Protokollverifikation in Temporallogik: Evolving Algebras und ein Tableaukalkül

Wolfgang May

Diplomarbeit

am

Institut für Logik, Komplexität und Deduktionssysteme
Universität Karlsruhe

Betreuer: Prof. Dr. P. H. Schmitt

Karlsruhe, Februar 1995

Zusammenfassung

In dieser Arbeit wird ein Konzept zur Beschreibung und Verifikation von Protokollen vorgestellt. Dabei wird ein Beschreibungsformalismus gewählt, der einer modallogischen Axiomatisierung durch eine Temporallogik für verzweigende Zeit zugänglich ist. Für diese Temporallogik wird ein Tableaukalkül entwickelt, mit dem die Korrektheit der Protokolle formal bewiesen werden kann.

Bei dem verwendeten Ansatz werden Protokolle durch Evolving Algebras beschrieben. Diese wurden in [Gurevich; 1988] zur Beschreibung operationaler Semantik im Sinne von Turings Satz eingeführt. Zu einem gegebenen Algorithmus können auf beliebigen Abstraktionsebenen Evolving Algebras entworfen werden, die diesen Algorithmus modellieren. Die einzelnen Berechnungszustände werden dabei durch (statische) Algebren, d.h. prädikatenlogische Strukturen, deren einziges Prädikat die Gleichheit ist, modelliert. Eine Evolving Algebra wird durch eine statische Algebra als Anfangskonfiguration und Übergangsregeln gegeben. Dies definiert einen Zustandsübergangsgraphen, der aus logischer Sicht als verzweigende temporale Kripke-Struktur aufgefaßt wird.

Zu deren Beschreibung dient eine auf den temporalen (Aussagen)-Logiken CTL bzw. CTL* [Ben-Ari, Manna, Pnueli; 1981], [Emerson, Halpern; 1983] aufbauende Quantorenlogik inklusive Gleichheit.
Aus der Modellierung des Protokolls als Evolving Algebra wird eine Axiomatisierung in CTL-Syntax generiert. Dies kann rein syntaktisch erfolgen. Die weiteren in den Beweis eingehenden Voraussetzungen, wie z.B. die nicht in CTL beschreibbaren Fairnessannahmen, sowie die zu beweisenden Aussagen werden in CTL* formuliert.

Es wird eine auf prädikatenlogischen Tableaux basierende Tableausemantik für Logik verzweigender Zeit entwickelt, bei der Zustände und Pfade der Kripke-Strukturen explizit benannt werden. Sie ermöglicht es, von endlich langen Pfadabschnitten zu abstrahieren, um so Erreichbarkeitsaussagen ohne graphentheoretische Zusatzalgorithmen verarbeiten zu können. Für diese Semantik wird ein Tableaukalkül für prädikatenlogisches CTL definiert. Weiter wird eine CTL*-Formelklasse spezifiziert, mit der Fairness- und Persistenzeigenschaften, die in CTL nicht formulierbar sind, beschrieben werden können. Diese Formelklasse kann aufgrund ihrer speziellen Semantik in den Kalkül integriert werden. Zur Behandlung der in der Spezifikation vorkommenden Datenstrukturen werden zusätzlich entsprechende Induktionsaxiome aufgenommen.

Für die komplette Logik CTL* wird eine Erweiterung des Tableaukalküls angegeben.

Damit liegt ein Verfahren vor, mit dem sich (Korrektheits)beweise formal führen lassen. Um die praktische Relevanz des Verfahrens zu zeigen, wird die Korrektheit des Alternating-Bit-Protokolls basierend auf einer Modellierung als Evolving Algebra bewiesen.

Das Verfahren ist ohne Einschränkung auch auf beliebige Prozesse anwendbar, falls diese in Einzelschritte zerlegbar sind.

Inhaltsverzeichnis

Einleitung

Zielsetzung und Problematik

Am Anfang ... war der Wunsch nach nachweisbar korrekten Protokollen (bzw. allgemein Prozessen). Solche Korrektheitsbeweise werden derzeit – wenn überhaupt – im mathematisch-argumentativen Stil geführt.

Zielsetzung dieser Arbeit war es, den Prozeß des Designs korrekter Protokolle bzw. der Verifikation existierender Protokolle zu uniformisieren und zu formalisieren. Durch Anwendung formaler Spezifikations-, Modellierungs- und Verifikationstechniken sollte ein durchgängiges Konzept, das ein Protokoll von der Idee über Spezifikation, Modellierung, Axiomatisierung und Verifikation bis zur Korrektheit begleitet, entworfen werden.

Als „Fixpunkte“ waren die Modellierung durch Evolving Algebras [Gu88, Gu91], die Axiomatisierung in einer Temporallogik für verzweigende Zeit aus der CTL-Familie [CE81, BMP81, EH83] sowie die Verwendung eines auf [BMP81, Wol85] basierenden Tableaukalküls vorgesehen.

Der Schritt von der Idee über bekannte formale Spezifikationstechniken (algebraische Spezifikation, Z, Zustandsübergangsdiagramme, ...) zur Modellierung als Evolving Algebra ist dabei unproblematisch. Der Wechsel von der dynamischen Sichtweise der Evolving Algebra als Transitionssystem zur logischen, statischen Sichtweise als temporale Kripke-Struktur erfolgt durch geeignete Definitionen. Die Axiomatisierung dieser Kripke-Struktur in einer prädikatenlogischen Version einer Temporallogik der CTL-Familie ist aus den gegebenen Übergangsregeln rein syntaktisch vornehmbar.

Eine unerwartete Lücke im Konzept ergab sich aus der Tatsache, daß der genannte Tableaukalkül die sich bei der Verifikation von Prozessen ergebenden Anforderungen nicht erfüllt und auch nicht in die entsprechenden Richtungen erweiterbar ist:

Der Kalkül ist für (aussagenlogisches) CTL konstruiert. Dabei wird vom Startzustand ausgehend iterativ jeder mögliche Nachfolgerzustand betrachtet. Sein Funktionieren basiert darauf, daß nur endlich viele verschiedene Zustände existieren können, womit die Tableaukonstruktion endlich ist und Zyklen enthält. Damit ist es möglich und notwendig, zur Überprüfung von Erreichbarkeitsaussagen eine graphentheoretische Nachbehandlung anzuschließen. Weiterhin sind Fairnessaussagen nicht in CTL, sondern erst in CTL* formulierbar, und damit ebenfalls nicht mit dem Kalkül zu verarbeiten.

Eine Einschränkung auf Aussagenlogik bzw. endlich viele Zustände erscheint in der vorgesehenen Anwendung aufgrund der vorzunehmenden Abstraktionen nicht attraktiv. Da damit auch das Problem der Fairnessaussagen noch nicht gelöst würde, und für dieses auch keine Lösung in Sicht ist, wurde von einer Verwendung dieses Tableaukalküls Abstand genommen. Statt dessen wurde diese Lücke durch Konstruktion eines neuen Tableaukalküls geschlossen, was entsprechend den Schwerpunkt der Arbeit etwas verlagerte.

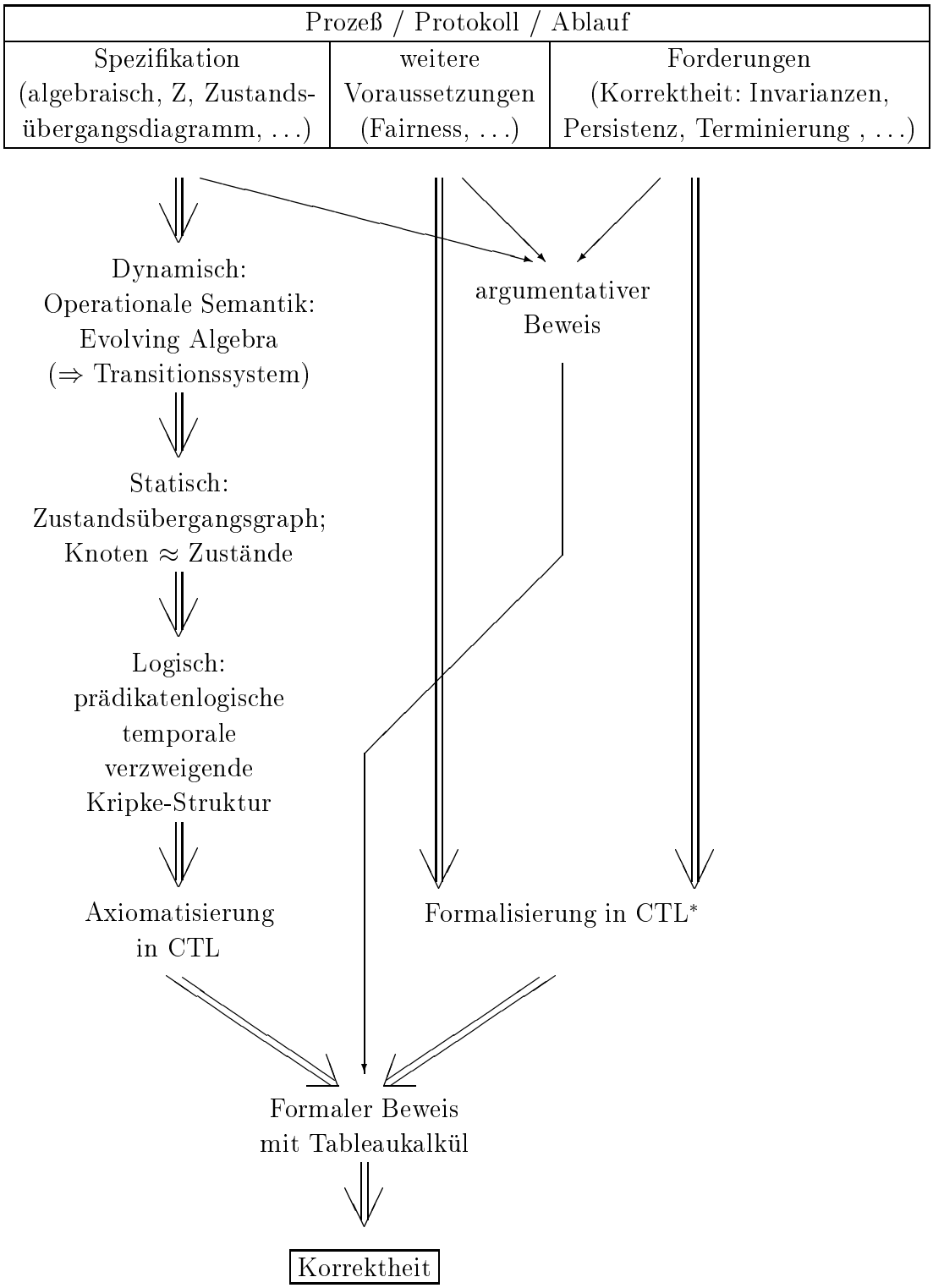
Ausgehend von den bei dieser Untersuchung gewonnenen Ergebnissen wurden die Grundgedanken für den neu zu konstruierenden Tableaukalkül aufgestellt:

- prädikatenlogisches CTL,
- explizite Beschreibung der „geographischen“ Struktur eines Modells durch Benennung von Zuständen und Pfaden,
- daher Verwendung eines Kalküls mit Präfixen und Codierung dieser „geographischen“ Informationen im Tableau,
- Abstraktion von endlich langen Pfadabschnitten, um so Erreichbarkeitsaussagen direkt verarbeiten zu können.

Diesen Anforderungen entsprechend wurde eine Tableausyntax und -semantik sowie ein Tableaukalkül für prädikatenlogisches CTL entwickelt. Die gewählte Semantik ermöglicht die Einbeziehung einiger Formelklassen aus CTL*, so daß mit diesem Kalkül Fairnessaussagen ebenfalls verarbeitet werden können. Als Variante entstand für dieselbe Semantik ein Tableaukalkül für prädikatenlogisches CTL*.

Damit ist das angestrebte Verfahren komplettiert.

Aufgrund von Umfang und Komplexität der zu führenden Beweise müssen diese in Teilbeweise, den einzelnen Teilüberlegungen während des Entwurfs entsprechend, aufgeteilt werden. Da Evolving Algebras auf unterschiedlichen Abstraktionsebenen erstellt werden können, ist aus dieser Richtung ebenfalls eine Aufspaltung in Teilaufgaben zu erwarten. Die formale Durchführung des Verfahrens orientiert sich somit weiterhin an argumentativ geführten Beweis(skizz)en, womit sich der Gesamtablauf folgendermaßen präsentiert:



Ablauf des Gesamtverfahrens zur Modellierung und Verifikation von Prozessen

Aufbau der Arbeit

Im 1. Kapitel werden die Grundlagen der verwendeten Themengebiete zusammengestellt und die in dieser Arbeit verwendete Bezeichnungsweise eingeführt. In diesem Zusammenhang sei auch auf den am Ende der Arbeit zu findenden Überblick über die verwendete Notation hingewiesen.

In Kapitel 2 wird der Weg vom Protokoll über eine Evolving Algebra zu einer Kripke-Struktur und deren Axiomatisierung in prädikatenlogischem CTL beschrieben. Dabei zeigt sich, daß die Evolving Algebra neben der unmittelbaren Beschreibung der operationalen Semantik i.a. zusätzliche Kontrollinformationen enthalten muß. Die Übersetzung der erhaltenen Evolving Algebra in eine temporale Kripke-Struktur sowie deren Axiomatisierung kann dann rein syntaktisch erfolgen. Danach werden die daraus resultierenden Anforderungen an die Logik beschrieben sowie zwei bekannte Entscheidungsverfahren aus diesem Bereich vorgestellt.

Im 3. Kapitel wird eine auf prädikatenlogischen Tableaux aufbauende Tableausemantik entwickelt, die für verzweigende temporale Logiken der CTL-Familie geeignet ist, und das Grundprinzip der Behandlung temporallogischer Formeln vorgestellt.

In Kapitel 4 wird ein Tableaukalkül für CTL für diese Tableausemantik entwickelt. Dabei werden bestimmte Eigenschaften bei der Zerlegung von CTL-Formeln ausgenutzt, die eine relativ einfache Behandlung verzweigender Pfade ermöglichen.

Kapitel 5 enthält die theoretische Aufbereitung von Kapitel 4. Den zentralen Punkt bildet dabei der Korrektheitsbeweis des Kalküls. Ein Vollständigkeitssatz ist nicht zu erhalten, da die auf CTL basierende Prädikatenlogik nicht kompakt ist. Die Übertragung der Begriffe einer fairen Tableauprozedur und einer Hintikka-Menge ergeben eine relativierte Vollständigkeitsaussage, die zu einer Betrachtung der sich aus der Modellierung der Pfade ergebenden Induktionsproblematik überleitet. Danach wird ein erstes Fazit gezogen, wie eine Beweisführung mit Hilfe des Kalküls aussehen wird.

In Kapitel 6 wird der Kalkül im Hinblick auf einen praktischen Einsatz überarbeitet, womit der Verzweigungsgrad der Tableaux reduziert wird.

Kapitel 7 erweitert den Kalkül auf CTL^+, wobei sich zeigt, daß CTL^+ ohne grundlegende Änderungen zu verarbeiten ist, da die Ausdruckskraft von CTL und CTL^+ dieselbe ist.

Kapitel 8 betrachtet die Einbindung von Formeln linearer Zeit, woraus die Möglichkeit resultiert, u.a. auch Fairnessforderungen an die beschriebene Kripke-Struktur mit dem Kalkül zu verarbeiten. Danach wird die Ausdruckskraft des so erhaltenen Kalküls bezüglich temporallogischer Formeln untersucht.

Kapitel 9 beschäftigt sich mit allgemeingültigen Formeln und zustandsunabhängig interpretierten Termen sowie deren Verwendung zu globalen Schlüssen. Im weiteren wird eine spezielle Optimierung für die Behandlung von Evolving Algebras angegeben.

In Kapitel 10 wird eine Erweiterung auf CTL^* vorgenommen. Die in Kapitel 4 zur Behandlung verzweigender Pfade ausgenutzten Eigenschaften werden von CTL^* nicht erfüllt. Daher muß an dieser Stelle die

Behandlung verzweigender Pfade grundlegend anders gestaltet werden. Es zeigt sich aber, daß auch dies möglich ist, und somit *prinzipiell* die Möglichkeit besteht, auch in CTL* zu verifizieren.

Kapitel 11 beschäftigt sich mit der Einschränkung des vorliegenden Kalküls auf Logik linearer Zeit sowie einer Betrachtung, in welchen Fällen diese ausreicht.

Kapitel 12 schließt den theoretischen Teil mit einem Fazit ab.

Kapitel 13 enthält einige – aussagenlogische – Beispieltableaux, die die Vorgehensweise des Kalküls veranschaulichen.

Den Abschluß bildet eine Zusammenstellung der verwendeten Notation sowie der verwendeten Literatur.

Der Korrektheitsbeweis des Alternating-Bit-Protokolls ist in [May95b] zu finden.

1 Grundlagen

In diesem Kapitel werden die behandelten Themengebiete sowie die jeweils verwendete Terminologie eingeführt.

1.1 Systeme, Protokolle

Die in diesem Abschnitt verwendeten Begriffe und Definitionen sind sinngemäß aus [LKK93] entnommen.

Unter einem *System* versteht man eine Menge von Elementen, die untereinander in wohldefinierten Beziehungen stehen. Diese Elemente werden als *Instanzen* bezeichnet. Sie müssen entsprechend der Bestimmung des Systems ein wohldefiniertes, durch Regeln beschreibbares Verhalten aufweisen. Unter den Instanzen können wiederum *Dienstgeber* und *Dienstnehmer* unterschieden werden, wobei diese Unterteilung jedoch stark von der eingenommenen Perspektive abhängt.
Systeme, deren Aufgabe die ordnungsgemäße Abwicklung von Kommunikationsfunktionen ist, werden als *Kommunikationssysteme* bezeichnet. Die meisten (informationsverarbeitenden) Systeme beinhalten als Subsystem ein Kommunikationssystem, das dem zweckgerichteten Austausch von Informationen zwischen Instanzen dient.
Betrachtet man die Kommunikation zwischen Instanzen, die eine übergeordnete Aufgabe erledigen, so stellt das Kommunikationssystem einen Dienstgeber (*Medium*) dar, der die technischen Mittel für die Kommunikation bereitstellt. In diesem Fall spricht man von *horizontaler Kommunikation* zwischen *Teilnehmern*. Ein Medium wird somit nicht als Instanz mit einem eigenen aktiven Verhalten betrachtet.
Ist zusätzlich das Verhalten des Kommunikationssystems von Bedeutung, so muß auch die Kommunikation zwischen Dienstnehmer und Dienstgeber betrachtet werden. In diesem Fall tritt das Kommunikationssystem als Instanz mit aktivem, von innen gesteuertem Verhalten in Erscheinung. Diese Form wird als *vertikale Kommunikation* bezeichnet.
In beiden Fällen beschreibt eine Menge von Regeln, das *(Kommunikations)protokoll*, wie die Kommunikation zwischen zwei oder mehr Instanzen abgewickelt wird.
Vertikale Kommunikationsprotokolle beschreiben, wie Dienstleistungen vom Dienstnehmer beim Dienstgeber anzufordern und umgekehrt die Ergebnisse mitzuteilen sind. Demgegenüber legen *horizontale Kommunikationsprotokolle* die Regeln fest, nach denen ein Informationsaustausch zwischen Teilnehmern abzulaufen hat.
Entsprechend den verschiedenen Abstraktionsebenen eines Systems können auch verschiedene Abstraktionsebenen der verwendeten Protokolle existieren.
Bekannte Beispiele für Protokolle sind das Alternating-Bit-Protokoll, Kermit (mit mehreren verschiedenen Ebenen) und die Protokolle HDLC, LLC, TCP/IP usw. entsprechend dem ISO/OSI-Schichtenmodell.

Bei dem vorgestellten Verfahren werden Protokolle durch Evolving Algebras modelliert.

1.2 Mathematische Grundlagen

Die Menge $\mathbf{N}$ der natürlichen Zahlen ist als $\mathbf{N} := \{0, 1, 2, \ldots\}$ definiert. Ergänzend ist $\hat{\mathbf{N}} := \mathbf{N} \cup \{\infty\}$, wobei die $<$-Relation durch $\forall n \in \mathbf{N} : n < \infty$ auf $\hat{\mathbf{N}} \times \hat{\mathbf{N}}$ fortgesetzt wird.

Definition 1 *Eine* Algebra *ist ein Tupel*

$$\mathcal{A} = \langle A, \mathcal{F} \rangle \quad mit \quad \exists n \in \mathbf{N} : \ \mathcal{F} = \{f_i : 1 \leq i \leq n\} \quad ,$$

wobei A als Trägermenge *bzw.* Universum *bezeichnet wird und die $f_i : A^{n_i} \to A$ n_i-stellige Operatoren sind. Diese Stelligkeit wird mit $\mathsf{ord}(f_i) := n_i$ bezeichnet. Ist $\mathsf{ord}(f_i) = 0$, so ist f_i eine* Konstante *und wird mit einem Element aus A identifiziert.*

Definition 2 *Seien M und N Mengen. Dann ist*

1. *M^N die Menge aller Abbildungen $\phi : N \to M$.*
2. *$2^N := \{0,1\}^N$ die* Potenzmenge *von N,*
3. *$M^{\mathbf{N}}$ die Menge aller* Folgen *in M, d.h.*

   $$M^{\mathbf{N}} = \{\bar{a} : \bar{a} = (a_n)_{n \in \mathbf{N}} = (a_0, a_1, a_2, \ldots) : a_i \in M\} \quad .$$

   Gemäß Teil 1 dieser Definition ist somit eine Folge $\bar{a}$ als Abbildung

   $$\bar{a} : \mathbf{N} \to M : i \mapsto a_i$$

   aufzufassen.

Definition 3 *Seien M_i, $i \in \{1, \ldots, n\}$ Mengen, Dann ist eine n-*stellige Relation *R (Bezeichnung: $\mathsf{ord}(R) := n$) über $M_1, \ldots, M_n$ eine Menge $R \subseteq M_1 \times M_2 \times \ldots \times M_n$.*

Ist $R \subseteq M \times M$ eine zweistellige Relation über M, so definiert man deren transitive bzw. reflexiv-transitive Hülle als

$$\begin{aligned} (a,b) \in R^+ \quad &\Leftrightarrow \quad \exists n > 0, \ a = g_0, \ g_1, \ldots, \ g_n = b : \forall i < n : (g_i, g_{i+1}) \in R \\ und \quad (a,b) \in R^* \quad &\Leftrightarrow \quad \exists n \geq 0, \ a = g_0, \ g_1, \ldots, \ g_n = b : \forall i < n : (g_i, g_{i+1}) \in R \quad . \end{aligned}$$

Definition 4 *Die Menge der* Wahrheitswerte *ist definiert als $\{true, false\}$.*

Damit kann eine Relation R über $M_1, \ldots, M_n$ ebenfalls durch ihre *charakteristische Funktion*

$$f_R : M_1 \times \ldots \times M_n \to \{true, false\} : \ f_R(m_1, \ldots, m_n) = true \ \Leftrightarrow \ (m_1, \ldots, m_n) \in R \ \text{ für } \ m_i \in M_i$$

beschrieben werden.

1.3 Prädikatenlogik 1.Ordnung

Die Prädikatenlogik der 1.Ordnung bildet die Grundlage des in dieser Arbeit vorgestellten Ansatzes. Die folgende Zusammenfassung basiert auf [MS91].

Syntax:

Jede Sprache der PL1 enthält eine Menge von Sonderzeichen:

Klammern: (,)
Bezeichner für Wahrheitswerte: true, false
boolesche Junktoren: $\neg$, $\wedge$, $\vee$, $\rightarrow$
Quantoren: $\forall$, $\exists$
(objektsprachliches) Gleichheitssymbol: $=$
Variablen: $x, y, x_1, x_2,$ eine unendliche Menge von Bezeichnern
Die Menge der Variablen wird mit Var bezeichnet.

Eine einzelne Sprache ist dann durch ihre *Signatur* Σ gegeben. Dies ist eine Menge von Funktionssymbolen und Prädikatssymbolen, jedes mit einer festgelegten Stelligkeit. Die Stelligkeit eines Funktionssymbols f bzw. eines Prädikatssymbols p wird hier ebenfalls mit $\mathsf{ord}(f)$ bzw. $\mathsf{ord}(p)$ bezeichnet. Nullstellige Funktionen werden auch als *Konstanten*, nullstellige Prädikatssymbole als *aussagenlogische Atome* bezeichnet.

Die Menge Term_Σ der *Terme* über Σ ist induktiv definiert durch

1. Jede Variable ist ein Term.
2. ist f ein Funktionssymbol, $\mathsf{ord}(f) = n$, $t_1, \ldots, t_n$ Terme, so ist $f(t_1, \ldots, t_n)$ ein Term.

Für $t = f(t_1, \ldots, t_n) \in \mathsf{Term}_\Sigma$ mit $\mathsf{ord}(f) = n$ ist $\hat{t} := f \in \Sigma$ das führende Funktionssymbol und $\mathsf{arg}(t) := (\mathsf{arg}_1(t), \ldots, \mathsf{arg}_n(t)) = (t_1, \ldots, t_n) \in \mathsf{Term}_\Sigma^n$ die Folge der Argumente.

Die Menge der *atomaren Formeln* über Σ ist definiert als

$$At := \{s = t : s, t \in \mathsf{Term}_\Sigma\} \cup \{p(t_1, \ldots, t_n) : p \text{ Prädikatssymbol}, \mathsf{ord}(p) = n, t_1, \ldots, t_n \in \mathsf{Term}_\Sigma\} \quad .$$

Die Menge der *Formeln* über Σ ist induktiv definiert durch

1. Alle atomaren Formeln sind Formeln.
2. true und false sind Formeln.
3. Sind A und B Formeln und x eine Variable, so sind auch $\neg A$, $A \wedge B$, $A \vee B$, $\forall x : A$ und $\exists x : A$ Formeln.

Die Begriffe *freie Variablen* und *gebundene Variablen* sind wie üblich definiert.

Eine Substitution (über einer Signatur Σ) ist eine Abbildung

$$\sigma : \mathsf{Var} \rightarrow \mathsf{Term}_\Sigma \quad \text{mit} \quad \sigma(x) = x \text{ für fast alle } x \in \mathsf{Var}.$$

Eine Substitution $\sigma : \sigma(x) = t$ wird als $[x \leftarrow t]$ geschrieben. Die Fortsetzung von Substitutionen auf Terme und Formeln ist wie üblich definiert.

Semantik:

Eine *prädikatenlogische Struktur* (auch als *Interpretation* bezeichnet) $\boldsymbol{I} = (I, \boldsymbol{U})$ zu einer Signatur Σ besteht aus einer nichtleeren Menge $\boldsymbol{U}$ (*Universum*) und einer Abbildung I der Signatursymbole, die

1. jeder Konstanten c ein Element $I(c) \in \boldsymbol{U}$,
2. für jedes n jedem Funktionssymbol f mit $\mathsf{ord}(f) = n$ eine Funktion $I(f) : \boldsymbol{U}^n \to \boldsymbol{U}$,
3. jedem aussagenlogischen Atom p einen Wahrheitswert $I(p) \in \{true, false\}$,
4. für jedes n jedem Prädikatssymbol p mit $\mathsf{ord}(p) = n$ eine n-stellige Relation $I(p) \subseteq \boldsymbol{U}^n$

zuordnet.

Eine *Variablenbelegung* zu einer Struktur $\boldsymbol{I}$ mit Universum $\boldsymbol{U}$ ist eine Funktion

$$\chi : \mathsf{Var} \to \boldsymbol{U} \quad .$$

Die Menge der Variablenbelegungen wird mit Ξ bezeichnet.

Für eine Variablenbelegung χ, Variablen x, y und ein $d \in \boldsymbol{U}$ ist die Variablenbelegung

$$\chi_x^d : \mathsf{Var} \to \boldsymbol{U} : \begin{cases} y \mapsto \chi(y) & \text{falls } y \neq x \quad , \\ x \mapsto d & \text{sonst} \end{cases}$$

die *Modifikation von χ an der Stelle x.*

Die zu einer Struktur $\boldsymbol{I}$ gehörende Auswertung

$$\boldsymbol{I} : \mathsf{Term}_\Sigma \times \Xi \to \boldsymbol{U}$$

von Termen unter einer Variablenbelegung $\chi \in \Xi$ ist mit entsprechender Fortsetzung auf Tupel von Termen $\boldsymbol{I} : \mathsf{Term}_\Sigma^n \times \Xi \to \boldsymbol{U}^n$ folgendermaßen definiert:

$$\begin{aligned}
&\boldsymbol{I}(x, \chi) := \chi(x) \quad \text{für eine Variable } x, \\
&\boldsymbol{I}((t_1, \ldots, t_n), \chi) := (\boldsymbol{I}(t_1, \chi), \ldots, \boldsymbol{I}(t_n, \chi)) \quad \text{für Terme } t_1, \ldots, t_n. \\
&\boldsymbol{I}(f(t_1, \ldots, t_n), \chi) := (I(f))(\boldsymbol{I}((t_1, \ldots, t_n), \chi)) = (I(f))(\boldsymbol{I}(t_1, \chi), \ldots, \boldsymbol{I}(t_n, \chi)) \\
&\qquad \text{für ein Funktionssymbol } f \in \Sigma, \mathsf{ord}(f) = n \text{ und Terme } t_1, \ldots, t_n.
\end{aligned}$$

Damit läßt sich die *Wahrheitsrelation* $\models := \models_{\mathrm{PL}}$ zwischen einer prädikatenlogischen Struktur $\boldsymbol{I}$, einer Variablenbelegung χ und Formeln wie folgt definieren:
Seien s, t Terme, p ein Prädikatssymbol, $\mathsf{ord}(f) = n$, $t_1, \ldots, t_n$ Terme, x eine Variable, A und B Formeln.

$$\begin{aligned}
&(\boldsymbol{I}, \chi) \models \mathsf{true} \\
&(\boldsymbol{I}, \chi) \models s = t && :\Leftrightarrow && \boldsymbol{I}(s, \chi) = \boldsymbol{I}(t, \chi) \\
&(\boldsymbol{I}, \chi) \models p(t_1, \ldots, t_n) && :\Leftrightarrow && (\boldsymbol{I}(t_1, \chi), \ldots, \boldsymbol{I}(t_n, \chi)) \in I(p) \\
&(\boldsymbol{I}, \chi) \models \neg A && :\Leftrightarrow && \text{nicht } (\boldsymbol{I}, \chi) \models A \\
&(\boldsymbol{I}, \chi) \models A \wedge B && :\Leftrightarrow && (\boldsymbol{I}, \chi) \models A \text{ und } (\boldsymbol{I}, \chi) \models B \\
&(\boldsymbol{I}, \chi) \models A \vee B && :\Leftrightarrow && (\boldsymbol{I}, \chi) \models A \text{ oder } (\boldsymbol{I}, \chi) \models B \\
&(\boldsymbol{I}, \chi) \models A \to B && :\Leftrightarrow && (\boldsymbol{I}, \chi) \models \neg A \text{ oder } (\boldsymbol{I}, \chi) \models B \\
&(\boldsymbol{I}, \chi) \models \forall x : A && :\Leftrightarrow && \text{für alle } d \in \boldsymbol{U} \text{ gilt } (\boldsymbol{I}, \chi_x^d) \models A \\
&(\boldsymbol{I}, \chi) \models \exists x : A && :\Leftrightarrow && \text{es gibt ein } d \in \boldsymbol{U} \text{ mit } (\boldsymbol{I}, \chi_x^d) \models A
\end{aligned}$$

Für Formeln ohne freie Variablen ist somit $\models$ eine Relation zwischen Strukturen und Formeln.

In diesem Zusammenhang sei das *Substitutionslemma* der Prädikatenlogik zitiert:

Sei Σ eine Signatur, $\boldsymbol{I} = (I, \boldsymbol{U})$ eine Interpretation, χ eine Variablenbelegung, x eine Variable und $s, t \in \mathsf{Term}_\Sigma$ Terme, F eine Formel. Für

$$a := \boldsymbol{I}(s, \chi) \in \boldsymbol{U} \quad \text{ist} \quad \boldsymbol{I}([x \leftarrow s]t, \chi) = \boldsymbol{I}(t, \chi^a_x) \quad \text{und} \quad (\boldsymbol{I}, \chi) \models [x \leftarrow s]F \;\Leftrightarrow\; (\boldsymbol{I}, \chi^a_x) \models F \quad .$$

Für die Prädikatenlogik ist ein Tableaukalkül durch die folgenden α- und β-Regeln [Sm68], die γ- und δ-Regeln [Ree87, Schm87, Fit90] bzw. (liberalized-δ-rule) [HS94] sowie die Abschlußregel gegeben:
Seien A eine atomare Formel, F und G Formeln, $\mathsf{free}(F)$ die Folge der in F freien Variablen und bezeichne $F[X/x]$ die durch Ersetzung aller Auftreten von x durch X aus F entstehende Formel.

α: $\dfrac{F \wedge G}{\begin{array}{c} F \\ G \end{array}}$ $\qquad$ $\dfrac{\neg(F \vee G)}{\begin{array}{c} \neg F \\ \neg G \end{array}}$

β: $\dfrac{F \vee G}{F \mid G}$ $\qquad$ $\dfrac{\neg(F \wedge G)}{\neg F \mid \neg G}$

γ: $\dfrac{\forall x : F}{F[X/x]}$ $\qquad$ $\dfrac{\neg\exists x : F}{\neg F[X/x]}$

wobei X eine noch nicht vorkommende Variable ist.

δ: $\dfrac{\exists x : F}{F[f(\mathsf{free}(F))/x]}$ $\qquad$ $\dfrac{\neg\forall x : F}{\neg F[f(\mathsf{free}(F))/x]}$

wobei f ein noch nicht verwendetes (Skolem)-funktionssymbol ist.

Abschlußregel: Sei σ eine Substitution:

$$\frac{\begin{array}{c} \sigma(A) \\ \neg\sigma(A) \end{array}}{\bot}$$

σ auf das gesamte Tableau anwenden.

1.4 Evolving Algebras

Das Konzept der „Evolving Algebras“ wurde 1988 von Y. Gurevich [Gu88, Gu91, Gu92] zur Beschreibung operationaler Semantik eingeführt.

Eine Struktur der Prädikatenlogik erster Ordnung, die als einziges Prädikat die objektsprachliche Gleichheit umfaßt, wird im Bereich der Universellen Algebra als *Algebra* bezeichnet. Um den Unterschied zur Evolving Algebra zu betonen, wird in dieser Arbeit dafür in Anlehnung an [Gu92] der Begriff *statische Algebra* verwendet.

In [Gu92] werden Evolving Algebras jedoch nicht aus logischer Sicht, sondern als Mittel zur Beschreibung operationaler Semantik im Sinne von Turings Satz „Jeder Algorithmus wird durch eine geeignete Turingmaschine simuliert“ eingeführt. Dabei können zu einem gegebenen Algorithmus auf beliebigen Abstraktionsebenen Evolving Algebras entworfen werden, die diesen Algorithmus darstellen.

Zur Beschreibung der einzelnen statischen Algebren werden – da sie prinzipiell prädikatenlogische Strukturen sind – dieselben Begriffe definiert:

Definitionen:
Die *Signatur* Σ einer statischen Algebra ist eine endliche Menge von Funktionssymbolen, jedes mit einer festgelegten Stelligkeit. Nullstellige Funktionen werden auch als *Konstanten* bezeichnet. Die Begriffe *Term* und *geschlossener Term* sind wie in der Prädikatenlogik definiert.
Eine *statische Algebra* $g = \mathcal{I} = (I, \mathcal{S})$ zu einer Signatur Σ ist eine nichtleere Menge $\mathcal{S}$, das *Superuniversum*, zusammen mit einer Interpretation I der Funktionssymbole. Für jedes Funktionssymbol f ist damit eine Abbildung $I(f) : \mathcal{S}^{\mathsf{ord}(f)} \to \mathcal{S}$ gegeben. Damit ist auch hier $\mathcal{I}$ eine Auswertung von Termen.
Um im weiteren auch partielle Funktionen (etwa Division auf $\mathbb{N}$) darstellen zu können, enthält Σ immer eine Konstante undef und $\mathcal{S}$ entsprechend das Element *undef*. Für jedes Funktionssymbol ist der *Definitionsbereich* (*domain*) als die Menge

$$\mathsf{dom}(f) := \{(a_1, \ldots, a_{\mathsf{ord}(f)}) \in \mathcal{S}^{\mathsf{ord}(f)} : I(f)(a_1, \ldots, a_{\mathsf{ord}(f)}) \neq \mathit{undef}\}$$

definiert. Als weiteres enthält Σ die Konstanten true und false und $\mathcal{S}$ entsprechend die Elemente *true* und *false*.
Die Darstellung von verschiedenen Sorten wird durch Funktionen U mit $I(U) : \mathcal{S} \to \{\mathit{true}, \mathit{false}\}$ (vgl. charakteristische Funktion einer Menge) ermöglicht: Die Menge $\mathcal{U} := \{a \in \mathcal{S} : (I(U))(a) = \mathit{true}\}$ wird als ein *Universum* bezeichnet.
Jede statische Algebra umfaßt ein ausgezeichnetes Universum $\mathsf{Bool} := \{\mathit{true}, \mathit{false}\}$, die üblichen Booleschen Operatoren als Funktionen, sowie die Gleichheitsrelation.
Diese ist die einzige Relation in Evolving Algebras. Durch die Übersetzung

$$\mathcal{I}(a = b) = \mathit{true} \;\Leftrightarrow\; (\mathcal{I}(a), \mathcal{I}(b)) \in I(=) \;\Leftrightarrow\; \mathcal{I}(a) = \mathcal{I}(b)$$

kann auch „=“ als Funktion – und damit der Ausdruck $a = b$ als Term – aufgefaßt werden. Weitergehend kann man sich überlegen, daß jede Relation durch ihre charakteristische Funktion repräsentierbar ist.

Eine *Evolving Algebra* $\mathcal{A}$ besteht aus statischen Algebren. Diese werden auch als (Berechnungs)Zustände von $\mathcal{A}$ bezeichnet. Die Signatur und das Superuniversum sind dabei in allen Zuständen gleich. Damit

kann man von einer Signatur $\Sigma(\mathcal{A})$ und einem Superuniversum $\mathcal{S}(\mathcal{A})$ sprechen.

Die Funktionssymbole werden i.a. in jeder einzelnen statischen Algebra unterschiedlich interpretiert. Daher unterscheidet man zustandsabhängig und zustandsunabhängig interpretierte Funktionen. Die Menge der zustandsunabhängig interpretierten Funktionssymbole aus $\Sigma(\mathcal{A})$ wird mit $\Sigma^c(\mathcal{A})$ bezeichnet.

Um die schrittweise Entfaltung einer Evolving Algebra darzustellen, wird ein Startzustand $\mathcal{Z}(\mathcal{A})$ durch die initialen Werte der zustandsabhängig interpretierten Funktionen beschrieben, sowie eine Menge von *Übergangsregeln*, die die Veränderungen der zustandsabhängig interpretierten Funktionen in einem Schritt beschreiben, in einer festgelegten Pascal-ähnlichen Syntax angegeben.

Die einfachste Form einer Übergangsregel ist eine Veränderung (*update*) u einer Funktion an einer Stelle:

$$u: \quad f(t_1, \ldots, t_n) := t_0$$

für eine n-stellige Funktion f und geschlossene Terme t_i.

Auf dieser Basis lassen sich auch bedingte Veränderungen (*guarded updates*) definieren: Sei b ein Term und u eine Veränderung wie oben beschrieben. Dann ist

$$\mathsf{if}\ b\ \mathsf{then}\ u\ \mathsf{endif}$$

folgendermaßen zu interpretieren: Ist – in der statischen Algebra, in der man sich gerade befindet – $\mathcal{I}(b) = true$, so soll u ausgeführt werden, ansonsten geschieht keine Veränderung.

Die komplizierteste – und auch die allgemeinste – Form einer Übergangsregel ist für $k \in \mathbf{N}$

$$\begin{array}{l}\mathsf{if}\ b_0\ \mathsf{then}\ u_0\\ \mathsf{elsif}\ b_1\ \mathsf{then}\ u_1\\ \vdots\\ \mathsf{elsif}\ b_k\ \mathsf{then}\ u_k\\ \mathsf{else}\ u_{k+1}\\ \mathsf{endif}\quad .\end{array}$$

Ein *Programm* einer Evolving Algebra ist eine Menge von Übergangsregeln.

Bezüglich der Behandlung von Parallelität in der Ausführung von Regeln – falls mehrere Regeln in einem Zustand anwendbar sind – werden in der aufgeführten Literatur unterschiedliche Annahmen gemacht. In dieser Arbeit wird vereinbart, daß beliebig viele anwendbare (allerdings mindestens eine) Regeln gleichzeitig ausgeführt werden können, soweit sie sich nicht widersprechen.

Damit stellt eine Evolving Algebra ein Transitionssystem, gegeben durch ein Tupel

$$\mathcal{A} = (\mathcal{Z}(\mathcal{A}), \mathcal{R}(\mathcal{A}))$$

dar, wobei $\mathcal{Z}(\mathcal{A})$ der Startzustand und $\mathcal{R}(\mathcal{A})$ die Menge der Übergangsregeln ist.

Die durch Anwendung einer Übergangsregel $r \in \mathcal{R}(\mathcal{A})$ auf eine statische Algebra g entstehende statische Algebra wird mit $r(g)$ bezeichnet. Werden in einem Schritt mehrere Regeln $r_1, r_2, \ldots, r_n$ parallel angewandt, so wird für $\vec{r} = (r_1, r_2, \ldots, r_n) \in \mathcal{R}(\mathcal{A})^*$ die durch Anwendung dieser Übergangsregeln auf g

entstehende statische Algebra mit $\vec{r}(g)$ bezeichnet.

Definition 5 *Eine Folge* $p = (g_0 = \mathcal{Z}(\mathcal{A}), g_1, g_2, \ldots)$ *ist eine* Berechnungsfolge *in* $\mathcal{A}$*, wenn es für jedes* n *eine endliche Menge* $\{r_1, \ldots, r_k\} \subseteq \mathcal{R}(\mathcal{A})$ *gibt mit* $g_n = (r_1, \ldots, r_k)(g_{n-1})$.

Definition 6 *Die Menge* $\mathcal{G}(\mathcal{A})$ *aller in* $\mathcal{A}$ *von* $\mathcal{Z}(\mathcal{A})$ *erreichbaren statischen Algebren erhält man als*

$$\begin{aligned}
&\mathcal{G}^0_{\mathcal{A}} := \{\mathcal{Z}(\mathcal{A})\} \\
&\mathcal{G}^{n+1}_{\mathcal{A}} := \{\vec{r}(g) : \vec{r} \in \mathcal{R}(\mathcal{A})^*, g \in \mathcal{G}^n_{\mathcal{A}}\} \\
&\mathcal{G}(\mathcal{A}) := \textstyle\bigcup_{n\geq 0} \mathcal{G}^n_{\mathcal{A}} \quad .
\end{aligned}$$

1.5 Kripke-Strukturen

Definition 7 *Eine prädikatenlogische* Kripke-Struktur *ist ein Tripel*

$$\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M}) \quad ,$$

wobei $\boldsymbol{G}$ *eine Menge von Elementen (interpretierbar z.B. als Zustände oder mögliche Welten) ist, zwischen denen* $\boldsymbol{R} \subseteq \boldsymbol{G} \times \boldsymbol{G}$ *eine* Zugänglichkeitsrelation *beschreibt. Jedem* $g \in \boldsymbol{G}$ *ist eine gewöhnliche prädikatenlogische Struktur* $\boldsymbol{M}(g) = (M(g), \boldsymbol{U}(g))$ *zu einer Signatur* $\Sigma(g)$ *und Universum* $\boldsymbol{U}(g)$ *zugeordnet.*

Analog erhält man eine aussagenlogische Kripke-Struktur, wenn die einzelnen $g \in \boldsymbol{G}$ aussagenlogische Strukturen sind.

Definition 8 *Zu der Zugänglichkeitsrelation* $\boldsymbol{R}$ *definiert man deren transitive bzw. reflexiv-transitive Hülle als*

$$\begin{aligned}
\boldsymbol{R}^+(a,b) \quad &\Leftrightarrow \quad \exists n > 0,\ a = g_0,\ g_1, \ldots,\ g_n = b : \forall i < n : \boldsymbol{R}\,(g_i, g_{i+1}) \\
\textit{und} \quad \boldsymbol{R}^*(a,b) \quad &\Leftrightarrow \quad \exists n \geq 0,\ a = g_0,\ g_1, \ldots,\ g_n = b : \forall i < n : \boldsymbol{R}\,(g_i, g_{i+1}) \quad .
\end{aligned}$$

Definition 9 *Ein* Pfad p *in einer Kripke-Struktur* $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ *ist eine Folge* $p = (g_0, g_1, g_2, \ldots)$, $g_i \in \boldsymbol{G}$ *wobei für alle* n $\boldsymbol{R}\,(g_n, g_{n+1})$ *gilt.*
Entsprechend Abschnitt 1.2 kann somit p *als* $p \in \boldsymbol{G}^{\mathbf{N}} : \ \mathbf{N} \to \boldsymbol{G} : n \mapsto g_n$ *aufgefaßt werden. Die Menge der Pfade in einer Kripke-Struktur* $\boldsymbol{K}$ *wird mit* $\boldsymbol{P}(\boldsymbol{K})$ *bezeichnet.*

Definition 10 *Für einen Pfad* $p = (g_0, g_1, g_2, \ldots)$ *ist* $p{\downarrow_i} := g_i$ *und es bezeichnet* $p|_i$ *das bei* g_i *beginnende Endstück* $(g_i, g_{i+1}, \ldots)$ *von* p*, das nach Definition für sich gesehen wieder ein Pfad ist.*

Da ein Pfad einen Zustand einer Kripke-Struktur mehrfach durchlaufen kann, ist die Abbildung $\downarrow$ i.a. nicht injektiv. Aus diesem Grund muß auch als Argument der Abbildung $|$ eine natürliche Zahl angegeben werden, und nicht etwa ein Zustand.

Definition 11 *Zwei Pfade* $p = (g_0, g_1, g_2, \ldots)$ *und* $p' = (g'_0, g'_1, g'_2, \ldots)$ *heißen* parallel *zwischen zwei Zuständen* h *und* h'*, wenn* p *und* q *dieselbe Sequenz* $h, \ldots, h'$ *enthalten, d.h. es gibt* i, j, k *mit* $g_i = g'_j = h$ *und* $g_{i+k} = g'_{j+k} = h'$ *und für alle* $l : 0 < l < k$ *gilt* $g_{i+l} = g'_{j+l}$*.*

Die konkrete Formelsyntax wird erst in den folgenden Abschnitten definiert, so daß an dieser Stelle noch abstrakt von Formeln die Rede ist.

Die folgenden Definitionen werden nur für variablenfreie Formeln gegeben, da der Begriff einer Variablenbelegung im Kontext nichtkonstanter Universen problematisch ist.

Definition 12 *Für Kripke-Strukturen* $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ *wird eine Wahrheitsrelation* $\models = \models_{\mathrm{Kr}}$ *zwischen Zuständen* $g \in \boldsymbol{G}$ *und variablenfreien Formeln der jeweiligen noch festzulegenden Sprache definiert mit der Bedeutung*

$$g \models F \quad \Leftrightarrow \quad \text{„}F \text{ ist wahr im Zustand } g\text{“}.$$

Definition 13 *Eine Formel* F *ohne freie Variablen heißt* gültig in einer Kripke-Struktur $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$*, in Zeichen* $\boldsymbol{K} \models F$*, falls* $g \models F$ *für alle* $g \in \boldsymbol{G}$ *gilt.*

Definition 14 *Eine Formel* F *ohne freie Variablen heißt* allgemeingültig *bzw. eine* Tautologie*, falls für alle Kripke-Strukturen* $\boldsymbol{K}$ *mit entsprechendem Vokabular* $\boldsymbol{K} \models F$ *gilt.*

Definition 15 *Eine Formel* F *ohne freie Variablen heißt* erfüllbar*, falls es eine Kripke-Struktur* $\boldsymbol{K}$ *mit* $\boldsymbol{K} \models F$ *gibt.*

Die zustandsunabhängigen Anteile einer Kripke-Struktur werden speziell betrachtet:

Definition 16 *Eine Kripke-Struktur besitzt ein konstantes Universum, mit* $\boldsymbol{U}(\boldsymbol{K})$ *bezeichnet, falls für alle* $g, h \in \boldsymbol{G}$ *gilt* $\boldsymbol{U}(g) = \boldsymbol{U}(h)$*.*

Im Verlauf dieser Arbeit werden nur Kripke-Strukturen mit konstantem Universum betrachtet.

Für diese wird der Begriff einer Variablenbelegung wie in der Prädikatenlogik definiert:

Definition 17 *Eine* Variablenbelegung *zu einer Kripke-Struktur* $\boldsymbol{K}$ *mit* konstantem *Universum* $\boldsymbol{U}(\boldsymbol{K})$ *ist eine Funktion*

$$\chi : \mathsf{Var} \to \boldsymbol{U}(\boldsymbol{K}) \quad .$$

Die Auswertung von Termen unter einer Variablenbelegung χ geschieht wie im prädikatenlogischen Fall: Da Terme bei der vorliegenden Definition von $\models$ nie global bezüglich der Kripke-Struktur $\boldsymbol{K}$, sondern nur lokal innerhalb von PL1-Strukturen $\boldsymbol{M}(g)$ ausgewertet werden müssen, kann man innerhalb dieser die entsprechende prädikatenlogische Auswertung $(\boldsymbol{M}(g))(f(t_1,\ldots,t_n),\chi)$ verwenden.

Wie bereits in Abschnitt 1.4 gibt es ebenfalls zustandsunabhängig interpretierte Funktionssymbole (und im allgemeinen Fall auch Prädikatssymbole):

Definition 18 *Die Menge der in einer Kripke-Struktur* $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ *zustandsunabhängig interpretierten Funktions- und Prädikatssymbole wird mit*

$$\begin{aligned}\Sigma^c(\boldsymbol{K}) := \quad & \{f\text{: für alle } g,h \in \boldsymbol{G} \text{ ist } f \in \Sigma(g) \text{ und es gilt } (M(g))(f) = (M(h))(f)\} \cup \\ & \{p\text{: für alle } g,h \in \boldsymbol{G} \text{ ist } p \in \Sigma(g) \text{ und es gilt } (M(g))(p) = (M(h))(p)\}\end{aligned}$$

bezeichnet.

Damit induziert dieser zustandsunabhängige Anteil einen einer prädikatenlogischen Interpretation ähnlichen Begriff:

$$\boldsymbol{K}, \Sigma^c(\boldsymbol{K}) \rightsquigarrow (K, \boldsymbol{U}(\boldsymbol{K})) \quad ,$$

wobei K eine Abbildung der Signatursymbole aus $\Sigma^c(\boldsymbol{K})$ mit denselben Eigenschaften wie auf Seite 9 ist und folgendermaßen definiert wird:
Sei $g \in \boldsymbol{G}$ beliebig gewählt.
Dann ist

$$\begin{aligned}\text{für jedes Funktionssymbol } f \in \Sigma^c(\boldsymbol{K}): \quad & K(f) := (M(g))(f) \\ \text{und für jedes Prädikatssymbol } p \in \Sigma^c(\boldsymbol{K}): \quad & K(p) := (M(g))(p) \quad .\end{aligned}$$

Definition 19 *Die mit einer Kripke-Struktur* $\boldsymbol{K}$ *assoziierte – ebenfalls mit* $\boldsymbol{K}$ *bezeichnete – Auswertung*

$$\boldsymbol{K} : \mathsf{Term}_{\Sigma^c(\boldsymbol{K})} \times \Xi \to \boldsymbol{U}(\boldsymbol{K})$$

von Termen (und Tupeln von Termen), die nur zustandsunabhängig interpretierte Funktionen enthalten, ist folgendermaßen definiert: Sei $\chi \in \Xi$ *eine Variablenbelegung.*

$$\begin{aligned}&\boldsymbol{K}(x,\chi) := \chi(x) \quad \textit{für eine Variable } x, \\ &\boldsymbol{K}((t_1,\ldots,t_n),\chi) := (\boldsymbol{K}(t_1,\chi),\ldots,\boldsymbol{K}(t_n,\chi)) \quad \textit{für } t_1,\ldots,t_n \in \mathsf{Term}_{\Sigma^c(\boldsymbol{K})}. \\ &\boldsymbol{K}(f(t_1,\ldots,t_n),\chi) := (K(f))(\boldsymbol{K}((t_1,\ldots,t_n),\chi)) = (K(f))(\boldsymbol{K}(t_1,\chi),\ldots,\boldsymbol{K}(t_n,\chi)) \\ &\qquad \textit{für ein Funktionssymbol } f \in \Sigma^c(\boldsymbol{K}),\ \mathsf{ord}(f) = n \textit{ und } t_1,\ldots,t_n \in \mathsf{Term}_{\Sigma^c(\boldsymbol{K})}.\end{aligned}$$

1.6 Modal- und Temporallogik

In den folgenden Abschnitten wird die Entwicklung der in dieser Arbeit verwendeten Temporallogik beschrieben sowie diese formal definiert.

Durch die historische Entwicklung der Modallogik bedingt entstanden die beiden Modaloperatoren $\Box$ ($\Box F$: F ist notwendigerweise wahr, bzw. F gilt in allen vorstellbaren Welten) und $\Diamond$ ($\Diamond F$: F ist möglicherweise wahr, bzw. es ist eine Welt vorstellbar, in der F gilt).
In der Umsetzung für modale Logik der Zeit, Temporallogik, ergab dies die Semantik $\Box F$: "always": F gilt in allen Folgezuständen und $\Diamond F$: "sometimes": F gilt in einem Folgezustand.
Für die Modellierung einer Problemstellung als temporale Kripke-Struktur bieten sich zwei Alternativen: Eine Kripke-Struktur beschreibt genau eine lineare Zeitachse (*lineare Zeit*) oder einen Zeitbaum, d.h. viele voneinander abzweigende Zeitachsen (*verzweigende Zeit*). Die beiden Möglichkeiten unterscheiden sich nur in den Anforderungen an die Zugänglichkeitsrelation $\boldsymbol{R}$.
Dabei ergibt sich entsprechend eine unterschiedliche Semantik der Modaloperatoren und eine unterschiedliche Ausdruckskraft der Logiken (vgl. [La80]: "sometimes" is sometimes "not never" und [EH83]):
Für lineare Zeit bedeutet die Gültigkeit von $\Diamond F$, daß F auf *jeder* Zeitachse erreichbar ist ("sometimes"), während dieselbe Formel für verzweigende Zeit „es gibt eine Zeitachse, auf der F erreichbar ist" bedeutet ("not never"). Derselbe Sachverhalt ist in der jeweils anderen Logik nicht formulierbar.

Um die Ausdruckskraft beider Ansätze zu kombinieren wird in [BMP81] die Logik UB (*unified branching time*) eingeführt, die das Problem durch zwei Pfadquantoren A und E löst. Damit unterscheidet sich diese Temporallogik grundsätzlich von den historischen Modallogiken. Zusätzlich wird der weitere Modaloperator $\circ$ ("nexttime") eingeführt, so daß sich in UB sechs verschiedene Modalitäten ergeben: $\mathsf{A}\Box$, $\mathsf{E}\Box$, $\mathsf{A}\Diamond$, $\mathsf{E}\Diamond$, $\mathsf{A}\circ$ und $\mathsf{E}\circ$. Als weitere Ergänzung wird der until-Operator hinzugefügt, womit man die in [CE81] eingeführte Logik CTL (*computation tree logic*) erhält.

In der Literatur werden temporale Logiken in den meisten Fällen – insbesondere in den zitierten grundlegenden Arbeiten – nur in aussagenlogischen Versionen vorgestellt. Da im weiteren Verlauf der Betrachtungen jedoch Prädikatenlogik benötigt wird, werden hier bereits prädikatenlogische Versionen definiert. Die entsprechenden Kripke-Strukturen $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ müssen zum korrekten „Verständnis" der Quantoren die Bedingung

$$\text{für alle } g, h \in \boldsymbol{G}: \ \boldsymbol{R}(g,h) \ \Rightarrow \ \boldsymbol{U}(g) \subseteq \boldsymbol{U}(h)$$

erfüllen.

Durch den Übergang von Prädikatenlogik zu Temporallogik – bzw. von einer prädikatenlogischen Struktur zu einer Kripke-Struktur, die aus vielen einzelnen prädikatenlogischen Strukturen besteht – erhält man zwei grundsätzlich verschiedene Klassen von Formeln:

Zustandsformeln:	Sie gelten *in* einem Zustand.
	Zu ihnen gehören alle rein aussagenlogischen Formeln.
Pfadformeln:	Sie gelten für einen Pfad, d.h. eine Folge von Zuständen.

Mit den einzelnen Logiken werden beide Klassen formal definiert.

1.7 Die Logiken UB und CTL

Die in [CE81] eingeführte Logik CTL (*computation tree logic*) ist eine Aussagenlogik zur Beschreibung verzweigender Zeit.
Sie basiert auf der einfachsten temporalen Logik für lineare und verzweigende Zeit, UB [BMP81]. Diese enthält nur die drei einstelligen Modaloperatoren $\Box$ ("always"), $\Diamond$ ("sometimes") und $\circ$ ("nexttime"). CTL umfaßt als weiteres noch den until-Operator. Durch Ergänzung um prädikatenlogische Quantoren sowie prädikatenlogischer anstelle aussagenlogischer Atome erhält man prädikatenlogisches CTL.

Syntax:

Um den temporalen Aspekt auszudrücken, werden zu den Symbolen der Prädikatenlogik für CTL die folgenden Symbole hinzugenommen:

(temporallogische) Junktoren:	$\circ$, $\Box$, $\Diamond$, until
Pfadquantoren:	E, A

Dabei lassen sich die folgenden Symbole durch until und E definieren:

$$\Diamond P := \mathsf{true}\ \mathsf{until}\ P \quad , \quad \Box P := \neg\Diamond\neg P \quad , \quad \mathsf{A}P := \neg\mathsf{E}\neg P \quad .$$

Die Menge $\mathcal{L}(\mathrm{CTL})$ der CTL-Formeln wird mit diesen Symbolen wie folgt rekursiv definiert:

(1) Jede PL1-Formel ist eine CTL-Formel.

(2) Sind F und G CTL-Formeln, so sind auch $\neg F$, $F \wedge G$ und $F \vee G$ CTL-Formeln.

(3) Sind F und G CTL-Formeln, so sind auch $\mathsf{A}\circ F$, $\mathsf{A}\neg\circ F$, $\mathsf{E}\circ F$, $\mathsf{E}\neg\circ F$, $\mathsf{A}\Box F$, $\mathsf{A}\neg\Box F$, $\mathsf{E}\Box F$, $\mathsf{E}\neg\Box F$, $\mathsf{A}\Diamond F$, $\mathsf{A}\neg\Diamond F$, $\mathsf{E}\Diamond F$, $\mathsf{E}\neg\Diamond F$, $\mathsf{A}(F\ \mathsf{until}\ G)$, $\mathsf{A}\neg(F\ \mathsf{until}\ G)$, $\mathsf{E}(F\ \mathsf{until}\ G)$ und $\mathsf{E}\neg(F\ \mathsf{until}\ G)$ CTL-Formeln.

(4) Ist F eine CTL-Formel und x eine Variable, so sind auch $\forall x : F$ und $\exists x : F$ CTL-Formeln.

An dieser Definition sieht man, daß in CTL unmittelbar vor *jedem* (eventuell negierten) Modaloperator ein Pfadquantor steht.

In Termini der o.g. Formelklassen erhält man die folgende Definition:

(ZA) Jede PL1-Formel ist eine CTL-Zustandsformel.

(Z1) Sind F und G CTL-Zustandsformeln, so sind auch $\neg F$, $F \wedge G$ und $F \vee G$ CTL-Zustandsformeln.

(P1) Sind F und G CTL-Zustandsformeln, so sind $\circ F$, $\Box F$, $\Diamond F$ und $(F\ \mathsf{until}\ G)$ CTL-Pfadformeln.

(P2) Ist P eine CTL-Pfadformel, so ist $\neg P$ eine CTL-Pfadformel.

(Z2) Ist P eine CTL-Pfadformel, so sind $\mathsf{A}P$ und $\mathsf{E}P$ CTL-Zustandsformeln.

(ZQ) Ist F eine CTL-Zustandsformel und x eine Variable, so sind auch $\forall x : F$ und $\exists x : F$ CTL-Zustandsformeln.

(F) Jede CTL-Zustandsformel ist eine CTL-Formel.

Semantik:

In Worten formuliert haben die gegenüber der Prädikatenlogik neuen Symbole die folgende Bedeutung:

$\circ$ ("Nexttime"):

$\mathsf{A}\circ F$: Auf allen Pfaden gilt im nächsten Zustand F.

$\mathsf{E}\circ F$: Es gibt einen Pfad, auf dem im nächsten Zustand F gilt.

$\Diamond$ ("Sometimes"):

$\mathsf{A}\Diamond F$: $\mathsf{A}(\mathsf{true\ until}\ F)$: Auf allen Pfaden gilt F in der Zukunft irgendwann einmal.

$\mathsf{E}\Diamond F$: $\mathsf{E}(\mathsf{true\ until}\ F)$: Es gibt einen Pfad, auf dem F in der Zukunft einmal gilt.

$\Box$ ("Always"):

$\mathsf{A}\Box F$: $\neg\mathsf{E}\Diamond(\neg F)$: Auf jedem Pfad gilt F in allen Folgezuständen.

$\mathsf{E}\Box F$: $\neg\mathsf{A}\Diamond(\neg F)$: Es gibt einen Pfad, auf dem F in allen Folgezuständen gilt.

until:

$\mathsf{A}(F\ \mathsf{until}\ G)$: Auf allen Pfaden gibt es einen Folgezustand, in dem G gilt, und bis unmittelbar vor diesem Zustand gilt F.

$\mathsf{E}(F\ \mathsf{until}\ G)$: Es gibt einen Pfad, auf dem es einen Folgezustand gibt, in dem G gilt, und bis unmittelbar vor diesem Zustand gilt F.

Formal wird die Wahrheitsrelation $\models:=\models_{\mathrm{TL}}$ entsprechend der Syntaxdefinition rekursiv bezüglich einer prädikatenlogischen Kripke-Struktur $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ unter Rückgriff auf die prädikatenlogische Wahrheitsrelation $\models_{\mathrm{PL}}$ wie folgt definiert:
Sei A eine atomare PL1-Formel, F und G CTL-Formeln und χ eine Variablenbelegung. Dann gilt für $g \in \boldsymbol{G}$:

(1) $(g,\chi) \models A$ $:\Leftrightarrow$ $(\boldsymbol{M}(g),\chi) \models_{\mathrm{PL}} A$.

(2a) $(g,\chi) \models \neg F$ $:\Leftrightarrow$ nicht $(g,\chi) \models F$.

(2b) $(g,\chi) \models F \wedge G$ $:\Leftrightarrow$ $(g,\chi) \models F$ und $(g,\chi) \models G$.

(3a) $(g,\chi) \models \mathsf{A}\circ F$ $:\Leftrightarrow$ für alle $h \in \boldsymbol{G}$ mit $(g,h) \in \boldsymbol{R}$ gilt $(h,\chi) \models F$.

(3b) $(g,\chi) \models \mathsf{E}\circ F$ $:\Leftrightarrow$ es gibt ein $h \in \boldsymbol{G}$ mit $(g,h) \in \boldsymbol{R}$, so daß $(h,\chi) \models F$ gilt.

(3c) $(g,\chi) \models \mathsf{A}(F\ \mathsf{until}\ G)$ $:\Leftrightarrow$ für alle Pfade $(g = g_0, g_1, \ldots)$ gibt es ein $i \geq 0$ mit $(g_i,\chi) \models G$ und für alle $j : 0 \leq j < i$ gilt $(g_i,\chi) \models F$.

(3d) $(g,\chi) \models \mathsf{E}(F\ \mathsf{until}\ G)$ $:\Leftrightarrow$ es gibt einen Pfad $(g = g_0, g_1, \ldots)$, für den es ein $i \geq 0$ gibt mit $(g_i,\chi) \models G$ und für alle $j : 0 \leq j < i$ gilt $(g_i,\chi) \models F$.

(4a) $(g,\chi) \models \forall x : F$ $:\Leftrightarrow$ für alle $d \in \boldsymbol{U}(g)$ gilt $(g,\chi^d_x) \models F$.

(4b) $(g,\chi) \models \exists x : F$ $:\Leftrightarrow$ es gibt ein $d \in \boldsymbol{U}(g)$ mit $(g,\chi^d_x) \models F$.

