Automatisches Beweisen—Vertiefung
Modallogik & Temporallogik

Christoph Zengler

Arbeitsbereich Symbolisches Rechnen
Prof. Dr. Wolfgang Küchlin
Universität Tübingen

20. Dezember 2011

Motivation

Bisher

- Formeln in AL oder FOL beschreiben einen Zustand
- Was wir nicht modellieren können:
  - Zeitlichen Ablauf
  - Konzepte wie *Möglichkeit* oder *Notwendigkeit*
- Modellierung von komplexen Systemen (z.B. Hardware, Software) erfordert jedoch solche Modellierungsmethoden
  - Verschränkung von Threads
  - Ampelschaltung
  - …

$\Downarrow$

Modallogik & Temporallogik

Motivation

Modallogik

Syntax

Semantik

Normalformen

Tableau Verfahren

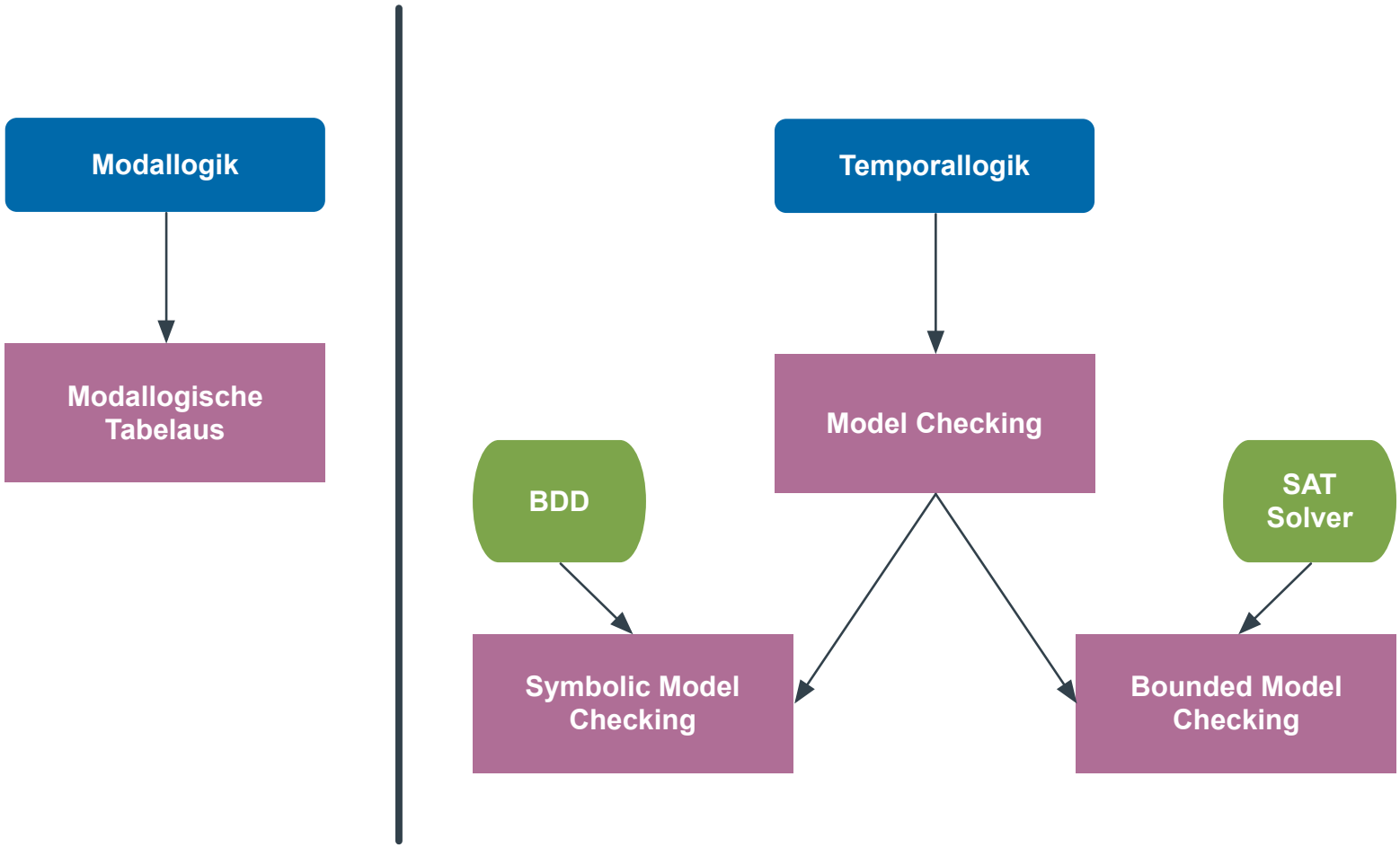
Temporallogiken

Linear Time Logic

Branching Time Logic

Literatur

Der Plan für die nächsten drei Wochen



Motivation

Modallogik

- Syntax
- Semantik
- Normalformen
- Tableau Verfahren

Temporallogiken

- Linear Time Logic
- Branching Time Logic

Literatur

Modallogik

- **Aussagenlogik:** Eine Aussage ist entweder wahr oder falsch
- **Modallogik:** Eine Aussage ist *manchmal* wahr und *manchmal* falsch, abhängig von äußeren Umständen

Einführung zweier neuer logischen Operatoren $\Diamond$ und $\Box$.

Betrachtungsweisen

1. **Philosophische Betrachtungen**
   - $\Box A$: Die Aussage A gilt notwendigerweise
   - $\Diamond A$: Die Aussage A gilt möglicherweise
2. **Zeitliche Abläufe**
   - $\Box A$: Die Aussage A gilt immer
   - $\Diamond A$: Die Aussage A gilt irgendwann einmal
3. **Künstliche Intelligenz**
   - $\Box A$: Ein Agent/Prozessor weiß A
   - $\Diamond A$: Ein Agent/Prozessor glaubt, dass A gilt

Motivation
Modallogik
Syntax
Semantik
Normalformen
Tableau Verfahren
Temporallogiken
Linear Time Logic
Branching Time Logic
Literatur

Syntax

Grammatik

EBNF-Grammatik für modallogische Formeln

Formel	=	$\top \mid \bot$	Konstanten
	$\mid$	*Variable*	$\in \mathcal{V}$
	$\mid$	$\neg$ Formel	Negation
	$\mid$	Formel $\wedge$ Formel	Konjunktion
	$\mid$	Formel $\vee$ Formel	Disjunktion
	$\mid$	$\Box$ Formel	Notwendigkeit
	$\mid$	$\Diamond$ Formel	Möglichkeit
	$\mid$	(Formel)	Klammerung

