Rightarrow & & \mathcal{I}', \mathcal{V}, [0,0] \models_{DC} \Diamond_L(\lceil \pi \rceil ; \lceil \neg\pi \rceil) \\
& \Rightarrow & & \mathcal{I}', \mathcal{V}, [0,0] \models_{DC} \neg\Diamond_L \Box_L \lceil \neg\pi \rceil \vee \Diamond_L(\lceil \pi \rceil ; \lceil \neg\pi \rceil) \\
& \Rightarrow & & \mathcal{I}', \mathcal{V}, [0,0] \models_{DC} \neg\big(\Diamond_L \Box_L \lceil \neg\pi \rceil \wedge \neg\Diamond_L(\lceil \pi \rceil ; \lceil \neg\pi \rceil)\big) \\
& \Rightarrow & & \exists t : \mathcal{I}', \mathcal{V}, [0,t] \models_{DC} \neg\big(\Diamond_L \Box_L \lceil \neg\pi \rceil \wedge \neg\Diamond_L(\lceil \pi \rceil ; \lceil \neg\pi \rceil)\big) \\
& \Rightarrow & & \not\forall t : \mathcal{I}', \mathcal{V}, [0,t] \models_{DC} \big(\Diamond_L \Box_L \lceil \neg\pi \rceil \wedge \neg\Diamond_L(\lceil \pi \rceil ; \lceil \neg\pi \rceil)\big) \\
& \Rightarrow & & \mathcal{I}', \mathcal{V} \models \neg\big[\Diamond_L \Box_L \lceil \neg\pi \rceil \wedge \neg\Diamond_L(\lceil \pi \rceil ; \lceil \neg\pi \rceil)\big]
\end{aligned}$$

Damit gilt in jedem Fall

$$\mathcal{I}', \mathcal{V} \models \neg\big[\Diamond_L \Box_L \lceil \neg\pi \rceil \wedge \neg\Diamond_L(\lceil \pi \rceil ; \lceil \neg\pi \rceil)\big]$$

„⇐“-Richtung

$$\mathcal{I}, \mathcal{V} \models \neg[\Diamond_L \Box_L \lceil \pi \rceil \wedge \neg\Diamond_L(\lceil \pi \rceil ; \lceil \neg\pi \rceil)]. \tag{4.4.4.21}$$

Wegen der endlichen Variabilität von $\mathcal{I}$ kann unterschieden werden, ob ein Anfangsstück π erfüllt oder nicht.

1. Fall Auf einem Anfangsstück von $\mathcal{I}$ ist π erfüllt. Wegen 4.4.4.21 kann nicht die ganze Zeit π erfüllt sein, d.h. auf das Anfangsstück, das π erfüllt wird von einem Stück gefolgt, das $\neg\pi$ erfüllt, d.h. es existieren t_π und $t_{\neg\pi}$, so dass gilt

$$\mathcal{I}, \mathcal{V}, [0, t_\pi] \models \lceil \pi \rceil \tag{4.4.4.22}$$

$$\mathcal{I}, \mathcal{V}, (t_\pi, t_{\neg\pi}] \models \lceil \neg\pi \rceil. \tag{4.4.4.23}$$

Dabei ist $t_{\neg\pi}$ nicht eindeutig bestimmt und der Fall, dass nach t_π für alle Zeit $\neg\pi$ erfüllt ist, ist ebenfalls eingeschlossen. Dann können $\mathcal{I}'$ und $\mathcal{V}'$ definiert werden durch:

$$\bullet\, \mathcal{I}'(X) \stackrel{df}{=} \mathcal{I}(X) \quad \text{für alle Observablen } X \neq \mathsf{ph}_\mathcal{A} \tag{4.4.4.24}$$

$$\bullet\, \mathcal{I}'(\mathsf{ph}_\mathcal{A})(t) \stackrel{df}{=} \begin{cases} p_2 & \text{falls } t \in [0, t_\pi] \\ p_3 & \text{falls } t \in (t_\pi, t_{\neg\pi}] \\ p_4 & \text{sonst} \end{cases} \tag{4.4.4.25}$$

$$\bullet\, \mathcal{V}'(x) \stackrel{df}{=} \begin{cases} \mathcal{V}(x) & \text{falls } x \neq x_{c\neg\pi} \\ t_\pi & \text{falls } x = x_{c\neg\pi} \end{cases} \tag{4.4.4.26}$$

Mit dieser Definition gilt:

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg(\lceil \mathsf{ph}_\mathcal{A} \notin \{p_1, p_2\} \rceil; \mathsf{true})$ $\qquad (= D_{init}(\overline{\mathcal{A}_E}))$
- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_\mathcal{A} = p_1 \rceil; \lceil \mathsf{ph}_\mathcal{A} = p3 \vee \mathsf{ph}_\mathcal{A} = p_4 \rceil)$ $\qquad (= D_{suc}^{p_1}(\overline{\mathcal{A}_E}))$

  Zustand p_1 wird von $\mathcal{I}(\mathsf{ph}_\mathcal{A})$ nicht angenommen.

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_\mathcal{A} = p_2 \rceil; \lceil \mathsf{ph}_\mathcal{A} = p1 \vee \mathsf{ph}_\mathcal{A} = p_4 \rceil)$ $\qquad (= D_{suc}^{p_2}(\overline{\mathcal{A}_E}))$

  Def. (4.4.4.25)

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_\mathcal{A} = p_3 \rceil; \lceil \mathsf{ph}_\mathcal{A} = p1 \vee \mathsf{ph}_\mathcal{A} = p_2 \rceil)$ $\qquad (= D_{suc}^{p_3}(\overline{\mathcal{A}_E}))$

  Def. (4.4.4.25)

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_\mathcal{A} = p_4 \rceil; \lceil \mathsf{ph}_\mathcal{A} \neq p_4 \rceil)$ $\qquad (= D_{suc}^{p_4}(\overline{\mathcal{A}_E}))$

  Def. (4.4.4.25)

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond\lceil \mathsf{ph}_\mathcal{A} = p_1 \wedge \pi \rceil$ $\qquad (= D_{out}^{p_1}(\overline{\mathcal{A}_E}))$

  Zustand p_1 wird von $\mathcal{I}(\mathsf{ph}_\mathcal{A})$ nicht angenommen.

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond\lceil \mathsf{ph}_\mathcal{A} = p_2 \wedge \neg\pi \rceil$ $\qquad (= D_{out}^{p_2}(\overline{\mathcal{A}_E}))$

Def. (4.4.4.22) und (4.4.4.25)

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}} = p_3 \wedge \pi \rceil$ $\qquad (= D^{p_3}_{out}(\overline{\mathcal{A}_E}))$

  Def. (4.4.4.23) und (4.4.4.25)

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}} = p_3 \wedge \mathsf{false} \rceil$ $\qquad (= D^{p_4}_{out}(\overline{\mathcal{A}_E}))$
- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}} = p_2 \rceil^{>x_{c\neg\pi}}$ $\qquad (= D^{c\neg\pi}_{pr}(\overline{\mathcal{A}_E}))$

  Def. (4.4.4.25) und (4.4.4.26)

Damit gilt $\mathcal{I}', \mathcal{V}' \models_{DC} D(\overline{\mathcal{A}_E})$ und damit

$$\mathcal{I}, \mathcal{V} \models \exists\, x_{c\neg\pi} : F(\overline{\mathcal{A}_E}).$$

2. Fall Auf einem Anfangsstück von $\mathcal{I}$ ist $\neg\pi$ erfüllt. Weiter kann dann unterschieden werden, ob überhaupt irgendwann π erfüllt ist oder ob immer $\neg\pi$ wahr ist.

2. Fall (a) Für jedes Teilintervall gilt $\lceil\neg\pi\rceil$. Dann sei

- $\mathcal{I}'(X) \stackrel{df}{=} \mathcal{I}(X)$ für alle Observablen $X \neq \mathsf{ph}_{\mathcal{A}}$ (4.4.4.27)
- $\mathcal{I}'(\mathsf{ph}_{\mathcal{A}})(t) \stackrel{df}{=} p_1$ (4.4.4.28)

und es gilt

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg(\lceil \mathsf{ph}_{\mathcal{A}} \notin \{p_1, p_2\} \rceil ; \mathsf{true})$ $\qquad (= D_{init}(\overline{\mathcal{A}_E}))$
- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond(\lceil \mathsf{ph}_{\mathcal{A}} = p_1 \rceil ; \lceil \mathsf{ph}_{\mathcal{A}} = p_3 \vee \mathsf{ph}_{\mathcal{A}} = p_4 \rceil)$ $\qquad (= D^{p_1}_{suc}(\overline{\mathcal{A}_E}))$

  Zustand p_3 oder p_4 wird von $\mathcal{I}(\mathsf{ph}_{\mathcal{A}})$ nicht angenommen.

- $\mathcal{I}', \mathcal{V}' \models_{DC} \neg\Diamond\lceil \mathsf{ph}_{\mathcal{A}} = p_1 \wedge \mathsf{false} \rceil$ $\qquad (= D^{p_1}_{out}(\overline{\mathcal{A}_E}))$

und alle anderen Bedingungen sind erfüllt, weil außer p_1 kein anderer Zustand angenommen wird. Auch hier gilt dann:

$$\mathcal{I}, \mathcal{V} \models \exists\, x_{c\neg\pi} : F(\overline{\mathcal{A}_E}).$$

2.Fall (b) Ein Anfangsstück erfüllt $\neg\pi$ und irgendwann gilt π, d.h. analog zum 1. Fall existieren t_1, t_2, t_3 mit

$$\mathcal{I}, \mathcal{V}, [0, t_1] \models \lceil \neg\pi \rceil \tag{4.4.4.29}$$
$$\mathcal{I}, \mathcal{V}, (t_1, t_2] \models \lceil \pi \rceil. \tag{4.4.4.30}$$
$$\mathcal{I}, \mathcal{V}, (t_2, t_3] \models \lceil \neg\pi \rceil. \tag{4.4.4.31}$$
$$\tag{4.4.4.32}$$

und damit können $\mathcal{I}'$ und $\mathcal{V}'$ definiert werden durch

$$\bullet\, \mathcal{I}'(X) \stackrel{df}{=} \mathcal{I}(X) \quad \text{für alle Observablen } X \neq \mathsf{ph}_{\mathcal{A}} \tag{4.4.4.33}$$

$$\bullet\, \mathcal{I}'(\mathsf{ph}_{\mathcal{A}})(t) \stackrel{df}{=} \begin{cases} p_1 & \text{falls } t \in [0, t_1) \\ p_2 & \text{falls } t \in [t_1, t_2 \\ p_3 & \text{falls } t \in (t_2, t_3] \\ p_4 & \text{sonst} \end{cases} \tag{4.4.4.34}$$

$$\bullet\, \mathcal{V}'(x) \stackrel{df}{=} \begin{cases} \mathcal{V}(x) & \text{falls } x \neq x_{c\neg\pi} \\ (t_2 - t_1) & \text{falls } x = x_{c\neg\pi} \end{cases} \tag{4.4.4.35}$$

und analog zum 1. Fall gilt

$$\mathcal{I}, \mathcal{V} \models \exists x_{c\neg\pi} : F(\overline{\mathcal{A}_E}).$$

$\square$

3. Pattern

Der Automat $\mathcal{A}_S$ zu $\Diamond(\lceil \pi_1 \rceil \wedge b_1 \leq \ell \leq e_1; \ldots; \lceil \pi_n \rceil \wedge b_n \leq \ell \leq e_n)$ hat die in Abbildung 4.5 dargestellte Form. Sei x_{cwait} die neue, sonst nicht benutzte Variable für das Intervall $(0, \infty)$ zur Uhr c_{wait}. Der Automat $\mathcal{A}_Z$ erfüllt dann

$$\models (\exists\, x_{cwait} : F(\mathcal{A}_S)) \iff [\Diamond(\lceil \pi_1 \rceil \wedge b_1 \leq \ell \leq e_1; \ldots; \lceil \pi_n \rceil \wedge b_n \leq \ell \leq e_n)]$$

Der Beweis verläuft analog zum Beweis der entsprechenden Beziehung im 1. Pattern (erreichbarer Zustand).

3. Pattern (Komplement)

Da der nicht-komplementierbare Phasenautomat in Abschnitt 3.3.5 zu dem 3. Pattern passt, existiert nicht zu jedem Automaten zum Sequenz-Pattern ein Komplement. Aus diesem Grund wird einschränkend vorausgesetzt, dass jeweils zwei Zustandsausdrücke einander ausschließen, d.h.

$$\pi_i \wedge \pi_j \equiv \mathsf{false} \quad \text{für alle } 1 \leq i \neq j \leq n \tag{4.4.4.36}$$

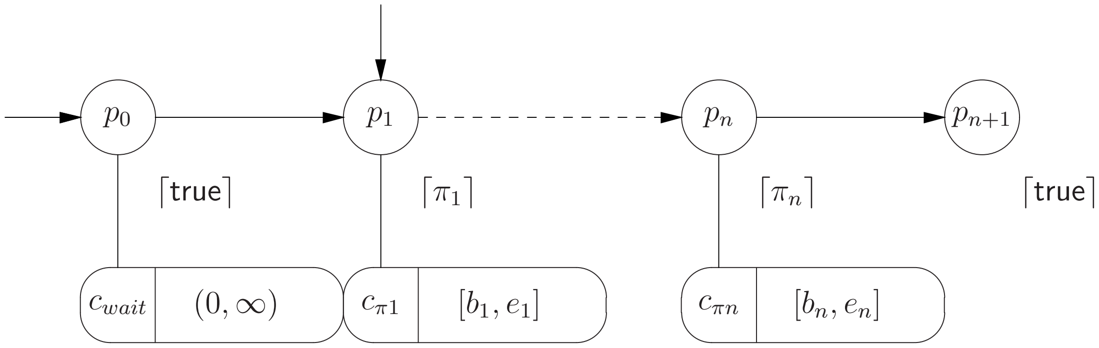


Abbildung 4.5: Automat $\mathcal{A}_S$ zum 3. Pattern (Sequenz)

Konstruktionsidee Der Automat $\overline{\mathcal{A}_S}$ soll alle Interpretationen akzeptieren, die nicht die angegebene Sequenz enthalten. Für jeden vorkommenden Zustandsausdruck π_i werden 4 Phasen in den zu konstruierende Phasenautomaten eingefügt.

- Eine Phase p_i, die angenommen werden soll, falls die Zusicherung π und die Uhrbedingung erfüllt sind, aber davor nicht die Sequenz bis π_{i-1} „abgearbeitet„ wurde. Deshalb gibt es eine Phase p_1 nicht.

- Eine Phase p_i^*, die angenommen werden soll, falls die Zusicherung π und die Uhrbedingung erfüllt sind und davor die Sequenz bis π_{i-1} „abgearbeitet„ wurde. Eine Phase p_n^* gibt es deswegen nicht, weil dann die Sequenz abgearbeitet wäre.

- Eine Phase $p_{i<}$, die angenommen werden soll, falls die Zusicherung π und die Uhrbedingung nicht erfüllt ist, weil die $\lceil \pi \rceil$-Phase zu kurz ist.

- Eine Phase $p_{i>}$, die angenommen werden soll, falls die Zusicherung π und die Uhrbedingung nicht erfüllt ist, weil die $\lceil \pi \rceil$-Phase zu lange ist.

Zusätzlich existiert eine Phase p_{sonst}, die angenommen wird, falls keine der Zusicherungen in der Sequenz erfüllt sind. Die Phasen werden so verbunden, dass keine Übergänge zwischen Phasen mit gleicher Zusicherung möglich sind und dass die Phase p_i^* genau dann angenommen wird, falls die Sequenz $\lceil \pi_1 \rceil; \ldots ; \lceil \pi_i \rceil$ durchlaufen wurde. Sonst sollen alle Übergänge möglich sein.

Konstruktion von $\overline{\mathcal{A}_S}$ Der Phasenautomat $\overline{\mathcal{A}_S}$ sei definiert durch

$$\overline{\mathcal{A}_S} = (P, E, C, cl, s, d, P_0)$$

$$P \stackrel{df}{=} P^1 \cup P^2 \cup P^3 \cup P^4 \cup \{p_{sonst}\} \tag{4.4.4.37}$$
$$P^1 \stackrel{df}{=} \{p_2, \dots p_n\} \tag{4.4.4.38}$$
$$P^2 \stackrel{df}{=} \{p_1^*, \dots p_{n-1}^*\} \tag{4.4.4.39}$$
$$P^3 \stackrel{df}{=} \{p_{1<}, \dots p_{n<}\} \tag{4.4.4.40}$$
$$P^4 \stackrel{df}{=} \{p_{1>}, \dots p_{n>}\} \tag{4.4.4.41}$$

$$E \stackrel{df}{=} E_1 \cup E_2 \cup E_3 \cup E_4 \cup E_5 \tag{4.4.4.42}$$
$$E_1 \stackrel{df}{=} \{p_i \rightarrow p_j | i \neq j\} \tag{4.4.4.43}$$
$$\cup \{p_i \rightarrow p_{j<} | i \neq j\} \tag{4.4.4.44}$$
$$\cup \{p_i \rightarrow p_{j>} | i \neq j\} \tag{4.4.4.45}$$
$$\cup \{p_i \rightarrow p_{sonst}\} \tag{4.4.4.46}$$
$$\cup \{p_i \rightarrow p_1^*\} \tag{4.4.4.47}$$
$$E_2 \stackrel{df}{=} \{p_{i<} \rightarrow p_j | i \neq j\} \tag{4.4.4.48}$$
$$\cup \{p_{i<} \rightarrow p_{j<} | i \neq j\} \tag{4.4.4.49}$$
$$\cup \{p_{i<} \rightarrow p_{j>} | i \neq j\} \tag{4.4.4.50}$$
$$\cup \{p_{i<} \rightarrow p_{sonst}\} \tag{4.4.4.51}$$
$$\cup \{p_{i<} \rightarrow p_1^*\} \tag{4.4.4.52}$$
$$E_3 \stackrel{df}{=} \{p_{i>} \rightarrow p_j | i \neq j\} \tag{4.4.4.53}$$
$$\cup \{p_{i>} \rightarrow p_{j<} | i \neq j\} \tag{4.4.4.54}$$
$$\cup \{p_{i>} \rightarrow p_{j>} | i \neq j\} \tag{4.4.4.55}$$
$$\cup \{p_{i>} \rightarrow p_{sonst}\} \tag{4.4.4.56}$$
$$\cup \{p_{i>} \rightarrow p_1^*\} \tag{4.4.4.57}$$
$$E_4 \stackrel{df}{=} \{p_i^* \rightarrow p_{i+1}^* | (i+1) < n\} \tag{4.4.4.58}$$
$$\cup \{p_i^* \rightarrow p_j | i \neq j \wedge i+1 \neq j\} \tag{4.4.4.59}$$
$$\cup \{p_i^* \rightarrow p_{j<} | i \neq j\} \tag{4.4.4.60}$$
$$\cup \{p_i^* \rightarrow p_{j>} | i \neq j\} \tag{4.4.4.61}$$
$$\cup \{p_i^* \rightarrow p_1^*\} \tag{4.4.4.62}$$
$$\cup \{p_i^* \rightarrow p_{sonst}\} \tag{4.4.4.63}$$
$$E_5 \stackrel{df}{=} \{p_{sonst} \rightarrow p_i\} \tag{4.4.4.64}$$
$$\cup \{p_{sonst} \rightarrow p_{i<}\} \tag{4.4.4.65}$$
$$\cup \{p_{sonst} \rightarrow p_{i>}\} \tag{4.4.4.66}$$
$$\cup \{p_{sonst} \rightarrow p_1^*\} \tag{4.4.4.67}$$

$$C \stackrel{df}{=} C^1 \cup C^2 \cup C^3 \cup C^4 \tag{4.4.4.68}$$

$$C^1 \stackrel{df}{=} \{cl_2, \dots cl_n\} \tag{4.4.4.69}$$

$$C^2 \stackrel{df}{=} \{cl_1^*, \dots cl_{n-1}^*\} \tag{4.4.4.70}$$

$$C^3 \stackrel{df}{=} \{cl_{1<}, \dots cl_{n<}\} \tag{4.4.4.71}$$

$$C^4 \stackrel{df}{=} \{cl_{1>}, \dots cl_{n>}\} \tag{4.4.4.72}$$

$$cl(p) \stackrel{df}{=} \begin{cases} c_i & \text{für } p = p_i \\ c_i^* & \text{für } p = p_i^* \\ c_{i<} & \text{für } p = p_{i<} \\ c_{i>} & \text{für } p = p_{i>} \end{cases} \tag{4.4.4.73}$$

$$d(cl) \stackrel{df}{=} \begin{cases} [b_i, e_i] & \text{für } c = c_i \\ [b_i, e_i] & \text{für } c = c_i^* \\ (0, b_i) & \text{für } c = c_{i<} \\ (e_1, \infty] & \text{für } c = c_{i>} \end{cases} \tag{4.4.4.74}$$

$$s(p) \stackrel{df}{=} \begin{cases} \pi_i & \text{für } p = p_i \\ \pi_i & \text{für } p = p_i^* \\ \pi_i & \text{für } p = p_{i<} \\ \pi_i & \text{für } p = p_{i>} \\ \neg \bigvee_{i=1}^n \pi_i & \text{für } p = p_{sonst} \end{cases} \tag{4.4.4.75}$$

$$P_0 \stackrel{df}{=} P_1 \cup P_3 \cup P_4 \cup \{p_1^*, p_{sonst}\} \tag{4.4.4.76}$$

Der so konstruierte Automat erfüllt dann

$$\models F(\overline{\mathcal{A}_S}) \iff \neg[\Diamond_L(\lceil \pi_1 \rceil \wedge b_1 \leq \ell \leq e_1; \dots ; \lceil \pi_n \rceil \wedge b_n \leq \ell \leq e_n)] \tag{4.4.4.77}$$

Beweis „⇒“-Richtung (Beweis durch Widerspruch) Gelte

$$\mathcal{I}, \mathcal{V} \models F(\overline{\mathcal{A}_S}) \tag{4.4.4.78}$$

und

$$\mathcal{I}, \mathcal{V} \models [\Diamond_L(\lceil \pi_1 \rceil \wedge b_1 \leq \ell \leq e_1; \dots ; \lceil \pi_n \rceil \wedge b_n \leq \ell \leq e_n)] \tag{4.4.4.79}$$

Sei $\mathcal{I}'$ die bis auf $\mathsf{ph}_{\mathcal{A}}$ mit $\mathcal{I}$ übereinstimmende Interpretation, die $D(\overline{\mathcal{A}_S})$ erfüllt. Nach Definition von $[\cdot]$ und weil in 4.4.4.79 die Observable $\mathsf{ph}_{\mathcal{A}}$ nicht vorkommt, gilt auch

$$\mathcal{I}', \mathcal{V}, [0,0] \models_{DC} \diamondsuit(\lceil \pi_1 \rceil \wedge b_1 \leq \ell \leq e_1; \ldots ; \lceil \pi_n \rceil \wedge b_n \leq \ell \leq e_n) \tag{4.4.4.80}$$

Wegen (4.4.4.80) gibt es $t_0, \ldots, t_n$, so dass

$$\begin{aligned} \mathcal{I}', \mathcal{V}, [t_0, t_1] &\models_{DC} \lceil \pi_1 \rceil \wedge b_1 \leq \ell \leq e_1 \\ &\ldots \\ \mathcal{I}', \mathcal{V}, [t_{n-1}, t_n] &\models_{DC} \lceil \pi_n \rceil \wedge b_n \leq \ell \leq e_n \end{aligned}$$

Da sich die Zusicherungen π_i paarweise ausschließen, sind $p_i, p_i^*, p_{i<}, p_{i>}$ die einzigen Phasen, die der Phasenautomat annehmen kann, wenn π_i erfüllt ist.

$$\begin{aligned} \Rightarrow \qquad \mathcal{I}', \mathcal{V}, [t_0, t_1] &\models_{DC} \lceil \mathsf{ph}_{\mathcal{A}} \in \{p_1^*, p_{1<}, p_{1>}\} \rceil \wedge b_1 \leq \ell \leq e_1 \\ \mathcal{I}', \mathcal{V}, [t_1, t_2] &\models_{DC} \lceil \mathsf{ph}_{\mathcal{A}} \in \{p_2, p_2^*, p_{2<}, p_{2>}\} \rceil \wedge b_2 \leq \ell \leq e_2 \\ &\ldots \\ \mathcal{I}', \mathcal{V}, [t_{n-1}, t_n] &\models_{DC} \lceil \mathsf{ph}_{\mathcal{A}} \in \{p_n, p_{n<}, p_{n>}\} \rceil \wedge b_n \leq \ell \leq e_n \end{aligned}$$

Es gibt keine Übergänge zwischen den Phasen $p_i, p_{i<}, p_{i>}$ und p_i^*.

$$\begin{aligned} \Rightarrow \qquad \mathcal{I}', \mathcal{V}, [t_0, t_1] &\models_{DC} \big(\lceil \mathsf{ph}_{\mathcal{A}} = p_1^* \rceil \vee \lceil \mathsf{ph}_{\mathcal{A}} = p_{1<} \rceil \vee \lceil \mathsf{ph}_{\mathcal{A}} = p_{1>} \rceil\big) \wedge b_1 \leq \ell \leq e_1 \\ \mathcal{I}', \mathcal{V}, [t_1, t_2] &\models_{DC} \big(\lceil \mathsf{ph}_{\mathcal{A}} = p_2^* \rceil \vee \lceil \mathsf{ph}_{\mathcal{A}} = p_2 \rceil \vee \lceil \mathsf{ph}_{\mathcal{A}} = p_{2<} \rceil \vee \lceil \mathsf{ph}_{\mathcal{A}} = p_{2>} \rceil\big) \\ &\quad \wedge b_2 \leq \ell \leq e_2 \\ &\ldots \\ \mathcal{I}', \mathcal{V}, [t_{n-1}, t_n] &\models_{DC} \big(\lceil \mathsf{ph}_{\mathcal{A}} = p_n \rceil \vee \lceil \mathsf{ph}_{\mathcal{A}} = p_{n<} \rceil \vee \lceil \mathsf{ph}_{\mathcal{A}} = p_{n>} \rceil\big) \wedge b_n \leq \ell \leq e_n \end{aligned}$$

Wegen der Uhrbedingungen können die Phasen $p_{i>}$ und $p_{i>}$ nicht angenommen werden.

$$\begin{aligned} \Rightarrow \qquad \mathcal{I}', \mathcal{V}, [t_0, t_1] &\models_{DC} \lceil \mathsf{ph}_{\mathcal{A}} = p_1^* \rceil \wedge b_1 \leq \ell \leq e_1 \\ \mathcal{I}', \mathcal{V}, [t_1, t_2] &\models_{DC} \big(\lceil \mathsf{ph}_{\mathcal{A}} = p_2^* \rceil \vee \lceil \mathsf{ph}_{\mathcal{A}} = p_2 \rceil\big) \\ &\quad \wedge b_2 \leq \ell \leq e_2 \\ &\ldots \\ \mathcal{I}', \mathcal{V}, [t_{n-1}, t_n] &\models_{DC} \lceil \mathsf{ph}_{\mathcal{A}} = p_n \rceil \wedge b_n \leq \ell \leq e_n \end{aligned}$$

Nach Konstruktion gibt es keine Übergänge von p_i^* nach p_{i+1}. Damit folgt der Widerspruch.