Entsprechend der zweiten Syntaxdefinition kann man auch die $\models$-Relation für Zustandsformeln und Pfadformeln getrennt definieren:
Sei $g \in \boldsymbol{G}$ ein Zustand, $p = (g_0, g_1, \ldots)$ ein Pfad in $\boldsymbol{K}$, A eine atomare PL1-Formel, F und G CTL-Zustandsformeln, P eine CTL-Pfadformel und χ eine Variablenbelegung. Dann gilt:

(ZA)	$(g,\chi) \models A$	$:\Leftrightarrow$	$(M(g),\chi) \models_{\mathrm{PL}} A$.
(Z1a)	$(g,\chi) \models \neg F$	$:\Leftrightarrow$	nicht $(g,\chi) \models F$.
(Z1b)	$(g,\chi) \models F \wedge G$	$:\Leftrightarrow$	$(g,\chi) \models F$ und $(g,\chi) \models G$.
(Z1c)	$(g,\chi) \models F \vee G$	$:\Leftrightarrow$	$(g,\chi) \models F$ oder $(g,\chi) \models G$.
(P1a)	$(p,\chi) \models \circ F$	$:\Leftrightarrow$	$(g_1,\chi) \models F$.
(P1b)	$(p,\chi) \models \Box F$	$:\Leftrightarrow$	für alle i gilt $(g_i,\chi) \models F$.
(P1c)	$(p,\chi) \models \Diamond F$	$:\Leftrightarrow$	es gibt ein i mit $(g_i,\chi) \models F$.
(P1d)	$(p,\chi) \models F \,\mathsf{until}\, G$	$:\Leftrightarrow$	es gibt ein $i \geq 0$ mit $(g_i,\chi) \models G$ und für alle $j : 0 \leq j < i$ gilt $(g_i,\chi) \models F$.
(P2)	$(p,\chi) \models \neg P$	$:\Leftrightarrow$	nicht $(p,\chi) \models P$.
(Z2a)	$(g,\chi) \models \mathsf{A}P$	$:\Leftrightarrow$	für alle Pfade $p = (g_0, g_1, \ldots)$ in $\boldsymbol{K}$ und alle n mit $g_n = g$ gilt $(p\|_n,\chi) \models P$.
(Z2b)	$(g,\chi) \models \mathsf{E}P$	$:\Leftrightarrow$	es gibt einen Pfad $p = (g_0, g_1, \ldots)$ in $\boldsymbol{K}$ und ein n mit $g_n = g$ und $(p\|_n,\chi) \models P$.
(ZQa)	$(g,\chi) \models \forall x : F$	$:\Leftrightarrow$	für alle $d \in \boldsymbol{U}(g)$ gilt $(g,\chi^d_x) \models F$.
(ZQb)	$(g,\chi) \models \exists x : F$	$:\Leftrightarrow$	es gibt ein $d \in \boldsymbol{U}(g)$ mit $(g,\chi^d_x) \models F$.

Die Definitionen (1) bzw. (ZA) dienen dazu, den Modellbegriff von der Ebene der einzelnen PL1-Strukturen $\boldsymbol{M}(g)$ auf die Ebene der Zustände $g \in \boldsymbol{G}$ zu fortzusetzen.

Für die Modaloperatoren werden ebenfalls *strikte* Versionen, die sich nur auf die *echten* Folgezustände beziehen, folgendermaßen definiert:

Definition 20 *Sei p ein Pfad, F und G Zustandsformeln und $\mathsf{op} \in \{\Box, \Diamond, \mathsf{until}\}$ ein Modaloperator. Dann ist dessen* strikte *Version op^+ folgendermaßen definiert:*
Sei χ eine Variablenbelegung.

$$(p,\chi) \models \mathsf{op}^+ F \;:\Leftrightarrow\; (p|_1,\chi) \models \mathsf{op}\, F \quad ,$$
$$(p,\chi) \models F\, \mathsf{op}^+ G \;:\Leftrightarrow\; (p|_1,\chi) \models F\, \mathsf{op}\, G \quad .$$

1.8 Die Logik CTL$^+$

In CTL sind die folgenden zusammengesetzten Modaloperatoren nicht formulierbar:

$$\begin{aligned} F \text{ unless } G &:= (F \text{ until } G) \vee (\Box(F \wedge \neg G)) \\ F \text{ before } G &:= \neg G \text{ until } (F \wedge \neg G) \\ F \text{ atnext } G &:= (\neg G \text{ until } (F \wedge G)) \vee \Box\neg G \end{aligned}$$

Daher wird die Syntax zu CTL$^+$ [EH83] erweitert, indem logische Verknüpfungen auch für Pfadformeln zugelassen werden (P3$^+$):

(ZA$^+$) Jedes Atom ist eine CTL$^+$-Zustandsformel.

(Z1$^+$) Sind F und G CTL$^+$-Zustandsformeln, so sind auch $\neg F$, $F \wedge G$ und $F \vee G$ CTL$^+$-Zustandsformeln.

(P1$^+$) Sind F und G CTL$^+$-Zustandsformeln, so sind $\circ F$, $\Box F$, $\Diamond F$ und $(F \text{ until } G)$ CTL$^+$-Pfadformeln.

(P2$^+$) Ist F eine CTL$^+$-Pfadformel, so ist $\neg P$ eine CTL$^+$-Pfadformel.

(P3$^+$) Sind P und Q CTL$^+$-Pfadformeln, so sind $P \wedge Q$ und $P \vee Q$ CTL$^+$-Pfadformeln.

(Z2$^+$) Ist P eine CTL$^+$-Pfadformel, so sind $\mathsf{A}P$ und $\mathsf{E}P$ CTL$^+$-Zustandsformeln.

(ZQ$^+$) Ist F eine CTL$^+$-Zustandsformel und x eine Variable, so sind auch $\forall x : F$ und $\exists x : F$ CTL$^+$-Zustandsformeln.

(F$^+$) Jede CTL$^+$-Zustandsformel ist eine CTL$^+$-Formel.

Die Semantik der neuen Formeln wird wie folgt definiert: Sei $p = (g_0, g_1, \ldots)$ ein Pfad in $\boldsymbol{K}$, P und Q CTL$^+$-Pfadformeln.

$$\begin{aligned} &\text{(P3}^+\text{a)} \quad (p,\chi) \models P \wedge Q \quad :\Leftrightarrow \quad (p,\chi) \models P \text{ und } (p,\chi) \models Q. \\ &\text{(P3}^+\text{b)} \quad (p,\chi) \models P \vee Q \quad :\Leftrightarrow \quad (p,\chi) \models P \text{ oder } (p,\chi) \models Q. \end{aligned}$$

Da der unless-Operator in dieser Arbeit eine wichtige Rolle spielt, wird an dieser Stelle seine Semantik angegeben: Sei χ eine Variablenbelegung.

$(g_0,\chi) \models \mathsf{A}(F \text{ unless } G) \quad :\Leftrightarrow$ Für alle Pfade $(g_0, g_1, \ldots)$ gilt:
es gibt ein $i \geq 0$ mit $(g_i,\chi) \models G$ und für alle $j : 0 \leq j < i$ gilt $(g_j,\chi) \models F$,
oder es gibt kein $i \geq 0$ mit $(g_i,\chi) \models G$ und für alle $j \geq 0$ gilt $(g_j,\chi) \models F$.

$(g_0,\chi) \models \mathsf{E}(F \text{ unless } G) \quad :\Leftrightarrow$ Es gibt einen Pfad $(g_0, g_1, \ldots)$, für den es entweder
ein $i \geq 0$ mit $(g_i,\chi) \models G$ und für alle $j : 0 \leq j < i$ gilt $(g_j,\chi) \models F$ gibt,
oder es gibt kein $i \geq 0$ mit $(g_i,\chi) \models G$ und für alle $j \geq 0$ gilt $(g_j,\chi) \models F$.

In [EH82] und [EH83] wurde festgestellt, daß CTL und CTL$^+$ dieselbe Ausdruckskraft besitzen, sowie ein Algorithmus zur Umformung von CTL$^+$-Formeln in CTL-Formeln angegeben.

1.9 Die Logik CTL*

Die volle Ausdruckskraft für verzweigende Zeit bietet erst die Logik CTL* [EH83], die zusätzlich geschachtelte Modaloperatoren (P4*) und weitere Kombinationen von prädikatenlogischen Quantoren und Pfadquantoren (PQ*) zuläßt (neu ist außerdem (PZ*)):

(ZA*) Jedes Atom ist eine CTL*-Zustandsformel.

(PZ*) Jede rein prädikatenlogische Formel ist eine CTL*-Pfadformel[1].

(Z1*) Sind F und G CTL*-Zustandsformeln, so sind auch $\neg F$ und $F \wedge G$ und $F \vee G$ CTL*-Zustandsformeln.

(P1*) Sind F und G CTL*-Zustandsformeln, so sind $\circ F$, $\Box F$, $\Diamond F$ und $(F \;\mathsf{until}\; G)$ CTL*-Pfadformeln.

(P2*) Ist P eine CTL*-Pfadformel, so ist $\neg P$ eine CTL*-Pfadformel.

(P3*) Sind P und Q CTL*-Pfadformeln, so sind $P \wedge Q$ und $P \vee Q$ CTL*-Pfadformeln.

(P4*) Sind P und Q CTL*-Pfadformeln, so sind $\circ P$, $\Box P$, $\Diamond P$ und $(P \;\mathsf{until}\; Q)$ CTL*-Pfadformeln.

(Z2*) Ist P eine CTL*-Pfadformel, so sind $\mathsf{A}P$ und $\mathsf{E}P$ CTL*-Zustandsformeln.

(ZQ*) Ist F eine CTL*-Zustandsformel und x eine Variable, so sind auch $\forall x : F$ und $\exists x : F$ CTL*-Zustandsformeln.

(PQ*) Ist P eine CTL*-Pfadformel und x eine Variable, so sind auch $\forall x : P$ und $\exists x : P$ CTL*-Pfadformeln.

(F*) Jede CTL*-Zustandsformel ist eine CTL*-Formel.

Die entsprechende Semantik ist folgendermaßen gegeben:
Sei $p = (g_0, g_1, \ldots)$ ein Pfad in $\boldsymbol{K}$, F eine CTL$^+$-Zustandsformel, P und Q CTL-Pfadformeln und χ eine Variablenbelegung.

(PZ*)	$(p, \chi) \models F$	$:\Leftrightarrow$	$(p{\downarrow_0}, \chi) \models F \quad \Leftrightarrow \quad (g_0, \chi) \models F.$
(P4*a)	$(p, \chi) \models \circ P$	$:\Leftrightarrow$	$(p\vert_1, \chi) \models P.$
(P4*b)	$(p, \chi) \models \Box P$	$:\Leftrightarrow$	für alle i : $(p\vert_i, \chi) \models P.$
(P4*c)	$(p, \chi) \models \Diamond P$	$:\Leftrightarrow$	es gibt ein i : $(p\vert_i, \chi) \models P.$
(P4*d)	$(p, \chi) \models P \;\mathsf{until}\; Q$	$:\Leftrightarrow$	es gibt ein $i \geq 0$ mit $(p\vert_i, \chi) \models Q$ und für alle $j : 0 \leq j < i$ gilt $(p\vert_j, \chi) \models P.$
(PQ*a)	$(p, \chi) \models \forall x : P$	$:\Leftrightarrow$	für alle $d \in \boldsymbol{U}(\boldsymbol{K})$ gilt $(p, \chi_x^d) \models P.$
(PQ*b)	$(p, \chi) \models \exists x : P$	$:\Leftrightarrow$	es gibt ein $d \in \boldsymbol{U}(\boldsymbol{K})$ mit $(p, \chi_x^d) \models P.$

Die Regel (PZ*) beschreibt dabei, daß eine prädikatenlogische Formel auch als Pfadformel verstanden werden kann: Sie beschreibt eine Eigenschaft des ersten Zustandes auf dem Pfad.

---

[1] in [EH83] sind hier nur Atome zugelassen, was aber an der Gesamtsyntax nichts ändert.

1.10 Temporallogik linearer Zeit: PTL

Mit diesen Modaloperatoren – aber ohne die Pfadquantoren – ist analog eine Sprache PTL [Wol85, Pn77] für Temporallogik linearer Zeit definiert, die näher mit den historischen Modallogiken verwandt ist:

(L0) Jede PL1-Formel ist eine PTL-Formel.

(L1) Sind F und G PTL-Formeln, so sind auch $\neg F$, $F \wedge G$ und $F \vee G$ PTL-Formeln.

(L2) Sind F und G PTL-Formeln, so sind $\circ F$, $\Box F$, $\Diamond F$ und $(F \ \mathsf{until}\ G)$ PTL-Formeln.

(L3) Ist F eine PTL-Formel und x eine Variable, so sind $\forall x : F$ und $\exists x : F$ PTL-Formeln.

Alle in PTL formulierbaren einen (den dort einzigen) Pfad beschreibenden Aussagen lassen sich – sowohl universell als auch existentiell pfadquantifiziert – auch in CTL* formulieren:

Eine PTL-Formel F wird – je nachdem, was ausgedrückt werden soll – als $\mathsf{A}F$ oder $\mathsf{E}F$ zu einer CTL*-Formel.

1.11 Fairness

Bei der Beschreibung und Verifikation von nichtdeterministischen Prozessen spielen Fairnessannahmen eine bedeutende Rolle. In den meisten Fällen ist der Nachweis der Erreichbarkeit eines bestimmten Zustands (z.B. Terminierung) überhaupt nur unter solchen Annahmen möglich. Durch Fairnessannahmen werden Berechnungsfolgen von der Betrachtung ausgeschlossen, die als in der realen Ausführung unmöglich angenommen werden. Es werden verschiedene Arten von Fairness unterschieden [LPS81]:

Impartiality:
Impartiality fordert, daß eine bestimmte Aktion in jeder unendlichen Berechnungsfolge unendlich oft ausgeführt wird:

PTL: $\Box\Diamond$(Aktion wird ausgeführt)
CTL: $\mathsf{A}\Box(\mathsf{A}\Diamond$(Aktion wird ausgeführt))

Da diese Forderung offensichtlich an keinerlei Vorbedingungen – etwa die Ausführbarkeit der Aktion – gekoppelt ist, ist die Forderung im allgemeinen zu streng.

Justice (weak Fairness):
Justice beschreibt die Forderung, daß eine Aktion, die ab einem Zustand ununterbrochen möglich ist, auch irgendwann ausgeführt wird:

PTL: $(\Diamond\Box$(Aktion ausführbar)) $\rightarrow \Diamond$(Aktion wird ausgeführt)
CTL*: $\mathsf{A}((\Diamond\Box$(Aktion ausführbar)) $\rightarrow \Diamond$(Aktion wird ausgeführt))

Compassion (strong Fairness):
(Strenge) Fairness beschreibt die strengere Forderung, daß jede Aktion, die im weiteren Verlauf einer Berechnungsfolge unendlich oft ausführbar ist, auch (unendlich oft) ausgeführt wird:

PTL: $(\Box\Diamond(\text{Aktion ausführbar})) \rightarrow \Diamond(\text{Aktion wird ausgeführt})$
CTL*: $\mathsf{A}((\Box\Diamond(\text{Aktion ausführbar})) \rightarrow \Diamond(\text{Aktion wird ausgeführt}))$

Aus [La80], [EC80] und [EH83] folgt, daß Fairness nicht in CTL formuliert werden kann.

Es gilt Fairness $\Rightarrow$ Justice.

1.12 Problematik zustandsabhängig ausgewerteter Terme

Durch die zustandsabhängige Interpretation von Funktionen – und damit auch zustandsabhängige Auswertung von Termen – ergeben sich auch für zustandsunabhängig interpretierte Prädikate und Funktionen problematische Aspekte: Ein (Teil-)Term bezeichnet nicht in allen Zuständen dasselbe Element des Universums.

Soll ein Element des Universums, das sich in einem Zustand als Auswertung eines bestimmten Terms ergibt, festgehalten werden, und über *dieses Element* Aussagen in einem anderen Zustand gemacht werden, muß also ein dieses Element beschreibender Term erzeugt werden, der *zustandsunabhängig* interpretiert wird.

Diese Problematik tritt im folgenden primär an drei Stellen auf:

Gleichheitsschlüsse: Mit „Gleichheit“ ist normalerweise die Gleichheit zwischen Elementen des Universums gemeint. Syntaktisch stehen in Formeln jedoch Terme als Argumente. $g \models t = a$ und $h \models t = b$ lassen den Schluß $\boldsymbol{K} \models a = b$ nur dann zu, wenn a, b und t zustandsunabhängig interpretiert werden.

Invarianzen: Seien x und y zustandsabhängig und c zustandsunabhängig interpretiert. Die Formeln
$x = y \wedge \mathsf{A}\Box(x = c)$: Invarianz von x, keine Aussage über Verhalten von y und
$x = c \wedge \mathsf{A}\Box(x = y)$: Gleichheit zwischen x und y, welche Werte dabei angenommen werden, ist nicht gesagt,
beschreiben unterschiedliche Sachverhalte.

Abschlußsubstitutionen in Tableaux: Die durch die γ-Regeln eingeführten freien Variablen werden beim Abschluß eines Zweiges durch Terme substituiert. Die Substitution wird für das gesamte Tableau, also i.a. auch für Formeln, die sich auf andere Zustände beziehen, vorgenommen.
Dabei ist semantisch nicht der Term als syntaktisches Konstrukt, sondern das durch diesen Term beschriebene Element des Universums gemeint (vgl. Substitutionslemma, Seite 74).

Damit muß in diesen Fällen ein zustandsunabhängig interpretierter Term eingeführt werden, um dieses Element zu beschreiben. Dies geschieht durch eine Signaturerweiterung in Form einer Skolemkonstante oder -funktion oder eines abgeleiteten Bezeichners. Beide Varianten werden im folgenden vorgestellt.

2 Axiomatisierung einer Evolving Algebra als Kripke-Struktur

In diesem Kapitel wird der Begriff „Evolving Algebra“ in Beziehung zu dem der „Kripke-Struktur“ gesetzt und die Behandlung einiger Besonderheiten gegenüber allgemeinen prädikatenlogischen Kripke-Strukturen aufgezeigt. Darauf basierend wird eine Vorgehensweise zur Axiomatisierung vorgestellt und die Anforderungen an die zu verwendende Logik zusammengestellt. Die verwendeten Beispiele sind der Fallstudie „Alternating-Bit-Protokoll“ ([May95b]) entnommen.

2.1 Modellierung von Protokollen als Evolving Algebras

Die Signatur Σ der ein Protokoll modellierenden Evolving Algebra $\mathcal{A}$ wird aus den Bezeichnern der Datenstrukturen der Instanzen und den Konstruktoren und Modifikatoren dieser Datenstrukturen gebildet. Die Bezeichner werden dabei zustandsabhängig interpretiert, während die Operatoren zustandsunabhängig zu interpretieren sind.

Entsprechend umfaßt das Superuniversum neben den in Abschnitt 1.4 aufgeführten immer vorhandenen Elementen alle zur Definition dieser Datenstrukturen verwendeten Wertebereiche.

Der Startzustand $\mathcal{Z}(\mathcal{A})$ wird durch die Spezifikation der Initialisierung des Protokolls gegeben.

Entsprechend dem Ablauf des Protokolls wird für jede „Aktion“ auf dem gewählten Abstraktionsniveau eine Übergangsregel formuliert. Da dieser Ablauf durch das Verhalten – und die Algorithmen – der beteiligten *internen* Instanzen bestimmt wird, kann so jede Regel derjenigen Instanz, die die betreffende Aktion ausführt, zugeordnet werden. Daraus ergibt sich, daß jede Regel nur auf die „ihrer“ Instanz bekannten Datenstrukturen, d.h. Bezeichner der Evolving Algebra zugreift. Eine Ausnahme bilden Regeln, die ein i.a. unerwünschtes und nicht von Algorithmen kontrolliertes Verhalten modellieren, das von der Umgebung, durch *externe* Instanzen modelliert, ausgeht. Ein Beispiel dafür ist das Verhalten des Mediums, falls es z.B. Daten verfälscht oder verliert.

Eine Evolving Algebra zur *Beschreibung* eines Protokolls sollte nur Funktionen und Übergangsregeln enthalten, die ein Pendant in der Spezifikation des Verhaltens des Protokolls besitzen. Dabei ist insbesondere bei den Übergangsregeln zu beachten, daß sie sich an die Grenzen der entsprechenden Instanzen halten: Die Bedingungen der Übergangsregeln sollten nur Bezeichner enthalten, deren Pendant im Protokoll derjenigen Instanz, deren Verhalten die Regel beschreiben soll, auch zugänglich ist.

Aus der Spezifikation eines Protokolls (oder allgemeiner eines Algorithmus) mit den üblichen Spezifikationsmethoden lassen sich die Konstruktionselemente einer entsprechenden Evolving Algebra ablesen:

- Algebraische Spezifikation von Datentypen: Die Konstruktoren, Modifikatoren und Selektoren ergeben die zustandsunabhängig interpretierten Bezeichner.
- Z: Die Regelschemata werden in Übergangsregeln der Evolving Algebra umgesetzt.
- Zustandsübergangsdiagramme: Die einzelnen Zustände werden als statische Algebren codiert und die Übergangsregeln entsprechend transformiert.

Bei der Modellierung von parallelen Systemen auf höheren Abstraktionsebenen ist zu berücksichtigen, daß eventuell in einem Schritt von verschiedenen Instanzen Aktionen ausgeführt werden können, die konkurrierend verändernd auf die Datenstrukturen zugreifen (Bsp: Der Sender hängt ein Element an eine Schlange von Nachrichten an, während der Empfänger das erste Element dieser Schlange entnimmt). In diesem Fall ist es notwendig, in vollständiger Fallunterscheidung verschiedene Regeln zu definieren (Senden $\wedge$ $\neg$ Empfangen, Empfangen $\wedge$ $\neg$ Senden, Senden $\wedge$ Empfangen).

Zur *Verifikation* werden allerdings diese Regeln nicht ausreichen. Dies liegt in erster Linie daran, daß häufig Informationen über den Zustand der Evolving Algebra verwendet werden müssen, die die auf den direkten Ablauf des Protokolls keinen Einfluß haben und nicht explizit in deren aktuellem Datenbestand gespeichert sind, sowie, daß für einige Regeln – vornehmlich solche, die Datenverluste repräsentieren – nicht interessant ist, welche Daten sie verändern, sondern, was dabei in ihrer Umgebung unverändert bleibt. Dabei handelt es sich häufig um Eigenschaften, die Informationen mehrerer beteiligter Instanzen akkumulieren, also nicht mit einer einzelnen Instanz assoziierbar sind.

Aus diesem Grund geht man zu einer EEA (Erweiterte/Extended Evolving Algebra) und der entsprechenden Kripke-Struktur über, die neben den Funktionen der Evolving Algebra entsprechende zusätzliche Funktionen und erweiterte Übergangsregeln enthält.

Des weiteren sollten nach Möglichkeit die neuen Funktionen nur durch solche Regeln verändert werden, die von internen Instanzen ausgeführt werden. Damit beschreiben diese zwar nicht mehr das Wissen einer einzelnen Instanz, aber immer noch das kumulative Wissen der internen Instanzen.

Ein Beispiel einer solchen Kontrollinformation ist die Anzahl aller Nachrichten mit Kontrollbit 1 in einer Schlange von Nachrichten. Diese Information kann zu Induktionszwecken verwendet werden. Das kumulative Wissen von Sender und Empfänger liefert zumindest eine obere Schranke für diesen Wert.

2.2 Evolving Algebras als Kripke-Strukturen

Die ein Protokoll modellierende Evolving Algebra $\mathcal{A}$ kann als (evtl. unendliche) prädikatenlogische Kripke-Struktur $\boldsymbol{K}^{[\mathcal{A}]} = (\boldsymbol{G}^{[\mathcal{A}]}, \boldsymbol{R}^{[\mathcal{A}]}, \boldsymbol{M}^{[\mathcal{A}]})$ angesehen werden, wobei die statischen Algebren die einzelnen Welten der Kripke-Struktur darstellen.

Die Signatur der einzelnen Zustände $g \in \boldsymbol{G}^{[\mathcal{A}]}$ ist jeweils die Signatur $\Sigma(\mathcal{A})$ der Evolving Algebra. Damit hat man eine konstante Signatur $\Sigma(\boldsymbol{K}^{[\mathcal{A}]})$ (Dies ist nicht zu verwechseln mit der Menge $\Sigma^c(\boldsymbol{K}^{[\mathcal{A}]})$ der zustandsunabhängig interpretierten Funktionssymbole!).

Für die Universen $\boldsymbol{U}(g)$ der Zustände der Kripke-Struktur wird ebenfalls das Superuniversum $\mathcal{S}(\mathcal{A})$ übernommen, womit eine Kripke-Struktur mit konstantem Universum, d.h. für $g, h \in \boldsymbol{G}$ gilt $\boldsymbol{U}(g) = \boldsymbol{U}(h) = \boldsymbol{U}(\boldsymbol{K}^{[\mathcal{A}]})$, vorliegt. In dieser gilt die Barcansche Formel: $\boldsymbol{K}^{[\mathcal{A}]} \models (\forall x : \mathsf{A}\Box p(x)) \rightarrow \mathsf{A}\Box(\forall x : p(x))$.

In der Darstellung der Zustandsübergänge repräsentiert sich der Wechsel der Sichtweise von operationaler Semantik zu beschreibender Logik:

Während Übergänge in der Evolving Algebra als auf statische Algebren anwendbare Regeln eines Transitionssystems formuliert sind, äußern sie sich in der entsprechenden Kripke-Struktur als Einträge in der Zugänglichkeitsrelation. Damit kann man eine Evolving Algebra nun formal mit der für Kripke-Strukturen üblichen Terminologie beschreiben:

Die Evolving Algebra

$$\mathcal{A} = (\mathcal{Z}(\mathcal{A}), \mathcal{R}(\mathcal{A}))$$

mit dem Startzustand $\mathcal{Z}(\mathcal{A})$ und der Menge $\mathcal{R}(\mathcal{A})$ von Übergangsregeln entpricht der Kripke-Struktur

$$\boldsymbol{K}^{[\mathcal{A}]} = (\boldsymbol{G}^{[\mathcal{A}]}, \boldsymbol{R}^{[\mathcal{A}]}, \boldsymbol{M}^{[\mathcal{A}]}) \quad ,$$

wobei

$$\begin{aligned} \boldsymbol{G}^{[\mathcal{A}]} &= \mathcal{G}(\mathcal{A}) \quad , \\ \boldsymbol{R}^{[\mathcal{A}]} &= \{(g, \vec{r}(g)) : \vec{r} \in \mathcal{R}^*(\mathcal{A}), g \in \mathcal{G}(\mathcal{A})\} \quad , \end{aligned}$$

und $\boldsymbol{M}^{[\mathcal{A}]}(g) = (M^{[\mathcal{A}]}(g), \mathcal{S}(\mathcal{A}))$ für $g \in \boldsymbol{G}^{[\mathcal{A}]}$ die einzelnen statischen Algebren (dort als $g = \mathcal{I} = (I, \mathcal{S})$ bezeichnet) darstellt. Damit wird die Zugänglichkeitsrelation $\boldsymbol{R}^{[\mathcal{A}]}$ als unmittelbare Nachfolgerrelation (nicht-transitiv) modelliert.

In dieser Kripke-Struktur nimmt der Zustand $\mathcal{Z}(\mathcal{A})$ eine Sonderrolle ein:

Definition 21 *Ein Zustand $w \in \boldsymbol{G}$ einer Kripke-Struktur $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ heißt* (eine) Wurzel *von $\boldsymbol{K}$, wenn gilt:*

$$\textit{für alle } g \in \boldsymbol{G} \textit{ gilt } (w, g) \in \boldsymbol{R}^*.$$

Definition 22 *Ein Pfad p in einer Kripke-Struktur mit Wurzel w heißt* vollständig, *wenn $p{\downarrow_0} = w$ ist.*

Satz 1 *Ist $\mathcal{A}$ eine Evolving Algebra, $\boldsymbol{K}^{[\mathcal{A}]} = (\mathcal{G}(\mathcal{A}), \boldsymbol{R}^{[\mathcal{A}]}, \boldsymbol{M}^{[\mathcal{A}]})$ eine diese modellierende Kripke-Struktur, so ist der Zustand $[0] := \mathcal{Z}(\mathcal{A}) \in \mathcal{G}(\mathcal{A})$ eine Wurzel.*

Beweis: Folgt direkt aus der Definition von $\mathcal{G}(\mathcal{A})$ als Menge aller von $\mathcal{Z}(\mathcal{A})$ erreichbaren statischen Algebren. □

Dazu ist anzumerken, daß es durchaus $g \in \boldsymbol{G}^{[\mathcal{A}]}$ geben kann mit $(g, [0]) \in \boldsymbol{R}^{[\mathcal{A}]}$ (vgl. Senden einer Nachricht, die sofort wieder vom Medium verloren wird).

Die vollständigen Pfade in $\boldsymbol{K}^{[\mathcal{A}]}$ repräsentieren somit genau die möglichen Berechnungsfolgen der Evolving Algebra $\mathcal{A}$.

Die Beschreibung der *Berechnungen* der Evolving Algebra ist jedoch insofern nicht vollständig, daß aus der Zugänglichkeitsrelation nicht mehr ersichtlich ist, durch welche Regelanwendung(en) ein Übergang begründet wurde. Dieses ist aber für die Behandlung der Fairnessforderungen notwendig und erleichert die Axiomatisierung. Daher muß bei der logischen Beschreibung in jedem Zustand zusätzlich Information abgelegt werden, welche Regeln bei dem Übergang aus dem Vorgängerzustand angewendet wurden.

Zu diesem Zweck wird die Signatur $\Sigma(\mathcal{A})$ um die Namen der Regeln aus $\mathcal{R}(\mathcal{A})$ zu $\Sigma^+(\mathcal{A})$ ergänzt. Diese werden den zustandsabhängig interpretierten nullstelligen Funktionen zugeschlagen.

Durch State-Splitting erhält man eine detailliertere Kripke-Struktur $\boldsymbol{K}^{\mathcal{A}} = (\boldsymbol{G}^{\mathcal{A}}, \boldsymbol{R}^{\mathcal{A}}, \boldsymbol{M}^{\mathcal{A}})$ über der Signatur $\Sigma^+(\mathcal{A})$ mit

$$(M^{\mathcal{A}}(g))(\mathsf{name}) = \mathit{true} \;\Leftrightarrow\; \begin{array}{l}\text{im Übergang zu dem Zustand } g \text{ wurde die}\\ \text{mit } \mathit{name} \text{ bezeichnete Regel angewendet.}\end{array}$$

Da ein Zustand von $\boldsymbol{K}^{[\mathcal{A}]}$ – aus verschiedenen Vorgängerzuständen – durch Anwendung verschiedener Regeln erreichbar sein kann, ergeben sich mit der erweiterten Signatur aus diesem bis zu $2^{|\mathcal{R}(\mathcal{A})|}$ neue Zustände für $\boldsymbol{K}^{\mathcal{A}}$.

Insbesondere erhält man aus dem Startzustand $\mathcal{Z}(\mathcal{A}) = [0] \in \boldsymbol{G}^{[\mathcal{A}]}$ einen Zustand 0 mit

$$\begin{array}{ll}(M^{\mathcal{A}}(0))(f) = (M^{\mathcal{A}}([0]))(f) & \text{für alle } f \in \Sigma\ ,\\ (M^{\mathcal{A}}(0))(\mathsf{name}) = \mathit{false} & \text{für alle Regelbezeichner } \mathit{name}.\end{array}$$

Satz 2 *Mit der eben durchgeführten Konstruktion ist* 0 *die einzige Wurzel von* $\boldsymbol{K}^{\mathcal{A}}$.

Beweis: Das State-Splitting kann iterativ ausgehend von [0] vorgenommen werden, damit ist 0 eine Wurzel. Da jeder Eintrag in der Zugänglichkeitsrelation durch Anwendung von Übergangsregel(n) begründet ist, kann 0 von keinem anderen Zustand erreichbar sein, es kommt also kein anderer Zustand als Wurzel in Frage. □

Definition 23 *Für* $\boldsymbol{K}^{\mathcal{A}}$ *und* 0 *wie eben konstruiert heißt* 0 die Wurzel *von* $\boldsymbol{K}^{\mathcal{A}}$.

Im übrigen ist i.a. weder $\boldsymbol{K}^{[\mathcal{A}]}$ noch $\boldsymbol{K}^{\mathcal{A}}$ ein Baum (man kann aus verschiedenen Zuständen durch Anwendung derselben Regel einen gemeinsamen Zustand erreichen; vgl. Verlieren einer Nachricht aus einer Schlange).

Die Aussage „in Zustand g einer Kripke-Struktur $\boldsymbol{K}^{\mathcal{A}} = (\boldsymbol{G}^{\mathcal{A}}, \boldsymbol{R}^{\mathcal{A}}, \boldsymbol{M}^{\mathcal{A}})$ zu einer Evolving Algebra $\mathcal{A} = (\mathcal{Z}(\mathcal{A}), \mathcal{R}(\mathcal{A}))$ ist Regel $\mathit{name} \in \mathcal{R}(\mathcal{A})$ anwendbar“ ist damit äquivalent zu „es existiert ein Nachfolgezustand h von g, in dem Regel *name* angewendet wurde“ bzw. der Beziehung $g \models \mathsf{E}\circ(\mathsf{name} = \mathsf{true})$. Die Behandlung der Fairnessanforderungen wird von dieser Äquivalenz Gebrauch machen.

Im weiteren soll die Kripke-Struktur $\boldsymbol{K}^{\mathcal{A}}$ temporallogisch axiomatisiert werden. Dabei muß insbesondere den speziellen mit der herausragenden Rolle der Gleichheit zusammenhängenden Erfordernissen der Evolving Algebras Rechnung getragen werden.

2.3 Gleichheit in Evolving Algebras

Wie in Abschnitt 1.12 erwähnt lassen sich globale Schlüsse aus Gleichheitsatomen nur unter der Bedingung ziehen, daß dabei ausschließlich zustandsunabhängig interpretierte Bezeichner involviert sind.

Da es in Evolving Algebras außer der Gleichheit keine weiteren Prädikate gibt, beruhen alle Schlüsse auf der Ausnutzung von Gleichheitsinformationen, d.h. Aussagen der Form „in Zustand g hat der Bezeichner const den Wert x“ und „in Zustand g gilt $y = z$“.

Erst global mit Hilfe einer Gleichheitstheorie zu ziehende Schlüsse aus solchen lokalen Aussagen liefern das Ergebnis. Die (temporal)logische Deduktion ist somit nur ein Mittel, um die für einen endgültigen Schluß notwendigen (Un)gleichungen zu erhalten.

Jede lokale Gleichung $g \models \mathsf{const} = x$ bzw. $g \models y = z$ muß daher zu einer globalen Gleichung umgeformt werden. Enthält die lokale Gleichung keine zustandsabhängig interpretierten Bezeichner, so ist dieser Schritt trivial: $g \models x = y \rightsquigarrow \boldsymbol{K} \models x = y$. Enthält sie jedoch zustandsabhängig interpretierte Bezeichner, so müssen diese mit dem Zustand, auf den sich die Gleichung bezieht, gekennzeichnet werden:

Ist $\boldsymbol{K}^{\mathcal{A}} = (\boldsymbol{G}^{\mathcal{A}}, \boldsymbol{R}^{\mathcal{A}}, \boldsymbol{M}^{\mathcal{A}})$, so wird die Signatur $\Sigma(\boldsymbol{K}^{\mathcal{A}}) = \Sigma^{+}(\mathcal{A})$ ein weiteres Mal, diesmal um die Menge $\{\mathsf{bezeichner}_g : \mathsf{bezeichner} \in \Sigma(\mathcal{A}) \backslash \Sigma^{c}(\mathcal{A}),\ g \in \boldsymbol{G}^{\mathcal{A}}\}$ erweitert. Die durch diese Signaturerweiterung entstehende Kripke-Struktur kann in gewisser Weise als „Vervollständigung“ von $\boldsymbol{K}^{\mathcal{A}}$ verstanden werden, und soll wieder mit $\boldsymbol{K}^{\mathcal{A}}$ bezeichnet werden. Die neuen Elemente werden zustandsunabhängig interpretiert als

$$\text{für alle } h \in \boldsymbol{G}^{\mathcal{A}}\text{: } (M^{\mathcal{A}}(h))(\mathsf{bezeichner}_g) = (M^{\mathcal{A}}(g))(\mathsf{bezeichner}) \quad ,$$

womit man entsprechende global gültige Gleichungen durch

$$g \models \mathsf{bezeichner} = x \rightsquigarrow \boldsymbol{K}^{\mathcal{A}} \models \mathsf{bezeichner}_g = x$$

enthält.

Mit diesen können dann globale gleichungslogische Schlüsse (insbesondere Widersprüche) gezogen werden.

2.4 Axiomatisierung einer Evolving Algebra in CTL

Die durch eine Evolving Algebra $\mathcal{A}$ induzierte Kripke-Struktur $\boldsymbol{K}^{\mathcal{A}}$ kann damit in auf Prädikatenlogik basierender Temporallogik durch eine Axiomenmenge $\mathsf{Ax}(\mathcal{A})$ in CTL axiomatisiert werden.

Dabei gibt es grundsätzlich zwei Möglichkeiten der zu verwendenden Modellrelation, die jeweils unterschiedliche Sichtweisen der Kripke-Struktur repräsentieren:

- Es wird die für Kripke-Strukturen übliche Modellrelation $\models$ aus Definition 13 verwendet:
  Eine Formel F ohne freie Variablen heißt gültig in einer Kripke-Struktur $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$, *in Zeichen $\boldsymbol{K} \models F$, falls $g \models F$ für alle $g \in \boldsymbol{G}$ gilt.*
  Diese Definition ermöglicht offensichtlich eine einfache Behandlung allgemeingültiger Formeln, während die Beschreibung der Wurzel bzw. der von ihr ausgehenden Pfade problematisch ist.

- Um die Wurzel 0 der Kripke-Struktur zu betonen, wird eine neue Modellrelation $\models^0$ eingeführt:

  Definition 24 *Sei $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$, $0 \in \boldsymbol{G}$ Wurzel, F eine Formel der verwendeten Temporallogik ohne freie Variablen. Dann gilt*

  $$\boldsymbol{K} \models^0 F \;:\Leftrightarrow\; 0 \models F \quad .$$

  Für eine Formelmenge $\mathcal{F}$ sind die Begriffe $\models^0$-erfüllbar und $\models^0$-allgemeingültig entsprechend der PL1 definiert.

  Damit werden in allen Zuständen von $\boldsymbol{K}^{\mathcal{A}}$ gültige Formeln durch

  $$\boldsymbol{K}^{\mathcal{A}} \models^0 \mathsf{A}\Box F$$

  formuliert. Da 0 eine Wurzel von $\boldsymbol{K}^{\mathcal{A}}$ ist, gilt

  $$\boldsymbol{K}^{\mathcal{A}} \models^0 \mathsf{A}\Box F \;\Leftrightarrow\; \boldsymbol{K}^{\mathcal{A}} \models F \quad .$$

  Im weiteren wird daher zur Beschreibung von Evolving Algebras die Relation $\models^0$ verwendet.

Damit setzt sich die Axiomenmenge $\mathsf{Ax}(\mathcal{A})$ zur Beschreibung von $\boldsymbol{K}^{\mathcal{A}}$ aus fünf Gruppen zusammen:

- Existenzaxiome
- Gleichheitsaxiome
- Datenstrukturaxiome
- Aktivitätsaxiome
- Beschreibung des Startzustandes

Bei der Beschreibung einer Evolving Algebra stehen deren Bezeichner und die in den einzelnen Zuständen von ihnen angenommenen Werte im Mittelpunkt. Um Zugriff auf letztere zu haben, wird für jeden Bezeichner *bezeichner* der Evolving Algebra ein Axiom

$$\mathsf{A}\Box(\exists x : \mathsf{bezeichner} = x) \quad \text{bzw.} \quad \mathsf{A}\Box(\forall \vec{y} : \exists x : \mathsf{bezeichner}(\vec{y}) = x)$$

aufgenommen.

Die Gleichheits- und Datenstrukturaxiome enthalten Hintergrundinformationen über die verwendeten Datenstrukturen. In den Datenstrukturaxiomen wird die Spezifikation der verwendeten Datenstrukturen beschrieben, z.B. $\mathsf{A}\Box(\mathsf{RIM} = \bot \vee \exists D, B : \mathsf{RIM} = \mathsf{msg}(D, B))$.
Die Gleichheitsaxiome bilden dabei die Basis für die den logischen Kalkül ergänzende Gleichheitstheorie. Sie umfassen immer $\mathsf{true} \neq \mathsf{false}$, als weiteres können durch sie explizit ausgezeichnete (Leer-)Elemente beschrieben werden (z.B. $\mathsf{A}\Box(\forall D, B : \mathsf{msg}(D, B) \neq \bot)$).

Bei ihrer Formulierung muß man berücksichtigen, daß der verwendete Kalkül nicht typisiert ist. Daher müssen unsinnige Instantiierungen ausgeschlossen werden: Beschreibt man etwa Queues durch $\forall Q : Q = \varepsilon \vee \exists D, Q' : Q = D|Q'$, so muß die implizite Annahme, daß Q in der Evolving Algebra eine Queue beschreibt, explizit ausgedrückt werden. Dies kann geschehen, indem anstelle $\forall Q$ die möglichen Bezeichner der Evolving Algebra eingesetzt werden: $\mathsf{SQ} = \varepsilon \vee \exists D, Q' : \mathsf{SQ} = D|Q'$, oder mit Hilfe der in Abschnitt 1.4 beschriebenen Universen: $\forall X : \mathsf{SQ} = X \rightarrow \mathsf{Queue}(X) = \mathsf{true}$, $\forall X : \mathsf{Queue}(X) = \mathsf{true} \rightarrow (X = \varepsilon \vee (\exists D, X' : X = D|X' \wedge \mathsf{Queue}(X') = \mathsf{true}))$.

Die Aktivitätsaxiome erhält man direkt syntaktisch aus den für die Evolving Algebra angegebenen Regeln. Sie unterteilen sich noch einmal in drei Gruppen:

Anwendbarkeit: Axiome, die beschreiben, welche Regeln im entsprechenden Zustand angewendet werden können, bzw. rückblickend, welche Bedingungen im vorhergehenden Zustand erfüllt gewesen sein müssen, wenn angenommen wird, eine bestimmte Regel sei beim Übergang von diesem in den aktuellen Zustand angewendet worden.

Auswirkungen: Axiome, die die Auswirkungen der Regeln beschreiben. Bezugnehmend auf die Werte der Funktionen in einem Zustand werden die Werte der Funktionen im Folgezustand nach Anwendung einer bestimmten Regel – soweit diese von den Zuweisungen dieser Regel beeinflußt werden – beschrieben.

Invarianzen: Axiome, die für jeden zustandsabhängig ausgewerteten Bezeichner der Evolving Algebra beschreiben, welche Regeln den Wert dieses Bezeichners beeinflussen – bzw. bis wann dieser Wert, als Element des Universums verstanden, unverändert bleibt.

Aus einer Regel

$$\mathit{name}: \quad \mathsf{if}\ \text{Vorbedingung}\ \mathsf{then}\ \text{Zuweisung}$$

erhält man zwei Axiome:

Anwendbarkeit:
$\mathsf{A}\Box(\text{Vorbedingung} \leftrightarrow \mathsf{E}\circ(\mathsf{name} = \mathsf{true}))$

Auswirkungen:
$\mathsf{A}\Box(\text{Vorbedingung} \wedge \text{Werte vor Zuweisung} \rightarrow (\mathsf{A}\circ(\mathsf{name} = \mathsf{true} \rightarrow \text{Werte nach Zuweisung})))$

Die Invarianzaussagen erhält man aus der Betrachtung der gesamten Regelmenge:

$$\mathsf{A}\Box(\forall x : \mathsf{bezeichner} = x \rightarrow \mathsf{A}\circ(\mathsf{A}(\mathsf{bezeichner} = x\ \mathsf{unless}\ (\bigvee\{(\mathsf{name} = \mathsf{true}) : \mathsf{bezeichner} \in \text{Zuweisung}(\mathsf{name})\}))))$$

Das eingeschobene $\mathsf{A}\circ$ ist notwendig, da hier nur *echte* Folgezustände betrachtet werden. Mit der strikten Version des unless-Operators erhält man den einfacheren Ausdruck

$$\mathsf{A}\Box(\forall x : \mathsf{bezeichner} = x \rightarrow \mathsf{A}(\mathsf{bezeichner} = x\ \mathsf{unless}^{+}\ (\bigvee\{(\mathsf{name} = \mathsf{true}) : \mathsf{bezeichner} \in \mathit{Zuweisung}(\mathsf{name})\}))) \quad .$$

Korrekte Behandlung zustandsabhängig interpretierter Bezeichner

Dabei stellt sich ein weiteres der in Abschnitt 1.12 erwähnten Probleme: Um den Anforderungen der zustandsabhängig interpretierten Funktionen gerecht zu werden, muß bei der Anwendung eines solchen Invarianzaxioms auf der rechten Seite der Gleichheit ein zustandsunabhängig interpretierter Bezeichner stehen, um das entsprechende Element des Universums festzuhalten:

Bsp: erlaubt: $\mathsf{bezeichner} = \mathsf{unabh}\ \mathsf{unless}\ \mathsf{name}$
nicht erlaubt: $\mathsf{bezeichner} = \mathsf{bezeichner}'\ \mathsf{unless}\ \mathsf{name}$

Dies kann etwa ein durch Skolemisierung aus einem Existenzaxiom gewonnener Bezeichner sein.

Dies ist soweit der direkte prädikatenlogische Weg. Mit dieser Axiomenmenge können Beweise mit jedem temporal-prädikatenlogischen Beweiser (das muß nicht einmal ein Tableaubeweiser sein) geführt werden, falls er über die in Abschnitt 2.3 beschriebene Gleichungsbehandlung verfügt.

Mit der im vorhergehenden Abschnitt vorgenommenen Signaturerweiterung kann man folgende Formulierung verwenden:

$$\mathsf{bezeichner} = \mathsf{bezeichner}_g \ \mathsf{unless\ name} \quad .$$

Optimierungen in dieser Richtung werden in Abschnitt 9.3 behandelt.

2.5 Anforderungen an die Logik

Als minimale Grundlage, um temporale Logik nutzbringend anzuwenden, kann CTL gelten, da erst mit dem until-Operator temporal-kausale Beziehungen formulierbar sind.

Zur Beschreibung von Abläufen ist insbesondere der erst in CTL$^+$ enthaltene unless-Operator notwendig, der beschreibt, daß eine im aktuellen Zustand gültige Formel so lange weiter gilt, bis eine bestimmte andere Formel in einem Folgezustand erfüllt ist (z.B. um auszudrücken „Wenn das Licht leuchtet, leuchtet es, bis der Schalter wieder gedrückt wird – eventuell für immer“).

Um Fairnessaussagen einbringen zu können, ist es notwendig, zumindest über das Endstück eines Pfades Aussagen in CTL* machen zu können.

2.6 Bestehende Entscheidungsverfahren

Model Checking:
In [CES86] und [EL85] wird ein Model-Checking-Verfahren angegeben, mit dem überprüft werden kann, ob ein gegebenes System mit endlich vielen Zuständen eine in aussagenlogischer CTL bzw CTL*-Syntax beschriebene Spezifikation erfüllt. Die Einbeziehung von Fairnessforderungen erfolgt durch Erweiterungen des Algorithmus.

Durch die Beschränkung auf Aussagenlogik und endlich viele verschiedene Zustände ist das Verfahren für Datenstrukturen mit unendlichem Wertebereich (z.B. Queues) ungeeignet. Jede Modellierung eines Protokolls als finite-state abstrahiert von dem Verhalten solcher Strukturen, womit prinzipiell die Korrektheit und Vollständigkeit der vorgenommenen Abstraktion bewiesen werden müßte.

Tableaux:
In [BMP81, EH82, Wol85] wird ein Tableaukalkül für aussagenlogisches CTL beschrieben. Die Tableauprozedur basiert auf den Äquivalenzen $\Box F \equiv F \wedge \circ\Box F$, $\Diamond F \equiv F \vee \circ\Diamond F$ und $F \ \mathsf{until}\ G \equiv G \vee ((F \wedge \neg G) \wedge \circ(F \ \mathsf{until}\ G))$. Dabei wird nach der Idee der Blocktableaux vorgegangen: Jeder Tableauknoten enthält eine Menge von Formeln. Es werden iterativ vom Startzustand ausgehend alle möglichen Nachfolgezustände nach der Fragestellung

- Was gilt im aktuellen Zustand?

- Was gilt in den direkten Nachfolgezuständen?

beschrieben. Somit wird die Komplexität der betrachteten Formeln i.a. nicht reduziert.
Erst wenn *alle* Formeln des aktuellen Zustands aufgelöst sind, werden die Nachfolgezustände betrachtet, d.h. Knoten für sie erzeugt.

Zustände, die identisch mit einem schon erzeugten Zustand sind, werden nicht noch einmal erzeugt; statt dessen wird der Pfad des Tableaux zu dem schon existierenden Knoten weitergeführt. Dabei werden Zyklen im Tableau erzeugt. Durch die Beschränkung auf Aussagenlogik können nur endlich viele verschiedene Zustände (vgl. Fischer-Ladner-Closure, [EH82, FL79]) beschrieben werden, womit die Terminierung des Algorithmus garantiert ist. Im Gegensatz zu den klassischen PL1-Tableaux, wo jeder Zweig des Tableaux eine komplette PL1-Struktur beschreibt, ist die Semantik hier grundsätzlich anders: Es besteht eine eindeutige Zuordnung von Tableauknoten zu Zuständen einer Kripke-Struktur sowie von Zweigen des Tableaux zu Pfaden einer Kripke-Struktur. Im Falle eines nicht geschlossenen Tableaux beschreibt somit die „geographische“ Struktur des Tableaux als Graphen die Zugänglichkeitsrelation, während die einzelnen Knoten des Tableaux die einzelnen aussagenlogischen Strukturen eines Modells beschreiben.

Die Überprüfung von Erreichbarkeitsaussagen bei nicht geschlossenen Tableaux erfordert eine Nachbehandlung, die Graphenalgorithmen benötigt. Da nur CTL-Syntax verarbeitet werden kann, kann Fairness nicht beschrieben werden.

Beide Verfahren können nicht auf Prädikatenlogik erweitert werden, da die endliche Anzahl möglicher Zustände einen zentralen Punkt ihres Funktionierens darstellt.

Um bei Tableaux diese Einschränkung zu überwinden muß die Behandlung von Erreichbarkeitsaussagen grundlegend anders gestaltet werden: Bei diesen muß von der Anzahl der (beliebig vielen) Zwischenzustände, die vor der Erfüllung der Erreichbarkeitsaussage liegen, abstrahiert werden können.

Im weiteren ist wieder eine 1:1-Korrespondenz von Tableauzweigen zu Kripke-Strukturen anzustreben.

3 Semantische Modellierung verzweigender Zeit durch Tableaux

In diesem Kapitel wird die verwendete Modellierung von Kripke-Strukturen durch Tableaux vorgestellt.

Um eine begriffliche Unterscheidung zwischen den beiden verwendeten Graphen „Kripke-Struktur“ und „Tableau“ zu schaffen, werden die Begriffe „Pfad“ und „Zustand“ für Kripke-Strukturen bzw. die durch sie repräsentierten Evolving Algebras verwendet, während für Tableaux die Begriffe „Zweig“ und „Knoten“ Anwendung finden.

Wie bei Tableaux üblich, wird, um die Allgemeingültigkeit einer Formel F nachzuweisen, ein Widerspruchsbeweis angesetzt: Es wird gezeigt, daß die Formel $\neg F$ inkonsistent ist – also kein Modell hat. Dazu wird systematisch versucht, ein Modell von $\neg F$ zu konstruieren, mit der Absicht, nachzuweisen, daß jeder solche Versuch mißlingen muß.
Im vorliegenden Fall soll also eine temporale Kripke-Struktur konstruiert werden, die $\neg F$ bzgl. der für Evolving Algebras zu bevorzugenden Modellrelation $\models^0$ erfüllt.
Die aus der Prädikatenlogik bekannte Situation, eine eine Formelmenge verifizierende Interpretation der Prädikate zu finden, tritt hier vielfach auf: Für jeden einzelnen Zustand muß eine solche Interpretation gefunden werden. Zu diesem Zweck wird der bekannte Tableaukalkül der Prädikatenlogik in den zu konstruierenden Tableaukalkül eingebettet. Zusätzlich muß aus diesen prädikatenlogischen Strukturen eine temporale verzweigende Struktur gebildet werden.

Aus semantischer Sicht muß man dazu sowohl beliebig viele einzelne Zustände als auch die zwischen diesen bestehenden Beziehungen durch das Tableau beschreiben können. Diese beinhalten einerseits die Reihenfolge von Zuständen auf einem Pfad, andererseits die Zusammenhänge verschiedener Pfade.

Damit sind drei Arten von Entities zu beschreiben: Elemente des Universums innerhalb eines Zustandes, Zustände und Pfade. In der gewählten Semantik werden Zustände und Pfade explizit benannt (im Gegensatz zu dem in [BMP81] vorgestellten Kalkül).

Im Zuge der Konstruktion des Tableaux werden solche Entities jeweils problemorientiert explizit benannt – bzw. für das Tableau „erzeugt“. Dies geschieht genau dann, wenn eine Existenzformel die Existenz einer Entity einer der drei Arten fordert:

- Elemente des Universums: Es wird die Vorgehensweise aus der Prädikatenlogik übernommen. Neben den Bezeichnern des Vokabulars der Struktur werden bei Bedarf (Auflösung einer δ-Formel $(\exists x : F(x))$) Skolemkonstanten und -funktionen eingeführt, die das gesuchte Element beschreiben. Zwischen diesen Konstanten besteht – soweit nicht explizit durch Formeln angegeben – keine Ordnung. Dabei wird jeweils eine neue Skolemkonstante verwendet – also oEdA. angenommen, daß das beschriebene Element von den anderen verschieden ist.
- Zustände: Zustände werden jeweils auch nur als Reaktion auf Existenzaussagen (diese sind z.B. von der Form $\Diamond F$ oder $\circ F$) benannt. Dabei ist zu berücksichtigen, daß ein Zustand in der gewählten Semantik auf einem Pfad liegen muß, und so zwischen ihm und den anderen Zuständen eine

(partielle) Ordnungsrelation besteht. Neue Zustände können nicht nur am Ende eines Pfades benannt werden, sondern auch auf Pfadabschnitten zwischen schon benannten Zuständen. Aus diesem Grund muß bei der Benennung eines neuen Zustandes sowohl die Möglichkeit betrachtet werden, daß dieser Zustand mit einem bereits beschriebenen identisch ist, als auch, daß es sich um einen momentan noch nicht beschriebenen Zustand handelt. Bei Benennung eines neuen Zustandes muß sichergestellt werden, daß die im Tableau enthaltenen Informationen über die verschiedenen Pfade vollständig und konsistent bleiben.

- Pfade: Ein Pfad wird benannt, wenn seine Existenz durch eine Aussage der Form $\mathsf{E}P$ gefordert wird. Da zwischen den einzelnen Pfaden keine Ordnung besteht, kann ein neuer Pfad ab dem Zustand, in dem seine Existenz gefordert wird, als von allen anderen verschieden angenommen werden. Pfade werden als Folge von Zuständen beschrieben. Bei der Benennung von Pfaden gibt es grundsätzlich zwei Möglichkeiten (beide werden im weiteren Verlauf verwendet):
  1. Ein Pfad wird in dem Zustand, in dem seine Existenz gefordert wird, als Abzweig von einem anderen beginnend angenommen, womit bei seiner Erzeugung nur dieser Zustand auf ihm bekannt ist.
  2. Es werden nur vollständige Pfade beschrieben, d.h. jeder Pfad wird als an der Wurzel beginnend angenommen. Damit verläuft ein neu zu benennender Pfad von der Wurzel bis zu dem Zustand, in dem seine Existenz gefordert wird, parallel zu allen Pfaden, die ebenfalls durch diesen Zustand verlaufen. Bei seiner Erzeugung sind auf ihm genau die zwischen der Wurzel und diesem Zustand liegenden schon bekannten Zustände bekannt.

Dabei können im allgemeinen zwischen zwei bekannten Zuständen beliebig viele weitere (noch unbekannte) Zustände liegen und ggf. benannt werden. Dies ermöglicht eine direkte Auflösung von Erreichbarkeitsaussagen und daraus folgend, daß jede Formel zu jedem beliebigen Zeitpunkt aufgelöst werden kann.

Um die Benennung von Zuständen an beliebigen Stellen der beschriebenen Struktur zu ermöglichen, enthalten die Pfadbeschreibungen neben der Reihenfolge der auf ihnen benannten Zustände auch Informationen, welche Formeln auf den Abschnitten zwischen diesen Zuständen gelten müssen. Diese werden dann für eventuell dort neu benannte Zustände verwendet.

Durch diese Vorgehensweise wird von den einzelnen auf diesen Pfadabschnitten liegenden Zuständen sowie deren Anzahl insoweit abstrahiert, als daß nur diejenigen Formeln aufgeführt werden, die in allen Zuständen dieses Abschnittes gelten, und nur festgelegt ist, daß sich auf einem solchen Pfadabschnitt endlich viele Zustände befinden.

Als konzeptionelle Fortsetzung der PL1-Tableaux entspricht jeder Zweig des Tableaux (bzw. die auf ihm enthaltene Formelmenge) einer kompletten Kripke-Struktur. Damit besteht im Gegensatz zu dem in [BMP81] vorgestellten Kalkül keine Zuordnung von Zweigen des Tableaux zu Pfaden der Kripke-Struktur.

Aufgrund der durchgeführten Abstraktion ist es allerdings nicht mehr möglich, aus einem nicht zu schließenden Tableau direkt eine die Formelmenge erfüllende Kripke-Struktur zu erzeugen. Ebenso wird die Vollständigkeit des Kalküls negativ beeinflußt – im Vorgriff auf Abschnitt 5.7 ist jedoch festzustellen, daß

die praktische Nutzbarkeit mit dem Ziel der Beschreibung des Verhaltens von Evolving Algebras darunter nicht leidet.

3.1 Konstruktion der Kalkülfamilie $\mathcal{TK}$

Ausgehend von einer Eingabemenge $\mathcal{F}$ von Formeln über einer Signatur Σ wird systematisch versucht, eine Struktur zu erzeugen, die Modell (bzgl. $\models^0$) von $\mathcal{F}$ ist.

Da jeder Zweig des Tableaux für sich genommen eine komplette Kripke-Struktur repräsentiert, werden neben dem prädikatenlogischen Teil auch Informationen über den Aufbau der Kripke-Struktur durch Tableauknoten beschrieben.

Zur Unterscheidung und Bezeichnung der im Tableau beschriebenen Zustände wird für den prädikatenlogischen Anteil ein auf dem freie-Variablen-Tableau-Kalkül aus [Ree87, Schm87] basierender Tableaukalkül mit *Präfixen* verwendet. Eine Zustandsformel F, die in einem bestimmten Zustand gültig sein soll, tritt damit im Tableau als *Präfixformel*

$$\gamma : F$$

auf, wobei das Präfix γ zur Unterscheidung der verschiedenen Zustände dient[2]. Die im Tableau beschriebenen Pfade werden durch *Pfadbezeichner* benannt. Für diese Pfadbezeichner enthalten *Pfadinformationsformeln* Informationen über die auf ihnen liegenden Präfixe.

Die Signatur $\Sigma_{\mathcal{T}}$ des Tableaux teilt sich somit in Σ_L für den prädikatenlogischen Anteil sowie Σ_T für den temporalen Anteil auf. Σ_L umfaßt neben Σ für jedes $n \in \mathbf{N}$ eine abzählbar unendliche Menge von n-stelligen (Skolem-)Funktionssymbolen und eine (ebenfalls abzählbar unendliche) Menge von Variablen X_i. Σ_T besteht aus einer Menge $\hat{\Gamma}$ von *Präfixsymbolen* sowie einer Menge $\hat{\Lambda}$ von *Pfadsymbolen*, beide enthalten für jedes $n \in \mathbf{N}$ eine abzählbar unendliche Menge von n-stelligen Präfix- bzw. Pfadsymbolen.

Eine Interpretation von $\Sigma_{\mathcal{T}}$ teilt sich entsprechend auf. Die Interpretation von Σ_L wird durch eine geeignete Kripke-Struktur übernommen. Dabei werden die zusätzlichen Skolemfunktionssymbole zustandsunabhängig interpretiert (da sie Elemente des Universums festhalten sollen, vgl. Abschnitt 1.12). Entsprechend ist Σ_L^c der gesamte zustandsunabhängig interpretierte Teil von Σ_L. Die Definition einer Interpretation für Σ_T wird aufgeschoben, bis etwas mehr über Präfix- und Pfadsymbole bzw. Präfixe und Pfadbezeichner bekannt ist.

Das Vorgehen bei der Bildung von Präfixen und Pfadbezeichnern aus Präfix- und Pfadsymbolen entspricht grundsätzlich der δ-Regel des PL1-Tableaukalküls, wobei die Präfix- und Pfadsymbole die Rolle der Skolemfunktionen übernehmen:
Im Zuge der Tableaukonstruktion müssen die aus der γ-Regel entstehenden freien Variablen und die darauf angewendeten Substitutionen berücksichtigt werden. Präfixe γ und Pfadbezeichner λ sind daher Funktionsterme, die aus einem Präfixsymbol $\hat{\gamma}$ bzw. einem Pfadsymbol $\hat{\lambda}$ als Funktionssymbol einer sich

[2] der Leser wird gebeten, die doppelte Verwendung von γ als Name einer Tableauregel sowie als „typisches“ Präfix zu entschuldigen.

aus der Tableaukonstruktion ergebenden Stelligkeit n und einem n-Tupel von Termen als Argument bestehen. Im allgemeinen gibt es ein ausgezeichnetes nullstelliges Präfixsymbol $\hat{0}$, das die Wurzel der modellierten Kripke-Struktur bezeichnet. Zusätzlich gibt es ein Symbol $\hat{\infty}$, das *kein* Präfixsymbol ist, aber ähnlich verwendet wird.

Definition 25 *Es sei $\hat{\Gamma}$ die Menge der Präfixsymbole, $\hat{\Lambda}$ die Menge der Pfadsymbole, Σ_L^c wie oben definiert. Damit ist*

$$\Gamma := \{\hat{\gamma}(t_1, \ldots, t_n) : \hat{\gamma} \in \hat{\Gamma} \text{ ein } n\text{-stelliges Präfixsymbol, } t_1, \ldots, t_n \in \mathsf{Term}_{\Sigma_L^c}\}$$

die Menge der Präfixe *und*

$$\Lambda := \{\hat{\lambda}(t_1, \ldots, t_n) : \hat{\lambda} \in \hat{\Lambda} \text{ ein } n\text{-stelliges Pfadsymbol, } t_1, \ldots, t_n \in \mathsf{Term}_{\Sigma_L^c}\}$$

die Menge der Pfadbezeichner.

Bei den so entstehenden Termmengen $\Gamma, \Lambda \subset \mathsf{Term}_{\Sigma_\mathcal{T}}$ ist genau das führende Funktionssymbol ein Präfix- bzw. Pfadsymbol aus Σ_T und alle Argumentterme aus $\mathsf{Term}_{\Sigma_L^c}$. Diese werden somit durch $\boldsymbol{K}$ ausgewertet.

Als Ergänzung zu der Σ_L interpretierenden Kripke-Struktur $\boldsymbol{K}$ wird eine „Interpretation“ der in Σ_T enthaltenen Präfix- und Pfadsymbole definiert. Die durch diese gegebenen Auswertungen ordnen den Präfixen und Pfadbezeichnern die durch sie beschriebenen Entities zu:

Definition 26 *Eine P&P-Interpretation („Präfixe-und-Pfade-I.“) zu einer Kripke-Struktur $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ mit konstantem Universum $\boldsymbol{U}$ und einer Menge $\boldsymbol{P}(\boldsymbol{K})$ von Pfaden ist ein Tripel*

$$\Omega = (\Phi, \Pi, \Psi) \quad ,$$

wobei

$\Phi : \Lambda \times \Xi \to \boldsymbol{P}(\boldsymbol{K})$	*eine Auswertung der Pfadbezeichner,*
$\Pi : \Lambda \times (\Gamma \cup \{\hat{\infty}\}) \times \Xi \to \hat{\mathbb{N}}$	*eine Auswertung von Paaren von Pfadbezeichnern und Präfixen und*
$\Psi : \Gamma \times \Xi \to \boldsymbol{G}$	*eine Auswertung der Präfixe*

ist.

Dazu hat man entsprechend die folgenden Abbildungen ϕ, π und ψ der einzelnen Pfad- bzw. Präfixsymbole:

$\phi : \hat{\Lambda} \to \boldsymbol{U}^n \to \boldsymbol{P}(\boldsymbol{K})$,
eine Funktion, die jedem n-stelligen $\hat{\lambda} \in \hat{\Lambda}$ eine Funktion $\phi(\hat{\lambda}) : \boldsymbol{U}^n \to \boldsymbol{P}(\boldsymbol{K})$ bzw. $\phi(\hat{\lambda}) : \boldsymbol{U}^n \to \mathbb{N} \to \boldsymbol{G}$ zuordnet,

$\pi \in \hat{\Lambda} \times \hat{\Gamma} \to (\boldsymbol{U}^n \times \boldsymbol{U}^m) \to \mathbb{N}$,
eine (i.a. nicht totale) Funktion, die Paaren von einem n-stelligen $\hat{\lambda} \in \hat{\Lambda}$ und einem m-stelligen $\hat{\gamma} \in \hat{\Gamma}$ eine Funktion $\pi(\hat{\lambda}, \hat{\gamma}) : \boldsymbol{U}^n \times \boldsymbol{U}^m \to \mathbb{N}$ zuordnet,

und

$\psi : \hat{\Gamma} \to \boldsymbol{U}^n \to \boldsymbol{G}$,
eine Funktion, die jedem n-stelligen $\hat{\gamma} \in \hat{\Gamma}$ eine Funktion $\psi(\hat{\gamma}) : \boldsymbol{U}^n \to \boldsymbol{G}$ zuordnet.