- Man kann Modallogik auch über FOL betrachten, wir wählen jedoch die aussagenlogische Variante
- $\rightarrow$, $\leftrightarrow$ und $\oplus$ sind wie in der Aussagenlogik nur abkürzende Schreibweisen
- $\Box$ und $\Diamond$ binden stärker als die anderen logischen Operatoren (wie Quantoren)

Syntax

Beispiele

Beispiel (Syntax)

Unterhaltung zwischen Professoren

1. “Wenn Freiburg irgendwann Meister wird, fress ich einen Besen.”
2. “Irgendwann wird deine Aussage eintreten”
3. “Dass du Recht hast, ist doch das Gleiche, wie dass Freiburg von nun an immer Meister wird”

Aussagen

- A = Freiburg wird Meister
- B = Professor 1) frisst einen Besen

Formalisierung

1. $\Diamond A \to B$
2. $\Diamond(\Diamond A \to B)$
3. $\Diamond(\Diamond A \to B) \leftrightarrow \Box A$

Semantik

Kripke-Strukturen

- Wir können den Wahrheitsgehalt einer Formel nicht nur an der Formel ablesen
- Frage: Wann gelten die äußeren Bedingungen? (z.B. wann wird Freiburg Meister?)

$$\Downarrow$$

Menge von Zuständen oder möglichen Welten, an jedem Zustand gibt es eine Menge von gültigen Zuständen oder erreichbaren Welten.

Definition (Kripke-Struktur)

Eine Kripke Struktur ist ein Tripel $\mathcal{M} = (S, \longrightarrow, L)$ mit

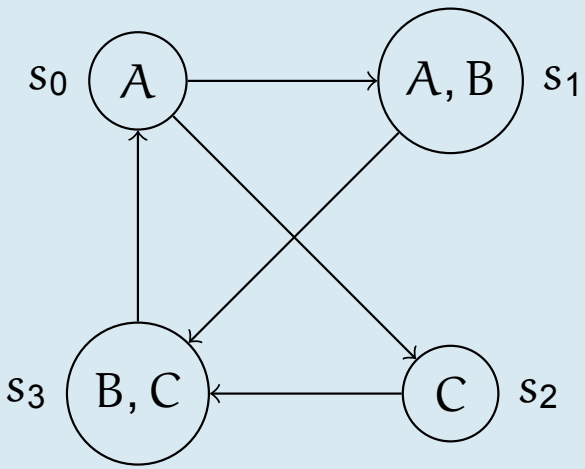
- S: Nicht-leere Menge von Zuständen (möglichen Welten)
- $\longrightarrow \subseteq S \times S$: Zustandsübergangsrelation
- L: Beschriftung jedes Zustandes mit dort gültigen aussagenlogischen Formeln

Semantik

Kripke-Strukturen—Beispiel

Beispiel (Kripke Struktur)

- $S = \{s_0, s_1, s_2, s_3\}$
- $\longrightarrow = \{(s_0, s_1), (s_0, s_2), (s_1, s_3), (s_2, s_3), (s_3, s_0)\}$
- $L : L(s_0) = \{A\}, L(s_1) = \{A, B\}, L(s_2) = \{C\}, L(s_3) = \{B, C\}$



- Anstelle von $(s, t) \in \longrightarrow$ schreiben wir auch $s \longrightarrow t$

Semantik

Auswertung einer Formel

Auswertung einer Formel bzgl. eines bestimmten Zustandes s

Algorithmus: $\texttt{holds}(\mathcal{M}, s, \psi)$

Eingabe: Kripke Struktur $\mathcal{M} = (S, \longrightarrow, L)$, Zustand $s \in S$, Formel ψ
Ausgabe: Evaluation von ψ im Zustand s

$\texttt{holds}(\mathcal{M}, s, \psi) = \psi$ *match*

$\top \rightsquigarrow$ true

| $\bot \rightsquigarrow$ false

| $v \in \mathcal{V} \rightsquigarrow$ *if* $v \in L(s)$ *then* true *else* false

| $\neg\varphi \rightsquigarrow$ *if* $\texttt{holds}(\mathcal{M}, s, \varphi)$ *then* false *else* true

| $\varphi_1 \wedge \varphi_2 \rightsquigarrow$ *if* $\texttt{holds}(\mathcal{M}, s, \varphi_1)$ *and* $\texttt{holds}(\mathcal{M}, s, \varphi_2)$ *then* true *else* false

| $\varphi_1 \vee \varphi_2 \rightsquigarrow \texttt{holds}(\mathcal{M}, s, \neg(\neg\varphi_1 \wedge \neg\varphi_2))$

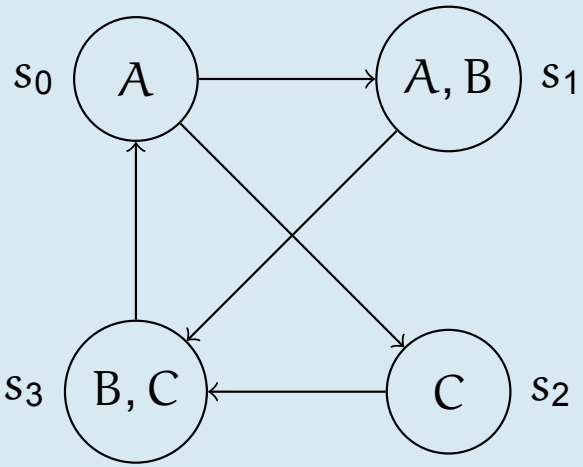
| $\Box\varphi \rightsquigarrow$ *for all* $t \in S$: $(s \longrightarrow t) \rightarrow \texttt{holds}(\mathcal{M}, t, \varphi)$

| $\Diamond\varphi \rightsquigarrow$ *exists* $t \in S$: $(s \longrightarrow t) \wedge \texttt{holds}(\mathcal{M}, t, \varphi)$

Semantik

Auswertung einer Formel—Beispiel

Beispiel (Kripke Struktur)



- $\texttt{holds}(\mathcal{M}, s_0, \mathrm{A}) = \texttt{true}$
- $\texttt{holds}(\mathcal{M}, s_0, \mathrm{B}) = \texttt{false}$
- $\texttt{holds}(\mathcal{M}, s_1, \mathrm{A} \wedge \mathrm{B}) = \texttt{true}$
- $\texttt{holds}(\mathcal{M}, s_0, \Box(\mathrm{A} \vee \mathrm{C})) = \texttt{true}$
- $\texttt{holds}(\mathcal{M}, s_0, \Diamond(\mathrm{A} \wedge \mathrm{C})) = \texttt{false}$

Semantik

- Kripke-Struktur $\mathcal{M} = (S, \longrightarrow, \mathrm{L})$
- Zustand $s \in S$

Definition (Gültigkeit)

Ist $\texttt{holds}(\mathcal{M}, s, \varphi) = \texttt{true}$, so sagen wir, dass die Formel φ in der Struktur $\mathcal{M}$ im Zustand s gilt: $s \vDash_{\mathcal{M}} \varphi$. Ist $\mathcal{M}$ klar, so schreiben wir auch $s \vDash \varphi$.