„⇐“-Richtung Gelte

$$\mathcal{I}, \mathcal{V} \models [\Diamond(\lceil \pi_1 \rceil \wedge b_1 \leq \ell \leq e_1; \ldots ; \lceil \pi_n \rceil \wedge b_n \leq \ell \leq e_n)]. \tag{4.4.4.81}$$

Sei $\mathcal{I}'$ eine Interpretation mit $\mathcal{I}' =_{\setminus \mathsf{ph}_\mathcal{A}} \mathcal{I}$. Der Lauf $\mathcal{I}'(\mathsf{ph}_\mathcal{A})$ des Phasenautomaten wird wieder induktiv über die Anzahl der Änderungen von $\mathcal{I}$ konstruiert. Seien dazu wieder $0 = t_0, t_1 \ldots$ die Unstetigkeitsstellen von $\mathcal{I}$ und $\mathrm{dur}_\mathcal{I}$ wie in Gleichung 3.4.3.7 die Funktion, die zu jedem Zeitpunkt angibt, wie lange $\mathcal{I}$ noch konstant bleibt.

Induktionsanfang Setze

$$\mathcal{I}'(\mathsf{ph}_\mathcal{A})(t) \overset{df}{=} \begin{cases} p_1^* & \text{falls } \mathcal{I}, \mathcal{V},\ [0, t_1] \models_{DC} \lceil \pi_1 \rceil \text{ und } b_1 \leq t_1 \leq e_1 \text{ und } 0 \leq t < t_1 \\ p_{1<} & \text{falls } \mathcal{I}, \mathcal{V},\ [0, t_1] \models_{DC} \lceil \pi_1 \rceil \text{ und } t_1 < b_1 \text{ und } 0 \leq t < t_1 \\ p_{1<} & \text{falls } \mathcal{I}, \mathcal{V},\ [0, t_1] \models_{DC} \lceil \pi_1 \rceil \text{ und } e_1 < t_1 \text{ und } 0 \leq t < t_1 \\ p_2 & \text{falls } \mathcal{I}, \mathcal{V},\ [0, t_1] \models_{DC} \lceil \pi_2 \rceil \text{ und } b_2 \leq t_1 \leq e_2 \text{ und } 0 \leq t < t_1 \\ p_{2<} & \text{falls } \mathcal{I}, \mathcal{V},\ [0, t_1] \models_{DC} \lceil \pi_2 \rceil \text{ und } t_1 < b_2 \text{ und } 0 \leq t < t_1 \\ p_{2<} & \text{falls } \mathcal{I}, \mathcal{V},\ [0, t_1] \models_{DC} \lceil \pi_2 \rceil \text{ und } e_2 < t_1 \text{ und } 0 \leq t < t_1 \\ \ldots & \\ p_{sonst} & \text{falls } \mathcal{I}, \mathcal{V},\ [0, t_1] \models_{DC} \neg \bigvee_{i=1}^{n} \lceil \pi_i \rceil \text{ und } 0 \leq t < t_1 \\ \text{undef.} & \text{sonst} \end{cases} \tag{4.4.4.82}$$

Mit dieser Definition gilt für alle $t \in [t_0, t_1)$:

$$\mathcal{I}', \mathcal{V}, [0, t] \models_{DC} D(\overline{\mathcal{A}_S})$$

und

$$\begin{aligned} & \mathcal{I}', \mathcal{V}, [0, t_1) \models_{DC} \lceil \mathsf{ph}_\mathcal{A} = p_i^* \rceil \\ \Longleftrightarrow\ & \mathcal{I}, \mathcal{V}, [0, t_1) \models_{DC} \Diamond(\lceil \pi_1 \rceil \wedge b_1 \leq \ell \leq e_1; \ldots ; \lceil \pi_i \rceil \wedge b_i \leq \ell \leq e_i) \end{aligned}$$

Induktionsschritt Sei $\mathcal{I}'$ schon so konstruiert, dass für alle $t \in [0, t_m)$ gilt

$$\mathcal{I}', \mathcal{V}, [0, t] \models_{DC} D(\overline{\mathcal{A}_S})$$

und

$$\begin{aligned} & \mathcal{I}', \mathcal{V}, [t_{m-1}, t_m) \models_{DC} \lceil \mathsf{ph}_\mathcal{A} = p_i^* \rceil \\ \Longleftrightarrow\ & \mathcal{I}, \mathcal{V}, [t_{m-1}, t_m) \models_{DC} \Diamond(\lceil \pi_1 \rceil \wedge b_1 \leq \ell \leq e_1; \ldots ; \lceil \pi_i \rceil \wedge b_i \leq \ell \leq e_i) \end{aligned}$$

1. Fall Der Automat ist im Intervall $[t_{m-1}, t_m)$ in einer mit *-gekennzeichneten Phase, d.h. es gibt ein i, so dass $\mathcal{I}', \mathcal{V}, [t_{m-1}, t_m) \models_{DC} \lceil \mathsf{ph}_{\mathcal{A}} = p_i^* \rceil$. Setze dann:

$$l \stackrel{df}{=} t_{m+1} - t_m$$

$$\hat{\mathcal{I}}(\mathsf{ph}_{\mathcal{A}})(t) \stackrel{df}{=} \begin{cases} \mathcal{I}'(\mathsf{ph}_{\mathcal{A}})(t) & \text{falls } t < t_m \\ p_1^* & \text{falls } \mathcal{I}, \mathcal{V}, [t_m, t_{m+1}] \models_{DC} \lceil \pi_1 \rceil \text{ und } b_1 \leq l \leq e_1 \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ p_{1<} & \text{falls } \mathcal{I}, \mathcal{V}, [t_m, t_{m+1}] \models_{DC} \lceil \pi_1 \rceil \text{ und } l < b_1 \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ p_{1<} & \text{falls } \mathcal{I}, \mathcal{V}, [t_m, t_{m+1}] \models_{DC} \lceil \pi_1 \rceil \text{ und } e_1 < l \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ p_2 & \text{falls } \mathcal{I}, \mathcal{V}, [t_m, t_{m+1}] \models_{DC} \lceil \pi_2 \rceil \text{ und } b_2 \leq l \leq e_2 \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ p_{2<} & \text{falls } \mathcal{I}, \mathcal{V}, [t_m, t_{m+1}] \models_{DC} \lceil \pi_2 \rceil \text{ und } l < b_2 \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ p_{2<} & \text{falls } \mathcal{I}, \mathcal{V}, [t_m, t_{m+1}] \models_{DC} \lceil \pi_2 \rceil \text{ und } e_2 < l \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ \ldots & \\ p_{i+1}^* & \text{falls } \mathcal{I}, \mathcal{V}, [t_m, t_{m+1}] \models_{DC} \lceil \pi_i \rceil \text{ und } b_2 \leq l \leq e_2 \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ p_{(i+1)<} & \text{falls } \mathcal{I}, \mathcal{V}, [t_m, t_{m+1}] \models_{DC} \lceil \pi_i \rceil \text{ und } l < b_2 \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ p_{(i+1)<} & \text{falls } \mathcal{I}, \mathcal{V}, [t_m, t_{m+1}] \models_{DC} \lceil \pi_i \rceil \text{ und } e_2 < l \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ \ldots & \\ p_{sonst} & \text{falls } \mathcal{I}, \mathcal{V}, [t_m, t_{m+1}] \models_{DC} \neg \bigvee_{i=1}^{n} \lceil \pi_i \rceil \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ \text{undef.} & \text{sonst} \end{cases}$$

Der Fall, dass $i = n_1$ gilt und auf dem Intervall $[t_n, t_{n+1}]$ die Zusicherung π_n erfüllt und die Länge l im Intervall $[b_n, e_n]$ liegt kann nicht eintreten, da dann zusammen mit der Induktionsvoraussetzung die gesamte Sequenz durchlaufen wäre.

2. Fall Der Automat ist im Intervall $[t_{m-1}, t_m)$ nicht in einer mit *-gekennzeichneten Phase. Setze dann:

$$l \stackrel{df}{=} t_{m+1} - t_m$$

$$\hat{\mathcal{I}}(\mathsf{ph}_{\mathcal{A}})(t) \stackrel{df}{=} \begin{cases} \mathcal{I}'(\mathsf{ph}_{\mathcal{A}})(t) & \text{falls } t < t_m \\ p_1^* & \text{falls } \mathcal{I}, \mathcal{V},\ [t_m, t_{m+1}] \models_{DC} \lceil \pi_1 \rceil \text{ und } b_1 \leq l \leq e_1 \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ p_{1<} & \text{falls } \mathcal{I}, \mathcal{V},\ [t_m, t_{m+1}] \models_{DC} \lceil \pi_1 \rceil \text{ und } l < b_1 \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ p_{1<} & \text{falls } \mathcal{I}, \mathcal{V},\ [t_m, t_{m+1}] \models_{DC} \lceil \pi_1 \rceil \text{ und } e_1 < l \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ p_2 & \text{falls } \mathcal{I}, \mathcal{V},\ [t_m, t_{m+1}] \models_{DC} \lceil \pi_2 \rceil \text{ und } b_2 \leq l \leq e_2 \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ p_{2<} & \text{falls } \mathcal{I}, \mathcal{V},\ [t_m, t_{m+1}] \models_{DC} \lceil \pi_2 \rceil \text{ und } l < b_2 \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ p_{2<} & \text{falls } \mathcal{I}, \mathcal{V},\ [t_m, t_{m+1}] \models_{DC} \lceil \pi_2 \rceil \text{ und } e_2 < l \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ \ldots & \\ p_{sonst} & \text{falls } \mathcal{I}, \mathcal{V},\ [t_m, t_{m+1}] \models_{DC} \neg \bigvee_{i=1}^{n} \lceil \pi_i \rceil \text{ und} \\ & \quad t_m \leq t < t_{m+1} \\ undef & \text{sonst} \end{cases}$$

Im Intervall $[t_m, t_{m+1})$ sind alle Uhr- und Ausgabedingungen erfüllt und der Phasenübergang von $\mathcal{I}(\mathsf{ph}_{\mathcal{A}})(t_{m-1})$ nach $\mathcal{I}(\mathsf{ph}_{\mathcal{A}})(t_m)$ ist ebenfalls erlaubt. Die Interpretation $\hat{\mathcal{I}}$ wird wieder $\mathcal{I}'$ genannt und es gilt dann wegen der Induktionsvoraussetzung auch für alle $t \in [0, t_{m+1})$:

$$\mathcal{I}', \mathcal{V}, [0, t] \models_{DC} D(\overline{\mathcal{A}_S})$$

und nach Konstruktion auch

$$\begin{aligned} & \mathcal{I}', \mathcal{V}, [t_m, t_{m+1}) \models_{DC} \lceil \mathsf{ph}_{\mathcal{A}} = p_i^* \rceil \\ \Longleftrightarrow\ & \mathcal{I}, \mathcal{V}, [t_m, t_{m+1}) \models_{DC} \Diamond_L (\lceil \pi_1 \rceil \wedge b_1 \leq \ell \leq e_1; \ldots ; \lceil \pi_i \rceil \wedge b_i \leq \ell \leq e_i) \end{aligned}$$

Damit folgt:

$$\mathcal{I}', \mathcal{V} \models_{DC} D(\overline{\mathcal{A}_S})$$

und damit auch

$$\mathcal{I}, \mathcal{V} \models F(\overline{\mathcal{A}_S}).$$

$\square$

4.4.3 Model-Checking für Gatter mit Ursache-Wirkung-Beziehung

Zur Beschreibung der Eingangsereignisse von Ursache-Wirkungs-Gattern, kommen sinnvollerweise nur Formeln infrage, die einen Zustand oder u.U. eine Folge von Zuständen beschreiben, da das übergeordnete Ereignis eintreten soll *nachdem* die Unterereignisse eingetreten sind. Eine Aussage, dass ein Ereignis eintritt *nachdem* der Zustand p angenommen und nie wieder verlassen wurde, macht keinen Sinn. Nachfolgend wird deshalb vorausgesetzt, dass alle vorkommenden Formeln einen Zustand beschreiben. Die Ergebnisse können aber auf Folgen von Zuständen verallgemeinert werden. Es wird also vorausgesetzt, dass alle DCL-Formeln, die in Eingangs- oder Ausgangsereignissen von Gattern mit Ursache-Wirkungs-Beziehungen vorkommen, die Form

$$A = \lceil \pi \rceil \wedge b \leq \ell \leq e$$

haben. Die <-Relation kann wieder analog behandelt werden. Für das Model-Checking dieser Gatter können die Konstruktionen zum 1. Pattern (erreichbarer Zustand) benutzt werden, so dass auf Beweise verzichtet werden kann und nur die Konstruktionsideen dargestellt werden.

AND-CC-Gatter

Korrektheit Die Korrektheitsbedingung hat die Form

$$\begin{aligned}\boxed{L}_0 A_{\mathcal{M}} \Rightarrow \Big(\big(\bigwedge_{i=1}^{n} \Diamond_L \lceil \pi^{(i)} \rceil \wedge b^{(i)} \leq \ell \leq e^{(i)} \big) \\ \Rightarrow \Diamond_L (\lceil \pi \rceil \wedge b \leq \ell \leq e \Big)\end{aligned}$$

Die Formeln entsprechen dem 1. Pattern und für das Model-Checking müssen dann nur wie in Abschnitt 4.4.2 beschrieben, Automaten zu

$$\Diamond_L \lceil \pi^{(i)} \rceil \wedge b^{(i)} \leq \ell \leq e^{(i)}$$

und zu

$$\neg \Diamond_L \lceil \pi \rceil \wedge b \leq \ell \leq e$$

konstruiert und zusammen mit der Modellierung auf Leerheit getestet werden.

Vollständigkeit Die Vollständigkeitsformel lautet

$$\boxed{L}_0 A_{\mathcal{M}} \Rightarrow \boxed{L}_0 \neg \left(\Big(\neg \bigwedge_{i=1}^{n} \Diamond (\lceil \pi^{(i)} \rceil \wedge e^{(i)} \leq \ell \leq b^{(i)}) \Big) ; \lceil \pi \rceil \wedge e \leq \ell \leq b \right).$$

Um diese Bedingung zu überprüfen, kann ein Nopa $\mathcal{N}_V$ zu

$$\left(\left(\neg \bigwedge_{i=1}^{n} \Diamond(\lceil \pi^{(i)} \rceil \wedge e^{(i)} \leq \ell \leq b^{(i)})\right); \lceil \pi \rceil \wedge e \leq \ell \leq b\right) \rhd \mathsf{true}$$

konstruiert und zusammen mit der Modellierung auf Leerheit getestet werden. Das Netz $\mathcal{N}_V$ akzeptiert alle Paare $(\mathcal{I}, \mathcal{V})$, die irgendwann π erfüllen und davor nicht alle Zustandsausdrücke $\pi_1, \ldots, \pi_n$ erfüllt haben und kann wie folgt konstruiert werden:

- Sei $\mathsf{Sync} : (Syn, noSyn)$ eine neue Observable. Sie wird zur Synchronisation von Phasenautomaten verwendet.
- Konstruiere Phasenautomaten $\overline{\mathcal{A}_1}, \ldots, \overline{\mathcal{A}_n}$ zu $\neg \Diamond\!\!\!\!\!\raisebox{0.1ex}{\tiny L}\;\lceil \pi^{(i)} \rceil \wedge e^{(i)} \leq \ell \leq b^{(i)}$ mit der für das 1. Pattern angegebenen Konstruktion.
- Konstruiere aus $\overline{\mathcal{A}_1}, \ldots, \overline{\mathcal{A}_n}$ den Oder-Automaten $\mathcal{A}_\vee$, der alle Paare $(\mathcal{I}, \mathcal{V})$ akzeptiert, die nicht irgendwann alle der Ausdrücke aus $\pi_1, \ldots, \pi_n$ mit zugehöriger Zeitbedingung erfüllt haben.
- Füge jeder Zustandszusicherung in $\mathcal{A}_\vee$ die Bedingung $\wedge Syn$ hinzu.
- Füge $\mathcal{A}_\vee$ den Zustand p_{true} mit Zusicherung $noSyn$ hinzu und Transitionen von jedem anderem Zustand zu p_{true}. Füge p_{true} den Startzuständen hinzu. Der so konstruierte Automat erlaubt, dass Syn gilt, solange zumindestens eine Zusicherung von $\pi_1, \ldots, \pi_n$ und die zugehörige Zeitbedingung noch nicht erfüllt ist.
- Konstruiere $\mathcal{A}_\pi$ zu

  $$(\lceil Syn \rceil \vee \lceil\rceil; \lceil \pi \rceil \wedge b \leq \ell \leq e \wedge \lceil noSyn \rceil; \lceil noSyn \rceil) \rhd \mathsf{true}.$$

  wie in Abbildung 4.6 dargestellt.

Das so konstruierte Netzwerk $\mathcal{N}_V \stackrel{df}{=} \mathcal{A}_\vee \| \mathcal{A}_\pi$ akzeptiert dann alle Paare $(\mathcal{I}, \mathcal{V})$, die irgendwann $\lceil \pi \rceil \wedge b \leq \ell \leq e$ erfüllen und davor mindestens eine Bedingung $\lceil \pi_i \rceil \wedge b_i \leq \ell \leq e_i$ $1 \leq i \leq n$ nicht.

AND-CC-Syn-Gatter

Korrektheit Bei Gattern mit synchroner Ursache-Wirkung-Beziehung lautet die Korrektheitsformel

$$\boxed{L}_0 A_{\mathcal{M}} \Rightarrow \Big(\Diamond_L (\bigwedge_{i=1}^{n} \lceil \pi^{(i)} \rceil \wedge b^{(i)} \leq \ell \leq e^{(i)}) \\ \Rightarrow \Diamond_L (\lceil \pi \rceil \wedge b \leq \ell \leq e \Big)$$

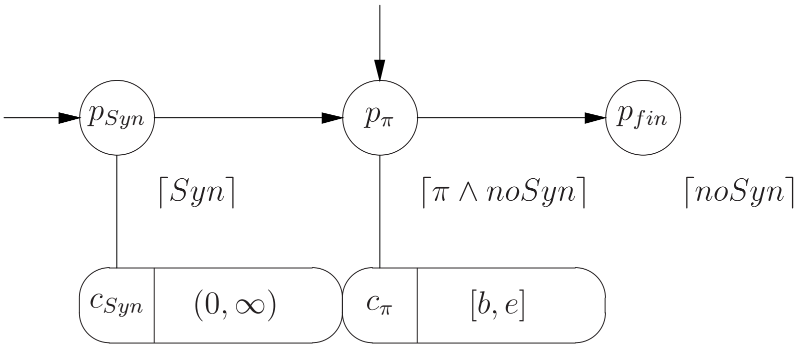


Abbildung 4.6: Automat $\mathcal{A}_\pi$ (AND-CC-Gatter)

Für das Model-Checking müssen dann nur Phasenautomaten zu

$$\Diamond_{L}\Big(\bigwedge_{i=1}^{n} \lceil \pi^{(i)} \rceil \wedge b^{(i)} \leq \ell \leq e^{(i)} \Big)$$

und

$$\neg \Diamond_{L} \lceil \pi \rceil \wedge b \leq \ell \leq e$$

konstruiert und zusammen mit der Modellierung auf Leerheit getestet werden. Beide Formeln entsprechen dem 1. Pattern (erreichbarer Zustand), so dass die dafür entwickelten Konstruktionen benutzt werden können. Dabei kann es vorkommen, dass die Bedingung $\bigwedge_{i=1}^{n} \lceil \pi^{(i)} \rceil \wedge b^{(i)} \leq \ell \leq e^{(i)}$ unerfüllbar und damit äquivalent zu false ist. In diesem Fall ist die Zerlegung unsinnig, weil die Ereignisse nicht zusammen eintreten können. Die Korrektheit kann dann natürlich auch nicht formal gezeigt werden.

Vollständigkeit Auch der Ansatz für das Model-Checking der Vollständigkeitsbedingung

$$\boxed{L}_0 A_{\mathcal{M}} \Rightarrow \boxed{L}_0 \neg \left(\left(\neg \Diamond \bigwedge_{i=1}^{n} (\lceil \pi^{(i)} \rceil \wedge e^{(i)} \leq \ell \leq b^{(i)}) \right); \lceil \pi \rceil \wedge e \leq \ell \leq b \right).$$

entspricht dem Ansatz für asynchrone AND-CC-Gatter. Es wird Nopa $\mathcal{N}_V^{(Syn)}$ zu

$$\left(\left(\neg \Diamond \bigwedge_{i=1}^{n} (\lceil \pi^{(i)} \rceil \wedge e^{(i)} \leq \ell \leq b^{(i)}) \right); \lceil \pi \rceil \wedge e \leq \ell \leq b \right) \rhd \mathsf{true}$$

konstruiert und das Netzwerk aus Modellierung und $\mathcal{N}_V^{(Syn)}$ auf Leerheit überprüft. Die Konstruktion von $\mathcal{N}_V^{(Syn)}$ ist etwas einfacher als die von $\mathcal{N}_V$ für AND-CC-Gatter.

- Sei Sync : $(Syn, noSyn)$ wieder eine neue Observable, die zur Synchronisation von Phasenautomaten verwendet wird.

- Konstruiere einen Phasenautomaten $\overline{\mathcal{A}^{(1)}}$ zu $\neg\Diamond \bigwedge_{i=1}^{n} \lceil \pi^{(i)} \rceil \wedge b^{(i)} \leq \ell \leq e^{(i)}$ mit der für das 1. Pattern angegebenen Konstruktion.

- Füge jeder Zustandszusicherung in $\mathcal{A}_1$ die Bedingung $\wedge Syn$ hinzu.

- Füge $\mathcal{A}^{(1)}$ den Zustand p_{true} mit Zusicherung $noSyn$ hinzu und Transitionen von jedem anderem Zustand zu p_{true}. Füge p_{true} den Startzuständen hinzu. Der so konstruierte Automat erlaubt, dass Syn gilt, solange nicht gleichzeitig alle Zusicherungen aus $\pi_1, \ldots, \pi_n$ samt zugehöriger Zeitbedingung erfüllt sind.

- Konstruiere $\mathcal{A}_\pi$ wie für AND-CC-Gatter zu

$$(\lceil Syn \rceil \vee \lceil \rceil; \pi \wedge b \leq \ell \leq e \wedge \lceil noSyn \rceil ; \lceil noSyn \rceil) \rhd \mathsf{true}.$$

Diese Automaten bilden das Netz $\mathcal{N}_V^{(Syn)} \overset{df}{=} \mathcal{A}^{(1)} \| \mathcal{A}_\pi$.