Auf der Basis von ϕ, π und ψ werden Φ, Π und Ψ folgendermaßen definiert: Sei $\lambda = \hat{\lambda}(t_1, \ldots, t_n) \in \Lambda$ und $\gamma = \hat{\gamma}(s_1, \ldots, s_m) \in \Gamma$.

$$\begin{aligned}
&\Phi(\lambda, \chi) := (\phi(\hat{\lambda}))(\boldsymbol{K}(t_1, \chi), \ldots, \boldsymbol{K}(t_n, \chi)) \quad , \\
&\Pi(\lambda, \gamma, \chi) := (\pi(\hat{\lambda}, \hat{\gamma}))(\boldsymbol{K}(t_1, \chi), \ldots, \boldsymbol{K}(t_n, \chi), \boldsymbol{K}(s_1, \chi), \ldots, \boldsymbol{K}(s_m, \chi)) \quad , \\
&\Pi(\lambda, \hat{\infty}, \chi) := \infty \in \hat{\mathbb{N}} \quad , \\
&\Psi(\gamma, \chi) := (\psi(\hat{\gamma}))(\boldsymbol{K}(s_1, \chi), \ldots, \boldsymbol{K}(s_m, \chi)) \quad .
\end{aligned}$$

Prinzipiell ist Ω ähnlich organisiert wie eine prädikatenlogische Interpretation $\boldsymbol{I} = (I, \boldsymbol{U})$, wenn man die entsprechenden „Universen“ bestimmt:

$$\Phi = (\phi, \boldsymbol{P}(\boldsymbol{K})) \quad , \quad \Pi = (\pi, \hat{\mathbb{N}}) \quad , \quad \Phi = (\psi, \boldsymbol{G}) \quad .$$

Damit sind Φ, Π und Ψ Auswertungen von Termen, bei denen nur das führende Funktionssymbol durch Ω interpretiert wird, während alle Argumentterme durch $\boldsymbol{K}$ als „übliche“ zustandsunabhängig interpretierte Terme ausgewertet werden.

Darauf basierend kann nun die in den Tableaux verwendete Syntax festgelegt werden: Es sei $\mathcal{L}$ die Menge der jeweiligen im Tableau verwendeten Zustandsformeln.

Der Aufbau der Kripke-Struktur wird in *Pfadinformationsformeln* der Form

$$\lambda : [\gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}]$$

codiert, wobei $\lambda \in \Lambda$, $\gamma_i \in \Gamma$ und $L_i \in 2^{\mathcal{L}} \cup \{\circ\}$ ist.

Definition 27 *Für eine P&P-Interpretation $\Omega = (\Phi, \Pi, \Psi)$ heißt eine Pfadinformationsformel $I = \lambda : [\gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}]$* konsistent zu Ω *für eine Belegung χ, falls für alle i*

$$\Pi(\lambda, \gamma_0, \chi) = 0 \quad , \quad \Pi(\lambda, \gamma_i, \chi) < \Pi(\lambda, \gamma_{i+1}, \chi) \quad \textit{und} \quad \Psi(\gamma_i, \chi) = \Phi(\lambda, \chi, \Pi(\lambda, \gamma_i, \chi))$$

gilt und jedes $\hat{\gamma}$ in I höchstens einmal vorkommt.

Dies bedeutet, daß der Pfad $\Phi(\lambda, \chi) = (g_0, g_1, \ldots)$ in $\boldsymbol{K}$ in dem Zustand $g_0 = \Psi(\gamma_0, \chi)$ beginnt und die weiteren auf ihm bekannten Zustände $g_{\Pi(\lambda,\gamma_1,\chi)} = \Psi(\gamma_1, \chi), \ldots, g_{\Pi(\lambda,\gamma_n,\chi)} = \Psi(\gamma_n, \chi)$ in der angegebenen Reihenfolge durchläuft.

Die im weiteren verwendeten P&P-Interpretationen und Pfadinformationsformeln werden so konstruiert, daß sie, dort wo es relevant ist, konsistent sind. Da an dieser Stelle noch kein passender Modellbegriff definiert ist, kann diese Aussage noch nicht bewiesen werden. Ihr Beweis ist in den Beweis des Korrektheitslemmas (Satz 12) eingebaut.

Logische Formeln treten im Tableau als *Präfixformeln* der Gestalt

$\gamma : F$, wobei $\gamma \in \Gamma$ ist, derselbe Zweig des Tableaux eine Pfadinformationsformel $\lambda : [\ldots, \gamma, \ldots]$ enthalten muß und $F \in \mathcal{L}$ eine Zustandsformel gemäß der jeweiligen tableauinternen Syntax ist,

auf.

Jeder Tableauknoten enthält somit entweder eine Pfadinformationsformel oder eine Präfixformel.

Ist $\mathcal{F}$ eine Menge von Pfadinformationsformeln und Präfixformeln, so kann eine Modellrelation $\models\!\!\models$ zwischen einer Kripke-Struktur $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ mit der Menge $\boldsymbol{P}(\boldsymbol{K})$ von Pfaden, einer Belegung χ der in $\mathcal{F}$ freien Variablen und $\mathcal{F}$ bzw. einzelnen Pfadinformationsformeln und Präfixformeln folgendermaßen erklärt werden:

$$(\boldsymbol{K}, \chi) \models\!\!\models \mathcal{F} \;:\Leftrightarrow\; \text{es gibt eine P\&P-Interpretation } \Omega = (\Phi, \Pi, \Psi), \text{ so daß } (\boldsymbol{K}, \Omega, \chi) \models\!\!\models \mathcal{F} \text{ gilt,}$$

wobei $\models$ als Relation zwischen einer Kripke-Struktur $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ mit der Menge $\boldsymbol{P}(\boldsymbol{K})$ von Pfaden, einer P&P-Interpretation Ω, einer Formelmenge $\mathcal{F}$ und einer Belegung χ der in $\mathcal{F}$ freien Variablen unter Verwendung der modallogischen Wahrheitsrelation $\models := \models_{\mathrm{Kr}}$ folgendermaßen definiert ist:

1. für alle Präfixformeln $\gamma : F$:

   $$(\boldsymbol{K}, \Omega, \chi) \models\!\!\models \gamma : F \quad :\Leftrightarrow \quad (\Psi(\gamma, \chi), \chi) \models F \quad ,$$

   d.h. in dem dem Präfix γ unter der Belegung χ entsprechenden Zustand ist unter der Variablenbelegung χ die Formel F wahr.

2. für alle Pfadinformationsformeln $I = \lambda : [\gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}]$:

   $$(\boldsymbol{K}, \Omega, \chi) \models\!\!\models \lambda : [\gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}]$$

   genau dann, wenn I für χ konsistent zu Ω ist und für alle $0 \leq i \leq n$ gilt

   $$\begin{aligned} L_i = \circ \;&\Rightarrow\; \Pi(\lambda, \gamma_{i+1}, \chi) = \Pi(\lambda, \gamma_i, \chi) + 1 \quad , \\ L_i \neq \circ \;&\Rightarrow\; \text{für alle } j : \Pi(\lambda, \gamma_i, \chi) < j < \Pi(\lambda, \gamma_{i+1}, \chi) \;\text{ gilt }\; (\Phi(\lambda, \chi, j), \chi) \models L_i \quad , \end{aligned}$$

   d.h. falls $L_i = \circ$ ist, sind $\Pi(\lambda, \gamma_i, \chi)$ und $\Pi(\lambda, \gamma_{i+1}, \chi)$ direkt aufeinanderfolgende Indizes und sonst gilt für alle (endlich, aber beliebig vielen) zwischen $\Phi(\lambda, \chi, \Pi(\lambda, \gamma_i, \chi))$ und $\Phi(\lambda, \chi, \Pi(\lambda, \gamma_{i+1}, \chi))$ auf $\Phi(\lambda, \chi)$ liegenden Zustände g_j $(g_j, \chi) \models L_i$.

Da ein Tableauzweig eine solche Formelmenge ist, ist $\models\!\!\models$ damit eine Relation zwischen Kripke-Strukturen und Tableauzweigen.

Der expliziten Benennung der Pfade durch den Kalkül folgend haben die im Kalkül intern verwendeten Zustandsformeln eine zum Teil detailliertere Form als in den Abschnitten 1.7-1.9 beschrieben.
Insbesondere wird eine syntaktische Möglichkeit vorgesehen, Pfadbezeichner in logischen Formeln zu verwenden: Um die Gültigkeit einer Pfadformel in einem bestimmten Zustand der durch das Tableau beschriebenen Kripke-Struktur für einen bestimmten durch einen Pfadbezeichner λ beschriebenen, diesen Zustand durchlaufenden Pfad $\Phi(\lambda, \chi)$ zu fordern, kann λ als *Pfadselektor* syntaktisch an die Stelle des in der Pfadformel ersten Pfadquantors treten.

Es ergibt sich die folgende Syntaxbeschreibung der in einem Tableau vorkommenden Knotenformeln, auf der alle diesem Ansatz folgenden Tableaukalküle aufbauen:

(TA) Jedes Atom ist eine $\mathcal{TK}$-Zustandsformel.

(TZ1) Sind F und G $\mathcal{TK}$-Zustandsformeln, so sind auch $\neg F$, $F \wedge G$, $F \vee G$ und $F \rightarrow G$ $\mathcal{TK}$-Zustandsformeln.

(TZQ) Ist F eine $\mathcal{TK}$-Zustandsformel und x eine Variable, so sind auch $\forall x : F$ und $\exists x : F$ $\mathcal{TK}$-Zustandsformeln.

(TP1) Sind F und G $\mathcal{TK}$-Zustandsformeln, so sind $\circ F$, $\Box F$, $\Diamond F$, $(F \text{ until } G)$ und $(F \text{ unless } G)$ $\mathcal{TK}$-Pfadformeln.

(TP2) Ist P eine $\mathcal{TK}$-Pfadformel, so ist $\neg P$ eine $\mathcal{TK}$-Pfadformel.

(TZ2) Ist P eine $\mathcal{TK}$-Pfadformel, so sind $\mathsf{A}P$ und $\mathsf{E}P$ $\mathcal{TK}$-Zustandsformeln.

(TC1) Jede $\mathcal{TK}$-Zustandsformel ist eine $\mathcal{TK}$-Prä-Knotenformel

(TC2) Ist P eine $\mathcal{TK}$-Pfadformel und $\lambda \in \Lambda$, so ist λP eine $\mathcal{TK}$-Prä-Knotenformel.

(TK1) Alle Pfadinformationsformeln sind $\mathcal{TK}$-Knotenformeln.

(TK2) Ist F eine $\mathcal{TK}$-Prä-Knotenformel und $\gamma \in \Gamma$ ein Präfix, so ist $\gamma : F$ eine $\mathcal{TK}$-Präfixformel. Alle $\mathcal{TK}$-Präfixformeln sind $\mathcal{TK}$-Knotenformeln.

Wie man sieht, ist die Menge der $\mathcal{TK}$-Pfad-/Zustandsformeln die Menge der CTL-Pfad-/Zustandsformeln, wobei zusätzlich der unless-Operator zugelassen ist.

Definition 28 *Die Semantik der Pfadselektoren ist dabei folgendermaßen erklärt (zum Vergleich wird die Semantik der Pfadquantoren in der entsprechenden Form ebenfalls angegeben):*

Seien $\boldsymbol{K}$, Ω und χ wie oben, $\gamma \in \Gamma$, $\lambda \in \Lambda$.

TZ2a: $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models \gamma : \mathsf{A}P \;:\Leftrightarrow$ *Für alle Pfade $p = (g_0, g_1, \ldots)$ in $\boldsymbol{K}$ und alle n mit $g_n = \Psi(\gamma, \chi)$ gilt $(p|_n, \chi) \models P$.*

TZ2b: $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models \gamma : \mathsf{E}P \;:\Leftrightarrow$ *Es gibt einen Pfad $p(\chi) = (g_0, g_1, \ldots)$ in $\boldsymbol{K}$ und ein $n(\chi)$ mit $g_{n(\chi)} = \Psi(\gamma, \chi)$ und $(p(\chi)|_{n(\chi)}, \chi) \models P$.*

TC2: $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models \gamma : \lambda P \;:\Leftrightarrow\; (\Phi(\lambda, \chi)|_{\Pi(\lambda,\gamma,\chi)}, \chi) \models P$.

Durch die Konstruktion des Kalküls ist es im Fall (TC2) immer so, daß, wenn für einen Tableauzweig T $\gamma : \lambda P \in T$ ist, auch $\hat{\lambda} \in \hat{\Lambda}$ und $\hat{\gamma} \in \hat{\Gamma}$ ist, diese Symbole durch Ω interpretiert werden und T eine Pfadinformationsformel $I = \lambda : [\ldots, \gamma, \ldots]$ enthält, womit die Semantik wohldefiniert ist (vgl. auch Satz 3).

Für das Tableau folgt aus (TZ2a):

Korollar 1 $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models \gamma : \mathsf{A}P \;\Rightarrow$ *Für jede Pfadinformationsformel $I = \lambda : [\gamma_0, L_0, \gamma_1, L_1, \ldots]$ und jedes $\gamma_n = \gamma$ auf T gilt $(\Phi(\lambda, \chi)|_{\Pi(\lambda,\gamma_n,\chi)}, \chi) \models P$.*

Beweis: $\Phi(\Lambda, \chi) \subseteq \boldsymbol{P}(\boldsymbol{K})$. Jeder durch eine Pfadinformationsformel beschriebene Pfad wird auf einen Pfad in $\boldsymbol{K}$ abgebildet. □

Bei der Initialisierung des Tableaux wird entsprechend der Definition der Modellrelation $\models^0$ vorgegangen:

$$\frac{\text{Eingabemenge } \mathcal{F}}{\hat{0} : F \quad \text{für alle } F \in \mathcal{F}}$$

Am Anfang erhält das Tableau keine Pfadinformationsformeln. Dies entspricht einer P&P-Interpretation $\Omega = (\Phi, \Pi, \Psi)$, bei der ψ das nullstellige Präfixsymbol $\hat{0}$ auf die Nullabbildung abbildet. $\Psi(\hat{0}, \chi) = 0$ ist damit die Wurzel der angenommenen Kripke-Struktur.

Nach dieser Auslegung kann eine Menge universell pfadquantifizierter Formeln nicht inkonsistent sein, da sie von einer Kripke-Struktur mit $\boldsymbol{G} = \{0\}$ und $\boldsymbol{R} = \emptyset$ erfüllt werden kann. Eine Variante ist eine Initialisierung mit einer Pfadinformationsformel $\hat{\lambda} : [\hat{0}, \emptyset, \hat{\infty}]$, bei der vorausgesetzt wird, daß diese triviale Kripke-Struktur nicht als Modell akzeptiert werden soll.

Bei der Konstruktion des Tableaux werden durch die Anwendung von Regeln neue Formeln erzeugt. Im Fall prädikatenlogischer Tableaux genügt es, aus einer logischen Formel jeweils einige neue Formeln zu erzeugen und mit diesen einen Zweig des Tableaux fortzusetzen. Ein solcher prädikatenlogischer Anteil ist allen Versionen des Kalküls gemeinsam. Er setzt sich wie üblich aus α-, β-, γ- und δ-Regeln sowie der prädikatenlogischen Abschlußregel zusammen. Dabei werden die α- und β-Regeln aus [Sm68] und die γ- und δ-Regeln aus [Ree87, Schm87, Fit90] bzw. (liberalized-δ-rule) aus [HS94] für freie-Variablen-Tableaux übernommen und an die verwendeten Formeln mit Präfixen angepaßt:

Seien F und G $\mathcal{TK}$-Zustandsformeln, A ein Atom. Sei im folgenden für eine Formelmenge $\mathcal{F}$ $\mathsf{free}(\mathcal{F})$ die Folge der in $\mathcal{F}$ freien Variablen und bezeichne $F[X/x]$ die durch Ersetzung aller Auftreten von x durch X aus F entstehende Formel.

α: $\dfrac{\gamma : F \wedge G}{\begin{array}{c}\gamma : F \\ \gamma : G\end{array}}$ $\qquad$ $\dfrac{\gamma : \neg(F \vee G)}{\begin{array}{c}\gamma : \neg F \\ \gamma : \neg G\end{array}}$

β: $\dfrac{\gamma : F \vee G}{\gamma : F \mid \gamma : G}$ $\qquad$ $\dfrac{\gamma : \neg(F \wedge G)}{\gamma : \neg F \mid \gamma : \neg G}$

γ: $\dfrac{\gamma : \forall x : F}{\gamma : F[X/x]}$ $\qquad$ $\dfrac{\gamma : \neg\exists x : F}{\gamma : \neg F[X/x]}$

wobei X eine noch nicht vorkommende Variable ist.

δ: $\dfrac{\gamma : \exists x : F}{\gamma : F[f(\mathsf{free}(T))/x]}$ $\qquad$ $\dfrac{\gamma : \neg\forall x : F}{\gamma : \neg F[f(\mathsf{free}(T))/x]}$

wobei f ein noch nicht verwendetes (Skolem)-funktionssymbol und T der aktuelle Tableauzweig ist.

Abschlußregel: Sei σ eine Substitution mit $\mathsf{Bild}(\sigma) \subset \mathsf{Term}_{\Sigma_L^c}$ (d.h. es werden nur zustandsunabhängig interpretierte Terme eingesetzt, vgl. Substitutionslemma, Seite 74):

$$\frac{\gamma : \sigma(A) \qquad \gamma : \neg\sigma(A)}{\bot}$$

σ auf das gesamte Tableau anwenden.

Im folgenden werden zu den einzelnen Logiken darauf basierende Tableaukalküle angegeben. Sie unterscheiden sich entsprechend den unterschiedlichen Anforderungen der Logiken in der Behandlung der verschiedenen Pfade bzw. Pfadinformationsformeln sowie den Regeln zur Auflösung der Formeln.

Zur Auflösung temporallogischer Präfixformeln muß neben der aufzulösenden Formel auf Informationen über den Aufbau der Kripke-Struktur zugegriffen werden. Dieser ist in den Pfadinformationsformeln abgelegt. Eine solche Präfixformel wird damit in einem Schritt jeweils „entlang“ einer Pfadinformationsformel aufgelöst, womit die Tableauregeln im allgemeinen die folgende Form haben:

$$\frac{\begin{array}{c}\text{Präfixformel}\\ \text{Pfadinformationsformel}\end{array}}{\begin{array}{c}\text{Präfixformeln}\\ \text{Pfadinformationsformeln}\end{array}}$$

Die Verbindung zwischen der aufzulösenden Präfixformel und der mitverwendeten Pfadinformationsformel wird durch das Präfix und ggf. den in der Präfixformel genannten Pfadselektor hergestellt.

Durch die in den folgenden Abschnitten angegebenen Tableauregeln erfüllt jedes Tableau diesbezüglich die folgenden Bedingungen:

Satz 3 *Sei $\mathcal{T}$ ein Tableau.*

1. *Ist $\hat{\lambda}$ ein Pfadsymbol, $\lambda = \hat{\lambda}(t_1, \ldots, t_n)$ und $\lambda' = \hat{\lambda}(t'_1, \ldots, t'_n)$ in $\mathcal{T}$ vorkommende Pfadbezeichner (einschließlich den in Präfixformeln vorkommenden Auftreten als Pfadselektoren), so ist für alle $i : t_i = t'_i$.*
   Anders formuliert: Ein Tableau enthält zu jeder Zeit für jedes Pfadsymbol höchstens einen mit diesem gebildeten Pfadbezeichner.
2. *Ist $\hat{\gamma}$ ein Präfixsymbol, $\gamma = \hat{\gamma}(t_1, \ldots, t_n)$ und $\gamma' = \hat{\gamma}(t'_1, \ldots, t'_n)$ in $\mathcal{T}$ vorkommende Präfixe (einschließlich den in Pfadinformationsformeln vorkommenden Auftreten), so ist für alle $i : t_i = t'_i$.*
   Anders formuliert:
   Ein Tableau enthält zu jeder Zeit für jedes Präfixsymbol höchstens ein mit diesem gebildetes Präfix.

! *Ist f ein Skolemfunktionssymbol, $f(t_1, \ldots, t_n)$ und $f(t'_1, \ldots, t'_n)$ in $\mathcal{T}$ vorkommende Auftreten von f, so ist für alle $i : t_i = t'_i$.*
Dieser Punkt ist für den weiteren Verlauf nicht wichtig, deutet aber an, warum (1.) und (2.) gelten: Ein solches Symbol wird genau einmal mit n freien Variablen eingeführt. Danach wird der entsprechende Term nur noch durch im gesamten Tableau ausgeführte Substitutionen verändert.

3. *Ist* $I = \hat{\lambda}(t_1, \ldots, t_n) : [I']$ *eine in* $\mathcal{T}$ *vorkommende Pfadinformationsformel, so kommt jedes Präfixsymbol (und jedes Präfix) höchstens einmal in* I' *vor.*

4. *Ist* T *ein Tableauzweig,* $\hat{\lambda}$ *ein Pfadsymbol,* $I = \hat{\lambda}(t_1, \ldots, t_n) : [I'] \in T$ *und* $J = \hat{\lambda}(t_1, \ldots, t_n) : [J'] \in T$ *zwei Pfadinformationsformeln, wobei oEdA.* J *die „neuere" ist, so kommen alle in* I' *vorkommenden Präfixe auch in* J' *vor.*

5. *Tritt ein Pfadbezeichner* λ *in einer Präfixformel* $\gamma : \lambda P$ *als Pfadselektor auf, so gibt es auf demselben Tableauzweig auch eine (aktuellste) Pfadinformationsformel für* λ *der Form* $\lambda : [\ldots, \gamma, \ldots]$.

6. *Ist* $\gamma : F$ *eine Präfixformel, so gibt es auf demselben Tableauzweig auch mindestens eine Pfadinformationsformel, in der* γ *als Präfix enthalten ist.*

7. *Ist* $\gamma \in \Gamma$ *auf einem Tableauzweig* T *vorkommendes Präfix, so gibt es jeweils genau eine Folge* $(\hat{0} = \gamma_0, \gamma_1, \ldots, \gamma_n = \gamma)$ *von Präfixen und* $(\lambda_1, \ldots, \lambda_n)$ *von Pfadbezeichnern, so daß es für alle* i *eine Pfadinformationsformel* $\lambda_i : [\ldots, \gamma_0, \ldots, \gamma_{i+1}, \ldots] \in T$ *gibt.*

Beweis: folgt nach Angabe der Tableauregeln, der Inhalt ist jedoch für das Verständnis der folgenden Abschnitte wichtig.

Für die weiteren Überlegungen ergeben sich daraus die folgenden Aussagen:

aus 1.: Die Begriffe „aktuellste Pfadinformationsformel für ein bestimmtes Pfadsymbol" und „... für einen bestimmten Pfadbezeichner" sind identisch.

aus 2.: Die Begriffe „der einem Präfixsymbol $\hat{\gamma}$ entsprechende Zustand" und „der einem Präfix γ entsprechende Zustand" sind identisch.

aus 3.: Die Lage eines Präfixsymbols auf einer Pfadinformationsformel ist eindeutig bestimmt.

aus 4.: Die Beschreibung der „geographischen Struktur" durch die Pfadinformationsformeln wird zunehmend detaillierter und die darin enthaltenen Informationen sind persistent.

aus 5. und 6.: Die benötigten Zusammenhänge von Pfaden und Zuständen bzw. Pfadbezeichnern und Präfixen sind verfügbar.

aus 7.: Die durch die Pfadinformationsformeln eines Tableauzweiges beschriebene „geographische Struktur" ist ein zusammenhängender Baum mit Wurzel $\hat{0}$.

Durch die Benennung neuer Zustände im Zuge der Konstruktion des Tableaux werden die einzelnen Pfade zunehmend detaillierter beschrieben. Damit muß als Prämisse bei der Anwendung einer Tableauregel jeweils die für das betreffende Pfadsymbol auf dem behandelten Tableauzweig *zuletzt erzeugte*, also detaillierteste Pfadinformationsformel verwendet werden.

Bei der Auflösung von temporallogischen Präfixformeln ist die Vorgehensweise immer dieselbe:
Es wird jeweils ein Pfadquantor/-selektor gemeinsam mit dem darauffolgenden Modaloperator aufgelöst.

- Bei der Auflösung von Formeln der Form $\gamma : \mathsf{E}P$ wird entsprechend den vorausgegangenen Überlegungen ein Pfad neu benannt.
- Formeln der Form $\gamma : \mathsf{A}P$ werden für jede Pfadinformationsformel, die das Präfix γ enthält, einzeln betrachtet.
- Formeln der Form $\gamma : \lambda P$ werden entlang der Pfadinformationsformel für λ aufgelöst.

Wie bei der Konstruktion von Tableaukalkülen üblich, wird dabei die Forderung, daß in dem durch das aktuelle Präfix beschriebenen Zustand eine bestimmte Formel gelten soll, in einfachere Teilaussagen zerlegt.

In der vorliegenden Modellierung wird von Folgen von Zuständen, zwischen denen im Tableau (noch) keine Unterschiede bekannt sind, abstrahiert. Der durch die Pfadinformationsformel, entlang der die Formel angewendet werden soll, beschriebene Pfad wird bis zu dem nächsten auf ihm enthaltenen Zustand, der sich von seinem Vorgänger unterscheidet – also in gewisser Weise „relevant“ ist – nach folgender Fragestellung betrachtet:

- Welche Formeln müssen im aktuellen Zustand gelten ?
- Was muß im nächsten relevanten Zustand gelten ?
- Was muß in allen dazwischen liegenden Zuständen gelten ?

Entsprechend ihrer Semantik wird die aufzulösende Formel bis zu dem nächsten auf dem durch die Pfadinformationsformel beschriebenen Pfad relevanten Zustand „fortgepflanzt“, indem die diesen Pfadabschnitt betreffenden Anforderungen formuliert werden, sowie die „Restforderung“ in dem den Abschnitt beendenden Zustand aufgestellt wird.

Aufgrund ihres Fortpflanzungsverhaltens werden die Pfadformeln in 4 Klassen eingeteilt (als Fortsetzung der Einteilung der prädikatenlogischen Formeln in α-, β-, γ- und δ-Formeln):

μ: Diese Formeln machen nur eine Aussage über den nächsten Zustand – im Vergleich zu der unendlichen Struktur ein sehr kleiner (μ-)Schritt ($P = \circ F$, $P = \neg\circ F$). Eine solche Formel pflanzt sich entlang eines Pfades genau einmal fort.

ν: Diese Formeln machen Aussagen über alle Nachfolgezustände entlang eines Pfades ($P = \Box F$, $P = \neg\Diamond F$). Eine solche Formel pflanzt sich entlang eines Pfades bis zu seinem Ende fort.

π: Diese Formeln machen eine Aussage über einen endlichen Abschnitt eines Pfades. In dem letzten Zustand dieses Abschnitts muß eine bestimmte Bedingung erfüllt sein ($P = \Diamond F$, $P = \neg\Box F$, $P = (F \;\mathsf{until}\; G)$ und $P = \neg(F \;\mathsf{unless}\; G)$). Die Fortpflanzung einer solchen Formel entlang eines Pfades muß vor dem Ende des Pfades beendet werden.

ρ: Diese Formeln sind eine Mischform aus ν- und π-Formeln: Es gibt zwei Alternativen, sie zu erfüllen: Eine ν- und eine π-Formel ($P = \neg(F \;\mathsf{until}\; G)$, $P = (F \;\mathsf{unless}\; G)$).

4 Tableaukalkül für CTL

In diesem Abschnitt wird, basierend auf den vorangegangenen Überlegungen, ein Tableaukalkül für CTL *entworfen.*

In CTL ist die oben beschriebene Vorgehensweise, jeweils einen Pfadquantor/-selektor gemeinsam mit dem darauffolgenden Modaloperator aufzulösen, problemlos möglich, da auf einen Pfadquantor immer direkt ein Modaloperator folgt und die auf diesen folgende Teilformel eine Zustandsformel ist. Damit wird deutlich, daß sich das Verfahren nicht direkt auf CTL^+ und CTL^* übertragen läßt.

4.1 Fortpflanzungssätze für CTL

Im weiteren ist die Frage interessant, ob die ausschließliche Verwendung von vollständigen Pfaden notwendig ist, oder ob es genügt, einen Pfad ab dem Zustand zu betrachten, in dem er sich explizit (d.h. durch die Eigenschaft, die seine Erzeugung erfordert) von den anderen beschriebenen Pfaden unterscheidet. Die Verwendung vollständiger Pfade erfordert einen signifikant höheren Verwaltungsaufwand: Wird auf einem Pfadabschnitt ein neuer Zustand benannt, muß diese Information allen Pfaden, die diesen Abschnitt umfassen, mitgeteilt werden – d.h. die entsprechenden aktualisierten Pfadinformationsformeln erzeugt werden.

Zur Beantwortung dieser Frage muß betrachtet werden, ob die Fortpflanzung von Formeln in abzweigende Pfade korrekt (bzw. vollständig) erfolgt. Dabei sind E-Formeln offensichtlich unproblematisch: Wenn eine E-Formel aufgelöst wird, wird angenommen, daß der erfüllende Pfad von allen anderen beschriebenen Pfaden verschieden ist, und ein neuer Pfad benannt, der diese Formel erfüllen soll. Damit wird dieser genau ab diesem Zustand getrennt betrachtet. Da die Formel dann genau für diesen Pfad gefordert wird, muß sie sich nicht in eventuelle Abzweige fortpflanzen.
Komplizierter ist die Situation bei A-Formeln: Sie müssen entlang allen – evtl. erst viel später benannten, durch den betrachteten Zustand verlaufenden – Pfaden gelten. Es ist also die Frage, inwieweit *ein* entlang eines festen Abschnittes verlaufender Pfad für alle anderen diesen Abschnitt beinhaltenden Pfade repräsentativ ist.

Die Grundlage bilden dabei die Äquivalenzen (F und G CTL-Zustandsformeln)

$$
\begin{aligned}
&\mathsf{A}\Box F \;\Leftrightarrow\; F \wedge \mathsf{A}\circ\mathsf{A}\Box F \quad ,\\
&\mathsf{E}\Box F \;\Leftrightarrow\; F \wedge \mathsf{E}\circ\mathsf{E}\Box F \quad ,\\
&\mathsf{A}\Diamond F \;\Leftrightarrow\; F \vee \mathsf{A}\circ\mathsf{A}\Diamond F \quad ,\\
&\mathsf{E}\Diamond F \;\Leftrightarrow\; F \vee \mathsf{E}\circ\mathsf{E}\Diamond F \quad ,\\
&\mathsf{A}(F \;\mathsf{until}\; G) \;\Leftrightarrow\; G \vee ((F \wedge \neg G) \wedge \mathsf{A}\circ\mathsf{A}(F \;\mathsf{until}\; G)) \quad ,\\
&\mathsf{E}(F \;\mathsf{until}\; G) \;\Leftrightarrow\; G \vee ((F \wedge \neg G) \wedge \mathsf{E}\circ\mathsf{E}(F \;\mathsf{until}\; G)) \quad .
\end{aligned}
$$

Sei in den folgenden Sätzen $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ eine Kripke-Struktur, $g \in \boldsymbol{G}$ und χ eine Belegung der vorkommenden freien Variablen.

Satz 4 *Für jede* CTL-*Zustandsformel* $\mathsf{A}P$*, wobei* P *eine* ν*-,* π*- oder* ρ*-Formel ist, gibt es* CTL-*Zustandsformeln* P_0*,* P_1 *und* P_2 *mit*

(1) $(g,\chi) \models P_0 \;\Rightarrow\; (g,\chi) \models \mathsf{A}P$,

(2) $(g,\chi) \models (\mathsf{A}P \wedge P_1) \;\Rightarrow\; (g,\chi) \models \mathsf{A}\circ\mathsf{A}P$,

(3) *Gilt* $(g,\chi) \models \mathsf{A}P$ *und* $(g,\chi) \not\models P_0$*, so gilt* $(g,\chi) \models P_1$ *und* $(g,\chi) \models P_2$ *und auf allen Pfaden* $p = (\ldots,g,\ldots)$ *gilt für jeden Folgezustand* h $(h,\chi) \models P_2 \wedge \mathsf{A}P$*, bis ein Zustand* k *erreicht wird, für den* $(k,\chi) \models P_0$ *gilt.*

Dabei ist (3) *absichtlich etwas umgangssprachlich formuliert, um die zugrundeliegende Überlegung auszudrücken. Insbesondere spielt es keine Rolle, ob ein solches* k *je erreicht wird.*

Beweis: Setze P_i folgendermaßen:

P	$\Box F$	$\neg\Diamond F$	$\Diamond F$	$\neg\Box F$	$(F \text{ until } G)$	$\neg(F \text{ unless } G)$	$(F \text{ unless } G)$	$\neg(F \text{ until } G)$
P_0	false	false	F	$\neg F$	G	$\neg F \wedge \neg G$	G	$\neg F \wedge \neg G$
P_1	true	true	$\neg F$	F	$\neg G$	F	$\neg G$	F
P_2	F	$\neg F$	$\neg F$	F	$F \wedge \neg G$	$F \wedge \neg G$	$F \wedge \neg G$	$F \wedge \neg G$

Der $\Diamond$-Operator wird dabei so interpretiert, daß der beschriebene erreichbare Zustand der erste ist, in dem die betreffende Formel erfüllt ist. Dies ermöglicht eine unproblematische Durchführung von Induktionsbeweisen (vgl. Beispiele).

In allen Fällen folgen die Eigenschaften (1) - (3) direkt aus der Semantik der Modaloperatoren bzw. den o.g. Äquivalenzen. □

Das Interessante an diesem Satz sind somit auch nicht die Eigenschaften (1) - (3), sondern daß eine solche Formulierung für CTL unter ausschließlicher Verwendung von *Zustandsformeln* möglich ist (für CTL* ist dies nicht möglich).

Dabei haben die Formeln P_0, P_1 und P_2 die folgenden Eigenschaften:
P_0 beendet die Fortpflanzung von P entlang eines Pfades, (daher ist bei ν-Formeln $P_0 = \mathsf{false}$)
P_1 erzwingt die Fortpflanzung von P entlang eines Pfades,
P_2 beschreibt das „Durchlaufen“ der Formel $\mathsf{A}P$ durch einen Zustand.

Mit denselben P_i gelten entsprechende Aussagen auch für Formeln der Form λP:

Satz 5 *Sei* $p = (g_0, g_1, \ldots) = \Phi(\lambda,\chi) \in \boldsymbol{P}(\boldsymbol{K})$*. Für jede* CTL-*Zustandsformel* λP*, wobei* P *eine* ν*-,* π*- oder* ρ*-Formel ist, gibt es* CTL-*Zustandsformeln* P_0*,* P_1 *und* P_2 *mit*

(1) $(g_i,\chi) \models P_0 \;\Rightarrow\; (g_i,\chi) \models \lambda P$,

(2) $(g_i,\chi) \models (\lambda P \wedge P_1) \;\Rightarrow\; (g_{i+1},\chi) \models \lambda P$,

(3) *Gilt* $(g_i,\chi) \models \lambda P$ *und* $(g_i,\chi) \not\models P_0$*, so gilt* $(g_i,\chi) \models P_1$ *und* $(g_i,\chi) \models P_2$ *und für jeden Folgezustand* h *entlang* $\Phi(\lambda,\chi)$ *gilt* $(h,\chi) \models P_2[\wedge\, \lambda P]$*, bis ein Zustand* k *erreicht wird, für den* $(k,\chi) \models P_0$ *gilt.*

Korollar 2 *Es gilt dabei i.a. nicht* $P_0 \leftrightarrow \neg P_2$*, sondern nur* $P_2 \rightarrow \neg P_0$*. Weiter gilt* $\mathsf{A}P \rightarrow ((P_0 \vee P_1) \wedge \neg(P_0 \wedge P_1) \wedge (\neg P_2 \leftrightarrow P_0))$ *und* $\lambda P \rightarrow ((P_0 \vee P_1) \wedge \neg(P_0 \wedge P_1) \wedge (\neg P_2 \leftrightarrow P_0))$.
Damit ist im Fall $(g,\chi) \models \mathsf{A}P$ *bzw.* $(g,\chi) \models \lambda P$ *eine disjunkte Fallunterscheidung nach* $(g,\chi) \models P_0$ *oder* $(g,\chi) \models P_1$ *möglich.*

Die P_i bilden die Grundlage für eine uniforme Behandlung der Pfadformeln, die Anforderungen an die Repräsentation der Pfade sowie das Vorgehen bei der Auflösung von Präfixformeln.

Die μ-Formeln weichen von dem Schema ab, da ihre Fortpflanzung genau im nächsten Zustand endet:

Satz 6 *Für jede* CTL-*Zustandsformel* $\mathsf{A}P$*, wobei* P *eine* μ*-Formel ist, gibt es eine* CTL-*Zustandsformel* P_0*, so daß aus* $g \models \mathsf{A}P$ *folgt, daß auf allen durch* g *verlaufenden Pfaden in dem unmittelbaren Nachfolgerzustand* P_0 *gilt.*
Analog für λP*.*

Beweis: Für $P = \circ F$ setze $P_0 = F$, für $P = \neg\circ F$ setze $P_0 = \neg F$. □

Mit diesen Sätzen werden alle möglichen mit einem universellen Pfadquantor beginnenden CTL-Zustandsformeln erfaßt.

Damit kann man den im vorangegangenen Abschnitt intuitiv beschriebenen Begriff eines „relevanten“ Zustandes formal definieren. Dabei wird offensichtlich, daß damit nicht nur ein Zustand, sondern eine Entfernung entlang des betrachteten Pfades beschrieben wird. Falls der “relevante“ Zustand in der Folge mehrmals durchlaufen wird, ist der erste Durchlauf „relevant“. Dies entspricht der Intuition, die erste Veränderung bezüglich der aktuellen Fragestellung zu betrachten.

Definition 29 *Ist* T *ein Tableauzweig,* $(\boldsymbol{K},\Omega,\chi) \not\models T$*,* $\gamma_i : \lambda P$ *oder* $\gamma_i : \mathsf{A}P \in T$ *die im nächsten Tableauschritt entlang* λ *aufzulösende* $\mathcal{TK}$*-Präfixformel,* $I = \lambda : [\gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}] \in T$ *die neueste Pfadinformationsformel für* λ *auf* T*.*
Ist P *eine* ν*-,* π*- oder* ρ*-Formel, so ist*

1. $\Phi(\lambda,\chi,\Pi(\lambda,\gamma_i,\chi)) = \Psi(\gamma_i,\chi)$ *der* nächste relevante Zustand*, falls* $(\Psi(\gamma_i,\chi),\chi) \models P_0$*,*
2. $\Phi(\lambda,\chi,k)$ *mit* $\Pi(\lambda,\gamma_i,\chi) < k < \Pi(\lambda,\gamma_{i+1},\chi)$ *der* nächste relevante Zustand*, falls für alle* $j : \Pi(\lambda,\gamma_i,\chi) \leq j < k : (\Phi(\lambda,\chi,j),\chi) \models P_1$ *und* $(\Phi(\lambda,\chi,k),\chi) \models P_0$*.*
3. $\Phi(\lambda,\chi,\Pi(\lambda,\gamma_{i+1},\chi)) = \Psi(\gamma_{i+1},\chi)$ *der* nächste relevante Zustand*, falls* $i \neq n$ *und für alle* $j : \Pi(\lambda,\gamma_i,\chi) \leq j < \Pi(\lambda,\gamma_{i+1},\chi) : (\Phi(\lambda,\chi,j),\chi) \models P_1$*.*
4. ∞ *der* nächste relevante Zustand*, falls* $i = n$ *und für alle* $j > \Pi(\lambda,\gamma_n,\chi) : (\Phi(\lambda,\chi,j),\chi) \models P_1$*.*
   Die Tatsache, daß $\infty \notin \boldsymbol{G}$ *kein Zustand im eigentlichen Sinne ist, stellt beim weiteren Vorgehen kein Problem dar.*

Ist P *eine* μ*-Formel, so ist* $\Phi(\lambda,\chi,\Pi(\lambda,\gamma_i,\chi)+1)$ *der* nächste relevante Zustand*.*

Korollar 3 *In der Situation von Definition 29 gilt:*

- *Ist P eine π-Formel, so ist der nächste relevante Zustand nicht ∞.*
- *Ist $i \neq n$, so liegt der nächste relevante Zustand auf $\Phi(\lambda,\chi)$ zwischen $\Phi(\lambda,\chi,\Pi(\lambda,\gamma_i,\chi))$ und $\Phi(\lambda,\chi,\Pi(\lambda,\gamma_{i+1},\chi))$ (einschließlich).*

Aufgrund der disjunkten Fallunterscheidung $\mathsf{A}P \rightarrow ((P_0 \vee P_1) \wedge \neg(P_0 \wedge P_1))$ ist der Begriff des nächsten relevanten Zustandes bezüglich einer Belegung χ, eines Präfixes γ, einer Präfixformel $\gamma : \mathsf{A}P$ bzw $\gamma : \lambda P$ und einem Pfadbezeichner λ auf einem Tableauzweig T wohldefiniert in $\boldsymbol{G} \cup \{\infty\}$, wenn $(\boldsymbol{K},\Omega,\chi) \not\models T$ vorausgesetzt wird.

Korollar 4 *Ist bezüglich einem Pfadbezeichner λ mit entsprechender Pfadinformationsformel, einem in dieser vorkommenden Präfix γ, $m := \Pi(\lambda,\gamma,\chi)$, einer $\mathcal{TK}$-Präfixformel $\gamma : \mathsf{A}P$ oder $\gamma : \lambda P$ und einer Belegung χ der nächste relevante Zustand bezüglich einer ν-, π- oder ρ-Formel als $\Phi(\lambda,\chi,k) \neq \infty$ gegeben, so ist für alle $j : m \leq j < k$: $\Phi(\lambda,\chi,j) \neq \Phi(\lambda,\chi,k)$.*

Beweis: Folgt aus obiger Definition (Anm.: Falls $k = m$ ist, gibt es kein solches j). □

Aufgrund dieses Korollars kann von einem „nächsten relevanten Zustand“ g entlang eines Pfades p gesprochen werden, in dem Sinne, daß damit wirklich das nächste Durchlaufen von p durch g gemeint ist.

Zur Untersuchung der diesen Abschnitt motivierenden Fragestellung geht man im weiteren von einer Tableauprozedur und einem Ω aus, die die folgenden Eigenschaften erfüllen:

- alle Pfadinformationsformeln beschreiben ausschließlich vollständige Pfade, d.h. alle Pfadinformationsformeln sind von der Form $I = \lambda : [\hat{0}, \ldots]$,
- Sind $I = \lambda : [\hat{0}, I', \gamma, \ldots]$ und $J = \kappa : [\hat{0}, J', \gamma, \ldots]$ die aktuellen Pfadinformationsformeln für λ und κ auf demselben Tableauzweig T, so ist $\gamma \neq \infty$, außerdem
- $I' = J'$ und es gibt ein k, so daß γ in beiden Pfadinformationsformeln an k-ter Stelle steht und
- für alle $(\boldsymbol{K},\Omega,\chi) \not\models T$ ist für alle $i : 0 \leq i \leq k : \Pi(\lambda,\gamma_i,\chi) = \Pi(\kappa,\gamma_i,\chi)$ und für alle $j : 0 \leq j \leq \Pi(\lambda,\gamma_k,\chi)$ gilt $\Phi(\lambda,\chi,j) = \Phi(\kappa,\chi,j)$, d.h. die Pfade $\Phi(\lambda,\chi)$ und $\Phi(\kappa,\chi)$ verlaufen in diesem Abschnitt parallel.

Eine ähnliche Tableauprozedur ist in Abschnitt 10 beschrieben.

Satz 7 *Sei T ein Tableauzweig, $(\boldsymbol{K},\Omega,\chi) \not\models T$, $\gamma_i \neq \gamma_j \in \Gamma$, $\lambda,\kappa \in \Lambda$ mit den Pfadinformationsformeln $\lambda : [\hat{0},\ldots,\gamma_i,\ldots,\gamma_j,I_1] \in T$ bzw. $\kappa : [\hat{0},\ldots,\gamma_i,\ldots,\gamma_j,I_2] \in T$, $F = \gamma_i : \mathsf{A}P \in T$ eine $\mathcal{TK}$-Präfixformel. Dann ist der bzgl. λ, χ und F nächste relevante Zustand gleich dem bzgl. κ, χ und F nächsten relevanten Zustand, und dieser ist nicht ∞.*

Beweis:
Nach Voraussetzung über die Tableauprozedur ist $\gamma_j \neq \infty$.
Es wird eine Fallunterscheidung nach dem bezüglich λ, γ_i, F und χ nächsten relevanten Zustand durchgeführt. Dabei wird ausgenutzt, daß die P_i Zustandsformeln sind.

P eine ν-, π- oder ρ-Formel:

- $\Phi(\lambda, \chi, \Pi(\lambda, \gamma_i, \chi)) = \Psi(\gamma_i, \chi)$ der nächste bezüglich λ relevante Zustand:
  Nach Definition 29 muß also $(\Psi(\gamma_i, \chi), \chi) \models P_0$ gelten. Dann ist bezüglich κ ebenfalls $\Psi(\gamma_i, \chi) = \Phi(\kappa, \chi, \Pi(\kappa, \gamma_i, \chi))$ der nächste relevante Zustand.
- $\Phi(\lambda, \chi, k)$ mit $\Pi(\lambda, \gamma_i, \chi) < k < \Pi(\lambda, \gamma_{i+1}, \chi)$ der nächste bezüglich λ relevante Zustand:
  Dann gilt nach Definition 29 und der Voraussetzung über die Tableauprozedur
  für alle n: $\Pi(\lambda, \gamma_i, \chi) \leq n < k : (\Phi(\lambda, \chi, n), \chi) = (\Phi(\kappa, \chi, n), \chi) \models P_1$ und $(\Phi(\lambda, \chi, k), \chi) = (\Phi(\kappa, \chi, k), \chi) \models P_0$. Damit ist wegen $k \leq \Pi(\lambda, \gamma_{i+1}, \chi) = \Pi(\kappa, \gamma_{i+1}, \chi)$ auch $\Phi(\kappa, \chi, k) = \Phi(\lambda, \chi, k)$ der nächste relevante Zustand bezüglich κ.
- $\Phi(\lambda, \chi, \Pi(\lambda, \gamma_{i+1}, \chi)) = \Psi(\gamma_{i+1}, \chi)$ der nächste relevante Zustand bezüglich λ:
  Dann gilt nach Definition 29 und der Voraussetzung über die Tableauprozedur
  für alle n: $\Pi(\lambda, \gamma_i, \chi) \leq n < \Pi(\lambda, \gamma_{i+1}, \chi) = \Pi(\lambda, \gamma_{i+1}, \chi) : (\Phi(\lambda, \chi, n), \chi) = (\Phi(\kappa, \chi, n), \chi) \models P_1$,
  womit $\Psi(\gamma_{i+1}, \chi)$ bezüglich κ ebenfalls der nächste relevante Zustand ist.

P eine μ-Formel: Es ist $\Phi(\lambda, \chi, \Pi(\lambda, \gamma_i, \chi)+1)$ der nächste relevante Zustand bezüglich λ. Wegen $\gamma_i \neq \gamma_j$ ist nach der Voraussetzung über die Tableauprozedur $\Phi(\lambda, \chi, \Pi(\lambda, \gamma_i, \chi)+1) = \Phi(\kappa, \chi, \Pi(\kappa, \gamma_i, \chi)+1)$ und damit ebenfalls nächster relevanter Zustand bezüglich κ. □

Satz 8 (CTL**-Fortpflanzungssatz)** *Unter denselben Voraussetzungen wie im vorhergehenden Satz wird die Fragestellung von Seite 43 bei Betrachtung von F entlang λ und κ exakt gleich beantwortet.*

Beweis:
Mit der auf Definition 29 beruhenden Fallunterscheidung liefern die Sätze 4 und 6 die folgenden Antworten:

P eine ν-, π- oder ρ-Formel:

- $\Phi(\lambda, \chi, \Pi(\lambda, \gamma_i, \chi)) = \Phi(\kappa, \chi, \Pi(\kappa, \gamma_i, \chi)) = \Psi(\gamma_i, \chi)$ der nächste relevante Zustand:
  $(\Psi(\gamma_i, \chi), \chi) \models P_0$.
- $\Phi(\lambda, \chi, k) = \Phi(\kappa, \chi, k)$ mit $\Pi(\lambda, \gamma_i, \chi) < k < \Pi(\lambda, \gamma_{i+1}, \chi)$ der nächste relevante Zustand:
  Für alle $n : \Pi(\lambda, \gamma_i, \chi) \leq n < k : (\Phi(\lambda, \chi, n), \chi) = (\Phi(\kappa, \chi, n), \chi) \models P_2 \wedge \mathsf{A}P$ und $(\Phi(\lambda, \chi, k), \chi) = (\Phi(\kappa, \chi, k), \chi) \models P_0$.
- $\Phi(\lambda, \chi, \Pi(\lambda, \gamma_{i+1}, \chi)) = \Phi(\kappa, \chi, \Pi(\kappa, \chi, \gamma_{i+1}, \chi)) = \Psi(\gamma_{i+1}, \chi)$ der nächste relevante Zustand:
  Für alle $n : \Pi(\lambda, \gamma_i, \chi) \leq j < \Pi(\lambda, \gamma_{i+1}, \chi) : (\Phi(\lambda, \chi, n), \chi) = (\Phi(\kappa, \chi, n), \chi) \models P_2 \wedge \mathsf{A}P$ und $(\Psi(\gamma_{i+1}, \chi), \chi) = (\Phi(\lambda, \chi, \Pi(\lambda, \gamma_{i+1}, \chi)), \chi) = (\Phi(\kappa, \chi, \Pi(\kappa, \gamma_{i+1}, \chi)), \chi) \models \mathsf{A}P$.
- Nach dem vorhergehenden Satz ist der nächste relevante Zustand nicht ∞.

P eine μ-Formel: $(\Phi(\lambda, \chi, \Pi(\lambda, \gamma_i, \chi) + 1), \chi) = (\Phi(\kappa, \chi, \Pi(\kappa, \gamma_i, \chi) + 1), \chi) \models P_0$. □

Damit genügt es, entlang eines Pfadabschnitts einen Pfad repräsentativ für alle anderen, dort parallel verlaufenden Pfade zu untersuchen. Die Konsequenz ist, parallel verlaufende Pfadabschnitte in dem Kalkül für CTL nicht mehrfach zu repräsentieren, sondern verschiedene Pfade erst ab dem Zustand zu behandeln, in dem sie eigene Eigenschaften besitzen, die sie von anderen Pfaden unterscheiden. Dies sind genau diejenigen Eigenschaften, die durch E-Formeln beschrieben werden, womit also der entsprechende

Pfad als Abzweig angesehen werden kann. Damit beschreibt ein Tableau i.a. keine vollständigen Pfade, sondern jeder Pfad wird als in einem Zustand von einem anderen abzweigender (nicht vollständiger) Pfad betrachtet (vollständige Pfade sind dann nur diejenigen, die bereits im Startzustand durch Auflösen eines E-Quantors entstehen).

Die im Beweis als „Antworten“ angegebenen Modellbeziehungen bilden die Grundlage für die Konstruktion der Tableauregeln. Dieses Ergebnis formuliert – auch für Formeln der Form λP – der folgende Satz:

Satz 9 *Sei T ein Tableauzweig, $(\boldsymbol{K}, \Omega, \chi) \not\models T$, $\lambda : [\hat{0}, \ldots, \gamma_i, \ldots] \in T$ die aktuellste Pfadinformationsformel für $\lambda \in \Lambda$, P eine $\mathcal{TK}$-Pfadformel und $\gamma_i : \mathsf{A}P$ bzw. $\gamma_i : \lambda P$ die im nächsten Schritt entlang λ aufzulösende $\mathcal{TK}$-Präfixformel auf T. Dann wird die Fragestellung von Seite 43 folgendermaßen beantwortet:*

$\gamma_i : \mathsf{A}P$: *P eine ν-, π- oder ρ-Formel:*

- $\Phi(\lambda, \chi, \Pi(\lambda, \gamma_i, \chi)) = \Psi(\gamma_i, \chi)$ *der nächste relevante Zustand:*
  $(\Psi(\gamma_i, \chi), \chi) \models P_0$.
- $\Phi(\lambda, \chi, k)$ *mit* $\Pi(\lambda, \gamma_i, \chi) < k < \Pi(\lambda, \gamma_{i+1}, \chi)$ *der nächste relevante Zustand:*
  Für alle $n : \Pi(\lambda, \gamma_i, \chi) \leq n < k : (\Phi(\lambda, \chi, n), \chi) \models P_2 \wedge \mathsf{A}P$ *und* $(\Phi(\lambda, \chi, k), \chi) \models P_0$.
- $\Phi(\lambda, \chi, \Pi(\lambda, \gamma_{i+1}, \chi)) = \Psi(\gamma_{i+1}, \chi)$ *der nächste relevante Zustand:*
  Für alle $n : \Pi(\lambda, \gamma_i, \chi) \leq j < \Pi(\lambda, \gamma_{i+1}, \chi) : (\Phi(\lambda, \chi, n), \chi) \models P_2 \wedge \mathsf{A}P$ *und*
  $(\Psi(\gamma_{i+1}, \chi), \chi) \models \mathsf{A}P$.
- ∞ *der nächste relevante Zustand: Für alle* $n > \Pi(\lambda, \gamma_i, \chi) : (\Phi(\lambda, \chi, n), \chi) \models P_2 \wedge \mathsf{A}P$.

P eine μ-Formel: $(\Phi(\lambda, \chi, \Pi(\lambda, \gamma_i, \chi) + 1), \chi) \models P_0$.

$\gamma_i : \lambda P$: *P eine ν-, π- oder ρ-Formel:*

- $\Phi(\lambda, \chi, \Pi(\lambda, \gamma_i, \chi)) = \Psi(\gamma_i, \chi)$ *der nächste relevante Zustand:*
  $(\Psi(\gamma_i, \chi), \chi) \models P_0$.
- $\Phi(\lambda, \chi, k)$ *mit* $\Pi(\lambda, \gamma_i, \chi) < k < \Pi(\lambda, \gamma_{i+1}, \chi)$ *der nächste relevante Zustand:*
  Für alle $n : \Pi(\lambda, \gamma_i, \chi) \leq n < k : (\Phi(\lambda, \chi, n), \chi) \models P_2$ *und* $(\Phi(\lambda, \chi, k), \chi) \models P_0$.
- $\Phi(\lambda, \chi, \Pi(\lambda, \gamma_{i+1}, \chi)) = \Psi(\gamma_{i+1}, \chi)$ *der nächste relevante Zustand:*
  Für alle $n : \Pi(\lambda, \gamma_i, \chi) \leq j < \Pi(\lambda, \gamma_{i+1}, \chi) : (\Phi(\lambda, \chi, n), \chi) \models P_2$ *und* $(\Psi(\gamma_{i+1}, \chi), \chi) \models \lambda P$.
- ∞ *der nächste relevante Zustand: Für alle* $n > \Pi(\lambda, \gamma_i, \chi) : (\Phi(\lambda, \chi, n), \chi) \models P_2$.

P eine μ-Formel: $(\Phi(\lambda, \chi, \Pi(\lambda, \gamma_i, \chi) + 1), \chi) \models P_0$.

Beweis: Für $\mathsf{A}P$ Satz 8, für λP die Sätze 4 und 6 analog zu Beweis von Satz 8. Der Fall, daß der nächste relevante Zustand ∞ ist, folgt aus Definition 29 und Satz 4. □

4.2 Der Tableaukalkül zu CTL – theoretisch

Der bei der Konstruktion von Tableaukalkülen üblichen Vorgehensweise folgend liegt den angegebenen Tableauregeln jeweils eine Zerlegung der $\not\models$-Relation zugrunde. Die Zerlegung basiert dabei auf den P_i

der Fortpflanzungssätze. Die Prämisse besteht in den meisten Fällen aus der aufzulösenden Präfixformel und einer Pfadinformationsformel, die das betreffende Präfix enthält.

Um zum besseren Verständnis bereits dem Korrektheitssatz vorzugreifen, gilt dabei:
Ist $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ eine Kripke-Struktur über einer Signatur Σ, $\Omega = (\Phi, \Pi, \Psi)$ eine P&P-Interpretation der Mengen $\hat{\Gamma}$ und $\hat{\Lambda}$ und χ eine Belegung der (einschließlich der gewählten Konklusion) auftretenden freien Variablen und gilt

$$(\boldsymbol{K}, \Omega, \chi) \models \text{Prämisse} \quad ,$$

so ist

$$(\boldsymbol{K}', \Omega', \chi) \models \text{eine der möglichen Konklusionen} \quad ,$$

wobei $\boldsymbol{K}'$ eine Kripke-Struktur über Σ' und Ω' eine P&P-Interpretation von $\hat{\Gamma}'$ und $\hat{\Lambda}'$ ist und die folgenden Bedingungen erfüllt sind:

$\Sigma' \supseteq \Sigma, \quad \hat{\Gamma}' \supseteq \hat{\Gamma}, \quad \hat{\Lambda}' \supseteq \hat{\Lambda}$,
$\boldsymbol{K}' = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M}')$,
für alle $g \in \boldsymbol{G}$: $M'(g) \mid_{\Sigma} = M(g)$,
(Erweiterung um die Interpretation einer Skolemfunktion)
$\Omega' = (\Phi', \Pi', \Psi')$,
$\Phi' \mid_{\Lambda} = \Phi$,
(Erweiterung um die Interpretation eines Pfadsymbols)
$\Pi' \mid_{\Lambda \times \Gamma} = \Pi$,
(Erweiterung um die Interpretation eines Pfad- oder Präfixsymbols)
$\Psi' \mid_{\Gamma} = \Psi$
(Erweiterung um die Interpretation eines Präfixsymbols)

Dabei muß $\boldsymbol{K}$ (bzw. $\boldsymbol{M}$) genau dann erweitert werden, wenn die δ-Regel angewendet wird (Einführung einer Skolemfunktion). Die Menge der freien Variablen wird genau dann erweitert, wenn die γ-Regel angewendet wird. Für die Auflösung der Modaloperatoren muß $\boldsymbol{K}$ nicht modifiziert werden. Ω wird genau dann erweitert, wenn ein neues Präfix oder ein neuer Pfadbezeichner eingeführt wird.

Die Auflösung der verschiedenen Pfadquantoren und -selektoren basiert auf Definition 28 und Korollar 1:

$\gamma : \mathsf{E}P$: nach Definition 28 ist

$(\boldsymbol{K}, \Omega, \chi) \Vdash\!\!\!\Vdash \gamma : \mathsf{E}P \;\Leftrightarrow$ Es gibt einen Pfad $p(\chi) = (g_0, g_1, \ldots)$ in $\boldsymbol{K}$ und ein $n(\chi)$ mit $g_{n(\chi)} = \Psi(\gamma, \chi)$ und $(p(\chi)|_{n(\chi)}, \chi) \models P$.
Nach den Betrachtungen des letzten Abschnittes wird dieser Pfad erst ab dem Zustand γ betrachtet:

$\Leftrightarrow$ $(\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \hat{\kappa}(\mathsf{free}(T)) : [\gamma, \emptyset, \hat{\infty}]$ und
$(\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \gamma : \hat{\kappa}(\mathsf{free}(T))\ P$
für ein noch nicht vorkommendes Pfadsymbol $\hat{\kappa}$.
(jedes Ω' mit $\Phi' \supset \Phi \cup \{(\kappa, \chi) \mapsto p(\chi)|_{n(\chi)}\}$ und
$\Pi' \supset \Pi \cup \{(\kappa, \gamma, \chi) \mapsto 0\}$ erfüllt es)

Bei Auflösung eines E-Quantors wird dieser Pfad benannt, entlang dem die entsprechende Formel gelten soll:

$$\frac{\gamma : \mathsf{E}P}{\begin{array}{c}\hat{\kappa}(\mathsf{free}(T)) : [\gamma, \emptyset, \hat{\infty}] \\ \gamma : \hat{\kappa}(\mathsf{free}(T))\ P\end{array}}$$

wobei $\hat{\kappa}$ ein noch nicht vorkommendes Pfadsymbol ist.

Die $\mathsf{A}P$- und $\lambda\, P$-Formeln werden mit Hilfe der Pfadinformationsformeln aufgelöst:

$\gamma : \mathsf{A}P$: Aus Korollar 1 hat man

$(\boldsymbol{K}, \Omega, \chi) \Vdash\!\!\!\Vdash \gamma : \mathsf{A}P \;\Rightarrow$ Für jede Pfadinformationsformel $I = \lambda : [\ldots, \gamma, \ldots]$ gilt $(\Phi(\lambda, \chi)|_{\Pi(\lambda,\gamma,\chi)}, \chi) \models P$.

Damit werden A-Formeln entlang jeder das entsprechende Präfix enthaltenden Pfadinformationsformel aufgelöst.

$\gamma : \lambda P$: Nach Definition 28 ist

$(\boldsymbol{K}, \Omega, \chi) \Vdash\!\!\!\Vdash \gamma : \lambda P \;:\Leftrightarrow\; (\Phi(\lambda, \chi)|_{\Pi(\lambda,\gamma,\chi)}, \chi) \models P$.

Die λ-Formeln werden genau entlang der Pfadinformationsformel für den Pfadbezeichner λ aufgelöst.

In beiden Fällen geschieht dies gemäß den in Satz 9 formulierten Erkenntnissen.

Auf Basis der im Zuge der Fortpflanzungssätze beschriebenen Formeln P_0, P_1 und P_2 kann man die Tableauregeln in einem ersten Schritt abstrakt für die verschiedenen Klassen modallogischer Formeln konstruieren.
Im darauffolgenden Abschnitt werden die Regeln für die einzelnen Modalitäten angegeben.

Entsprechend der verschiedenen Möglichkeiten einer Auflösung einer Formel $\alpha : F$ entlang einem durch eine Pfadinformationsformel $I = \lambda : [\ldots, \alpha, \ldots]$ beschriebenen Pfad nach der beschriebenen Fragestellung erhält man für jede Formelklasse vier Regeln, von denen jeweils genau eine anwendbar ist:

- Eine Formel der Form $F = \mathsf{A}P$ soll aufgelöst werden

oder

- eine Formel der Form $F = \lambda P$ soll aufgelöst werden

und

- I ist von der Form $\lambda : [\ldots, \alpha, L, \beta, \ldots]$, $L \neq \circ$, d.h. auf α folgt ein Pfadabschnitt unbekannter Länge, repräsentiert durch eine Liste von Formeln,

oder

- I ist von der Form $\lambda : [\ldots, \alpha, \circ, \beta, \ldots]$, d.h. der Nachfolgezustand von α ist explizit benannt.

Sei im folgenden $\hat{\gamma}$ ein noch nicht verwendetes Präfixsymbol, T der betrachtete Tableauzweig. Es werden nur diejenigen Formeln betrachtet, die in Zuständen auf dem durch λ beschriebenen Pfad gelten. Daher sind die vorkommenden $\Leftrightarrow$-Beziehungen entsprechend eingeschränkt zu lesen.