Definition (Modell)

Gilt für **alle** $s \in S$ $\texttt{holds}(\mathcal{M}, s, \varphi)$, so sagen wir, dass die Formel φ in $\mathcal{M}$ gültig ist. $\mathcal{M}$ ist dann ein Modell für φ: $\mathcal{M} \vDash \varphi$.

Definition (Rahmen)

Für eine Menge von Zuständen S und eine Übergangsfunktion $\longrightarrow \subseteq S \times S$ definieren wir das Paar $\mathcal{R} = (S, \longrightarrow)$ als modallogischen Rahmen.

Ist eine Formel φ in allen auf $\mathcal{R}$ basierenden Strukturen $\mathcal{M} = (S, \longrightarrow, \mathrm{L})$ gültig, so heißt φ gültig in $\mathcal{R}$: $\mathcal{R} \vDash \varphi$.

Semantik

Beispiel

Wir betrachten erneut die Formel $\varphi = \Diamond(\Diamond A \to B) \leftrightarrow \Box A$

- Rahmen $\mathcal{R} = (\mathbb{N}, <)$ (Jahre)
- $A \in L(s)$ falls $s > 2$ (Ab Jahr 2 wird Freiburg immer Meister)
- $B \in L(s)$ falls s ungerade (Jedes ungerade Jahr frisst Prof. 1 einen Besen)

Beispiel (Rahmen und Gültigkeit)

1. Wahrheitswert von φ im Jahr 3
   - $\texttt{holds}(\mathcal{M}, 3, \Box A) = \texttt{true}$
   - $\texttt{holds}(\mathcal{M}, 3, \Diamond(\Diamond A \to B)) = \texttt{true}$

   und damit $\texttt{holds}(\mathcal{M}, 3, \varphi) = \texttt{true}$
2. Wahrheitswert von φ im Jahr 0
   - $\texttt{holds}(\mathcal{M}, 0, \Box A) = \texttt{false}$
   - $\texttt{holds}(\mathcal{M}, 0, \Diamond(\Diamond A \to B)) = \texttt{true}$

   und damit $\texttt{holds}(\mathcal{M}, 0, \varphi) = \texttt{false}$

Damit ist $\mathcal{M}$ **kein Modell** für φ und daher ist auch φ **nicht gültig in** $\mathcal{R}$.

Modaler Rang

- Bestimmung des Wahrheitswert einer Formel in einem Zustand s kann Berechnung an anderen Zuständen benötigen
- **Frage:** Wie viele Zustände braucht man?

Definition (Modaler Rang)

Der modale Rang $\mathrm{MR}(\varphi)$ einer Formel φ wird wie folgt bestimmt:

1. Für $\varphi \in \mathcal{V}$ ist $\mathrm{MR}(\varphi) = 0$
2. Für $\varphi = \neg\psi_1$ oder $\varphi = \psi_1 \wedge \psi_2$ oder $\varphi = \psi_1 \vee \psi_2$ ist $\mathrm{MR}(\varphi) = \max(\mathrm{MR}(\psi_1), \mathrm{MR}(\psi_2))$
3. Für $\varphi = \Box\psi$ oder $\varphi = \Diamond\psi$ ist $\mathrm{MR}(\varphi) = \mathrm{MR}(\psi) + 1$

Definition (Erreichbarkeit)

Für einen Rahmen $\mathcal{R}$ und $s \in S$ heißt ein Zustand $t \in S$ **von s in n Schritten erreichbar**, wenn es $t_0, \ldots, t_n$ gibt mit $s = t_0 \longrightarrow t_1 \longrightarrow \ldots \longrightarrow t_n = t$

Definition (n-te Iteration)

Für einen Rahmen $\mathcal{R}$ und $n \geq 0$ ist die n-te Iteration von $\longrightarrow$:

$$\longrightarrow^{(n)} = \{(s,t) \in S \times S \mid t \text{ ist von } s \text{ in } \leq n \text{ Schritten erreichbar}\}$$

Das Koinzidenzlemma

Beispiel (Modaler Rang)

- $\mathrm{MR}(\Diamond(\Diamond A \to B) \leftrightarrow \Box A) = 2$

Lemma (Koinzidenzlemma)

- φ *modallogische Formel*
- $m = \mathrm{MR}(\varphi)$
- $\mathcal{R} = (S, \longrightarrow)$ *Rahmen und* $s \in S$
- *Zwei Strukturen* $\mathcal{M}_1 = (S, \longrightarrow, L_1)$ *und* $\mathcal{M}_2 = (S, \longrightarrow, L_2)$ *mit*
  - $L_1(t) = L_2(t)$ *bzgl. der Symbole, die in* φ *vorkommen für alle* $t \in S$ *mit* $(s, t) \in \longrightarrow^{(m)}$

Dann gilt $\mathcal{M}_1 \vDash \varphi$ *gdw.* $\mathcal{M}_2 \vDash \varphi$

D.h. für $\mathtt{holds}(\mathcal{M}, s, \varphi)$ spielen nur die Beschriftungen $L(t)$ eine Rolle, die in φ vorkommen und t von s in höchstens $\mathrm{MR}(\varphi)$ Schritten erreichbar ist.

Weitere semantische Begriffe

- φ ist **allgemeingültig** oder eine **Tautologie**, wenn φ in jedem zu φ passenden Rahmen gilt.
- φ heißt eine (semantische) **Folgerung** der Menge Γ, falls φ in jeder zu φ und Γ passenden Struktur $\mathcal{M}$ gilt, in der alle Formeln aus Γ gültig sind.
- Zwei Formeln φ und ψ heißen **semantisch äquivalent**, wenn eine zu φ und ψ passende Struktur $\mathcal{M}$ genau dann ein Modell für φ ist, wenn sie auch ein Modell für ψ ist. Notation $\varphi \equiv \psi$.
- φ ist eine **Rahmenfolgerung** einer Menge Γ, wenn für jeden Rahmen $\mathcal{R}$ gilt, dass aus $\mathcal{R} \vDash \gamma$ für alle $\gamma \in \Gamma$ schon $\mathcal{R} \vDash \varphi$ folgt. Notation: $\Gamma \vDash \varphi$.
- φ heißt **rahmenäquivalent** zu ψ, wenn für jeden Rahmen $\mathcal{R}$ gilt $\mathcal{R} \vDash \varphi$ gdw. $\mathcal{R} \vDash \psi$.

