Automatentheorie und ihre Anwendungen
Teil 4: endliche Automaten auf unendlichen Bäumen

Wintersemester 2018/19 Thomas Schneider

AG Theorie der künstlichen Intelligenz (TdKI)

`http://tinyurl.com/ws1819-autom`

Automatentheorie und ihre Anwendungen
Teil 4: endliche Automaten auf unendlichen Bäumen

Wintersemester 2018/19 Thomas Schneider

AG Theorie der künstlichen Intelligenz (TdKI)

http://tinyurl.com/ws1819-autom

8:30 11.1. $\longrightarrow$ **Folie 23**

TODO: siehe Verbesserungsvorschläge in TODO.txt

Überblick

Computation Tree Logic (CTL)

- Grenzen von LTL: kann nicht über Pfade quantifizieren
- Berechnungsbäume und CTL
- Ausdrucksvermögen von LTL und CTL im Vergleich
- Model-Checking mit CTL

Büchi-Automaten auf unendlichen Bäumen

- Definitionen und Beispiele
- äquivalente Automatenmodelle: Muller-, Paritätsautomaten
- Abschlusseigenschaften;
  Komplementierung von Muller-Automaten ⚠

8:30

Überblick

Computation Tree Logic (CTL)
- Grenzen von LTL: kann nicht über Pfade quantifizieren
- Berechnungsbäume und CTL
- Ausdrucksvermögen von LTL und CTL im Vergleich
- Model-Checking mit CTL

Büchi-Automaten auf unendlichen Bäumen
- Definitionen und Beispiele
- äquivalente Automatenmodelle: Muller-, Paritätsautomaten
- Abschlusseigenschaften;
  Komplementierung von Muller-Automaten ⚠

Überblick

2019-02-01

Teil 4: unendliche Bäume

└─Überblick

8:32

Überblick

1. *Model-Checking mit CTL*
2. Automaten auf unendlichen Bäumen
3. Komplementierung

Und nun . . .

2019-02-01

Teil 4: unendliche Bäume
└*Model-Checking mit CTL*
└Und nun . . .

Und nun . . .

1 *Model-Checking mit CTL*

2 Automaten auf unendlichen Bäumen

3 Komplementierung

Erinnerung an LTL *(Linear Temporal Logic)*

- System gegeben als Kripke-Struktur $\mathcal{S} = (S, S_0, R, \ell)$
- LTL-Formel φ_E beschreibt Pfade, die Eigenschaft E erfüllen
- Beispiel:
  „Wenn Fehler auftritt, ist er nach endlicher Zeit behoben.“
  $G(e \to F\neg e)$ ($e \in$ AV steht für „Error“)
- Umwandlung φ_E in GNBA $\mathcal{A}_E$, der zulässige Pfade beschreibt
- lösen damit Model-Checking-Problem:
  - Gilt E für *alle* Pfade ab S_0 in $\mathcal{S}$?
    (universelle Variante)
  - Gilt E für *mindestens einen* Pfad ab S_0 in $\mathcal{S}$?
    (existenzielle Variante)

LTL 1977 eingeführt durch Amir Pnueli, 1941-2009,
israelischer Informatiker (Haifa, Weizmann-Inst., Stanford, Tel Aviv, New York)

Erinnerung an LTL (Linear Temporal Logic)
System gegeben als Kripke-Struktur $\mathcal{S} = (S, S_0, R, \ell)$
LTL-Formel φ_E beschreibt Pfade, die Eigenschaft E erfüllen
Beispiel: „Wenn Fehler auftritt, ist er nach endlicher Zeit behoben.“ $G(e \to F\neg e)$ ($e \in$ AV steht für „Error“)
Umwandlung φ_E in GNBA $\mathcal{A}_E$, der zulässige Pfade beschreibt
lösen damit Model-Checking-Problem:
Gilt E für alle Pfade ab S_0 in $\mathcal{S}$? (universelle Variante)
Gilt E für mindestens einen Pfad ab S_0 in $\mathcal{S}$? (existenzielle Variante)
LTL 1977 eingeführt durch Amir Pnueli, 1941-2009,
israelischer Informatiker (Haifa, Weizmann-Inst., Stanford, Tel Aviv, New York)