μ:

$\mathsf{A}P_\mu$:

$$
\begin{array}{lcl}
(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : \mathsf{A}P, & \Leftrightarrow & (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,\circ,\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] \\
(\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ & & (\boldsymbol{K},\chi) \Vdash\!\!\!\models \hat{\gamma}(\mathsf{free}(T)) : L \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\!\models \hat{\gamma}(\mathsf{free}(T)) : P_0 \\
 & & \vee \\
 & & \text{falls } \beta \neq \hat{\circ}\text{:} \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : P_0
\end{array}
$$

$$
\begin{array}{lcl}
(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : \mathsf{A}P, & \Leftrightarrow & (\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : P_0 \\
(\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & &
\end{array}
$$

λP_μ:

$$
\begin{array}{lcl}
(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : \lambda P, & \Leftrightarrow & (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,\circ,\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] \\
(\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ & & (\boldsymbol{K},\chi) \Vdash\!\!\!\models \hat{\gamma}(\mathsf{free}(T)) : L \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\!\models \hat{\gamma}(\mathsf{free}(T)) : P_0 \\
 & & \vee \\
 & & \text{falls } \beta \neq \hat{\circ}\text{:} \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : P_0
\end{array}
$$

$$
\begin{array}{lcl}
(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : \lambda P, & \Leftrightarrow & (\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : P_0 \\
(\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & &
\end{array}
$$

Anforderungen an die Interpretation Ω':

$$\Psi' \supset \Psi \cup \{(\gamma,\chi) \mapsto \Phi(\lambda,\chi,\Pi(\lambda,\alpha,\chi)+1)\} \quad \text{und}$$

$$\Pi' \supset \Pi \cup \{(\lambda,\gamma,\chi) \mapsto \Pi(\lambda,\alpha,\chi)+1\}$$

$$
\frac{\begin{array}{c}\alpha : \mathsf{A}P \\ \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ\end{array}}{\begin{array}{l|c}\lambda : [\ldots,\alpha,\circ,\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] & \text{falls } \beta \neq \hat{\circ}\text{:} \\ \hat{\gamma}(\mathsf{free}(T)) : L & \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \\ \hat{\gamma}(\mathsf{free}(T)) : P_0 & \beta : P_0\end{array}}
\qquad
\frac{\begin{array}{c}\alpha : \mathsf{A}P \\ \lambda : [\ldots,\alpha,\circ,\beta,\ldots]\end{array}}{\beta : P_0}
$$

$$
\frac{\begin{array}{c}\alpha : \lambda P \\ \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ\end{array}}{\begin{array}{l|c}\lambda : [\ldots,\alpha,\circ,\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] & \text{falls } \beta \neq \hat{\circ}\text{:} \\ \hat{\gamma}(\mathsf{free}(T)) : L & \lambda : [\ldots,a,\circ,\beta,\ldots] \\ \hat{\gamma}(\mathsf{free}(T)) : P_0 & \beta : P_0\end{array}}
\qquad
\frac{\begin{array}{c}\alpha : \lambda P \\ \lambda : [\ldots,\alpha,\circ,\beta,\ldots]\end{array}}{\beta : P_0}
$$

ν:

$\mathsf{A}P_\nu$:

$$(\boldsymbol{K},\chi) \nVdash \alpha : \mathsf{A}P, \quad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L \cup \{P_2,\mathsf{A}P\},\beta,\ldots]$$
$$(\boldsymbol{K},\chi) \nVdash \alpha : P_2$$
$$(\boldsymbol{K},\chi) \nVdash \beta : \mathsf{A}P \quad \text{falls } \beta \neq \hat{\infty}$$

$$(\boldsymbol{K},\chi) \nVdash \alpha : \mathsf{A}P, \quad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \nVdash \alpha : P_2, \quad (\boldsymbol{K},\chi) \nVdash \beta : \mathsf{A}P$$

λP_ν:

$$(\boldsymbol{K},\chi) \nVdash \alpha : \lambda P, \quad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L \cup \{P_2\},\beta,\ldots]$$
$$(\boldsymbol{K},\chi) \nVdash \alpha : P_2$$
$$(\boldsymbol{K},\chi) \nVdash \beta : \lambda P \quad \text{falls } \beta \neq \hat{\infty}$$

$$(\boldsymbol{K},\chi) \nVdash \alpha : \lambda P, \quad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \nVdash \alpha : P_2, \quad (\boldsymbol{K},\chi) \nVdash \beta : \lambda P$$

Dabei sind keine Modifikationen an Ω notwendig.

$$\frac{\alpha : \mathsf{A}P \qquad \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ}{\lambda : [\ldots,\alpha,L \cup \{P_2,\mathsf{A}P\},\beta,\ldots] \qquad \alpha : P_2 \qquad \beta : \mathsf{A}P \quad \text{falls } \beta \neq \hat{\infty}} \qquad \frac{\alpha : \mathsf{A}P \qquad \lambda : [\ldots,\alpha,\circ,\beta,\ldots]}{\alpha : P_2 \qquad \beta : \mathsf{A}P}$$

$$\frac{\alpha : \lambda P \qquad \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ}{\lambda : [\ldots,\alpha,L \cup \{P_2\},\beta,\ldots] \qquad \alpha : P_2 \qquad \beta : \lambda P \quad \text{falls } \beta \neq \hat{\infty}} \qquad \frac{\alpha : \lambda P \qquad \lambda : [\ldots,\alpha,\circ,\beta,\ldots]}{\alpha : P_2 \qquad \beta : \lambda P}$$

π:

$\mathsf{A}P_\pi$:

$$
\begin{array}{lcl}
(\boldsymbol{K},\chi) \Vdash \alpha : \mathsf{A}P, & \Leftrightarrow & (\boldsymbol{K},\chi) \Vdash \alpha : P_0 \\
(\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ & & \vee \\
 & & (\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,L \cup \{P_2,\mathsf{A}P\},\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] \\
 & & (\boldsymbol{K},\chi) \Vdash \alpha : P_2 \\
 & & (\boldsymbol{K},\chi) \Vdash \hat{\gamma}(\mathsf{free}(T)) : L \\
 & & (\boldsymbol{K},\chi) \Vdash \hat{\gamma}(\mathsf{free}(T)) : P_0 \\
 & & \vee \\
 & & \text{falls } \beta \neq \hat{\infty}\text{:} \\
 & & (\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,L \cup \{P_2,\mathsf{A}P\},\beta,\ldots] \\
 & & (\boldsymbol{K},\chi) \Vdash \alpha : P_2 \\
 & & (\boldsymbol{K},\chi) \Vdash \beta : \mathsf{A}P \\
(\boldsymbol{K},\chi) \Vdash \alpha : \mathsf{A}P, & \Leftrightarrow & (\boldsymbol{K},\chi) \Vdash \alpha : P_0 \\
(\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & & \vee \\
 & & (\boldsymbol{K},\chi) \Vdash \alpha : P_2 \\
 & & (\boldsymbol{K},\chi) \Vdash \beta : \mathsf{A}P
\end{array}
$$

λP_π:

$$
\begin{array}{lcl}
(\boldsymbol{K},\chi) \Vdash \alpha : \lambda P, & \Leftrightarrow & (\boldsymbol{K},\chi) \Vdash \alpha : P_0 \\
(\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ & & \vee \\
 & & (\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,L \cup \{P_2\},\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] \\
 & & (\boldsymbol{K},\chi) \Vdash \alpha : P_2 \\
 & & (\boldsymbol{K},\chi) \Vdash \hat{\gamma}(\mathsf{free}(T)) : L \\
 & & (\boldsymbol{K},\chi) \Vdash \hat{\gamma}(\mathsf{free}(T)) : P_0 \\
 & & \vee \\
 & & \text{falls } \beta \neq \hat{\infty}\text{:} \\
 & & (\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,L \cup \{P_2\},\beta,\ldots] \\
 & & (\boldsymbol{K},\chi) \Vdash \alpha : P_2 \\
 & & (\boldsymbol{K},\chi) \Vdash \beta : \lambda P \\
(\boldsymbol{K},\chi) \Vdash \alpha : \lambda P, & \Leftrightarrow & (\boldsymbol{K},\chi) \Vdash \alpha : P_0 \\
(\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & & \vee \\
 & & (\boldsymbol{K},\chi) \Vdash \alpha : P_2 \\
 & & (\boldsymbol{K},\chi) \Vdash \beta : \lambda P
\end{array}
$$

Dabei wird ggf. Ω so ergänzt, daß $\hat{\gamma}(\chi(\mathsf{free}(T)))$ den passenden Zustand beschreibt (betrifft Ψ und Π).

$$\frac{\alpha : \mathsf{A}P \qquad \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ}{\begin{array}{r|l|c} \alpha : P_0 & \lambda : [\ldots, \alpha, L \cup \{P_2, \mathsf{A}P\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \text{falls } \beta \neq \hat{\infty}\text{:} \\ & \alpha : P_2 & \lambda : [\ldots, \alpha, L \cup \{P_2, \mathsf{A}P\}, \beta, \ldots] \\ & \hat{\gamma}(\mathsf{free}(T)) : L & \alpha : P_2 \\ & \hat{\gamma}(\mathsf{free}(T)) : P_0 & \beta : \mathsf{A}P \end{array}}$$

$$\frac{\alpha : \mathsf{A}P \qquad \lambda : [\ldots, \alpha, \circ, \beta, \ldots]}{\begin{array}{r|l} \alpha : P_0 & \alpha : P_2 \\ & \beta : \mathsf{A}P \end{array}}$$

$$\frac{\alpha : \lambda P \qquad \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ}{\begin{array}{r|l|c} \alpha : P_0 & \lambda : [\ldots, \alpha, L \cup \{P_2\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \text{falls } \beta \neq \hat{\infty}\text{:} \\ & \alpha : P_2 & \lambda : [\ldots, \alpha, L \cup \{P_2\}, \beta, \ldots] \\ & \hat{\gamma}(\mathsf{free}(T)) : L & \alpha : P_2 \\ & \hat{\gamma}(\mathsf{free}(T)) : P_0 & \beta : \lambda P \end{array}}$$

$$\frac{\alpha : \lambda P \qquad \lambda : [\ldots, \alpha, \circ, \beta, \ldots]}{\begin{array}{r|l} \alpha : P_0 & \alpha : P_2 \\ & \beta : \lambda P \end{array}}$$

ρ:

$\mathsf{A}P_\rho$:

$$
\begin{array}{lcl}
(\boldsymbol{K},\chi) \Vdash\!\!\Vdash \alpha : \mathsf{A}P, & \Leftrightarrow & (\boldsymbol{K},\chi) \Vdash\!\!\Vdash \alpha : P_0 \\
(\boldsymbol{K},\chi) \Vdash\!\!\Vdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ & & \vee \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\Vdash \lambda : [\ldots,\alpha,L\cup\{P_2,\mathsf{A}P\},\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\Vdash \alpha : P_2 \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\Vdash \hat{\gamma}(\mathsf{free}(T)) : L \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\Vdash \hat{\gamma}(\mathsf{free}(T)) : P_0 \\
 & & \vee \\
 & & \text{falls } \beta \neq \hat{\infty}\text{:} \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\Vdash \lambda : [\ldots,\alpha,L\cup\{P_2,\mathsf{A}P\},\beta,\ldots] \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\Vdash \alpha : P_2 \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\Vdash \beta : \mathsf{A}P \\
 & & \vee \\
 & & \text{falls } \beta = \hat{\infty}\text{:} \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\Vdash \lambda : [\ldots,\alpha,L\cup\{P_2,\mathsf{A}P\},\hat{\infty}] \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\Vdash \alpha : P_2 \\
 & & \\
(\boldsymbol{K},\chi) \Vdash\!\!\Vdash \alpha : \mathsf{A}P, & \Leftrightarrow & (\boldsymbol{K},\chi) \Vdash\!\!\Vdash \alpha : P_0 \\
(\boldsymbol{K},\chi) \Vdash\!\!\Vdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & & \vee \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\Vdash \alpha : P_2 \\
 & & (\boldsymbol{K},\chi) \Vdash\!\!\Vdash \beta : \mathsf{A}P
\end{array}
$$

Dabei wird ggf. Ω so ergänzt, daß $\hat{\gamma}(\chi(\mathsf{free}(T)))$ den passenden Zustand beschreibt (betrifft Ψ und Π).

$$
\begin{array}{c}
\alpha : \mathsf{A}P \\
\lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ \\
\hline
\begin{array}{r|l|c}
\alpha : P_0 & \lambda : [\ldots,\alpha,L\cup\{P_2,\mathsf{A}P\},\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] & \text{falls } \beta \neq \hat{\infty}\text{:} \\
 & \alpha : P_2 & \lambda : [\ldots,\alpha,L\cup\{P_2,\mathsf{A}P\},\beta,\ldots] \\
 & \hat{\gamma}(\mathsf{free}(T)) : L & \alpha : P_2 \\
 & \hat{\gamma}(\mathsf{free}(T)) : P_0 & \beta : \mathsf{A}P \\
 & & \\
 & & \text{falls } \beta = \hat{\infty}\text{:} \\
 & & \lambda : [\ldots,\alpha,L\cup\{P_2,\mathsf{A}P\},\hat{\infty}] \\
 & & \alpha : P_2
\end{array}
\end{array}
$$

$$
\begin{array}{c}
\alpha : \mathsf{A}P \\
\lambda : [\ldots,\alpha,\circ,\beta,\ldots] \\
\hline
\begin{array}{r|l}
\alpha : P_0 & \alpha : P_2 \\
 & \beta : \mathsf{A}P
\end{array}
\end{array}
$$

λP_ρ:

$$
\begin{array}{lcl}
(\boldsymbol{K},\chi) \Vdash \alpha : \lambda P, & \Leftrightarrow & (\boldsymbol{K},\chi) \Vdash \alpha : P_0 \\
(\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ & & \vee \\
 & & (\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,L \cup \{P_2\},\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] \\
 & & (\boldsymbol{K},\chi) \Vdash \alpha : P_2 \\
 & & (\boldsymbol{K},\chi) \Vdash \hat{\gamma}(\mathsf{free}(T)) : L \\
 & & (\boldsymbol{K},\chi) \Vdash \hat{\gamma}(\mathsf{free}(T)) : P_0 \\
 & & \vee \\
 & & \text{falls } \beta \neq \hat{\infty}\text{:} \\
 & & (\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,L \cup \{P_2\},\beta,\ldots] \\
 & & (\boldsymbol{K},\chi) \Vdash \alpha : P_2 \\
 & & (\boldsymbol{K},\chi) \Vdash \beta : \lambda P \\
 & & \vee \\
 & & \text{falls } \beta = \hat{\infty}\text{:} \\
 & & (\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,L \cup \{P_2\},\hat{\infty}] \\
 & & (\boldsymbol{K},\chi) \Vdash \alpha : P_2 \\
\\
(\boldsymbol{K},\chi) \Vdash \alpha : \lambda P, & \Leftrightarrow & (\boldsymbol{K},\chi) \Vdash \alpha : P_0 \\
(\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & & \vee \\
 & & (\boldsymbol{K},\chi) \Vdash \alpha : P_2 \\
 & & (\boldsymbol{K},\chi) \Vdash \beta : \lambda P
\end{array}
$$

Dabei wird ggf. Ω so ergänzt, daß $\hat{\gamma}(\chi(\mathsf{free}(T)))$ den passenden Zustand beschreibt (betrifft Ψ und Π).

$$
\begin{array}{c}
\alpha : \lambda P \\
\lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ \\
\hline
\begin{array}{r|c|c}
\alpha : P_0 & \lambda : [\ldots,\alpha,L \cup \{P_2\},\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] & \text{falls } \beta \neq \hat{\infty}\text{:} \\
 & \alpha : P_2 & \lambda : [\ldots,\alpha,L \cup \{P_2\},\beta,\ldots] \\
 & \hat{\gamma}(\mathsf{free}(T)) : L & \alpha : P_2 \\
 & \hat{\gamma}(\mathsf{free}(T)) : P_0 & \beta : \lambda P \\
 & & \\
 & & \text{falls } \beta = \hat{\infty}\text{:} \\
 & & \lambda : [\ldots,\alpha,L \cup \{P_2\},\hat{\infty}] \\
 & & \alpha : P_2
\end{array}
\end{array}
$$

$$
\begin{array}{c}
\alpha : \lambda P \\
\lambda : [\ldots,\alpha,\circ,\beta,\ldots] \\
\hline
\begin{array}{r|l}
\alpha : P_0 & \alpha : P_2 \\
 & \beta : \lambda P
\end{array}
\end{array}
$$

4.3 Die Regeln im Einzelnen

Nachdem im vorherigen Abschnitt abstrakte Regelschemata für die verschiedenen Formelklassen angegeben wurden, werden an dieser Stelle deren Instantiierungen für die einzelnen Modalitäten angegeben.

Sei im folgenden wieder $\hat{\gamma}$ ein noch nicht verwendetes Präfixsymbol, T der aktuelle Tableauzweig.

$(\boldsymbol{K}, \chi) \Vdash \alpha : \mathsf{A}\circ F$:

$(\boldsymbol{K}, \chi) \Vdash \alpha : \mathsf{A}\circ F$,
$(\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, L, \beta, \ldots]$, $L \neq \circ$

$\Leftrightarrow$

$(\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, \circ, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots]$
$(\boldsymbol{K}, \chi) \Vdash \hat{\gamma}(\mathsf{free}(T)) : L$
$(\boldsymbol{K}, \chi) \Vdash \hat{\gamma}(\mathsf{free}(T)) : F$
$\vee$
falls $\beta \neq \hat{\circ}$:
$(\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, \circ, \beta, \ldots]$
$(\boldsymbol{K}, \chi) \Vdash \beta : F$

$(\boldsymbol{K}, \chi) \Vdash \alpha : \mathsf{A}\circ F$,
$(\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, \circ, \beta, \ldots]$

$\Leftrightarrow$

$(\boldsymbol{K}, \chi) \Vdash \beta : F$

$\alpha : \mathsf{A}\circ F$ $\lambda : [\ldots, \alpha, L, \beta, \ldots]$, $L \neq \circ$		$\alpha : \mathsf{A}\circ F$ $\lambda : [\ldots, \alpha, \circ, \beta, \ldots]$
$\lambda : [\ldots, \alpha, \circ, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots]$ $\hat{\gamma}(\mathsf{free}(T)) : L$ $\hat{\gamma}(\mathsf{free}(T)) : F$	falls $\beta \neq \hat{\circ}$: $\lambda : [\ldots, \alpha, \circ, \beta, \ldots]$ $\beta : F$	$\beta : F$

$(\boldsymbol{K}, \chi) \Vdash \alpha : \lambda \circ F$:

$(\boldsymbol{K}, \chi) \Vdash \alpha : \lambda \circ F$,
$(\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, L, \beta, \ldots]$, $L \neq \circ$

$\Leftrightarrow$

$(\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, \circ, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots]$
$(\boldsymbol{K}, \chi) \Vdash \hat{\gamma}(\mathsf{free}(T)) : L$
$(\boldsymbol{K}, \chi) \Vdash \hat{\gamma}(\mathsf{free}(T)) : F$
$\vee$
falls $\beta \neq \hat{\circ}$:
$(\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, \circ, \beta, \ldots]$
$(\boldsymbol{K}, \chi) \Vdash \beta : F$

$(\boldsymbol{K}, \chi) \Vdash \alpha : \lambda \circ F$,
$(\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, \circ, \beta, \ldots]$

$\Leftrightarrow$

$(\boldsymbol{K}, \chi) \Vdash \beta : F$

$\alpha : \lambda \circ F$ $\lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ$ $\lambda : [\ldots, \alpha, \circ, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots]$ $\hat{\gamma}(\mathsf{free}(T)) : L$ $\hat{\gamma}(\mathsf{free}(T)) : F$	falls $\beta \neq \hat{\infty}$: $\lambda : [\ldots, \alpha, \circ, \beta, \ldots]$ $\beta : F$

$\alpha : \lambda \circ F$ $\lambda : [\ldots, \alpha, \circ, \beta, \ldots]$ $\beta : F$	

$(\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \alpha : \mathsf{A}\Box F$:

$$(\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \alpha : \mathsf{A}\Box F,\ (\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ \quad \Leftrightarrow \quad \begin{array}{l}(\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \lambda : [\ldots, \alpha, L \cup \{F, \mathsf{A}\Box F\}, \beta, \ldots] \\ (\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \alpha : F \\ (\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \beta : \mathsf{A}\Box F \quad \text{falls } \beta \neq \hat{\infty}\end{array}$$

$$(\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \alpha : \mathsf{A}\Box F,\ (\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \lambda : [\ldots, \alpha, \circ, \beta, \ldots] \quad \Leftrightarrow \quad \begin{array}{l}(\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \alpha : F \\ (\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \beta : \mathsf{A}\Box F\end{array}$$

$\alpha : \mathsf{A}\Box F$ $\lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ$ $\lambda : [\ldots, \alpha, L \cup \{F, \mathsf{A}\Box F\}, \beta, \ldots]$ $\alpha : F$ $\beta : \mathsf{A}\Box F \quad \text{falls } \beta \neq \hat{\infty}$	$\alpha : \mathsf{A}\Box F$ $\lambda : [\ldots, \alpha, \circ, \beta, \ldots]$ $\alpha : F$ $\beta : \mathsf{A}\Box F$

$(\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \alpha : \lambda \Box F$:

$$(\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \alpha : \lambda \Box F,\ (\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ \quad \Leftrightarrow \quad \begin{array}{l}(\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \lambda : [\ldots, \alpha, L \cup \{F\}, \beta, \ldots] \\ (\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \alpha : F \\ (\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \beta : \lambda \Box F \quad \text{falls } \beta \neq \hat{\infty}\end{array}$$

$$(\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \alpha : \lambda \Box F,\ (\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \lambda : [\ldots, \alpha, \circ, \beta, \ldots] \quad \Leftrightarrow \quad \begin{array}{l}(\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \alpha : F \\ (\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \beta : \lambda \Box F\end{array}$$

$\alpha : \lambda \Box F$ $\lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ$ $\lambda : [\ldots, \alpha, L \cup \{F\}, \beta, \ldots]$ $\alpha : F$ $\beta : \lambda \Box F \quad \text{falls } \beta \neq \hat{\infty}$	$\alpha : \lambda \Box F$ $\lambda : [\ldots, \alpha, \circ, \beta, \ldots]$ $\alpha : F$ $\beta : \lambda \Box F$

$(\boldsymbol{K}, \chi) \nVdash \alpha : \mathsf{A}\Diamond F$:

$$
\begin{array}{lcl}
(\boldsymbol{K}, \chi) \nVdash \alpha : \mathsf{A}\Diamond F, & \Leftrightarrow & (\boldsymbol{K}, \chi) \nVdash \alpha : F \\
(\boldsymbol{K}, \chi) \nVdash \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ & & \vee \\
& & (\boldsymbol{K}, \chi) \nVdash \lambda : [\ldots, \alpha, L \cup \{\neg F, \mathsf{A}\Diamond F\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] \\
& & (\boldsymbol{K}, \chi) \nVdash \alpha : \neg F \\
& & (\boldsymbol{K}, \chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : L \\
& & (\boldsymbol{K}, \chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : F \\
& & \vee \\
& & \text{falls } \beta \neq \hat{\circ}\text{:} \\
& & (\boldsymbol{K}, \chi) \nVdash \lambda : [\ldots, \alpha, L \cup \{\neg F, \mathsf{A}\Diamond F\}, \beta, \ldots] \\
& & (\boldsymbol{K}, \chi) \nVdash \alpha : \neg F \\
& & (\boldsymbol{K}, \chi) \nVdash \beta : \mathsf{A}\Diamond F
\end{array}
$$

$$
\begin{array}{lcl}
(\boldsymbol{K}, \chi) \nVdash \alpha : \mathsf{A}\Diamond F, & \Leftrightarrow & (\boldsymbol{K}, \chi) \nVdash \alpha : F \\
(\boldsymbol{K}, \chi) \nVdash \lambda : [\ldots, \alpha, \circ, \beta, \ldots] & & \vee \\
& & (\boldsymbol{K}, \chi) \nVdash \alpha : \neg F \\
& & (\boldsymbol{K}, \chi) \nVdash \beta : \mathsf{A}\Diamond F
\end{array}
$$

$$
\frac{\alpha : \mathsf{A}\Diamond F \qquad \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ}{\begin{array}{l|l|l} \alpha : F & \lambda : [\ldots, \alpha, L \cup \{\neg F, \mathsf{A}\Diamond F\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \text{falls } \beta \neq \hat{\circ}\text{:} \\ & \alpha : \neg F & \lambda : [\ldots, \alpha, L \cup \{\neg F, \mathsf{A}\Diamond F\}, \beta, \ldots] \\ & \hat{\gamma}(\mathsf{free}(T)) : L & \alpha : \neg F \\ & \hat{\gamma}(\mathsf{free}(T)) : F & \beta : \mathsf{A}\Diamond F \end{array}}
$$

$$
\frac{\alpha : \mathsf{A}\Diamond F \qquad \lambda : [\ldots, \alpha, \circ, \beta, \ldots]}{\begin{array}{l|l} \alpha : F & \alpha : \neg F \\ & \beta : \mathsf{A}\Diamond F \end{array}}
$$

$(\boldsymbol{K},\chi) \nVdash \alpha : \lambda \Diamond F$:

$$
\begin{array}{ll}
(\boldsymbol{K},\chi) \nVdash \alpha : \lambda \Diamond F & \Leftrightarrow \quad (\boldsymbol{K},\chi) \nVdash \alpha : F \\
(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ & \qquad \vee \\
 & \qquad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L \cup \{\neg F\},\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] \\
 & \qquad (\boldsymbol{K},\chi) \nVdash \alpha : \neg F \\
 & \qquad (\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : L \\
 & \qquad (\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : F \\
 & \qquad \vee \\
 & \qquad \text{falls } \beta \neq \hat{\infty}\text{:} \\
 & \qquad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L \cup \{\neg F\},\beta,\ldots] \\
 & \qquad (\boldsymbol{K},\chi) \nVdash \alpha : \neg F \\
 & \qquad (\boldsymbol{K},\chi) \nVdash \beta : \lambda \Diamond F
\end{array}
$$

$$
\begin{array}{ll}
(\boldsymbol{K},\chi) \nVdash \alpha : \lambda \Diamond F, & \Leftrightarrow \quad (\boldsymbol{K},\chi) \nVdash \alpha : F \\
(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & \qquad \vee \\
 & \qquad (\boldsymbol{K},\chi) \nVdash \alpha : \neg F \\
 & \qquad (\boldsymbol{K},\chi) \nVdash \beta : \lambda \Diamond F
\end{array}
$$

$$
\begin{array}{c}
\alpha : \lambda \Diamond F \\
\lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ \\
\hline
\begin{array}{r|c|c}
\alpha : F & \lambda : [\ldots,\alpha,L \cup \{\neg F\},\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] & \text{falls } \beta \neq \hat{\infty}\text{:} \\
 & \alpha : \neg F & \lambda : [\ldots,\alpha,L \cup \{\neg F\},\beta,\ldots] \\
 & \hat{\gamma}(\mathsf{free}(T)) : L & \alpha : \neg F \\
 & \hat{\gamma}(\mathsf{free}(T)) : F & \beta : \lambda \Diamond F
\end{array}
\end{array}
$$

$$
\begin{array}{c}
\alpha : \lambda \Diamond F \\
\lambda : [\ldots,\alpha,\circ,\beta,\ldots] \\
\hline
\begin{array}{r|l}
\alpha : F & \alpha : \neg F \\
 & \beta : \lambda \Diamond F
\end{array}
\end{array}
$$

Analog werden $\neg\Box F \equiv \Diamond\neg F$ und $\neg\Diamond F \equiv \Box\neg F$ behandelt.

$(\boldsymbol{K}, \chi) \nVdash \alpha : \mathsf{A}(F \text{ until } G)$:

$$
\begin{array}{lcl}
(\boldsymbol{K}, \chi) \nVdash \alpha : \mathsf{A}(F \text{ until } G), & \Leftrightarrow & (\boldsymbol{K}, \chi) \nVdash \alpha : G \\
(\boldsymbol{K}, \chi) \nVdash \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ & & \vee \\
& & (\boldsymbol{K}, \chi) \nVdash \lambda : [\ldots, \alpha, L \cup \{F, \neg G, \mathsf{A}(F \text{ until } G)\}, \\
& & \qquad\qquad \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] \\
& & (\boldsymbol{K}, \chi) \nVdash \alpha : F \\
& & (\boldsymbol{K}, \chi) \nVdash \alpha : \neg G \\
& & (\boldsymbol{K}, \chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : L \\
& & (\boldsymbol{K}, \chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : G \\
& & \vee \\
& & \text{falls } \beta \neq \hat{\infty} : \\
& & (\boldsymbol{K}, \chi) \nVdash \lambda : [\ldots, \alpha, L \cup \{F, \neg G, \mathsf{A}(F \text{ until } G)\}, \beta, \ldots] \\
& & (\boldsymbol{K}, \chi) \nVdash \alpha : F \\
& & (\boldsymbol{K}, \chi) \nVdash \alpha : \neg G \\
& & (\boldsymbol{K}, \chi) \nVdash \beta : \mathsf{A}(F \text{ until } G)
\end{array}
$$

$$
\begin{array}{lcl}
(\boldsymbol{K}, \chi) \nVdash \alpha : \mathsf{A}(F \text{ until } G), & \Leftrightarrow & (\boldsymbol{K}, \chi) \nVdash \alpha : G \\
(\boldsymbol{K}, \chi) \nVdash \lambda : [\ldots, \alpha, \circ, \beta, \ldots] & & \vee \\
& & (\boldsymbol{K}, \chi) \nVdash \alpha : F \\
& & (\boldsymbol{K}, \chi) \nVdash \alpha : \neg G \\
& & (\boldsymbol{K}, \chi) \nVdash \beta : \mathsf{A}(F \text{ until } G)
\end{array}
$$

$$
\frac{\alpha : \mathsf{A}(F \text{ until } G) \qquad \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ}{\begin{array}{l|l|l} \alpha : G & \lambda : [\ldots, \alpha, L \cup \{F, \neg G, \mathsf{A}(F \text{ until } G)\}, & \text{falls } \beta \neq \hat{\infty} : \\ & \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \lambda : [\ldots, \alpha, L \cup \{F, \neg G, \mathsf{A}(F \text{ until } G)\}, \beta, \ldots] \\ & \alpha : F & \alpha : F \\ & \alpha : \neg G & \alpha : \neg G \\ & \hat{\gamma}(\mathsf{free}(T)) : L & \beta : \mathsf{A}(F \text{ until } G) \\ & \hat{\gamma}(\mathsf{free}(T)) : G & \end{array}}
$$

$$
\frac{\alpha : \mathsf{A}(F \text{ until } G) \qquad \lambda : [\ldots, \alpha, \circ, \beta, \ldots]}{\begin{array}{l|l} \alpha : G & \alpha : F \\ & \alpha : \neg G \\ & \beta : \mathsf{A}(F \text{ until } G) \end{array}}
$$

$(\boldsymbol{K}, \chi) \nVdash \alpha : \lambda\,(F \text{ until } G)$:

$(\boldsymbol{K}, \chi) \nVdash \alpha : \lambda\,(F \text{ until } G)$,
$(\boldsymbol{K}, \chi) \nVdash \lambda : [\ldots, \alpha, L, \beta, \ldots]$, $L \neq \circ$

$\Leftrightarrow$

$(\boldsymbol{K}, \chi) \nVdash \alpha : G$
$\vee$
$(\boldsymbol{K}, \chi) \nVdash \lambda : [\ldots, \alpha, L \cup \{F, \neg G\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots]$
$(\boldsymbol{K}, \chi) \nVdash \alpha : F$
$(\boldsymbol{K}, \chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K}, \chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : L$
$(\boldsymbol{K}, \chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : G$
$\vee$
falls $\beta \neq \hat{\infty}$:
$(\boldsymbol{K}, \chi) \nVdash \lambda : [\ldots, \alpha, L \cup \{F, \neg G\}, \beta, \ldots]$
$(\boldsymbol{K}, \chi) \nVdash \alpha : F$
$(\boldsymbol{K}, \chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K}, \chi) \nVdash \beta : \lambda\,(F \text{ until } G)$

$(\boldsymbol{K}, \chi) \nVdash \alpha : \lambda\,(F \text{ until } G)$,
$(\boldsymbol{K}, \chi) \nVdash \lambda : [\ldots, \alpha, \circ, \beta, \ldots]$

$\Leftrightarrow$

$(\boldsymbol{K}, \chi) \nVdash \alpha : G$
$\vee$
$(\boldsymbol{K}, \chi) \nVdash \alpha : F$
$(\boldsymbol{K}, \chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K}, \chi) \nVdash \beta : \lambda\,(F \text{ until } G)$

$\alpha : \lambda\,(F \text{ until } G)$
$\lambda : [\ldots, \alpha, L, \beta, \ldots]$, $L \neq \circ$

$\alpha : G$	$\lambda : [\ldots, \alpha, L \cup \{F, \neg G\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots]$	falls $\beta \neq \hat{\infty}$:
	$\alpha : F$	$\lambda : [\ldots, \alpha, L \cup \{F, \neg G\}, \beta, \ldots]$
	$\alpha : \neg G$	$\alpha : F$
	$\hat{\gamma}(\mathsf{free}(T)) : L$	$\alpha : \neg G$
	$\hat{\gamma}(\mathsf{free}(T)) : G$	$\beta : \lambda\,(F \text{ until } G)$

$\alpha : \lambda\,(F \text{ until } G)$
$\lambda : [\ldots, \alpha, \circ, \beta, \ldots]$

$\alpha : G$	$\alpha : F$
	$\alpha : \neg G$
	$\beta : \lambda\,(F \text{ until } G)$

$(\boldsymbol{K},\chi) \nVdash \alpha : \mathsf{A}\neg(F \text{ until } G)$:

$(\boldsymbol{K},\chi) \nVdash \alpha : \mathsf{A}\neg(F \text{ until } G)$, $(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L,\beta,\ldots]$, $L \neq \circ$ $\Leftrightarrow$

$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg F$
$\vee$
$(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha, L \cup \{F, \neg G, \mathsf{A}\neg(F \text{ until } G)\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots]$
$(\boldsymbol{K},\chi) \nVdash \alpha : F$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : L$
$(\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : \neg F$
$(\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : \neg G$
$\vee$
falls $\beta \neq \hat{\infty}$:
$(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha, L \cup \{F,\ \neg G, \mathsf{A}\neg(F \text{ until } G)\}, \beta, \ldots]$
$(\boldsymbol{K},\chi) \nVdash \alpha : F,$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K},\chi) \nVdash \beta : \mathsf{A}\neg(F \text{ until } G)$
$\vee$
falls $\beta = \hat{\infty}$:
$(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha, L \cup \{F, \neg G, \mathsf{A}\neg(F \text{ until } G)\}, \hat{\infty}]$
$(\boldsymbol{K},\chi) \nVdash \alpha : F$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$

$(\boldsymbol{K},\chi) \nVdash \alpha : \mathsf{A}\neg(F \text{ until } G)$ $(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots]$ $\Leftrightarrow$

$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg F$
$\vee$
$(\boldsymbol{K},\chi) \nVdash \alpha : F$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K},\chi) \nVdash \beta : \mathsf{A}\neg(F \text{ until } G)$

$$\frac{\alpha : \mathsf{A}\neg(F \text{ until } G) \qquad \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ}{\begin{array}{l|l|l} \alpha : \neg G & \lambda : [\ldots, \alpha, L \cup \{F, \neg G, \mathsf{A}\neg(F \text{ until } G)\}, & \text{falls } \beta \neq \hat{\infty} : \\ \alpha : \neg F & \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \lambda : [\ldots, \alpha, L \cup \{F,\ \neg G, \mathsf{A}\neg(F \text{ until } G)\}, \beta, \ldots] \\ & \alpha : F & \alpha : F \\ & \alpha : \neg G & \alpha : \neg G \\ & \hat{\gamma}(\mathsf{free}(T)) : L & \beta : \mathsf{A}\neg(F \text{ until } G) \\ & \hat{\gamma}(\mathsf{free}(T)) : \neg F & \\ & \hat{\gamma}(\mathsf{free}(T)) : \neg G & \text{falls } \beta = \hat{\infty} : \\ & & \lambda : [\ldots, \alpha, L \cup \{F, \neg G, \mathsf{A}\neg(F \text{ until } G)\}, \hat{\infty}] \\ & & \alpha : F \\ & & \alpha : \neg G \end{array}}$$

$$\frac{\alpha : \mathsf{A}\neg(F \text{ until } G) \qquad \lambda : [\ldots, \alpha, \circ, \beta, \ldots]}{\begin{array}{l|l} \alpha : \neg G & \alpha : F \\ \alpha : \neg F & \alpha : \neg G \\ & \beta : \mathsf{A}\neg(F \text{ until } G) \end{array}}$$

$(\boldsymbol{K},\chi) \nVdash \alpha : \lambda\,\neg(F \text{ until } G)$:

$(\boldsymbol{K},\chi) \nVdash \alpha : \lambda\,\neg(F \text{ until } G)$
$(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ$

$\Leftrightarrow$

$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg F$
$\vee$
$(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L \cup \{F,\neg G\},$
$\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots]$
$(\boldsymbol{K},\chi) \nVdash \alpha : F$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : L$
$(\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : \neg F$
$(\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : \neg G$
$\vee$
falls $\beta \neq \hat{\infty}$:
$(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L \cup \{F,\ \neg G\},\beta,\ldots]$
$(\boldsymbol{K},\chi) \nVdash \alpha : F$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K},\chi) \nVdash \beta : \lambda\,\neg(F \text{ until } G)$
$\vee$
falls $\beta = \hat{\infty}$:
$(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L \cup \{F,\neg G\},\hat{\infty}]$
$(\boldsymbol{K},\chi) \nVdash \alpha : F, \quad (\boldsymbol{K},\chi) \nVdash \alpha : \neg G$

$(\boldsymbol{K},\chi) \nVdash \alpha : \lambda\,\neg(F \text{ until } G)$
$(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots]$

$\Leftrightarrow$

$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg F$
$\vee$
$(\boldsymbol{K},\chi) \nVdash \alpha : F$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K},\chi) \nVdash \beta : \lambda\,\neg(F \text{ until } G)$

$$\alpha : \lambda \neg(F \text{ until } G)$$
$$\lambda : [\ldots, \alpha, L, \beta, \ldots], \ L \neq \circ$$

$\alpha : \neg G$	$\lambda : [\ldots, \alpha, L \cup \{F, \neg G\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots]$	falls $\beta \neq \hat{\infty}$:
$\alpha : \neg F$	$\alpha : F$	$\lambda : [\ldots, \alpha, L \cup \{F, \ \neg G\}, \beta, \ldots]$
	$\alpha : \neg G$	$\alpha : F$
	$\hat{\gamma}(\mathsf{free}(T)) : L$	$\alpha : \neg G$
	$\hat{\gamma}(\mathsf{free}(T)) : \neg F$	$\beta : \lambda \neg(F \text{ until } G)$
	$\hat{\gamma}(\mathsf{free}(T)) : \neg G$	falls $\beta = \hat{\infty}$:
		$\lambda : [\ldots, \alpha, L \cup \{F, \neg G\}, \hat{\infty}]$
		$\alpha : F$
		$\alpha : \neg G$

$$\alpha : \lambda \neg(F \text{ until } G)$$
$$\lambda : [\ldots, \alpha, \circ, \beta, \ldots]$$

$\alpha : \neg G$	$\alpha : F$
$\alpha : \neg F$	$\alpha : \neg G$
	$\beta : \lambda \neg(F \text{ until } G)$

$(\boldsymbol{K},\chi) \nVdash \alpha : \mathsf{A}(F \ \ \mathsf{unless} \ \ G)$:

$(\boldsymbol{K},\chi) \nVdash \alpha : \mathsf{A}(F \ \ \mathsf{unless} \ \ G)$, $(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L,\beta,\ldots]$, $L \neq \circ$ $\Leftrightarrow$

$(\boldsymbol{K},\chi) \nVdash \alpha : G$
$\vee$
$(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L \cup \{F,\neg G,\mathsf{A}(F \ \ \mathsf{unless} \ \ G)\}, \hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots]$
$(\boldsymbol{K},\chi) \nVdash \alpha : F$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : L$
$(\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : G$
$\vee$
falls $\beta \neq \hat{\infty}$:
$(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L \cup \{F,\neg G,\mathsf{A}(F \ \ \mathsf{unless} \ \ G)\},\beta,\ldots]$
$(\boldsymbol{K},\chi) \nVdash \alpha : F$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K},\chi) \nVdash \beta : \mathsf{A}(F \ \ \mathsf{unless} \ \ G)$
$\vee$
falls $\beta = \hat{\infty}$:
$(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L \cup \{F,\neg G,\mathsf{A}(F \ \ \mathsf{unless} \ \ G)\},\hat{\infty}]$
$(\boldsymbol{K},\chi) \nVdash \alpha : F$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$

$(\boldsymbol{K},\chi) \nVdash \alpha : \mathsf{A}(F \ \ \mathsf{unless} \ \ G)$, $(\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots]$ $\Leftrightarrow$

$(\boldsymbol{K},\chi) \nVdash \alpha : G$
$\vee$
$(\boldsymbol{K},\chi) \nVdash \alpha : F$
$(\boldsymbol{K},\chi) \nVdash \alpha : \neg G$
$(\boldsymbol{K},\chi) \nVdash \beta : \mathsf{A}(F \ \ \mathsf{unless} \ \ G)$

$$\frac{\alpha : \mathsf{A}(F \ \mathsf{unless} \ G) \qquad \lambda : [\ldots, \alpha, L, \beta, \ldots], \ L \neq \circ}{\begin{array}{l|l|l} \alpha : G & \lambda : [\ldots, \alpha, L \cup \{F, \neg G, \mathsf{A}(F \ \mathsf{unless} \ G)\}, & \text{falls } \beta \neq \hat{\infty} : \\ & \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \lambda : [\ldots, \alpha, L \cup \{F, \neg G, \mathsf{A}(F \ \mathsf{unless} \ G)\}, \beta, \ldots] \\ & \alpha : F & \alpha : F \\ & \alpha : \neg G & \alpha : \neg G \\ & \hat{\gamma}(\mathsf{free}(T)) : L & \beta : \mathsf{A}(F \ \mathsf{unless} \ G) \\ & \hat{\gamma}(\mathsf{free}(T)) : G & \\ & & \text{falls } \beta = \hat{\infty} : \\ & & \lambda : [\ldots, \alpha, L \cup \{F, \neg G, \mathsf{A}(F \ \mathsf{unless} \ G)\}, \hat{\infty}] \\ & & \alpha : F \\ & & \alpha : \neg G \end{array}}$$

$$\frac{\alpha : \mathsf{A}(F \ \mathsf{unless} \ G) \qquad \lambda : [\ldots, \alpha, \circ, \beta, \ldots]}{\begin{array}{l|l} \alpha : G & \alpha : F \\ & \alpha : \neg G \\ & \beta : \mathsf{A}(F \ \mathsf{unless} \ G) \end{array}}$$

Analog für $\neg(F \ \mathsf{unless} \, G)$.

Manche zusammengesetzte Modaloperatoren lassen sich ebenfalls als μ-, ν-, π- oder ρ-Formel einordnen, womit für sie durch entsprechende Wahl der P_i die entsprechenden Regelschemata instantiiert werden können.
Andere bilden dagegen Mischformen dieser Regeln; in diesen Fällen müssen aus einer geeigneten Zerlegung der $\Vdash$-Relation Regeln konstruiert werden.

Es werden weitere Regeln benötigt, um Zustände aus Listen zu gewinnen. Für endliche, aber beliebig lange Pfadabschnitte kann man den letzten Zustand des Abschnitts benennen:

$$(\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, L, \beta, \ldots], \ \beta \neq \hat{\infty} \ \Leftrightarrow \begin{array}{l} (\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, L, \hat{\gamma}(\mathsf{free}(T)), \circ, \beta, \ldots] \\ (\boldsymbol{K}, \chi) \Vdash \hat{\gamma}(\mathsf{free}(T)) : L \\ \vee \\ (\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, \circ, \beta, \ldots] \end{array}$$

$$\frac{\lambda : [\ldots, \alpha, L, \beta, \ldots], \ L \neq \circ \qquad \beta \neq \hat{\infty}}{\begin{array}{l|l} \lambda : [\ldots, \alpha, L, \hat{\gamma}(\mathsf{free}(T)), \circ, \beta, \ldots] & \lambda : [\ldots, \alpha, \circ, \beta, \ldots] \\ \hat{\gamma}(\mathsf{free}(T)) : L & \end{array}}$$

Für unendliche Endabschnitte eines Pfades kann man einen beliebigen Zustand benennen:

$$(\boldsymbol{K},\chi) \models\!\!\models \lambda : [\ldots,\alpha,L,\hat{\infty}],\ L \neq \emptyset \ \Leftrightarrow \quad \begin{array}{l}(\boldsymbol{K},\chi) \models\!\!\models \lambda : [\ldots,\alpha,L,\hat{\gamma}(\mathsf{free}(T)),L,\hat{\infty}] \\ (\boldsymbol{K},\chi) \models\!\!\models \hat{\gamma}(\mathsf{free}(T)) : L\end{array}$$

$$\frac{\lambda : [\ldots,\alpha,L,\hat{\infty}],\ L \neq \emptyset}{\begin{array}{c}\lambda : [\ldots,\alpha,L,\hat{\gamma}(\mathsf{free}(T)),L,\hat{\infty}], \\ \hat{\gamma}(\mathsf{free}(T)) : L\end{array}}$$

In beiden Fällen muß Ω modifiziert werden, um $\hat{\gamma}(\mathsf{free}(T))$ wie beabsichtigt zu interpretieren (betrifft Ψ und Π).

An dieser Stelle werden einige Eigenschaften der so entstehenden Tableaux betrachtet:

Satz 10 *Sei T ein Tableauzweig, $\lambda : [\gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}] \in T$ eine Pfadinformationsformel und $F_\gamma := \bigwedge\{F : (\gamma : F) \in T\}$.*
Dann gilt $F_{\gamma_i} \rightarrow L_i$, d.h. die Menge der über $\Psi(\gamma_i, \chi)$ bekannten Formeln ist eine (i.a. echte) Obermenge der über den nachfolgenden Pfadabschnitt bekannten Formeln.

Beweis: Über Induktion nach der Anzahl der angewandten Tableauregeln. Jede Tableauregel erhält diese Eigenschaft: Es gilt $P_2 \in L_i \rightarrow P_2 \in F_{\gamma_i}$ (durch Anwendung der Tableauregeln) und $\mathsf{A}P \in L_i \rightarrow \mathsf{A}P \in F_{\gamma_i}$ (aus der Prämisse der Tableauregeln). □

Als weiteres steht noch der Beweis von Satz 3 aus. Punkt (4.) des Satzes kann stärker formuliert werden:

4. *Ist T ein Tableauzweig, $\hat{\lambda}$ ein Pfadsymbol, $I = \hat{\lambda}(t_1,\ldots,t_n) : [I'] \in T$ und $J = \hat{\lambda}(t_1,\ldots,t_n) : [J'] \in T$ zwei Pfadinformationsformeln, wobei oEdA. J die „neuere" ist, so gilt*

- *Alle in I' vorkommenden Präfixe kommen auch in J' vor.*
- *Kommt „γ_i, L_i, γ_j" in I' vor, so gilt für alle in J' zwischen γ_i und γ_j stehenden Listenkomponenten $L' \rightarrow L_i$.*
- $(\boldsymbol{K},\Omega,\chi) \not\models\!\!\models I \ \Rightarrow\ (\boldsymbol{K},\Omega,\chi) \models\!\!\models J$.

Der Beweis des Satzes erfolg prinzipiell induktiv über die Regelanwendungen. Jede Regelanwendung erhält diese Eigenschaften:

zu 1.: Ein Pfadsymbol $\hat{\lambda}$ wird genau einmal bei Auflösung eines E-Quantors mit n freien Variablen als Pfadbezeichner $\lambda = \hat{\lambda}(t_1,\ldots,t_n)$ eingeführt. Danach wird der entsprechende Term nur noch durch im gesamten Tableau ausgeführte (Abschluß)substitutionen verändert.

zu 2.: Analog (1.): Ein Präfixsymbol wird genau einmal neu eingeführt. Alle Veränderungen an dem Präfixterm erfolgen im gesamten Tableau.

zu 3.: Ein Präfixsymbol $\hat{\gamma}$ wird bei seiner Einführung in eine schon vorhandene Pfadinformationsformel an genau einer Stelle als Präfix γ eingefügt. Kommt es in weiteren Pfadinformationsformeln vor (diese beschreiben dann in γ beginnende bzw. abzweigende Pfade), so steht es dort genau an erster Stelle.

zu 4.: Folgt aus der Aktualisierung der Pfadinformationsformeln durch die einzelnen Regeln.

zu 5.: Eine Präfixformel mit einem Pfadselektor entsteht genau bei der Auflösung eines E-Quantors. In diesem Fall wird eine entsprechende Pfadinformationsformel erzeugt. Pflanzt sich eine Formel der Form $\gamma : \lambda P$ fort, so geschieht dies entlang der Pfadinformationsformel für λ, womit jedes dabei tangierte Präfix auf dieser enthalten ist.

zu 6.: Jedes Präfix wird bei seiner Einführung in einer Pfadinformationsformel plaziert, und tritt als Präfix nur auf Fortsetzungen dieses Zweiges auf.

zu 7.: Jedes Präfix wird bei seiner Einführung in einer Pfadinformationsformel plaziert. Nach Induktionsvoraussetzung erfüllt das erste Präfix dieser Pfadinformationsformel, womit das neu eingeführte Präfix ebenfalls erreichbar ist. □

5 Eigenschaften und Grenzen von $\mathcal{TK}$

In diesem Abschnitt werden einige theoretische Eigenschaften des Kalküls $\mathcal{TK}$ bewiesen.

5.1 Substitutionslemma

Das Substitutionslemma (Seite 10) läßt sich auf diese Situation übertragen:

Satz 11 (Substitutionslemma für zustandsunabhängig interpretierte Ausdrücke) *Sei $\boldsymbol{K}$ eine Kripke-Struktur, $\Omega = (\Phi, \Pi, \Psi)$ eine P&P-Interpretation, $\Sigma^c(\boldsymbol{K})$ der zustandsunabhängig interpretierte Anteil der Signatur $\Sigma(\boldsymbol{K})$, χ eine Variablenbelegung, X eine freie Variable, $g \in \boldsymbol{G}$,*
$s \in \mathsf{Term}_{\Sigma^c(\boldsymbol{K})}$, $t \in \mathsf{Term}_{\Sigma(\boldsymbol{K})}$ Terme, $\gamma \in \Gamma$ ein Präfix, $\lambda \in \Lambda$ ein Pfadbezeichner, F eine $\mathcal{TK}$-Zustandsformel, I eine Pfadinformationsformel.
Für $a := \boldsymbol{K}(s,\chi) \in \boldsymbol{U}(\boldsymbol{K})$ ist

$$\boldsymbol{K}([X \leftarrow s]t, \chi) = \boldsymbol{K}(t, \chi^a_X) \quad , \quad \Phi([X \leftarrow s]\lambda, \chi) = \Phi(\lambda, \chi^a_X) \quad ,$$
$$\Pi([X \leftarrow s]\lambda, \gamma, \chi) = \Pi(\lambda, \gamma, \chi^a_X) \quad , \quad \Psi([X \leftarrow s]\gamma, \chi) = \Psi(\gamma, \chi^a_X) \quad ,$$
$$(g,\chi) \models [X \leftarrow s]F \;\Leftrightarrow\; (g, \chi^a_X) \models F \quad , \quad (\boldsymbol{K}, \Omega, \chi) \not\models [X \leftarrow s](\gamma : F) \;\Leftrightarrow\; (\boldsymbol{K}, \Omega, \chi^a_X) \not\models (\gamma : F) \quad ,$$
$$(\boldsymbol{K}, \Omega, \chi) \not\models [X \leftarrow s]I \;\Leftrightarrow\; (\boldsymbol{K}, \Omega, \chi^a_X) \not\models I \qquad \textit{und}$$
$$[X \leftarrow s]I \textit{ konsistent zu } \Omega \textit{ bzgl. } \chi \;\Leftrightarrow\; I \textit{ konsistent zu } \Omega \textit{ bzgl. } \chi^a_X \quad .$$

Dabei wird deutlich, daß s zustandsunabhängig interpretiert werden (also $\in \mathsf{Term}_{\Sigma^c(\boldsymbol{K})}$ sein) muß, damit $a \in \boldsymbol{U}(\boldsymbol{K})$ – das Element, das für die modifizierte Variablenbelegung, die global stattfindet, benötigt wird – wohldefiniert ist.

Beweis:
Für Terme kann der Beweis direkt aus der Prädikatenlogik übernommen werden. Für Präfixe und Pfadbezeichner gilt dasselbe wie für Terme, da sie auch aus Funktionssymbolen und Argumenttermen bestehen. Für die Argumentterme findet dieselbe Auswertung $\boldsymbol{K}(t,\chi)$ wie für Terme Anwendung, für die Pfad- und Präfixsymbole wird Φ, ϕ bzw. Ψ, ψ anstelle $\boldsymbol{K}, K$ verwendet. Exemplarisch wird dies für ein Präfix ausgeführt:

$$\begin{aligned}
\Psi([X \leftarrow s]\hat{\gamma}(t_1, \ldots, t_n), \chi) &= \Psi(\hat{\gamma}([X \leftarrow s]t_1, \ldots, [X \leftarrow s]t_n), \chi) \\
&= (\psi(\hat{\gamma}))(\boldsymbol{K}([X \leftarrow s]t_1, \chi), \ldots, \boldsymbol{K}([X \leftarrow s]t_n, \chi)) \\
&\overset{\text{Terme}}{=} (\psi(\hat{\gamma}))(\boldsymbol{K}(t_1, \chi^a_X), \ldots, \boldsymbol{K}(t_n, \chi^a_X)) = \Psi(\hat{\gamma}(t_1, \ldots, t_n), \chi^a_X)
\end{aligned}$$

Für Formeln wird der Beweis über strukturelle Induktion über die Formelsyntax geführt, wobei als Basis das Substitutionslemma der PL1 verwendet wird. Dabei müssen alle verschiedenen Arten von $\mathcal{TK}$-Formeln mit passenden Modellrelationen betrachtet werden.

(1.) (entspricht Syntaxdefinition TA, Basisfall)
Sei F eine atomare prädikatenlogische Formel. Für $g \in G$ gilt

$$(g,\chi) \models [X \leftarrow s]F \;\Leftrightarrow\; (\boldsymbol{M}(g), \chi) \models [X \leftarrow s]F \overset{\text{PL}}{\Leftrightarrow} (\boldsymbol{M}(g), \chi^a_X) \models F \;\Leftrightarrow\; (g, \chi^a_X) \models F \quad .$$

(2.) (TZ1)

Sei F eine $\mathcal{TK}$-Zustandsformel. Exemplarisch wird $F \wedge G$ betrachtet. Für $g \in G$ gilt

$$\begin{aligned}
(g,\chi) \models [X \leftarrow s](F \wedge G) \quad &\Leftrightarrow \quad (g,\chi) \models ([X \leftarrow s]F) \wedge ([X \leftarrow s]G) \\
&\Leftrightarrow \quad (g,\chi) \models [X \leftarrow s]F \text{ und } (g,\chi) \models [X \leftarrow s]G \\
&\overset{\text{I.V.}}{\Leftrightarrow} \quad (g,\chi^a_X) \models F \text{ und } (g,\chi^a_X) \models G \\
&\Leftrightarrow \quad (g,\chi^a_X) \models F \wedge G \quad .
\end{aligned}$$

(3.) (TZQ)

Sei F eine $\mathcal{TK}$-Zustandsformel, y eine Variable (damit ist $y \neq X$). Stellvertretend wird $\forall y : F$ betrachtet.

$$\begin{aligned}
(g,\chi) \models [X \leftarrow s](\forall y : F) \quad &\Leftrightarrow \quad (g,\chi) \models \forall y : [X \leftarrow s]F \\
&\Leftrightarrow \quad \text{für alle } d \in \boldsymbol{U}(g) = \boldsymbol{U}(\boldsymbol{K})\text{: } (g,\chi^d_y) \models [X \leftarrow s]F \\
&\overset{\text{I.V.}}{\Leftrightarrow} \quad \text{für alle } d \in \boldsymbol{U}(g) = \boldsymbol{U}(\boldsymbol{K})\text{: } (g,(\chi^d_y)^a_X) \models F \\
&\Leftrightarrow \quad \text{für alle } d \in \boldsymbol{U}(g) = \boldsymbol{U}(\boldsymbol{K})\text{: } (g,(\chi^a_X)^d_y) \models F \\
&\Leftrightarrow \quad (g,\chi^a_X) \models \forall y : F \quad .
\end{aligned}$$

(4.) (TP1, TP2)

Stellvertretend für die in TP1 aufgezählten Formeln wird $F \,\mathsf{until}\, G$ betrachtet:

$$\begin{aligned}
&(p,\chi) \models [X \leftarrow s](F \,\mathsf{until}\, G) \quad \Leftrightarrow \\
&\qquad \text{Für } p = (g = g_0, g_1, \ldots) \text{ gilt:} \\
&\qquad \text{Es gibt ein } i \geq 0 \text{ mit } (g_i,\chi) \models [X \leftarrow s]G \text{ und für alle } j : 0 \leq j < i \text{ gilt} \\
&\qquad (g_i,\chi) \models [X \leftarrow s]F. \\
&\overset{\text{I.V.}}{\Leftrightarrow} \quad \text{Für } p = (g = g_0, g_1, \ldots) \text{ gilt:} \\
&\qquad \text{Es gibt ein } i \geq 0 \text{ mit } (g_i,\chi^a_X) \models G \text{ und für alle } j : 0 \leq j < i \text{ gilt } (g_i,\chi^a_X) \models F. \\
&\Leftrightarrow \quad (p,\chi^a_X) \models F \,\mathsf{until}\, G \quad .
\end{aligned}$$

(5.) (TZ2)

Stellvertretend wird diesmal E betrachtet:

$$\begin{aligned}
&(g,\chi) \models [X \leftarrow s](\mathsf{E}P) \quad \Leftrightarrow \\
&\qquad \text{Es gibt einen Pfad } p = (g_0, g_1, \ldots) \text{ in } \boldsymbol{K} \text{ mit } g_n = g \text{ und } (p|_n,\chi) \models [X \leftarrow s]P. \\
&\overset{\text{I.V.}}{\Leftrightarrow} \quad \text{Es gibt einen Pfad } p = (g_0, g_1, \ldots) \text{ in } \boldsymbol{K} \text{ mit } g_n = g \text{ und } (p|_n,\chi^a_X) \models P. \\
&\Leftrightarrow \quad (g,\chi^a_X) \models \mathsf{E}P \quad .
\end{aligned}$$

(6.) (TC1 & TK2)

Es ist $[X \leftarrow s](\gamma : F) = ([X \leftarrow s]\gamma) : ([X \leftarrow s]F)$, für das Präfix γ gilt $\Psi([X \leftarrow s]\gamma,\chi) = \Psi(\gamma,\chi^a_X)$.

$$\begin{aligned}
(\boldsymbol{K},\Omega,\chi) \models\!\models [X \leftarrow s](\gamma : F) \quad &\Leftrightarrow \quad (\boldsymbol{K},\Omega,\chi) \models\!\models ([X \leftarrow s]\gamma) : ([X \leftarrow s]F) \\
&\Leftrightarrow \quad (\Psi([X \leftarrow s]\gamma,\chi),\chi) \models [X \leftarrow s]F \\
&\Leftrightarrow \quad (\Psi(\gamma,\chi^a_X),\chi) \models [X \leftarrow s]F \\
&\overset{\text{I.V.}}{\Leftrightarrow} \quad (\Psi(\gamma,\chi^a_X),\chi^a_X) \models F \\
&\Leftrightarrow \quad (\boldsymbol{K},\Omega,\chi^a_X) \models\!\models \gamma : F \quad .
\end{aligned}$$

(7.) (TC2 & TK2)

Es ist $[X \leftarrow s](\gamma : \lambda P) = ([X \leftarrow s]\gamma) : ([X \leftarrow s]\lambda)([X \leftarrow s]P)$, für die Terme γ und λ gilt $\Psi([X \leftarrow s]\gamma, \chi) = \Psi(\gamma, \chi^a_X)$, $\Phi([X \leftarrow s]\lambda, \chi) = \Phi(\lambda, \chi^a_X)$ und $\Pi([X \leftarrow s]\lambda, [X \leftarrow s]\gamma, \chi) = \Pi(\lambda, \gamma, \chi^a_X)$.

$$
\begin{aligned}
(\boldsymbol{K}, \Omega, \chi) \models\!\!\models [X \leftarrow s](\gamma : \lambda P) \quad &\Leftrightarrow \quad (\boldsymbol{K}, \Omega, \chi) \models\!\!\models ([X \leftarrow s]\gamma) : ([X \leftarrow s]\lambda)([X \leftarrow s]P) \\
&\Leftrightarrow \quad (\Psi([X \leftarrow s]\gamma, \chi), \chi) \models ([X \leftarrow s]\lambda)([X \leftarrow s]P) \\
&\Leftrightarrow \quad (\Phi([X \leftarrow s]\lambda, \chi)|_{\Pi(([X \leftarrow s]\lambda), ([X \leftarrow s]\gamma), \chi)}, \chi) \models [X \leftarrow s]P \\
&\Leftrightarrow \quad (\Phi(\lambda, \chi^a_X)|_{\Pi(\lambda, \gamma, \chi^a_X)}, \chi) \models [X \leftarrow s]P \\
&\overset{\text{I.V.}}{\Leftrightarrow} \quad (\Phi(\lambda, \chi^a_X)|_{\Pi(\lambda, \gamma, \chi^a_X)}, \chi^a_X) \models P \\
&\Leftrightarrow \quad (\Psi(\gamma, \chi), \chi) \models \lambda P \\
&\Leftrightarrow \quad (\boldsymbol{K}, \Omega, \chi^a_X) \models\!\!\models \gamma : \lambda P \quad .
\end{aligned}
$$

(8.) (TK1)

Für die Funktionsterme γ_i und λ gilt $\Psi([X \leftarrow s]\gamma_i, \chi) = \Psi(\gamma_i, \chi^a_X)$, $\Phi([X \leftarrow s]\lambda, \chi) = \Phi(\lambda, \chi^a_X)$ und $\Pi([X \leftarrow s]\lambda, [X \leftarrow s]\gamma_i, \chi) = \Pi(\lambda, \gamma_i, \chi^a_X)$.
Sei $I = \lambda : [\gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}]$ die zu betrachtende Pfadinformationsformel.

Konsistenz: Sei

$$
\begin{aligned}
([X \leftarrow s]I) = ([X \leftarrow s]\lambda) : [&([X \leftarrow s]\gamma_0), ([X \leftarrow s]L_0), ([X \leftarrow s]\gamma_1), ([X \leftarrow s]L_1), \ldots, \\
&\ldots ([X \leftarrow s]\gamma_n), ([X \leftarrow s]L_n), \hat{\infty}]
\end{aligned}
$$

konsistent, d.h. für alle χ ist

$$
\begin{aligned}
&\Psi(([X \leftarrow s]\gamma_i), \chi) = \Phi(([X \leftarrow s]\lambda), \chi, \Pi(([X \leftarrow s]\lambda), ([X \leftarrow s]\gamma_i), \chi)) \quad \text{und} \\
&\Pi(([X \leftarrow s]\lambda), ([X \leftarrow s]\gamma_i), \chi) < \Pi(([X \leftarrow s]\lambda), ([X \leftarrow s]\gamma_{i+1}), \chi) \quad .
\end{aligned}
$$

Nach obigen Aussagen über die Auswertungen Φ und Π ergibt sich direkt für λ und γ_i

$$
\Leftrightarrow \quad \Psi(\gamma_i, \chi^a_X) = \Phi(\lambda, \chi^a_X, \Pi(\lambda, \gamma_i, \chi^a_X)) \quad \text{und} \quad \Pi(\lambda, \gamma_i, \chi^a_X) < \Pi(\lambda, \gamma_{i+1}, \chi^a_X) \quad ,
$$

was die Konsistenz von I bzgl. der Belegung χ^a_X beschreibt.

Gültigkeit:

$$
\begin{aligned}
&(\boldsymbol{K}, \Omega, \chi) \models\!\!\models ([X \leftarrow s]\lambda) : \\
&\qquad [([X \leftarrow s]\gamma_0), ([X \leftarrow s]L_0), ([X \leftarrow s]\gamma_1), ([X \leftarrow s]L_1), \ldots, ([X \leftarrow s]\gamma_n), ([X \leftarrow s]L_n), \hat{\infty}] \\
\Leftrightarrow \quad &\text{Für alle } i\text{:} \\
&\qquad L_i = \circ \;\Rightarrow\; \Pi(([X \leftarrow s]\lambda), ([X \leftarrow s]\gamma_{i+1}), \chi) = \Pi(([X \leftarrow s]\lambda), ([X \leftarrow s]\gamma_i), \chi) + 1 \\
&\qquad L_i \neq \circ \;\Rightarrow\; \forall j : \Pi(([X \leftarrow s]\lambda), ([X \leftarrow s]\gamma_i), \chi) < j < \Pi(([X \leftarrow s]\lambda), ([X \leftarrow s]\gamma_{i+1}), \chi) : \\
&\qquad\qquad (\Phi(([X \leftarrow s]\lambda), \chi, j), \chi) \models [X \leftarrow s]L_i \\
\overset{\text{I.V.}}{\Leftrightarrow} \quad &\text{Für alle } i\text{:} \\
&\qquad L_i = \circ \;\Rightarrow\; \Pi(\lambda, \gamma_{i+1}, \chi^a_X) = \Pi(\lambda, \gamma_i, \chi^a_X) + 1 \quad , \\
&\qquad L_i \neq \circ \;\Rightarrow\; \forall j : \Pi(\lambda, \gamma_i, \chi^a_X) < j < \Pi(\lambda, \gamma_{i+1}, \chi^a_X) : (\Phi(\lambda, \chi^a_X, j), \chi^a_X) \models L_i \quad , \\
\Leftrightarrow \quad &(\boldsymbol{K}, \Omega, \chi^a_X) \models\!\!\models I \quad .
\end{aligned}
$$

5.2 Korrektheit

Eventuell kann die Verarbeitung der freien Variablen bei der Einführung neuer Präfix- und Pfadsymbole anstelle $\mathsf{free}(T)$ (T aktueller Tableauzweig) reduziert werden (vgl. δ-Regel). In Frage käme $\mathsf{free}(F \cup I)$, wobei I die in der Prämisse verwendete Pfadinformationsformel ist. Da diese Untersuchung mit der Gesamtidee aber wenig zu tun hat, sondern eher im Bereich der Verfeinerungen anzusiedeln ist, wird sie in dieser Arbeit nicht weiter verfolgt.

Definition 30 *Ein Tableau $\mathcal{T}$ heißt* erfüllbar*, wenn es eine Kripke-Struktur $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ und eine P&P-Interpretation $\Omega = (\Phi, \Pi, \Psi)$ der im Tableau verwendeten Mengen $\hat{\Lambda}$ und $\hat{\Gamma}$ gibt, so daß es für jede Belegung χ der freien Variablen einen Zweig T in $\mathcal{T}$ gibt, für den $(\boldsymbol{K}, \Omega, \chi) \not\models T$ gilt.*

Definition 31 *Sei $\mathcal{T}$ ein Tableau. Ein Zweig T in $\mathcal{T}$ heißt* geschlossen*, falls er die Formel $\bot$ enthält. Das Tableau $\mathcal{T}$ heißt* geschlossen*, wenn jeder Zweig T in $\mathcal{T}$ geschlossen ist.*

Satz 12 (Korrektheitslemma) *Ist $\mathcal{T}$ ein erfüllbares Tableau, und entstehe $\mathcal{T}'$ aus $\mathcal{T}$ durch Anwendung einer Regel des obigen Kalküls, so ist auch $\mathcal{T}'$ erfüllbar.*

Beweis: Sei $\mathcal{T}$ erfüllbar, $\hat{\Lambda}$ die Menge der in $\mathcal{T}$ vorkommenden Pfadsymbole, $\hat{\Gamma}$ die Menge der in $\mathcal{T}$ vorkommenden Präfixsymbole. Dann gibt es also eine Kripke-Struktur $\boldsymbol{K}$ und eine P&P-Interpretation $\Omega = (\Phi, \Pi, \Psi)$ von $\hat{\Lambda}$ und $\hat{\Gamma}$, so daß es für jede Belegung ξ der freien Variablen einen Zweig T_ξ in $\mathcal{T}$ gibt, für den $(\boldsymbol{K}, \Omega, \xi) \not\models T_\xi$ gilt.
An dieser Stelle wird eine Fallunterscheidung nach der zur Erweiterung des Tableaux angewendeten Regel durchgeführt.

Die Korrektheit der α-, β-, γ- und δ-Regeln kann aus der Prädikatenlogik übertragen werden. Für die zusätzlichen Regeln wurde ein großer Teil der Arbeit mit der Angabe der Zerlegung der $\not\models$-Relation bei der Herleitung der Regeln bereits geleistet, so daß an dieser Stelle nur einige Fälle exemplarisch betrachtet werden:

- $\mathsf{E}P$: Sei $T \in \mathcal{T}$ derjenige Zweig, auf den die Regel angewandt wird. Dieser enthalte n freie Variablen $X_1, \ldots, X_n$. Damit führt die E-Regel ein neues n-stelliges Pfadsymbol $\hat{\kappa}$ und den Pfadbezeichner $\kappa = \hat{\kappa}(X_1, \ldots, X_n)$ ein.