Modallogische Äquivalenzen

a)	$\neg \Box \varphi \equiv \Diamond \neg \varphi$	1. Dualitätsgesetz
b)	$\neg \Diamond \varphi \equiv \Box \neg \varphi$	2. Dualitätsgesetz
c)	Aus $\Box(\varphi \rightarrow \psi)$ folgt $\Box\varphi \rightarrow \Box\psi$	1. Distributionsregel
d)	$\Diamond(\varphi \rightarrow \psi) \equiv \Box\varphi \rightarrow \Diamond\psi$	2. Distributionsregel
e)	Aus $\Box(\varphi \rightarrow \psi)$ folgt $\Diamond\varphi \rightarrow \Diamond\psi$	3. Distributionsregel
f)	$\Box(\varphi \wedge \psi) \equiv \Box\varphi \wedge \Box\psi$	1. Distributivgesetz
g)	$\Diamond(\varphi \vee \psi) \equiv \Diamond\varphi \vee \Diamond\psi$	2. Distributivgesetz
h)	Ist φ eine Tautologie, so ist auch $\Box\varphi$ eine Tautologie	Necessitationsregel

Quasi-disjunktive Normalform

Definition (QDNF)

Eine modallogische Formel φ ist in quasi-disjunktiver Normalform (QDNF), wenn sie von der Form

$$\varphi = (L_{11} \wedge \cdots \wedge L_{1n_1}) \vee \cdots \vee (L_{r1} \wedge \cdots \wedge L_{rn_r})$$

ist mit Literalen L_{ij}, die entweder aussagenlogische Literale sind oder Formeln der Form $\Box\psi$ oder $\Diamond\psi$ mit ψ einer modallogischen Formel.

Vorgehen

1. Bringe die Formel in NNF *(AL NNF + Äquivalenzen a) und b))*.
2. Minimiere den Scope der Operatoren $\Box$ und $\Diamond$ *(Regeln f) und g))*.
3. Benutze den üblichen DNF Algorithmus, jedoch mit der erweiterten Literaldefinition.

QDNF—Beispiel

$$\Diamond(\Diamond A \to B) \leftrightarrow \Box A$$

Beispiel (QDNF)

1. Auflösen der Booleschen Konnektive

$$(\Diamond(\neg \Diamond A \vee B) \wedge \Box A) \vee (\neg \Diamond (\neg \Diamond A \vee B) \wedge \neg \Box A)$$

2. NNF

$$(\Diamond(\Box \neg A \vee B) \wedge \Box A) \vee (\Box(\Diamond A \wedge \neg B) \wedge \Diamond \neg A)$$

3. Minimieren der Scopes

$$((\Diamond \Box \neg A \vee \Diamond B) \wedge \Box A) \vee ((\Box \Diamond A \wedge \Box \neg B) \wedge \Diamond \neg A)$$

4. QDNF

$$(\Diamond \Box \neg A \wedge \Box A) \vee (\Diamond B \wedge \Box A) \vee (\Box \Diamond A \wedge \Box \neg B \wedge \Diamond \neg A)$$

Wiederholung: AL Tableaus

Konstruktion eines Aussagenlogischen Tableaus

Algorithmus: `ALTableau`(φ)

Eingabe: Aussagenlogische Formel φ in NNF
Ausgabe: Binärer Baum (AL Tableau)

---

1. Setze die Wurzel des Baumes auf φ
2. **Wähle** einen noch nicht markierten Knoten G des Baumes, der kein Literal ist. **Markiere** den Knoten und wende eine der folgenden beiden Tableau Regeln an:
   1. Ist G von der Form $\varphi \vee \psi$ so hänge am Ende eines jeden bei G beginnenden Pfads die beiden Zweige $\{\varphi\}$ und $\{\psi\}$ an
   2. Ist G von der Form $\varphi \wedge \psi$ so hänge am Ende eines jeden bei G beginnenden Pfads den Zweig $\{\varphi, \psi\}$ an
3. Gibt es keinen unmarkierten Knoten mehr, so ist der Baum ein **vollständiges Tableau**, das zurückgegeben wird. Ansonsten gehe zu Schritt 2 zurück.

Wiederholung: AL Tableaus

Definitionen

- Ein Pfad im Tableau heißt **geschlossen**, wenn er sowohl das Literal A als auch $\neg A$ als Knoten enthält.
- Ein Tableau heißt **geschlossen**, wenn all seine Pfade von der Wurzel bis zu Blättern geschlossen sind.
- Ein Tableau heißt **erfüllbar**, wenn es einen Pfad von der Wurzel zu einem Blatt gibt, so dass für die Formeln an den Knoten des Pfades ein gemeinsames Modell existiert.
- Ein Pfad heißt **vollständig**, wenn alle Formeln an seinen Kanten abgearbeitet sind, d.h. wenn sie markiert sind oder Literale sind.
- Ein Tableau heißt **vollständig**, wenn jeder seiner Pfade vollständig oder geschlossen ist.

Beispiel (An der Tafel . . .)

$$\varphi = (A \to B) \wedge (B \to C) \wedge \neg(A \to C)$$

Modallogische Tableaus

Erweiterung der AL Tableaus

- Zustand muss mit betrachtet werden (an Knoten annotiert werden)
- Regelerweiterung um $\Box$ und $\Diamond$

Umsetzung

- An den Knoten stehen keine Formeln, sondern Ausdrücke der Form $s \vDash \varphi$ mit $s \in S$ und φ eine ML Formel

Modallogische Tableaus

Konstruktion

Algorithmus: `MLTableau`$(\varphi, \mathcal{R}, s)$

Eingabe: Modallogische Formel φ in NNF, ein dazu passender Rahmen $\mathcal{R} = (S, \longrightarrow)$ und ein Zustand $s \in S$
Ausgabe: Binärer Baum (ML Tableau)

---

1. Setze die Wurzel des Baumes auf $s \vDash \varphi$
2. **Wähle** einen noch nicht markierten Knoten der Form $s' \vDash \psi$ wobei ψ kein AL Literal ist.
3. **Markiere** den Knoten $s' \vDash \psi$ und wende eine der folgenden Tableau Regeln an.
4. Gibt es keinen solchen Knoten mehr, so ist das Tableau fertig und wird ausgegeben.
5. Gehe zurück zu Schritt 2.