8:33

Grenzen von LTL

„LTL-Formel φ_E beschreibt Pfade, die Eigenschaft E erfüllen“

Nicht ausdrückbar: zu jedem Zeitpunkt ist es immer *möglich*, die Berechnung auf eine gewisse Weise fortzusetzen

Beispiel: „Wenn ein Fehler auftritt, ist es *möglich*, ihn nach endlicher Zeit zu beheben.“

$G(e \to F\neg e)$ oder $GF\neg e$ sind

- zu stark in Verbindung mit universellem Model-Checking **T 4.1**
- zu schwach in Verbindung mit existenziellem MC **T 4.1 Forts.**

Grenzen von LTL

„LTL-Formel φ_E beschreibt Pfade, die Eigenschaft E erfüllen“

Nicht ausdrückbar: zu jedem Zeitpunkt ist es immer *möglich*, die Berechnung auf eine gewisse Weise fortzusetzen

Beispiel: „Wenn ein Fehler auftritt, ist es *möglich*, ihn nach endlicher Zeit zu beheben.“
$G(e \to F\neg e)$ oder $GF\neg e$ sind

- zu stark in Verbindung mit universellem Model-Checking **T 4.1**
- zu schwach in Verbindung mit existenziellem MC **T 4.1 Forts.**

8:34 bis 8:46

Ein Fall für CTL *(Computation Tree Logic)*

Abhilfe: Betrachten Berechnungsbäume statt Pfaden

Sei also $\mathcal{S} = (S, S_0, R, \ell)$ eine Kripke-Struktur

Berechnungsbaum für $s_0 \in S_0$

- entsteht durch „Auffalten“ von $\mathcal{S}$ in s_0
- enthält *alle unendlichen* Pfade, die in s_0 starten
  d. h.: jeder Zustand $s \in S$ hat als Kinder
  alle seine Nachfolgerzustände aus $\mathcal{S}$

$\mathcal{S}$ ist eine endliche Repräsentation aller ∞ Berechnungsbäume

Ein Fall für CTL *(Computation Tree Logic)*

Abhilfe: Betrachten Berechnungsbäume statt Pfaden

Sei also $\mathcal{S} = (S, S_0, R, \ell)$ eine Kripke-Struktur

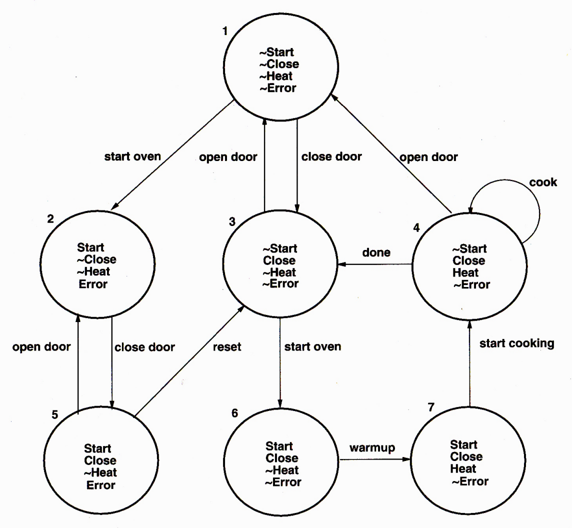
Berechnungsbaum für $s_0 \in S_0$

- entsteht durch „Auffalten“ von $\mathcal{S}$ in s_0
- enthält *alle unendlichen* Pfade, die in s_0 starten
  d. h.: jeder Zustand $s \in S$ hat als Kinder
  alle seine Nachfolgerzustände aus $\mathcal{S}$

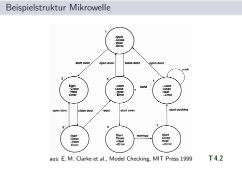
$\mathcal{S}$ ist eine endliche Repräsentation aller ∞ Berechnungsbäume