Für alle Belegungen χ gilt

$$(\boldsymbol{K}, \Omega, \chi) \not\models \gamma : \mathsf{E}P \;\Leftrightarrow\; \text{Es gibt einen Pfad } p(\chi) = (g_0, g_1, \ldots) \text{ in } \boldsymbol{K} \text{ und ein } n(\chi) \text{ mit } g_{n(\chi)} = \Psi(\gamma, \chi) \text{ und } (p(\chi)|_{n(\chi)}, \chi) \not\models P.$$

Setze damit

$$\begin{aligned}
&\phi'(\hat{\lambda}) := \phi(\hat{\lambda}) \text{ für } \hat{\lambda} \in \hat{\Lambda} \quad , \\
&\phi'(\hat{\kappa}) := \begin{cases} (\chi(X_1), \ldots, \chi(X_n)) \mapsto p(\chi)|_{n(\chi)} & \text{falls } (\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models \gamma : \mathsf{E}P, \\ (\chi(X_1), \ldots, \chi(X_n)) \mapsto \text{beliebig} & \text{sonst} , \end{cases} \\
&\pi'(\hat{\lambda}, \hat{\alpha}) := \pi(\hat{\lambda}, \hat{\alpha}) \quad \text{für } \hat{\lambda} \in \hat{\Lambda},\ \hat{\alpha} \in \hat{\Gamma} \quad , \\
&\pi'(\hat{\kappa}, \hat{\gamma}) := \{\mathcal{U}^{n+\mathsf{ord}(\hat{\gamma})} \mapsto 0\}
\end{aligned}$$

und $\Omega' := (\Phi', \Pi', \Psi)$.

Für alle bereits in $\mathcal{T}$ vorkommenden Pfadbezeichner und Präfixe ist $\Omega = \Omega'$, so daß für alle Knoten $F \in \mathcal{T}$

$$(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\models F \;\Leftrightarrow\; (\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models F$$

gilt.

Gilt $(\boldsymbol{K}, \Omega, \chi) \not\models\!\!\!\models T$, so gibt es einen Zweig T_χ mit $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models T_\chi$. Dieser wird nicht erweitert, ist also ebenfalls ein Zweig in $\mathcal{T}'$, und es gilt $(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\models T_\chi$.

Gilt $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models T$, so ist insbesondere $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models \gamma : \mathsf{E}P$, womit $\Phi'(\kappa, \chi) = (\phi'(\hat{\kappa}))(\chi(X_1), \ldots, \chi(X_n)) = p(\chi)|_{n(\chi)}$ und $(p(\chi)|_{n(\chi)}, \chi) \models P$ ist.

Für die durch die Regelanwendung entstehende Pfadinformationsformel $\kappa : [\gamma, \emptyset, \hat{\infty}]$ erhält man die Konsistenz zu Ω' durch

$$\begin{aligned}
&\Pi'(\kappa, \gamma, \chi) = \pi'(\hat{\kappa}, \hat{\gamma}, (\chi(X_1), \ldots, \chi(X_n), \boldsymbol{K}(\mathsf{arg}(\gamma), \chi))) = 0 \quad \text{und} \\
&\Psi'(\gamma, \chi) = \Psi(\gamma, \chi) = g_{n(\chi)} = p(\chi)\!\downarrow_{n(\chi)} = \Phi'(\kappa, \chi)\!\downarrow_0 = \Phi'(\kappa, \chi, \Pi'(\kappa, \gamma, \chi)) \quad ,
\end{aligned}$$

die Gültigkeit der Pfadinformationsformel ist trivialerweise erfüllt.
Für die entstehende Präfixformel $\gamma : \kappa\, P$ ist

$$(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\models \gamma : \kappa\, P \;\Leftrightarrow\; (\Phi'(\kappa, \chi)|_{\Pi'(\kappa,\gamma,\chi)}, \chi) \models P \quad .$$

Es ist $\Phi'(\kappa, \chi) = p(\chi)|_{n(\chi)}$, womit $\Phi'(\kappa, \chi)|_{\Pi'(\kappa,\gamma,\chi)} = (p(\chi)|_{n(\chi)})|_0 = p(\chi)|_{n(\chi)}$ und diese Forderung äquivalent zu $(p(\chi)|_{n(\chi)}, \chi) \models P$ ist. Dies ist nach Annahme erfüllt.

- $\mathsf{A}P$, λP: Die Vorgehensweise für $\mathsf{A}P$ und λP ist prinzipiell dieselbe, so daß die Betrachtung des A-Falles genügen möge:
Die ν-Formeln können als Spezialfall der ρ-Formeln für $\nu(F) = \rho(F, \mathsf{false})$ betrachtet werden (vgl. $\Box F \equiv F \;\mathsf{unless}\; \mathsf{false}$). Die Beweise für π- und ρ-Formeln sind sich sehr ähnlich:

- Sei $P_\rho = \rho(F, G)$ (also $P_0 = G$, $P_1 = \neg G$, $P_2 = F \wedge \neg G$), $\gamma_i, \gamma_{i+1} \in \Gamma$, $\lambda \in \Lambda$.
Sei T der durch die Regelanwendung erweiterte Tableauzweig, $\gamma_i : \mathsf{A}P_\rho \in T$ und $I := \lambda : [\ldots, \gamma_i, L, \gamma_{i+1}, \ldots]$, $L \neq \circ \in T$, $\gamma_{i+1} \neq \hat{\infty}$, T enthalte n freie Variablen $X_1, \ldots, X_n$. Damit führt die ρ-Regel ggf. ein neues n-stelliges Präfixsymbol $\hat{\alpha}$ und das Präfix $\alpha = \hat{\alpha}(X_1, \ldots, X_n)$ ein.

Falls $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models T$ gilt, sei $k(\chi) := \Pi(\lambda, \gamma_i, \chi)$, $l(\chi) := \Pi(\lambda, \gamma_{i+1}, \chi)$, $g(\chi)$ der nächste relevante Zustand bezüglich λ, I, der Präfixformel $\gamma_i : \mathsf{A}P_\rho \in T$ und χ. Setze $n(\chi) := \min\{j : k(\chi) \leq j : \Phi(\lambda, \chi, j) = g(\chi)\}$, woraus nach Korollar 3 $k(\chi) \leq n(\chi) \leq l(\chi)$ folgt.

Setze damit

$$\psi'(\hat{\gamma}) := \psi(\hat{\gamma}) \text{ für } \hat{\gamma} \in \hat{\Gamma} \ ,$$

$$\psi'(\hat{\alpha}) := \begin{cases} (\chi(X_1), \ldots, \chi(X_n)) \mapsto g(\chi) = \Phi(\lambda, \chi, n(\chi)) & \text{falls } (\boldsymbol{K}, \Omega, \chi) \models\!\!\!\not\models \gamma_i : \mathsf{A}P_\rho \ , \\ (\chi(X_1), \ldots, \chi(X_n)) \mapsto \text{beliebig} & \text{sonst} \ , \end{cases}$$

$$\pi'(\hat{\kappa}, \hat{\gamma}) := \pi(\hat{\kappa}, \hat{\gamma}) \quad \text{für } \hat{\kappa} \in \hat{\Lambda}, \ \hat{\gamma} \in \hat{\Gamma} \ ,$$

$$\pi'(\hat{\lambda}, \hat{\alpha}) := \left\{ (\boldsymbol{K}(\mathsf{arg}(\lambda), \chi), \chi(X_1), \ldots, \chi(X_n)) \mapsto \left\{ \begin{array}{ll} n(\chi) & \text{falls } (\boldsymbol{K}, \Omega, \chi) \models\!\!\!\not\models \gamma_i : \mathsf{A}P_\rho \\ & \text{und } k < n(\chi) < l(\chi) \ , \\ \text{beliebig} & \text{sonst} \end{array} \right\} \right.$$

und $\Omega' := (\Phi, \Pi', \Psi')$.

Da $\hat{\alpha}$ in $\mathcal{T}'$ nur in der Pfadinformationsformel für $\hat{\lambda}$ vorkommt, wird π für $(\hat{\kappa}, \hat{\alpha})$, $\hat{\kappa} \neq \hat{\lambda}$, nicht definiert.

Für alle bereits in $\mathcal{T}$ vorkommenden Pfadbezeichner und Präfixe ist $\Omega = \Omega'$, so daß für alle Knoten $F \in \mathcal{T}$

$$(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\not\models F \ \Leftrightarrow \ (\boldsymbol{K}, \Omega, \chi) \models\!\!\!\not\models F$$

gilt.

Gilt $(\boldsymbol{K}, \Omega, \chi) \not\models\!\!\!\not\models T$, so gibt es einen Zweig T_χ mit $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\not\models T_\chi$. Dieser wird nicht erweitert, ist also ebenfalls ein Zweig in $\mathcal{T}'$, und es gilt $(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\not\models T_\chi$.

Für $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\not\models T$ wird eine Fallunterscheidung nach der Lage von $n(\chi)$ relativ zu $k(\chi)$ und $l(\chi)$ durchgeführt:

$n(\chi) = k(\chi)$: In diesem Fall ist $\Omega' = \Omega$. Nach Definition von $n(\chi)$ ist $\Phi(\lambda, \chi, n(\chi)) = \Phi(\lambda, \chi, k(\chi)) = \Psi(\gamma_i, \chi)$ der nächste relevante Zustand, $(\Psi(\gamma_i, \chi), \chi) \models P_0$ und damit nach Satz 8 $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\not\models \gamma_i : P_0$, womit die erste Alternative erfüllt ist.

$k(\chi) < n(\chi) < l(\chi)$: Durch die unveränderte Übernahme von $\pi|_{\hat{\Lambda} \times \hat{\Gamma}}$ und die Voraussetzung $k(\chi) < n(\chi) < l(\chi)$ ist $I' = \lambda : [\ldots, \gamma_i, L, \alpha, L, \gamma_{i+1}, \ldots]$ bezüglich Ω konsistent.
Wegen $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\not\models I = \lambda : [\ldots, \gamma_i, L, \gamma_{i+1}, \ldots]$ gilt für alle $j : k(\chi) < j < l(\chi) : (\Phi(\lambda, \chi, j), \chi) \models L$ und daraus direkt $(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\not\models \lambda : [\ldots, \gamma_i, L, \alpha, L, \gamma_{i+1}, \ldots]$.
Aus $k(\chi) < n(\chi) < l(\chi)$ und $\Psi'(\alpha, \chi) = \Phi(\lambda, \chi, n(\chi))$ folgt $(\Psi'(\alpha, \chi), \chi) \models L$, also $(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\not\models \alpha : L$.
Nach Voraussetzung ist $\Phi(\lambda, \chi, n(\chi)) = \Psi'(\alpha, \chi)$ der nächste relevante Zustand, woraus aus Satz 8 $\Psi'(\alpha, \chi) \models P_0$ und für alle $j : k(\chi) \leq j < n(\chi) : (\Phi(\lambda, \chi, j), \chi) \models P_2 \wedge \mathsf{A}P$ folgt.
Zusammen ergibt sich $(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\not\models \alpha : P_0$, $(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\not\models \lambda : [\ldots, \gamma_i, L \cup \{P_2, \mathsf{A}P\}, \alpha, L, \gamma_{i+1}, \ldots]$ und $(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\not\models \gamma_i : P_2$, womit die zweite Alternative erfüllt ist.

$n(\chi) = l(\chi)$: In diesem Fall ist $\Omega' = \Omega$ und $\Phi(\lambda, \chi, n(\chi)) = \Phi(\lambda, \chi, l(\chi)) = \Psi(\gamma_{i+1}, \chi)$ der nächste relevante Zustand. Da kein neues Präfix eingeführt wird, ist die verwendete Pfadinformationsformel weiterhin konsistent zu Ω bzgl. χ.
Aus Satz 8 erhält man für alle $j : k(\chi) \leq j < l(\chi) : (\Phi(\lambda, \chi, j), \chi) \models P_2 \wedge \mathsf{A}P$ und $(\Psi(\gamma_{i+1}, \chi), \chi) \models \mathsf{A}P$, also $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\not\models \lambda : [\ldots, \gamma_i, L \cup \{P_2, \mathsf{A}P\}, \gamma_{i+1}, \ldots]$, $\boldsymbol{K} \models\!\!\!\not\models \gamma_i : P_2$, und $\boldsymbol{K} \models\!\!\!\not\models \gamma_{i+1} : \mathsf{A}P$, womit die dritte Alternative erfüllt ist.

Soweit ist der Beweis für π-Formeln genau derselbe. Ein Unterschied ergibt sich für den nicht ausgeführten Fall $\beta = \hat{\infty}$, der, ebenso wie der Fall $L = \circ$, analog zu zeigen ist.

- Für μ-Formeln ergibt sich der folgende Beweis:

Sei T der durch die Regelanwendung erweiterte Tableauzweig, $\gamma_i : \mathsf{A}\circ F$, $P = \circ F$, also $P_0 = F$ mit der Pfadinformationsformel $I := \lambda : [\ldots, \gamma_i, L, \gamma_{i+1}, \ldots]$, $L \neq \circ \in T$, $\gamma_{i+1} \neq \hat{\infty}$ aufzulösen.
Wie eben wird ggf. ein neues Präfixsymbol $\hat{\alpha}$ als n-stelliges Funktionssymbol und das Präfix $\alpha = \hat{\alpha}(X_1, \ldots, X_n)$ eingeführt.

Für alle Belegungen χ gilt

$$\begin{aligned}
(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models \mathsf{A}\circ F \quad &\Leftrightarrow \text{ für alle } h \in G \text{ mit } (g = \Psi(\gamma_i, \chi), h) \in \boldsymbol{R} \text{ gilt } (h, \chi) \models F \\
&\Leftrightarrow \text{ Für alle Pfade } p = (\Psi(\gamma_i, \chi) = g_0, g_1, \ldots) \text{ gilt } (p{\downarrow_1}, \chi) = (g_1, \chi) \models F \\
&\Rightarrow \text{ Für alle } \lambda \in \Lambda \text{ gilt } ((\Phi(\lambda, \chi)|_{\Pi(\lambda,\gamma,\chi)}){\downarrow_1}, \chi) \models F \\
&\Leftrightarrow \text{ Für alle } \lambda \in \Lambda \text{ gilt } (\Phi(\lambda, \chi, \Pi(\lambda, \gamma, \chi) + 1), \chi) \models F
\end{aligned}$$

Analog zu oben wird das ggf. eingeführte Präfixsymbol $\hat{\alpha}$ in Ω' integriert:

$\psi'(\hat{\gamma}) := \psi(\hat{\gamma})$ für $\hat{\gamma} \in \hat{\Gamma}$,

$$\psi'(\hat{\alpha}) := \begin{cases} (\chi(X_1), \ldots, \chi(X_n)) \mapsto \Phi(\lambda, \chi, \Pi(\lambda, \gamma, \chi) + 1) & \text{falls } (\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models \gamma_i : \mathsf{A}\circ F \text{ ,} \\ \chi(X_1), \ldots, \chi(X_n) \mapsto \text{beliebig} & \text{sonst ,} \end{cases}$$

$\pi'(\hat{\kappa}, \hat{\lambda}) := \pi(\hat{\kappa}, \hat{\lambda})$ für $\hat{\lambda} \in \hat{\Lambda}$, $\hat{\gamma} \in \hat{\Gamma}$,

$$\pi'(\hat{\lambda}, \hat{\alpha}, \chi) := \left\{ (\boldsymbol{K}(\mathsf{arg}(\lambda), \chi), \chi(X_1), \ldots, \chi(X_n)) \mapsto \left\{ \begin{array}{ll} \Pi(\gamma, \chi) + 1 & \text{falls } (\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models \gamma_i : \mathsf{A}\circ F \text{ und} \\ & \Pi(\lambda, \gamma_{i+1}, \chi) \neq \Pi(\lambda, \gamma_i, \chi) + 1 \text{ ,} \\ \text{beliebig} & \text{sonst} \end{array} \right\} \right.$$

und $\Omega' := (\Phi, \Pi', \Psi')$.

Für alle bereits in $\mathcal{T}$ vorkommenden Pfadbezeichner und Präfixe ist $\Omega = \Omega'$, so daß für alle Knoten $F \in \mathcal{T}$

$$(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\models F \;\Leftrightarrow\; (\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models F$$

gilt.

Gilt $(\boldsymbol{K}, \Omega, \chi) \not\models\!\!\!\models T$, so gibt es einen Zweig T_χ mit $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models T_\chi$. Dieser wird nicht erweitert, ist also ebenfalls ein Zweig in $\mathcal{T}'$, und es gilt $(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\models T_\chi$.

Im weiteren wird unterschieden, ob $\Pi(\lambda, \gamma_i, \chi) + 1 = \Pi(\lambda, \gamma_{i+1}, \chi)$ ist.
Ist dies der Fall, so ist wegen $\Psi(\gamma_{i+1}, \chi) = \Phi(\lambda, \chi, \Pi(\lambda, \gamma_{i+1}, \chi))$ nach obiger Feststellung $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models \gamma_{i+1} : P_0$. Da in diesem Fall $\Omega = \Omega'$ ist, ist somit die erste Alternative erfüllt.

Im anderen Fall folgt aus $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\models \lambda : [\ldots, \gamma_i, L, \gamma_{i+1}, \ldots]$, daß für alle $j : \Pi(\lambda, \gamma_i, \chi) < j < \Pi(\lambda, \gamma_{i+1}, \chi)$ gilt $(\Phi(\lambda, \chi, j), \chi) \models L$.
Nach Konstruktion von Ω ist $\Pi(\lambda, \alpha, \chi) = \Pi(\lambda, \gamma_i, \chi) + 1 < \Pi(\lambda, \gamma_{i+1}, \chi)$, womit
$I' = \lambda : [\ldots, \gamma_i, \circ, \alpha, L, \gamma_{i+1}, \ldots]$ konsistent ist und $(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\models I'$ gilt.
Es ist $\Psi'(\alpha, \chi) = \Phi(\lambda, \chi, \Pi'(\lambda, \gamma_i, \chi)+1)$ und $\Pi'(\lambda, \gamma_i, \chi)+1$ ein solches j wie oben, womit $(\Psi(\alpha, \chi), \chi) \models L$ gilt. Nach obiger Feststellung gilt weiter $(\Psi'(\alpha, \chi), \chi) \models P_0$, also $(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\models \alpha : L$ und $(\boldsymbol{K}, \Omega', \chi) \models\!\!\!\models \alpha : P_0$, womit die zweite Alternative erfüllt ist.

Der Fall $L = \circ$ wird entsprechend bewiesen, ebenso nach demselben Prinzip die Regeln für die ν-Formeln sowie die beiden Regeln zur Instantiierung von Zuständen aus Listen.

- Abschließend soll noch die Anwendung der Abschlußregel betrachtet werden:

OEdA. betrifft die dabei angewendete Substitution nur eine in $\mathcal{T}$ freie Variable Y und ist durch $[Y \leftarrow s]$, $s \in \mathsf{Term}_{\Sigma^c(\boldsymbol{K})}$ ein zustandsunabhängig interpretierter Term, gegeben.
Sei χ eine Variablenbelegung für $\mathsf{free}(\mathcal{T}')$. Setze

$$\xi := \chi \cup \{Y \mapsto \boldsymbol{K}(\sigma(Y), \chi)\} \quad .$$

Damit ist

$$\xi = \chi_Y^{\xi(Y)} = \chi_Y^{\boldsymbol{K}(\sigma(Y),\chi)}$$

eine Belegung der in $\mathcal{T}$ freien Variablen, womit es nach Voraussetzung einen Zweig T_ξ in $\mathcal{T}$ gibt mit $(\boldsymbol{K}, \Omega, \xi) \not\models T_\xi$.
Ist T'_ξ der entsprechende Zweig in $\mathcal{T}'$, so gilt nach dem Substitutionslemma

$$(\boldsymbol{K}, \Omega, \xi) \not\models T_\xi \;\Leftrightarrow\; (\boldsymbol{K}, \Omega, \chi_Y^{\boldsymbol{K}(\sigma(Y),\chi)}) \not\models T_\xi \;\Leftrightarrow\; (\boldsymbol{K}, \Omega, \chi) \not\models [Y \leftarrow \sigma(Y)]T_\xi \quad .$$

Wegen $T'_\xi = [Y \leftarrow \sigma(Y)]T_\xi$ hat man also $(\boldsymbol{K}, \Omega, \chi) \not\models T'_\xi$. □

Satz 13 (Korrektheit des Tableaukalküls für CTL**)** *Gibt es ein geschlossenes Tableau über $\mathcal{F} \subseteq \mathcal{L}$(CTL), so ist $\mathcal{F}$ $\models^0$-unerfüllbar.*

Beweis:
In einem geschlossenen Tableau enthält jeder Zweig die Abschlußformel $\bot$, die als unerfüllbar definiert ist. Somit kann es nicht erfüllbar sein, und mit dem Korrektheitslemma erhält man, daß bereits das initiale Tableau nicht erfüllbar gewesen sein kann, es also keine Kripke-Struktur mit $0 \models \mathcal{F}$ gibt. Dies ist nach Definition äquivalent zu der $\models^0$-Unerfüllbarkeit von $\mathcal{F}$. □

5.3 Keine Vollständigkeit

In der vorliegenden Form für Prädikatenlogik ist der Kalkül *nicht vollständig* im üblichen Sinn.

Als Vorüberlegung wird gezeigt, daß die auf CTL basierende Temporallogik nicht kompakt ist:

Definition 32 *Eine Logik $\mathcal{L}$ heißt* kompakt, *wenn für jede Formelmenge $\mathcal{F} \in 2^{\mathcal{L}}$ gilt*

$\mathcal{F}$ *ist widersprüchlich* $\Leftrightarrow$ *es gibt eine endliche Teilmenge von $\mathcal{F}$, die widersprüchlich ist,*
(*wobei „widersprüchlich“ bezüglich einer gewählten Modellrelation zu verstehen ist*).

Satz 14 *Die auf* CTL *basierende Prädikatenlogik ist bezüglich der Modellrelation $\models^0$ nicht kompakt.*

Beweis:

$$M := \{\mathsf{A}\Box(\neg q \to \exists x : (\neg p(x) \wedge (\mathsf{A}\circ p(x)) \wedge \forall y : ((p(y) \to \mathsf{A}\circ p(y)) \wedge ((\neg p(y) \wedge x \neq y) \to \mathsf{A}\circ\neg p(y))))),$$
$$\mathsf{A}\Box(q \to (\mathsf{A}\circ\forall x : \neg p(x))),\ \mathsf{A}\Box(q \to \mathsf{A}\circ q),\ \forall x : \neg p(x)\} \cup$$
$$\{\mathsf{A}\Diamond(\exists^{\geq n} x : p(x)) \mid n \in \mathbf{N}\}$$
$$A := \{\mathsf{A}\Diamond q\}$$

Jede endliche Teilmenge $\mathcal{G}$ von $\mathcal{F} := M \cup A$ hat ein Modell:
Setze $\boldsymbol{G} := \mathbf{N}$, $\boldsymbol{R} := \{n, n+1\}$, $\boldsymbol{U}(\boldsymbol{K})$ beliebig.
Sei $n := \max\{i : \mathsf{A}\Diamond(\exists^{\geq i} x : p(x)) \in \mathcal{G}\}$. Setze $(M(0))(p) = \emptyset$, $(M(i))(q) = \mathit{false}$ für $0 \leq i \leq n$, womit $n = |\{d \in \boldsymbol{U}(\boldsymbol{K}) : d \in (M(n+1))(p)\}|$ ist. Für $j \geq n+1$ setze $(M(j))(q) = \mathit{true}$ und für alle $k \geq n+2 : (M(k))(p) = \emptyset$.

Die ganze Menge hat aber kein Modell: Das Problem dabei ist A; M alleine ist konsistent.
Es ist $\mathsf{A}\Box(q \to \mathsf{A}\Box q)$ ableitbar, und damit $\mathsf{A}\Box(q \to \mathsf{A}\circ\mathsf{A}\Box(\forall x : \neg p(x)))$. Sei $p = (0, g_1, g_2, \ldots)$ ein beliebiger Pfad in $\boldsymbol{K}$. Vor dem ersten Eintreten von q liegen nur endlich viele (=: n) Zustände mit $g_i \models \neg q$, und damit im $n+1$-ten Zustand $|\{d \in \boldsymbol{U}(\boldsymbol{K}) : d \in (M(g_{n+1}))(p)\}| = n$ und in allen folgenden Zuständen ist $|\{d \in \boldsymbol{U}(\boldsymbol{K}) : d \in (M(g_k))(p)\}| = 0$. Damit gibt es auf diesem Pfad keinen Zustand mit $g \models \exists^{\geq n+1} x : p(x)$. □

Satz 15 *Es gibt keinen korrekten und vollständigen Kalkül für prädikatenlogisches* CTL *bezüglich der Modellrelation $\models^0$.*

Beweis:
Angenommen, es gäbe einen korrekten und vollständigen Kalkül. Dessen Ableitbarkeitsrelation sei mit $\vdash$ bezeichnet, $\models^*$ bezeichne die Folgerbarkeitsrelation zu $\models^0$. Es ist $M \models^* \neg A$, also wegen der angenommenen Vollständigkeit des Kalküls $M \vdash \neg A$. (Für den Tableaukalkül $\mathcal{TK}$: $M \cup \{A\} \vdash_{TK} \perp$ ein geschlossenes Tableau.) Dabei wird nur eine endliche Teilmenge N von M verwendet: $N \cup \{A\} \vdash_{TK} \perp$, also ist wegen der Korrektheit $N \models^* \neg A$, also ist $N \cup \{A\}$ bereits eine inkonsistente endliche Teilmenge von M. □

Damit wirft dieses Ergebnis neue Fragen auf:

(1.) Kann es einen für *endliche* Formelmengen vollständigen Kalkül geben? Bei Evolving Algebras ist die Spezifikation im allgemeinen endlich.

(2.) Ist mit Hilfe geeigneter zusätzlicher Konstanten jede solche nicht-kompakte Menge als Evolving Algebra auch endlich formulierbar?

(3.) Ist der vorgestellte Kalkül für endliche Formelmengen vollständig?

(4.) Inwieweit ist der Kalkül unter Hinzunahme geeigneter Induktionsregeln für die in der modellierten Evolving Algebra vorkommenden Datenstrukturen vollständig?

Die obige Formelmenge läßt sich bei Einführung einer Konstanten anz, die angibt, für wieviele x $p(x)$ gilt, sowie des $>$-Prädikates endlich beschreiben:

$$M := \{\mathsf{A}\Box(\forall n : (\neg q \wedge anz = n \rightarrow \mathsf{A}\circ(anz = n+1))),\ \mathsf{A}\Box(\exists n : anz = n),$$
$$\mathsf{A}\Box(q \rightarrow (\mathsf{A}\circ(anz = 0))),\ \mathsf{A}\Box(q \rightarrow \mathsf{A}\circ q),\ anz = 0,\ \forall n : \mathsf{A}\Diamond(anz = n)\}$$
$$A := \{\mathsf{A}\Diamond q\}$$

Damit läßt sich bei gezielter Vorgehensweise ein Widerspruch finden. (Einige der Formeln sind nicht CTL-Syntax, aber CTL$^+$. Damit sind sie äquivalent in CTL umformulierbar.)

- mit Induktion über die Zustandsfolgen:

1. $\mathsf{A}\Box(q \rightarrow \mathsf{A}\Box q)$
2. $\mathsf{A}\Box(q \rightarrow \mathsf{A}\circ(\mathsf{A}\Box(anz = 0)))$
3. $\mathsf{A}\Box(\mathsf{E}\circ(anz = 0) \rightarrow q)$
4. $\mathsf{A}\Box(\forall n : anz = n \rightarrow \mathsf{A}\Box(anz = 0 \vee anz > n))$
5. $\mathsf{A}\Box(\forall i : (anz = i \rightarrow (\mathsf{A}\circ(anz = i+1) \vee \mathsf{A}\circ\mathsf{A}\Box(anz = 0))))$
6. $\mathsf{A}\Box\forall i : ((q \wedge anz = i) \rightarrow \mathsf{A}\circ(anz = 0))$

- mit Induktion über $\mathbb{N} \geq 1$ und Zustandsfolge:

7. $\mathsf{A}\Box\forall n : n = 0 \vee (anz = n \rightarrow \mathsf{A}\circ\mathsf{A}\Box(anz \neq n))$
8. $\mathsf{A}\Box\forall n : \mathsf{A}((\Diamond(anz = n \wedge \mathsf{E}\circ(anz = 0))) \rightarrow (\Box\neg(anz = n+1)))$

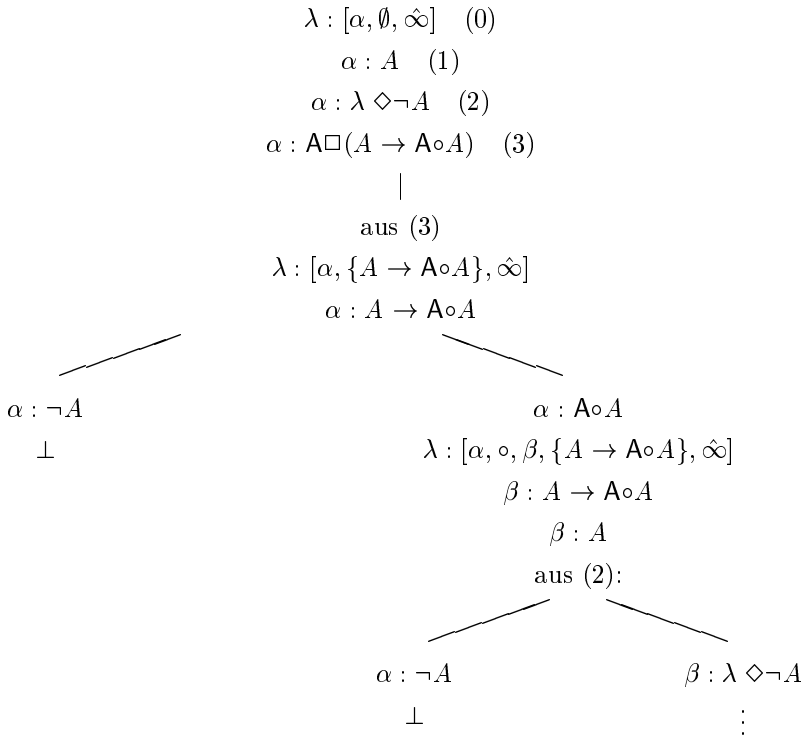
   mit A und 3., 4., 7. läßt sich dann ein Tableau über
9. $\forall n : \mathsf{A}\Diamond(anz = n)$

   schließen.

5.4 Fairness der Tableauprozedur

Analog dem Vorgehen bei der Betrachtung des prädikatenlogischen Tableaukalküls wird der Begriff der Fairness einer Tableauprozedur definiert.

Für den prädikatenlogischen Tableaukalkül genügt es, zu fordern, daß jede Formel irgendwann aufgelöst wird, sowie daß jede γ-Formel unendlich oft aufgelöst wird. Im vorliegenden Fall genügt dies nicht, wie das folgende Beispiel zeigt:
Es seien die Formeln (0) bis (3) gegeben.



Löst man in jedem Zustand zuerst $A \to \mathsf{A}\circ A$, und dann erst $\Diamond\neg A$ auf, wird so das Ereignis $\neg A$ immer weiter aufgeschoben, da die π-Formel $\Diamond\neg A$ nie in der Situation aufgelöst wird, daß das nächste Präfix in der Pfadinformationsformel gleich $\hat{\infty}$ ist, was ihre „Erledigung“ durch die Erzeugung einer Formel $\delta : \neg A$ zur Folge hätte.

Definition 33 *Sei $\mathcal{T}$ ein Tableau, T ein Zweig in $\mathcal{T}$, $F = \gamma : \mathsf{A}P$ oder $F = \gamma : \lambda P$ eine $\mathcal{TK}$-Präfixformel. Dann sind die Präfixformeln $G = \delta : \mathsf{A}P$, $G = \delta : P_0$ und $G = \delta : P_2$ sowie die Formeln $H = P_2$ und $H = \mathsf{A}P$ als Elemente von L-Komponenten* Nachkommen *von F, falls*

- *G durch eine Regelanwendung entstanden ist, wobei die in der Prämisse verwendete Präfixformel ebenfalls ein Nachkomme von F (oder F selber) ist oder*
- *$G = \delta : H$ im Zuge der Benennung eines neuen Präfixes mit Hilfe einer Pfadinformationsformel aus einer L-Komponente entstanden ist, in der H Nachkomme von F ist oder*

- *H als L-Komponente einer Pfadinformationsformel durch eine Regelanwendung entstanden ist, bei der H bereits als Nachfolger von F in einer L-Komponente der in der Prämisse verwendeten Pfadinformationsformel enthalten ist oder*
- *H als L-Komponente einer Pfadinformationsformel durch eine Regelanwendung entstanden ist, bei der die in der Prämisse verwendete Präfixformel ein Nachfolger von F ist.*

Definition 34 *Ein Folge $(\mathcal{T}_i)_{i\in\mathbf{N}}$, $\mathcal{T}_i \subseteq T_{i+1}$ von Tableaux ist* fair, *wenn für jeden Zweig T in $\mathcal{T}_i$ gilt:*

(1) Ist P eine ν-Formel, $F = \alpha : \mathsf{A}P \in T$, und $\lambda : [\ldots, \alpha, \ldots, \beta, \ldots] \in T$ so gibt es $\mathcal{T}_j$, $j \geq i$, so daß jede Fortsetzung des Zweiges T Formeln der folgenden Form enthält:

$$\lambda : [\ldots, \alpha, \ldots, \beta, L' \cup \{P_2, \mathsf{A}P\}, \hat{\infty}]$$
$$\beta : \mathsf{A}P \quad ,$$

wobei P_2, $\mathsf{A}P$ und $\beta : \mathsf{A}P$ Nachkommen von F sind.

(2) Ist P eine π-Formel, $F = \alpha : \mathsf{A}P \in T$, und $\lambda : [\ldots, \alpha, \ldots] \in T$ so gibt es $\mathcal{T}_j$, $j \geq i$, so daß jede Fortsetzung des Zweiges T Formeln der folgenden Form enthält:

$$\lambda : [\ldots, \alpha, \ldots, \gamma, \ldots]$$
$$\gamma : P_0 \quad ,$$

wobei $\gamma : P_0$ Nachkomme von F ist (dabei ist $\gamma = \alpha$ zugelassen).

(3) Ist P eine ρ-Formel, $F = \alpha : \mathsf{A}P \in T$, und $\lambda : [\ldots, \alpha, \ldots, \beta, \ldots] \in T$ so gibt es $\mathcal{T}_j$, $j \geq i$, so daß jede Fortsetzung des Zweiges T Formeln der folgenden Form umfaßt: Entweder

$$\lambda : [\ldots, \alpha, \ldots, \gamma, \ldots]$$
$$\gamma : P_0 \quad ,$$

wobei $\gamma : P_0$ Nachkomme von F ist (dabei ist $\gamma = \alpha$ zugelassen), oder

$$\lambda : [\ldots, \alpha, \ldots, \beta, L' \cup \{P_2, \mathsf{A}P\}, \hat{\infty}]$$
$$\beta : \mathsf{A}P \quad ,$$

wobei P_2, $\mathsf{A}P$ und $\beta : \mathsf{A}P$ Nachkommen von F sind.

(4) Ist P eine μ-Formel, $F = \alpha : \mathsf{A}P \in T$, und $\lambda : [\ldots, \alpha, \ldots] \in T$ so gibt es $\mathcal{T}_j$, $j \geq i$, so daß jede Fortsetzung des Zweiges T Formeln der folgenden Form umfaßt:

$$\lambda : [\ldots, \alpha, \circ, \beta, L, \ldots]$$
$$\beta : P_0$$

(5) Analog zu (1)-(4) für $F = \alpha : \lambda\, P \in T$.

(6) Ist $\lambda = [\ldots, \alpha, L, \beta, \ldots] \in T$, $L \neq \circ$, $\beta \neq \hat{\infty}$, so gibt es $\mathcal{T}_j$, $j \geq i$, so daß jede Fortsetzung des Zweiges T entweder $\lambda = [\ldots, \alpha, \circ, \beta, \ldots]$ oder eine Formel der Form $\lambda = [\ldots, \alpha, \ldots, \gamma, \circ, \beta, \ldots]$ enthält.

(7) Ist $\lambda = [\ldots, \alpha, L, \hat{\infty}] \in T$, $L \neq \emptyset$, so gibt es $\mathcal{T}_j$, $j \geq i$, so daß jede Fortsetzung des Zweiges T eine Formel der Form $\lambda = [\ldots, \alpha, \ldots, \beta, \ldots, \hat{\infty}]$ mit $\beta \neq \alpha$ enthält.

(8) Die Fairnessanforderungen für PL1-Tableaux sind erfüllt.

Dieses beschreibt die Forderungen, daß

- jede ν-Formel entlang aller Pfade, die sie betreffen soll, bis zum Ende fortgepflanzt wird,
- jede π-Formel (Erreichbarkeit) auf jedem Pfad, den sie betreffen soll, irgendwann erfüllt wird,
- jede ρ-Formel auf jedem Pfad, den sie betreffen soll, entweder irgendwann erfüllt wird oder bis zum Ende fortgepflanzt wird,
- für jeden Zustand, auf den ein beliebig langer Pfadabschnitt folgt, und über dessen Nachfolgezustand eine Aussage gemacht werde kann, dieser auch betrachtet wird.
- für jeden Zustand, der auf einen beliebig langen Pfadabschnitt folgt, der Vorgängerzustand auch betrachtet wird.

Eine Einhaltung dieser Fairnessanforderungen an das Tableauverfahren garantiert eine Vollständigkeit bezüglich aller im Tableau beschriebenen Entities:

- Alle Existenzaussagen (worunter auch Erreichbarkeitsaussagen fallen) werden erfüllt, d.h. sie erfüllende Entities beschrieben.
- Alle universellen Aussagen werden für alle Entities, die sie betreffen und die im Tableau beschrieben werden, erfüllt. Dies beinhaltet insbesondere die vollständige Fortpflanzung von Formeln der Form $\mathsf{A}P$ in alle Verzweigungen.

In der praktischen Durchführung sind die Forderungen (1)-(5) einfach zu erfüllen:
Jede Regel, die eine Formel der Form $\alpha : \mathsf{A/E}P$, P eine ν-, π-, oder ρ-Formel, entlang einem Pfad $\lambda : [\ldots, a, L, b, \ldots]$ auflöst, erzeugt maximal eine neue Formel der Form $\beta : \mathsf{A/E}P$, wobei ein solcher Schritt nur endlich oft möglich ist. Da es des weiteren nur endlich viele Pfadinformationsformeln gibt, ist auch jede A-Formel nach endlich vielen Regelanwendungen abgearbeitet. Bei Benennung eines neuen Pfades durch $\kappa : [\gamma, \ldots]$ sind daher alle Formeln der Form $\gamma : \mathsf{A}P$ für diesen Pfadbezeichner auszuwerten.

5.5 Eine Hintikka-Überlegung in Richtung Vollständigkeit

Analog dem durch die Definition einer Hintikka-Menge bestimmten Vorgehen zum Nachweis der Vollständigkeit bei prädikatenlogischen Tableaukalkülen kann man diesem Fall eine ähnliche Konstruktion bilden:

Definition 35 *Sei* $\boldsymbol{C} := \{\circ, -\}$.
Eine CTL-Hintikka-Entwicklung *zu einer Formelmenge* $\mathcal{F} \subset \mathcal{L}(\mathrm{CTL})$ *ist ein Tripel* $\boldsymbol{L} = (\boldsymbol{H}, \boldsymbol{P}, \boldsymbol{N})$, *wobei* $\boldsymbol{H}$ *eine Menge von Bezeichnern,* $\boldsymbol{P} \subseteq \boldsymbol{H}^{\mathrm{N}}$ *eine Menge von Folgen in* $(\boldsymbol{H} \times \boldsymbol{C})$ *ist und jedem* $h \in \boldsymbol{H}$ *eine Formelmenge* $\boldsymbol{N}(h)$ *zugeordnet ist mit den folgenden Eigenschaften:*

Bedingungen an $\boldsymbol{P}$:

P0: Jedes $p \in \boldsymbol{P}$ enthält jedes $h \in \boldsymbol{H}$ höchstens einmal.

P1: Sind $p_1, p_2 \in \boldsymbol{P}$, $h \in \boldsymbol{H}$ und gibt es $i, j \in \mathrm{N}$, $c_1, c_2 \in \boldsymbol{C}$, so daß $p_1(i) = (h, c_1)$ und $p_2(j) = (h, c_2)$ ist, so ist $i = j$ und für alle $0 \leq k < i$ $p_1(k) = p_2(k)$.

Bedingungen an $\boldsymbol{N}$: Dort, wo die zweiten Komponenten von $\boldsymbol{P}$ irrelevant sind, wird nur die Projektion auf die erste Komponente betrachtet.

H0: es gibt ein $0 \in \boldsymbol{H}$ mit $\mathcal{F} \subset \boldsymbol{N}(0)$

Für alle $h \in \boldsymbol{H}$:

H1 bis H6 definieren die bekannten prädikatenlogischen Hintikka-Mengen:

H1: $\neg F \in \boldsymbol{N}(h) \;\Rightarrow\; F \notin \boldsymbol{N}(h)$

H2: $\neg\neg F \in \boldsymbol{N}(h) \rightarrow F \in \boldsymbol{N}(h)$

H3: $F \wedge G \in \boldsymbol{N}(h) \;\Rightarrow\; F \in \boldsymbol{N}(h)$ und $G \in \boldsymbol{N}(h)$

H4: $F \vee G \in \boldsymbol{N}(h) \;\Rightarrow\; F \in \boldsymbol{N}(h)$ oder $G \in \boldsymbol{N}(h)$

H5: $\forall x : F \in \boldsymbol{N}(h) \;\Rightarrow\; [x \leftarrow t]F \in \boldsymbol{N}(h)$ für jeden variablenfreien Term t.

H6: $\exists x : F \in \boldsymbol{N}(h) \;\Rightarrow\;$ es gibt einen variablenfreien Term t mit $[x \leftarrow t]F \in \boldsymbol{N}(h)$

H7 bis H16 betrachten den modallogischen Anteil:

H7: $\mathsf{E}\Box F \in \boldsymbol{N}(h) \;\Rightarrow\; \exists p : \forall h' : p = (\ldots, h, \ldots, h', \ldots) \;\Rightarrow\; F \in \boldsymbol{N}(h')$

H8: $\mathsf{A}\Box F \in \boldsymbol{N}(h) \;\Rightarrow\; \forall p : \forall h' : p = (\ldots, h, \ldots, h', \ldots) \;\Rightarrow\; F \in \boldsymbol{N}(h')$

H9: $\mathsf{E}\Diamond F \in \boldsymbol{N}(h) \;\Rightarrow\; F \in \boldsymbol{N}(h)$ oder $\exists p, h' : p = (\ldots, h, \ldots, h', \ldots)$ mit $F \in \boldsymbol{N}(h')$ und $\forall i : p = (\ldots, h, \ldots, i, \ldots, h', \ldots) : \neg F \in \boldsymbol{N}(i)$

H10: $\mathsf{A}\Diamond F \in \boldsymbol{N}(h) \;\Rightarrow\; F \in \boldsymbol{N}(h)$ oder $\forall p : p = (\ldots, h, \ldots) \;\Rightarrow\; \exists h' : p = (\ldots, h, \ldots, h', \ldots)$ mit $F \in \boldsymbol{N}(h')$ und $\forall i : p = (\ldots, h, \ldots, i, \ldots, h', \ldots) : \neg F \in \boldsymbol{N}(i)$

analog für $\mathsf{E}/\mathsf{A}\neg\Diamond$, $\mathsf{E}/\mathsf{A}\neg\Box$

H11: $\mathsf{E}\circ F \in \boldsymbol{N}(h) \;\Rightarrow\; \exists p, h', c' : p = (\ldots, (h, \circ), (h', c'), \ldots)$ mit $F \in \boldsymbol{N}(h')$

H12: $\mathsf{A}\circ F \in \boldsymbol{N}(h) \;\Rightarrow\; \forall p : \exists h', c' : p = (\ldots, (h, \circ), (h', c), \ldots)$ mit $F \in \boldsymbol{N}(h')$

H13: $\mathsf{E}(F \;\mathsf{until}\; G) \in \boldsymbol{N}(h) \;\Rightarrow\; G \in \boldsymbol{N}(h)$ oder $\exists p, h' : p = (\ldots, h, \ldots, h', \ldots)$ mit $G \in \boldsymbol{N}(h')$ und $\forall i : p = (\ldots, h, \ldots, i, \ldots, h', \ldots) : F \in \boldsymbol{N}(i)$

H14: $\mathsf{A}(F \;\mathsf{until}\; G) \in \boldsymbol{N}(h) \;\Rightarrow\; G \in \boldsymbol{N}(h)$ oder $\forall p : p = (\ldots, h, \ldots) \;\Rightarrow\; \exists h' : p = (\ldots, h, \ldots, h', \ldots)$ mit $G \in \boldsymbol{N}(h')$ und $\forall i : p = (\ldots, h, \ldots, i, \ldots, h', \ldots) : \neg F \in \boldsymbol{N}(i)$

H15: $\mathsf{E}\neg(F \;\mathsf{until}\; G) \in \boldsymbol{N}(h) \;\Rightarrow\; \exists p : p = (\ldots, h, \ldots) : (\forall h' : p = (\ldots, h, \ldots, h', \ldots) \;\Rightarrow\; \neg G \in \boldsymbol{N}(h'))$ oder $(\forall h' : (p = (\ldots, h, \ldots, h', \ldots) \wedge G \in \boldsymbol{N}(h')) \;\Rightarrow\; \exists i : p = (\ldots, h, \ldots, i, \ldots, h', \ldots)$ mit $\neg F \in \boldsymbol{N}(i)$

H16: $\mathsf{A}\neg(F \;\mathsf{until}\; G) \in \boldsymbol{N}(h) \;\Rightarrow\; \forall p : p = (\ldots, h, \ldots) : (\forall h' : p = (\ldots, h, \ldots, h', \ldots) \;\Rightarrow\; \neg G \in \boldsymbol{N}(h'))$ oder $(\forall h' : (p = (\ldots, h, \ldots, h', \ldots) \wedge G \in \boldsymbol{N}(h')) \;\Rightarrow\; \exists i : p = (\ldots, h, \ldots, i, \ldots, h', \ldots)$ mit $\neg F \in \boldsymbol{N}(i)$

Die Konstruktion eines Tableaux entspricht der schrittweisen systematischen Konstruktion solcher Hintikka-Entwicklungen für die Menge der Eingabeformeln. Sei für $L \in 2^{\mathcal{L}}(\mathrm{CTL}) \cup \{\circ\}$

$$\bar{L} := \begin{cases} \circ & \text{falls } L = \circ, \\ - & \text{sonst.} \end{cases}$$

Aus jedem Tableauzweig T läßt sich eine Hintikka-Entwicklung ablesen:

$$\boldsymbol{H} := \Gamma \quad , \quad \boldsymbol{N}(\gamma) := F_\gamma \quad , \quad \boldsymbol{P} := \bar{\lambda} \quad ,$$

wobei für $\lambda \in \Lambda$ mit der aktuellsten Pfadformel $\lambda : [\gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}]$

$$\bar{\lambda} := ((\gamma_i, \bar{L}_i)_{i=1\ldots n}) \quad \text{ist.}$$

Allerdings ist man durch die vorgenommene Abstraktion in einer solchen Hintikka-Entwicklung noch weit von einer Beschreibung einer $\mathcal{F}$ modellierenden Kripke-Struktur entfernt: Aus der Hintikka-Entwicklung ist nicht ersichtlich, wie die dort durch die $(\gamma, -)$-Elemente der Pfade abstrahierten Abschnitte durch endlich viele Zustände ausgefüllt werden können.

Ist andererseits $\mathcal{F}$ konsistent, so gibt es eine erfüllende Kripke-Struktur. Eine solche wird ebenfalls durch den Tableaualgorithmus gefunden: Die Vorgänger-und Nachfolger-Regeln versuchen systematisch, die beiden bekannten Endzustände eines Pfadabschnittes entweder als direkte Nachfolger aufzufassen („zusammenzubinden“), oder von einem Ende ausgehend einen Zwischenzustand zu benennen. Im Falle einer konsistenten Formelmenge ist dieses Zusammenbinden für jeden Pfadabschnitt irgendwann erfolgreich, so daß man aus dem (unendlichen) Tableauzweig eine Kripke-Struktur ablesen kann.

Aufgrund der Abstraktion beliebig langer, aber endlicher Pfadabschnitte ist es daher i.a. nicht möglich, aus einem nicht geschlossenen Tableau ein Modell zu konstruieren. Diese Abstraktion macht es aber erst möglich, Erreichbarkeitsaussagen und insbesondere Fairnesszusicherungen im Kalkül ohne graphentheoretische Nachbearbeitung zu verarbeiten. Auf diese Weise können außerdem verschiedene Berechnungsfolgen beschrieben werden, die sich nur in irrelevanten Punkten unterscheiden.

Damit ist der Kalkül bis auf induktiv zu zeigende Aussagen vollständig.

5.6 Induktionsproblematik

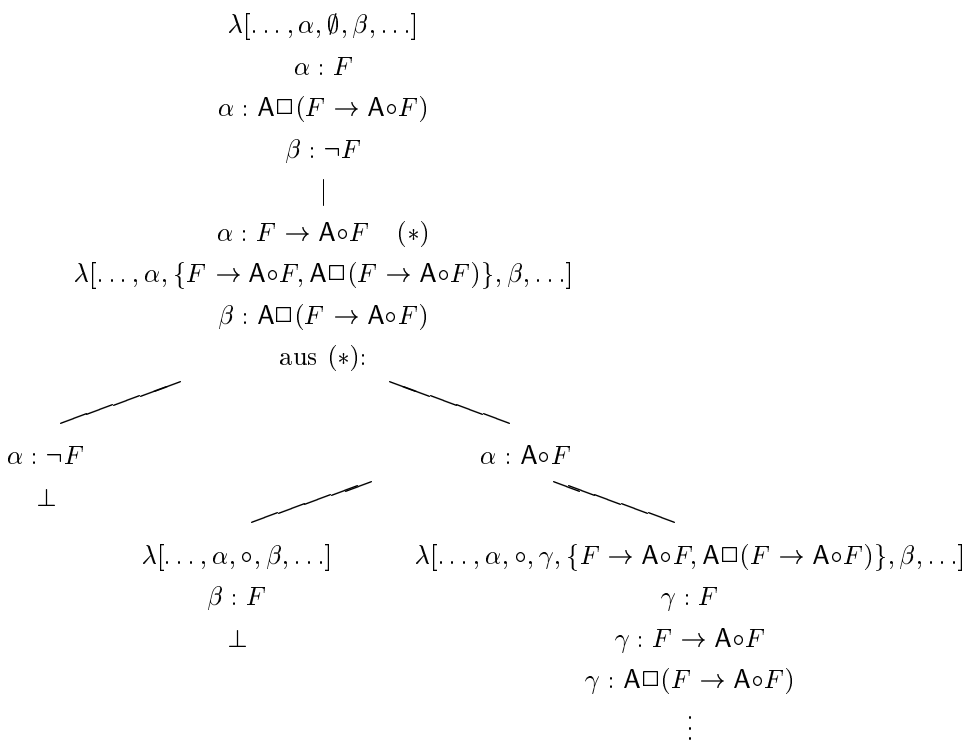
Eine herausragende Rolle spielen induktiv über die Folge der Zustände beweisbare Aussagen. Diese treten z.B. – insbesondere für die Anwendung auf Evolving Algebras – in Form von Invarianzen auf. Solche Aussagen müssen von den restlichen Aussagen getrennt betrachtet und bewiesen werden:

Das Fragment

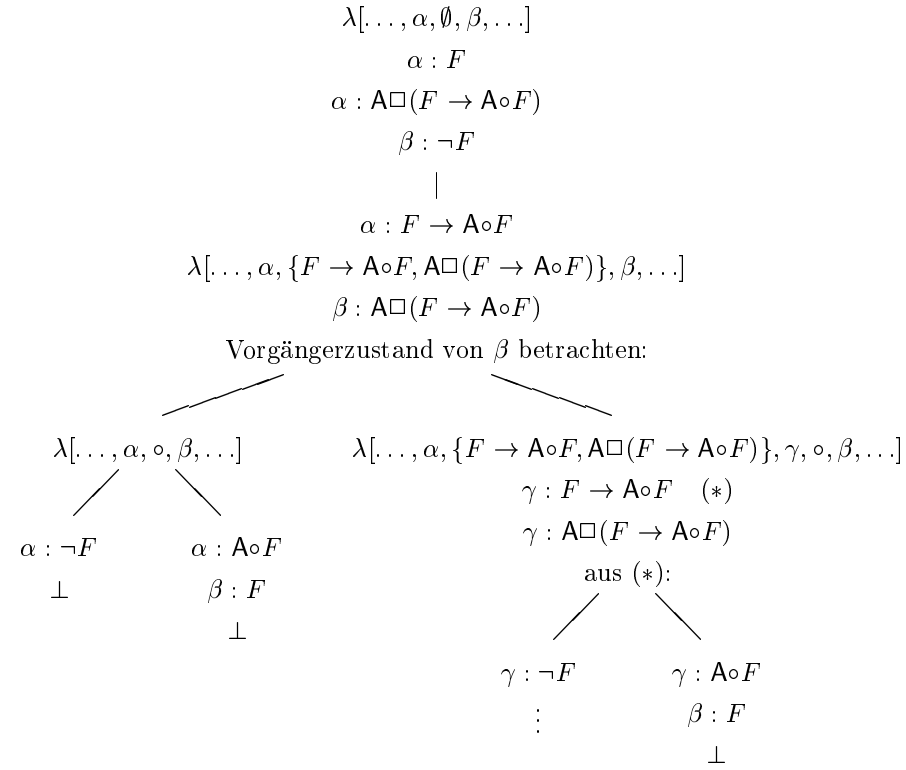
$$\lambda[\ldots, \alpha, \emptyset, \beta, \ldots]$$
$$\alpha : F$$
$$\alpha : \mathsf{A}\Box(F \rightarrow \mathsf{A}\circ F)$$
$$\beta : \neg F$$

(entspricht dem induktiven Beweis von $\alpha : \mathsf{A}\Box F$) kann mit den gegebenen Regeln nicht als inkonsistent nachgewiesen werden:

1.



2.



Im ersten Fall erreicht man mit γ dieselbe Situation wie in α, im zweiten Fall dieselbe wie in β. In beiden Fällen läßt das Auftreten derselben Situation, nur um einen Zustand verschoben, auf eine Lösung durch

Induktion schließen. Durch die unbekannte Länge des abstrahierten Pfadabschnitts zwischen α und β kann eine induktiv zu zeigende Aussage diesen nicht überwinden.

Um solche induktiv zu zeigenden Aussagen einbringen zu können, gibt es zwei formale Möglichkeiten – die im Grunde genommen dieselbe Idee repräsentieren: Lemmaformulierung, wobei das Lemma induktiv über die Zustandsfolge bewiesen wird.

Die erste ist eine Cut-Regel:

$$\frac{\lambda : [\ldots, \alpha, \ldots]}{\alpha : \neg Z \mid \alpha : Z} \quad , \quad \text{wobei } Z \text{ eine beliebige } \mathcal{TK}\text{-Zustandsformel ist ,}$$

die zweite ist, solche Lemmata isoliert zu formulieren und zu beweisen und als Formeln in das Tableau einzubringen (im obigen Beispiel wäre $Z = \mathsf{A}\Box F$).

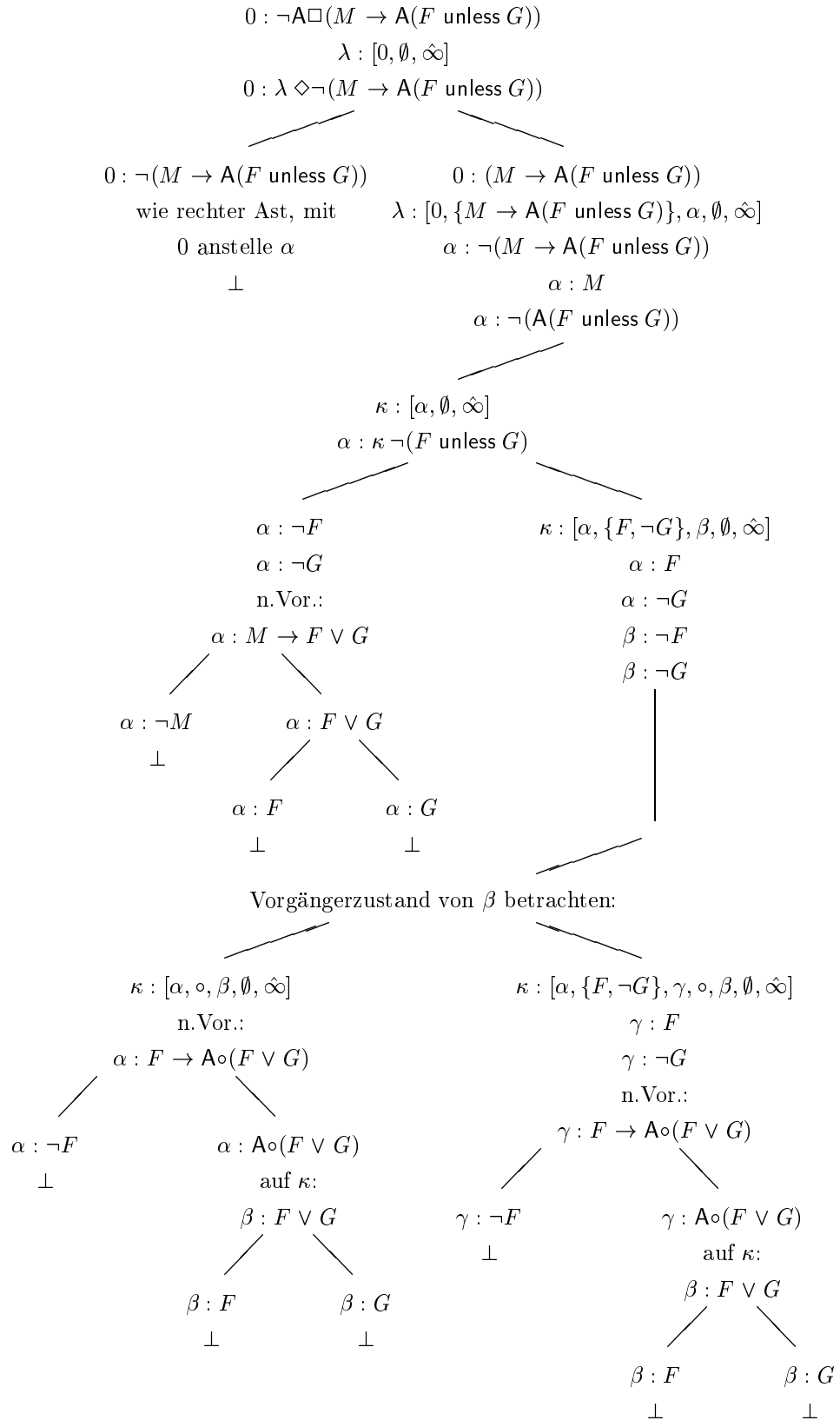
Der Aufwand ist bei beiden Möglichkeiten derselbe: Ein Tableau über dem negierten Lemma „$\alpha : \neg Z$“ und der benötigten Menge von Ausgangsformeln muß – gegebenenfalls unter Rückgriff auf weitere Lemmata – geschlossen werden.

Per Induktion sind Formeln der folgenden Form, wobei F und G Zustandsformeln sind, zu zeigen:

$$\mathsf{A}\Box F,\ \mathsf{E}\Box F,\ \mathsf{A}(F\ \mathsf{until}\ G),\ \mathsf{E}(F\ \mathsf{until}\ G),\ \mathsf{A}(F\ \mathsf{unless}\ G),\ \mathsf{E}(F\ \mathsf{unless}\ G) \quad .$$

Wie ein solcher Beweis mit dem beschriebenen Kalkül zu führen ist, wird am Beispiel von $\mathsf{A}\Box(M \rightarrow \mathsf{A}(F\ \mathsf{unless}\ G))$ gezeigt, wobei M eine Zustandsformel sei. Zusätzlich sei N eine Menge von als in der gesamten zu beschreibenden Kripke-Struktur gültig angenommenen Formeln.
Sei also $M \rightarrow \mathsf{A}(F\ \mathsf{unless}\ G)$ induktiv temporallogisch nachweisbar, d.h. es existieren geschlossene Tableaux für die Formeln $\mathsf{A}\Box(M \rightarrow (F \vee G))$ (Induktionsanfang) und $\mathsf{A}\Box(F \rightarrow \mathsf{A}\circ(F \vee G))$ (Induktionsschritt), zu deren Nachweis mindestens ein Induktionsbeweis weniger notwendig ist. Daher kann man diese Formeln als Lemmata betrachten:

Der entsprechende Tableauansatz ist



Damit liegt in solchen Fällen das Problem darin, zu erkennen, ob und mit welcher Induktionsaussage eine solche Induktion zum Ziel führt.

Der Induktionsschritt dabei von der Form $\mathsf{A}\Box(F \rightarrow \mathsf{A}\circ G)$, also ein Tableau über $\mathsf{E}\Diamond\neg(F \rightarrow \mathsf{A}\circ G)$ unter Verwendung in der Kripke-Struktur als allgemeingültig angenommener Formeln zu schließen. In dem – häufig vorkommenden – Fall, daß eine prädikatenlogische Invariante (also eine Formel von der Form $\mathsf{A}\Box F$, wobei F eine PL1-Formel ist) zu zeigen ist (Induktionsschritt $\mathsf{A}\Box(F \rightarrow \mathsf{A}\circ F)$), werden dabei nur zwei Zustände a und b, wobei $\boldsymbol{R}\,(a,b)$ gilt, mit $a \models F$ und $b \models \neg F$ betrachtet, wie es in den beiden untersten Teilbäumen des obigen Tableaux geschieht. Dieses Problem läßt sich in Prädikatenlogik codieren und kann damit prinzipiell von einem prädikatenlogischen Beweiser bearbeitet werden.

Im Kontext der Anwendung auf Evolving Algebras bzw. der Beschreibung dynamischer Systeme ergibt sich zusätzlich die Möglichkeit und Notwendigkeit der Induktion über (termerzeugte) Datenstrukturen.

5.7 Ein erstes Fazit

Nach der Vorstellung des Kalküls sowie seiner Grenzen ist es – sozusagen auf halber Strecke – an der Zeit, ein erstes Fazit zu ziehen:

Die Vorstellung, Aussagen wie Korrektheit oder Terminierung eines durch eine Evolving Algebra beschriebenen Prozesses in einem Zug, nur auf den Axiomen basierend, automatisch zu beweisen, ist utopisch. Aufgrund des Reichtums an anwendbaren Formeln ist – nicht nur für einen Tableaubeweiser – der Suchraum zu umfangreich.

Andererseits ist davon auszugehen, daß zur Synthese eines Prozesses bereits Überlegungen angestellt werden, die dessen Korrektheit begründen. Im Rahmen einer guten Dokumentation sollten diese als Teil des Projektes ebenfalls verfügbar sein. Im weiteren Sinn gehört auch der Beweis, daß der ablaufende Prozeß Modell jeder einzelnen solchen Überlegung ist, zum Korrektheitsbeweis des Prozesses.

Aus Sicht des Gesamtbeweises stellen diese Überlegungen Lemmata dar, die, einzeln gezeigt, zur Durchführung des Gesamtbeweises verwendet werden können.

Dabei ist für Teilprobleme auch die Verwendung eines prädikatenlogischen Beweisers zu erwägen.

Im Zuge dieses Vorgehens wird die Ausgangsformelmenge $\mathcal{F}$ i.a. aus einer (konsistenten) Menge $\mathcal{F}_A$ von Axiomen und Lemmata sowie einer negierten Behauptung F bestehen.

Damit ist der Begriff „Vollständigkeit“ in dem Sinne zu verstehen, daß die Argumentation eines in mathematischem Stil zielgerichtet geführten Beweises vollständig nachvollziehbar ist.

In diesem Zusammenhang wird auch der feine Unterschied zwischen einem Beweiser und einem Verifikationssystem sichtbar. Der hier vorgestellte Kalkül ist als Vertreter der letztgenannten Art anzusehen.

6 Der Tableaukalkül zu CTL – praktisch

Bei näherer Betrachtung des vorgestellten Kalküls stellt man fest, daß Tableaux, in denen viele Pfade vorkommen, sehr viele relativ einfach zu schließende Zweige enthalten, die mit der Aussage der Formeln wenig zu tun haben, sondern aus einer starken Redundanz des Kalküls folgen. Die theoretische Grundlage bilden dazu die Fortpflanzungssätze für CTL.

Bei der Zerlegung von Präfixformeln der Form $\gamma : \mathsf{A}P$, wobei P eine π- oder ρ-Formel ist, werden für jeden Pfad alle Möglichkeiten in das Tableau aufgenommen. Diese unterscheiden sich in erster Linie darin, ob $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\!/\ \gamma : P_0$ oder $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\!/\ \gamma : \neg P_0$ gilt, was der in Korollar 2 beschriebenen Fallunterscheidung

$$(g, \chi) \models\!\!\!\!/\ \mathsf{A}P \ \Rightarrow\ ((g, \chi) \models\!\!\!\!/\ P_0 \text{ exkl. oder } (g, \chi) \models\!\!\!\!/\ P_1)$$

entspricht. Diese „Entscheidung“ ist dann – da P_0 eine Zustandsformel ist – für alle von γ ausgehenden Pfade zwingend. Damit stellt sich nicht für jeden einzelnen Pfad die Frage, ob $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\!/\ \gamma : P_0$ oder $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\!/\ \gamma : \neg P_0$ gilt, sondern die Alternative liegt nur darin, ob $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\!/\ \gamma : \mathsf{A}P$ in γ schon – durch $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\!/\ \gamma : P_0$ – für alle Pfade erfüllt wird oder durch $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\!/\ \gamma : P_1$ aufgeschoben wird.