Modallogische Tableaus

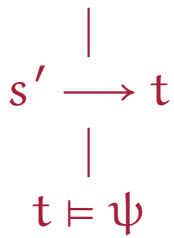
Tableauregeln

Aussagenlogischer Knoten

Ist der Knoten von der Form $s \vDash \psi_1 \vee \psi_2$ oder $s \vDash \psi_1 \wedge \psi_2$, so wende die entsprechende Regel aus dem AL Tableau an.

$\Diamond$ Knoten

Ist der Knoten von der Form $s' \vDash \Diamond\psi$, so hänge am Ende eines jeden bei diesem Knoten beginnende Pfads den Zweig



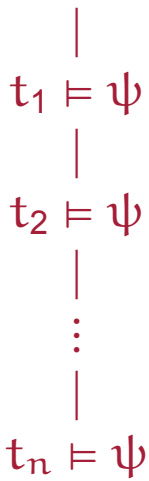
an, wobei t ein neuer Name für einen Zustand ist. Für jeden bereits markierten Knoten der Form $s' \vDash \Box\gamma$ füge am Ende eines jeden bei ihm beginnenden und durch $s' \vDash \Diamond\psi$ laufenden Pfads den Zweig $\{t \vDash \gamma\}$ an.

Modallogische Tableaus

Tableauregeln

□ Knoten

Ist der Knoten von der Form $s' \vDash \Box\psi$, so hänge am Ende eines jeden durch diesen Knoten laufenden Pfads den Zweig



an, wobei $t_1, \ldots, t_n$ die in diesem Pfad in Knoten der Form $s' \longrightarrow t_i$ auftretenden Zustände sind.

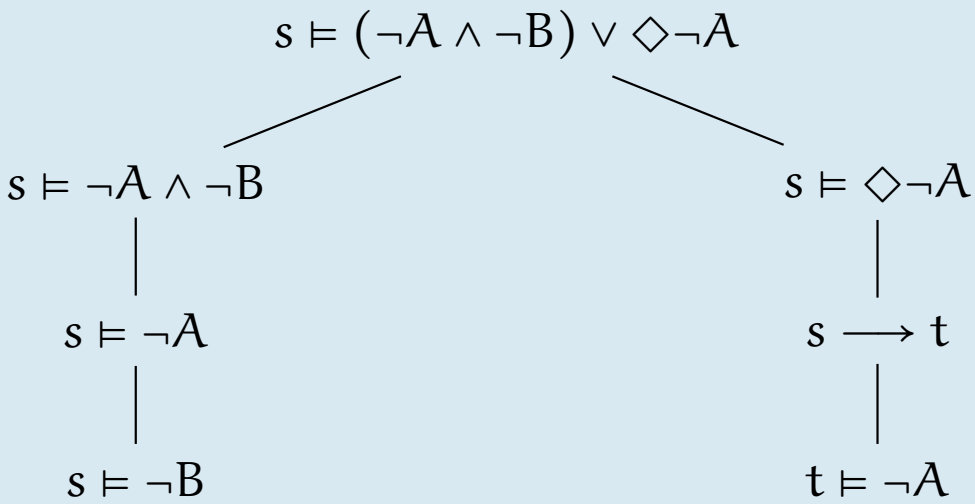
Modallogische Tableaus

Beispiel

Beispiel (Modallogisches Tableau)

$$\varphi = (A \vee B) \rightarrow \neg \Box A$$

Tableau



- Tableau ist vollständig
- Keine geschlossenen Pfade
- Formel ist erfüllbar. (Z.B. in allen Strukturen mit $L(s) = \{\}$)

Temporallogiken

Modallogik

- Beliebige Rahmen
- Bei den Operatoren $\Box$ und $\Diamond$ wird immer nur der nächste Schritt betrachtet

Temporallogiken

- Einschränkung des Rahmens auf zeitliche Abfolgen
- Damit beziehen sich Operatoren implizit auf alle möglichen zukünftigen Zustände

Definition (Zeitlogischer Rahmen)

Ein Rahmen $\mathcal{R} = (S, \longrightarrow)$ heißt zeitlogischer Rahmen, wenn für $\longrightarrow$ gilt:

1. Für alle $s \in S$ gilt $s \not\longrightarrow s$ *(irreflixiv)*
2. Für alle $s, t, u \in S$ gilt $s \longrightarrow t$ und $t \longrightarrow u$ impliziert $s \longrightarrow u$ *(transitiv)*

d.h. $\longrightarrow$ definiert eine strikte partielle Ordnung.

Linear Time Logic

- Wir betrachten einzelne Zeitpfade auf denen Aussagen irgendwann wahr oder falsch werden
- Es wird immer nur ein Pfad gleichzeitig betrachtet

LTL Operatoren

- $\mathbf{F}\,\varphi$ (Finally): φ gilt in einem zukünftigen Zustand auf dem Pfad
- $\mathbf{G}\,\varphi$ (Globally): φ gilt in allen Zuständen auf dem Pfad
- $\mathbf{X}\,\varphi$ (Next): φ gilt im nächsten Zustand auf dem Pfad
- $\varphi\,\mathbf{U}\,\psi$ (Until): φ gilt, bis ein Zustand erreicht wird, an dem ψ gilt
- $\varphi\,\mathbf{W}\,\psi$ (Weak Until): Wie Until, jedoch muss ψ nie eintreten
- $\varphi\,\mathbf{R}\,\psi$ (Release): ψ gilt bis zu einschließlich dem Zustand, an dem φ erfüllt ist

Syntax

EBNF-Grammatik für LTL Formeln

Formel	=	$\top \mid \bot$	Konstanten
	|	*Variable*	$\in \mathcal{V}$
	|	$\neg$ Formel	Negation
	|	Formel $\wedge$ Formel	Konjunktion
	|	Formel $\vee$ Formel	Disjunktion
	|	Formel $\rightarrow$ Formel	Implikation
	|	$\mathbf{X}$ Formel	Next
	|	$\mathbf{F}$ Formel	Finally
	|	$\mathbf{G}$ Formel	Globally
	|	Formel $\mathbf{U}$ Formel	Until
	|	Formel $\mathbf{W}$ Formel	Weak Until
	|	Formel $\mathbf{R}$ Formel	Release
	|	(Formel)	Klammerung

- Unäre Operatoren binden am stärksten, gefolgt von $\mathbf{R}, \mathbf{U}, \mathbf{W}$, gefolgt von $\wedge, \vee, \rightarrow$.