OR-CC-Gatter

Das Vorgehen bei OR-CC-Gattern ist wieder ganz analog.

Korrektheit Die Korrektheitsformel

$$\boxed{L}_0 A_{\mathcal{M}} \Rightarrow \Big((\bigvee_{i=1}^{n} \Diamond \lceil \pi^{(i)} \rceil \wedge b^{(i)} \leq \ell \leq e^{(i)}) \\ \Rightarrow \Diamond (\lceil \pi \rceil \wedge b \leq \ell \leq e \Big)$$

wird durch den Leerheitstest von Modellierung und den Automaten zu

$$\bigvee_{i=1}^{n} \Diamond \lceil \pi^{(i)} \rceil \wedge b^{(i)} \leq \ell \leq e^{(i)}$$

und zu

$$\neg \Diamond \lceil \pi \rceil \wedge b \leq \ell \leq e$$

verifiziert. Der Automat zu $\neg\Diamond \lceil \pi \rceil \wedge b \leq \ell \leq e$ wird wie für das 1. Pattern beschrieben konstruiert. Für anderen Automaten werden die Automaten zu $\Diamond \lceil \pi^{(i)} \rceil \wedge b^{(i)} \leq \ell \leq e^{(i)}$ $(1 \leq i \leq n)$ durch die Oder-Konstruktion vereinigt.

Vollständigkeit Die Vollständigkeitsformel für OR-CC-Gatter lautet

$$\boxed{L}_0 A_{\mathcal{M}} \Rightarrow \boxed{L}_0 \neg \left(\left(\neg \bigvee_{i=1}^{n} \Diamond(\lceil \pi^{(i)} \rceil \wedge e^{(i)} \leq \ell \leq b^{(i)}) \right); \lceil \pi \rceil \wedge e \leq \ell \leq b \right).$$

Um die Bedingung zu überprüfen, kann ein Nopa $\mathcal{N}_V^{ODER}$ zu

$$\left(\left(\neg \bigvee_{i=1}^{n} \Diamond(\lceil \pi^{(i)} \rceil \wedge e^{(i)} \leq \ell \leq b^{(i)}) \right); \lceil \pi \rceil \wedge e \leq \ell \leq b \right) \rhd \mathsf{true}$$

konstruiert und zusammen mit der Modellierung auf Leerheit getestet werden. Das Netz $\mathcal{N}_V^{ODER}$ akzeptiert alle Paare $(\mathcal{I}, \mathcal{V})$, die irgendwann π erfüllen und davor keinen der Zustandsausdrücke $\pi_1, \ldots, \pi_n$ und die Zeitbedingung erfüllt haben. Es kann wie folgt konstruiert werden:

- Sei $\mathsf{Sync} : (Syn, noSyn)$ wieder eine neue Synchronisations-Observable.
- Konstruiere Phasenautomaten $\overline{\mathcal{A}_1}, \ldots, \overline{\mathcal{A}_n}$ zu $\neg \Diamond\!\!\!\!\raisebox{0.1ex}{\tiny L}\ \lceil \pi^{(i)} \rceil \wedge e^{(i)} \leq \ell \leq b^{(i)}$ mit der für das 1. Pattern angegebenen Konstruktion.
- Konstruiere aus $\overline{\mathcal{A}_1}, \ldots, \overline{\mathcal{A}_n}$ den Produkt-Automaten $\mathcal{A}_{\|}$, der alle Paare $(\mathcal{I}, \mathcal{V})$ akzeptiert, die keinen der Ausdrücke aus $\pi_1, \ldots, \pi_n$ mit zugehöriger Zeitbedingung erfüllt hat.
- Füge jeder Zustandszusicherung in $\mathcal{A}_{\|}$ die Bedingung $\wedge Syn$ hinzu.
- Füge $\mathcal{A}_{\|}$ den Zustand p_{true} mit Zusicherung $noSyn$ hinzu und Transitionen von jedem anderem Zustand zu p_{true}. Füge p_{true} den Startzuständen hinzu. Der so konstruierte Automat erlaubt, dass Syn gilt, solange keine Zusicherung aus $\pi_1, \ldots, \pi_n$ und die zugehörige Zeitbedingung erfüllt sind.
- Konstruiere $\mathcal{A}_\pi$ zu wie in den anderen Fällen als Automat zu

  $$(\lceil Syn \rceil \vee \lceil \rceil; \pi \wedge b \leq \ell \leq e \wedge \lceil noSyn \rceil; \lceil noSyn \rceil) \rhd \mathsf{true}.$$

  wie in Abbildung 4.6 dargestellt.

Aus diesem Automaten wird das Netzwerk $\mathcal{N}_V^{ODER} \overset{df}{=} \mathcal{A}_{\|} \| \mathcal{A}_\pi$ konstruiert.

5 Beispiele

5.1 Pressure-Tank

Das Fallbeispiel *Pressure-Tank* stammt aus [VGRH81] und wurde am Lehrstuhl für Softwaretechnik und Programmiersprachen an der Universität Augsburg ([Thu01]) mit Hilfe des Model-Checkers **Raven** für diskrete Zeit bearbeitet. Hier soll das Beispiel mit MOBY/DC für kontinuierliche Zeit untersucht werden.

5.1.1 Beschreibung des Systems

In Abbildung 5.1 ist ein Kontrollsystem für eine Drucktank-Anlage dargestellt. Der

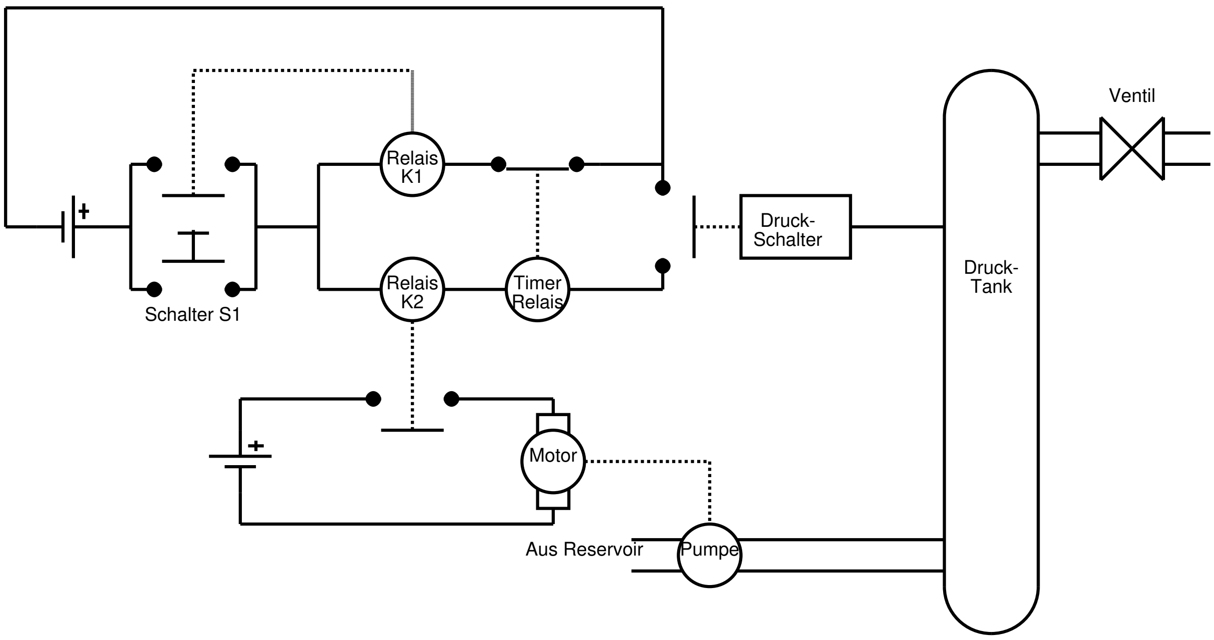


Abbildung 5.1: Schaltbild: Pressure Tank System

Drucktank wird durch die Pumpe innerhalb von 60 Sekunden unter Druck gesetzt und durch das Ventil in vernachlässigbarer Zeit wieder geleert. Es wird angenommen, dass

die Pumpe die Flüssigkeit aus einem unendlich großen Reservoir bezieht. Solange der Drucktank noch nicht vollständig gefüllt ist, sind die Kontakte des Druck-Schalters geschlossen und sobald der gewünschte Druck erreicht ist, öffnen sie sich. Dadurch wird die Spule von Relais K2 vom Strom getrennt, das Relais öffnet sich und die Pumpe hört auf zu arbeiten. Wenn der Tank geleert ist, schließen sich die Kontakte des Druckschalters wieder und der Zyklus beginnt erneut.
Initial befindet sich das System im Ruhezustand:

- Die Kontakte des Schalters S1 sind offen.
- Die Kontakte von Relais K1 und Relais K2 sind offen.
- Es fließt kein Strom durch das Kontrollsystem.
- Die Kontakte des Timer-Relais sind geschlossen.
- Der Tank ist leer und deshalb sind die Kontakte des Druckschalters ebenfalls geschlossen.

Das System wird durch kurzzeitiges Schließen von Schalter S1 gestartet. Dadurch wird der Stromkreis geschlossen, die Spule von K1 aktiviert und das Relais K1 geschlossen. Dadurch sorgt K1 selbst dafür, dass seine Spule mit Strom versorgt wird. Außerdem wird auch die Spule von K2 unter Strom gesetzt, so dass K2 schließt und die Pumpe anfängt zu arbeiten.
Das Timer-Relais dient als redundantes System, falls der Druck-Schalter ausfällt. Sobald das Relais 60 Sekunden kontinuierlichen Stromfluß misst, werden die Kontakte des Relais geöffnet, die Spule von K1 vom Strom getrennt. Damit öffnen sich die Kontakte von K1 und der Strom an K2 wird abgeschaltet, so dass die Pumpe stoppt.
Für die Analyse wird in [VGRH81] angenommen, dass der Tank in jedem Fall berstet, wenn die Pumpe länger als 60 Sekunden arbeitet.

5.1.2 Modellierung des Systems

Damit der Fehlerbaum mit einem Model-Checker überprüft werden kann, muss ein formales Modell des Systems *Pressure Tank* existieren. Das Modell besteht hier aus einer Menge von Phasenautomaten, die das Verhalten des Systems beschreiben. Die Modellierung durch Phasenautomaten berücksichtigt auch den Ausfall einzelner Komponenten, erlaubt aber keine „Selbstheilung“ gestörter Komponenten, d.h. eine einmal ausgefallene Komponente bleibt ausgefallen.

Modellierung Drucktank

Die Modellierung des Drucktanks selber ist in Abbildung 5.2 dargestellt. Die Observablen $\mathsf{tank} : (empty, fill, full, rupture)$ und $\mathsf{pipe} : (flow, noflow)$ beschreiben den

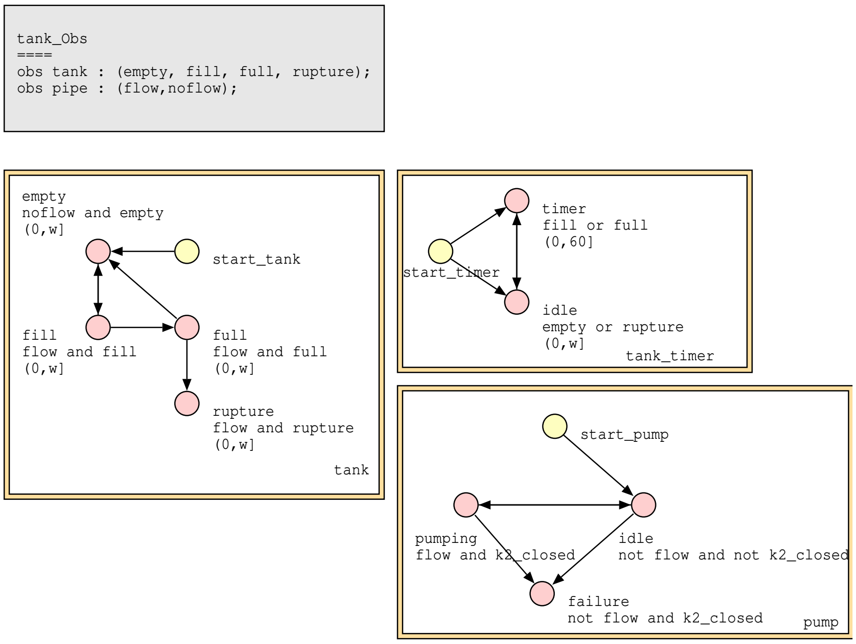


Abbildung 5.2: Modellierung durch Phasenautomaten (Drucktank)

Zustand des Tanks und ob Flüssigkeit in den Tank fließt. Der Phasenautomat **tank** modelliert, dass der Tank gefüllt wird, wenn Flüssigkeit durch die Zuleitung fließt. Durch **tank_timer** wird sichergestellt, dass nur 60 ZE kontinuierlich Flüssigkeit in den Tank gepumpt werden kann. Danach darf keine Flüssigkeit mehr in den Tank fließen oder der Tank muss bersten. Der Automat modelliert nicht das Timer Relais. Da hier kontinuierliche Zeit verwendet wird, kann die zwar sehr theoretische Annahme, dass der Tank nach 60 ZE kontinuierlichen Zuflusses sicher berstet, direkt modelliert werden. Das Verhalten der Pumpe wird durch den Automaten **pump** bestimmt. Wenn der untere Stromkreis geschlossen ist, sorgt die Pumpe für Zufluss ($flow$) in den Tank oder die Pumpe ist ausgefallen und pumpt nicht. Der Fall, dass kein Strom fließt, aber die Pumpe trotzdem arbeitet, ist ausgeschlossen, da er in der Realität nicht vorkommen kann. Da Verkabelungs- und Leitungsausfälle ausgeschlossen sind, ist der untere Stromkreis genau dann geschlossen, wenn das Relais K2 geschlossen ist. Das wird durch die Observable $\mathsf{k2} : (k2\_open, k2\_close)$ modelliert.

Modellierung Relais K2

Die Phasenautomaten in Abbildung 5.3 modellieren das Verhalten des Relais K2. Ob das Relais offen oder geschlossen ist, wird durch die Observable $\mathsf{k2} : (k2\_open, k2\_close)$ und den Phasenautomaten **k2** bestimmt. Das Relais kann auch ausfallen, d.h. obwohl kein elektromagnetisches Feld an der Spule anliegt, verbleibt das Relais in ge-

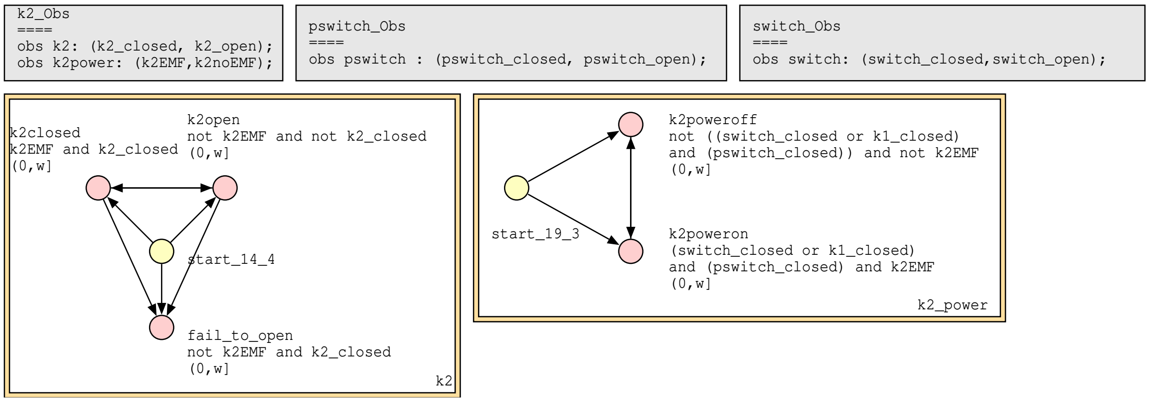


Abbildung 5.3: Modellierung durch Phasenautomaten (Relais K2)

schlossenem Zustand. Aus diesem Grund wird zusätzlich die Observable k2power: $(k2\_EMF, k2\_noEMF)$ verwendet, die beschreibt, ob ein elektromagnetische Feld (EMF) an der Spule des Relais anliegt oder nicht. Der Automat **k2_power** modelliert, dass an Relais K2 genau dann ein elektromagnetisches Feld anliegt, wenn der Druckschalter geschlossen (Observable pswitch) und Relais K1 oder der Schalter S1 geschlossen sind (Observablen k1, switch)

Modellierung Relais k1

Das Relais K1 ist ganz analog zum Relais K2 modelliert durch zwei Phasenautomaten **k1** und **k1_power** modelliert (Abbildung 5.5).

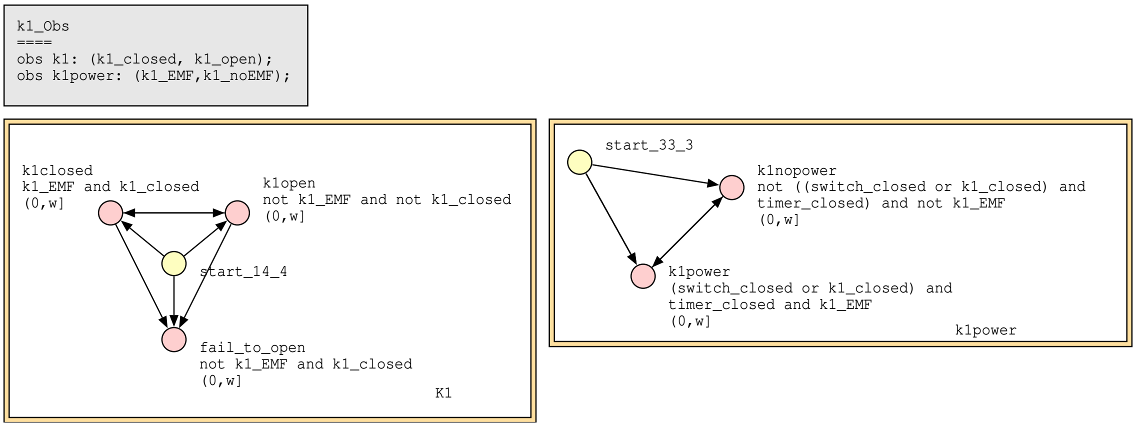


Abbildung 5.4: Modellierung durch Phasenautomaten (Relais K1)

Modellierung des Timer Relais

Das Zustand des Timer Relais wird durch die Observable timer beschrieben und ob das Timer Relais korrekt arbeitet, wird durch die Observable timer_ok beschrieben.

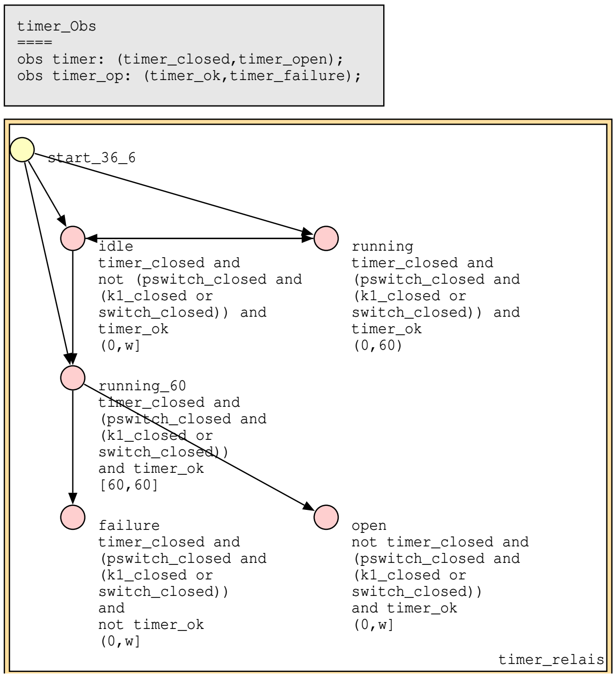


Abbildung 5.5: Modellierung durch Phasenautomaten (Timer Relais)

Die möglichen Werte von timer sind *timer_open* und *timer_closed* und von timer_ok *timer_ok* und *timer_failure*. Der Automat **timer_relais** beschreibt das normale und das fehlerhafte Verhalten des Timer Relais. Wenn der obere Stromkreis nach 60 ZE noch geschlossen ist, muss entweder das Timer Relais öffnen oder das Relais ist fehlerhaft.

5.1.3 Fehlerbaum-Analyse

In [VGRH81] und [Thu01] wurde eine Fehlerbaumanalyse des Systems für das Ereignis „Tank berstet“ durchgeführt. Es wird hier die Analyse aus [Thu01] übernommen. Dabei werden die Ereignisse mit dem Buchstaben E und einer Nummer und die Gatter mit dem Buchstaben G und einer Nummer beschriftet. Die Numerierung wurde ebenfalls aus [Thu01] übernommen. Zusätzlich wird hier jedes Ereignis formal durch eine DCL-Formel beschrieben.

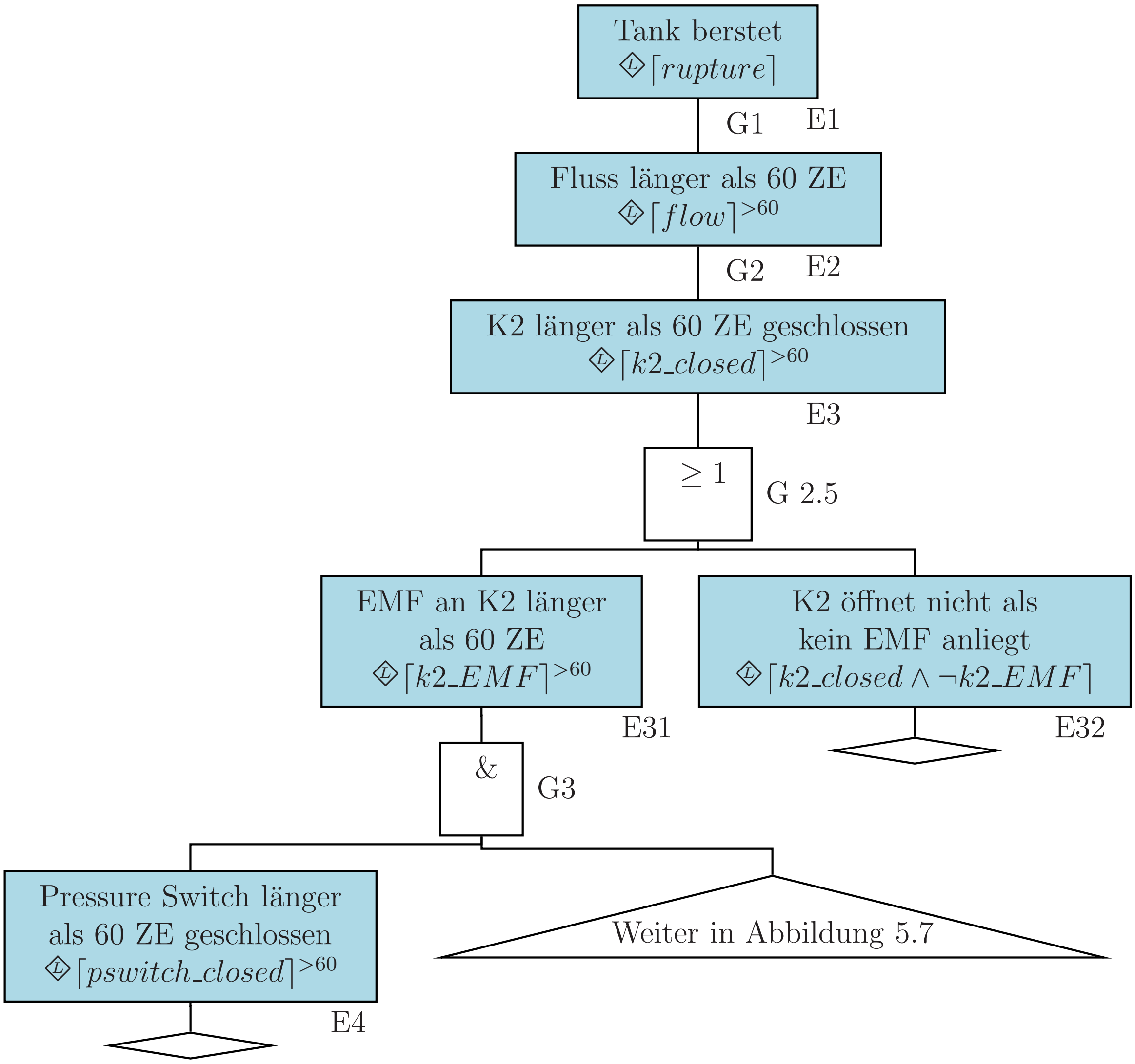


Abbildung 5.6: Fehlerbaum zum Beispiel Pressure Tank (Teil 1)

Definition Sei $\pi_{Cl} \stackrel{df}{=} pswitch\_closed \wedge (switch\_closed \vee k1\_closed)$.
π_{Cl} beschreibt, dass der obere Stromkreis geschlossen ist.