Diesem trägt eine Modifikation der entsprechenden Regeln Rechnung, wobei die Zerlegung zweistufig erfolgt:

Formeln der Formen $\gamma : \mathsf{A}P_\pi$ und $\gamma : \mathsf{A}P_\rho$ sind nur noch einmal anwendbar, dabei wird festgelegt, für welche Formeln P_i die Aussage $(\boldsymbol{K}, \Omega, \chi) \models\!\!\!\!/\ \gamma : P_i$ gelten soll und gegebenenfalls eine weitere Formel erzeugt, die die Fortsetzungen der Pfade behandelt. Dieser Schritt kann ohne Betrachtung der Pfadinformationsformeln geschehen.

Diese Unterteilung schließt alle Zweige des Tableaux aus, die aus unterschiedlichen (sogar entgegengesetzten) Entscheidungen bezüglich des momentanen Präfixes resultieren und somit sowieso geschlossen würden.

Ein weiterer Vorteil – weshalb sie für λ-Formeln übernommen wurde – liegt darin, daß man in vielen Fällen allein durch Betrachtung des momentanen Präfixes Zweige des Tableaux schließen kann, und mögliche verschiedene Alternativen einer Fortpflanzung der Formel entlang des Pfades nicht einzeln abschließen muß.

Um eine gleichartige Behandlung aller Formeln zu erreichen, wird diese Unterteilung für Formeln der Formen $\gamma : \mathsf{A}P_\nu$ und $\gamma : \mathsf{A}P_\mu$ übernommen. In beiden Fällen erspart dies keine Zweige, da keine Verzweigungen erzeugt werden. Im Fall von ν-Formeln hat man noch den kleinen Vorteil, eventuell Zweige schließen zu können, indem man nur das aktuelle Präfix betrachtet, ohne die Formel entlang eines Pfades fortpflanzen zu müssen.

Zur formalen Beschreibung dieser Unterteilung werden als weitere syntaktische Elemente (λ) für $\lambda \in \Lambda$ und (A) eingeführt, die an erster Stelle einer Zustandsformel stehen können, womit sich die folgende, um (TC3) erweiterte Syntaxbeschreibung der in einem Tableau vorkommenden Knotenformeln ergibt:

(TA) Jedes Atom ist eine $\mathcal{TK}$-Zustandsformel.

(TZ1) Sind F und G $\mathcal{TK}$-Zustandsformeln, so sind auch $\neg F$, $F \wedge G$, $F \vee G$ und $F \rightarrow G$ $\mathcal{TK}$-Zustandsformeln.

(TZQ) Ist F eine $\mathcal{TK}$-Zustandsformel und x eine Variable, so sind auch $\forall x : F$ und $\exists x : F$ $\mathcal{TK}$-Zustandsformeln.

(TP1) Sind F und G $\mathcal{TK}$-Zustandsformeln, so sind $\circ F$, $\Box F$, $\Diamond F$ und $(F \,\mathsf{until}\, G)$ $\mathcal{TK}$-Pfadformeln.

(TP2) Ist P eine $\mathcal{TK}$-Pfadformel, so ist $\neg P$ eine $\mathcal{TK}$-Pfadformel.

(TZ2) Ist P eine $\mathcal{TK}$-Pfadformel, so sind $\mathsf{A}P$ und $\mathsf{E}P$ $\mathcal{TK}$-Zustandsformeln.

(TC1) Jede $\mathcal{TK}$-Zustandsformel ist eine $\mathcal{TK}$-Prä-Knotenformel

(TC2) Ist P eine $\mathcal{TK}$-Pfadformel und λ ein Pfadselektor, so ist λP eine $\mathcal{TK}$-Prä-Knotenformel.

(TC3) Ist P eine $\mathcal{TK}$-Pfadformel und λ ein Pfadselektor, so sind $(\mathsf{A})P$ und $(\lambda)P$ $\mathcal{TK}$-Prä-Knotenformeln.

(TK1) Alle Pfadinformationsformeln sind $\mathcal{TK}$-Knotenformeln.

(TK2) Ist F eine $\mathcal{TK}$-Prä-Knotenformel und γ ein Präfix, so ist $\gamma : F$ eine $\mathcal{TK}$-Präfixformel.
Alle $\mathcal{TK}$-Präfixformeln sind $\mathcal{TK}$-Knotenformeln.

Mit der intendierten Aufteilung der Pfadformeln in Anforderungen an den aktuellen Zustand und Anforderungen an den weiteren Verlauf der Pfade

$$(\boldsymbol{K}, \Omega, \chi) \not\models \gamma : \mathsf{A}P \quad \Leftrightarrow \quad (\boldsymbol{K}, \Omega, \chi) \not\models \gamma : P_0 \text{ oder } ((\boldsymbol{K}, \Omega, \chi) \not\models \gamma : P_2 \text{ und } (\boldsymbol{K}, \Omega, \chi) \not\models \gamma : (\mathsf{A})P)$$
$$(\boldsymbol{K}, \Omega, \chi) \not\models \gamma : \lambda P \quad \Leftrightarrow \quad (\boldsymbol{K}, \Omega, \chi) \not\models \gamma : P_0 \text{ oder } ((\boldsymbol{K}, \Omega, \chi) \not\models \gamma : P_2 \text{ und } (\boldsymbol{K}, \Omega, \chi) \not\models \gamma : (\lambda)P)$$

ergibt sich der folgende Überblick über die Semantik der Grundtypen von Zustandsformeln:

(TZ2a) $(\boldsymbol{K}, \Omega, \chi) \not\models \gamma : \mathsf{A}P \quad :\Leftrightarrow$ Für alle Pfade $p = (g_0, g_1, \ldots)$ in $\boldsymbol{K}$ und alle n mit $g_n = \Psi(\gamma, \chi)$ gilt $(p|_n, \chi) \models P$.

(TZ2b) $(\boldsymbol{K}, \Omega, \chi) \not\models \gamma : \mathsf{E}P \quad :\Leftrightarrow$ Es gibt einen Pfad $p(\chi) = (g_0, g_1, \ldots)$ in $\boldsymbol{K}$ und ein $n(\chi)$ mit $g_{n(\chi)} = \Psi(\gamma, \chi)$ und $(p(\chi)|_{n(\chi)}, \chi) \models P$.

(TC2) $(\boldsymbol{K}, \Omega, \chi) \not\models \gamma : \lambda P \quad :\Leftrightarrow$ Es gilt $(\Phi(\lambda, \chi)|_{\Pi(\lambda,\gamma,\chi)}, \chi) \models P$.

(TC3a) $(\boldsymbol{K}, \Omega, \chi) \not\models \gamma : (\mathsf{A})P \quad :\Leftrightarrow$ Für alle Pfade $p = (g_0, g_1, \ldots)$ in $\boldsymbol{K}$ und alle n mit $g_n = \Psi(\gamma, \chi)$ gilt $(p|_{n+1}, \chi) \models P$.

(TC3b) $(\boldsymbol{K}, \Omega, \chi) \not\models \gamma : (\lambda)P \quad :\Leftrightarrow$ Es gilt $(\Phi(\lambda, \chi)|_{\Pi(\lambda,\gamma,\chi)+1}, \chi) \models P$.

Damit kann man als weiteres die Symbole (A) und (λ) in Beziehung zu den strikten Versionen der Modaloperatoren setzen:

Satz 16 *Sei P eine* CTL*-Pfadformel, P^+ die Pfadformel, die aus P entsteht, wenn man den äußersten Modaloperator $op \in \{\Box, \Diamond, \mathsf{unless}, \mathsf{until}\}$ durch seine strikte Version op^+ ersetzt. Seien $\boldsymbol{K}$, $\Omega = (\Phi, \Pi, \Psi)$, Λ und Γ wie immer. Dann gilt für alle Belegungen χ der freien Variablen, alle Präfixe $\gamma \in \Gamma$ und alle Pfadbezeichner $\lambda \in \Lambda$:*

$$(\boldsymbol{K}, \Omega, \chi) \models \gamma : (\mathsf{A})P \;\Leftrightarrow\; (\boldsymbol{K}, \Omega, \chi) \models \gamma : \mathsf{A}P^+ \quad \textit{und}$$
$$(\boldsymbol{K}, \Omega, \chi) \models \gamma : (\lambda)P \;\Leftrightarrow\; (\boldsymbol{K}, \Omega, \chi) \models \gamma : \lambda P^+ \quad .$$

Die (A)- bzw. (λ)-Formeln übernehmen die fortpflanzende Funktion von den A- bzw. (λ)-Formeln. Formeln der Form $\gamma : \mathsf{A}P$ werden damit genau einmal ohne Betrachtung der einzelnen Pfadinformationsformeln aufgelöst, Formeln der Form $\gamma : (\mathsf{A})P$ werden für jede das Präfix γ enthaltende Pfadinformationsformel einmal aufgelöst.

Modifizierte Regeln:

$\gamma : \mathsf{E}P$: wird unverändert beibehalten:

$$\frac{\gamma : \mathsf{E}P}{\begin{array}{c}\hat{\kappa}(\mathsf{free}(T)) : [\gamma, \emptyset, \hat{\infty}] \\ \gamma : \hat{\kappa}(\mathsf{free}(T))P\end{array}}$$

wobei $\hat{\kappa}$ ein noch nicht vorkommendes Pfadsymbol ist.

$\gamma : \mathsf{A}P$, $\gamma : \lambda P$:

Sei im folgenden $\hat{\gamma}$ ein noch nicht verwendetes Präfixsymbol, T der betrachtete Tableauzweig.

μ:

$\mathsf{A}P_\mu$:

$$(\boldsymbol{K}, \chi) \Vdash \alpha : \mathsf{A}P \quad \Leftrightarrow \quad (\boldsymbol{K}, \chi) \Vdash \alpha : (\mathsf{A})P$$

$(\mathsf{A})P_\mu$:

$$\begin{array}{ll}(\boldsymbol{K}, \chi) \Vdash \alpha : (\mathsf{A})P, & \Leftrightarrow \quad (\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, \circ, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] \\ (\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ & \quad\ (\boldsymbol{K}, \chi) \Vdash \hat{\gamma}(\mathsf{free}(T)) : L \\ & \quad\ (\boldsymbol{K}, \chi) \Vdash \hat{\gamma}(\mathsf{free}(T)) : P_0 \\ & \quad\ \vee \\ & \quad\ \text{falls } \beta \neq \hat{\infty}: \\ & \quad\ (\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, \circ, \beta, \ldots] \\ & \quad\ (\boldsymbol{K}, \chi) \Vdash \beta : P_0\end{array}$$

$$\begin{array}{ll}(\boldsymbol{K}, \chi) \Vdash \alpha : (\mathsf{A})P, & \Leftrightarrow \quad (\boldsymbol{K}, \chi) \Vdash \beta : P_0 \\ (\boldsymbol{K}, \chi) \Vdash \lambda : [\ldots, \alpha, \circ, \beta, \ldots] & \end{array}$$

Analog für λP_μ.

$$\frac{\alpha : \mathsf{A}P}{\alpha : (\mathsf{A})P} \qquad \frac{\alpha : \lambda P}{\alpha : (\lambda)P}$$

$$\frac{\begin{array}{c}\alpha : (\mathsf{A})P \\ \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ\end{array}}{\begin{array}{l|l}\lambda : [\ldots, \alpha, \circ, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \text{falls } \beta \neq \hat{\infty}: \\ \hat{\gamma}(\mathsf{free}(T)) : L & \lambda : [\ldots, \alpha, \circ, \beta, \ldots] \\ \hat{\gamma}(\mathsf{free}(T)) : P_0 & \beta : P_0\end{array}} \qquad \frac{\begin{array}{c}\alpha : (\mathsf{A})P \\ \lambda : [\ldots, \alpha, \circ, \beta, \ldots]\end{array}}{\beta : P_0}$$

$$\frac{\begin{array}{c}\alpha : (\lambda)P \\ \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ\end{array}}{\begin{array}{l|l}\lambda : [\ldots, \alpha, \circ, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \text{falls } \beta \neq \hat{\infty}: \\ \hat{\gamma}(\mathsf{free}(T)) : L & \lambda : [\ldots, \alpha, \circ, \beta, \ldots] \\ \hat{\gamma}(\mathsf{free}(T)) : P_0 & \beta : P_0\end{array}} \qquad \frac{\begin{array}{c}\alpha : (\lambda)P \\ \lambda : [\ldots, \alpha, \circ, \beta, \ldots]\end{array}}{\beta : P_0}$$

ν:

$\mathsf{A}P_\nu$:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : \mathsf{A}P \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : P_2$$
$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\mathsf{A})P$$

$(\mathsf{A})P_\nu$:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\mathsf{A})P, \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,L \cup \{P_2,(\mathsf{A})P\},\beta,\ldots]$$
$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : \mathsf{A}P \quad \text{falls } \beta \neq \hat{\infty}$$

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\mathsf{A})P, \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : \mathsf{A}P$$

λP_ν:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : \lambda P \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : P_2$$
$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\lambda)P$$

$(\lambda)P_\nu$:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\lambda)P, \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,L \cup \{P_2\},\beta,\ldots]$$
$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : \lambda P \quad \text{falls } \beta \neq \hat{\infty}$$

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\lambda)P, \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : \lambda P$$

$$\frac{\alpha : \mathsf{A}P}{\alpha : P_2} \qquad \frac{\alpha : \lambda P}{\alpha : P_2}$$
$$\alpha : (\mathsf{A})P \qquad \alpha : (\lambda)P$$

$$\frac{\alpha : (\mathsf{A})P \quad \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ}{\lambda : [\ldots,\alpha,L \cup \{P_2,(\mathsf{A})P\},\beta,\ldots]} \qquad \frac{\alpha : (\mathsf{A})P \quad \lambda : [\ldots,\alpha,\circ,\beta,\ldots]}{\beta : \mathsf{A}P}$$
$$\beta : \mathsf{A}P \quad \text{falls } \beta \neq \hat{\infty}$$

$$\frac{\alpha : (\lambda)P \quad \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ}{\lambda : [\ldots,\alpha,L \cup \{P_2\},\beta,\ldots]} \qquad \frac{\alpha : (\lambda)P \quad \lambda : [\ldots,\alpha,\circ,\beta,\ldots]}{\beta : \lambda P}$$
$$\beta : \lambda P \quad \text{falls } \beta \neq \hat{\infty}$$

π:

$\mathsf{A}P_\pi$:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : \mathsf{A}P \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : P_0 \;\vee\; (\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : P_2 \;(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\mathsf{A})P$$

$(\mathsf{A})P_\pi$:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\mathsf{A})P,\; (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,L,\beta,\ldots],\; L \neq \circ \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,L \cup \{P_2,(\mathsf{A})P\},\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots]$$
$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \hat{\gamma}(\mathsf{free}(T)) : L \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \hat{\gamma}(\mathsf{free}(T)) : P_0 \quad \vee$$
$$\text{falls } \beta \neq \hat{\infty}\text{:} \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,L \cup \{P_2,(\mathsf{A})P\},\beta,\ldots] \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : \mathsf{A}P$$

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\mathsf{A})P,\; (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : \mathsf{A}P$$

λP_π:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : \lambda P \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : P_0 \;\vee\; (\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : P_2 \;(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\lambda)P$$

$(\lambda)P_\pi$:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\lambda)P,\; (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,L,\beta,\ldots],\; L \neq \circ \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,L \cup \{P_2\},\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots]$$
$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \hat{\gamma}(\mathsf{free}(T)) : L \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \hat{\gamma}(\mathsf{free}(T)) : P_0 \quad \vee$$
$$\text{falls } \beta \neq \hat{\infty}\text{:} \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,L \cup \{P_2\},\beta,\ldots] \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : \lambda P$$

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\lambda)P,\; (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : \lambda P$$

$$\frac{\alpha : \mathsf{A}P}{\alpha : P_0 \mid \alpha : P_2 \quad \alpha : (\mathsf{A})P} \qquad \frac{\alpha : \lambda P}{\alpha : P_0 \mid \alpha : P_2 \quad \alpha : (\lambda)P}$$

$$\frac{\begin{array}{c}\alpha : (\mathsf{A})P \\ \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ\end{array}}{\begin{array}{c|c}\lambda : [\ldots, \alpha, L \cup \{P_2, (\mathsf{A})P\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \text{falls } \beta \neq \hat{\infty}: \\ \hat{\gamma}(\mathsf{free}(T)) : L & \lambda : [\ldots, \alpha, L \cup \{P_2, (\mathsf{A})P\}, \beta, \ldots] \\ \hat{\gamma}(\mathsf{free}(T)) : P_0 & \beta : \mathsf{A}P\end{array}}$$

$$\frac{\begin{array}{c}\alpha : (\lambda)P \\ \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ\end{array}}{\begin{array}{c|c}\lambda : [\ldots, \alpha, L \cup \{P_2\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \text{falls } \beta \neq \hat{\infty}: \\ \hat{\gamma}(\mathsf{free}(T)) : L & \lambda : [\ldots, \alpha, L \cup \{P_2\}, \beta, \ldots] \\ \hat{\gamma}(\mathsf{free}(T)) : P_0 & \beta : \lambda P\end{array}}$$

$$\frac{\begin{array}{c}\alpha : (\mathsf{A})P \\ \lambda : [\ldots, \alpha, \circ, \beta, \ldots]\end{array}}{\beta : \mathsf{A}P} \qquad \frac{\begin{array}{c}\alpha : (\lambda)P \\ \lambda : [\ldots, \alpha, \circ, \beta, \ldots]\end{array}}{\beta : \lambda P}$$

ρ:

$\mathsf{A}P_\rho$:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : \mathsf{A}P \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : P_0 \;\vee\; (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : P_2 \;(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\mathsf{A})P$$

$(\mathsf{A})P_\rho$:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\mathsf{A})P,\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ \quad \Leftrightarrow$$

$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,L\cup\{P_2,(\mathsf{A})P\},\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots]$
$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \hat{\gamma}(\mathsf{free}(T)) : L$
$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \hat{\gamma}(\mathsf{free}(T)) : P_0$
$\vee$
falls $\beta \neq \hat{\infty}$:
$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,L\cup\{P_2,(\mathsf{A})P\},\beta,\ldots]$
$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \beta : \mathsf{A}P$
falls $\beta = \hat{\infty}$:
$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,L\cup\{P_2,(\mathsf{A})P\},\hat{\infty}]$

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\mathsf{A})P,\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \beta : \mathsf{A}P$$

λP_ρ:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : \lambda P \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : P_0 \;\vee\; (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : P_2 \;(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\lambda)P$$

$(\lambda)P_\rho$:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\lambda)P,\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ \quad \Leftrightarrow$$

$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,L\cup\{P_2\},\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots]$
$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \hat{\gamma}(\mathsf{free}(T)) : L$
$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \hat{\gamma}(\mathsf{free}(T)) : P_0$
$\vee$
falls $\beta \neq \hat{\infty}$:
$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,L\cup\{P_2\},\beta,\ldots]$
$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \beta : \lambda P$
falls $\beta = \hat{\infty}$:
$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,L\cup\{P_2\},\hat{\infty}]$

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\lambda)P,\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \beta : \lambda P$$

$$\frac{\alpha : \mathsf{A}P}{\alpha : P_0 \mid \alpha : P_2 \quad \alpha : (\mathsf{A})P} \qquad \frac{\alpha : \lambda P}{\alpha : P_0 \mid \alpha : P_2 \quad \alpha : (\lambda)P}$$

$$\frac{\begin{array}{c}\alpha : (\mathsf{A})P \\ \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ\end{array}}{\begin{array}{l|l} \lambda : [\ldots, \alpha, L \cup \{P_2, (\mathsf{A})P\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \text{falls } \beta \neq \hat{\infty}: \\ \hat{\gamma}(\mathsf{free}(T)) : L & \lambda : [\ldots, \alpha, L \cup \{P_2, (\mathsf{A})P\}, \beta, \ldots] \\ \hat{\gamma}(\mathsf{free}(T)) : P_0 & \beta : \mathsf{A}P \\ & \\ & \text{falls } \beta = \hat{\infty}: \\ & \lambda : [\ldots, \alpha, L \cup \{P_2, (\mathsf{A})P\}, \hat{\infty}] \end{array}}$$

$$\frac{\begin{array}{c}\alpha : (\lambda)P \\ \lambda : [\ldots, \alpha, L, \beta, \ldots],\ L \neq \circ\end{array}}{\begin{array}{l|l} \lambda : [\ldots, \alpha, L \cup \{P_2\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \text{falls } \beta \neq \hat{\infty}: \\ \hat{\gamma}(\mathsf{free}(T)) : L & \lambda : [\ldots, \alpha, L \cup \{P_2\}, \beta, \ldots] \\ \hat{\gamma}(\mathsf{free}(T)) : P_0 & \beta : \lambda P \\ & \\ & \text{falls } \beta = \hat{\infty}: \\ & \lambda : [\ldots, \alpha, L \cup \{P_2\}, \hat{\infty}] \end{array}}$$

$$\frac{\begin{array}{c}\alpha : (\mathsf{A})P \\ \lambda : [\ldots, \alpha, \circ, \beta, \ldots]\end{array}}{\beta : \mathsf{A}P} \qquad \frac{\begin{array}{c}\alpha : (\lambda)P \\ \lambda : [\ldots, \alpha, \circ, \beta, \ldots]\end{array}}{\beta : \lambda P}$$

Konsequenterweise wird bei Formeln der Formen $\mathsf{A}P_\nu$, $\mathsf{A}P_\pi$ und A_ρ die Liste jeweils um P_2 und $(\mathsf{A})P$ (anstelle P_2 und $\mathsf{A}P$) erweitert: P_2 beschreibt die Anforderungen an die auf dem Pfad liegenden Zustände, $(\mathsf{A})P$ die Fortpflanzung von P entlang eventueller Abzweige.

6.1 Die Regeln im Einzelnen

Analog zu Abschnitt 4.3 ergeben sich die Tableauregeln für die in CTL einzelnen Modalitäten:

$(\boldsymbol{K},\chi) \Vdash \alpha : \mathsf{A}\Box F$:

$$(\boldsymbol{K},\chi) \Vdash \alpha : \mathsf{A}\Box F \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash \alpha : F \quad (\boldsymbol{K},\chi) \Vdash \alpha : (\mathsf{A})\Box F$$

$$\frac{\alpha : \mathsf{A}\Box F}{\begin{array}{c}\alpha : F \\ \alpha : (\mathsf{A})\Box F\end{array}}$$

$(\boldsymbol{K},\chi) \Vdash \alpha : (\mathsf{A})\Box F$:

$$(\boldsymbol{K},\chi) \Vdash \alpha : (\mathsf{A})\Box F, \quad (\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,L\cup\{F,(\mathsf{A})\Box F\},\beta,\ldots] \quad (\boldsymbol{K},\chi) \Vdash \beta : \mathsf{A}\Box F \quad \text{falls } \beta \neq \hat{\infty}$$

$$(\boldsymbol{K},\chi) \Vdash \alpha : (\mathsf{A})\Box F, \quad (\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash \beta : \mathsf{A}\Box F$$

$$\frac{\begin{array}{c}\alpha : (\mathsf{A})\Box F \\ \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ\end{array}}{\begin{array}{c}\lambda : [\ldots,\alpha,L\cup\{F,(\mathsf{A})\Box F\},\beta,\ldots] \\ \beta : \mathsf{A}\Box F \quad \text{falls } \beta \neq \hat{\infty}\end{array}} \qquad \frac{\begin{array}{c}\alpha : (\mathsf{A})\Box F \\ \lambda : [\ldots,\alpha,\circ,\beta,\ldots]\end{array}}{\beta : \mathsf{A}\Box F}$$

$(\boldsymbol{K},\chi) \Vdash \alpha : \lambda\Box F$:

$$(\boldsymbol{K},\chi) \Vdash \alpha : \lambda\Box F \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash \alpha : F \quad (\boldsymbol{K},\chi) \Vdash \alpha : (\lambda)\Box F$$

$$\frac{\alpha : \lambda\Box F}{\begin{array}{c}\alpha : F \\ \alpha : (\lambda)\Box F\end{array}}$$

$(\boldsymbol{K},\chi) \Vdash \alpha : (\lambda)\Box F$:

$$(\boldsymbol{K},\chi) \Vdash \alpha : (\lambda)\Box F, \quad (\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,L\cup\{F\},\beta,\ldots] \quad (\boldsymbol{K},\chi) \Vdash \beta : \lambda\Box F \quad \text{falls } \beta \neq \hat{\infty}$$

$$(\boldsymbol{K},\chi) \Vdash \alpha : (\lambda)\Box F, \quad (\boldsymbol{K},\chi) \Vdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash \beta : \lambda\Box F$$

$$\frac{\begin{array}{c}a : (\lambda)\Box F, \\ \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ\end{array}}{\begin{array}{c}\lambda : [\ldots,\alpha,L\cup\{F\},\beta,\ldots] \\ b : \lambda\Box F \quad \text{falls } \beta \neq \hat{\infty}\end{array}} \qquad \frac{\begin{array}{c}a : (\lambda)\Box F \\ \lambda : [\ldots,\alpha,\circ,\beta,\ldots]\end{array}}{b : \lambda\Box F}$$

$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : \mathsf{A}\Diamond F$:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : \mathsf{A}\Diamond F \quad \Leftrightarrow \quad \begin{array}{l}(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : F\\ \vee\\ (\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : \neg F\\ (\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\mathsf{A})\Diamond F\end{array}$$

$\alpha : \mathsf{A}\Diamond F$	
$\alpha : F$	$\alpha : \neg F$
	$\alpha : (\mathsf{A})\Diamond F$

$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\mathsf{A})\Diamond F$:

$$\begin{array}{ll}(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\mathsf{A})\Diamond F, & \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha, L \cup \{\neg F, (\mathsf{A})\Diamond F\},\\ (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ & \qquad\qquad \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots]\\ & \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \hat{\gamma}(\mathsf{free}(T)) : L\\ & \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \hat{\gamma}(\mathsf{free}(T)) : F\\ & \quad \vee\\ & \quad \text{falls } \beta \neq \hat{\infty}:\\ & \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha, L \cup \{\neg F, (\mathsf{A})\Diamond F\}, \beta, \ldots]\\ & \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : \mathsf{A}\Diamond F\end{array}$$

$$\begin{array}{ll}(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\mathsf{A})\Diamond F, & \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : \mathsf{A}\Diamond F\\ (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & \end{array}$$

$$\frac{\begin{array}{c}\alpha : (\mathsf{A})\Diamond F\\ \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ\end{array}}{\begin{array}{l|l}\lambda : [\ldots,\alpha, L \cup \{\neg F, (\mathsf{A})\Diamond F\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \text{falls } \beta \neq \hat{\infty}:\\ \hat{\gamma}(\mathsf{free}(T)) : L & \lambda : [\ldots,\alpha, L \cup \{\neg F, (\mathsf{A})\Diamond F\}, \beta, \ldots]\\ \hat{\gamma}(\mathsf{free}(T)) : F & \beta : \mathsf{A}\Diamond F\end{array}}$$

$$\frac{\begin{array}{c}\alpha : (\mathsf{A})\Diamond F\\ \lambda : [\ldots,\alpha,\circ,\beta,\ldots]\end{array}}{\beta : \mathsf{A}\Diamond F}$$

$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : \lambda\Diamond F$:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : \lambda\Diamond F \quad \Leftrightarrow \quad \begin{array}{l}(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : F\\ \vee\\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : \neg F\\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\lambda)\Diamond F\end{array}$$

$\alpha : \lambda\Diamond F$	
$\alpha : F$	$\alpha : \neg F$
	$\alpha : (\lambda)\Diamond F$

$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\lambda)\Diamond F$:

$$\begin{array}{ll}(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\lambda)\Diamond F, & \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha, L\cup\{\neg F\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots]\\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha, L, \beta, \ldots],\ L \neq \circ & \quad\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \hat{\gamma}(\mathsf{free}(T)) : L\\ & \quad\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \hat{\gamma}(\mathsf{free}(T)) : F\\ & \quad\ \vee\\ & \quad\ \text{falls } \beta \neq \hat{\circ\!\!\!\!\infty}:\\ & \quad\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha, L\cup\{\neg F\}, \beta, \ldots]\\ & \quad\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \beta : \lambda\Diamond F\end{array}$$

$$\begin{array}{ll}(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\lambda)\Diamond F, & \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \beta : \lambda\Diamond F\\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & \end{array}$$

$\alpha : (\lambda)\Diamond F$ $\lambda : [\ldots,\alpha, L, \beta, \ldots],\ L \neq \circ$		$\alpha : (\lambda)\Diamond F$ $\lambda : [\ldots,\alpha,\circ,\beta,\ldots]$
$\lambda : [\ldots,\alpha, L\cup\{\neg F\}, \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots]$ $\hat{\gamma}(\mathsf{free}(T)) : L$ $\hat{\gamma}(\mathsf{free}(T)) : F$	falls $\beta \neq \hat{\circ\!\!\!\!\infty}$: $\lambda : [\ldots,\alpha, L\cup\{\neg F\}, \beta, \ldots]$ $\beta : \lambda\Diamond F$	$\beta : \lambda\Diamond F$

$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : \mathsf{A}\circ F$:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : \mathsf{A}\circ F \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\mathsf{A})\circ F$$

$\alpha : \mathsf{A}\circ F$
$\alpha : (\mathsf{A})\circ F$

$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : \lambda\circ F$:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : \lambda\circ F \quad \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\lambda)\circ F$$

$\alpha : \lambda\circ F$
$\alpha : (\lambda)\circ F$

$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\mathsf{A})\circ F$:

$$\begin{array}{ll} (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\mathsf{A})\circ F, & \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,\circ,\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] \\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ & \qquad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \hat{\gamma}(\mathsf{free}(T)) : L \\ & \qquad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \hat{\gamma}(\mathsf{free}(T)) : F \\ & \qquad \vee \\ & \qquad \text{falls } \beta \neq \hat{\circ}\text{:} \\ & \qquad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \\ & \qquad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \beta : F \end{array}$$

$$\begin{array}{ll} (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\mathsf{A})\circ F, & \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \beta : F \\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & \end{array}$$

$$\frac{\begin{array}{c}\alpha : (\mathsf{A})\circ F \\ \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ\end{array}}{\begin{array}{l|l}\lambda : [\ldots,\alpha,\circ,\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] & \text{falls } \beta \neq \hat{\circ}\text{:} \\ \hat{\gamma}(\mathsf{free}(T)) : L & \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \\ \hat{\gamma}(\mathsf{free}(T)) : F & \beta : F\end{array}} \qquad \frac{\begin{array}{c}\alpha : (\mathsf{A})\circ F \\ \lambda : [\ldots,\alpha,\circ,\beta,\ldots]\end{array}}{\beta : F}$$

$(\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\lambda)\circ F$:

$$\begin{array}{ll} (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\lambda)\circ F, & \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,\circ,\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] \\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ & \qquad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \hat{\gamma}(\mathsf{free}(T)) : L \\ & \qquad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \hat{\gamma}(\mathsf{free}(T)) : F \\ & \qquad \vee \\ & \qquad \text{falls } \beta \neq \hat{\circ}\text{:} \\ & \qquad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \\ & \qquad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \beta : F \end{array}$$

$$\begin{array}{ll} (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \alpha : (\lambda)\circ F, & \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \beta : F \\ (\boldsymbol{K},\chi) \Vdash\!\!\!\Vdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & \end{array}$$

$$\frac{\begin{array}{c}\alpha : (\lambda)\circ F \\ \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ\end{array}}{\begin{array}{l|l}\lambda : [\ldots,\alpha,\circ,\hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] & \text{falls } \beta \neq \hat{\circ}\text{:} \\ \hat{\gamma}(\mathsf{free}(T)) : L & \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \\ \hat{\gamma}(\mathsf{free}(T)) : F & \beta : F\end{array}} \qquad \frac{\begin{array}{c}\alpha : (\lambda)\circ F \\ \lambda : [\ldots,\alpha,\circ,\beta,\ldots]\end{array}}{\beta : F}$$

$(\boldsymbol{K},\chi) \nVdash \alpha : \mathsf{A}(F \text{ until } G)$:

$$(\boldsymbol{K},\chi) \nVdash \alpha : \mathsf{A}(F \text{ until } G) \quad \Leftrightarrow \quad \begin{array}{l} (\boldsymbol{K},\chi) \nVdash \alpha : G \\ \vee \\ (\boldsymbol{K},\chi) \nVdash \alpha : F \\ (\boldsymbol{K},\chi) \nVdash \alpha : \neg G \\ (\boldsymbol{K},\chi) \nVdash \alpha : (\mathsf{A})(F \text{ until } G) \end{array}$$

$\alpha : \mathsf{A}(F \text{ until } G)$	
$\alpha : G$	$\alpha : F$
	$\alpha : \neg G$
	$\alpha : (\mathsf{A})(F \text{ until } G)$

$(\boldsymbol{K},\chi) \nVdash \alpha : (\mathsf{A})(F \text{ until } G)$:

$$\begin{array}{ll} (\boldsymbol{K},\chi) \nVdash \alpha : (\mathsf{A})(F \text{ until } G), & \Leftrightarrow \quad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})(F \text{ until } G)\}, \\ (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ & \qquad\qquad \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] \\ & \quad (\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : L \\ & \quad (\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : G \\ & \quad \vee \\ & \quad \text{falls } \beta \neq \hat{\circ\!\!\circ} : \\ & \quad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})(F \text{ until } G)\}, \beta, \ldots] \\ & \quad (\boldsymbol{K},\chi) \nVdash \beta : \mathsf{A}(F \text{ until } G) \end{array}$$

$$\begin{array}{ll} (\boldsymbol{K},\chi) \nVdash \alpha : (\mathsf{A})(F \text{ until } G), & \Leftrightarrow \quad (\boldsymbol{K},\chi) \nVdash \beta : \mathsf{A}(F \text{ until } G) \\ (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & \end{array}$$

$$\frac{\begin{array}{c} \alpha : (\mathsf{A})(F \text{ until } G) \\ \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ \end{array}}{\begin{array}{c|c} \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})(F \text{ until } G)\}, & \text{falls } \beta \neq \hat{\circ\!\!\circ} : \\ \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})(F \text{ until } G)\}, \beta, \ldots] \\ \hat{\gamma}(\mathsf{free}(T)) : L & \beta : \mathsf{A}(F \text{ until } G) \\ \hat{\gamma}(\mathsf{free}(T)) : G & \end{array}}$$

$$\frac{\begin{array}{c} \alpha : (\mathsf{A})(F \text{ until } G) \\ \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \end{array}}{\beta : \mathsf{A}(F \text{ until } G)}$$

$(\boldsymbol{K},\chi) \nVdash \alpha : \mathsf{A}\neg(F \text{ until } G)$:

$$(\boldsymbol{K},\chi) \nVdash \alpha : \mathsf{A}\neg(F \text{ until } G) \quad \Leftrightarrow \quad \begin{array}{l} (\boldsymbol{K},\chi) \nVdash \alpha : \neg G \\ (\boldsymbol{K},\chi) \nVdash \alpha : \neg F \\ \vee \\ (\boldsymbol{K},\chi) \nVdash \alpha : F \\ (\boldsymbol{K},\chi) \nVdash \alpha : \neg G \\ (\boldsymbol{K},\chi) \nVdash \alpha : (\mathsf{A})\neg(F \text{ until } G) \end{array}$$

$\alpha : \mathsf{A}\neg(F \text{ until } G)$	
$\alpha : \neg G$	$\alpha : F$
$\alpha : \neg F$	$\alpha : \neg G$
	$\alpha : (\mathsf{A})\neg(F \text{ until } G)$

$(\boldsymbol{K},\chi) \nVdash \alpha : (\mathsf{A})\neg(F \text{ until } G)$:

$$\begin{array}{ll} (\boldsymbol{K},\chi) \nVdash \alpha : (\mathsf{A})\neg(F \text{ until } G), & \Leftrightarrow \quad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})\neg(F \text{ until } G)\}, \\ (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ & \qquad\qquad \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] \\ & \quad (\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : L \\ & \quad (\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : \neg F \\ & \quad (\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : \neg G \\ & \quad \vee \\ & \quad \text{falls } \beta \neq \hat{\infty} : \\ & \quad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})\neg(F \text{ until } G)\}, \beta, \ldots] \\ & \quad (\boldsymbol{K},\chi) \nVdash \beta : \mathsf{A}\neg(F \text{ until } G) \\ & \quad \vee \\ & \quad \text{falls } \beta = \hat{\infty} : \\ & \quad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})\neg(F \text{ until } G)\}, \hat{\infty}] \end{array}$$

$$\begin{array}{ll} (\boldsymbol{K},\chi) \nVdash \alpha : (\mathsf{A})\neg(F \text{ until } G), & \Leftrightarrow \quad (\boldsymbol{K},\chi) \nVdash \beta : \mathsf{A}\neg(F \text{ until } G) \\ (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & \end{array}$$

$$\frac{\begin{array}{c} \alpha : (\mathsf{A})\neg(F \text{ until } G) \\ \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ \end{array}}{\begin{array}{l|l} \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})\neg(F \text{ until } G)\}, & \text{falls } \beta \neq \hat{\infty} : \\ \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})\neg(F \text{ until } G)\}, \beta, \ldots] \\ \hat{\gamma}(\mathsf{free}(T)) : L & \beta : \mathsf{A}\neg(F \text{ until } G) \\ \hat{\gamma}(\mathsf{free}(T)) : \neg F & \\ \hat{\gamma}(\mathsf{free}(T)) : \neg G & \text{falls } \beta = \hat{\infty} : \\ & \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})\neg(F \text{ until } G)\}, \hat{\infty}] \end{array}}$$

$$\frac{\begin{array}{c} \alpha : (\mathsf{A})\neg(F \text{ until } G) \\ \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \end{array}}{\beta : \mathsf{A}\neg(F \text{ until } G)}$$

$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : \mathsf{A}(F \ \mathsf{unless} \ G)$:

$$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : \mathsf{A}(F \ \mathsf{unless} \ G) \quad \Leftrightarrow \quad \begin{array}{l} (\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : G \\ \vee \\ (\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : F \\ (\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : \neg G \\ (\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\mathsf{A})(F \ \mathsf{unless} \ G) \end{array}$$

$\alpha : \mathsf{A}(F \ \mathsf{unless} \ G)$	
$\alpha : G$	$\alpha : F$
	$\alpha : \neg G$
	$\alpha : (\mathsf{A})(F \ \mathsf{unless} \ G)$

$(\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\mathsf{A})(F \ \mathsf{unless} \ G)$:

$$\begin{array}{ll} (\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\mathsf{A})(F \ \mathsf{unless} \ G), & \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})(F \ \mathsf{unless} \ G)\}, \\ (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,L,\beta,\ldots], \ L \neq \circ & \qquad\qquad \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] \\ & \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \hat{\gamma}(\mathsf{free}(T)) : L \\ & \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \hat{\gamma}(\mathsf{free}(T)) : G \\ & \quad \vee \\ & \quad \text{falls } \beta \neq \hat{\infty} : \\ & \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})(F \ \mathsf{unless} \ G)\}, \beta, \ldots] \\ & \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : \mathsf{A}(F \ \mathsf{unless} \ G) \\ & \quad \vee \\ & \quad \text{falls } \beta = \hat{\infty} : \\ & \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})(F \ \mathsf{unless} \ G)\}, \hat{\infty}] \end{array}$$

$$\begin{array}{ll} (\boldsymbol{K},\chi) \Vdash\!\!\!\models \alpha : (\mathsf{A})(F \ \mathsf{unless} \ G), & \Leftrightarrow \quad (\boldsymbol{K},\chi) \Vdash\!\!\!\models \beta : \mathsf{A}(F \ \mathsf{unless} \ G) \\ (\boldsymbol{K},\chi) \Vdash\!\!\!\models \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & \end{array}$$

$$\frac{\begin{array}{c} \alpha : (\mathsf{A})(F \ \mathsf{unless} \ G) \\ \lambda : [\ldots,\alpha,L,\beta,\ldots], \ L \neq \circ \end{array}}{\begin{array}{l|l} \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})(F \ \mathsf{unless} \ G)\}, & \text{falls } \beta \neq \hat{\infty} : \\ \quad \hat{\gamma}(\mathsf{free}(T)), L, \beta, \ldots] & \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})(F \ \mathsf{unless} \ G)\}, \beta, \ldots] \\ \quad \hat{\gamma}(\mathsf{free}(T)) : L & \beta : \mathsf{A}(F \ \mathsf{unless} \ G) \\ \quad \hat{\gamma}(\mathsf{free}(T)) : G & \\ & \text{falls } \beta = \hat{\infty} : \\ & \lambda : [\ldots,\alpha, L \cup \{F, \neg G, (\mathsf{A})(F \ \mathsf{unless} \ G)\}, \hat{\infty}] \end{array}}$$

$$\frac{\begin{array}{c} \alpha : (\mathsf{A})(F \ \mathsf{unless} \ G) \\ \lambda : [\ldots,\alpha,\circ,\beta,\ldots] \end{array}}{\beta : \mathsf{A}(F \ \mathsf{unless} \ G)}$$

$(\boldsymbol{K},\chi) \nVdash \alpha : \mathsf{A}\neg(F \ \ \mathsf{unless} \ \ G)$:

$$(\boldsymbol{K},\chi) \nVdash \alpha : \mathsf{A}\neg(F \ \ \mathsf{unless} \ \ G) \quad \Leftrightarrow \quad \begin{array}{l}(\boldsymbol{K},\chi) \nVdash \alpha : \neg F\\ (\boldsymbol{K},\chi) \nVdash \alpha : \neg G\\ \vee\\ (\boldsymbol{K},\chi) \nVdash \alpha : F\\ (\boldsymbol{K},\chi) \nVdash \alpha : \neg G\\ (\boldsymbol{K},\chi) \nVdash \alpha : (\mathsf{A})\neg(F \ \ \mathsf{unless} \ \ G)\end{array}$$

$\alpha : \mathsf{A}\neg(F \ \ \mathsf{unless} \ \ G)$	
$\alpha : \neg F$	$\alpha : F$
$\alpha : \neg G$	$\alpha : \neg G$
	$\alpha : (\mathsf{A})\neg(F \ \ \mathsf{unless} \ \ G)$

$(\boldsymbol{K},\chi) \nVdash \alpha : (\mathsf{A})\neg(F \ \ \mathsf{unless} \ \ G)$:

$$\begin{array}{ll}(\boldsymbol{K},\chi) \nVdash \alpha : (\mathsf{A})(F \ \ \mathsf{unless} \ \ G), & \Leftrightarrow \quad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha, L \cup \{F,\neg G,(\mathsf{A})\neg(F \ \ \mathsf{unless} \ \ G)\},\\ (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ & \qquad\qquad \hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots]\\ & \quad (\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : L\\ & \quad (\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : \neg F\\ & \quad (\boldsymbol{K},\chi) \nVdash \hat{\gamma}(\mathsf{free}(T)) : \neg G\\ & \quad \vee\\ & \quad \text{falls } \beta \neq \hat{\infty} :\\ & \quad (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha, L \cup \{F,\neg G,(\mathsf{A})\neg(F \ \ \mathsf{unless} \ \ G)\},\beta,\ldots]\\ & \quad (\boldsymbol{K},\chi) \nVdash \beta : \mathsf{A}\neg(F \ \ \mathsf{unless} \ \ G)\end{array}$$

$$\begin{array}{ll}(\boldsymbol{K},\chi) \nVdash \alpha : (\mathsf{A})\neg(F \ \ \mathsf{unless} \ \ G), & \Leftrightarrow \quad (\boldsymbol{K},\chi) \nVdash \beta : \mathsf{A}\neg(F \ \ \mathsf{unless} \ \ G)\\ (\boldsymbol{K},\chi) \nVdash \lambda : [\ldots,\alpha,\circ,\beta,\ldots] & \end{array}$$

$$\frac{\begin{array}{c}\alpha : (\mathsf{A})\neg(F \ \ \mathsf{unless} \ \ G)\\ \lambda : [\ldots,\alpha,L,\beta,\ldots],\ L \neq \circ\end{array}}{\begin{array}{l|l}\lambda : [\ldots,\alpha, L \cup \{F,\neg G,(\mathsf{A})\neg(F \ \ \mathsf{unless} \ \ G)\}, & \text{falls } \beta \neq \hat{\infty} :\\ \quad \hat{\gamma}(\mathsf{free}(T)),L,\beta,\ldots] & \lambda : [\ldots,\alpha, L \cup \{F,\neg G,(\mathsf{A})\neg(F \ \ \mathsf{unless} \ \ G)\},\beta,\ldots]\\ \quad \hat{\gamma}(\mathsf{free}(T)) : L & \quad \beta : \mathsf{A}\neg(F \ \ \mathsf{unless} \ \ G)\\ \quad \hat{\gamma}(\mathsf{free}(T)) : \neg F & \\ \quad \hat{\gamma}(\mathsf{free}(T)) : \neg G & \end{array}}$$

$$\frac{\begin{array}{c}\alpha : (\mathsf{A})\neg(F \ \ \mathsf{unless} \ \ G)\\ \lambda : [\ldots,\alpha,\circ,\beta,\ldots]\end{array}}{\beta : \mathsf{A}\neg(F \ \ \mathsf{unless} \ \ G)}$$

Dabei werden jetzt die Menge F_α der das momentane Präfix α betreffenden Formeln und die Liste L der auf dem darauffolgenden Pfadabschnitt geltenden Formeln zu getrennten Zeitpunkten erweitert: F_α bei Auflösung der A- bzw. λ-Formel, L erst bei Auflösung der (A)- bzw. (λ)-Formel.

Die Regeln zur Instantiierung von Zuständen aus den Pfadinformationsformeln werden ebenfalls beibehalten:

$$\frac{\lambda : [\ldots, \alpha, L, \beta, \ldots], L \neq \circ \qquad \beta \neq \hat{\infty}}{\lambda : [\ldots, \alpha, L, \hat{\gamma}(\mathsf{free}(T)), \circ, \beta, \ldots] \mid \lambda : [\ldots, \alpha, \circ, \beta, \ldots]}$$
$$\hat{\gamma}(\mathsf{free}(T)) : L$$

$$\frac{\lambda : [\ldots, \alpha, L, \hat{\infty}], \ L \neq \emptyset}{\lambda : [\ldots, \alpha, L, \hat{\gamma}(\mathsf{free}(T)), L, \hat{\infty}], \quad \hat{\gamma}(\mathsf{free}(T)) : L}$$

Da die Hintereinanderausführung der passenden Regeln jeweils einer Regelanwendung der theoretischen Version entspricht, übertragen sich die Eigenschaften von Abschnitt 5 bei entsprechender Erweiterung der Fairness der Regelauswahl.

7 Erweiterung auf CTL$^+$

In diesem Kapitel wird eine Erweiterung für CTL$^+$ *beschrieben. Dabei wird jeweils untersucht, welche Konstrukte dabei echte Probleme für den bestehenden Kalkül darstellen und Lösungswege angegeben. Dabei wird verwendet, daß die Ausdruckskraft von* CTL *gleich derjenigen von* CTL$^+$ *ist.*

In CTL$^+$ werden durch die erweiterte Syntaxdefinition P3$^+$ (S. 20) zusätzlich Konjunktionen und Disjunktionen von CTL$^+$-Pfadformeln als CTL$^+$-Pfadformeln zugelassen. In der tableauinternen Syntax drückt sich dies in der Definition (TP3) aus:

(A) Jedes Atom ist eine $\mathcal{TK}$-Zustandsformel.

(TZ1) Sind F und G $\mathcal{TK}$-Zustandsformeln, so sind auch $\neg F$, $F \wedge G$, $F \vee G$ und $F \rightarrow G$ $\mathcal{TK}$-Zustandsformeln.

(TZQ) Ist F eine $\mathcal{TK}$-Zustandsformel und x eine Variable, so sind auch $\forall x : F$ und $\exists x : F$ $\mathcal{TK}$-Zustandsformeln.

(TP1) Sind F und G $\mathcal{TK}$-Zustandsformeln, so sind $\circ F$, $\Box F$, $\Diamond F$ und $(F \,\mathsf{until}\, G)$ $\mathcal{TK}$-Pfadformeln.

(TP2) Ist P eine $\mathcal{TK}$-Pfadformel, so ist $\neg P$ eine $\mathcal{TK}$-Pfadformel.

(TP3) Sind P und Q $\mathcal{TK}$-Pfadformeln, so sind $P \wedge Q$ und $P \vee Q$ $\mathcal{TK}$-Pfadformeln.

(TZ2) Ist P eine $\mathcal{TK}$-Pfadformel, so sind $\mathsf{A}P$ und $\mathsf{E}P$ $\mathcal{TK}$-Zustandsformeln.

(TC1) Jede $\mathcal{TK}$-Zustandsformel ist eine $\mathcal{TK}$-Prä-Knotenformel

(TC2) Ist P eine $\mathcal{TK}$-Pfadformel und λ ein Pfadselektor, so ist λP eine $\mathcal{TK}$-Prä-Knotenformel.

(TC3) Ist P eine $\mathcal{TK}$-Pfadformel und λ ein Pfadbezeichner, so sind $(\mathsf{A})P$ und $(\lambda)P$ $\mathcal{TK}$-Prä-Knotenformeln.

(TK1) Alle Pfadinformationsformeln sind $\mathcal{TK}$-Knotenformeln.

(TK2) Ist F eine $\mathcal{TK}$-Prä-Knotenformel und γ ein Präfix, so ist $\gamma : F$ eine $\mathcal{TK}$-Knotenformel.

Mit der bisher angewandten Vorgehensweise sind die neu hinzugekommenen Formeln – bis auf einen Fall – unproblematisch:

Die Auflösung von Formeln der Form $\gamma : \mathsf{E}P$, erfolgt weiterhin indem P an einen neu benannten Pfadbezeichner gebunden wird:

$$\frac{\gamma : \mathsf{E}P}{\begin{array}{c}\hat{\kappa}(\mathsf{free}(T)) : [\gamma, \emptyset, \hat{\infty}] \\ \gamma : \hat{\kappa}(\mathsf{free}(T))\ P\end{array}}$$

wobei $\hat{\kappa}$ ein noch nicht vorkommendes Pfadsymbol ist.

Für die Auflösung von an einen Pfadbezeichner gebundenen disjunktiv oder konjunktiv verknüpften Pfadformeln werden die folgenden Regeln hinzugenommen:

Seien P, Q Pfadformeln:

$$\frac{\alpha : \lambda(P \wedge Q)}{\alpha : \lambda\, P \quad \alpha : \lambda\, Q} \qquad \frac{\alpha : \lambda\neg(P \vee Q)}{\alpha : \lambda\, \neg P \quad \alpha : \lambda\, \neg Q}$$

$$\frac{\alpha : \lambda(P \vee Q)}{\alpha : \lambda\, P \mid \alpha : \lambda\, Q} \qquad \frac{\alpha : \lambda\neg(P \wedge Q)}{\alpha : \lambda\, \neg P \mid \alpha : \lambda\, \neg Q}$$

Im Fall universell quantifizierter Pfade löst man Formeln konjunktiven Typs gemäß der Äquivalenz $\mathsf{A}(P \wedge Q) \equiv \mathsf{A}P \wedge \mathsf{A}Q$ auf:

$$\frac{\alpha : \mathsf{A}(P \wedge Q)}{\alpha : \mathsf{A}P \quad \alpha : \mathsf{A}Q} \qquad \frac{\alpha : \mathsf{A}\neg(P \vee Q)}{\alpha : \mathsf{A}\neg P \quad \alpha : \mathsf{A}\neg Q}$$

Einzig im Fall universell quantifizierter Pfadformeln disjunktiven Typs ist das Vorgehen nicht anwendbar.

An dieser Stelle wird die aus [EH82, EH83] bekannte Tatsache, daß die Ausdruckskraft von CTL gleich derjenigen von CTL$^+$ ist, ausgenutzt. In [EH82] wird ein Verfahren angegeben, wie beliebige CTL$^+$-Formeln in CTL-Formeln umgeformt werden können:

Aufgrund der Syntaxdefinition (Seite 17) kann man oEdA. annehmen, daß die definierten Zeichen A, $\Diamond$ und $\Box$ in der umzuformenden Formel nicht vorkommen ($\mathsf{A}Q \equiv \neg\mathsf{E}Q$ und $\Diamond Q \equiv \mathsf{true\ until}\ Q$ und $\Box Q \equiv \neg\Diamond\neg Q$). Es genügt, das Verfahren für Formeln der Form $\mathsf{E}P$, wobei P eine CTL$^+$-Pfadformel ist, anzugeben. Durch rekursive Anwendung auf die in P enthaltenen Teilformeln wird so die gesamte Formel transformiert. Die zu transformierende Formel ist also oEdA. von der Form $\mathsf{E}Q$, wobei Q eine boolesche Kombination von Formeln der Formen $F\ \mathsf{until}\ G$, $\neg(F\ \mathsf{until}\ G)$, $\circ F$ und $\neg\circ F$ ist.

Dabei werden die folgenden Äquivalenzen verwendet (wie immer bezeichnen F und G Zustandsformeln, P und Q Pfadformeln):

(1) $\neg\circ F \equiv \circ\neg F$

(2) $\neg(F\ \mathsf{until}\ G) \equiv ((F \wedge \neg G)\ \mathsf{until}\ (\neg F \wedge \neg G)) \vee \Box\neg G$

(3) $\mathsf{E}(P \vee Q) \equiv \mathsf{E}P \vee \mathsf{E}Q$

(4) $\circ(F \wedge G) \equiv \circ F \wedge \circ G$

(5) $\Box(F \wedge G) \equiv \Box F \wedge \Box G$

(6) $\mathsf{E}((\bigwedge_{i=1\ldots n}(F_i\ \mathsf{until}\ G_i)) \wedge \circ F \wedge \Box G) \equiv$
$\equiv \bigvee_{I\subseteq\{1\ldots n\}}((\bigwedge_{i\in I} G_i) \wedge G \wedge \mathsf{E}\circ(F \wedge \mathsf{E}((\bigwedge_{i\notin I} F_i\ \mathsf{until}\ G_i) \wedge \Box G)))$

(7) $\mathsf{E}((\bigwedge_{i=1\ldots n}(F_i\ \mathsf{until}\ G_i)) \wedge \Box G) \equiv$
$\equiv \bigvee_{\pi\in\mathrm{Perm}(\{1\ldots n\})}(\mathsf{E}((\bigwedge_{i\in I} F_i \wedge G)\ \mathsf{until}$
$(G_{\pi(1)} \wedge \mathsf{E}(((\bigwedge_{i\neq\pi(1)} F_i) \wedge G)\ \mathsf{until}$
$(G_{\pi(2)} \wedge \mathsf{E}(((\bigwedge_{i\neq\pi(1),\pi(2)} G_i) \wedge G)\ \mathsf{until}$
$(\ldots\ \mathsf{until}\ (G_{\pi(n)} \wedge \mathsf{E}\Box G)))))))$

Die rechte Seite in (7) beschreibt dabei alle Möglichkeiten, wie die vorkommenden until-Forderungen nacheinander erfüllt werden können. Nach [EH82] gilt bei einer Ausgangsformel der Länge n für die Länge n' der erhaltenen Formel $n' \leq 2^{(n \log n)}$.

Durch eine Anwendung dieser Regeln kann eine beliebige Formel der Form $\mathsf{A/E}(\mathsf{mop}\ P\ \mathsf{op}\ \mathsf{mop}\ Q)$, in eine oder mehrere Formeln der Form $\mathsf{A/E}\ \mathsf{mop}\ R$ (wobei P, Q, R CTL$^+$-Formeln sind) umgeformt werden. Rekursive Anwendung des Verfahrens auf R liefert schlußendlich eine CTL-Formel.

Nach obigen Feststellungen benötigt man die Umformung nur für universell quantifizierte Pfadformeln disjunktiven Typs. Weitere Fortschritte bringen die Äquivalenzen

$$\mathsf{A}(\circ F \vee \circ G) \equiv \mathsf{A}\circ(F \vee G) \quad \text{und} \quad \mathsf{A}(\Diamond F \vee \Diamond G) \equiv \mathsf{A}\Diamond(F \vee G) \quad .$$

Man kann also oEdA. davon ausgehen, daß die zu betrachtende Disjunktion höchstens jeweils einen Ausdruck der Form $\circ F$ und $\Diamond F$ enthält. Nach Anwendung von Regel (6) kann weiter angenommen werden, daß kein Ausdruck der Form $\circ F$ vorhanden ist.

Beispiel:

$$\begin{aligned}
&\mathsf{A}(\Diamond F \vee \bigvee \Box G_j \vee \bigvee (H_k\ \mathsf{until}\ K_k) \vee \bigvee \neg (L_l\ \mathsf{until}\ M_l)) \\
=\ & \neg\mathsf{E}\neg(\Diamond F \vee \bigvee \Box G_j \vee \bigvee (H_k\ \mathsf{until}\ K_k) \vee \bigvee \neg (L_l\ \mathsf{until}\ M_l)) \\
=\ & \neg\mathsf{E}(\neg\Diamond F \wedge \bigwedge \neg\Box G_j \wedge \bigwedge \neg (H_k\ \mathsf{until}\ K_k) \wedge \bigwedge (L_l\ \mathsf{until}\ M_l)) \\
\overset{(2)}{=}\ & \neg\mathsf{E}(\Box\neg F \wedge \bigwedge(\mathsf{true}\ \mathsf{until}\ \neg G_j) \wedge \bigwedge (L_l\ \mathsf{until}\ M_l) \wedge \\
& \quad \bigwedge (((H_k \wedge \neg K_k)\ \mathsf{until}\ (\neg H_k \wedge K_k)) \vee \Box\neg K_k))
\end{aligned}$$

in disjunktive Normalform überführen (betrifft die Konjunktion mit Indizes k):

$$\begin{aligned}
=\ & \neg\mathsf{E}(\textstyle\bigvee_{I \subset \{1\ldots k\}} (\Box\neg F \wedge \bigwedge(\mathsf{true}\ \mathsf{until}\ \neg G_j) \wedge \bigwedge (L_l\ \mathsf{until}\ M_l) \wedge \\
& \qquad \underbrace{\textstyle\bigwedge_{k \in I} ((H_k \wedge \neg K_k)\ \mathsf{until}\ (\neg H_k \wedge K_k)) \wedge \bigwedge_{k \notin I} \Box\neg K_k))}_{=: \bigwedge A_m} \\
\overset{(3)}{=}\ & \neg(\textstyle\bigvee_I \mathsf{E}(\bigwedge A_m))
\end{aligned}$$

mit (7) umformen:

$$\begin{aligned}
=\ & \neg\left(\textstyle\bigvee_I \bigvee_{\pi \in \mathsf{Perm}(\{1\ldots n\})} (\mathsf{E}B_{I,\pi})\right) \qquad \text{für geeignete } B_{I,\pi}. \\
=\ & \textstyle\bigwedge_I \bigwedge_\pi \neg(\mathsf{E}B_{I,\pi}) \\
=\ & \textstyle\bigwedge_I \bigwedge_\pi \mathsf{A}(\neg B_{I,\pi})
\end{aligned}$$

Für die Konstruktion eines Kalküls muß diese Umformung für einen Tableauschritt nur auf der obersten Ebene der Modaloperatoren, und auch nur im Fall universell quantifizierter Pfadformeln disjunktiven Typs durchgeführt werden:

Es bezeichne $f(P, Q)$ die durch obige Umformungen aus $P \vee Q$ enthaltene Formel.

$$\frac{\alpha : \mathsf{A}(P \vee Q)}{\alpha : \mathsf{A}f(P, Q)}$$

Für feste CTL$^+$-Formelschemata lassen sich jedoch – analog zu unless – einfache Tableauregeln definieren. Andererseits ist damit – wie festgestellt – an Ausdruckskraft nichts gewonnen.

8 Einbindung von Logik linearer Zeit

In diesem Abschnitt wird die Einbettung von Formeln linearer Zeit untersucht. Dabei zeigt sich, daß man bei Betrachtung des Verlaufes einzelner Pfade auch kompliziertere Formeln zulassen kann – insbesondere, daß Fairnessanforderungen verarbeitet werden können. Danach wird ein Blick auf die Ausdruckskraft des Kalküls bzw. der damit zu verarbeitenden Formeln geworfen.

8.1 Einbettung der Temporallogik linearer Zeit in $\mathcal{TK}$

Die Logik PTL läßt sich, wenn man die Aussagen jeweils auf *einen* Pfad bezieht, in den Kalkül $\mathcal{TK}$ einbetten. Durch die Veränderung (TC2) $\rightsquigarrow$ (TZ3) werden die Pfadselektoren $\lambda, \kappa, \ldots$ syntaktisch den Pfadquantoren gleichgestellt, können also auch innerhalb von Formeln auftreten:

(A) Jedes Atom ist eine $\mathcal{TK}$-Zustandsformel.

(TZ1) Sind F und G $\mathcal{TK}$-Zustandsformeln, so sind auch $\neg F$, $F \wedge G$, $F \vee G$ und $F \rightarrow G$ $\mathcal{TK}$-Zustandsformeln.

(TZQ) Ist F eine $\mathcal{TK}$-Zustandsformel und x eine Variable, so sind auch $\forall x : F$ und $\exists x : F$ $\mathcal{TK}$-Zustandsformeln.

(TP1) Sind F und G $\mathcal{TK}$-Zustandsformeln, so sind $\circ F$, $\Box F$, $\Diamond F$ und $(F \,\mathsf{until}\, G)$ $\mathcal{TK}$-Pfadformeln.

(TP2) Ist P eine $\mathcal{TK}$-Pfadformel, so ist $\neg P$ eine $\mathcal{TK}$-Pfadformel.

(TP3) Sind P und Q $\mathcal{TK}$-Pfadformeln, so sind $P \wedge Q$ und $P \vee Q$ $\mathcal{TK}$-Pfadformeln.

(TZ2) Ist P eine $\mathcal{TK}$-Pfadformel, so sind $\mathsf{A}P$ und $\mathsf{E}P$ $\mathcal{TK}$-Zustandsformeln.

(TC1) Jede $\mathcal{TK}$-Zustandsformel ist eine $\mathcal{TK}$-Prä-Knotenformel

(TZ3) Ist P eine $\mathcal{TK}$-Pfadformel und λ ein Pfadbezeichner, so ist λP eine $\mathcal{TK}$-Zustandsformel.

(TC3) Ist P eine $\mathcal{TK}$-Pfadformel und λ ein Pfadbezeichner, so sind $(\mathsf{A})P$ und $(\lambda)P$ $\mathcal{TK}$-Prä-Knotenformeln.

(TK1) Alle Pfadinformationsformeln sind $\mathcal{TK}$-Knotenformeln.

(TK2) Ist F eine $\mathcal{TK}$-Prä-Knotenformel und γ ein Zustandsbezeichner, so ist $\gamma : F$ eine $\mathcal{TK}$-Knotenformel.

Bei Verwendung einer PTL-Formel P soll diese als $\gamma : \lambda P$ für $\gamma \in \Gamma$, $\lambda \in \Lambda$ in das Tableau aufgenommen werden. Falls in ihr geschachtelte Modaloperatoren vorkommen, genügt sie noch nicht der o.g. Syntaxdefinition.

Definition 36 *Ein Auftreten eines Modaloperators* mop *in einer* CTL**-Pfadformel P heißt* geschützt, *wenn im Syntaxbaum von P zwischen ihm und der Wurzel ein Pfadquantor steht. Der führende Modaloperator ist geschützt.*

Anm.: Bei der hier an dieser Stelle betrachteten Einbindung von reinen PTL-Formeln gibt es noch keine geschützten Modaloperatoren. Diese treten erst im Zuge der Betrachtung von CTL* auf, wo diese Definition ebenfalls verwendet wird.

Definition 37 *Für eine* CTL**-Pfadformel* P *(vgl. Definition: in* CTL* *sind alle Zustandsformeln zugleich auch Pfadformeln) und einen Pfadbezeichner* λ *sei* $P[\lambda]$ *die Formel, die entsteht, wenn unmittelbar vor jeden in* P *nicht geschützt stehenden Modaloperator* λ *eingefügt wird.*

Für eine CTL-Pfadformel P ist $\lambda P = P[\lambda]$.

Die Bildung von $P[\lambda]$ kann folgendendermaßen rekursiv beschrieben werden: Sei op eine prädikatenlogische Verknüpfung, mop ein Modaloperator, A eine atomare prädikatenlogische Formel, P_1 und P_2 $\mathcal{TK}$-Pfadformeln.

P	$P[\lambda]$
$\mathsf{mop}\ P_1$	$\lambda\ \mathsf{mop}\ P_1[\lambda]$
$(P_1\ \mathsf{mop}\ P_2)$	$\lambda((P_1[\lambda])\ \mathsf{mop}\ (P_2[\lambda]))$
$\mathsf{op}\ P_1$	$\mathsf{op}\ P_1[\lambda]$
$(P_1\ \mathsf{op}\ P_2)$	$(P_1[\lambda]\ \mathsf{op}\ P_2[\lambda])$
$\mathsf{A}P_1$	$\mathsf{A}P_1$
$\mathsf{E}P_1$	$\mathsf{E}P_1$
A	A

Die Umformung wird durch die Regel

$$\frac{\gamma : \lambda P}{\gamma : P[\lambda]}$$

in den Kalkül integriert.

Damit lassen sich für *einzelne* Pfade auch komplexere Pfadformeln verarbeiten.