8:46

Beispielstruktur Mikrowelle



aus: E. M. Clarke et al., Model Checking, MIT Press 1999 **T 4.2**



8:47 bis 8:54

CTL intuitiv

CTL enthält **Pfadquantoren** $\boldsymbol{A}$, $\boldsymbol{E}$:

Operatoren, die über **alle** oder **einige** Berechnungen sprechen, die in einem bestimmten Zustand beginnen

8:54

CTL intuitiv

CTL enthält **Pfadquantoren** $\boldsymbol{A}$, $\boldsymbol{E}$:
Operatoren, die über **alle** oder **einige** Berechnungen sprechen,
die in einem bestimmten Zustand beginnen

CTL intuitiv

CTL enthält **Pfadquantoren *A*, *E***:

Operatoren, die über **alle** oder **einige** Berechnungen sprechen, die in einem bestimmten Zustand beginnen

Beispiel: $AGEF\neg e$

Für alle Berechnungen, die hier starten (A),
gibt es zu jedem Zeitpunkt in der Zukunft (G)
eine Möglichkeit, die Berechnung fortzusetzen (E),
so dass irgendwann in der Zukunft (F)
kein Fehler auftritt ($\neg e$)

8:54

CTL intuitiv

CTL enthält **Pfadquantoren *A*, *E***:
Operatoren, die über **alle** oder **einige** Berechnungen sprechen, die in einem bestimmten Zustand beginnen

Beispiel: $AGEF\neg e$
Für alle Berechnungen, die hier starten (A),
gibt es zu jedem Zeitpunkt in der Zukunft (G)
eine Möglichkeit, die Berechnung fortzusetzen (E),
so dass irgendwann in der Zukunft (F)
kein Fehler auftritt ($\neg e$)

CTL intuitiv

CTL enthält **Pfadquantoren A, E**:

Operatoren, die über **alle** oder **einige** Berechnungen sprechen,
die in einem bestimmten Zustand beginnen

Beispiel: $AGEF\neg e$

Für alle Berechnungen, die hier starten (A),
gibt es zu jedem Zeitpunkt in der Zukunft (G)
eine Möglichkeit, die Berechnung fortzusetzen (E),
so dass irgendwann in der Zukunft (F)
kein Fehler auftritt ($\neg e$)

CTL 1981 eingeführt durch
Edmund M. Clarke, *1945, Informatiker, Carnegie Mellon Univ. (Pittsburgh)
E. Allen Emerson, *1954, Informatiker, Univ. of Texas, Austin, USA
(beide Turing-Award-Träger 2007)

CTL intuitiv
CTL enthält Pfadquantoren A, E:
Operatoren, die über alle oder einige Berechnungen sprechen, die in einem bestimmten Zustand beginnen
Beispiel: $AGEF\neg e$
Für alle Berechnungen, die hier starten (A), gibt es zu jedem Zeitpunkt in der Zukunft (G) eine Möglichkeit, die Berechnung fortzusetzen (E), so dass irgendwann in der Zukunft (F) kein Fehler auftritt ($\neg e$)
CTL 1981 eingeführt durch Edmund M. Clarke, *1945, Informatiker, Carnegie Mellon Univ. (Pittsburgh) E. Allen Emerson, *1954, Informatiker, Univ. of Texas, Austin, USA (beide Turing-Award-Träger 2007)

8:54

CTL exakt

Trennung von Zustands- und Pfadformeln:

Zustandsformeln drücken Eigenschaften eines Zustandes aus

$$\zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$$

(p: Aussagenvariable; ζ, ζ_1, ζ_2: Zustandsformeln; ψ: Pfadformel)

CTL exakt
Trennung von Zustands- und Pfadformeln:
Zustandsformeln drücken Eigenschaften eines Zustandes aus
$$\zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$$
(p: Aussagenvariable; ζ, ζ_1, ζ_2: Zustandsformeln; ψ: Pfadformel)

8:56

CTL exakt

Trennung von Zustands- und Pfadformeln:

Zustandsformeln drücken Eigenschaften eines Zustandes aus

$$\zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$$

(p: Aussagenvariable; ζ, ζ_1, ζ_2: Zustandsformeln; ψ: Pfadformel)

Pfadformeln drücken Eigenschaften eines Pfades aus

$$\psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \; U \; \zeta_2$$

(ζ, ζ_1, ζ_2: Zustandsformeln)

CTL exakt

Trennung von Zustands- und Pfadformeln:

Zustandsformeln drücken Eigenschaften eines Zustandes aus

$$\zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$$

(p: Aussagenvariable; ζ, ζ_1, ζ_2: Zustandsformeln; ψ: Pfadformel)

Pfadformeln drücken Eigenschaften eines Pfades aus

$$\psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \; U \; \zeta_2$$

(ζ, ζ_1, ζ_2: Zustandsformeln)

8:56

CTL exakt

Trennung von Zustands- und Pfadformeln:

Zustandsformeln drücken Eigenschaften eines Zustandes aus

$$\zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$$

(p: Aussagenvariable; ζ, ζ_1, ζ_2: Zustandsformeln; ψ: Pfadformel)

Pfadformeln drücken Eigenschaften eines Pfades aus

$$\psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \, U \, \zeta_2$$

(ζ, ζ_1, ζ_2: Zustandsformeln)

$\rightsquigarrow$ in **zulässigen** CTL-Formeln muss

- jeder Pfadquantor von einem temporalen Operator gefolgt werden
- jeder temporale Operator direkt einem Pfadquantor folgen

8:56

CTL exakt

Trennung von Zustands- und Pfadformeln:

Zustandsformeln drücken Eigenschaften eines Zustandes aus

$$\zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$$

(p: Aussagenvariable; ζ, ζ_1, ζ_2: Zustandsformeln; ψ: Pfadformel)

Pfadformeln drücken Eigenschaften eines Pfades aus

$$\psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \, U \, \zeta_2$$

(ζ, ζ_1, ζ_2: Zustandsformeln)

$\rightsquigarrow$ in **zulässigen** CTL-Formeln muss

- jeder Pfadquantor von einem temporalen Operator gefolgt werden
- jeder temporale Operator direkt einem Pfadquantor folgen

Quiz: zulässige Formeln

Zur Erinnerung:

$$
\begin{aligned}
(ZF) \quad & \zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi \\
(PF) \quad & \psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \, U \, \zeta_2
\end{aligned}
$$

Quizfrage: Welche der folgenden Formeln sind zulässig?

$p \wedge q \quad EFp \quad AXp$
$E(p \, U \, q)$
$A((p \vee \neg p) \, U \, q)$
$E(p \vee AXq)$
$EX(p \vee AXq)$
$EF(p \, U \, q)$
$EFA(p \, U \, q)$

Quiz: zulässige Formeln
Zur Erinnerung:
(ZF) $\zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$
(PF) $\psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \, U \, \zeta_2$
Quizfrage: Welche der folgenden Formeln sind zulässig?
$p \wedge q \quad EFp \quad AXp$
$E(p \, U \, q)$
$A((p \vee \neg p) \, U \, q)$
$E(p \vee AXq)$
$EX(p \vee AXq)$
$EF(p \, U \, q)$
$EFA(p \, U \, q)$

9:00
Als Aufgabe, 2 min Nachdenken, 2 min Auflösung $\rightsquigarrow$ bis 9:04

Quiz: zulässige Formeln

Zur Erinnerung:

$$(ZF) \quad \zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$$
$$(PF) \quad \psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \, U \, \zeta_2$$

Quizfrage: Welche der folgenden Formeln sind zulässig?