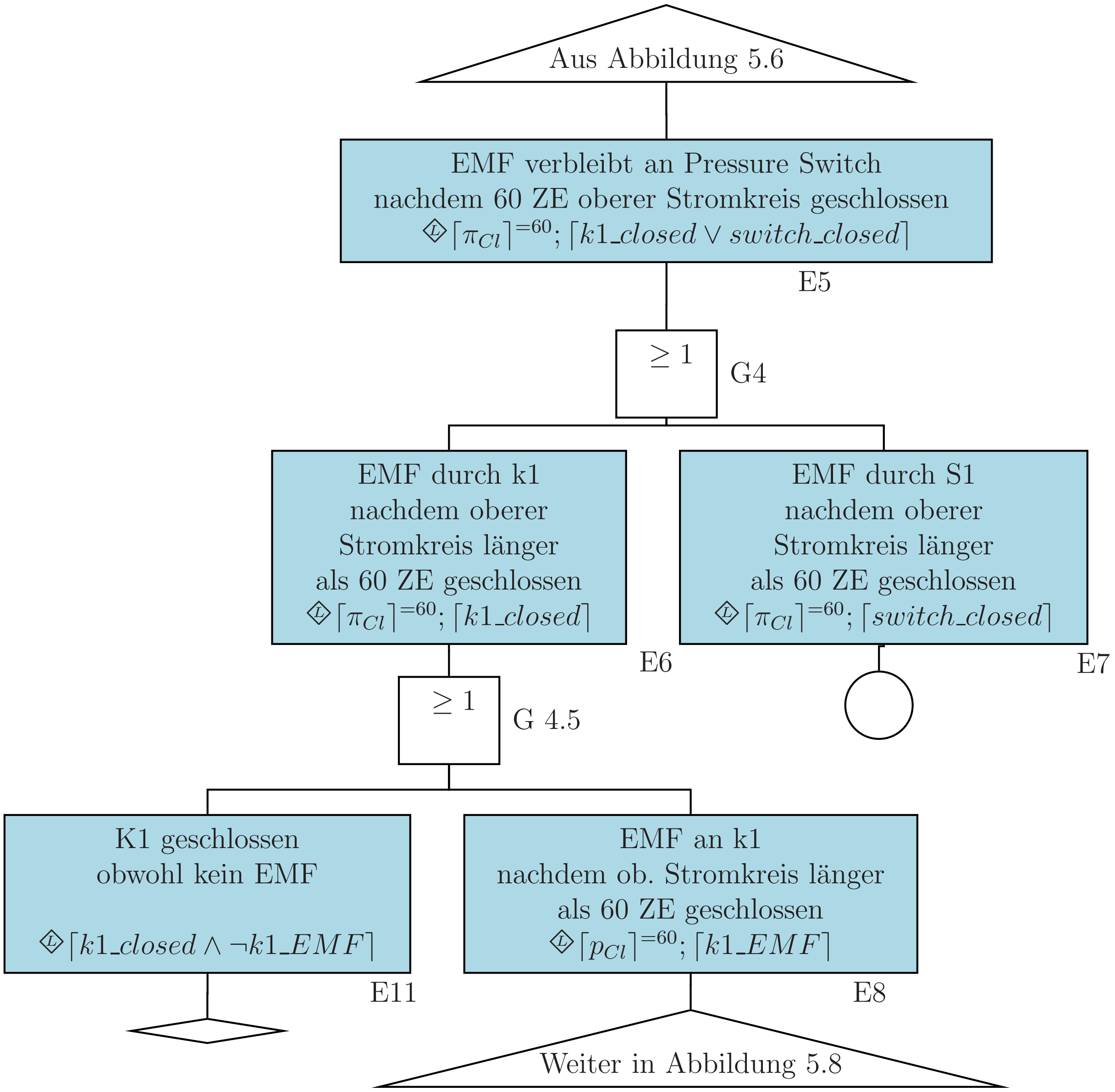


Abbildung 5.7: Fehlerbaum zum Beispiel Pressure Tank (Teil 2)

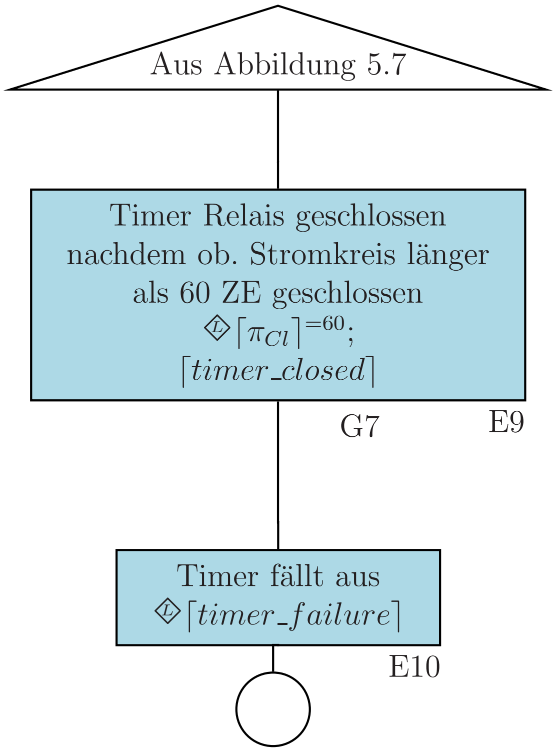
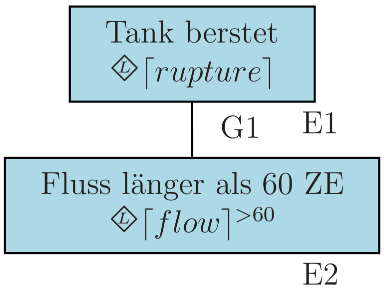


Abbildung 5.8: Fehlerbaum zum Beispiel Pressure Tank (Teil 3)

5.1.4 Model-Checking mit Moby/DC

Für den dargestellten Fehlerbaum wird gezeigt, dass keine Ereignisse vergessen worden sind, d.h. dass der Fehlerbaum vollständig ist. Die Korrektheitsbedingung ist nicht für jedes Gatter zu zeigen. Es wird dazu die Vollständigkeitsbedingung und falls möglich auch auch die Korrektheitsbedingung für jedes Gatter gezeigt. Für jedes einzelne Gatter werden dabei jeweils nur ein oder zwei Phasenautomaten benötigt. Dadurch ist die Model-Checking-Komplexität für jedes Gatter gering.

Gatter G1



Für das Gatter G1 kann sowohl die Vollständigkeitsbedingung als auch die Korrektheitsbedingung gezeigt werden. Für die Vollständigkeit wird

$$\mathbf{E1 \| neg\_E2 \| tank}$$

auf Leerheit geprüft, d.h. dass es keinen Ablauf gibt, der zwar die Modellierung und E1 erfüllt aber E2 nicht. Damit ist die Implikation

$$\Diamond\lceil rupture \rceil \Rightarrow \Diamond\lceil flow \rceil^{>60}$$

für diese Modellierung gezeigt. Die Automaten sind in Abbildung 5.9 dargestellt. Dabei wird nicht die ganze Modellierung, sondern nur der Phasenautomat **tank** benötigt. Die Korrektheitsbedingung

$$\Diamond\lceil flow \rceil^{>60} \Rightarrow \Diamond\lceil rupture \rceil$$

kann durch den Leerheitstest von

$$\mathbf{neg\_E1 \| E2 \| tank \| tank\_timer}$$

verifiziert werden. Der Automat **tank_timer** wird zusätzlich benötigt, um sicherzustellen, dass der Tank nach 60 ZE auch tatsächlich berstet. Da die DCL-Formeln $\Diamond\lceil rupture \rceil$ und $\Diamond\lceil flow \rceil^{>60}$ dem Zustands-Pattern entsprechen, sind die Phasenautomaten auch diesem Pattern entsprechend konstruiert.

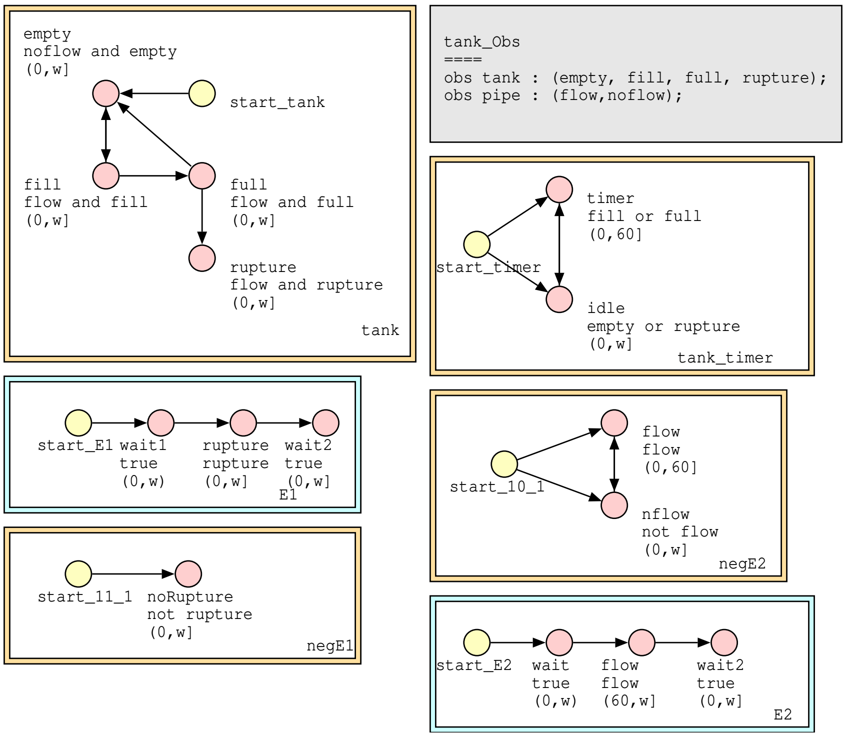
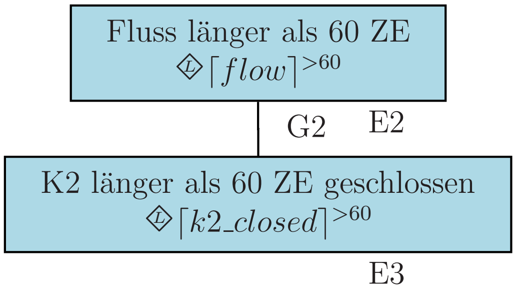


Abbildung 5.9: Phasenautomaten zu Gatter G1

Gatter G2

Die Bedingung *K2 länger als 60 ZE geschlossen* ist nur notwendig aber nicht hinreichend dafür, dass länger als 60 ZE der Tank befüllt wird. Es kann der Fall eintreten, dass das Relais K2 zwar länger als 60 ZE geschlossen ist, aber in dieser Zeit glücklicherweise auch die Pumpe ausfällt, so dass der Tank keine 60 ZE lang befüllt wird.



Die Vollständigkeitsformel

$$\Diamond\lceil flow \rceil^{>60} \Rightarrow \Diamond\lceil k2\_closed \rceil^{>60}$$

kann dann durch einen Leerheitstest der in Abbildung 5.10 dargestellten Automaten

$$\mathbf{E2 \| neg\_E3 \| pump}$$

verifiziert werden. Auch die Formel $\Diamond\lceil k2\_closed \rceil^{>60}$ entspricht dem Zustands-Pattern und der Automat **neg_E_3** ist nach der für dieses Pattern entwickelten Methode konstruiert.

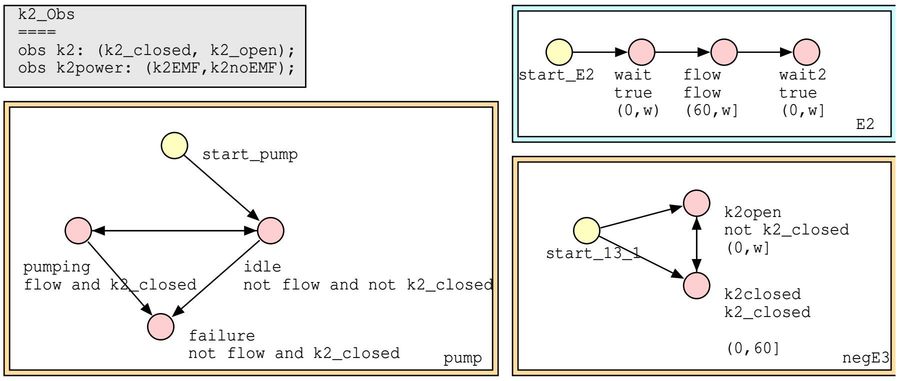
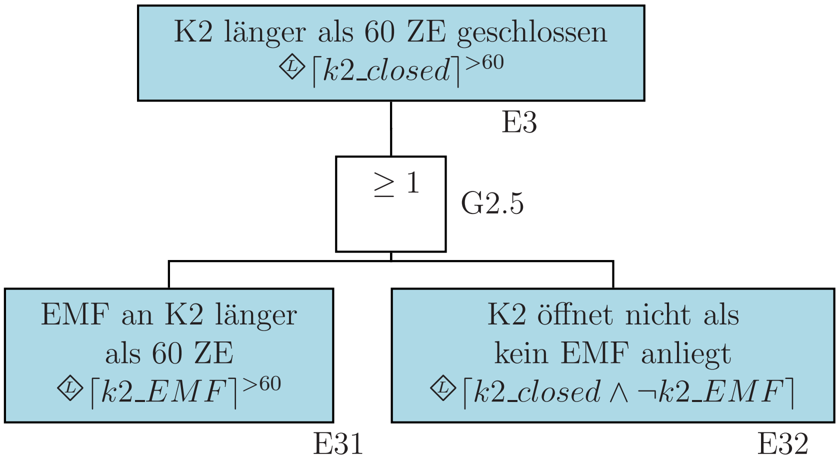


Abbildung 5.10: Phasenautomaten zu Gatter G2

Gatter G2.5



Für dieses Gatter können wieder sowohl Vollständigkeit als auch Korrektheit gezeigt werden. Für die Vollständigkeitsbedingung

$$\Diamond\lceil k2\_closed \rceil^{>60} \Rightarrow (\Diamond\lceil k2\_EMF \rceil^{>60} \vee \Diamond\lceil k2\_closed \wedge \neg k2\_EMF \rceil)$$

werden die in Abbildung 5.11 dargestellten Automaten

E3||neg_E31||neg_E32||K2

konstruiert und auf Leerheit getestet.
Für die Korrektheitsbedingung

$$(\Diamond\lceil k2\_EMF \rceil^{>60} \vee \Diamond\lceil k2\_closed \wedge \neg k2\_EMF \rceil) \Rightarrow \Diamond\lceil k2\_closed \rceil^{>60}$$

werden die ebenfalls in Abbildung Abbildung 5.11 dargestellten Automaten

neg_E3||E31_or_E32||K2

auf Leerheit überprüft. Der Automat **E31_or_E32** wird mit Hilfe der Automaten zu Zustands-Pattern und der Oder-Konstruktion konstruiert.

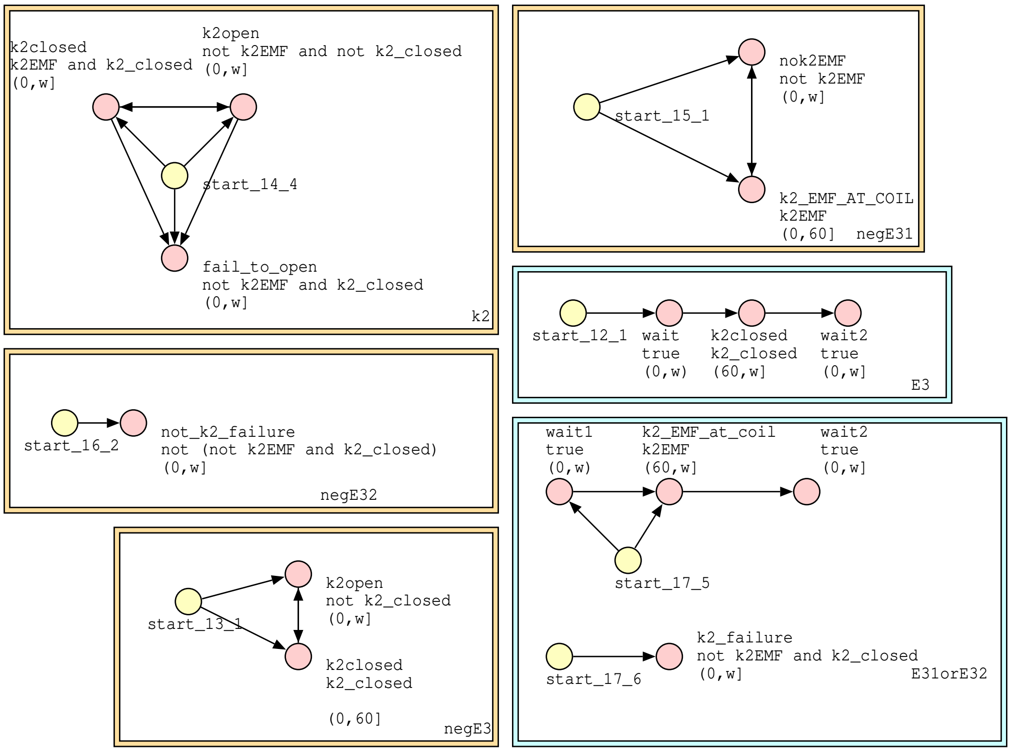
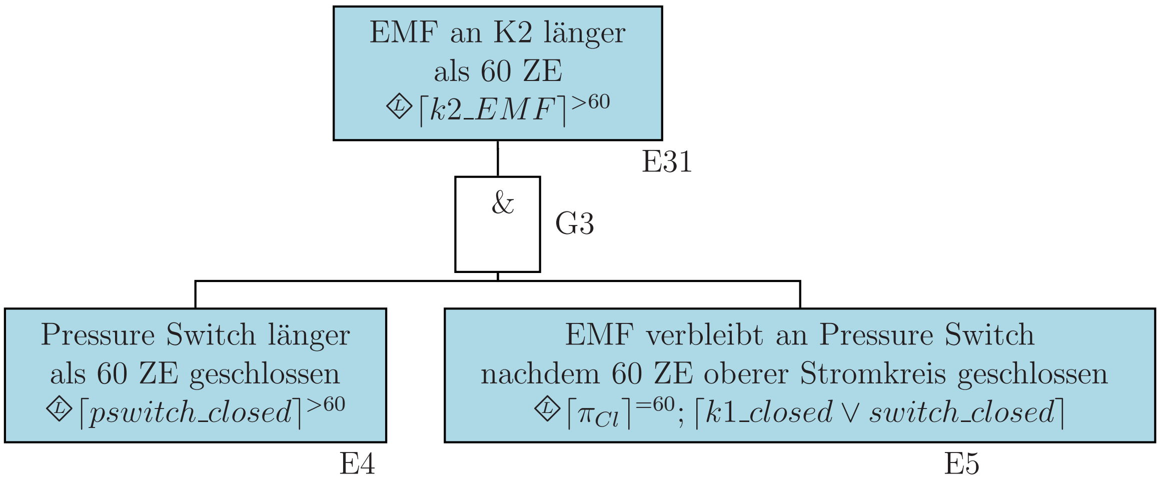


Abbildung 5.11: Phasenautomaten zu Gatter G2.5

Gatter G3



Die durch Gatter G3 dargestellte Zerlegung ist nur vollständig aber nicht korrekt, da es nicht hinreichend ist, dass der Druckschalter länger als 60 ZE geschlossen ist und länger als 60 ZE ein elektr. Feld an den Kontakten anliegt, da die Ereignisse nicht gleichzeitig eintreten müssen. Für das Model-Checking der Bedingung

$$\Diamond\lceil k2\_EMF\rceil^{>60} \Rightarrow (\Diamond\lceil pswitch\_closed\rceil^{>60} \wedge \Diamond\lceil \pi_{Cl}\rceil^{=60}; \lceil k1\_closed \vee switch\_closed\rceil)$$

wird das Netzwerk

E31||neg_E4_or_neg_E5||K2_power

aus Abbildung 5.12 aus Leerheit geprüft. Der Automat **neg_E4_or_neg_E5** ist dabei aus Automaten zu $\neg\Diamond\lceil pswitch\_closed\rceil^{>60}$ und zu $\neg\Diamond\lceil \pi_{Cl}\rceil^{=60}; \lceil k1\_closed \vee switch\_closed\rceil$ durch Oder-Konstruktion entstanden. Dabei ist die Abkürzung π_{Cl} durch $pswitch\_closed \wedge (k1\_closed \vee switch\_closed)$ ersetzt. Da der Ausdruck $\Diamond\lceil \pi_{Cl}\rceil^{=60}; \lceil k1\_closed \vee switch\_closed\rceil$ zwar dem Sequenz-Pattern entspricht, sich die beiden Phasen der Sequenz aber nicht ausschließen, kann die in Kapitel 4 entwickelte Technik zur Komplementbildung nicht direkt eingesetzt werden, sondern es muss per Hand nachgeholfen werden.

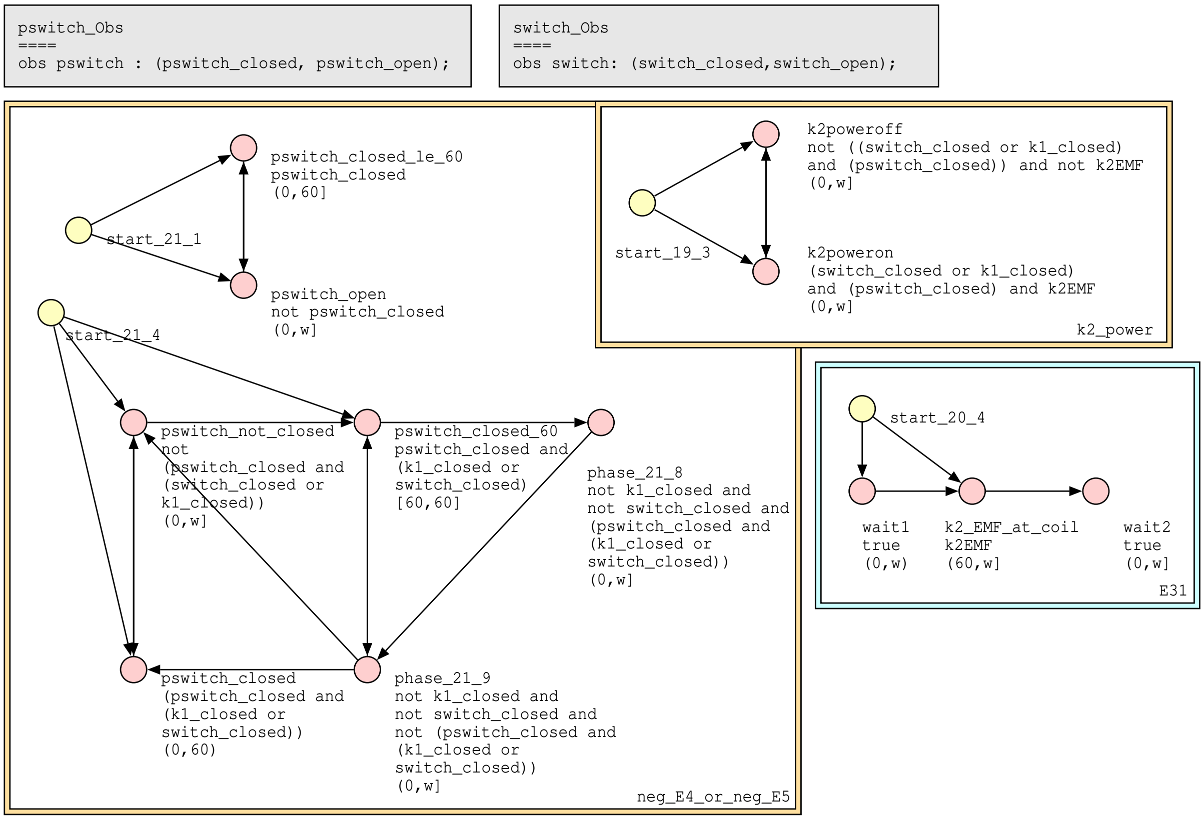
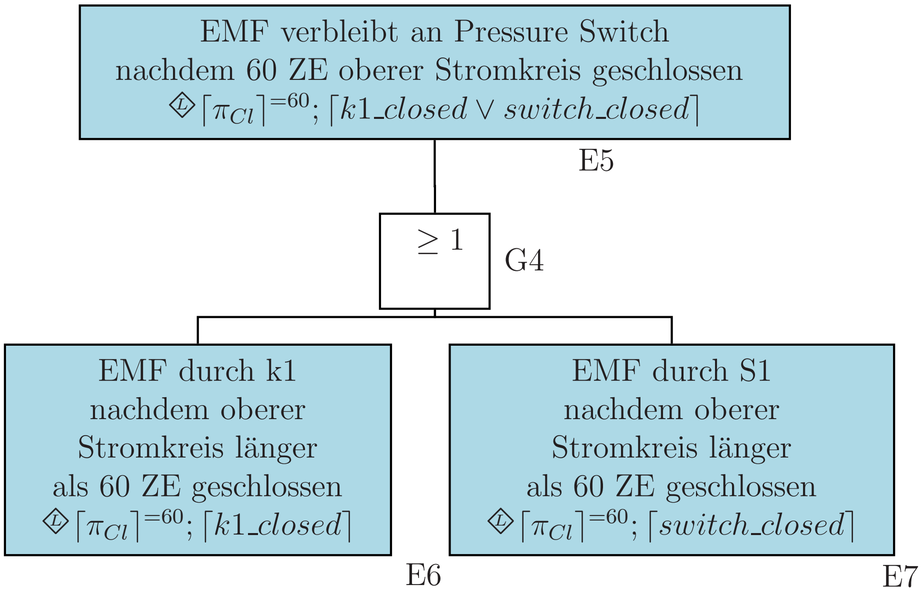


Abbildung 5.12: Phasenautomaten zu Gatter G3

Gatter G4



Die Äquivalenz der beiden Formeln

$$\Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil k1\_closed \vee switch\_closed \rceil$$

und

$$\Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil k1\_closed \rceil \vee \Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil switch\_closed \rceil$$

kann man direkt sehen und mit MOBY/DC beweisen. Für die Vollständigkeits- und Korrektheitsüberprüfungen wird die Modellierung deswegen auch nicht gebraucht. Die beiden auf Leerheit zu überprüfenden Netzwerke sind in den Abbildungen 5.13 und 5.14 dargestellt. Für die Vollständigkeit

$$\begin{aligned} &\Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil k1\_closed \vee switch\_closed \rceil \\ \Rightarrow &\Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil k1\_closed \rceil \vee \Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil switch\_closed \rceil \end{aligned}$$

wird das Netzwerk

E5||neg_E6||neg_E7

und für die Korrektheit

$$\begin{aligned} &\Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil k1\_closed \vee switch\_closed \rceil \\ \Leftarrow &\Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil k1\_closed \rceil \vee \Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil switch\_closed \rceil \end{aligned}$$

das Netzwerk

neg_E5||E6_or_E7

benutzt. Für die Automaten **E35** und **E6_or_E7** werden das Sequenz-Pattern und die Oder-Konstruktion verwendet. Die Komplemente sind analog zu dem in

neg_E4_or_neg_E5

vorkommenden Komplement von **E35** konstruiert.