8.2 Erweiterung für Fairness

Zur Beschreibung von Fairnesseigenschaften wird eine spezielle Klasse von Pfadformeln benötigt, die nicht in CTL-Syntax darstellbar ist. Diese wird in Anlehnung an [MP92] (vgl. auch Abschnitt 8.3) als „Reactivity“ bezeichnet und ist in CTL* als

$$\Box\Diamond F \vee \Diamond\Box G \quad ,$$

wobei F und G Zustandsformeln sind, darstellbar.

Bereits in der Formulierung der Definition „Eine Berechnungsfolge ist fair, wenn ...“ wird deutlich, daß Fairness in erster Linie eine Eigenschaft einer Berechnungsfolge bzw. des entsprechenden Pfades ist. Sie wird erst im unendlichen Bereich eines Pfades interessant, weshalb sie auch dem Tableaukalkül aus [BMP81] nicht zugänglich sein kann. In dem hier vorliegenden Kalkül ist es aber genau diese Eigenschaft, die ihre Behandlung erst möglich macht:

Sie beschreibt, daß es – bezüglich einer bestimmten Fragestellung – entlang einem Pfad in Richtung unendlich entweder irgendwann keine Veränderung mehr gibt, oder statuiert die Existenz eines sich bezüglich dieser Fragestellung von seinem Vorgänger unterscheidenden Zustands.

Dies ist eine typische Eigenschaft der Reactivity-Formeln:

Definition 38 *Ist $p = (g_0, g_1, \ldots)$ ein Pfad in einer Kripke-Struktur $\boldsymbol{K}$ und P eine Pfadformel und gilt für alle $n \in \mathbf{N}$:*

$$(\textit{Für alle } i < n : p|_i \models P) \;\Leftrightarrow\; p|_n \models P \quad ,$$

so ist P eine Formel der R-Klasse.

Satz 17 *Ist P eine Reactivity-Formel, so ist P eine Formel der* R*-Klasse.*

Beweis: Folgt direkt aus der Semantik der Modaloperatoren.

Satz 18 *Die Fairnessformeln $\Box\Diamond F \rightarrow \Diamond G$ sind* R*-Formeln und sie sind als Reactivity-Formeln darstellbar.*

Beweis: Sei p ein Pfad. Es gilt

$$p \models \Box(\Box\Diamond F \rightarrow \Diamond G) \;\Leftrightarrow\; p \models \Box(\Diamond\Box\neg F \vee \Diamond G) \;\Leftrightarrow\; p \models \Box(\Diamond\Box\neg F \vee \Box\Diamond G) \quad .$$

Um diese Formeln formal in die verwendete Logik einzubetten wird R als neues, syntaktisch den Pfadquantoren und -selektoren gleichgestelltes Zeichen eingeführt:

(TZ4) Ist P eine $\mathcal{TK}$-Pfadformel, so ist RP eine $\mathcal{TK}$-Zustandsformel.

Definition 39 *Sei $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ eine Kripke-Struktur, $g \in \boldsymbol{G}$, χ eine Variablenbelegung, P eine* R*-Formel.*
Dann ist

$$(g, \chi) \models \mathsf{R}P \;:\Leftrightarrow\; (g, \chi) \models \mathsf{A}\Box P \quad .$$

Nach obigen Erkenntnissen genügt es, die Gültigkeit einer R-Formel stellvertretend für alle auf einem Pfad liegenden durch Präfixe beschriebenen Zustände in dem letzten solchen zu überprüfen:

Satz 19 *Sei $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ eine Kripke-Struktur, $\Omega = (\Phi, \Pi, \Psi)$ eine P&P-Interpretation, T ein Tableauzweig, χ eine Variablenbelegung, $\gamma \in \Gamma$, $\lambda \in \Lambda$ und P eine* R*-Formel. Dann gilt*

$$\begin{aligned}
(\boldsymbol{K}, \Omega, \chi) \not\models \gamma : \mathsf{R}P \quad &\Leftrightarrow \quad (\boldsymbol{K}, \Omega, \chi) \models \gamma : \mathsf{A}\Box P \\
&\Leftrightarrow \quad (\boldsymbol{K}, \Omega, \chi) \models \gamma : \mathsf{A}\Box\mathsf{A}P \\
&\Leftrightarrow \quad \textit{Für alle Pfade } p = (g_0, \ldots, g_i = \Psi(\gamma, \chi), \ldots) \textit{ gilt für alle } j \geq i\textit{: } (p|_j, \chi) \models \\
&\qquad P \textit{ und } (g_j, \chi) \models \mathsf{R}P. \\
&\Rightarrow \quad \textit{Für alle Pfadinformationsformeln } I = \lambda : [\gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}] \\
&\qquad \textit{mit } \gamma = \gamma_i \textit{ auf } T \textit{ gilt für alle } i > j\textit{:} \\
&\qquad (\boldsymbol{K}, \Omega, \chi) \not\models \gamma_i : \lambda P \textit{ und } (\boldsymbol{K}, \Omega, \chi) \not\models \gamma_i : \mathsf{R}P. \\
&\Leftrightarrow \quad \textit{Für alle Pfadinformationsformeln } I = \lambda : [\gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}] \\
&\qquad \textit{mit } \gamma = \gamma_i \textit{ auf } T \textit{ gilt } (\boldsymbol{K}, \Omega, \chi) \not\models \gamma_n : \lambda P \textit{ und für alle } i > j\textit{:} \\
&\qquad (\boldsymbol{K}, \Omega, \chi) \not\models \gamma_i : \mathsf{R}P.
\end{aligned}$$

Damit ergibt sich die folgende Tableauregel:

$$\frac{\begin{array}{c}\gamma_i : \mathsf{R}P \\ \lambda : [\gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}]\end{array}}{\begin{array}{c}\gamma_n : P[\lambda] \\ \text{für alle } j > i\text{:} \\ \gamma_j : \mathsf{R}P\end{array}}$$

In den meisten Fällen werden Fairnessannahmen global für eine Kripke-Struktur gemacht:

Satz 20 *Ist $\boldsymbol{K}$ eine Kripke-Struktur mit Wurzel 0 und P eine Formel der* R*-Klasse, so gilt*

$$\boldsymbol{K} \models \mathsf{R}P \;\Leftrightarrow\; 0 \models \mathsf{R}P \quad .$$

In diesem Fall ergibt sich weiter

Korollar 5 *Sei $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ eine Kripke-Struktur mit Wurzel 0, $\Omega = (\Phi, \Pi, \Psi)$ eine P&P-Interpretation, T ein Tableauzweig, χ eine Variablenbelegung, $\gamma \in \Gamma$, $\lambda \in \Lambda$ und P eine* R*-Formel. Dann gilt*

$$
\begin{array}{lll}
(\boldsymbol{K}, \Omega, \chi) \not\models \hat{0} : \mathsf{R}P & \Leftrightarrow & (\boldsymbol{K}, \Omega, \chi) \not\models \hat{0} : \mathsf{A}\Box\mathsf{A}P \\
& \Rightarrow & \textit{Für alle } \gamma \in \Gamma \textit{ gilt } (\boldsymbol{K}, \Omega, \chi) \models \gamma : \mathsf{A}P. \\
& \overset{\text{Def. 38}}{\Leftrightarrow} & \textit{Für alle Pfadinformationsformeln } I = \lambda : [\gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}] \\
& & \textit{gilt } (\boldsymbol{K}, \Omega, \chi) \not\models \gamma_n : \lambda P.
\end{array}
$$

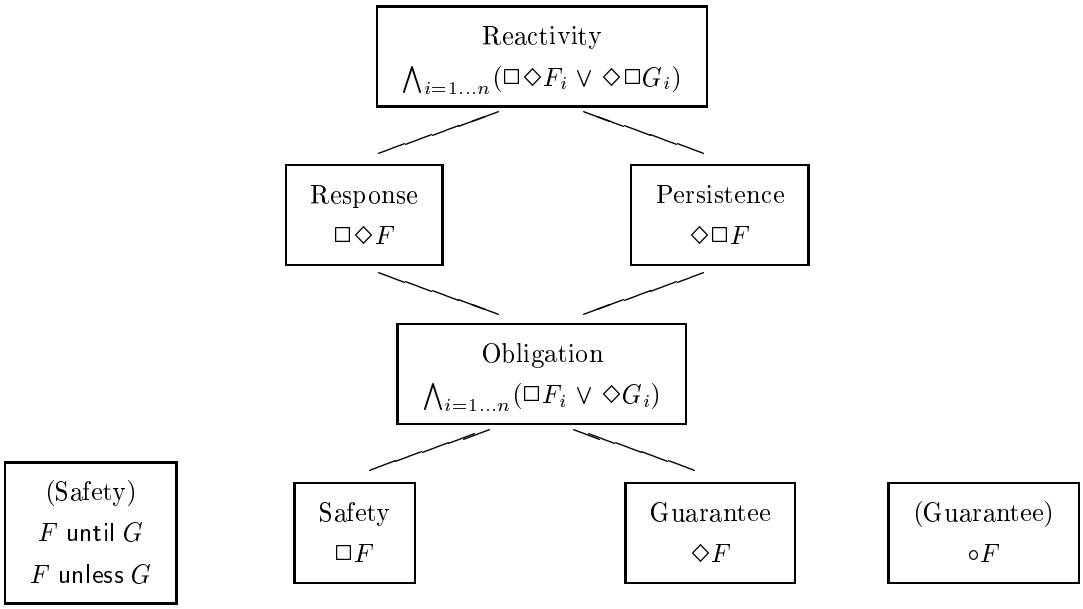
In diesem Fall erspart man sich die „Navigation“ der Formel $\mathsf{R}P$ zu allen erreichbaren Präfixen, da von $\hat{0}$ aus alle Präfixe über Pfadinformationsformeln zu erreichen sind.
Für das Tableau bedeutet dies, daß es genügt, die Gültigkeit einer R-Formel stellvertretend für alle im Tableau beschriebenen Zustände allein durch Formeln linearer Logik in den „Spitzen“ der Pfade zu testen. Dieses Vorgehen wird mit der zusätzlichen Regel

$$
\frac{\hat{0} : \mathsf{R}P \qquad \lambda : [\gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}]}{\gamma_n : P[\lambda]}
$$

in den Kalkül integriert.

8.3 Ausdruckskraft dieses Kalküls

In Anlehnung an [MP92] werden einige Klassen von temporallogischen Formeln definiert. Seien F und G $\mathcal{TK}$-Zustandsformeln.



Satz 21 *Alle universell pfadquantifizierten Formeln dieser Klassen sind mit dem Kalkül zu verarbeiten (falls sie als Axiome dienen) und nachzuweisen (falls sie gezeigt werden sollen).*

Beweis:

Safety, (Safety): Sind CTL-Formeln, Nachweis ggf. durch Induktion über die Zustandsfolge.

Guarantee, (Guarantee): Sind ebenfalls CTL-Formeln.

Obligation: Mit der Äquivalenz $\mathsf{A}(P \wedge Q) \equiv \mathsf{A}P \wedge \mathsf{A}Q$ kann oEdA. $n = 1$, also $P \equiv (\Box F \vee \Diamond G)$ angenommen werden, was eine (relativ einfache) CTL$^+$-Pfadformel ist.

Mit der Umformung von Seite 112 erhält man

$$\begin{aligned}\mathsf{A}(\Box F \vee \Diamond G) &\equiv \neg\mathsf{E}(\Box\neg G \wedge \Diamond\neg F)\\ &\equiv \neg\mathsf{E}(\Box\neg G \wedge (\mathsf{true\ until}\ \neg F))\\ &\equiv \neg\mathsf{E}((\neg G)\ \mathsf{until}\ (\neg F \wedge \mathsf{E}\Box\neg G))\\ &\equiv \mathsf{A}\neg((\neg G)\ \mathsf{until}\ (\neg F \wedge \mathsf{E}\Box\neg G)) \quad ,\end{aligned}$$

eine CTL-Formel.

Diese rein syntaktische Umformung liefert jedoch nicht das beste Ergebnis: Es ist

$$\mathsf{A}(\Box F \vee \Diamond G) \equiv \mathsf{A}(F\ \mathsf{unless}\ \mathsf{A}\Diamond G)$$

eine einfachere (und klarere) CTL-Formel.

Beide Ergebnisse lassen sich mit dem Kalkül verarbeiten.

Response, Persistence, Reactivity:

Dieses sind R-Formeln. Als Axiome sind sie mit der R-Regel einzubringen, ein Nachweis erfolgt unter Verwendung anderer R-Formeln (Axiome oder Lemmata). □

Anmerkung:

In [MP92] werden zusätzlich rückblickende Modaloperatoren („past-operators“) verwendet. Dort wird dieselbe Klassifikation ohne die geklammerten Klassen als vollständige Klassifikation aller temporallogischen Formeln angegeben, wobei F und G „past-formulas“ sind. Eine mögliche Übertragung von rückblickenden Modaloperatoren für die vorgestellte Tableausemantik wäre relativ einfach, da für jedes Präfix die Vergangenheit eindeutig durch die Pfadinformationsformeln bestimmt ist. Die damit auf den ersten Blick erhoffte Vollständigkeit in der Ausdruckskraft wird jedoch nicht erreicht, da [MP92] bei der Umformung temporallogischer Formeln in die o.g. Klassen eine Äquivalenz

$$P\ \mathsf{until}\ Q \equiv \Diamond(Q \wedge \boxminus^+ P) \quad ,$$

wobei $\boxminus$ der rückblickende $\Box$-Operator und $\boxminus^+$ dessen strikte Version ist, verwendet ([MP92], S.296). Diese ist jedoch nur in dem ersten Zustand eines Pfades gültig.

(Bsp.: $\boldsymbol{K} = (\mathbf{N}, \{(n, n+1) : n \in \mathbf{N}\}, (M(n))(p) = \mathit{true} \Leftrightarrow 10 < n < 20)$, so ist $15 \models p\ \mathsf{until}\ \neg p$, aber nicht $15 \models \Diamond((\neg p) \wedge \boxminus^+ p)$.)

9 Allgemeingültige Formeln und zustandsunabhängig interpretierte Ausdrücke

In diesem Abschnitt wird die Sonderrolle in einer Struktur allgemeingültiger Formeln untersucht. Insbesondere sind Formeln, die in einem Zustand gültig sein sollen, und nur zustandsunabhängig interpretierte Ausdrücke enthalten, allgemeingültig. Im allgemeinen werden solche Ergebnisse benötigt, um Inkonsistenzen in einer Formelmenge festzustellen.

9.1 Allgemeingültige Formeln

Soll eine CTL-Formel F, (z.B. ein Axiom) in einer Struktur allgemeingültig sein, muß es in dem Tableaukalkül die Möglichkeit geben, für beliebige in der dem Tableau vorhandene Präfixe γ auf diesem Zweig die Formel $\gamma : F$ zu erhalten.

Eine Möglichkeit ist, für diese Formeln denselben Weg vorzusehen, wie für beliebige andere Formeln auch, und sie durch $0 : \mathsf{A}\Box F$ in das Tableau einzubringen. Die Folge ist, daß jede solche Formel durch die gesamte Struktur fortgepflanzt werden muß, um so ihre Gültigkeit in einem bestimmten Zustand per Tableaukonstruktion herzuleiten.

Um Allgemeingültigkeit als tableau-syntaktische Aussage einzubringen, wird die Menge der CTL-Knotenformeln erweitert:

(KA) Ist F eine $\mathcal{TK}$-Zustandsformel, so ist $\boldsymbol{K} : F$ eine $\mathcal{TK}$-Knotenformel.

Semantik: Sei $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$ eine Kripke-Struktur, $\Omega = (\Phi, \Pi, \Psi)$ eine P&P-Interpretation, F eine CTL-Zustandsformel ohne freie Variablen.

$$(\boldsymbol{K}, \Omega, \chi) \models\!\!\models \boldsymbol{K} : F \quad :\Leftrightarrow \quad \boldsymbol{K} \models F \quad \Leftrightarrow \quad \text{Für jeden Zustand } g \in \boldsymbol{G} \text{ gilt } g \models F$$

Daraus folgt für jede Menge Γ von Präfixen und jede P&P-Interpretation $\Omega = (\Phi, \Pi, \Psi)$

$$\begin{aligned}(\boldsymbol{K}, \Omega, \chi) \models\!\!\models \boldsymbol{K} : F \;\Rightarrow \quad & \text{Für alle } \gamma \in \Gamma \text{ gilt } (\Psi(\gamma, \chi), \chi) \models F \\ \text{bzw.} \quad & \text{für alle } \gamma \in \Gamma \text{ gilt } (\boldsymbol{K}, \Omega, \chi) \models\!\!\models \gamma : F \quad .\end{aligned}$$

Dies wird mit der Tableauregel

$$\frac{\begin{array}{c}\boldsymbol{K} : F \\ \lambda : [\ldots, \gamma, \ldots]\end{array}}{\gamma : F}$$

in den Kalkül integriert.

9.2 Pfadabschlüsse durch zustandsunabhängig interpretierte Prädikate

Bisher enthält der Kalkül nur die prädikatenlogische Abschlußregel, die Widersprüche innerhalb der Formelmengen F_γ findet. In den meisten Fällen ist jedoch die Widersprüchlichkeit nicht lokal in einem Zustand begründet, sondern läßt sich erst durch Akkumulation der Informationen über verschiedene Zustände nachweisen. Die über zustandsunabhängig interpretierte Prädikate in verschiedenen Zuständen gewonnenen Informationen müssen somit erst in einem Zustand gesammelt werden. Dies ermöglichen Axiome der Form

$$\forall X_1, \ldots X_n : \mathsf{A}\Box(p(X_1, \ldots, X_n) \leftrightarrow \mathsf{A}\Box p(X_1, \ldots, X_n))$$

für ein zustandsunabhängig interpretiertes Prädikat p mit $\mathsf{ord}(p) = n$.

Deren Verwendung vergrößert jedoch den (ohnehin nicht gerade kleinen) Suchraum beträchtlich und erfordert somit einen gezielten Einsatz.

Um diese Navigation von Ergebnissen zwischen den einzelnen Präfixen bei einem – willkürlich gewählten – Präfix zu ersparen, bietet es sich an, solche Ergebnisse als Umkehrung des vorhergehenden Abschnittes zu globalisieren und die entsprechenden Schlüsse aus diesen globalen Formeln zu ziehen. Dies wird mit den folgenden Regeln ermöglicht:
Globalisierung:

$$\frac{\gamma : F}{\boldsymbol{K} : F} \quad \text{falls } F = p(t_1, \ldots, t_n),\ p \in \Sigma^c(\boldsymbol{K}),\ t_i \in \mathsf{Term}\Sigma^c(\boldsymbol{K})$$

Globale Abschlußregel:
Sei σ eine Substitution mit $\mathsf{Bild}(\sigma) \subset \mathsf{Term}_{\Sigma^c(\boldsymbol{K})}$ (d.h. es werden nur zustandsunabhängig interpretierte Terme eingesetzt, vgl. Substitutionslemma, Seite 74):

$$\frac{\begin{array}{c}\boldsymbol{K} : \sigma(A) \\ \boldsymbol{K} : \neg\sigma(A)\end{array}}{\begin{array}{c}\bot \\ \sigma \text{ auf das gesamte Tableau anwenden.}\end{array}}$$

Bei Verwendung von Gleichheit muß diese Abschlußregel zusätzlich mit der entsprechenden Gleichheitstheorie kombiniert werden.

9.3 Optimierungen zur Behandlung von Evolving Algebras

Die in diesem Abschnitt beschriebenen Modifikationen werden jeweils an aus [May95b] entnommenen Beispielen gezeigt.

Da es außer Gleichheit bei Evolving Algebras keine weiteren Prädikate gibt, werden Pfade immer durch widersprüchliche Gleichungen und Ungleichungen abgeschlossen. Dabei werden i.a. (vgl. Abschnitt 9.2) Informationen aus verschiedenen Zuständen benötigt. Dazu werden alle im Tableau enthaltenen atomaren Formeln, die Gleichheitsatome enthalten, in globale Gleichungen transformiert, wobei die eventuell vorhandenen zustandsabhängig interpretierten Bezeichner durch neu eingeführte zustandsunabhängig interpretierte Bezeichner (durch Indizierung mit dem Präfix) ersetzt werden und dann das Gleichheitsatom analog der Globalisierungsregel als allgemeingültig angesehen wird:

$$\frac{\gamma : \mathsf{bezeichner} = wert}{\mathsf{bezeichner}_\gamma = wert}$$

Die endgültige Behandlung der Gleichheit wird aus [BH92] übernommen.

Abschlüsse erfolgen dann entsprechend durch

$$\frac{\begin{array}{c}\sigma(wert_1 = wert_2)\\ \neg\sigma(wert_1 = wert_2)\end{array}}{\begin{array}{c}\bot\\ \sigma \text{ auf das gesamte Tableau anwenden,}\end{array}}$$

wobei σ eine Substitution mit $\mathsf{Bild}(\sigma) \subset \mathsf{Term}_{\Sigma^c(\boldsymbol{K})}$ (d.h. es werden nur zustandsunabhängig interpretierte Terme eingesetzt, vgl. Substitutionslemma, Seite 74) ist.

Zu jeder Zeit enthält das Tableau nur endlich viele nichtleere Äquivalenzklassen von Termen. So können beim Auftreten eines neuen atomaren Knotens sofort die Äquivalenzklassen erweitert und auf Widersprüchlichkeit geprüft werden.

Die Syntax und Semantik der PL1 ist zur Beschreibung von Evolving Algebras nicht optimal. Dies wird an zwei Stellen deutlich:

1. In den Existenzaxiomen: $\mathsf{A}\Box(\exists x : \mathit{bezeichner} = x)$
   In Wahrheit existiert nicht nur ein solches x, sondern genau eines, und man weiß sogar, welches.

2. In den das Verhalten eines zustandsabhängig interpretierten Bezeichners bezeichner beschreibenden Formeln der Formen
   $$\forall x : \mathsf{bezeichner} = x \rightarrow \mathsf{bezeichner}' = f(x) \quad \text{und}$$
   $$\forall x : \mathsf{bezeichner} = x \rightarrow (\mathsf{bezeichner} = x \ \mathsf{unless}\ \mathit{Bedingung}) \quad .$$
   Die Variable x muß aus prädikatenlogischer Sicht allquantifiziert werden, obwohl die Aussage nur für ein einziges x nichttrivial ist.

In der Behandlung durch einen Tableaukalkül handelt man sich dafür eine freie Variable X_1 und einen Zweig $\neg(\mathsf{bezeichner} = X_1)$ ein. Dieser kann sofort abgeschlossen werden, indem man für X_1 den – sicherlich existierenden (evtl. undef) – Wert von $\mathsf{bezeichner}$ einsetzt. Diesen wiederum muß man mit Hilfe des entsprechenden Existenzaxioms einführen, dessen Auflösung einen Skolembezeichner x_2 und die Gleichung $\gamma : \mathsf{bezeichner} = x_2$ liefert. Letztere wird im Zuge der Globalisierung der Gleichungen schlußendlich zu $\mathsf{bezeichner}_\gamma = x_2$ umgeformt:

Beispiel (mit Invarianzformel):

$$\vdots$$

Existenzaxiom:

$$\gamma : \ \exists B : \mathsf{SBit} = B \quad [1]$$
$$\gamma : \ \mathsf{SBit} = b_1 \quad [2]$$
$$\mathsf{SBit}_\gamma = b_1 \quad [3]$$
$$|$$
$$\gamma : \ \forall B : \mathsf{SBit} = B \rightarrow \mathsf{A}(\mathsf{SBit} = B \ \mathsf{unless}^+ \ \mathsf{ProcAck} = \mathsf{true}) \quad [4]$$
$$\gamma : \ \mathsf{SBit} = B_2 \rightarrow \mathsf{A}(\mathsf{SBit} = B_2 \ \mathsf{unless}^+ \ \mathsf{ProcAck} = \mathsf{true}) \quad [5]$$

$$\gamma : \ \neg(\mathsf{SBit} = B_2) \quad [6] \qquad \gamma : \ \mathsf{A}(\mathsf{SBit} = b_1 \ \mathsf{unless}^+ \ \mathsf{ProcAck} = \mathsf{true}) \quad [7]$$
$$\neg(\mathsf{SBit}_\gamma = B_2) \quad [8] \qquad \vdots$$
$$\bot \ \{B_2 \leftarrow b_1\}(3, 8)$$

Insbesondere muß man das Existenzaxiom anwenden, bevor sich der Zweig teilt, um die Gleichung [3] zur weiteren Verfügung zu haben.

Dieses prädikatenlogische Vorgehen kann die Eigenschaft der Evolving Algebra, daß in jedem Zustand die meisten benötigten Äquivalenzklassen von Termen mit den Bezeichnern der Evolving Algebra auf natürliche Weise gegebene Repräsentanten besitzen, nicht ausnutzen. (Gelegentlich kommen durch die Beschreibung der verwendeten Term- bzw. Datenstrukturen weitere Äquivalenzklassen hinzu.) In den globalen Formen der Gleichungen treten ohnehin schon mit den entsprechenden Zuständen indizierte Funktionen auf. Von daher liegt es nahe, anstelle der prädikatenlogischen Einführung von Skolemkonstanten an dieser Stelle gleich den Repräsentanten der Äquivalenzklassen zu verwenden:

Man ersetzt die Existenzaxiome durch die Möglichkeit, an jeder beliebigen Stelle des Tableaux den Gleichungsknoten $\gamma : \ \mathsf{bezeichner} = \mathsf{bezeichner}_\gamma$ einzufügen. Damit lassen sich nicht nur die oben erwähnten Äste der Form $\neg(\mathsf{bezeichner} = X_1)$ schließen, man substituiert zusätzlich gleich den Repräsentanten $\mathsf{bezeichner}_\gamma$ an die entsprechenden Stellen in der neu hinzugekommenen Formel. Die eingefügte Gleichung wird bei der Globalisierung zu $\mathsf{bezeichner}_\gamma = \mathsf{bezeichner}_\gamma$ und kann sofort wegfallen:

in obigem Beispiel:

$$\vdots$$

Existenzaxiom:

$$\gamma: \ \exists B : \mathsf{SBit} = B \quad [1]$$
$$\gamma: \ \mathsf{SBit} = \mathsf{SBit}_\gamma \quad [2]$$
$$\mathsf{SBit}_\gamma = \mathsf{SBit}_\gamma \quad [(3)]$$
$$|$$
$$\gamma: \ \forall B : \mathsf{SBit} = B \to \mathsf{A}(\mathsf{SBit} = B \ \mathsf{unless}^+ \ \mathsf{ProcAck} = \mathsf{true}) \quad [4]$$
$$\gamma: \ \mathsf{SBit} = B_2 \to \mathsf{A}(\mathsf{SBit} = B_2 \ \mathsf{unless}^+ \ \mathsf{ProcAck} = \mathsf{true}) \quad [5]$$

$$\gamma: \ \neg(\mathsf{SBit} = B_2) \quad [6] \qquad \gamma: \ \mathsf{A}(\mathsf{SBit} = \mathsf{SBit}_\gamma \ \mathsf{unless}^+ \ \mathsf{ProcAck} = \mathsf{true}) \quad [7]$$
$$\neg(\mathsf{SBit}_\gamma = B_2) \quad [8] \qquad \vdots$$
$$\bot \ \{B_2 \leftarrow \mathsf{SBit}_\gamma\}(3, 8)$$

Die natürliche Fortsetzung dieser Argumentation bei Tableaux ist das Hinzufügen einer Abschlußregel

$$\frac{\gamma : \neg(\mathsf{bezeichner} = X_1)}{\bot \ \{X_1 \leftarrow \mathsf{bezeichner}_\gamma\}}$$

in obigem Beispiel:

$$\vdots$$
$$\gamma: \ \forall B : \mathsf{SBit} = B \to \mathsf{A}(\mathsf{SBit} = B \ \mathsf{unless}^+ \ \mathsf{ProcAck} = \mathsf{true}) \quad [1]$$
$$\gamma: \ \mathsf{SBit} = B_2 \to \mathsf{A}(\mathsf{SBit} = B_2 \ \mathsf{unless}^+ \ \mathsf{ProcAck} = \mathsf{true}) \quad [2]$$

$$\gamma: \ \neg(\mathsf{SBit} = B_2) \quad [3] \qquad \gamma: \ \mathsf{A}(\mathsf{SBit} = \mathsf{SBit}_\gamma \ \mathsf{unless}^+ \ \mathsf{ProcAck} = \mathsf{true}) \quad [4]$$
$$\bot \ \{B_2 \leftarrow \mathsf{SBit}_\gamma\} \qquad \vdots$$

Ein Schritt zu einer eigenständigen, speziell auf Evolving Algebras abgestimmten Logik wäre die Einführung eines neuen Quantors ∇ – ähnlich dem in der Mathematik gebräuchlichen $\exists!$ – , der die Einmaligkeit des Wertes und den gegebenen Repräsentanten ausnutzt. Wie alle Quantoren bindet er eine in der Matrix frei auftretende Variable. Diese muß durch ihren Namen eindeutig einem Bezeichner der Evolving Algebra zuzuordnen sein (etwa $SBit$ für SBit). Semantisch ist eine Kripke-Struktur Modell einer solchen Formel, wenn sie Modell der Formel, wobei die entsprechenden Werte der Bezeichner in dem betrachteten Zustand für die solchermaßen gebundenen Variablen eingesetzt werden, ist:

$$(g, \chi) \models \nabla bezeichner : F(bezeichner) \ :\Leftrightarrow \ (g, \chi) \models \forall X : (\mathsf{bezeichner} = X \to F(X))$$

In einem Kalkül (nicht nur Tableau) wird er durch Einführung einer neuen Funktion und einer Termersetzung aufgelöst:

$$\frac{\gamma : \ \nabla bezeichner : F}{\gamma : \ F[\mathsf{bezeichner}_\gamma / bezeichner]}$$

Damit können unter anderem die Invarianzaxiome kürzer formuliert werden:

$$\nabla bezeichner : \mathsf{A}\circ(\mathsf{A}(\mathsf{bezeichner} = bezeichner \ \mathsf{unless} \ (\bigwedge\{\mathsf{name} : \mathsf{bezeichner} \in Zuweisung(\mathsf{name})\}))) \quad ,$$

womit das obige Beispielfragment auf zwei Zeilen verkürzt wird:

$$\begin{array}{c} \vdots \\ \gamma : \ \nabla SBit : \mathsf{A}(\mathsf{SBit} = SBit \ \mathsf{unless}^+ \ \mathsf{ProcAck} = \mathsf{true}) \quad [1] \\ \gamma : \ \mathsf{A}(\mathsf{SBit} = \mathsf{SBit}_\gamma \ \mathsf{unless}^+ \ \mathsf{ProcAck} = \mathsf{true}) \quad [2] \\ \vdots \end{array}$$

9.4 Optimierungen der Beweisführung

Von den hier angegebenen, vom prädikatenlogischen Tableaubeweisen [AB94] übernommenen Optimierungsmöglichkeiten wurde in der Fallstudie [May95b] Gebrauch gemacht. Mit ihnen lassen sich die Beweise erheblich verkürzen. Dies ist insbesondere im Hinblick auf eine interaktive Beweisführung von Vorteil.

1.: Beim Abschluß eines Tableauzweiges durch eine Substitution freier Variablen wird deren Anwendung auf nicht-universelle Variablen eingeschränkt:

Definition 40 *Sei $\gamma : F$ ein Knoten auf einem Tableauzweig T, X eine in F frei auftretende Variable. $\gamma : F$ heißt* universell bezüglich X, *falls für alle Grundsubstitutionen σ gilt:*

$$(\boldsymbol{K}, \Omega) \models \sigma(T) \ \Leftrightarrow \ (\boldsymbol{K}, \Omega) \models \sigma(\forall x : F) \quad .$$

Dies erspart in vielen Fällen die mehrfache Anwendung von γ-Regeln.

2.: Implikationen werden – falls der linke Zweig nicht trivial abgeschlossen wird – mit Lemmagenerierung $A \to B \equiv \neg A \vee (A \wedge B)$ aufgelöst.

10 Erweiterung auf CTL*

Nachdem in den vergangenen Abschnitten die Ausdruckskraft des ursprünglich für CTL *konstruierten Kalküls ausgereizt wurde, wird noch eine Variante für* CTL* *vorgestellt. Diese ist insbesondere notwendig, um die prädikatenlogischen Quantoren voll auszunutzen. Dabei muß die Behandlung verzweigender Pfade allerdings grundlegend anders gestaltet werden.*

Zusätzlich zu CTL$^+$ sind in CTL* geschachtelte Modaloperatoren (Syntaxregel P4*, S. 21) zugelassen. In der Erweiterung auf Prädikatenlogik muß man auch eine weitere Möglichkeit prädikatenlogischer Quantifizierung berücksichtigen ((PQ*): z.B. $\mathsf{A}\exists x : P$).

Die tableauinterne Syntax wird um (TPZ), (TP4) und (TPQ) erweitert:

(TA) Jedes Atom ist eine $\mathcal{TK}$-Zustandsformel.

(TPZ) Jede $\mathcal{TK}$-Zustandsformel ist eine $\mathcal{TK}$-Pfadformel.

(TZ1) Sind F und G $\mathcal{TK}$-Zustandsformeln, so sind auch $\neg F$, $F \wedge G$, $F \vee G$ und $F \rightarrow G$ $\mathcal{TK}$-Zustandsformeln.

(TZQ) Ist F eine $\mathcal{TK}$-Zustandsformel und x eine Variable, so sind auch $\forall x : F$ und $\exists x : F$ $\mathcal{TK}$-Zustandsformeln.

(TP1) Sind F und G $\mathcal{TK}$-Zustandsformeln, so sind $\circ F$, $\Box F$, $\Diamond F$ und $(F \,\mathsf{until}\, G)$ $\mathcal{TK}$-Pfadformeln.

(TP2) Ist P eine $\mathcal{TK}$-Pfadformel, so ist $\neg P$ eine $\mathcal{TK}$-Pfadformel.

(TP3) Sind P und Q $\mathcal{TK}$-Pfadformeln, so sind $P \wedge Q$ und $P \vee Q$ $\mathcal{TK}$-Pfadformeln.

(TP4) Sind P und Q $\mathcal{TK}$-Pfadformeln, so sind $\circ P$, $\Box P$, $\Diamond P$ und $(P \,\mathsf{until}\, Q)$ $\mathcal{TK}$-Pfadformeln.

(TPQ) Ist P eine $\mathcal{TK}$-Pfadformel und x eine Variable, so sind $\forall x : P$ und $\exists x : P$ $\mathcal{TK}$-Pfadformeln.

(TZ2) Ist P eine $\mathcal{TK}$-Pfadformel, so sind $\mathsf{A}P$ und $\mathsf{E}P$ $\mathcal{TK}$-Zustandsformeln.

(TC1) Jede $\mathcal{TK}$-Zustandsformel ist eine $\mathcal{TK}$-Prä-Knotenformel

(TZ3) Ist P eine $\mathcal{TK}$-Pfadformel und λ ein Pfadbezeichner, so ist λP eine $\mathcal{TK}$-Zustandsformel.

(TC3) Ist P eine $\mathcal{TK}$-Pfadformel und λ ein Pfadbezeichner, so sind $(\mathsf{A})P$ und $(\lambda)P$ $\mathcal{TK}$-Prä-Knotenformeln.

(TK1) Alle Pfadinformationsformeln sind $\mathcal{TK}$-Knotenformeln.

(TK2) Ist F eine $\mathcal{TK}$-Prä-Knotenformel und γ ein Zustandsbezeichner, so ist $\gamma : F$ eine $\mathcal{TK}$-Knotenformel.

Einige so entstehende Formelklassen stellen Problemfälle dar, die mit dem für CTL/CTL$^+$ entwickelten Kalkül nicht verarbeitet werden können.

Die Sätze über das Fortpflanzungsverhalten von CTL-Formeln sind in CTL* nicht gültig (die dort verwendeten Zustandsformeln P_0, P_2 und P_2 gibt es i.a. nicht). Die für CTL gültige Feststellung, daß man

verzweigende Pfade erst ab der Verzweigung getrennt betrachten muß, hat besitzt hier keine Gültigkeit mehr.

Daher muß in der CTL*-Variante des Kalküls die Behandlung der Pfade detaillierter gestaltet werden. Durch die ausschließliche Verwendung vollständiger Pfade können die problematischen Formeln auf Logik linearer Zeit zurückgeführt werden. Ansonsten kann die Tableausyntax und -semantik komplett übernommen werden. Aus dieser Sicht ist weniger die CTL*-Variante eine Erweiterung der CTL-Variante, als vielmehr letztere eine Vereinfachung und Optimierung der erstgenannten.

Anstelle der der bisher verwendeten Regel zur Auflösung von E-Quantoren wird die folgende Regel verwendet:

Sei T der Tableauzweig, auf den die Regel angewendet werden soll.

$$\frac{\begin{array}{c}\gamma : \mathsf{E}P \\ \lambda : [0, I', \gamma, \ldots, \hat{\infty}]\end{array}}{\begin{array}{c}\hat{\kappa}(\mathsf{free}(T)) : [0, J', \gamma, \emptyset, \hat{\infty}] \\ \gamma : \hat{\kappa}(\mathsf{free}(T))\ P\end{array}}$$

wobei $\hat{\kappa}$ ein noch nicht vorkommendes Pfadsymbol ist.

Dieses bringt speziell in dem Bereich, wo solche Pfade parallel verlaufen, einiges an organisatorischem Aufwand mit sich. Daher wird die Frage, wie man J' aus I' erhält, noch etwas aufgeschoben.

Analog der für CTL$^+$ durchgeführten Umformungen kann man oEdA. annehmen, daß direkt auf den führenden Pfadquantor ein Modaloperator folgt.

Nach den Überlegungen von Abschnitt 8.1 bereiten die zusätzlichen Formeln, falls sie existentiell pfadquantifiziert sind, keine Probleme, da sie mit der eben angegebenen Regel an einen *bestimmten* Pfadbezeichner gebunden werden. Diese werden durch die Umformung $\lambda P \rightsquigarrow P[\lambda]$ in eine im Kalkül zu verarbeitende Syntax gebracht. Eine solchermaßen umgeformte Formel kann – bis man bei ihrer Verarbeitung wieder auf einen Pfadquantor/-selektor stößt – mit den bei CTL angegebenen Tableauregeln verarbeitet werden.

Für prädikatenlogisch verknüpfte, bereits an einen Pfad gebundene Pfadformeln (entstanden aus existentiell quantifizierten Pfadformeln) werden die für die Einbindung von PTL-Formeln angegebenen Regeln übernommen.

Damit ist nur noch die Behandlung universell quantifizierter Pfadformeln in CTL*-Syntax zu klären. Einige Fälle können mit den folgenden Äquivalenzen auf CTL reduziert werden:

$$\mathsf{A}\circ P \equiv \mathsf{A}\circ(\mathsf{A}P) \quad , \quad \mathsf{A}\neg\circ P \equiv \mathsf{A}\circ(\mathsf{A}\neg P)$$
$$\mathsf{A}\Box P \equiv \mathsf{A}\Box(\mathsf{A}P) \quad , \quad \mathsf{A}\neg\Diamond P \equiv \mathsf{A}\Box(\mathsf{A}\neg P) \quad .$$

Für die Auflösung der verbleibenden Formeln wird ausgenutzt, daß nur vollständige Pfade verwendet werden.

Die Formeln $\gamma : \mathsf{A}P$ der Formen

$$\gamma : \mathsf{A}\Diamond P_1, \quad \gamma : \mathsf{A}\neg\Box P_1, \quad \gamma : \mathsf{A}(P_1 \text{ until } P_2), \quad \gamma : \mathsf{A}(\neg(P_1 \text{ until } P_2)),$$
$$\gamma : \mathsf{A}(P_1 \text{ unless } P_2) \quad \text{und} \quad \gamma : \mathsf{A}(\neg(P_1 \text{ unless } P_2))$$

werden aufgelöst, indem die Pfadformel P explizit an jeden im Tableau durch einen Pfadbezeichner λ beschriebenen, das Präfix γ durchlaufenden Pfad gebunden wird:

$$\frac{\gamma : \mathsf{A}\Diamond P \qquad \lambda : [0 = \gamma_0, \ldots, \gamma, \ldots, \hat{\infty}]}{\gamma : \lambda \Diamond P} \qquad \frac{\gamma : \mathsf{A}(P \text{ until } Q) \qquad \lambda : [0 = \gamma_0, \ldots, \gamma, \ldots, \hat{\infty}]}{\gamma : \lambda\ (P \text{ until } Q)} \qquad \frac{\gamma : \mathsf{A}(\neg(P \text{ until } Q)) \qquad \lambda : [0 = \gamma_0, \ldots, \gamma, \ldots, \hat{\infty}]}{\gamma : \lambda\ (\neg(P \text{ until } Q))}$$

Damit wird die endgültige Auflösung solcher Formeln auf die Anwendung von PTL-Formeln auf einzelne Pfade reduziert.

Behandlung der prädikatenlogischen Quantoren:

Es gelten die folgenden Äquivalenzen:

$$\mathsf{E}\exists x : P \equiv \exists x : \mathsf{E}P \quad \text{und} \quad \mathsf{A}\forall x : P \equiv \forall x : \mathsf{A}P \quad .$$

Für die beiden verbleibenden Möglichkeiten solchermaßen geschachtelter Pfad- und PL1-Quantoren ergeben sich die folgenden Regeln:

$$\frac{\gamma : \mathsf{A}\exists x : P \qquad \lambda : [0 = \gamma_0, \ldots, \gamma, \ldots, \hat{\infty}]}{\gamma : \exists x : \lambda P} \qquad \frac{\gamma : \mathsf{E}\forall x : P \qquad \lambda : [0 = \gamma_0, I', \gamma, \ldots, \hat{\infty}]}{\hat{\kappa}(\mathsf{free}(T)) : [0, J', \gamma, \emptyset, \hat{\infty}] \quad \gamma : \forall x : \hat{\kappa}(\mathsf{free}(T))P}$$

wobei $\hat{\kappa}$ ein noch nicht vorkommendes Pfadsymbol ist.

Bezüglich I' und J' muß darauf geachtet werden, die durch parallel verlaufende Pfade entstehende Redundanz so effizient wie möglich zu behandeln sowie die Pfadinformationsformeln konsistent zu halten:

(1) Wird bei einem Präfix γ, das durch eine Pfadinformationsformel $\lambda : [0 = \gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}]$, $\gamma = \gamma_i$, bekannt ist, ein neuer, dort abzweigender Pfadbezeichner κ eingeführt, geschieht dies mit der Pfadinformationsformel $\kappa : [0 = \gamma_0, \emptyset, \gamma_1, \emptyset, \ldots, \gamma_i, \emptyset, \hat{\infty}]$.

(2) Wird unter Verwendung einer Pfadinformationsformel $\lambda : [0 = \gamma_0, L_0, \gamma_1, L_1, \ldots, \gamma_n, L_n, \hat{\infty}]$ zwischen γ_i und γ_{i+1} ein neues Präfix δ benannt, wird für *jede* Pfadinformationsformel $\kappa : [0 = \gamma_0, \ldots \gamma_i, L, \gamma_{i+1}, \ldots, \hat{\infty}]$ eine neue Pfadinformationsformel $\kappa : [0 = \gamma_0, \ldots \gamma_i, L', \delta, L, \gamma_{i+1}, \ldots, \hat{\infty}]$, wobei $L' = L$ falls δ nicht direkter Nachfolgezustand von γ_i sein soll, $L' = \circ$ sonst ist, sowie die Knotenformeln für $\delta : L$ in den Tableauzweig eingefügt.

Damit ist garantiert, daß zu jeder Zeit für jeden Pfadbezeichner die neueste Pfadinformationsformel die Folge aller aktuell bekannten Präfixe auf ihm beschreibt. Die auf den einzelnen Pfadabschnitten geltenden

Formeln sind dabei nicht in jeder Pfadinformationsformel komplett aufgezählt, aber in der Gesamtheit aller diesen Abschnitt beinhaltenden Pfade enthalten. Da beim Einfügen eines neuen Zustandes alle Pfadinformationsformeln erneuert werden, können die Listen dabei alle aufgesammelt und für den neuen Zustand ausgewertet werden.

Für die Bearbeitung der einzelnen Formeln wird unterschieden, ob sie CTL oder CTL* sind: CTL-Formeln können wie im CTL-Kalkül aufgelöst werden. Insbesondere genügt es, sie auf jedem Abschnitt einer Pfadinformationsformel stellvertretend für alle dort parallel laufenden Pfadinformationsformeln aufzulösen. CTL*-Formeln müssen für jede Pfadinformationsformel einzeln aufgelöst werden.

Für die bereits aus CTL$^+$ bekannten Problemfälle (universell quantifizierte Disjunktionen von CTL$^+$-Pfadformeln) hat man nun zwei Alternativen:

(1) Wie in Abschnitt 7 beschrieben, wird die Formel in CTL umformuliert und dann als CTL-Formel aufgelöst.

(2) Die Formel $\mathsf{A}P$ wird für jede Pfadinformationsformel $\lambda : [\ldots]$ einzeln als λP aufgelöst.

11 Einschränkung auf Logik linearer Zeit

Dieser Abschnitt beschäftigt sich kritisch-konstruktiv mit der Frage der Notwendigkeit des Einsatzes von Logik verzweigender Zeit sowie der Übertragung des Kalküls $\mathcal{TK}$ auf lineare Zeit.

Die klassische Streitfrage, ob lineare oder verzweigende Zeit vorzuziehen ist ([La80, EH83]) resultierte in der Einigung auf CTL*, das die Ausdruckskraft beider Logiken kombiniert.

Aus der kombinierten Ausdruckskraft resultiert ebenfalls eine erhöhte Komplexität, die sich in dieser Arbeit nicht zuletzt in Kapitel 10 in der Notwendigkeit der vollständigen Repräsentation der Pfade äußert. Daher ist kritisch zu untersuchen, ob zur *Verifikation* von Prozessen dieser Aufwand überhaupt notwendig ist.

Bezüglich dieser Frage kommt [EH83] zu folgendem Schluß:
„Für Verifikationszwecke ist man hauptsächlich an Eigenschaften, die für *alle* Pfade gelten, interessiert. Dafür ist es ausreichend, *einen* beliebigen Pfad aufzugreifen und diesen zu betrachten. Trotzdem gibt es Anwendungen, in denen man die Fähigkeit benötigt, die Existenz *verschiedener* Pfade nachzuweisen, die ausschließlich in Logik verzweigender Zeit gegeben ist.“

Die erste Aussage entspricht den meisten Korrektheitsaussagen. Die zweite findet etwa Anwendung, wenn die Lebendigkeit eines Systems nachgewiesen werden soll, falls eine „erwartete“ Reaktion ausbleibt. Ein weiterer Punkt, wo verzweigende Zeit notwendig ist, ist die Beschreibung nichtdeterministischer Automaten, wo es ausreicht, daß es einen akzeptierenden Ablauf gibt.

Diese Beobachtung wird – erwartungsgemäß – bei der Verifikation des Alternating-Bit-Protokolls [May95b] bestätigt: Man geht von *einem* Pfad aus, der die zu zeigende Eigenschaft nicht besitzt, und zeigt, daß es

diesen Pfad nicht geben kann. Im Beweis treten abzweigende Pfade in zwei unterschiedlichen Situationen auf:

- Aus Eingabeformeln der Form $\neg \mathsf{A}\Box\mathsf{A}P$. In diesem Fall wird bei Auflösung des inneren A-Quantors (der dann negiert ist) ein abzweigender Pfad erzeugt. Der geradeauslaufende Pfad wird nicht weiter betrachtet. Die Formel ist in CTL* äquivalent zu $\neg\mathsf{A}\Box P$, womit im CTL*-Kalkül nur ein Pfad betrachtet werden müßte.
- Aus der Formulierung der Fairnessforderung, bei der die Tatsache „Regel r ist ausführbar“ durch „es gibt einen Pfad, auf dem der Nachfolgezustand durch Ausführung von Regel r erreicht wurde“ ausgedrückt wird. Von dem dabei eingeführten Pfad wird nur der erste Zustand betrachtet. Eine entsprechende Umformulierung der Fairnessforderung (mit einer zusätzlichen zustandsabhängig interpretierten Konstante für „Regel r ist ausführbar“) ließe alle diese Abzweige wegfallen.

Ein Kalkül für PTL läßt sich aus $\mathcal{TK}$ erzeugen, indem man die Pfadquantoren streicht, nur einen Pfadbezeichner λ behält und die in Kapitel 6 angegebenen Tableauregeln für λP anwendet. Der Grundgedanke der Modellierung, daß von endlich langen Pfadabschnitten abstrahiert werden kann, bleibt damit erhalten. Aus diesem Grund wird es auch keinen signifikant einfacheren Kalkül für prädikatenlogisches PTL geben, mit dem sich Fairnessaussagen verarbeiten lassen.

12 Fazit und Ausblick

Das als Ziel angestrebte Konzept zur kombinierten formalen Modellierung und Verifikation kann mit den beschriebenen Techniken realisiert werden: Prozesse werden aus operationaler Sicht als Evolving Algebras modelliert, diese als temporale Kripke-Strukturen beschrieben, in einer Logik der PTL/CTL-Familie axiomatisiert und mit Hilfe des angegebenen Kalküls formal verifiziert.

Eine Realisierung des Verfahrens sollte die folgenden Komponenten umfassen:

- Werkzeuge zur Transformation einer Spezifikation nach den üblichen Spezifikationstechniken (algebraische Spezifikation, Z, Zustandsübergangsdiagramme) in eine Modellierung als Evolving Algebra,
- einen komfortablen Evolving-Algebra-Editor und -Interpreter,
- ein Modul zur syntaktischen Transformation der Spezifikation einer Evolving Algebra in eine CTL-Axiomenmenge,
- ein Beweissystem auf Basis des beschriebenen Tableaukalküls.

Für das Beweissystem ergeben sich die folgenden Resultate:

Ein Beweis wird, wie bereits in Abschnitt 5.7 festgestellt, nicht in einem Schritt zu führen zu sein, sondern entsprechend den Entwurfsüberlegungen die Korrektheit des Entwurfs nachvollziehen. Als Grundlage des formalen Tableaubeweises dient daher ein in mathematischem Stil argumentativ geführter Beweis.

Aufgrund der verschiedenen möglichen Erweiterungen wird man einen modularen Kalkül verwenden, der aus den Grundregeln für CTL besteht und im Einzelfall um weitere Regeln, die bestimmte benötigte Formelschemata behandeln, erweitert wird.

Auf Basis der vorliegenden Spezifikation ist daher für jeden (Teil)Beweis zu entscheiden, ob ein Kalkül für PTL, CTL, oder CTL* benötigt wird.

Nach den Ergebnissen der Abschnitte 5.6 und 8.3 genügt bereits die Ausdruckskraft des erweiterten CTL-Kalküls, um die meisten Beweise zu führen. Einige Teilbeweise können wiederum bei geeigneter Codierung durch prädikatenlogische Beweiser durchgeführt werden. Nur in den Fällen, in denen kompliziert geschachtelte Modaloperatoren oder prädikatenlogische Quantoren in der Form $\mathsf{A}\exists x : P$ oder $\mathsf{E}\forall x : P$ auftreten, wird die CTL*-Version benötigt.

Eine Realisierung des Beweisverfahrens sollte nicht einen automatischen Beweiser, sondern ein interaktives Beweissystem zum Ziel haben. Mit diesem könnte durch Angabe des nächsten aufzulösenden Knotens ein großer Teil der Ausführung automatisiert werden, während der Benutzer die Richtung des Beweises bestimmt.

Damit ist eine formale Verifikation beliebiger Prozesse möglich, wobei im Einzelfall abzuwägen ist, ob das Prädikat „formal verifiziert“ den Aufwand wert ist.

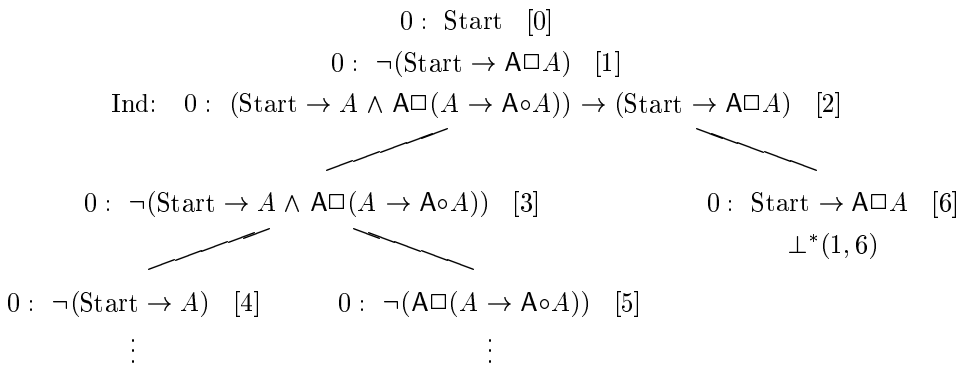
Anhang

A Beispiele

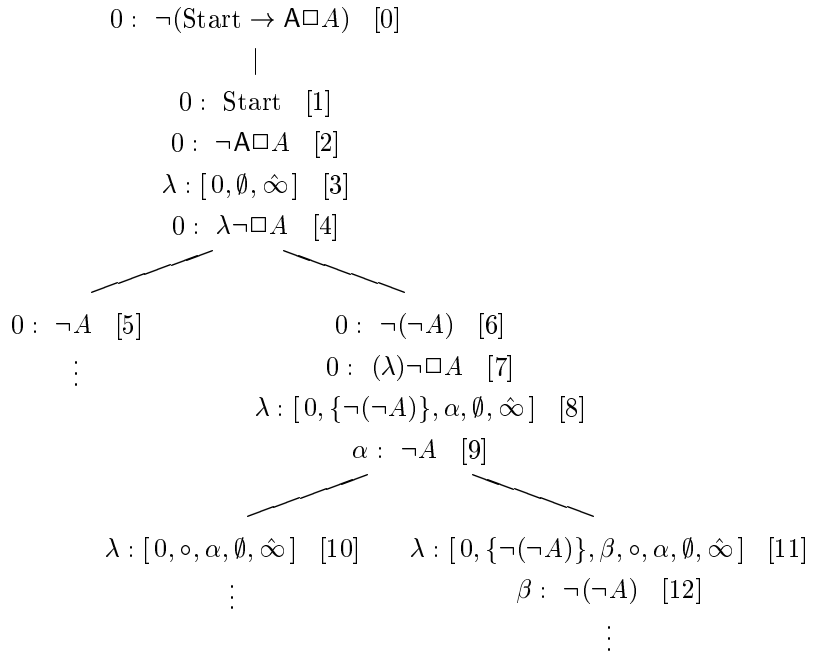
Dieser Abschnitt enthält einige aussagenlogische Beispiele, die die Vorgehensweise des Kalküls veranschaulichen.

- **Induktion:**

a) Mit dem Induktionsschema „Ind: $(\text{Start} \to A \wedge \mathsf{A}\Box(A \to \mathsf{A}\circ A)) \to (\text{Start} \to \mathsf{A}\Box A)$“:



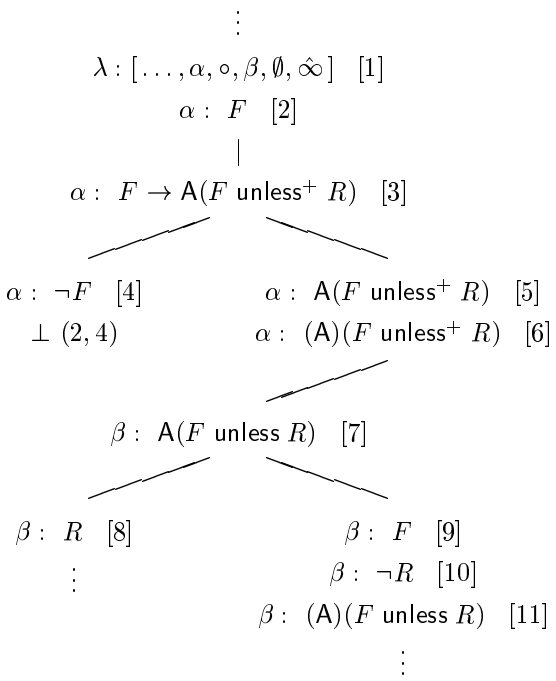
b) Als Erreichbarkeitsaussage:



Im allgemeinen wird der 2. Zweig vom 3. subsumiert, da man nur die Voraussetzungen $0 : A$ bzw. $\beta : A$ und $\alpha : \neg A$ verwendet und damit den Beweis übertragen kann. Ist A eine prädikatenlogische Formel (was

bei Invarianzen häufig der Fall ist), so kann der Induktionsanfang direkt sowie der Induktionsschritt nach geeigneter Codierung mit einem prädikatenlogischen Beweiser gezeigt werden.

- Auswerten einer Invarianzaussage in eine Aussage für den unmittelbaren Nachfolgezustand: Als Voraussetzungen gelte im aktuellen Zustand F und es sei $\boldsymbol{K} \models F \rightarrow \mathsf{A}(F \ \mathsf{unless}^{+} \ R)$ die Invarianzaussage.



Als Ergebnis erhält man wie erwartet die Aussage, daß im Nachfolgezustand entweder R ([8]) oder (sonst) F und nicht R ([9],[10]) gilt.

- Fairness:

Als erstes beweist man kurz ein nahezu triviales, aber umso folgenreicheres Lemma. Dieses gibt gleichzeitig Einblick in die Trickkiste der Einbindung von PTL-Formeln, strikten Modaloperatoren und logischen Äquivalenzumformungen. Durch Äquivalenzumformungen entstandene Knoten werden nicht fortlaufend numeriert, sondern durch Anhängen von .1, .2, etc. gekennzeichnet. Im weiteren zeigt sich hier, daß eine vorausgeplante Vorgehensweise notwendig ist.

Lemma: $\mathsf{A}\Box((\Box\mathsf{A}\circ\neg R) \to (\Diamond^+\Box\mathsf{A}\circ\neg R))$

Tableau:

$0: \ \neg(\mathsf{A}\Box((\Box\mathsf{A}\circ\neg R) \to (\Diamond^+\Box\mathsf{A}\circ\neg R)))$ [1]

|

$\lambda : [0, \emptyset, \hat{\infty}]$ [2]

$0: \ \lambda\neg\Box((\Box\mathsf{A}\circ\neg R) \to (\Diamond^+\Box\mathsf{A}\circ\neg R))$ [3]

$0: \ \lambda\Diamond\neg((\Box\mathsf{A}\circ\neg R) \to (\Diamond^+\Box\mathsf{A}\circ\neg R))$ [3.1]

Anwendung von $\lambda P \rightsquigarrow P[\lambda]$:

$0: \ \lambda\Diamond\neg((\lambda\Box\mathsf{A}\circ\neg R) \to (\lambda\Diamond^+\lambda\Box\mathsf{A}\circ\neg R))$ [4]

$0: \ \neg((\lambda\Box\mathsf{A}\circ\neg R) \to (\lambda\Diamond^+\lambda\Box\mathsf{A}\circ\neg R))$ [5]

wie [7], mit $\alpha = 0$

$0: \ (\lambda\Box\mathsf{A}\circ\neg R) \to (\lambda\Diamond^+\lambda\Box\mathsf{A}\circ\neg R)$ [6]

$0: \ (\lambda)\Diamond\neg((\lambda\Box\mathsf{A}\circ\neg R) \to (\lambda\Diamond^+\lambda\Box\mathsf{A}\circ\neg R))$ [7]

$\lambda : [0, \{\neg(\neg(\ldots))\}, \alpha, \emptyset, \hat{\infty}]$ [8]

$\alpha: \ \neg((\lambda\Box\mathsf{A}\circ\neg R) \to (\lambda\Diamond^+\lambda\Box\mathsf{A}\circ\neg R))$ [9]

$\alpha: \ \lambda\Box\mathsf{A}\circ\neg R$ [10]

$\alpha: \ \neg(\lambda\Diamond^+\lambda\Box\mathsf{A}\circ\neg R)$ [11]

$\alpha: \ \lambda\Box^+\neg(\lambda\Box\mathsf{A}\circ\neg R)$ [11.1]

$\alpha: \ \lambda\Box^+\lambda\neg(\Box\mathsf{A}\circ\neg R)$ [11.2]

$\alpha: \ \lambda\Box^+\lambda\Diamond\neg(\mathsf{A}\circ\neg R)$ [11.3]

$\alpha: \ \lambda\Box^+\lambda\Diamond\mathsf{E}\neg(\circ\neg R)$ [11.4]

$\alpha: \ \lambda\Box^+\lambda\Diamond\mathsf{E}\circ R$ [11.5]

|

aus [10]:

$\alpha: \ \mathsf{A}\circ\neg R$ [12]

$\alpha: \ (\lambda)\Box\mathsf{A}\circ\neg R$ [13]

|

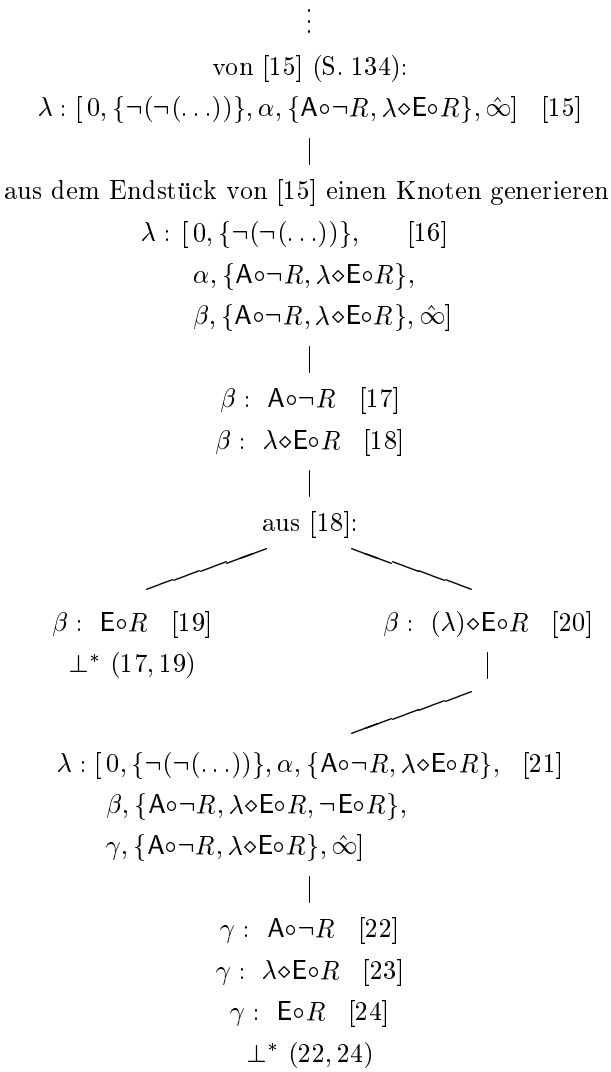
$\lambda : [0, \{\neg(\neg(\ldots))\}, \alpha, \{\mathsf{A}\circ\neg R\}, \hat{\infty}]$ [14]

|

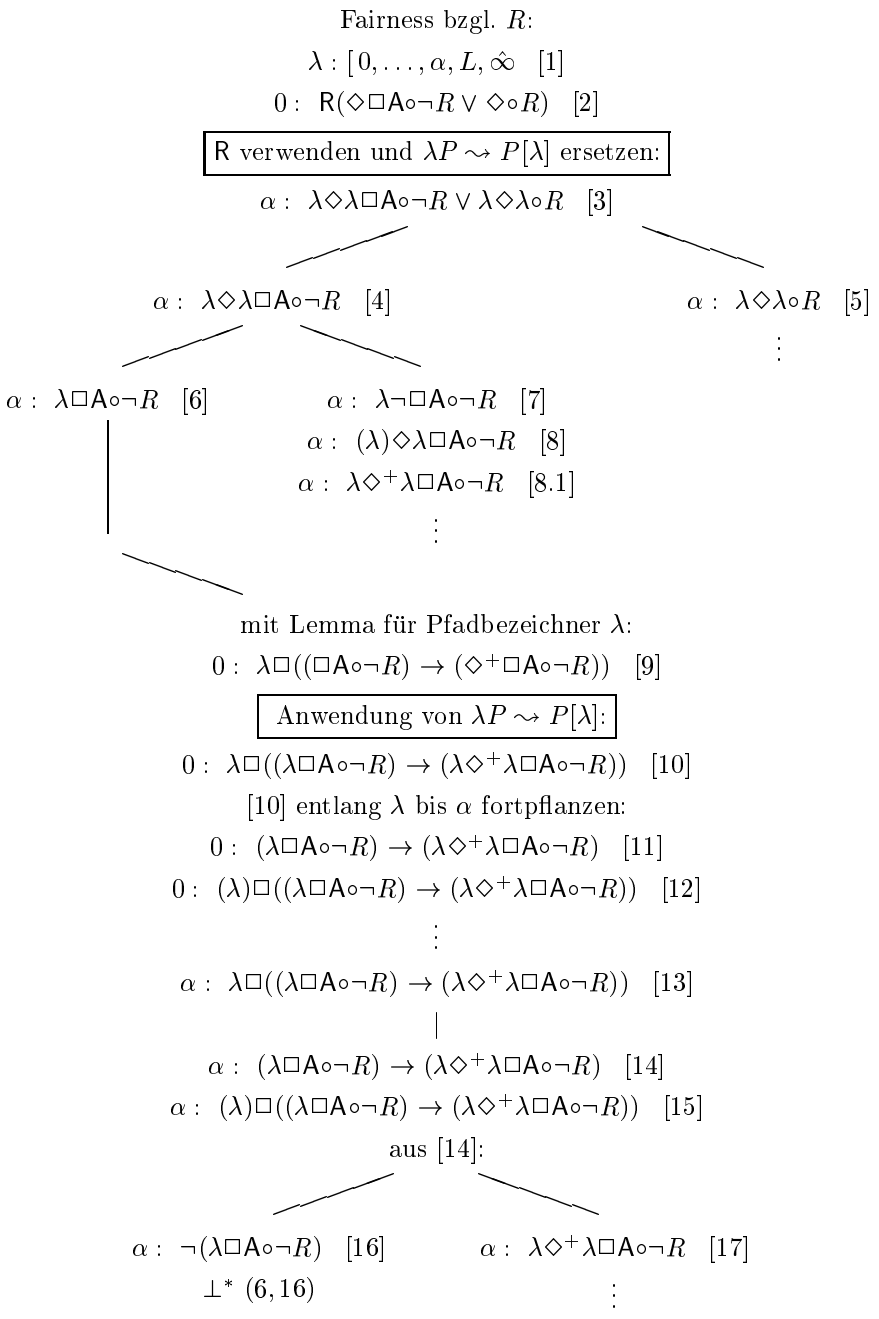
aus [11.5]:

$\lambda : [0, \{\neg(\neg(\ldots))\}, \alpha, \{\mathsf{A}\circ\neg R, \lambda\Diamond\mathsf{E}\circ R\}, \hat{\infty}]$ [15]

↪ S. 135


⋮
von [15] (S. 134):
$\lambda : [0, \{\neg(\neg(\ldots))\}, \alpha, \{\mathsf{A}\circ\neg R, \lambda\diamond\mathsf{E}\circ R\}, \hat{\infty}]$ [15]
aus dem Endstück von [15] einen Knoten generieren:
$\lambda : [0, \{\neg(\neg(\ldots))\},$ [16]
$\alpha, \{\mathsf{A}\circ\neg R, \lambda\diamond\mathsf{E}\circ R\},$
$\beta, \{\mathsf{A}\circ\neg R, \lambda\diamond\mathsf{E}\circ R\}, \hat{\infty}]$
$\beta :\ \mathsf{A}\circ\neg R$ [17]
$\beta :\ \lambda\diamond\mathsf{E}\circ R$ [18]
aus [18]:
$\beta :\ \mathsf{E}\circ R$ [19]
$\bot^*\ (17, 19)$
$\beta :\ (\lambda)\diamond\mathsf{E}\circ R$ [20]
$\lambda : [0, \{\neg(\neg(\ldots))\}, \alpha, \{\mathsf{A}\circ\neg R, \lambda\diamond\mathsf{E}\circ R\},$ [21]
$\beta, \{\mathsf{A}\circ\neg R, \lambda\diamond\mathsf{E}\circ R, \neg\mathsf{E}\circ R\},$
$\gamma, \{\mathsf{A}\circ\neg R, \lambda\diamond\mathsf{E}\circ R\}, \hat{\infty}]$
$\gamma :\ \mathsf{A}\circ\neg R$ [22]
$\gamma :\ \lambda\diamond\mathsf{E}\circ R$ [23]
$\gamma :\ \mathsf{E}\circ R$ [24]
$\bot^*\ (22, 24)$

Dieses Lemma dient dazu, Fairnessbeweise signifikant abzukürzen, da ein durch den Kalkül erzeugter Zweig einem anderen untergeordnet werden kann:

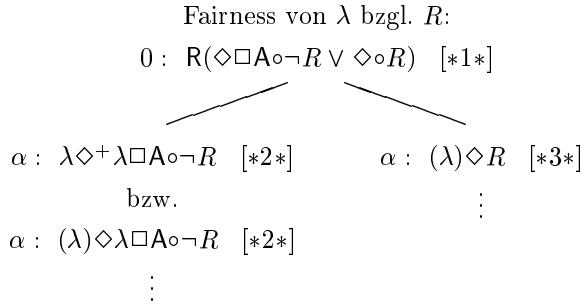


Man hat damit in [17] denselben Knoten wie in [8.1], und bis auf [6] und [7] bei beiden dieselben noch nicht aufgelösten Vorgängerknoten. Gelingt es, den bei [8.1] anschließenden Zweig ohne Verwendung von [7] zu schließen, kann man den Abschluß für den bei [16] anschließenden Zweig gerade übernehmen. Da die eigentliche Fairnessaussage in [8.1] steckt und [7] nur ein durch die vollständige Fallunterscheidung des Tableaux entstandenes Nebenprodukt ist, sollte dies in allen Beweisen der Fall sein.