Semantik

- Wir betrachten wieder Kripke-Strukturen (Modelle)

Technische Einschränkung für $\longrightarrow$

- Zu jedem Zustand $s \in S$ gibt es mindestens einen Übergang $s \longrightarrow s'$
- D.h. es gibt keine Deadlocks

i Keine echte Einschränkung

Dies schränkt jedoch die zu modellierenden Systeme nicht ein. Erfüllt ein Zustand s diese Eigenschaft nicht:

- füge einen neuen Zustand s_d (Deadlock) hinzu
- füge $s \longrightarrow s_d$ und $s_d \longrightarrow d_d$ hinzu

Definition (Pfad)

Ein Pfad in einem Modell $\mathcal{M} = (S, \longrightarrow, L)$ ist eine unendliche Folge von Zuständen $s_0, s_1, s_2, \ldots$ in S, so dass für jedes $i \geq 0$ $s_i \longrightarrow s_{i+1}$.

Für einen Pfad $\pi = s_0 \longrightarrow s_1 \longrightarrow \ldots$ schreiben wir π^i für den Teilpfad, der bei s_i beginnt.

Auswertung einer LTL Formel

Algorithmus: $\texttt{holds}(\mathcal{M}, \pi, \psi)$

Eingabe: Modell $\mathcal{M} = (S, \longrightarrow, \mathrm{L})$, Pfad $\pi = s_0 \longrightarrow \ldots$, LTL Formel ψ
Ausgabe: Evaluation von ψ auf dem Pfad π

$\texttt{holds}(\mathcal{M}, \pi, \psi) = \psi$ *match*

$\top \rightsquigarrow$ `true`

$\mid \bot \rightsquigarrow$ `false`

$\mid \nu \in \mathcal{V} \rightsquigarrow$ *if* $\nu \in \mathrm{L}(s_0)$ *then* `true` *else* `false`

$\mid \neg\varphi \rightsquigarrow$ *if* $\texttt{holds}(\mathcal{M}, \pi, \varphi)$ *then* `false` *else* `true`

$\mid \varphi_1 \wedge \varphi_2 \rightsquigarrow$ *if* $\texttt{holds}(\mathcal{M}, \pi, \varphi_1)$ *and* $\texttt{holds}(\mathcal{M}, \pi, \varphi_2)$ *then* `true` *else* `false`

$\mid \varphi_1 \vee \varphi_2 \rightsquigarrow \texttt{holds}(\mathcal{M}, \pi, \neg(\neg\varphi_1 \wedge \neg\varphi_2))$

$\mid \varphi_1 \rightarrow \varphi_2 \rightsquigarrow \texttt{holds}(\mathcal{M}, \pi, \neg\varphi_1 \vee \varphi_2))$

$\mid \ldots$

Auswertung einer LTL Formel

Algorithmus: $\texttt{holds}(\mathcal{M}, \pi, \psi)$

$$
\begin{aligned}
&| \ \ldots \\
&| \ \mathbf{X}\,\varphi \ \rightsquigarrow \texttt{holds}(\mathcal{M}, \pi^1, \varphi) \\
&| \ \mathbf{G}\,\varphi \ \rightsquigarrow \textit{for all } i \geq 0\text{: } \texttt{holds}(\mathcal{M}, \pi^i, \varphi) \\
&| \ \mathbf{F}\,\varphi \ \rightsquigarrow \textit{exists } i \geq 0\text{: } \texttt{holds}(\mathcal{M}, \pi^i, \varphi) \\
&| \ \varphi_1\,\mathbf{U}\,\varphi_2 \ \rightsquigarrow \textit{exists } i \geq 0\text{: } \texttt{holds}(\mathcal{M}, \pi^i, \varphi_2) \\
&\qquad\qquad \textit{and for all } 0 \leq j < i\text{: } \texttt{holds}(\mathcal{M}, \pi^j, \varphi_1) \\
&| \ \varphi_1\,\mathbf{W}\,\varphi_2 \ \rightsquigarrow \texttt{holds}(\mathcal{M}, \varphi_1\,\mathbf{U}\,\varphi_2) \textit{ or} \\
&\qquad\qquad \textit{for all } k \geq 0\text{: } \texttt{holds}(\mathcal{M}, \pi^k, \varphi_1) \\
&| \ \varphi_1\,\mathbf{R}\,\varphi_2 \ \rightsquigarrow [\textit{exists } i \geq 0\text{: } \texttt{holds}(\mathcal{M}, \pi^i, \varphi_1) \textit{ and for all} \\
&\qquad\qquad 0 \leq j \leq i\text{: } \texttt{holds}(\mathcal{M}, \pi^j, \varphi_2)] \textit{ or} \\
&\qquad\qquad [\textit{for all } k \geq 0\text{: } \texttt{holds}(\mathcal{M}, \pi^k, \varphi_2)]
\end{aligned}
$$

- Gilt $\texttt{holds}(\mathcal{M}, \pi, \varphi)$, so schreiben wir auch $\pi \vDash_{\mathcal{M}} \varphi$ oder $\pi \vDash \varphi$ wenn $\mathcal{M}$ aus dem Kontext klar ist.
- Gilt $\pi \vDash \varphi$ für jeden möglichen Ausführungspfad π beginnend bei einem Zustand s, so schreiben wir auch $s \vDash \varphi$.

Temporaloperatoren

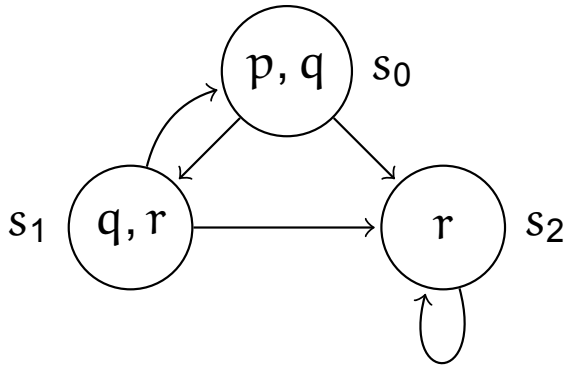
Beispiele

Beispiel (Temporaloperatoren)

	0	1	2	3	4
	○	○	○	○	○
$\mathbf{G}\,p$	p	p	p	p	p
$\mathbf{F}\,p$	-	-	-	p	-
$p\,\mathbf{U}\,q$	p	p	p	q	-
$p\,\mathbf{W}\,q$	p	p	p	q	-
	p	p	p	p	p
$p\,\mathbf{R}\,q$	q	q	q, p	-	-
	q	q	q	q	q

Auswertung einer LTL Formel

Beispiel



Beispiel (Auswertungen)

- $s_0 \vDash p \wedge q$
- $s_0 \vDash \neg r$
- $s_0 \vDash \mathbf{X}\, r$
- $s_0 \nvDash \mathbf{X}(q \wedge r)$
- $s_0 \vDash \mathbf{G} \neg (p \wedge r)$
- $s_2 \vDash \mathbf{G}\, r$
- Für jeden Zustand s gilt:
  $s \vDash \mathbf{F}(\neg q \wedge r) \rightarrow \mathbf{F}\,\mathbf{G}\, r$
- $s_0 \vDash \mathbf{G}\,\mathbf{F}\, p \rightarrow \mathbf{G}\,\mathbf{F}\, r$

Was geht mit LTL? Was geht nicht?