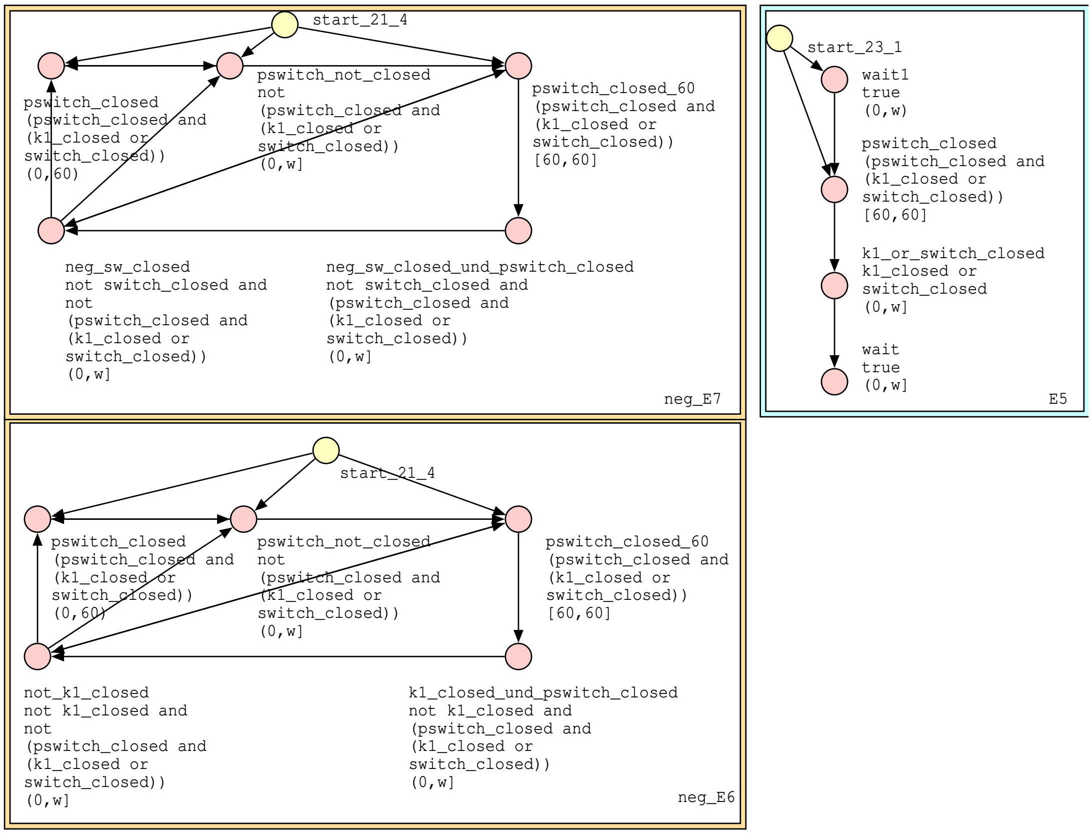


Abbildung 5.13: Phasenautomaten zu Gatter G4 (Vollständigkeit)

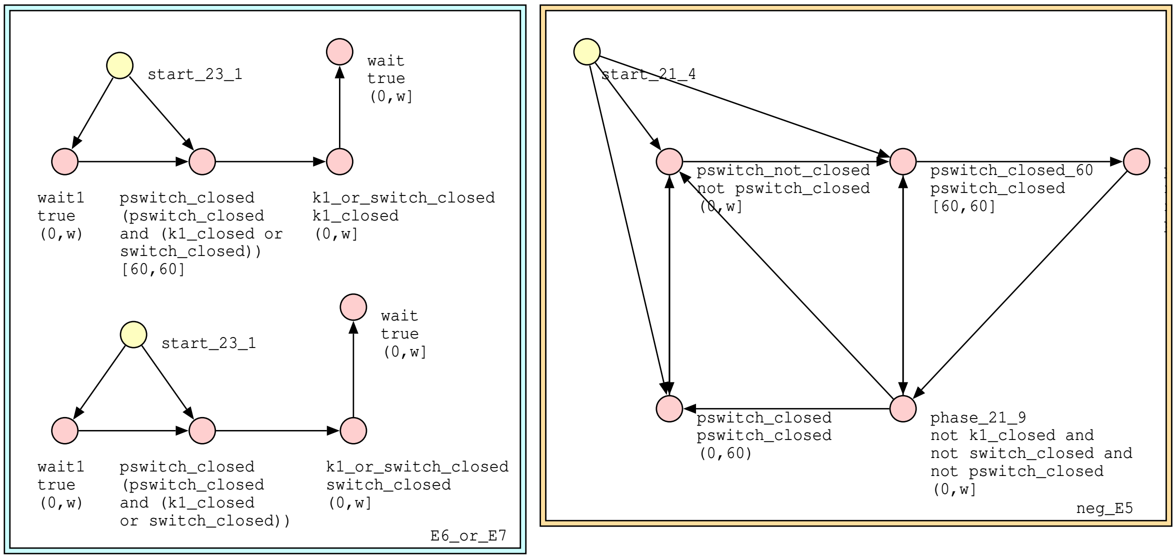
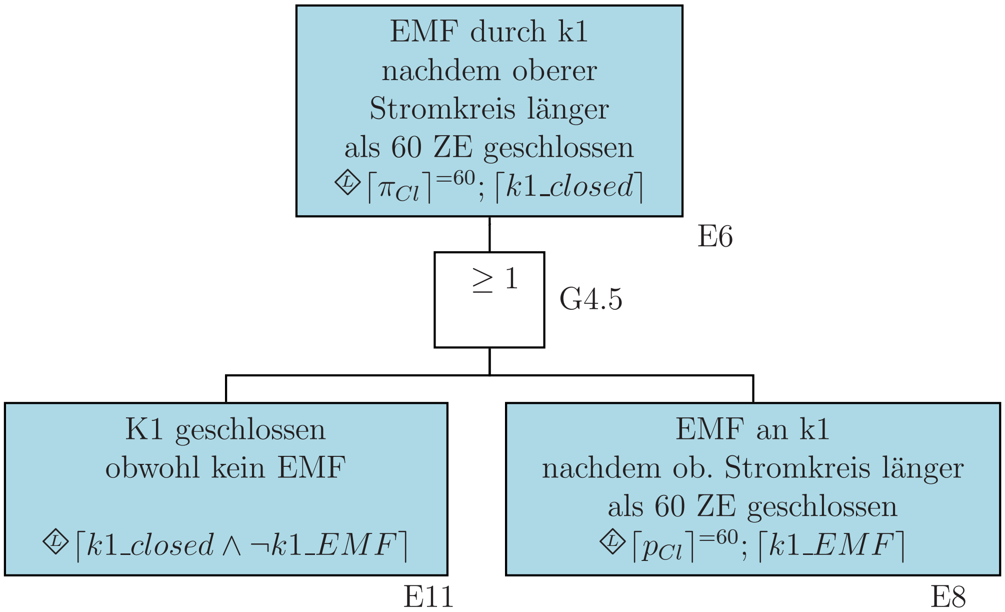


Abbildung 5.14: Phasenautomaten zu Gatter G4 (Korrektheit)

Gatter G4.5



Wenn das Relais geschlossen ist, nachdem 60 ZE lang der obere Stromkreis geschlossen war, muss entweder noch Strom durch die Spule des Relais fließen oder das Relais muss sich nicht geöffnet haben und damit ausgefallen sein. Die Umkehrung gilt nicht, da der Ausfall des Relais nicht eintreten muss, nachdem der Stromkreis 60 ZE geschlossen war, sondern auch danach oder davor eintreten kann. Die Korrektheitsbedingung kann aus diesem Grund nicht gezeigt werden. Die Vollständigkeitsbedingung

$$\begin{aligned} &\Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil k1\_closed \rceil \\ \Rightarrow &\Diamond \lceil k1\_closed \wedge \neg k1\_EMF \rceil \vee \Diamond \lceil p_{Cl} \rceil^{=60}; \lceil k1\_EMF \rceil \end{aligned}$$

kann als Leerheitstest von

$$\mathbf{E6} \| \mathbf{neg\_E11} \| \mathbf{neg\_E8} \| \mathbf{K1}$$

aus Abbildung 5.15 verifiziert werden.

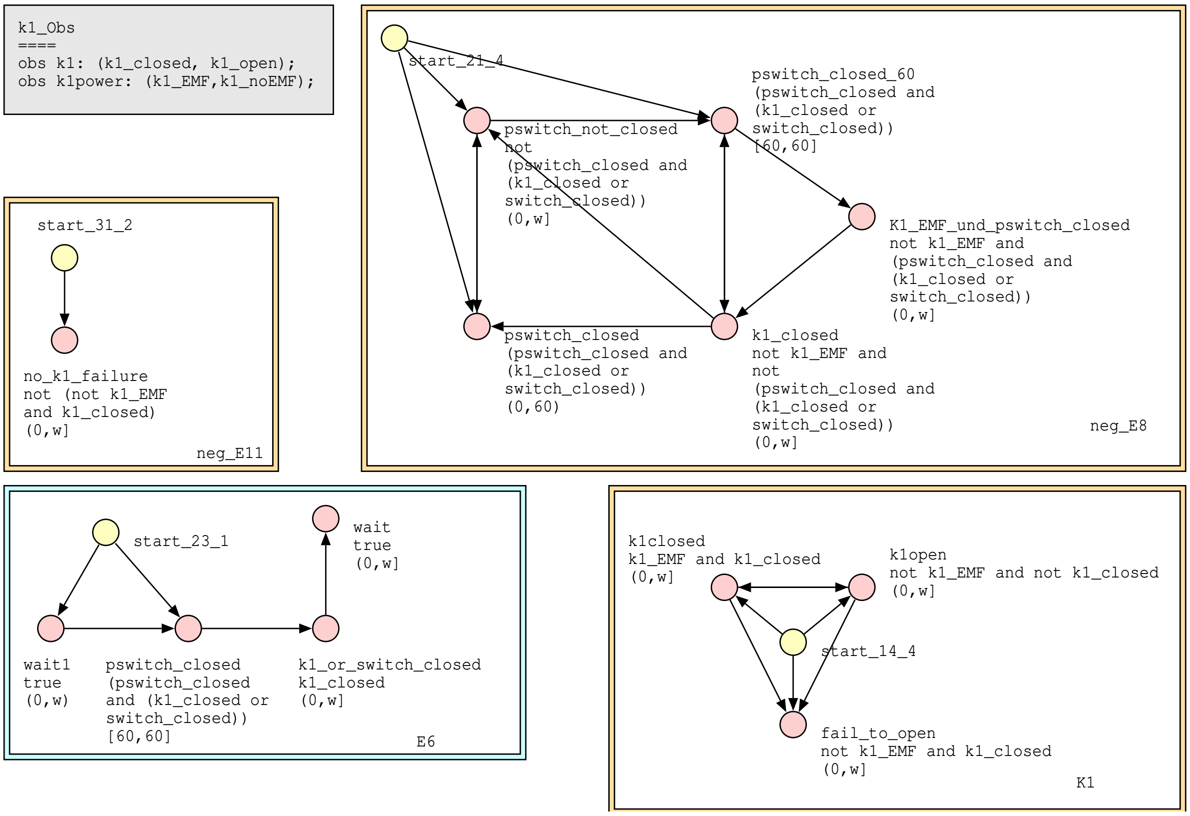
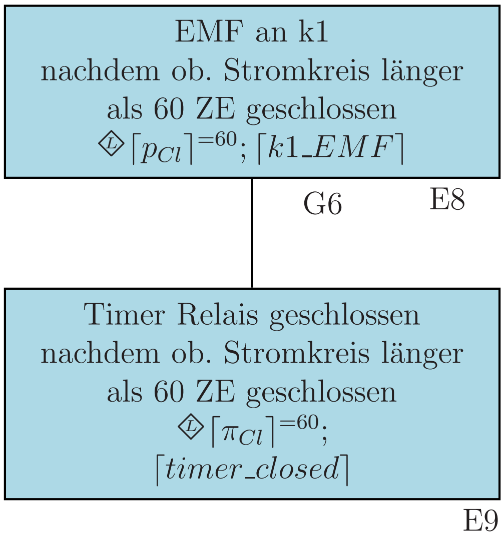


Abbildung 5.15: Phasenautomaten zu Gatter G4.5

Gatter G6



Für das Gatter kann mit Hilfe des Netzwerks von Phasenautomaten

$$\mathbf{E8 \| neg\_E9 \| k1\_power}$$

gezeigt werden, dass die Bedingung

$$\begin{gathered}\Diamond \lceil p_{Cl} \rceil^{=60}; \lceil k1\_EMF \rceil \\ \Rightarrow \Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil timer\_closed \rceil\end{gathered}$$

erfüllt ist. Die Phasenautomaten sind in Abbildung 5.16 dargestellt. In dieser Modellierung kann nicht gezeigt werden, dass auch die andere Richtung

$$\begin{gathered}\Diamond \lceil p_{Cl} \rceil^{=60}; \lceil k1\_EMF \rceil \\ \Leftarrow \Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil timer\_closed \rceil\end{gathered}$$

gilt, also die Zerlegung auch korrekt ist. Das liegt daran, dass in der Modellierung kein zeitlicher Unterschied zwischen dem Anlegen der Spannung und dem Schließen des Relais modelliert ist. Da die Korrektheit für den Fehlerbaum sowieso nicht gezeigt werden kann, wird auch die Modellierung nicht erweitert, um die Korrektheit der Zerlegung zeigen zu können.

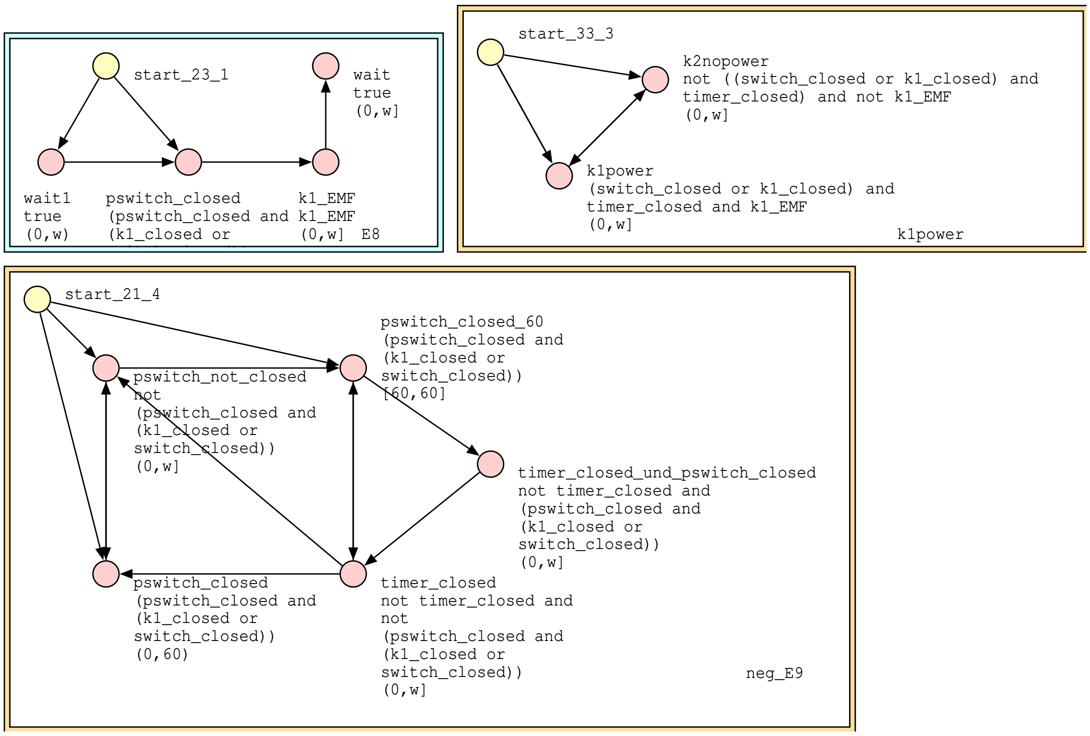
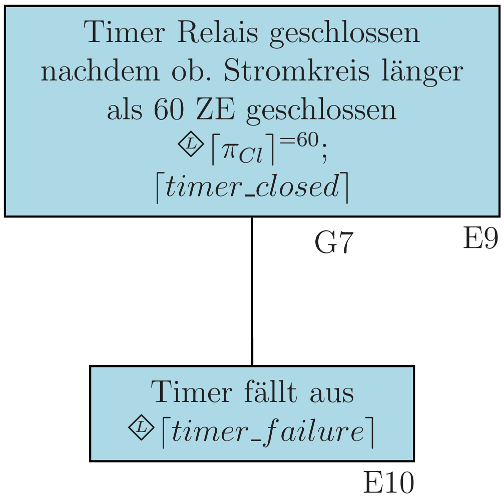


Abbildung 5.16: Phasenautomaten zu Gatter G6

Gatter G7



Da sich nach Modellierung das Timer Relais öffnen muss, wenn 60 ZE kontinuierlich Strom durch das Relais fließt, sind die beiden Ereignisse äquivalent. Durch die in Abbildungen 5.17 und 5.18 dargestellten Automaten lassen sich die Eigenschaften verifizieren. Mit dem Netz

$$\mathbf{E9 \| neg\_E10 \| timer\_relais}$$

wird die Vollständigkeitsbedingung

$$\begin{aligned} &\Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil timer\_closed \rceil \\ \Rightarrow &\Diamond \lceil timer\_failure \rceil \end{aligned}$$

gezeigt. Der Automat **E9** zu $\Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil timer\_closed \rceil$ wird nach dem Sequenz-Pattern, der Automat **neg_E10** nach dem Zustands-Pattern (Komplement) konstruiert. Für die Korrektheitsbedingung

$$\begin{aligned} &\Diamond \lceil \pi_{Cl} \rceil^{=60}; \lceil timer\_closed \rceil \\ \Leftarrow &\Diamond \lceil timer\_failure \rceil \end{aligned}$$

wird das Netz

$$\mathbf{neg\_E9 \| E10 \| timer\_relais}$$

benutzt. Der Automat **neg_E9** wurde schon bei der Verifikation der Zerlegung bei Gatter G8 benutzt, der Automat **E10** wird nach der Konstruktion zum Zustands-Pattern bestimmt.

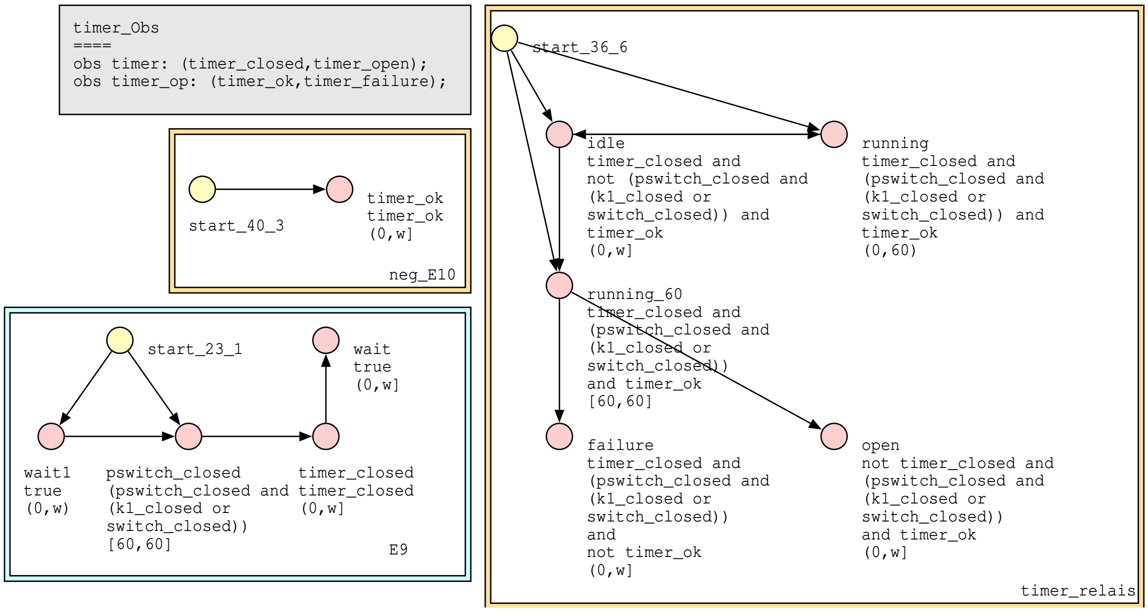


Abbildung 5.17: Phasenautomaten zu Gatter G7 (Vollständigkeit)

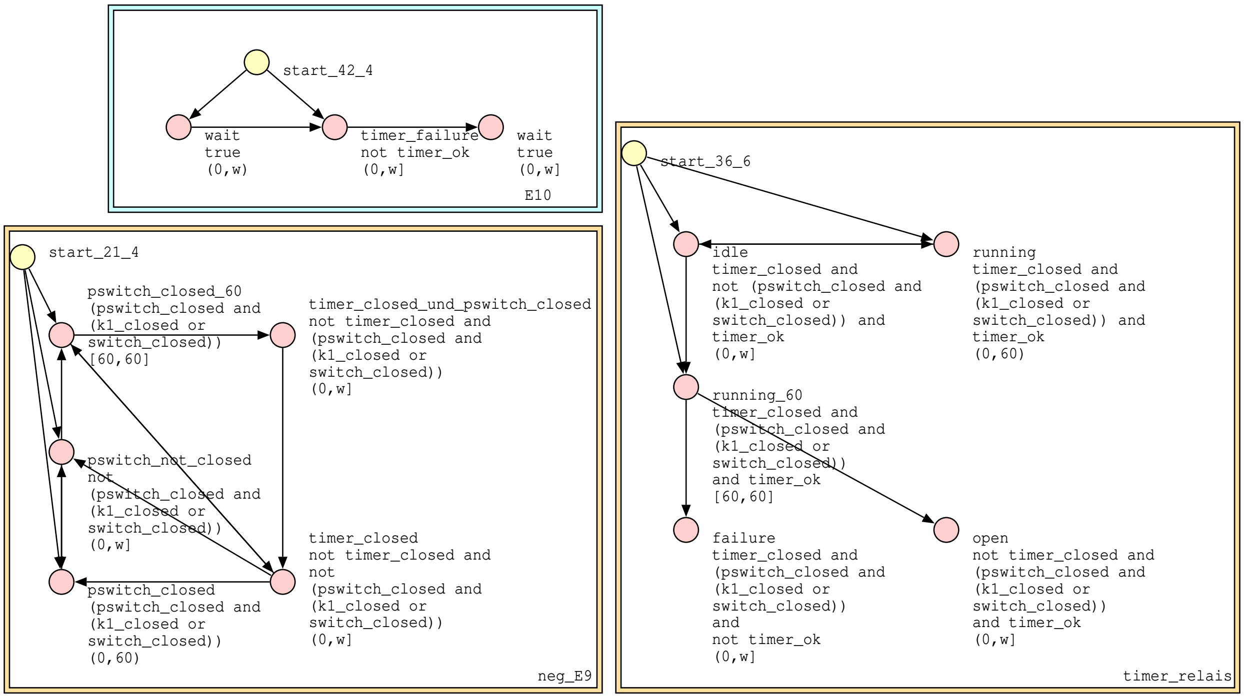


Abbildung 5.18: Phasenautomaten zu Gatter G7 (Korrektheit)

5.1.5 Zusammenfassung

Für das Fallbeispiel Pressure-Tank kann die Vollständigkeit des Fehlerbaums für eine bestimmte Modellierung verifiziert werden. Es genügt also die Blätter des Fehlerbaums auszuschließen, um sicher zu sein, dass der Tank intakt bleibt. Hingegen kann die Korrektheit vieler Zerlegungen nicht gezeigt werden. Somit führt auch das Eintreten aller Ereignisse in den Blättern nicht in jedem Fall zum Bruch des Tanks. Es zeigt sich aber auch, dass die Modellierung des Systems entscheidend ist. Eine andere Modellierung, etwa eine, die keine Ausfälle der Pumpe erlaubt, führt auch zu anderen Ergebnissen. Die verwendete formale Modellierung muss also das tatsächliche System gut beschreiben, damit die formale Fehlerbaumanalyse Rückschlüsse auf das System erlaubt.