$p \wedge q \quad EFp \quad AXp$ ✓
$E(p \, U \, q)$
$A((p \vee \neg p) \, U \, q)$
$E(p \vee AXq)$
$EX(p \vee AXq)$
$EF(p \, U \, q)$
$EFA(p \, U \, q)$

9:00
Als Aufgabe, 2 min Nachdenken, 2 min Auflösung $\rightsquigarrow$ bis 9:04

Quiz: zulässige Formeln

Zur Erinnerung:

$$(ZF) \quad \zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$$
$$(PF) \quad \psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \, U \, \zeta_2$$

Quizfrage: Welche der folgenden Formeln sind zulässig?

$p \wedge q \quad EFp \quad AXp$ ✓
$E(p \, U \, q)$ ✓
$A((p \vee \neg p) \, U \, q)$
$E(p \vee AXq)$
$EX(p \vee AXq)$
$EF(p \, U \, q)$
$EFA(p \, U \, q)$

Quiz: zulässige Formeln
Zur Erinnerung:
(ZF) ζ ::= p | ζ1 ∧ ζ2 | ζ1 ∨ ζ2 | ¬ζ | Eψ | Aψ
(PF) ψ ::= Fζ | Gζ | Xζ | ζ1 U ζ2
Quizfrage: Welche der folgenden Formeln sind zulässig?
p ∧ q EFp AXp ✓
E(p U q) ✓
A((p ∨ ¬p) U q)
E(p ∨ AXq)
EX(p ∨ AXq)
EF(p U q)
EFA(p U q)

9:00
Als Aufgabe, 2 min Nachdenken, 2 min Auflösung $\rightsquigarrow$ bis 9:04

Quiz: zulässige Formeln

Zur Erinnerung:

$$
\begin{array}{ll}
(ZF) & \zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi \\
(PF) & \psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \; U \; \zeta_2
\end{array}
$$

Quizfrage: Welche der folgenden Formeln sind zulässig?

- $p \wedge q$ $\quad EFp$ $\quad AXp$ ✓
- $E(p \; U \; q)$ ✓
- $A((p \vee \neg p) \; U \; q)$ ✓ (äquivalent zu AFq)
- $E(p \vee AXq)$
- $EX(p \vee AXq)$
- $EF(p \; U \; q)$
- $EFA(p \; U \; q)$

Quiz: zulässige Formeln
Zur Erinnerung:
(ZF) $\zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$
(PF) $\psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \; U \; \zeta_2$
Quizfrage: Welche der folgenden Formeln sind zulässig?
$p \wedge q$ EFp AXp ✓
$E(p \; U \; q)$ ✓
$A((p \vee \neg p) \; U \; q)$ ✓ (äquivalent zu AFq)
$E(p \vee AXq)$
$EX(p \vee AXq)$
$EF(p \; U \; q)$
$EFA(p \; U \; q)$

9:00
Als Aufgabe, 2 min Nachdenken, 2 min Auflösung $\rightsquigarrow$ bis 9:04

Quiz: zulässige Formeln

Zur Erinnerung:

$$
\begin{array}{ll}
(ZF) & \zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi \\
(PF) & \psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \; U \; \zeta_2
\end{array}
$$

Quizfrage: Welche der folgenden Formeln sind zulässig?

$p \wedge q \quad EFp \quad AXp$	✓	
$E(p \; U \; q)$	✓	
$A((p \vee \neg p) \; U \; q)$	✓	(äquivalent zu AFq)
$E(p \vee AXq)$	✘	(E nicht gefolgt von F, G, X, U)
$EX(p \vee AXq)$		
$EF(p \; U \; q)$		
$EFA(p \; U \; q)$		

Quiz: zulässige Formeln

Zur Erinnerung:

(ZF) $\zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$
(PF) $\psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \; U \; \zeta_2$

Quizfrage: Welche der folgenden Formeln sind zulässig?

$p \wedge q$ EFp AXp ✓
$E(p \; U \; q)$ ✓
$A((p \vee \neg p) \; U \; q)$ ✓ (äquivalent zu AFq)
$E(p \vee AXq)$ ✘ (E nicht gefolgt von F, G, X, U)
$EX(p \vee AXq)$
$EF(p \; U \; q)$
$EFA(p \; U \; q)$

9:00
Als Aufgabe, 2 min Nachdenken, 2 min Auflösung $\rightsquigarrow$ bis 9:04

Quiz: zulässige Formeln

Zur Erinnerung:

$$\begin{array}{ll} (ZF) & \zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi \\ (PF) & \psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \, U \, \zeta_2 \end{array}$$

Quizfrage: Welche der folgenden Formeln sind zulässig?