👍 Was geht mit LTL?

- Man kann keinen Zustand erreichen, in dem `started` gilt aber nicht `ready`: $\mathbf{G}\,\neg(\texttt{started} \wedge \neg\texttt{ready})$
- Wenn in einem beliebigen Zustand ein Request eintrifft, wird er auf jeden Fall in einem zukünftigen Zustand bestätigt: $\mathbf{G}(\texttt{req} \rightarrow \mathbf{F}\,\texttt{ack})$
- Wenn ein Prozess unendlich oft aktiviert wird, läuft er unendlich oft: $\mathbf{G}\,\mathbf{F}\,\texttt{enabled} \rightarrow \mathbf{G}\,\mathbf{F}\,\texttt{running}$

👎 Was geht nicht mit LTL?

- Von jedem Zustand aus ist es möglich in einen `restart` Zustand zu gelangen (d.h. *es gibt einen Pfad* von jedem Zustand aus zu einem Zustand, an dem `restart` gilt)
- Der Lift kann untätig mit geschlossenen Türen im dritten Stock stehen bleiben (d.h. vom Zustand in dem der Lift im dritten Stock ist, gibt es einen Pfad, in dem er dort bleibt)

Wichtige Äquivalenzen

Definition (Äquivalenz)

Zwei LTL Formeln φ und ψ sind semantisch äquivalent $\varphi \equiv \psi$, wenn für alle Modelle $\mathcal{M}$ und alle Pfade π in $\mathcal{M}$ gilt: $\pi \vDash \varphi$ gdw. $\pi \vDash \varphi$.

a)	$\neg \mathbf{G}\, \varphi \equiv \mathbf{F} \neg \varphi$	Dualität von $\mathbf{G}$ und $\mathbf{F}$
b)	$\neg \mathbf{F}\, \varphi \equiv \mathbf{G} \neg \varphi$	Dualität von $\mathbf{G}$ und $\mathbf{F}$
c)	$\neg \mathbf{X}\, \varphi \equiv \mathbf{X} \neg \varphi$	Dualität von $\mathbf{X}$
d)	$\neg(\varphi_1 \,\mathbf{U}\, \varphi_2) \equiv \neg\varphi_1 \,\mathbf{R}\, \neg\varphi_2$	Dualität von $\mathbf{R}$ und $\mathbf{U}$
e)	$\neg(\varphi_1 \,\mathbf{R}\, \varphi_2) \equiv \neg\varphi_1 \,\mathbf{U}\, \neg\varphi_2$	Dualität von $\mathbf{R}$ und $\mathbf{U}$
f)	$\mathbf{G}(\varphi_1 \wedge \psi) \equiv \mathbf{G}\, \varphi_1 \wedge \mathbf{G}\, \varphi_2$	1. Distributivgesetz
g)	$\mathbf{F}(\varphi_1 \vee \psi) \equiv \mathbf{F}\, \varphi_1 \vee \mathbf{F}\, \varphi_2$	2. Distributivgesetz

Branching Time Logic

LTL

- LTL Formeln werden auf Pfaden ausgewertet
- Eine Formel gilt an einem Zustand, wenn sie an allen möglichen Pfaden von diesem Zustand aus gilt
- d.h. implizite Allquantifizierung der Pfade
- Eigenschaften, die die Existenz eines Pfades fordern, können nicht in LTL ausgedrückt werden

Branching Time Logic

- Quantifizierung über Pfade ist erlaubt
- 2 bekannte Formen:
  - **CTL:** Jedem Temporaloperator geht genau ein Pfadquantor voraus
  - **CTL*:** Quantifizierung beliebiger Formeln

$$\mathbf{CTL} \subset \mathbf{CTL^*},\ \mathbf{LTL} \subset \mathbf{CTL^*},\ \mathbf{CTL} \not\subseteq \mathbf{LTL},\ \mathbf{LTL} \not\subseteq \mathbf{CTL}$$

CTL—Syntax

EBNF-Grammatik für CTL Formeln

Formel	=	$\top \mid \bot$	Konstanten
	$\mid$	*Variable*	$\in \mathcal{V}$
	$\mid$	$\neg$ Formel	Negation
	$\mid$	Formel $\wedge$ Formel	Konjunktion
	$\mid$	Formel $\vee$ Formel	Disjunktion
	$\mid$	Formel $\rightarrow$ Formel	Implikation
	$\mid$	$\mathbf{AX}$ Formel $\mid$ $\mathbf{EX}$ Formel	Next
	$\mid$	$\mathbf{AF}$ Formel $\mid$ $\mathbf{EF}$ Formel	Finally
	$\mid$	$\mathbf{AG}$ Formel $\mid$ $\mathbf{EG}$ Formel	Globally
	$\mid$	$\mathbf{A}$ [Formel $\mathbf{U}$ Formel]	Until
	$\mid$	$\mathbf{E}$ [Formel $\mathbf{U}$ Formel]	Until
	$\mid$	(Formel)	Klammerung

- $\mathbf{A}$: Auf allen Pfaden
- $\mathbf{E}$: Es existiert ein Pfad

Auswertung einer CTL Formel

Algorithmus: $\texttt{holds}(\mathcal{M}, s, \psi)$

Eingabe: Modell $\mathcal{M} = (S, \longrightarrow, \mathrm{L})$, Zustand $s \in S$, CTL Formel ψ
Ausgabe: Evaluation von ψ am Zustand s