5.2 Verteilte Weichen- und Signal-Steuerung

An dem schon in der Einleitung vorgestellten Fallbeispiel einer verteilten Weichen- und Signalsteuerung soll gezeigt werden, wie die formale Fehlerbaumanalyse mit der formalen Verifikation von Teilkomponenten verbunden werden kann. Für die Teilkomponenten konnten Aussagen mit Hilfe von PLC-Automaten und Moby/plc verifiziert werden. Durch eine Fehlerbaumanalyse soll gezeigt werden, dass aus diesen Aussagen und einigen nur informell beschriebenen Bedingungen, die Sicherheit des Systems folgt. Es genügt deshalb, nur die Vollständigkeit des Fehlerbaums zu überprüfen. Dazu wird von der Gefährdung "Zwei Züge stoßen zusammen" ausgegangen und eine Fehlerbaumanalyse durchgeführt bis die Ereignisse direkt oder durch Verifikation der Steuerungssoftware mit Moby/plc ausgeschlossen werden können.

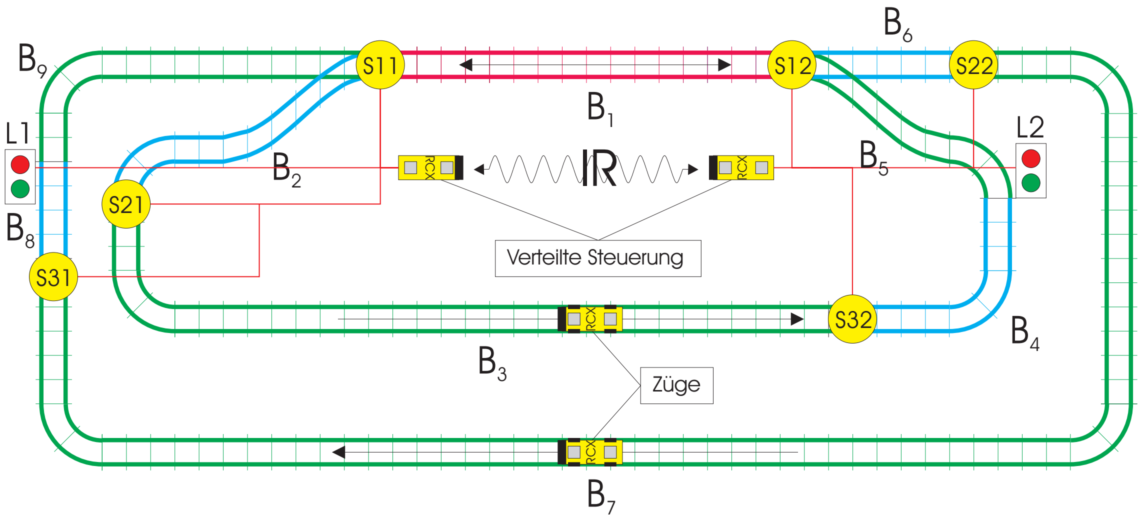


Abbildung 5.19: Gleissystem zum Fallbeispiel Zugsteuerung

5.2.1 Beschreibung des Systems

Die Anforderungen an die verteilte Steuerungen sind bereits in der Einleitung beschrieben. In der Realisierung der Steuerung wird alle 1200 ms ein Token zwischen den beiden Steuerungs-Rechnern (Lego Mindstorm, RCX) ausgetauscht, solange kein Zug an einem Signal wartet. Wenn ein RCX das Token besitzt und an dem ihm zugeordneten Signal ein Zug auf Einfahrt in den kritischen Bereich wartet, setzt das Signal auf grün und erlaubt so die Einfahrt. Sobald der Zug in den kritischen Bereich einfährt wird das Signal wieder auf rot gesetzt. In dieser Zeit kann ein zweiter Zug über das gleiche grüne Signal gefahren sein ("Folgefahrt"). Sobald alle Züge, die in den kritischen Bereich eingefahren waren, diesen wieder verlassen haben wird das Token an den anderen Rechner abgegeben und das Token wird wieder alle 1200 ms ausgetauscht.

5.2.2 Modellierung des Systems

Das System wird wieder durch eine Reihe von Phasenautomaten modelliert, die in Abbildung 5.20 dargestellt sind. Die Position des 1. Zuges auf dem Gleissystem wird durch die Observable

$$\mathsf{Zug1Pos} : (Z1B1, Z2B2, \ldots, Z1B9)$$

angezeigt. Die Observable

$$\mathsf{Zug1Richtung} : (Z1V, Z2R)$$

enthält die momentane Fahrtrichtung des Zuges. Der Wert $Z1V$ beschreibt, dass der Zug **v**orwärts, d.h. im Uhrzeigersinn und der Wert $Z1R$, dass der Zug **r**ückwärts, d.h. gegen den Uhrzeigersinn fährt. Die in Abbildung 5.20 dargestellten Automaten **Zug1** und **Zug1_timer** bestimmen,

1. welche Streckenabschnitte aufeinander folgen,
2. dass in den kritische Bereich nur bei grünen Signalen eingefahren werden darf,
3. dass nur in der zulässigen Fahrtrichtung gefahren werden darf,
4. dass im kritischen Bereich die Fahrtrichtung einmal gewechselt werden darf und
5. dass nur in dem Bereich vor den Signalen beliebig lange gewartet werden darf und sonst die Abschnitt in vorgegebener Zeit durchfahren werden. Der Zug darf also nur vor Signalen stehenbleiben.

Analog beschreiben die Observablen Zug2Pos und Zug2Richtung und die Phasenautomaten **Zug2** und **Zug2_timer** das Verhalten des 2. Zuges. Die Phasenautomaten **physik1** und **physik2** verbieten, dass zwei Züge von der selben Seite gleichzeitig einfahren. Die Observablen SignalL1 und SignalL2 geben an, ob das Signal rot oder grün ist. Entsprechend geben die Observablen Contr1Token und Contr2Token an, ob ein Steuerungsrechner ein Token besitzt. Dabei wird absichtlich noch *nicht* davon ausgegangen, dass immer genau ein Token existiert. Mit Moby/PLC ist leicht zu verifizieren, dass ein Signal nur dann grün sein kann, wenn der zugehörige Steuerungsrechner das Token besitzt. Diese Eigenschaft wird durch die Phasenautomaten **C1Token** und **C2Token** modelliert.

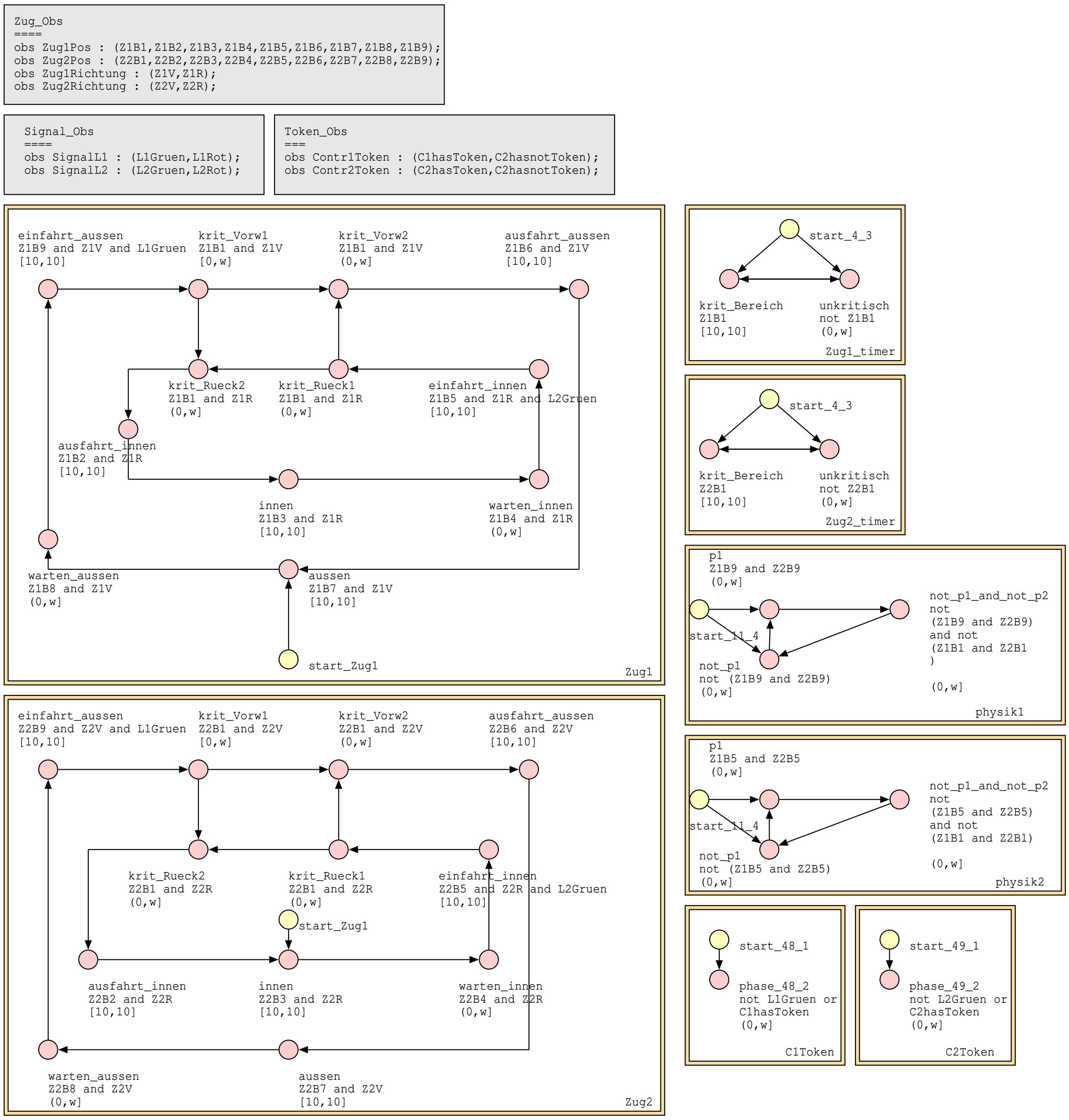


Abbildung 5.20: Modellierung durch Phasenautomaten (Fahrstrecke Zugbeispiel)

5.2.3 Fehlerbaumanalyse

Auswahl des Top-Ereignisses

Das Top-Ereignis der Fehlerbaumanalyse soll „Zwei Züge fahren im kritischen Bereich aufeinander zu“ lauten. Obwohl das Ereignis umgangssprachlich einfach zu beschreiben ist, stellt die Beschreibung durch eine DCL-Formeln ein Problem dar. Wenn sich beide Züge im kritischen Bereich befinden, muss es nicht zu einem Zusammenstoß kommen. Dieser Fall muss als “Folgefahrt” sogar ausdrücklich vom System zugelassen werden. Also ist eine Beschreibung durch die Formel $Z1B1 \wedge Z2B1$ zu scharf. Selbst wenn zusätzlich angenommen wird, dass beide Züge in entgegengesetzter Richtung fahren, können sie voneinander weg, statt aufeinander zu fahren. Für die Formel zum Top-Ereignis werden also alle Möglichkeiten aufgelistet.

$$A_1 \stackrel{df}{=} A_2 \vee A_3 \vee A_4 \vee A_5 \vee A_6 \vee A_7 \vee A_8 \vee A_9 \vee A_{10}$$

Zuerst werden die Fälle betrachtet, in denen sich der 1. Zug im kritischen Abschnitt befindet, als der 2. Zug einfährt.

Formel A_2 Die Formel

$$\begin{aligned} A_2 \stackrel{df}{=} \Diamond_{L}(&Z1B1 \wedge (Z2B9 \vee Z2B8 \vee Z2B7));\\ &(Z1B1 \wedge Z2B1 \wedge Z1V \wedge Z2V);\\ &(Z1B1 \wedge Z2B1 \wedge Z1R \wedge Z2V) \end{aligned}$$

beschreibt den Fall, dass der 1. Zug im kritischen Abschnitt ist, dann der 2. Zug von links einfährt und nach der Einfahrt der 1. Zug seine Fahrtrichtung ändert.

Formel A_3 Die Formel

$$\begin{aligned} A_3 \stackrel{df}{=} \Diamond_{L}(&Z1B1 \wedge (Z2B5 \vee Z2B4 \vee Z2B3));\\ &(Z1B1 \wedge Z2B1 \wedge Z1R \wedge Z2R);\\ &(Z1B1 \wedge Z2B1 \wedge Z1V \wedge Z2R) \end{aligned}$$

beschreibt den Fall, dass der 1. Zug im kritischen Abschnitt ist, dann der 2. Zug von rechts einfährt und nach der Einfahrt der 1. Zug seine Fahrtrichtung ändert.

Formel A_4 Die Formel

$$\begin{aligned} A_4 \stackrel{df}{=} \Diamond_{L}(&Z1B1 \wedge (Z2B9 \vee Z2B8 \vee Z2B7));\\ &(Z1B1 \wedge Z2B1 \wedge Z1R \wedge Z2V) \end{aligned}$$

beschreibt den Fall, dass der 1. Zug im kritischen Abschnitt ist, dann der 2. Zug von links einfährt und die Fahrtrichtung nicht geändert wird. Entsprechende Fälle gibt es auch, wenn der 2. Zug von rechts in den kritischen Bereich einfährt.

Formel A_5 Die Formel

$$\begin{aligned} A_5 \stackrel{df}{=} \Diamond_{L}&(Z1B1 \wedge (Z2B5 \vee Z2B4 \vee Z2B3));\\ &(Z1B1 \wedge Z2B1 \wedge Z1R \wedge Z2V) \end{aligned}$$

beschreibt den Fall, dass der 1. Zug im kritischen Abschnitt ist, dann der 2. Zug von rechts einfährt und die Fahrtrichtung nicht geändert wird.

Formel A_6 Eine andere Möglichkeit ist, dass beide Züge gleichzeigt in den kritischen Abschnitt einfahren.

$$\begin{aligned} A_6 \stackrel{df}{=} \Diamond_{L}&(\neg Z1B1 \wedge \neg Z2B1);\\ &(Z1B1 \wedge Z2B1 \wedge (Z1V \wedge Z2R \vee Z1R \wedge Z2V)) \end{aligned}$$

Die restlichen Formeln beschreiben den Fall, dass der 2. Zug zuerst im kritischen Abschnitt ist und sind ganz analog zu $A_1, \dots, A_4$ aufgebaut.

$$\begin{aligned} A_7 \stackrel{df}{=} \Diamond_{L}&(Z2B1 \wedge (Z1B9 \vee Z1B8 \vee Z1B7));\\ &(Z2B1 \wedge Z1B1 \wedge Z2V \wedge Z1V);\\ &(Z2B1 \wedge Z1B1 \wedge Z2R \wedge Z1V)\\ A_8 \stackrel{df}{=} \Diamond_{L}&(Z2B1 \wedge (Z1B5 \vee Z1B4 \vee Z1B3));\\ &(Z2B1 \wedge Z1B1 \wedge Z2R \wedge Z1R);\\ &(Z2B1 \wedge Z1B1 \wedge Z2V \wedge Z1R)\\ A_9 \stackrel{df}{=} \Diamond_{L}&(Z2B1 \wedge (Z1B9 \vee Z1B8 \vee Z1B7));\\ &(Z2B1 \wedge Z1B1 \wedge Z2R \wedge Z1V)\\ A_{10} \stackrel{df}{=} \Diamond_{L}&(Z2B1 \wedge (Z1B5 \vee Z1B4 \vee Z1B3));\\ &(Z2B1 \wedge Z1B1 \wedge Z2R \wedge Z1V) \end{aligned}$$

Es kann natürlich nicht gezeigt werden, dass durch diese Formeln auch alle Fälle abgedeckt werden. Dafür erhält man aus der Definition des Top-Ereignisses auch die erste Zerlegung. Der gesamte Fehlerbaum ist in Abbildung 5.21 dargestellt.

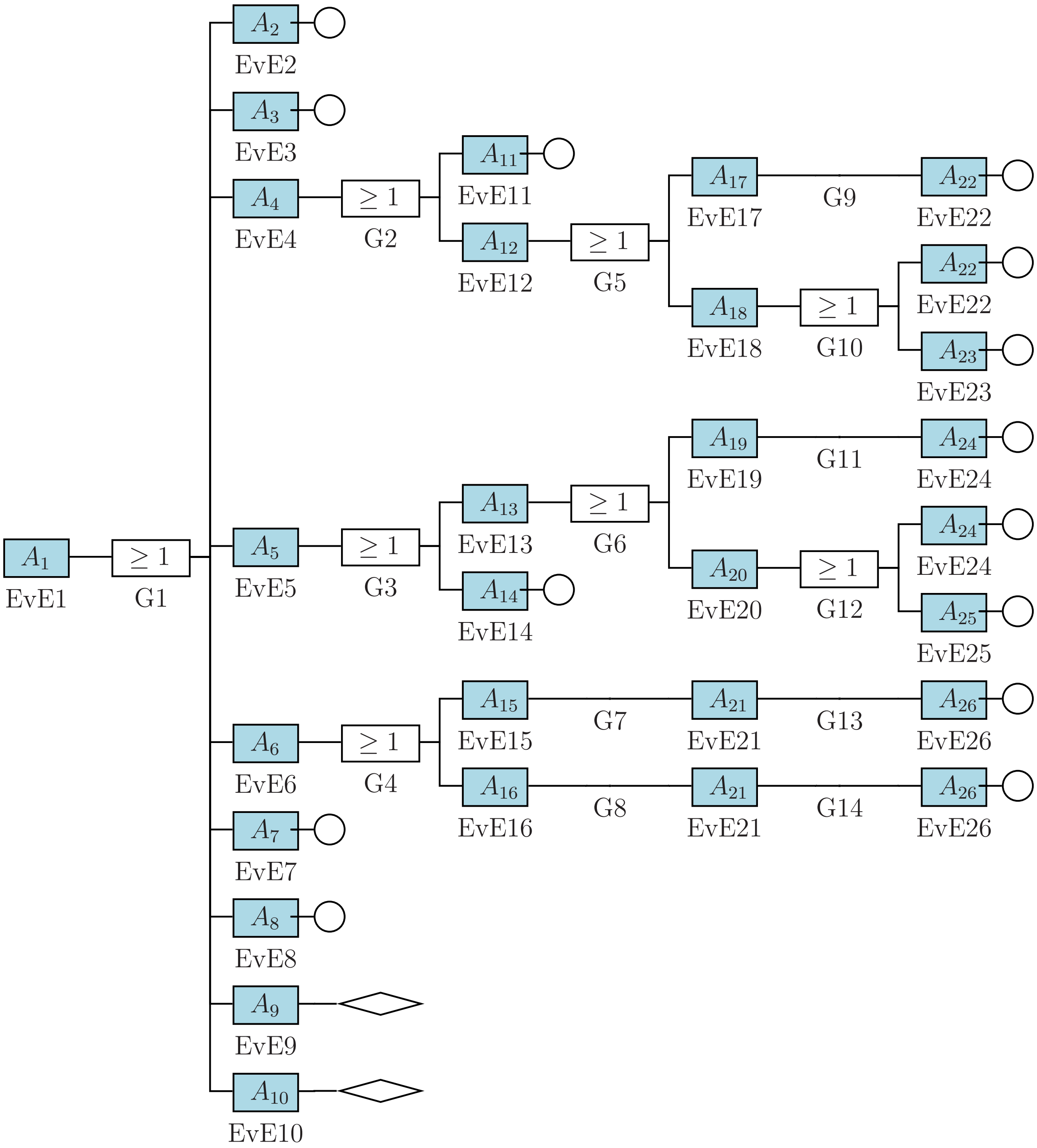


Abbildung 5.21: Fehlerbaum zum Beispiel Weichen- und Signalsteuerung

Ereignis	**Beschreibung**
EvE1	Zwei Züge fahren in kritischem Bereich aufeinander zu
EvE2	Zug1 ist in krit. Bereich. Einfahrt von Zug2 in krit. Bereich von links. Zug1 ändert Fahrtrichtung. Züge fahren aufeinander zu.
EvE3	Zug1 ist in krit. Bereich. Einfahrt von Zug2 in krit. Bereich von rechts. Zug1 ändert Fahrtrichtung. Züge fahren aufeinander zu.
EvE4	Zug1 ist in krit. Bereich. Einfahrt von Zug2 in krit. Bereich von links. Züge fahren aufeinander zu.
EvE5	Zug1 ist in krit. Bereich. Einfahrt von Zug2 in krit. Bereich von rechts. Züge fahren aufeinander zu.
EvE6	Beide Züge fahren gleichzeitig in krit. Bereich. Züge fahren aufeinander zu.
EvE7	Zug2 ist in krit. Bereich. Einfahrt von Zug1 in krit. Bereich von links. Zug2 ändert Fahrtrichtung. Züge fahren aufeinander zu.
EvE8	Zug2 ist in krit. Bereich. Einfahrt von Zug1 in krit. Bereich von rechts. Zug2 ändert Fahrtrichtung. Züge fahren aufeinander zu.
EvE9	Zug2 ist in krit. Bereich. Einfahrt von Zug1 in krit. Bereich von links. Züge fahren aufeinander zu.
EvE10	Zug2 ist in krit. Bereich. Einfahrt von Zug1 in krit. Bereich von rechts. Züge fahren aufeinander zu.
EvE11	Einfahrt Zug1 in krit. Bereich von links. Einfahrt von Zug2 in krit. Bereich von links. Zug1 ändert Fahrtrichtung. Züge fahren aufeinander zu.
EvE12	Einfahrt Zug1 in krit. Bereich von rechts. Einfahrt von Zug2 in krit. Bereich von links. Zug1 ändert Fahrtrichtung. Züge fahren aufeinander zu.
EvE13	Einfahrt Zug1 in krit. Bereich von links. Einfahrt von Zug2 in krit. Bereich von rechts. Züge fahren aufeinander zu.
EvE14	Einfahrt Zug1 in krit. Bereich von rechts. Einfahrt von Zug2 in krit. Bereich von rechts. Züge fahren aufeinander zu.
EvE15	Gleichzeitige Einfahrt Zug1 von links und Zug2 von rechts.
EvE16	Gleichzeitige Einfahrt Zug1 von rechts und Zug2 von links.
EvE17	L2 ist grün. Als Zug1 in krit. Bereich von rechts einfährt, wird L1 grün.
EvE18	L2 ist grün. Nachdem Zug1 in krit. Bereich von rechts eingefahren ist und bevor Zug1 wieder herausfährt, wird L1 grün.