$p \wedge q \quad EFp \quad AXp$	✓	
$E(p \, U \, q)$	✓	
$A((p \vee \neg p) \, U \, q)$	✓	(äquivalent zu AFq)
$E(p \vee AXq)$	✘	(E nicht gefolgt von F, G, X, U)
$EX(p \vee AXq)$	✓	
$EF(p \, U \, q)$		
$EFA(p \, U \, q)$		

Quiz: zulässige Formeln

Zur Erinnerung:

(ZF) $\zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$
(PF) $\psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \, U \, \zeta_2$

Quizfrage: Welche der folgenden Formeln sind zulässig?

$p \wedge q \quad EFp \quad AXp$ ✓
$E(p \, U \, q)$ ✓
$A((p \vee \neg p) \, U \, q)$ ✓ (äquivalent zu AFq)
$E(p \vee AXq)$ ✘ (E nicht gefolgt von F, G, X, U)
$EX(p \vee AXq)$ ✓
$EF(p \, U \, q)$
$EFA(p \, U \, q)$

9:00
Als Aufgabe, 2 min Nachdenken, 2 min Auflösung $\rightsquigarrow$ bis 9:04

Quiz: zulässige Formeln

Zur Erinnerung:

$$
\begin{array}{ll}
(ZF) & \zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi \\
(PF) & \psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \; U \; \zeta_2
\end{array}
$$

Quizfrage: Welche der folgenden Formeln sind zulässig?

$p \wedge q \quad EFp \quad AXp$	✓	
$E(p \; U \; q)$	✓	
$A((p \vee \neg p) \; U \; q)$	✓	(äquivalent zu AFq)
$E(p \vee AXq)$	✘	(E nicht gefolgt von F, G, X, U)
$EX(p \vee AXq)$	✓	
$EF(p \; U \; q)$	✘	(U folgt nicht E oder A)
$EFA(p \; U \; q)$		

Quiz: zulässige Formeln

Zur Erinnerung:

(ZF) $\zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$
(PF) $\psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1 \; U \; \zeta_2$

Quizfrage: Welche der folgenden Formeln sind zulässig?

$p \wedge q \quad EFp \quad AXp$ ✓
$E(p \; U \; q)$ ✓
$A((p \vee \neg p) \; U \; q)$ ✓ (äquivalent zu AFq)
$E(p \vee AXq)$ ✘ (E nicht gefolgt von F, G, X, U)
$EX(p \vee AXq)$ ✓
$EF(p \; U \; q)$ ✘ (U folgt nicht E oder A)
$EFA(p \; U \; q)$

9:00
Als Aufgabe, 2 min Nachdenken, 2 min Auflösung $\rightsquigarrow$ bis 9:04

Quiz: zulässige Formeln

Zur Erinnerung:

$$(ZF) \quad \zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$$
$$(PF) \quad \psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1\ U\ \zeta_2$$

Quizfrage: Welche der folgenden Formeln sind zulässig?

$p \wedge q \quad EFp \quad AXp$	✓	
$E(p\ U\ q)$	✓	
$A((p \vee \neg p)\ U\ q)$	✓	(äquivalent zu AFq)
$E(p \vee AXq)$	✘	(E nicht gefolgt von F, G, X, U)
$EX(p \vee AXq)$	✓	
$EF(p\ U\ q)$	✘	(U folgt nicht E oder A)
$EFA(p\ U\ q)$	✓	

2019-02-01

Teil 4: unendliche Bäume
└ *Model-Checking mit CTL*
└ Quiz: zulässige Formeln

Quiz: zulässige Formeln

Zur Erinnerung:

$(ZF) \quad \zeta ::= p \mid \zeta_1 \wedge \zeta_2 \mid \zeta_1 \vee \zeta_2 \mid \neg\zeta \mid E\psi \mid A\psi$
$(PF) \quad \psi ::= F\zeta \mid G\zeta \mid X\zeta \mid \zeta_1\ U\ \zeta_2$

Quizfrage: Welche der folgenden Formeln sind zulässig?