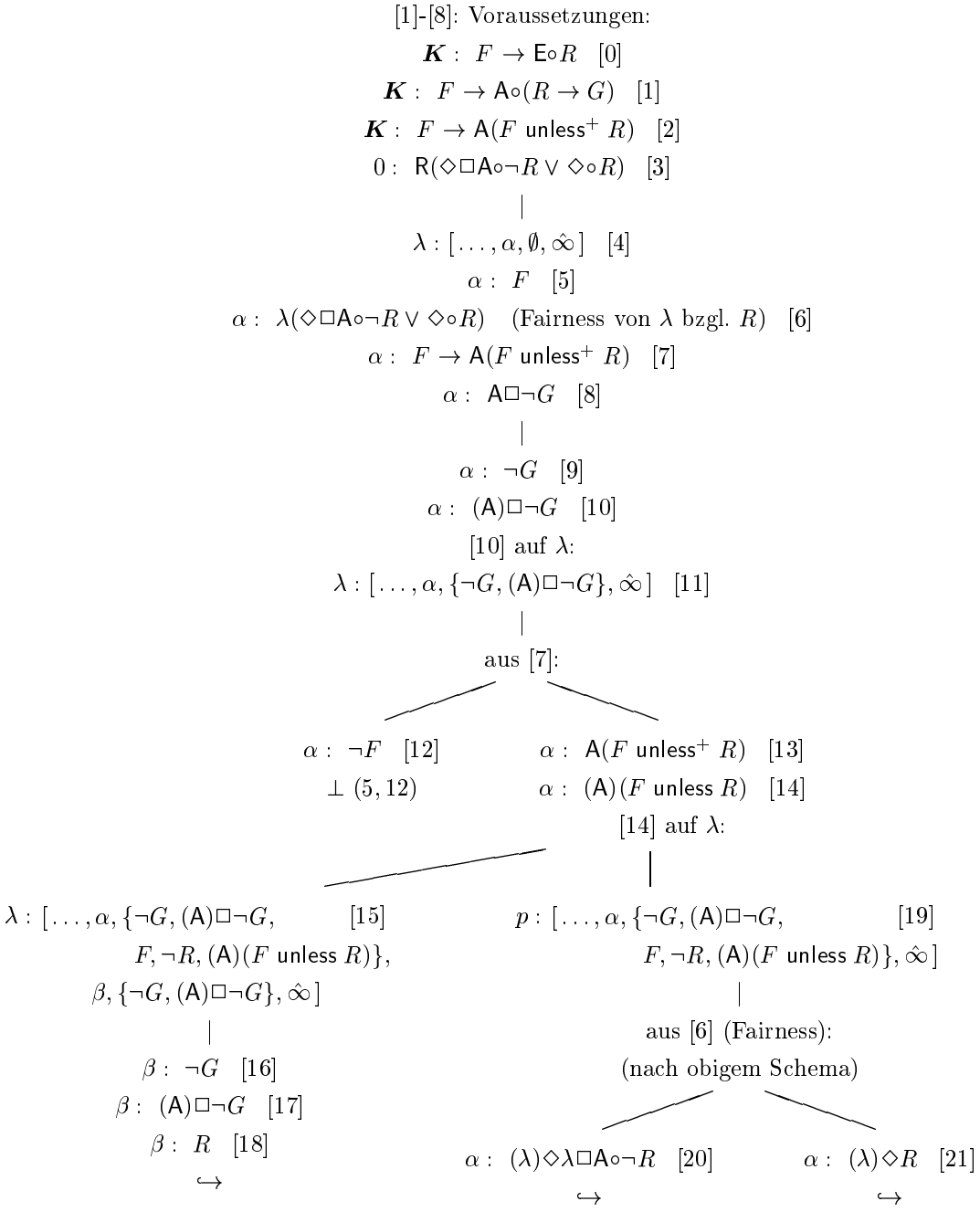
Aus rein praktischen Gründen wird die Formel [5] logisch äquivalent umgeformt:

$$\lambda\Diamond\lambda\circ R \quad \equiv \quad \lambda\circ\lambda\Diamond R \quad \equiv \quad (\lambda)\Diamond R$$

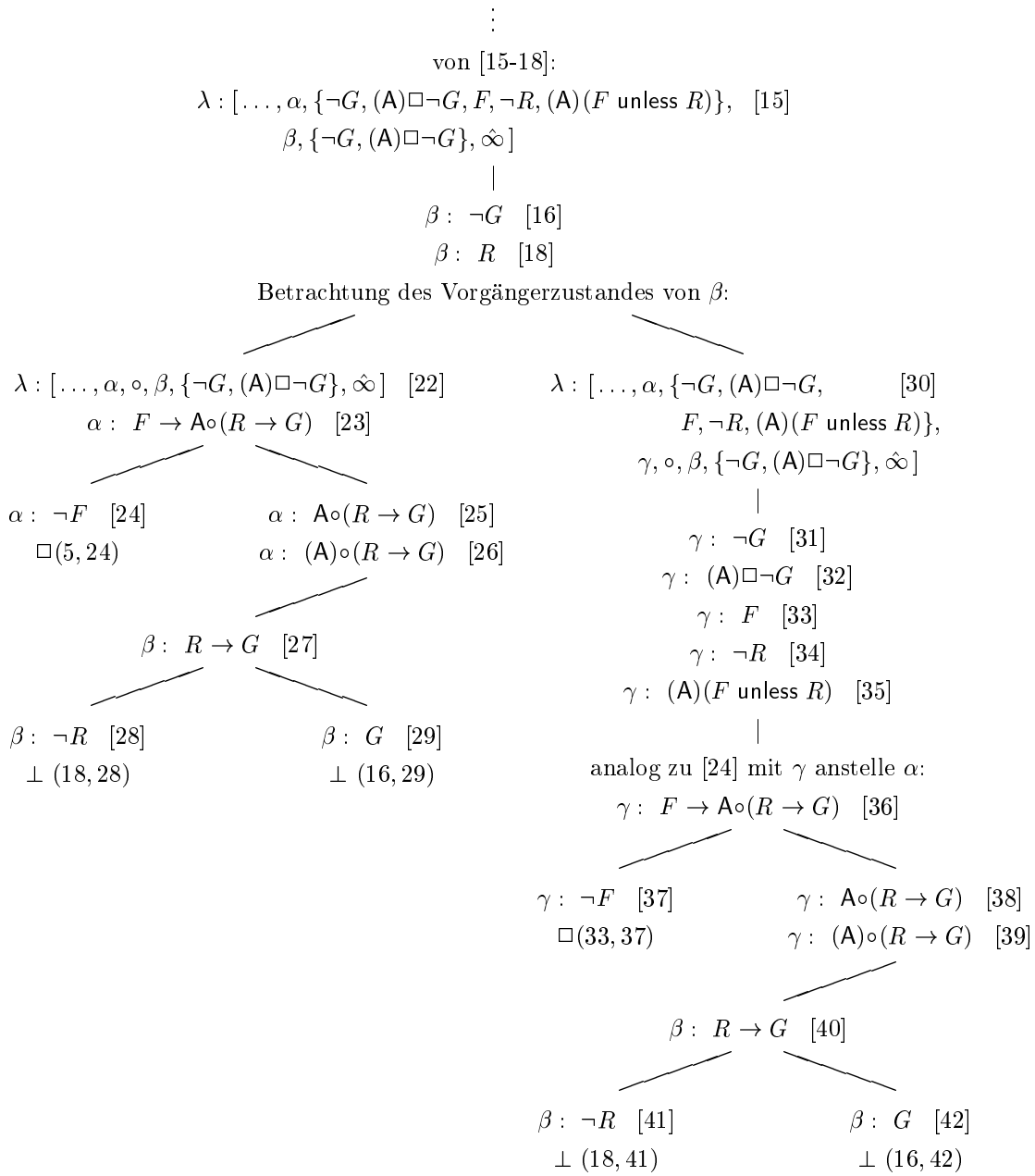
Damit wird das folgende Schema, das aus obigem Tableau durch Weglassen für Beweise im weiteren sachlich irrelevanter Knoten entsteht, zur Verwendung bei Fairnessanwendungen vorgeschlagen:



- Beispiel zu Fairness:



[15]-[18]: Der Zustand β ist zu seinem Vorgänger inkonsistent:



[19],[20]: In diesem Zweig wird die Fairnessbedingung ausgenutzt: Solange F gilt, gibt es immer einen Nachfolgezustand, in dem R gilt.

$$\vdots$$

von [19],[20]:

$$\lambda : [\ldots, \alpha, \{\neg G, (\mathsf{A})\Box\neg G, F, \neg R, (\mathsf{A})(F \text{ unless } R)\}, \hat{\infty}] \quad [19]$$

$$\alpha : \ (\lambda)\Diamond\lambda\Box\mathsf{A}\circ\neg R \quad [20]$$

$$|$$

$$\lambda : [\ldots, \alpha, \{\neg G, (\mathsf{A})\Box\neg G, F, \neg R, (\mathsf{A})(F \text{ unless } R), \lambda\neg\Box\mathsf{A}\circ\neg R\}, \quad [43]$$
$$\delta, \{\neg G, (\mathsf{A})\Box\neg G, F, \neg R, (\mathsf{A})(F \text{ unless } R)\}, \hat{\infty}]$$

$$|$$

$$\delta : \ \neg G \quad [44]$$

$$\delta : \ (\mathsf{A})\Box\neg G \quad [45]$$

$$\delta : \ F \quad [46]$$

$$\delta : \ \neg R \quad [47]$$

$$\delta : \ (\mathsf{A})(F \text{ unless } R) \quad [48]$$

$$\delta : \ \lambda\Box\mathsf{A}\circ\neg R \quad [49]$$

$$|$$

$$\delta : \ \mathsf{A}\circ\neg R \quad [50]$$

$$\delta : \ (\lambda)\Box\mathsf{A}\circ\neg R \quad [51]$$

$$|$$

aus [0]:

$$\delta : \ F \rightarrow \exists\circ R \quad [52]$$

$\delta : \ \neg F \quad [53]$

$\bot\ (46, 53)$

$\delta : \ \mathsf{E}\circ R \quad [54]$

$\kappa : [\delta, \emptyset, \hat{\infty}] \quad [55]$

$\delta : \ \kappa\circ R \quad [56]$

$\delta : \ (\kappa)\circ R \quad [57]$

$q : [\delta, \circ, \varepsilon, \emptyset, \hat{\infty}] \quad [58]$

$\varepsilon : \ R \quad [59]$

$|$

aus [50]:

$\varepsilon : \ (\mathsf{A})\circ\neg R \quad [60]$

[60] auf κ:

$\varepsilon : \ \neg R \quad [61]$

$\bot\ (59, 61)$

[19],[21]: Die Forderung [21] widerspricht dem Pfadinformationsknoten [19], der $\neg R$ für alle Folgezustände von α auf λ fordert:

$$\vdots$$

von [19],[21]:

$$\lambda : [\ldots, \alpha, \{\neg G, (\mathsf{A})\Box\neg G, F, \neg R, (\mathsf{A})(F \text{ unless } R)\}, \hat{\infty}] \quad [19]$$

$$\alpha : \ (\lambda)\Diamond R \quad [21]$$

$$|$$

$$p : [\ldots, \alpha, \{\neg G, (\mathsf{A})\Box\neg G, F, \neg R, (\mathsf{A})(F \text{ unless } R), \neg R\}, \quad [62]$$

$$\eta, \{\neg G, (\mathsf{A})\Box\neg G, F, \neg R, (\mathsf{A})(F \text{ unless } R)\}, \hat{\infty}]$$

$$|$$

$$\eta : \ \neg G \quad [63]$$

$$\eta : \ (\mathsf{A})\Box\neg G \quad [64]$$

$$\eta : \ F \quad [65]$$

$$\eta : \ \neg R \quad [66]$$

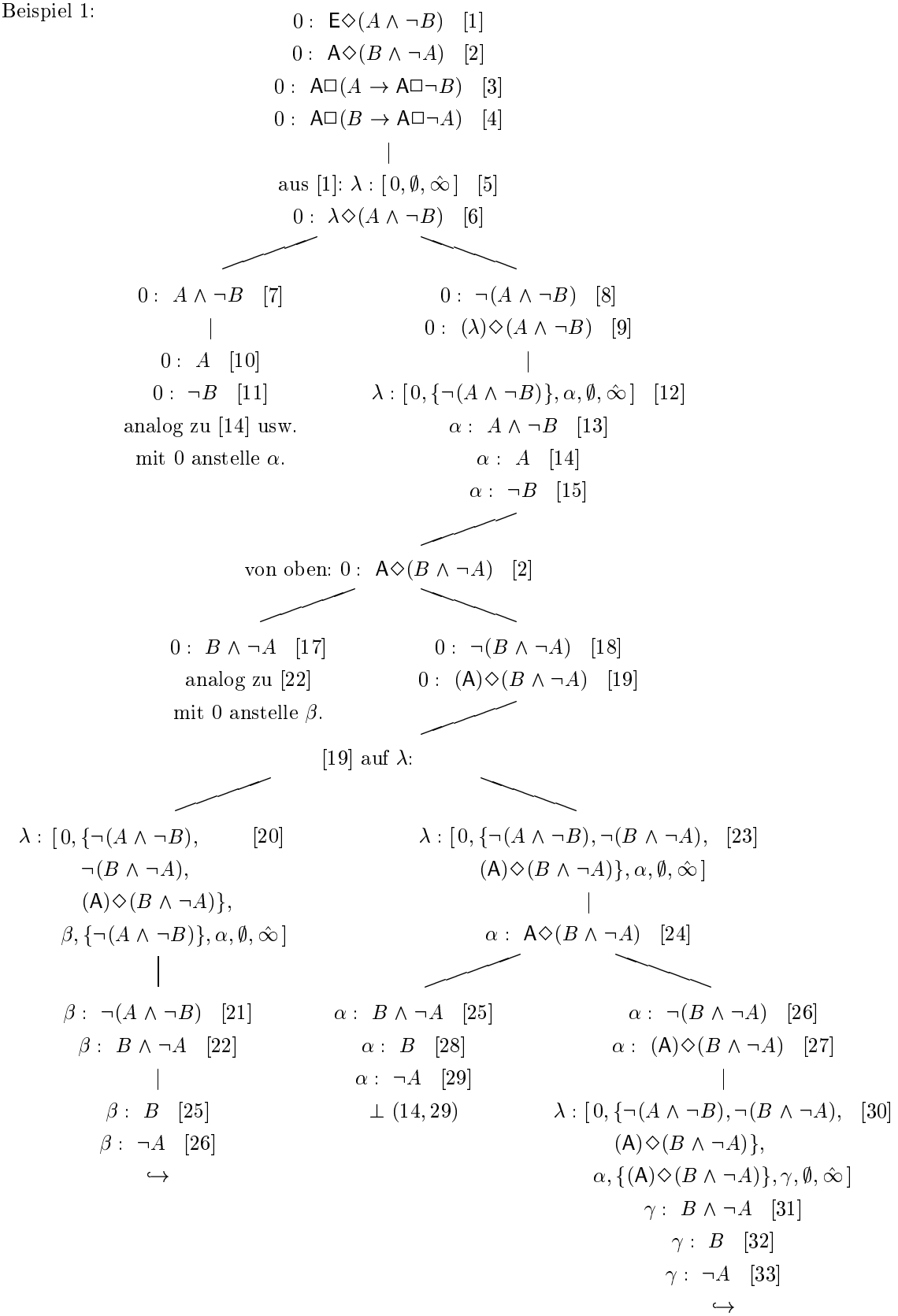
$$\eta : \ (\mathsf{A})(F \text{ unless } R) \quad [67]$$

$$\eta : \ R \quad [68]$$

$$\perp \ (65, 68)$$

- Es folgen noch zwei einfache CTL-Beispiele, die ohne „dirty Tricks“ nur mit CTL-Regeln auskommen.

Beispiel 1:


0 : E◇(A ∧ ¬B) [1]
0 : A◇(B ∧ ¬A) [2]
0 : A□(A → A□¬B) [3]
0 : A□(B → A□¬A) [4]
aus [1]: λ : [0, ∅, ∞̂] [5]
0 : λ◇(A ∧ ¬B) [6]
0 : A ∧ ¬B [7]
0 : A [10]
0 : ¬B [11]
analog zu [14] usw.
mit 0 anstelle α.
0 : ¬(A ∧ ¬B) [8]
0 : (λ)◇(A ∧ ¬B) [9]
λ : [0, {¬(A ∧ ¬B)}, α, ∅, ∞̂] [12]
α : A ∧ ¬B [13]
α : A [14]
α : ¬B [15]
von oben: 0 : A◇(B ∧ ¬A) [2]
0 : B ∧ ¬A [17]
analog zu [22]
mit 0 anstelle β.
0 : ¬(B ∧ ¬A) [18]
0 : (A)◇(B ∧ ¬A) [19]
[19] auf λ:
λ : [0, {¬(A ∧ ¬B), [20]
¬(B ∧ ¬A),
(A)◇(B ∧ ¬A)},
β, {¬(A ∧ ¬B)}, α, ∅, ∞̂]
β : ¬(A ∧ ¬B) [21]
β : B ∧ ¬A [22]
β : B [25]
β : ¬A [26]
↪
λ : [0, {¬(A ∧ ¬B), ¬(B ∧ ¬A), [23]
(A)◇(B ∧ ¬A)}, α, ∅, ∞̂]
α : A◇(B ∧ ¬A) [24]
α : B ∧ ¬A [25]
α : B [28]
α : ¬A [29]
⊥ (14, 29)
α : ¬(B ∧ ¬A) [26]
α : (A)◇(B ∧ ¬A) [27]
λ : [0, {¬(A ∧ ¬B), ¬(B ∧ ¬A), [30]
(A)◇(B ∧ ¬A)},
α, {(A)◇(B ∧ ¬A)}, γ, ∅, ∞̂]
γ : B ∧ ¬A [31]
γ : B [32]
γ : ¬A [33]
↪

$\vdots$

(von [20]-[26]):

$\lambda : [\,0, \{\neg(A \wedge \neg B), \neg(B \wedge \neg A), (\mathsf{A})\Diamond(B \wedge \neg A)\}, \quad [20]$
$\beta, \{\neg(A \wedge \neg B)\}, \alpha, \emptyset, \hat{\infty}\,]$

$|$

$\beta : \ B \quad [25]$

$|$

aus [4]:

$0 : \ B \to \mathsf{A}\Box\neg A \quad [34]$

$0 : \ (\mathsf{A})\Box(B \to \mathsf{A}\Box\neg A) \quad [35]$

[35] auf λ:

$|$

$\lambda : [\,0, \{\neg(A \wedge \neg B), \neg(B \wedge \neg A), (\mathsf{A})\Diamond(B \wedge \neg A), B \to \mathsf{A}\Box\neg A, (\mathsf{A})\Box(B \to \mathsf{A}\Box\neg A)\}, \quad [36]$
$\beta, \{\neg(A \wedge \neg B)\}, \alpha, \emptyset, \hat{\infty}\,]$

$|$

$\beta : \ \mathsf{A}\Box(B \to \mathsf{A}\Box\neg A) \quad [37]$

$|$

$\beta : \ B \to \mathsf{A}\Box\neg A \quad [38]$

$\beta : \ \mathsf{A}\Box(B \to \mathsf{A}\Box\neg A) \quad [39]$

aus [38]:

$\beta : \ \neg B \quad [39]$
$\bot \ (25, 39)$

$\beta : \ \mathsf{A}\Box\neg A \quad [40]$

$|$

$\beta : \ \neg A \quad [41]$

$\beta : \ (\mathsf{A})\Box\neg A \quad [42]$

[42] auf λ:

$|$

$\lambda : [\,0, \{\neg(A \wedge \neg B), \neg(B \wedge \neg A), (\mathsf{A})\Diamond(B \wedge \neg A), B \to \mathsf{A}\Box\neg A, (\mathsf{A})\Box(B \to \mathsf{A}\Box\neg A)\}, \quad [43]$
$\beta, \{\neg(A \wedge \neg B), \neg A, (\mathsf{A})\Box\neg A\}, \alpha, \emptyset, \hat{\infty}\,]$

$|$

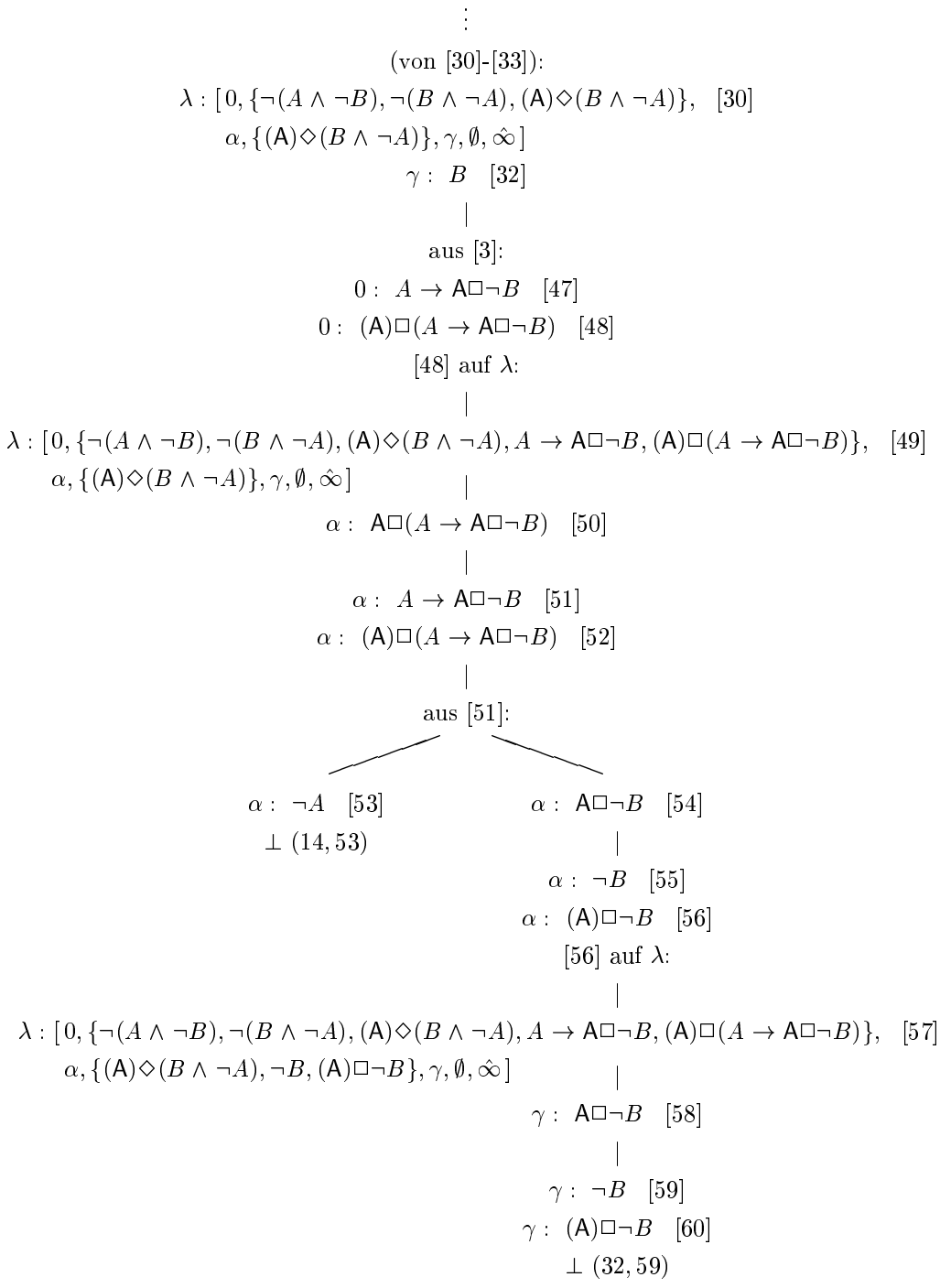
$\alpha : \ \mathsf{A}\Box\neg A \quad [44]$

$|$

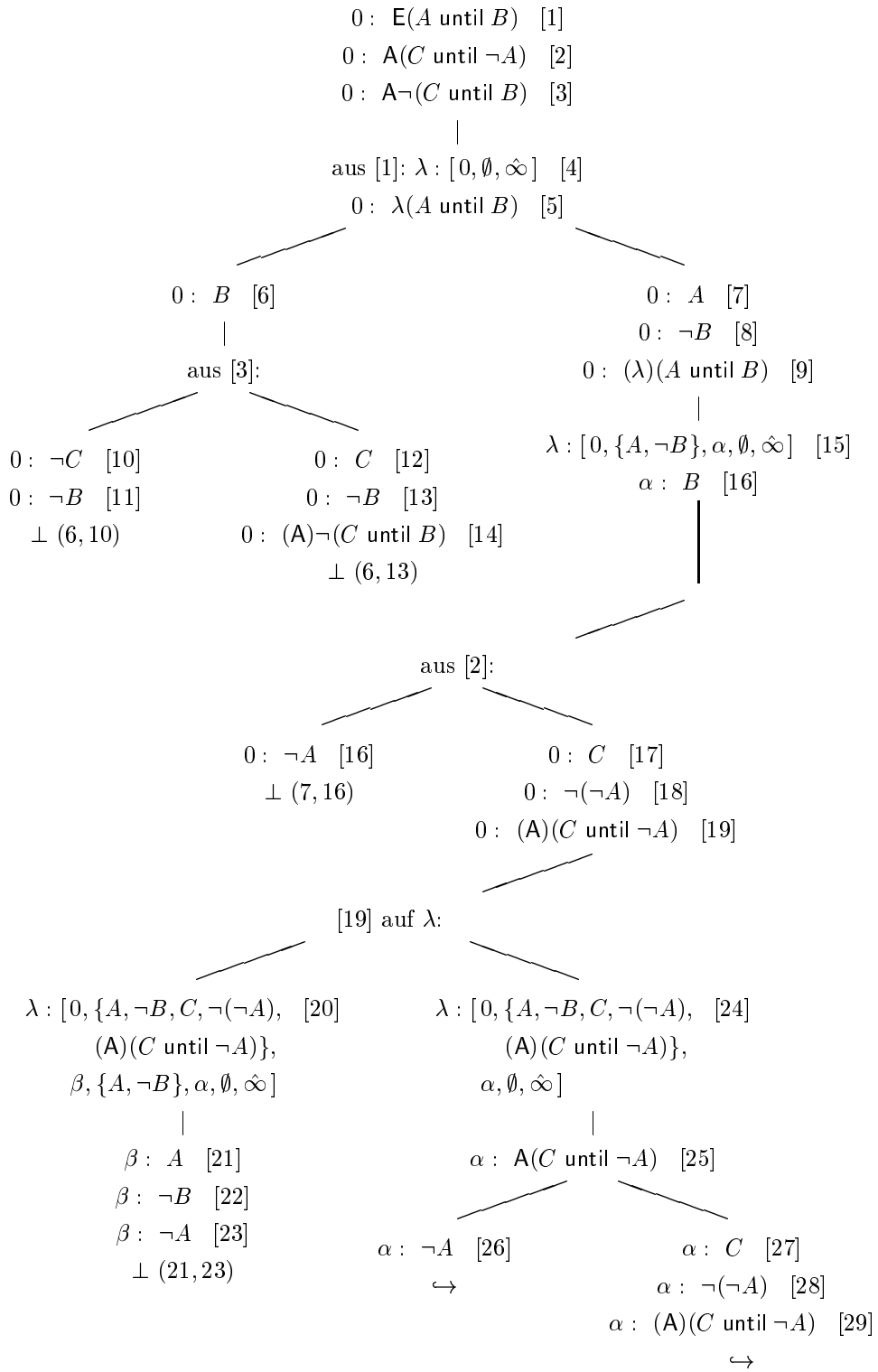
$\alpha : \ \neg A \quad [45]$

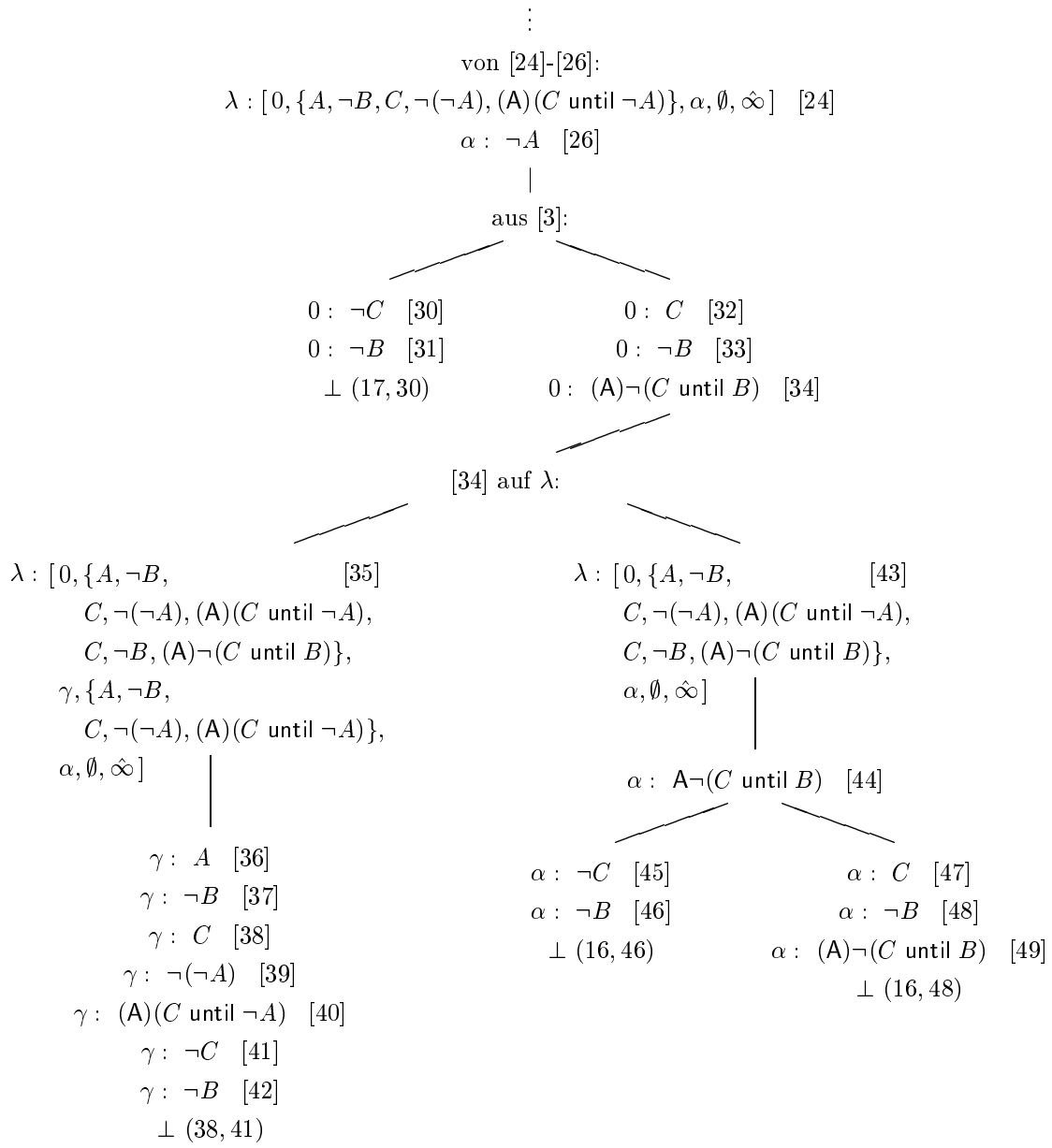
$\alpha : \ (\mathsf{A})\Box\neg A \quad [46]$

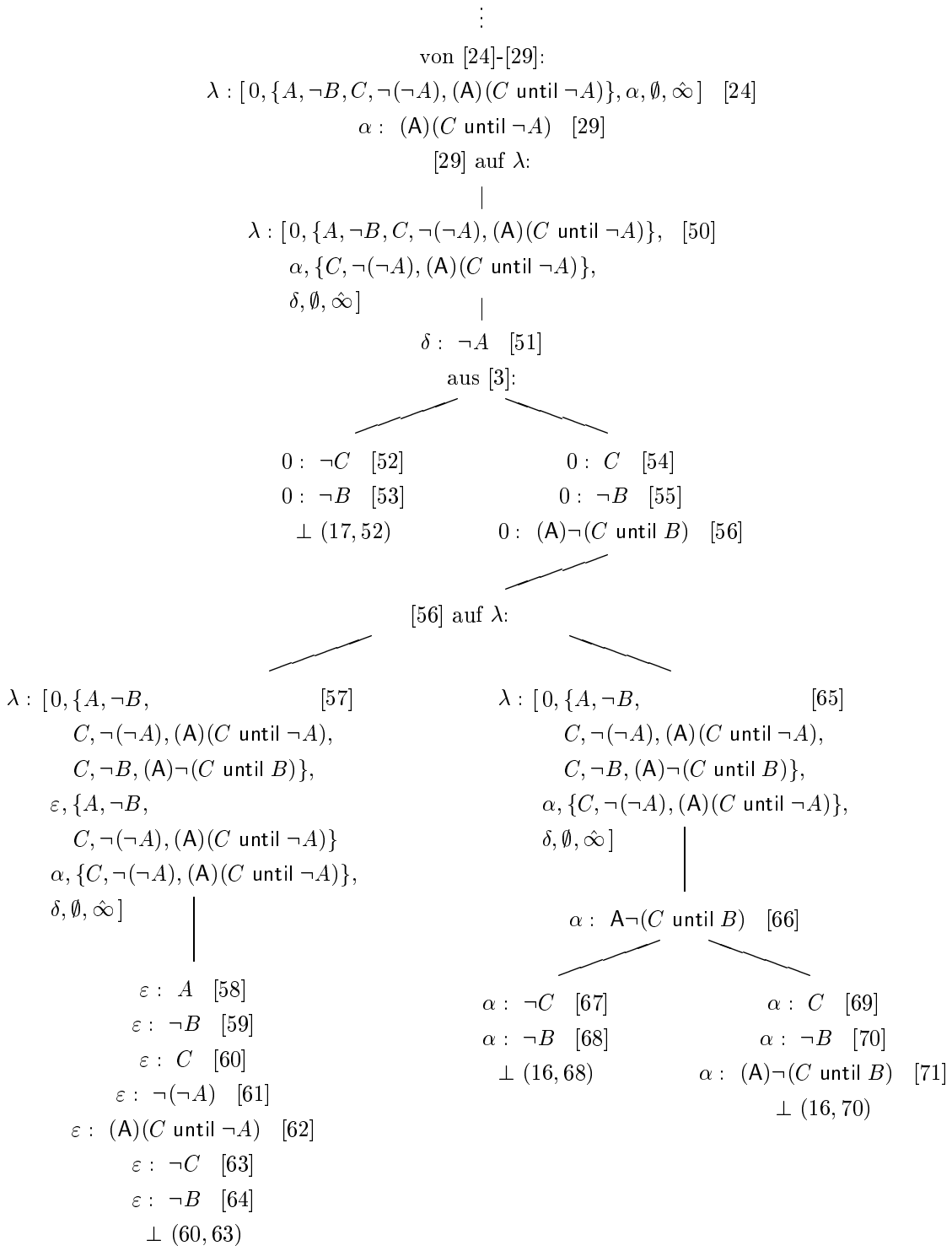
$\bot \ (14, 46)$


(von [30]-[33]):
$\lambda : [0, \{\neg(A \wedge \neg B), \neg(B \wedge \neg A), (\mathsf{A})\Diamond(B \wedge \neg A)\},$ [30]
$\alpha, \{(\mathsf{A})\Diamond(B \wedge \neg A)\}, \gamma, \emptyset, \hat{\infty}]$
$\gamma : B$ [32]
aus [3]:
$0 : A \to \mathsf{A}\Box\neg B$ [47]
$0 : (\mathsf{A})\Box(A \to \mathsf{A}\Box\neg B)$ [48]
[48] auf λ:
$\lambda : [0, \{\neg(A \wedge \neg B), \neg(B \wedge \neg A), (\mathsf{A})\Diamond(B \wedge \neg A), A \to \mathsf{A}\Box\neg B, (\mathsf{A})\Box(A \to \mathsf{A}\Box\neg B)\},$ [49]
$\alpha, \{(\mathsf{A})\Diamond(B \wedge \neg A)\}, \gamma, \emptyset, \hat{\infty}]$
$\alpha : \mathsf{A}\Box(A \to \mathsf{A}\Box\neg B)$ [50]
$\alpha : A \to \mathsf{A}\Box\neg B$ [51]
$\alpha : (\mathsf{A})\Box(A \to \mathsf{A}\Box\neg B)$ [52]
aus [51]:
$\alpha : \neg A$ [53]
$\perp$ (14, 53)
$\alpha : \mathsf{A}\Box\neg B$ [54]
$\alpha : \neg B$ [55]
$\alpha : (\mathsf{A})\Box\neg B$ [56]
[56] auf λ:
$\lambda : [0, \{\neg(A \wedge \neg B), \neg(B \wedge \neg A), (\mathsf{A})\Diamond(B \wedge \neg A), A \to \mathsf{A}\Box\neg B, (\mathsf{A})\Box(A \to \mathsf{A}\Box\neg B)\},$ [57]
$\alpha, \{(\mathsf{A})\Diamond(B \wedge \neg A), \neg B, (\mathsf{A})\Box\neg B\}, \gamma, \emptyset, \hat{\infty}]$
$\gamma : \mathsf{A}\Box\neg B$ [58]
$\gamma : \neg B$ [59]
$\gamma : (\mathsf{A})\Box\neg B$ [60]
$\perp$ (32, 59)

Beispiel 2:


$0:\ \mathsf{E}(A\ \mathsf{until}\ B)$ [1]
$0:\ \mathsf{A}(C\ \mathsf{until}\ \neg A)$ [2]
$0:\ \mathsf{A}\neg(C\ \mathsf{until}\ B)$ [3]
aus [1]: $\lambda : [0, \emptyset, \hat{\infty}]$ [4]
$0:\ \lambda(A\ \mathsf{until}\ B)$ [5]
$0:\ B$ [6]
aus [3]:
$0:\ \neg C$ [10]
$0:\ \neg B$ [11]
$\bot\ (6, 10)$
$0:\ C$ [12]
$0:\ \neg B$ [13]
$0:\ (\mathsf{A})\neg(C\ \mathsf{until}\ B)$ [14]
$\bot\ (6, 13)$
$0:\ A$ [7]
$0:\ \neg B$ [8]
$0:\ (\lambda)(A\ \mathsf{until}\ B)$ [9]
$\lambda : [0, \{A, \neg B\}, \alpha, \emptyset, \hat{\infty}]$ [15]
$\alpha:\ B$ [16]
aus [2]:
$0:\ \neg A$ [16]
$\bot\ (7, 16)$
$0:\ C$ [17]
$0:\ \neg(\neg A)$ [18]
$0:\ (\mathsf{A})(C\ \mathsf{until}\ \neg A)$ [19]
[19] auf λ:
$\lambda : [0, \{A, \neg B, C, \neg(\neg A),\ (\mathsf{A})(C\ \mathsf{until}\ \neg A)\}, \beta, \{A, \neg B\}, \alpha, \emptyset, \hat{\infty}]$ [20]
$\beta:\ A$ [21]
$\beta:\ \neg B$ [22]
$\beta:\ \neg A$ [23]
$\bot\ (21, 23)$
$\lambda : [0, \{A, \neg B, C, \neg(\neg A),\ (\mathsf{A})(C\ \mathsf{until}\ \neg A)\}, \alpha, \emptyset, \hat{\infty}]$ [24]
$\alpha:\ \mathsf{A}(C\ \mathsf{until}\ \neg A)$ [25]
$\alpha:\ \neg A$ [26]
$\hookrightarrow$
$\alpha:\ C$ [27]
$\alpha:\ \neg(\neg A)$ [28]
$\alpha:\ (\mathsf{A})(C\ \mathsf{until}\ \neg A)$ [29]
$\hookrightarrow$


⋮
von [24]-[26]:
$\lambda : [0, \{A, \neg B, C, \neg(\neg A), (\mathsf{A})(C \text{ until } \neg A)\}, \alpha, \emptyset, \hat{\infty}]$ [24]
$\alpha :\ \neg A$ [26]
aus [3]:
$0 :\ \neg C$ [30]
$0 :\ \neg B$ [31]
$\bot\ (17, 30)$
$0 :\ C$ [32]
$0 :\ \neg B$ [33]
$0 :\ (\mathsf{A})\neg(C \text{ until } B)$ [34]
[34] auf λ:
$\lambda : [0, \{A, \neg B,$ [35]
$C, \neg(\neg A), (\mathsf{A})(C \text{ until } \neg A),$
$C, \neg B, (\mathsf{A})\neg(C \text{ until } B)\},$
$\gamma, \{A, \neg B,$
$C, \neg(\neg A), (\mathsf{A})(C \text{ until } \neg A)\},$
$\alpha, \emptyset, \hat{\infty}]$
$\gamma :\ A$ [36]
$\gamma :\ \neg B$ [37]
$\gamma :\ C$ [38]
$\gamma :\ \neg(\neg A)$ [39]
$\gamma :\ (\mathsf{A})(C \text{ until } \neg A)$ [40]
$\gamma :\ \neg C$ [41]
$\gamma :\ \neg B$ [42]
$\bot\ (38, 41)$
$\lambda : [0, \{A, \neg B,$ [43]
$C, \neg(\neg A), (\mathsf{A})(C \text{ until } \neg A),$
$C, \neg B, (\mathsf{A})\neg(C \text{ until } B)\},$
$\alpha, \emptyset, \hat{\infty}]$
$\alpha :\ \mathsf{A}\neg(C \text{ until } B)$ [44]
$\alpha :\ \neg C$ [45]
$\alpha :\ \neg B$ [46]
$\bot\ (16, 46)$
$\alpha :\ C$ [47]
$\alpha :\ \neg B$ [48]
$\alpha :\ (\mathsf{A})\neg(C \text{ until } B)$ [49]
$\bot\ (16, 48)$


⋮
von [24]-[29]:
$\lambda : [0, \{A, \neg B, C, \neg(\neg A), (\mathsf{A})(C \text{ until } \neg A)\}, \alpha, \emptyset, \hat{\infty}]$ [24]
$\alpha :$ $(\mathsf{A})(C \text{ until } \neg A)$ [29]
[29] auf λ:
$\lambda : [0, \{A, \neg B, C, \neg(\neg A), (\mathsf{A})(C \text{ until } \neg A)\},$ [50]
$\alpha, \{C, \neg(\neg A), (\mathsf{A})(C \text{ until } \neg A)\},$
$\delta, \emptyset, \hat{\infty}]$
$\delta :$ $\neg A$ [51]
aus [3]:
$0 :$ $\neg C$ [52]
$0 :$ $\neg B$ [53]
$\perp$ (17, 52)
$0 :$ C [54]
$0 :$ $\neg B$ [55]
$0 :$ $(\mathsf{A})\neg(C \text{ until } B)$ [56]
[56] auf λ:
$\lambda : [0, \{A, \neg B,$ [57]
$C, \neg(\neg A), (\mathsf{A})(C \text{ until } \neg A),$
$C, \neg B, (\mathsf{A})\neg(C \text{ until } B)\},$
$\varepsilon, \{A, \neg B,$
$C, \neg(\neg A), (\mathsf{A})(C \text{ until } \neg A)\}$
$\alpha, \{C, \neg(\neg A), (\mathsf{A})(C \text{ until } \neg A)\},$
$\delta, \emptyset, \hat{\infty}]$
$\varepsilon :$ A [58]
$\varepsilon :$ $\neg B$ [59]
$\varepsilon :$ C [60]
$\varepsilon :$ $\neg(\neg A)$ [61]
$\varepsilon :$ $(\mathsf{A})(C \text{ until } \neg A)$ [62]
$\varepsilon :$ $\neg C$ [63]
$\varepsilon :$ $\neg B$ [64]
$\perp$ (60, 63)
$\lambda : [0, \{A, \neg B,$ [65]
$C, \neg(\neg A), (\mathsf{A})(C \text{ until } \neg A),$
$C, \neg B, (\mathsf{A})\neg(C \text{ until } B)\},$
$\alpha, \{C, \neg(\neg A), (\mathsf{A})(C \text{ until } \neg A)\},$
$\delta, \emptyset, \hat{\infty}]$
$\alpha :$ $\mathsf{A}\neg(C \text{ until } B)$ [66]
$\alpha :$ $\neg C$ [67]
$\alpha :$ $\neg B$ [68]
$\perp$ (16, 68)
$\alpha :$ C [69]
$\alpha :$ $\neg B$ [70]
$\alpha :$ $(\mathsf{A})\neg(C \text{ until } B)$ [71]
$\perp$ (16, 70)

Notationskonventionen

An dieser Stelle wird ein nach thematischen Gesichtspunkten geordneter Überblick über die verwendeten Zeichen gegeben. Es wurde versucht, für thematisch zusammengehörige Dinge denselben Schriftstil zu verwenden, sowie für inhaltlich zusammengehörige Elemente denselben Buchstaben (z.B. $\mathcal{G}$, $\boldsymbol{G}$, Γ, g, γ). Dennoch ließ es sich nicht vermeiden, daß einige Zeichen verschiedene Bedeutungen besitzen (z.B. p, π, α-δ). An den betreffenden Stellen sollte die jeweilige Bedeutung jedoch aus dem Kontext ersichtlich sein.

Grundlagen:

$\mathbf{N}$	Natürliche Zahlen, einschließlich 0
$\hat{\mathbf{N}}$	$\mathbf{N} \cup \{\infty\}$
M^N	Menge der Abbildungen von N nach M
2^N	Potenzmenge von N

Prädikatenlogik:

Σ	Signatur
$\boldsymbol{I}$	prädikatenlogische Interpretation einer Signatur, $\boldsymbol{I} = (I, \boldsymbol{U})$, auch Auswertungsfunktion (s. dort)
$I(f),\ I(p)$:	Interpretationen der einzelnen Signatursymbole, $I(f) : \boldsymbol{U}^n \to \boldsymbol{U},\ I(p) \subseteq \boldsymbol{U}^n$
f	ein einzelnes Funktionssymbol
p	ein einzelnes Prädikatssymbol
$\mathsf{ord}(f), \mathsf{ord}(p)$	Stelligkeit eines Funktions- bzw. Prädikatssymbols
x, X	Variablen
c	eine Konstante
Term_Σ	Menge der Terme über der Signatur Σ
s, t	Terme
$\hat{t}$	führendes Funktionssymbol eines Terms t
$\mathsf{arg}(t)$	Argumentvektor eines Terms: $\mathsf{arg}(f(t_1, \ldots, t_n)) = (t_1, \ldots, t_n)$
A	ein prädikatenlogisches Atom
F	eine prädikatenlogische Formel
σ	eine Substitution, einelementige Substitutionen werden als $[x \leftarrow s]$ geschrieben
$\sigma(t)$	auch $[x \leftarrow s]t$: Anwendung der Substitution σ auf t
$t[s_1/s_2]$	syntaktisches Ersetzen aller Auftreten von s_2 durch s_1 in t (auch für Formeln)
α-,β-,γ-,δ-Formeln:	Arten von PL-Formeln (konj.,disj.,univ.,exist.)
$\mathsf{free}(F)$	die in einer Formel(menge) frei auftretenden Variablen
$\boldsymbol{U}$	Universum
Ξ	Menge der Variablenbelegungen
χ, ξ	Variablenbelegungen $\chi : \mathsf{Var} \to \boldsymbol{U}$
χ^a_X	modifizierte Variablenbelegung
$\boldsymbol{I}(t, \chi)$	Auswertung des Terms t unter Belegung χ in Interpretation $\boldsymbol{I}$
$\models$	prädikatenlogische Wahrheitsrelation, $(\boldsymbol{I}, \chi) \models F$

Evolving Algebras:

$\mathcal{A}$	eine Evolving Algebra
$\Sigma(\mathcal{A})$	Signatur der EA
$\Sigma^c(\mathcal{A})$	zustandsunabhängig interpretierter Anteil von $\Sigma(\mathcal{A})$
$\mathcal{S}(\mathcal{A})$	Superuniversum der EA
$\mathcal{Z}(\mathcal{A})$	Startzustand von $\mathcal{A}$
$\mathcal{R}(\mathcal{A})$	Menge der Übergangsregeln von $\mathcal{A}$
$\mathcal{G}(\mathcal{A})$	Menge aller von $\mathcal{Z}(\mathcal{A})$ erreichbaren statischen Algebren
$g, h \in \mathcal{G}(\mathcal{A})$	einzelne statische Algebren
$r \in \mathcal{R}(\mathcal{A})$	eine einzelne Übergangsregel
$r(g)$	durch Anwendung von r auf g entstehende statische Algebra
p	eine Berechnungsfolge in einer Evolving Algebra

statische Algebren:

$g = \mathcal{I}$	eine einzelne statische Algebra, $\mathcal{I} = (I, \mathcal{S})$
Σ	Signatur
f	ein Funktionssymbol
$\mathsf{dom}(f)$	Definitionsbereich von f
$\mathcal{S}$	Superuniversum einer statischen Algebra
$\mathcal{U}$	ein einzelnes (Sorten-)Universum einer EA
I	Interpretation der Signatursymbole
$I(f)$	Interpretation einer einzelnen Funktion, $I(f) : \mathcal{U}^n \to \mathcal{U}$
$\mathcal{I}(t)$	Auswertung eines Termes $t \in \mathsf{Term}_\Sigma$

(temporallogische) Kripke-Strukturen:

$\boldsymbol{K}$	eine Kripke-Struktur $\boldsymbol{K} = (\boldsymbol{G}, \boldsymbol{R}, \boldsymbol{M})$
$\boldsymbol{K}^{[\mathcal{A}]}$	die zu einer Evolving Algebra $\mathcal{A}$ gehörende Kripke-Struktur
$\boldsymbol{K}^{\mathcal{A}}$	dgl. mit Zusatzinformationen über angewendete Regel(n)
$\Sigma(\boldsymbol{K})$	Signatur einer Kripke-Struktur $\boldsymbol{K}$
$\Sigma^c(\boldsymbol{K})$	zustandsunabhängig interpretierter Anteil von $\Sigma(\boldsymbol{K})$
$\boldsymbol{K}(t, \chi)$	Auswertung von $t \in \mathsf{Term}_{\Sigma^c(\boldsymbol{K})}$ unter Belegung χ in der Struktur $\boldsymbol{K}$
$\boldsymbol{G}$	Menge der Zustände in $\boldsymbol{K}$
$g, h \in \boldsymbol{G}$	einzelne Zustände
$\boldsymbol{R}$	Zugänglichkeitsrelation, $\boldsymbol{R} \subseteq \boldsymbol{G} \times \boldsymbol{G}$
$\boldsymbol{R}^+$	strikte transitive Hülle von $\boldsymbol{R}$
$\boldsymbol{R}^*$	reflexiv-transitive Hülle von $\boldsymbol{R}$
$\boldsymbol{M}$	ordnet jedem $g \in \boldsymbol{G}$ eine aussagen- oder prädikatenlogische Struktur zu:
$\boldsymbol{M}(g)$	die dem Zustand $g \in \boldsymbol{G}$ zugeordnete aussagen- oder prädikatenlogische Struktur, $\boldsymbol{M}(g) = (M(g), \boldsymbol{U}(g))$
$\boldsymbol{U}(\boldsymbol{K})$	konstantes Universum von $\boldsymbol{K}$
$\boldsymbol{P}(\boldsymbol{K})$	Menge der Pfade in $\boldsymbol{K}$, $\boldsymbol{P}(\boldsymbol{K}) \subseteq \boldsymbol{G}^{\mathrm{N}}$
p, q	einzelne Pfade in $\boldsymbol{K}$
$p{\downarrow_i}$	der i-te Zustand auf dem Pfad p
$p\,\vert_i$	das Endstück von p ab dem i-ten Zustand (einschließlich)

Temporallogik:

CTL, CTL$^+$, CTL* die hier verwendeten Temporallogiken

modallogische Operatoren:

$\circ$	„nexttime“
$\Box$	„always“
$\Diamond$	„sometimes“
until,unless	selbsterklärend

A, E	Pfadquantoren
F, G	temporallogische Zustandsformeln
P, Q	temporallogische Pfadformeln

μ-,ν-,π-,ρ-Formeln: Arten von TL-Formeln (next,alw.,somet.,sonst.)

x, X	Variablen
Ξ	Menge der Variablenbelegungen
χ, ξ	Variablenbelegungen $\chi : \mathsf{Var} \to \boldsymbol{U}$
χ^a_X	modifizierte Variablenbelegung
$\models$	Wahrheitsrelation in aussagenlogischen Kripke-Strukturen, $g \models F$, $p \models P$ und $\boldsymbol{K} \models F$
$\models$	Wahrheitsrelation in prädikatenlogischen Kripke-Strukturen, $(g, \chi) \models F$, $(p, \chi) \models P$ und $(\boldsymbol{K}, \chi) \models F$
$[0], 0$	$0 \in \boldsymbol{G}$, Wurzel der eine EA modellierenden Kripke-Struktur $\boldsymbol{K}^{[\mathcal{A}]}$ oder $\boldsymbol{K}^{\mathcal{A}}$, entspricht dem Startzustand der Evolving Algebra
$\models^0$	Wahrheitsrelation für Kripke-Strukturen, die eine Evolving Algebra modellieren; $\boldsymbol{K} \models^0 \mathcal{F} :\Leftrightarrow 0 \models \mathcal{F}$

Tableaux:

$\mathcal{T}$	ein Tableau
$\Sigma_{\mathcal{T}}$	Signatur, über der das Tableau erzeugt wird; $\Sigma_{\mathcal{T}} = \Sigma_L \cup \Sigma_T$
Σ_L	Logischer Anteil der Signatur, Signatur der Eingabeformelmenge und zusätzliche Skolemfunktionen, wird von $\boldsymbol{K}$ interpretiert
Σ_T	Temporaler Teil der Signatur, $\Sigma_T = \hat{\Gamma} \cup \hat{\Lambda}$, wird von Ω interpretiert
$\mathcal{F}$	eine Formelmenge
T	ein Zweig in einem Tableau; ist eine Formelmenge $\mathcal{F}$
F, G	Zustandsformeln
P, Q	Pfadformeln
I	eine Pfadinformationsformel
P_1, P_2, P_3	ausgezeichnete Pfadformeln zu einer Pfadformel P
R	universeller Pfadquantor für Reactivity-Formeln
$P[\lambda]$	Bindung einer PTL-Pfadformel an Pfadbezeichner λ
$\hat{\lambda}$	Menge der Pfadsymbole
$\hat{\lambda}, \hat{\kappa}$	Pfadsymbole
$\hat{\Lambda}$	Menge der Pfadbezeichner
λ, κ	Pfadbezeichner; $\lambda = \hat{\lambda}(t_1, \ldots, t_n)$
$\hat{\Gamma}$	Menge der Präfixsymbole
$\hat{\gamma}, \hat{\alpha}, \hat{0}$	Präfixsymbole
$\hat{\infty}$	Sonderzeichen, Verwendung ähnlich einem nullstelligen Präfixsymbol
$\hat{\Lambda}$	Menge der Präfixe
$\gamma, \alpha, \beta, \delta$	Präfixe; $\gamma = \hat{\gamma}(t_1, \ldots, t_n)$
Ω	eine P&P- („Präfixe-und-Pfade-“) Interpretation, $\Omega = (\Phi, \Pi, \Psi)$
Φ	Auswertung der Pfadbezeichner, $\Phi : \Lambda \times \Xi \to \boldsymbol{P}(\boldsymbol{K})$, auch $\Phi : \Lambda \times \Xi \times \mathbb{N} \to \boldsymbol{G}$
$\phi(\hat{\lambda})$	Interpretation eines Pfadsymbols: $\phi(\hat{\lambda}) : \mathcal{U}^n \to \boldsymbol{P}(\boldsymbol{K})$
Π	Auswertung von Paaren von Pfadbezeichnern und Präfixen zu Indizes: $\Pi : \Lambda \times \Gamma \times \Xi \to \mathbb{N}$
$\pi(\hat{\lambda}, \hat{\gamma})$	Interpretation von Pfadsymbol-Präfixsymbol-Paaren: $\pi(\hat{\lambda}, \hat{\gamma}) : \mathcal{U}^n \times \mathcal{U}^m \to \mathbb{N}$
Ψ	Auswertung der Präfixe: $\Psi : \Gamma \times \Xi \to \boldsymbol{G}$
$\psi(\hat{\gamma})$	Interpretation der Präfixsymbole: $\psi(\hat{\gamma}) : \mathcal{U}^n \to \boldsymbol{G}$
I	eine Pfadinformationsformel
L	eine Liste von Zustandsformeln
$\circ$	Zeichen für direkte Nachfolgerbeziehung zwischen Präfixen
$\Vdash\!\!\!\Vdash$	Wahrheitsrelation, $(\boldsymbol{K}, \chi) \Vdash\!\!\!\Vdash \mathcal{F}$
$\models$	Wahrheitsrelation, $(\boldsymbol{K}, \Omega, \chi) \models \mathcal{F}$
$\perp$	Abschlußsymbol für Tableaux.

Literatur

[AB94] R. Hähnle: *Vorlesung "Automatisches Beweisen".* Universität Karlsruhe, 1994.

[BH92] B. Beckert, R. Hähnle: *An Improved Method for Adding Equality to Free Variable Semantic Tableaux.* LNCS 507: CADE 11 (1992), S. 507-521.

[BMP81] M. Ben-Ari, Z. Manna, A. Pnueli: *The Temporal Logic of Branching Time.* Proceedings of the Annual ACM Symposium on Principles of Programming Languages, 1981.

[CE81] E. M. Clarke, E. A. Emerson: *Design and Synthesis of Synchronization Skeletons using Branching Time Temporal Logic.* Proc. of the IBM Workshop on Logics of Programs, LNCS 131, 1981.

[CES86] E. M. Clarke, E. A. Emerson, A. P. Sistla: *Automatic Verification of Finite-State Concurrent Systems Using Temporal Logic Specifications.* ACM Transactions on Programming Languages and Systems, Vol. 8, No. 2, April 1986, S. 244-263.

[EC80] E. A. Emerson, E. M. Clarke: *Characterizing Properties of Parallel Programs as Fixpoints.* Proc. of the 7th Int. Colloquium on Automata, Languages and Programming. Lecture Notes in Computer Science, 85, Springer Verlag, New York, 1981.

[EH82] E. A. Emerson, J. Y. Halpern: *Decision Procedures and Expressiveness in the Temporal Logic of Branching Time.* 14. Proceedings of the Annual ACM Symposium on Computing, 1982, S. 169-180.

[EH83] E. A. Emerson, J. Y. Halpern: *"Sometimes" and "not never" revisited: On Branching Time versus Linear Time in Temporal Logic.* Proceedings of the Annual ACM Symposium on Principles of Programming Languages (Austin, Texas, Januar 1983).

[EL85] E. A. Emerson, C.-L. Lei: *Modalities for Model Checking: Branching Time Strikes Back.* Proceedings of the 12th ACM Symposium on Principles of Programming Languages (New Orleans, Januar 1985).

[Fit90] M. C. Fitting: *First Order Logic and Automated Theorem Proving.* Springer, New York.

[FL79] M. J. Fischer, R. E. Ladner: *Propositional Dynamic Logic of Regular Programs.* JCSS vol. 18, 1979, S. 194-211.

[Gu88] Y. Gurevich: *Logic and the Challenge of Computer Science.* Trends in Theoretical Computer Science (E. Börger, ed.), Computer Science Press, 1988, S. 1-57.

[Gu91] Y. Gurevich: *Evolving Algebras. A Tutorial Introduction.* EATCS Bull. Feb. 1991.

[Gu92] Y. Gurevich: *Evolving Algebras: An Attempt to Discover Semantics.* Current Trends in Theoretical Computer Science, World Scientific, 1993, S. 266-292.

[HS94] R. Hähnle, P. H. Schmitt: *The liberalized δ-rule in free variable semantic Tableaux.* Journal of Automated Reasoning, Vol. 13.2, 1994.

[Hu94] J. K. Huggins: *An Evolving Algebra Interpreter.* Tech. Rept., Univ. of Michigan, Ann Arbor.

[La80] L. Lamport: *"Sometimes" is Sometimes "Not Never".* 7th Annual ACM Symp. on Principles of Programming Languages, 1980.

[LKK93] P. C. Lockemann, G. Krüger, H. Krumm: *Telekommunikation und Datenhaltung.* Hanser, 1993.

[LPS81] D. Lehmann, A. Pnueli, J. Stavi: *Impartiality, Justice, and Fairness: The Ethics of Concurrent Termination.* Automata, Languages and Programming. Lecture Notes in Computer Science 115, Springer, 1981, S. 264-277.

[May95b] W. May: *Fallstudie: Korrektheitsbeweis des Alternating-Bit-Protokolls.* Anhang zur Diplomarbeit, Universität Karlsruhe, 1995.

[MP92] Z. Manna, A. Pnueli: *The Temporal Logic of Reactive and Concurrent Systems: Specification.* Springer-Verlag, 1992.

[MS91] W. Menzel, P. H. Schmitt: *Formale Systeme.* Vorlesungsskript Universität Karlsruhe, 1991.

[Pn77] A. Pnueli: *The Temporal Logic of Programs.* 18th Annual Symp. on Foundations of Computer Science, Nov.77, S. 46 ff.

[Ree87] S. V. Reeves: *Semantic Tableaux as a Framework for Automated Theorem-Proving.* Dept. of Comp. Sc. and Statistics, Queen Mary College, Univ. of London.

[Schm87] P. H. Schmitt: *The THOT Theorem Prover.* Tech. rept. TR-87.09.007. IBM Heidelberg Scientific Center.

[Schm92] P. H. Schmitt: *Skriptum zur Vorlesung "Nichtklassische Logiken".* Universität Karlsruhe, 1992.

[Sm68] R. Smullyan: *First-Order Logic.* Springer, New York.

[Wol85] P. Wolper: *The Tableau Method for Temporal Logic.* Logique et Analyse, 110-111, Bd. 28, June-Sept. 1985.