Tabelle 5.1: Beschreibung der Ereignisse in Abbildung 5.21 (1. Teil)

Ereignis	Beschreibung
EvE19	L1 ist grün. Als Zug1 in krit. Bereich von links einfährt, wird L1 grün.
EvE20	L1 ist grün. Nachdem Zug1 in krit. Bereich von links eingefahren ist und bevor Zug1 wieder herausgefahren ist, wird L1 grün.
EvE21	L1 und L2 sind gleichzeitig grün.
EvE22	Controller2 hat Token. Als Zug1 in krit. Bereich von rechts einfährt, erhält Controller1 das Token.
EvE23	Controller2 hat Token. Nachdem Zug1 in krit. Bereich von rechts eingefahren ist und bevor Zug1 wieder herausgefahren ist, bekommt Controller1 das Token.
EvE24	Controller1 hat Token. Als Zug1 in krit. Bereich von links einfährt, erhält Controller2 das Token.
EvE25	Controller1 hat Token. Nachdem Zug1 in krit. Bereich von links eingefahren ist und bevor Zug1 wieder herausgefahren ist, bekommt Controller2 das Token.
EvE26	Controller1 und Controller2 haben gleichzeitig das Token.

Tabelle 5.2: Beschreibung der Ereignisse in Abbildung 5.21 (2. Teil)

Ereignis	**Formel**
EvE1	$A_1 \overset{df}{=} A_2 \vee \ldots \vee A_1 0$
EvE2	$A_2 \overset{df}{=} \Diamond_L (Z1B1 \wedge (Z2B9 \vee Z2B8 \vee Z2B7));$ $(Z1B1 \wedge Z2B1 \wedge Z1V \wedge Z2V);$ $(Z1B1 \wedge Z2B1 \wedge Z1R \wedge Z2V)$
EvE3	$A_3 \overset{df}{=} \Diamond_L (Z1B1 \wedge (Z2B5 \vee Z2B4 \vee Z2B3));$ $(Z1B1 \wedge Z2B1 \wedge Z1R \wedge Z2R);$ $(Z1B1 \wedge Z2B1 \wedge Z1V \wedge Z2R)$
EvE4	$A_4 \overset{df}{=} \Diamond_L (Z1B1 \wedge (Z2B9 \vee Z2B8 \vee Z2B7));$ $(Z1B1 \wedge Z2B1 \wedge Z1R \wedge Z2V)$
EvE5	$A_5 \overset{df}{=} \Diamond_L (Z1B1 \wedge (Z2B5 \vee Z2B4 \vee Z2B3));$ $(Z1B1 \wedge Z2B1 \wedge Z1R \wedge Z2V)$
EvE6	$A_6 \overset{df}{=} \Diamond_L (\neg Z1B1 \wedge \neg Z2B1);$ $(Z1B1 \wedge Z2B1 \wedge (Z1V \wedge Z2R \vee Z1R \wedge Z2V))$
EvE7	$A_7 \overset{df}{=} \Diamond_L (Z2B1 \wedge (Z1B9 \vee Z1B8 \vee Z1B7));$ $(Z2B1 \wedge Z1B1 \wedge Z2V \wedge Z1V);$ $(Z2B1 \wedge Z1B1 \wedge Z2R \wedge Z1V)$
EvE8	$A_8 \overset{df}{=} \Diamond_L (Z2B1 \wedge (Z1B5 \vee Z1B4 \vee Z1B3));$ $(Z2B1 \wedge Z1B1 \wedge Z2R \wedge Z1R);$ $(Z2B1 \wedge Z1B1 \wedge Z2V \wedge Z1R)$

Tabelle 5.3: Formeln zu den Ereignissen in Abbildung 5.21 (1. Teil)

Ereignis	Formel
EvE9	$A_9 \stackrel{df}{=} \Diamond_{L}(Z2B1 \wedge (Z1B9 \vee Z1B8 \vee Z1B7));$ $(Z2B1 \wedge Z1B1 \wedge Z2R \wedge Z1V)$
EvE10	$A_{10} \stackrel{df}{=} \Diamond_{L}(Z2B1 \wedge (Z1B5 \vee Z1B4 \vee Z1B3));$ $(Z2B1 \wedge Z1B1 \wedge Z2R \wedge Z1V)$
EvE11	$A_{11} \stackrel{df}{=} \Diamond_{L}(Z1B9);$ $(Z1B1 \wedge (Z2B9 \vee Z2B8 \vee Z2B7));$ $(Z1B1 \wedge Z1R \wedge Z2B1 \wedge Z2V)$
EvE12	$A_{12} \stackrel{df}{=} \Diamond_{L}(Z1B5);$ $(Z1B1 \wedge (Z2B9 \vee Z2B8 \vee Z2B7));$ $(Z1B1 \wedge Z1R \wedge Z2B1 \wedge Z2V)$
EvE13	$A_{13} \stackrel{df}{=} \Diamond_{L}(Z1B9);$ $(Z1B1 \wedge (Z2B5 \vee Z2B4 \vee Z2B3));$ $(Z1B1 \wedge Z1V \wedge Z2B1 \wedge Z2R)$
EvE14	$A_{14} \stackrel{df}{=} \Diamond_{L}(Z1B5);$ $(Z1B1 \wedge (Z2B5 \vee Z2B4 \vee Z2B3));$ $(Z1B1 \wedge Z1V \wedge Z2B1 \wedge Z2R)$
EvE15	$A_{15} \stackrel{df}{=} \Diamond_{L}(Z1B9 \wedge Z2B5);$ $(Z1B1 \wedge Z2B1 \wedge ((Z1V \wedge Z2R) \vee (Z1R \wedge Z2V))$
EvE16	$A_{16} \stackrel{df}{=} \Diamond_{L}(Z1B5 \wedge Z2B9);$ $(Z1B1 \wedge Z2B1 \wedge ((Z1V \wedge Z2R) \vee (Z1R \wedge Z2V))$

Tabelle 5.4: Formeln zu den Ereignissen in Abbildung 5.21 (2. Teil)

Ereignis	**Formel**
EvE17	$A_{17} \overset{df}{=} \Diamond_{L}(Z1B5 \wedge L2Gruen);$ $(Z1B1 \wedge L1Gruen)$
EvE18	$A_{18} \overset{df}{=} \Diamond_{L}(Z1B5 \wedge L2Gruen);$ $(Z1B1 \wedge \neg L1Gruen);$ $(Z1B1 \wedge L1Gruen)$
EvE19	$A_{19} \overset{df}{=} \Diamond_{L}(Z1B9 \wedge L1Gruen);$ $(Z1B1 \wedge L2Gruen)$
EvE20	$A_{20} \overset{df}{=} \Diamond_{L}(Z1B9 \wedge L1Gruen);$ $(Z1B1 \wedge \neg L2Gruen);$ $(Z1B1 \wedge L2Gruen)$
EvE21	$A_{21} \overset{df}{=} \Diamond_{L}(L1Gruen \wedge L2Gruen)$
EvE22	$A_{22} \overset{df}{=} \Diamond_{L}(Z1B5 \wedge C2hasToken);$ $(Z1B1 \wedge C1hasToken)$
EvE23	$A_{23} \overset{df}{=} \Diamond_{L}(Z1B5 \wedge C2hasToken);$ $(Z1B1 \wedge \neg C1hasToken);$ $(Z1B1 \wedge C1hasToken)$
EvE24	$A_{24} \overset{df}{=} \Diamond_{L}(Z1B9 \wedge C1hasToken);$ $(Z1B1 \wedge C2hasToken)$

Tabelle 5.5: Formeln zu den Ereignissen in Abbildung 5.21 (3. Teil)

Ereignis	**Formel**
EvE25	$A_{25} \overset{df}{=} \Diamond_L(Z1B9 \wedge C1hasToken);$ $(Z1B1 \wedge \neg C2hasToken);$ $(Z1B1 \wedge C2hasToken)$
EvE26	$A_{26} \overset{df}{=} \Diamond_L(C1hasToken \wedge C2hasToken)$

Tabelle 5.6: Formeln zu den Ereignissen in Abbildung 5.21 (4. Teil)

5.2.4 Model-Checking mit Moby/DC

Für den dargestellten Fehlerbaum soll gezeigt werden, dass keine Ereignisse vergessen worden sind, also der Baum vollständig ist. Dazu muss für jedes Gatter gezeigt werden, dass die Untereeignisse notwendig für das Ereignis ist. In dem Fehlerbaum treten nur OR-Gatter auf. Für jedes Gatter G an dem ein Ereignis A in die Untereeignisse $A^{(1)}, \dots, A^{(k)}$ zerlegt wird, muss die Formel

$$\boxed{L}_0 A_{\mathcal{M}} \Rightarrow (\boxed{L}_0 A \Rightarrow (\boxed{L}_0 A^{(1)} \vee \ldots \vee \boxed{L}_0 A^{(k)})). \qquad (5.2.5.1)$$

verifiziert werden. Dazu muss ein Automat $\mathcal{A}$ zu A und für jede Formel $A^{(i)}$ $1 \leq i \leq k$ ein Automat zu $\neg \boxed{L}_0 A^{(i)}$ konstruiert werden. Da alle Formeln dem 3. Pattern (Sequenz) entsprechen, können die dort angegebenen Konstruktionen benutzt werden. Das Netz aus Modellierung und den konstruierten Automaten wird auf Leerheit getestet, um die Implikation in 5.2.5.1 zu beweisen. Dabei werden meistens nicht alle Automaten, die zur Modellierung gehören, benötigt. In Tabelle 5.7 ist für jedes Gatter angegeben, mit welchen Automaten die Vollständigkeit bewiesen wurde. Nur für die erste Zerlegung wird die Verifikation ausführlich beschrieben.

Gatter	Für das Model-Checking benutztes Nopa	Abbildungen der Automaten
G1	**EvE1 ‖ neg_EvE2 ‖ neg_EvE3 ‖ neg_EvE4 ‖ neg_EvE5 ‖ neg_EvE6 ‖ neg_EvE7 ‖ neg_EvE8 ‖ neg_EvE9 ‖ neg_EvE10**	5.24, 5.25 und 5.26
G2	**EvE4 ‖ neg_EvE11 ‖ neg_EvE12 ‖ Zug1 ‖ Zug2 ‖ Zug1_timer ‖ Zug2_timer**	5.27 und 5.28
G3	**EvE5 ‖ neg_EvE13 ‖ neg_EvE14 ‖ Zug1 ‖ Zug2 ‖ Zug1_timer ‖ Zug2_timer**	5.29 und 5.30
G4	**EvE6 ‖ neg_EvE15 ‖ neg_EvE16 ‖ Zug1 ‖ Zug2 ‖ Zug1_timer ‖ Zug2_timer ‖ physik1 ‖ physik2**	5.31 und 5.32
G5	**EvE12 ‖ neg_EvE17 ‖ neg_EvE18 ‖ Zug1 ‖ Zug2**	5.33
G6	**EvE13 ‖ neg_EvE19 ‖ neg_EvE20 ‖ Zug1 ‖ Zug2**	5.34
G7	**EvE15 ‖ neg_EvE21 ‖ Zug1 ‖ Zug2**	5.35
G8	**EvE16 ‖ neg_EvE21 ‖ Zug1 ‖ Zug2**	5.35
G9	**EvE17 ‖ neg_EvE22 ‖ C1Token ‖ C2Token**	5.36
G10	**EvE17 ‖ neg_EvE22 ‖ neg_EvE23 ‖ C1Token ‖ C2Token**	5.36
G11	**EvE19 ‖ neg_EvE24 ‖ C1Token ‖ C2Token**	5.37
G12	**EvE20 ‖ neg_EvE24 ‖ neg_EvE25 ‖ C1Token ‖ C2Token**	5.37
G13/G14	**EvE21 ‖ neg_EvE26 ‖ C1Token ‖ C2Token**	5.38

Tabelle 5.7: Model-Checking des Fehlerbaums zum Beispiel Weichen- und Signalsteuerung

Zerlegung an Gatter G1

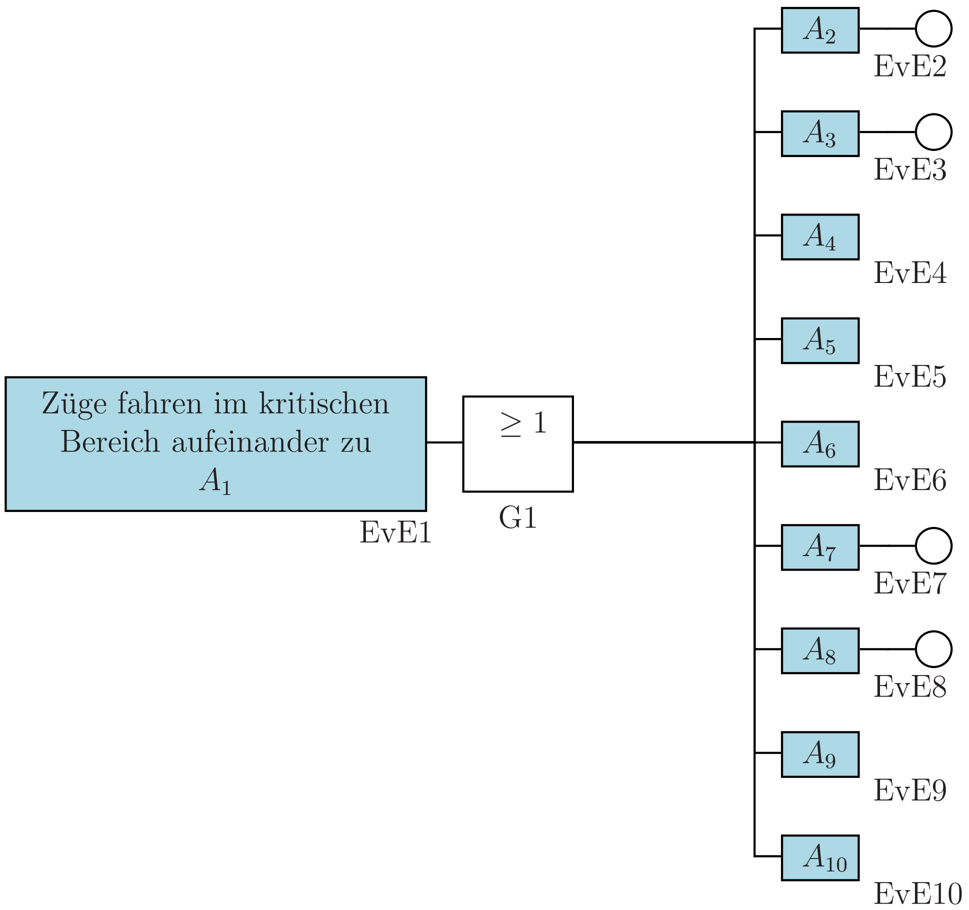
Das Top-Ereignis wird im ersten Schritt entsprechend der Formel A_1 in die möglichen Fälle zu zerlegen. Die Zerlegung ist in Abbildung 5.22 dargestellt. Für die Vollständigkeit muss die Beziehung folgende Beziehung gezeigt werden:

$$\begin{aligned}&\boxed{L}_0 A_{\mathcal{M}} \Rightarrow (\boxed{L}_0 A_1 \Rightarrow \\ &\qquad (\boxed{L}_0 A_2 \vee \boxed{L}_0 A_3 \vee \boxed{L}_0 A_4 \vee \boxed{L}_0 A_5 \vee \boxed{L}_0 A_6 \vee \boxed{L}_0 A_7 \vee \boxed{L}_0 A_8 \vee \boxed{L}_0 A_9 \vee \boxed{L}_0 A_{10})).\end{aligned} \tag{5.2.5.2}$$

Obwohl die Beziehung nach der Definition offensichtlich ist, kann sie auch über Phasenautomaten überprüft werden. Dazu wird ein Automat **EvE1** zu der Formel A_{Ev1} mit Hilfe der für das 3. Pattern (Sequenz-Pattern) angegebene Konstruktion und der Oder-Konstruktion erzeugt. Das Ergebnis ist in Abbildung 5.23 nur als Ausschnitt dargestellt, da der Automat sehr groß ist. Außerdem werden Automaten zu den Formeln $\neg\boxed{L}_0 A_n$, $1 \leq n \leq 9$ konstruiert wie für das 3. Pattern angegeben. Die Automaten sind in den Abbildungen 5.24, 5.25 und 5.26 dargestellt. Mit diesen Automaten kann die Bedingung 5.2.5.2 als Leerheitstest von

$$\begin{gathered}\mathbf{EvE1} \| \mathbf{neg\_EvE2} \| \mathbf{neg\_EvE3} \| \mathbf{neg\_EvE4} \| \mathbf{neg\_EvE5} \| \\ \mathbf{neg\_EvE6} \| \mathbf{neg\_EvE7} \| \mathbf{neg\_EvE8} \| \mathbf{neg\_EvE9} \| \mathbf{neg\_EvE10}\end{gathered}$$

überprüft werden.



Ereignis	**Beschreibung**
EvE2	Zug1 in krit. Bereich. Einfahrt Zug2 in krit. Bereich von links. Zug1 ändert Fahrtrichtung. Züge fahren aufeinander zu.
EvE3	Zug1 in krit. Bereich. Einfahrt Zug2 in krit. Bereich von rechts. Zug1 ändert Fahrtrichtung. Züge fahren aufeinander zu.
EvE4	Zug1 in krit. Bereich. Einfahrt Zug2 in krit. Bereich von links. Züge fahren aufeinander zu.
EvE5	Zug1 in krit. Bereich. Einfahrt Zug2 in krit. Bereich von rechts. Züge fahren aufeinander zu.
EvE6	Beide Züge fahren gleichzeitig in krit. Bereich. Züge fahren aufeinander zu.
EvE7	Zug2 in krit. Bereich. Einfahrt Zug1 in krit. Bereich von links. Zug2 ändert Fahrtrichtung. Züge fahren aufeinander zu.
EvE8	Zug2 in krit. Bereich. Einfahrt Zug1 in krit. Bereich von rechts. Zug2 ändert Fahrtrichtung. Züge fahren aufeinander zu.
EvE9	Zug2 in krit. Bereich. Einfahrt Zug1 in krit. Bereich von links. Züge fahren aufeinander zu.
EvE10	Zug2 in krit. Bereich. Einfahrt Zug1 in krit. Bereich von rechts. Züge fahren aufeinander zu.

Abbildung 5.22: Fehlerbaum zum Beispiel Weichen- und Signalsteuerung (Gatter G1)

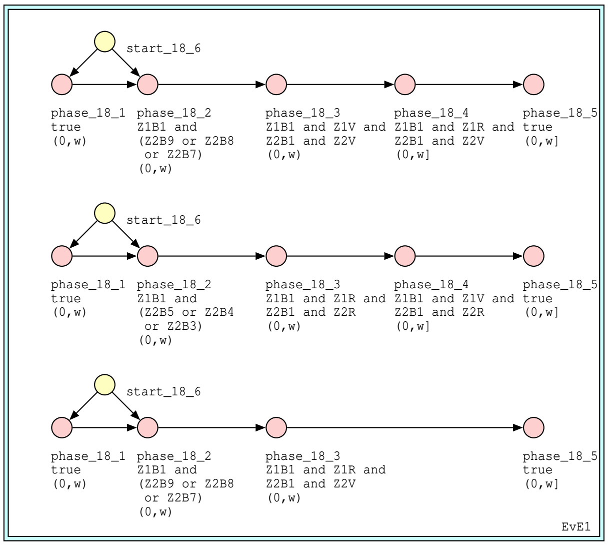


Abbildung 5.23: Phasenautomaten zu Ereignis EvE1 (nur ein Ausschnitt)

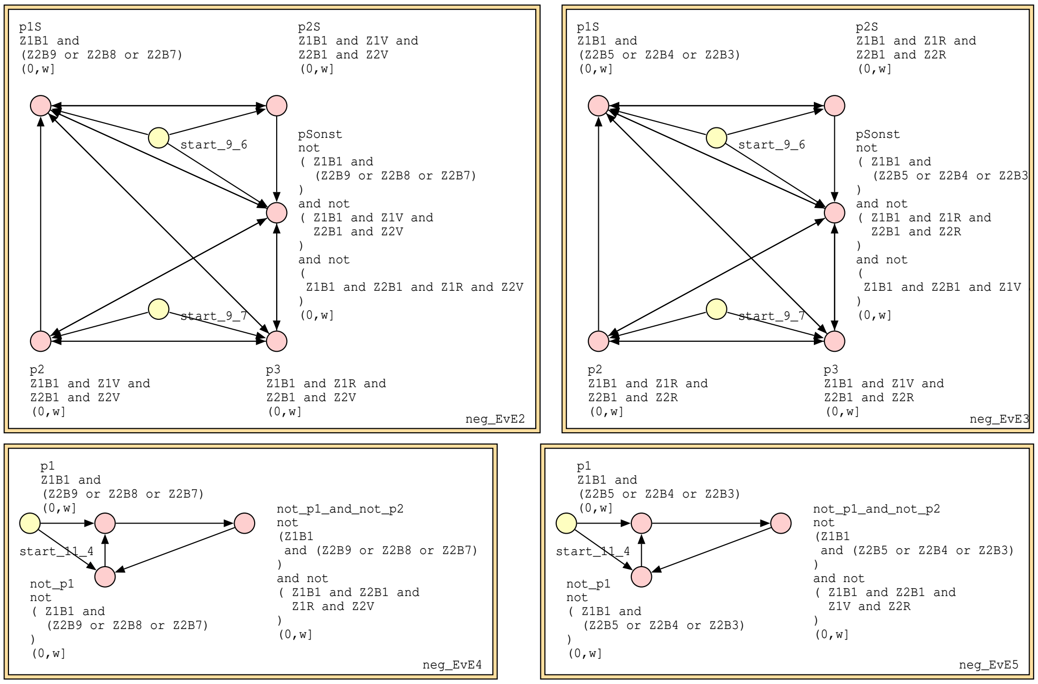


Abbildung 5.24: Phasenautomaten zu Ereignissen EvE2 bis EvE5 (Komplemente)

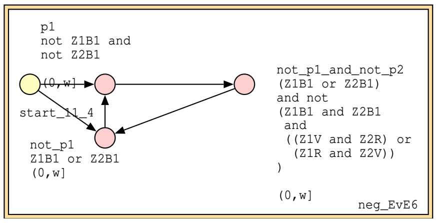


Abbildung 5.25: Phasenautomat zu Ereignis EvE6 (Komplement)

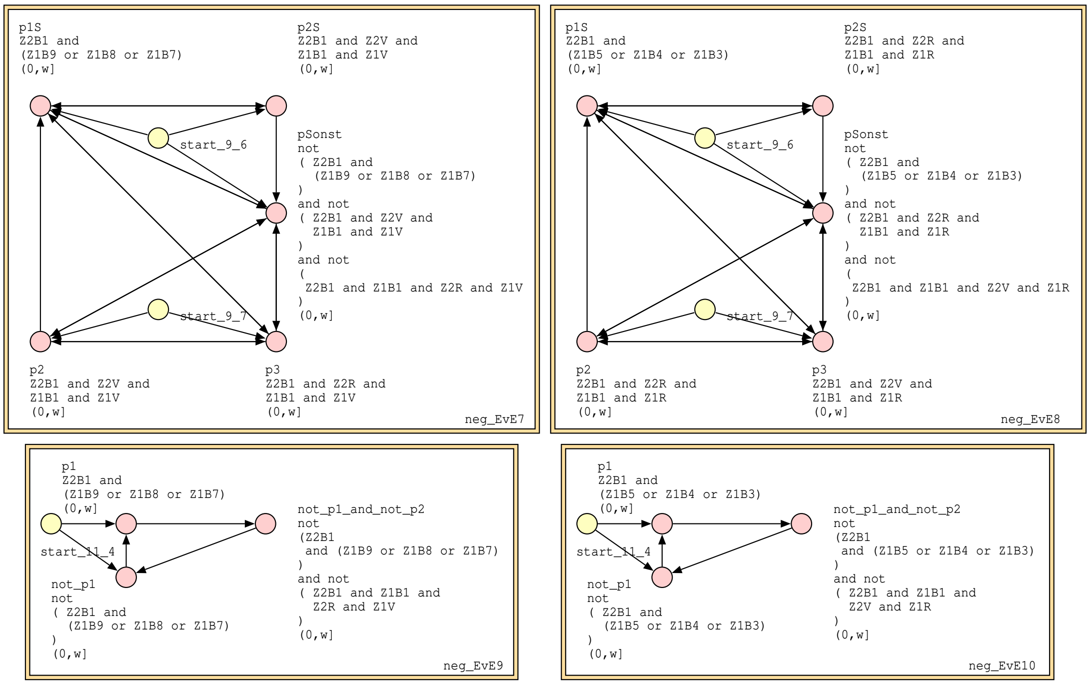


Abbildung 5.26: Phasenautomaten zu Ereignissen EvE7 bis EvE10 (Komplemente)