$\texttt{holds}(\mathcal{M}, s, \psi) = \psi$ *match*

$\top \rightsquigarrow$ true

| $\bot \rightsquigarrow$ false

| $v \in \mathcal{V} \rightsquigarrow$ *if* $v \in \mathrm{L}(s)$ *then* true *else* false

| $\neg\varphi \rightsquigarrow$ *if* $\texttt{holds}(\mathcal{M}, s, \varphi)$ *then* false *else* true

| $\varphi_1 \wedge \varphi_2 \rightsquigarrow$ *if* $\texttt{holds}(\mathcal{M}, s, \varphi_1)$ *and* $\texttt{holds}(\mathcal{M}, s, \varphi_2)$ *then* true *else* false

| $\varphi_1 \vee \varphi_2 \rightsquigarrow \texttt{holds}(\mathcal{M}, s, \neg(\neg\varphi_1 \wedge \neg\varphi_2))$

| $\varphi_1 \to \varphi_2 \rightsquigarrow \texttt{holds}(\mathcal{M}, s, \neg\varphi_1 \vee \varphi_2))$

| $\mathbf{AX}\,\varphi \rightsquigarrow$ *for all* $s' \in S$: $s \longrightarrow s' \to \texttt{holds}(\mathcal{M}, s', \varphi)$

| $\mathbf{EX}\,\varphi \rightsquigarrow$ *exists* $s' \in S$: $s \longrightarrow s' \wedge \texttt{holds}(\mathcal{M}, s', \varphi)$

| ...

Auswertung einer CTL Formel

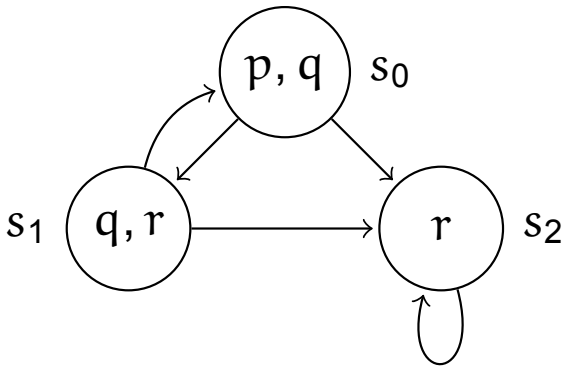
Algorithmus: $\texttt{holds}(\mathcal{M}, s, \psi)$

$$
\begin{aligned}
&\mid \ldots \\
&\mid \mathbf{AG}\,\varphi \rightsquigarrow \textit{for all } s \longrightarrow s_1 \longrightarrow \ldots \textit{ for all } s_i\colon \texttt{holds}(\mathcal{M}, s_i, \varphi) \\
&\mid \mathbf{EG}\,\varphi \rightsquigarrow \textit{exists } s \longrightarrow s_1 \longrightarrow \ldots \textit{ for all } s_i\colon \texttt{holds}(\mathcal{M}, s_i, \varphi) \\
&\mid \mathbf{AF}\,\varphi \rightsquigarrow \textit{for all } s \longrightarrow s_1 \longrightarrow \ldots \textit{ exists } s_i\colon \texttt{holds}(\mathcal{M}, s_i, \varphi) \\
&\mid \mathbf{EF}\,\varphi \rightsquigarrow \textit{exists } s \longrightarrow s_1 \longrightarrow \ldots \textit{ exists } s_i\colon \texttt{holds}(\mathcal{M}, s_i, \varphi) \\
&\mid \mathbf{A}[\varphi_1\,\mathbf{U}\,\varphi_2] \rightsquigarrow \textit{for all } \pi = s \longrightarrow s_1 \longrightarrow \ldots\colon \texttt{holds}(\mathcal{M}, \pi, \varphi_1\,\mathbf{U}\,\varphi_2) \\
&\mid \mathbf{E}[\varphi_1\,\mathbf{U}\,\varphi_2] \rightsquigarrow \textit{exists } \pi = s \longrightarrow s_1 \longrightarrow \ldots\colon \texttt{holds}(\mathcal{M}, \pi, \varphi_1\,\mathbf{U}\,\varphi_2)
\end{aligned}
$$

- $\mathbf{E}$ und $\mathbf{A}$ quantifizieren immer über Pfade
- $\mathbf{G}$ und $\mathbf{F}$ quantifizieren über Zustände
- $\mathbf{E}$ und $\mathbf{A}$ sind dual zueinander
- Gilt $\texttt{holds}(\mathcal{M}, s, \varphi)$, so schreiben wir auch $s \vDash_{\mathcal{M}} \varphi$ oder $s \vDash \varphi$

Auswertung einer CTL Formel

Beispiel



Beispiel (Auswertungen)

- $s_0 \models \mathbf{EX}(q \wedge r)$
- $s_0 \models \neg\,\mathbf{AX}(q \wedge r)$
- $s_0 \not\models \mathbf{EF}(p \wedge r)$
- $s_2 \models \mathbf{EG}\,r$
- $s_0 \models \mathbf{AF}\,r$
- $s_0 \models \mathbf{E}[(p \wedge q)\,\mathbf{U}\,r]$
- $s_0 \models \mathbf{A}[p\,\mathbf{U}\,r]$
- $s_0 \models \mathbf{AG}(p \vee q \vee r \rightarrow \mathbf{EF}\,\mathbf{EG}\,r)$

Was kann CTL?

Klassische Fragestellungen in CTL

- Gibt es einen Zustand, in dem `started` gilt, jedoch `ready` nicht: $\mathbf{EF}(\texttt{started} \wedge \neg \texttt{ready})$
- Wird in einem beliebigen Zustand ein Request empfangen, wird er auch irgendwann bestätigt: $\mathbf{AG}(\texttt{req} \rightarrow \mathbf{AF}\,\texttt{ack})$
- Was auch immer passiert, irgendwann wird ein bestimmter Prozess permanent im Deadlock sein: $\mathbf{AF}\,\mathbf{AG}\,\texttt{deadlock}$
- Aus jeden Zustand ist es möglich zu einen Restart Zustand zu kommen: $\mathbf{AG}\,\mathbf{EF}\,\texttt{restart}$
- Der Lift kann untätig mit geschlossenen Türen im dritten Stock stehen bleiben: $\mathbf{AG}(\texttt{floor3} \wedge \texttt{idle} \wedge \texttt{doorclosed} \rightarrow \mathbf{EG}(\texttt{floor3} \wedge \texttt{idle} \wedge \texttt{doorclosed}))$

? Fragestellung des Modelcheckings

Gegeben ein Modell, in welchen Zuständen gilt eine bestimmte Formel in CTL oder LTL?

Literaturtipps

Literaturhinweis

- *M. Huth & M. Ryan.* **Logic in Computer Science** **Chapter 3.** Cambridge University Press, 2004.
- *M. Kreuzer, S. Kühling.* **Logik für Informatiker** **Kapitel 6 & 7.** Pearson Studium, 2006.