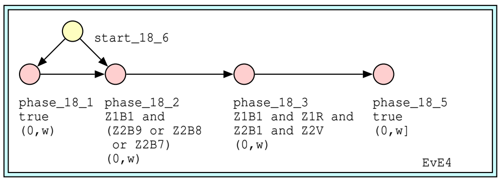


Abbildung 5.27: Phasenautomaten zu Ereignis EvE4

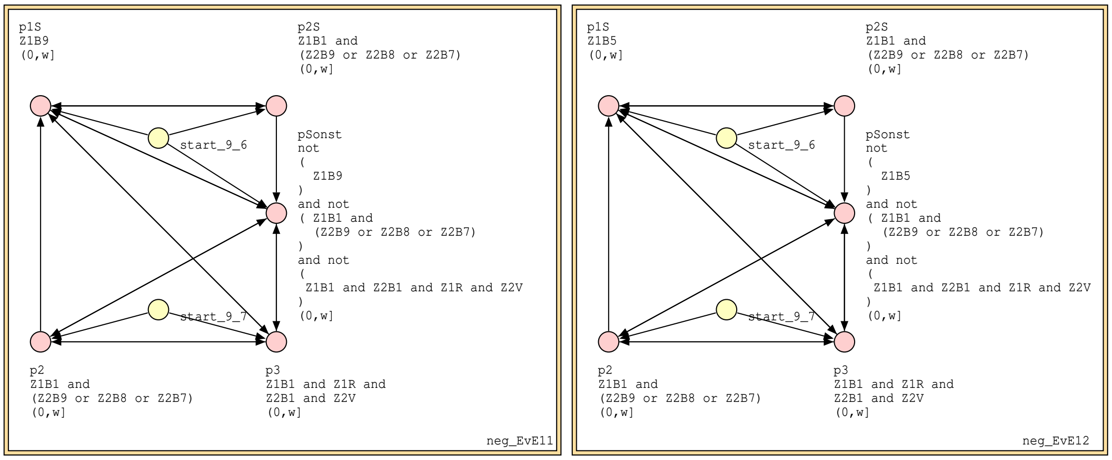


Abbildung 5.28: Phasenautomaten zu Ereignissen EvE11 und EvE12 (Komplemente)

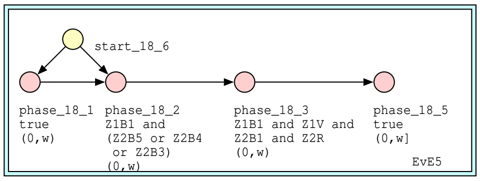


Abbildung 5.29: Phasenautomaten zu Ereignis EvE5

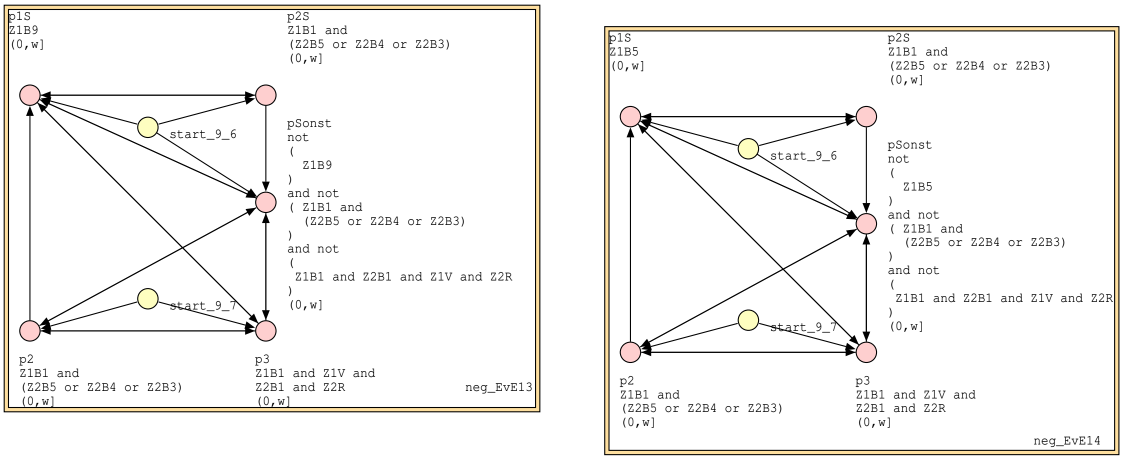


Abbildung 5.30: Phasenautomaten zu Ereignissen EvE13 und EvE14 (Komplemente)

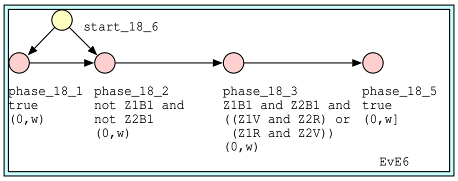


Abbildung 5.31: Phasenautomaten zu Ereignis EvE6

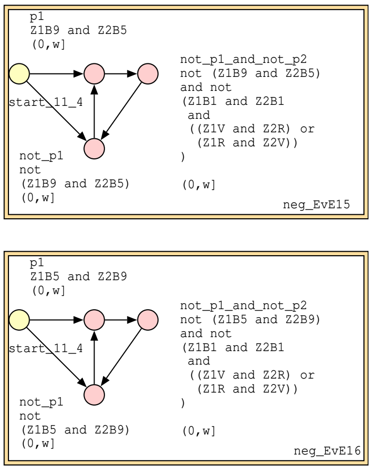


Abbildung 5.32: Phasenautomaten zu Ereignissen EvE15 und EvE16 (Komplemente)

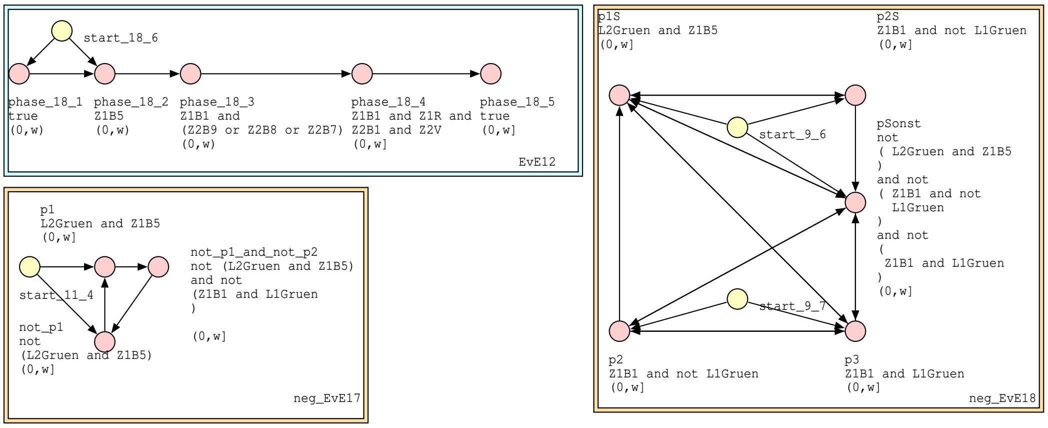


Abbildung 5.33: Phasenautomaten zu Gatter G5

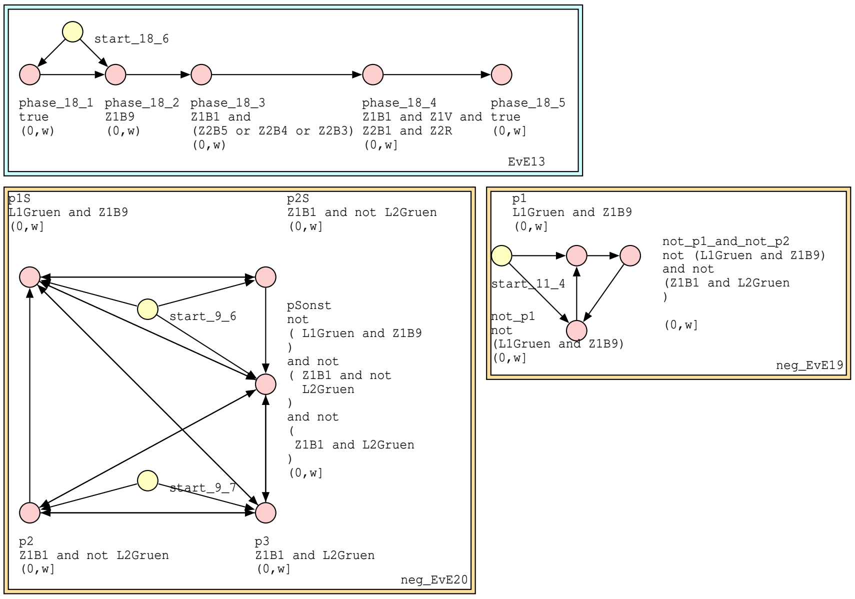


Abbildung 5.34: Phasenautomaten zu Gatter G6

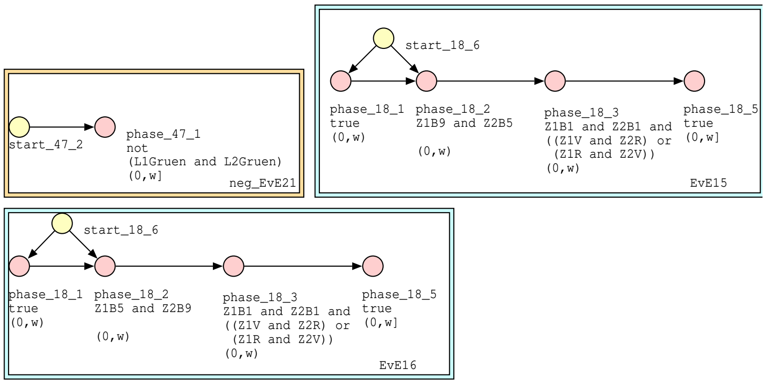


Abbildung 5.35: Phasenautomaten zu Gatter G7 und Gatter G8

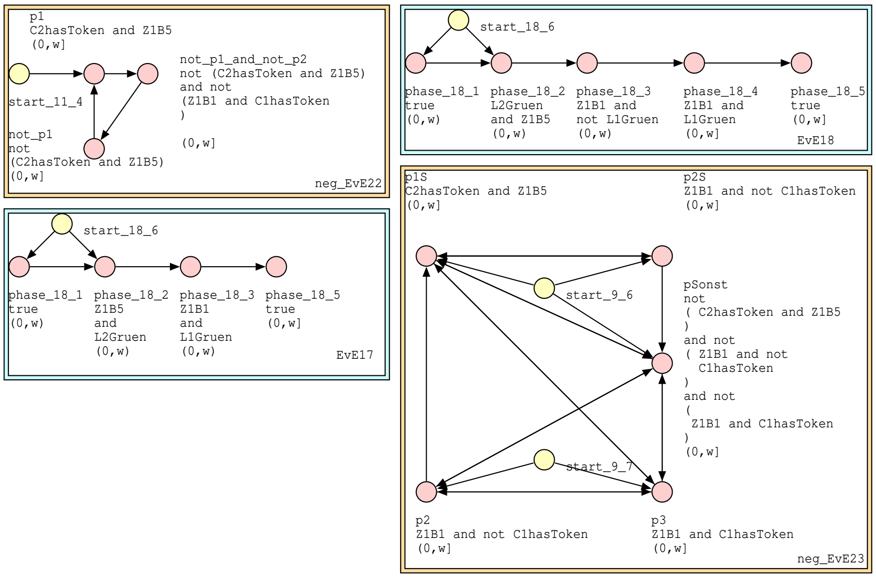


Abbildung 5.36: Phasenautomaten zu Gatter G9 und Gatter G10

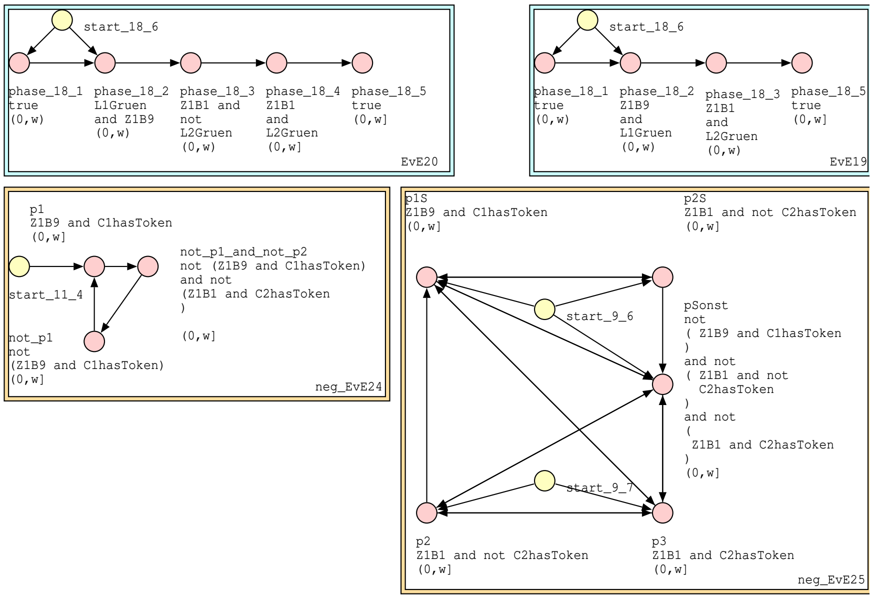


Abbildung 5.37: Phasenautomaten zu Gatter G11 und Gatter G12

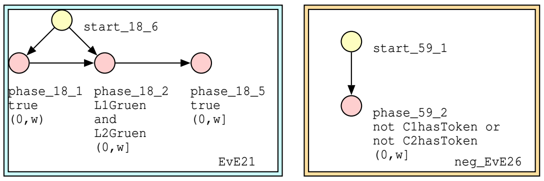


Abbildung 5.38: Phasenautomaten zu Gatter G13 und Gatter G14

5.2.5 Basisereignisse ausschließen

Das System hat folgende Eigenschaften.

Eig-1 Es existiert maximal ein Token.

Eig-2 Wenn über eine Weiche ein Zug in den kritischen Bereich einfährt, besitzt der zugeordnete Controller das Token und gibt es erst wieder ab, wenn alle eingefahrenen Züge den kritischen Bereich wieder verlassen haben.

Eig-3 Bei „Folgefahrten“ achten die Züge selbst darauf, die Richtung nicht zu wechseln.
Die Eigenschaften Eig-1 und Eig-2 konnten mit Moby/plc verifiziert werden. Die dritte Eigenschaft Eig-3 hat das System nach Voraussetzung. Mit diesen Eigenschaften können alle Basisereignisse ausgeschlossen werden. Da für den Fehlerbaum gezeigt ist, dass er vollständig ist, folgt auch, dass das Top-Ereignis „Züge fahren aufeinander zu“ nicht eintreten kann.

Eigenschaft Eig-1 schließt das Basisereigniss EvE26 aus.

Eigenschaft Eig-2 schließt die Basisereignisse EvE22, EvE23, EvE24 und EvE25 aus.

Eigenschaft Eig-3 schließt die Basisereignisse EvE2, EvE3, EvE7, EvE8, EvE11 und EvE14 aus.

Nicht weiter untersuchte Ereignisse Die Ereignisse EvE9 und EvE10 werden nicht weiter untersucht, da sie den Ereignissen EvE4 und EvE5 entsprechen, wobei nur Zug1 und Zug2 vertauscht sind.

5.2.6 Zusammenfassung

Für das Fallbeispiel konnte ein Fehlerbaum aufgestellt und die Vollständigkeit verifiziert werden. Da die Basisereignisse entweder durch formale Verifikation der Implementierung oder durch die Voraussetzungen ausgeschlossen werden konnten, ist damit gezeigt, dass das Ereignis nicht eintreten kann. Das Beispiel zeigt eine Möglichkeit des Model-Checkings auch für komplexe Systeme. Dabei werden die Verifikationsergebnisse von Teilkomponenten zu einer Aussage für das Gesamtsystem zusammengesetzt. Es ist allerdings noch unklar, ob dadurch die Verifikation vereinfacht wird. Zwar ist die Verifikation der Teilaussagen weniger aufwendig, als gleich die Verifikation, dass das

Top-Ereignis nicht eintreten kann. Dafür entsteht Aufwand durch das Model-Checking des Fehlerbaums.

6 Ausblick

Zum Abschluss sollen die erreichten Ergebnisse noch einmal im Zusammenhang eingeordnet und noch offene Fragen betrachtet werden.
Es konnte gezeigt werden, *dass und wie* für eine Teilklasse von Fehlerbäumen ein Model-Checking mit Hilfe von Phasenautomaten möglich ist. Einschränkungen beim Model-Checking ergeben sich unter anderem daraus, dass am Anfang gezeigt wurde, dass Phasenautomaten äquivalent zu Realzeitautomaten und somit nicht komplementierbar sind. Anhand der betrachteten Fallbeispiele wurde aber deutlich, dass die Teilklasse, für die Model-Checking möglich ist, groß genug ist, um realistische Probleme sinnvoll untersuchen zu können aber auch welche Probleme und Schwierigkeiten dabei auftreten.

- Aus der informellen Systembeschreibung muss eine Modellierung durch Phasenautomaten erstellt werden. Die Verwendbarkeit der Model-Checking-Ergebnisse hängt stark davon ab, wie genau die Modellierung zum System passt.
- Die umgangssprachliche Beschreibung der Ereignisse muss in DCL-Formeln übertragen werden.
- Es müssen eine Vielzahl von Phasenautomaten konstruiert werden. Für den praktischen Einsatz ist in Zukunft daher die Unterstützung durch Tools interessant.
- Die angegebenen Pattern sind nicht in jedem Fall ausreichend.

Trotz dieser Probleme scheint Model-Checking für Fehlerbäume Sinn zu machen, um die Sicherheit der Ursachenanalyse zu erhöhen oder um die Methode der Fehlerbaumanalyse als Werkzeug für formale Verfikation benutzen zu können. Ob durch den Einsatz der Fehlerbaumanalyse der Aufwand für das Model-Checking verringert werden kann, muss allerdings noch untersucht werden. Ebenfalls offen ist noch, ob die Ergebnisse auch auf andere Analysemethoden wie die Ereignisbaumanalyse übertragen werden können.

Index

Literaturverzeichnis

[ACM97] Eugene Asarin, Paul Caspi und Oded Maler. A Kleene Theorem for Timed Automata. In *Logic in Computer Science*, Seiten 160–171, 1997.

[AD94] Rajeev Alur und David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[Alu99] Rajeev Alur. Timed Automata. In Nicolas Halbwachs und Doron Peled, Hrsg., *Computer Aided Verification, 11th International Conference, CAV '99, Trento, Italy, July 6-10, 1999, Proceedings*, volume 1633 of Lecture Notes in Computer Science, Seiten 8–22. Springer, 1999.

[BA93] Glenn Bruns und Stuart Anderson. Validating Safety Models with Fault Trees. In *SAFECOMP '93: the 12th international Conference on Computer Safety*, Seiten 21–30. Springer, 1993.

[Bah97] N.J. Bahr. *System Safety Engineering and Risk Assessment: A Practical Approach*. Taylor & Francis, 1997.

[Bec83] Axel Becker. *Automatische Fehlerbaumerstellung*. Dissertation, Technische Universität Berlin, 1983.

[Bra01] Jens Braband. Lessons Learned from Risk Assessment Studies in Railway Signalling. In *Kolloquium Moderne Sicherheitsanalysen - Theorie und Praxis, 8/9. Oktober 2001, Braunschweig*, Seiten 3–15, 2001.

[BS93] Jonathan Bowen und Victoria Stavridou. Safety-critical systems, formal methods and standards. *Software Engineering Journal*, 8(4):189–209, Juli 1993.

[Buc00] Kerstin Buchacker. *Definition und Auswertung erweiterter Fehlerbäume für die Zuverlässigkeitsanalyse technischer Systeme*. Dissertation, Friedrich-Alexander-Universität Erlangen Nürnberg, 2000.

[CM93] Stephen J. Clarke und John A. McDermid. Software fault trees and weakest preconditions: a comparison and analysis. *Software Engineering Journal*, 8(4):225–236, Juli 1993.

[DD97] Henning Dierks und Cheryl Dietz. Graphical Specification and Reasoning: Case Study "Generalized Railroad Crossing". In C.B. Jones und P. Lucas, Hrsg., *FME'97*, volume 1313 of Lecture Notes in Computer Science, Seiten 20–39. Springer-Verlag, 1997.

[DIN81] DIN 25424: Fehlerbaumanalyse, Teil 1: Methode und Bildzeichen, 1981.

[DPS+01] Henning Dierks, Mahboubeh Pakdaman, Lemyah Salih, Andreas Schäfer, Rene Schumann und Tobe Toben. Ergebnisse des Fortgeschrittenen-Praktikums Realzeitsysteme WS 2000/2001, 2001. `http://semantik.informatik.uni-oldenburg.de/teaching/fp_realzeitsys_ws0001/result/main.html` .

[Dug00] Joanne Bechta Dugan. Galileo: A Tool for Dynamic Fault Tree Analysis. In B.R. Haverkort et al., Hrsg., *Computer performance evaluation : modelling techniques and tools; TOOLS 2000*, volume 1786 of Lecture Notes in Computer Science, Seiten 328–331. Springer-Verlag Berlin Heidelberg, 2000.

[Gór94] Janusz Górski. Extending Safety Analysis Techniques with Formal Semantics. In Felix Redmill, Hrsg., *Technology and assessment of safety-critical systems : proceedings of the Second Safety-Critical Systems Symposium*, Seiten 147–163. Springer Verlag Berlin, 1994.

[Gra97] V. Grabowski. Graphische Spezifikationsformalismen: Message Sequence Charts und Constraint Diagrams. Diplomarbeit, Carl von Ossietzky Universität Oldenburg, Fachbereich Informatik, 1997.

[Han96] Kirsten Mark Hansen. *Linking Safety Analysis to Safety Requirements.* Dissertation, Institut for Informationsteknologi, 1996.

[IEC93] DIN IEC 61025: Störungsbaumanalyse, 1993.

[Lev95] Nancy G. Leveson. *Software: System Safety and Computers.* Addison Wesley, 1995.

[Lut99] Wolfram Luther. Accurate robot reliability estimation by fault tree techniques. Technical Report 459, Fachbereich Mathematik, Gerhard-Mercator Universität Duisburg, 1999. Schriftenreihe des Fachbereichs Mathematik.

[Mru00] Christian Mrugalla. Übersetzung von Phasenautomaten in Timed Automata. Diplomarbeit, Carl von Ossietzky Universität Oldenburg, Fachbereich Informatik, 2000.

[MW90] David J. Mahar und James W. Wilbur. *Fault Tree Analysis Application Guide.* Reliability Analysis Center, Department of Defense USA, 1990.

[PC84] M. Elisabeth Pate-Cornell. Fault Trees vs. Event Trees in Reliability Analysis. *Risk Analysis*, 4(3):177–186, 1984.

[Ren01] Frank Renpenning. Gefährdungsanalyse am Beispiel des FunkFahrBetriebs. In *Kolloquium Moderne Sicherheitsanalysen - Theorie und Praxis, 8/9. Oktober 2001, Braunschweig*, 2001.

[RST00] W. Reif, G. Schellhorn und A. Thums. Safety Analysis of a Radio-Based Crossing Control System Using Formal Methods. In *Proceedings of the 9th IFAC Symposium Control in Transportation Systems 2000, June 13-15, Braunschweig, Germany*, 2000.

[RST01] W. Reif, G. Schellhorn und A. Thums. Integration formaler Spezifikationen und Sicherheitsanalyse. Technical Report 2001-6, Institut für Informatik, Universität Augsburg, Juni 2001.

[Sch84] Winfried G. Schneeweiss. Tutorial on Advanced Concepts in Fault Tree Analysis. Technical Report 52, FernUniversität Hagen, 12 1984.

[Sch99] Winfried G. Schneeweiss. *The Fault Tree Method*. LiLoLe-Verlag, 1999.

[Ska94] Jens Ulrik Skakkebæk. Liveness and Fairness in Duration Calculus. In B. Jonsson und J. Parrow, Hrsg., *CONCUR'94*, volume 836 of Lecture Notes in Computer Science, Seiten 283–298. Springer-Verlag, 1994.

[Sto96] Neil Storey. *Safety Critical Computer Systems*. Addison Wesley Longman, 1996.

[Tap01] Josef Tapken. *Model-Checking of Duration Calculus Specifikations*. Dissertation, Carl von Ossietzky Universität Oldenburg, 2001.

[Thu01] Andreas Thums. The Pressure Tank Example. Universität Augsburg, September 2001.

[VGRH81] W.E. Veseley, F.F. Goldberg, N.H. Roberts und D.F. Haasl. *Fault Tree Handbook*. Washington DC: US Nuclear Regulatory Commission, NUREG-0492, 1981.

[ZDL95] Zhou Chaochen, Dang Van Hung und Li Xiaoshan. A Duration Calculus with Inifite Intervals. In H. Reichel, Hrsg., *Fundamentals of Computation Theory. Proceedings.*, volume 965 of Lecture Notes in Computer Science, Seiten 16–41. Berlin, Heidelberg, New York, etc.: Springer Verlag, 1995.

[ZHR91] Zhou Chaochen, C.A.R. Hoare und Anders P. Ravn. A calculus of durations. *Information Processing Letters*, 40(5):269–276, 1991.

Erklärung

Hiermit versichere ich, dass ich diese Arbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel und Quellen benutzt habe.