$p \wedge q$ EFp AXp ✓
$E(p\ U\ q)$ ✓
$A((p \vee \neg p)\ U\ q)$ ✓ (äquivalent zu AFq)
$E(p \vee AXq)$ ✘ (E nicht gefolgt von F, G, X, U)
$EX(p \vee AXq)$ ✓
$EF(p\ U\ q)$ ✘ (U folgt nicht E oder A)
$EFA(p\ U\ q)$ ✓

9:00
Als Aufgabe, 2 min Nachdenken, 2 min Auflösung $\rightsquigarrow$ bis 9:04

CTL-Semantik

CTL-Formeln werden über **Zuständen und Pfaden** von Kripke-Strukturen $\mathcal{S} = (S, S_0, R, \ell)$ interpretiert

Schreibweisen

- $s \models \zeta$ für Zustände $s \in S$ und Zustandsformeln ζ
- $\pi \models \psi$ für Pfade π und Pfadformeln ψ

Hilfsbegriffe

- Paths(s): Menge aller Pfade, die in Zustand s beginnen
- $\pi[i]$: i-ter Zustand auf dem Pfad π
  d. h. wenn $\pi = s_0 s_1 s_2 \ldots$, dann $\pi[i] = s_i$

9:04

CTL-Semantik

CTL-Formeln werden über **Zuständen und Pfaden** von Kripke-Strukturen $\mathcal{S} = (S, S_0, R, \ell)$ interpretiert

Schreibweisen
- $s \models \zeta$ für Zustände $s \in S$ und Zustandsformeln ζ
- $\pi \models \psi$ für Pfade π und Pfadformeln ψ

Hilfsbegriffe
- Paths(s): Menge aller Pfade, die in Zustand s beginnen
- $\pi[i]$: i-ter Zustand auf dem Pfad π
  d. h. wenn $\pi = s_0 s_1 s_2 \ldots$, dann $\pi[i] = s_i$

CTL-Semantik

Sei $\mathcal{S} = (S, S_0, R, \ell)$ eine Kripke-Struktur.

Definition 4.1

Erfülltheit von Zustandsformeln in Zuständen $s \in S$

$s \models p$	falls	$p \in \ell(s)$, für alle $p \in \mathrm{AV}$	
$s \models \neg\zeta$	falls	$s \not\models \zeta$	
$s \models \zeta_1 \wedge \zeta_2$	falls	$s \models \zeta_1$ und $s \models \zeta_2$	(analog für $\zeta_1 \vee \zeta_2$)
$s \models E\psi$	falls	$\pi \models \psi$ für ein $\pi \in \mathrm{Paths}(s)$	
$s \models A\psi$	falls	$\pi \models \psi$ für alle $\pi \in \mathrm{Paths}(s)$	

2019-02-01

Teil 4: unendliche Bäume
└*Model-Checking mit CTL*
└CTL-Semantik

CTL-Semantik
Sei $\mathcal{S} = (S, S_0, R, \ell)$ eine Kripke-Struktur.
Definition 4.1
Erfülltheit von Zustandsformeln in Zuständen $s \in S$
$s \models p$ falls $p \in \ell(s)$, für alle $p \in \mathrm{AV}$
$s \models \neg\zeta$ falls $s \not\models \zeta$
$s \models \zeta_1 \wedge \zeta_2$ falls $s \models \zeta_1$ und $s \models \zeta_2$ (analog für $\zeta_1 \vee \zeta_2$)
$s \models E\psi$ falls $\pi \models \psi$ für ein $\pi \in \mathrm{Paths}(s)$
$s \models A\psi$ falls $\pi \models \psi$ für alle $\pi \in \mathrm{Paths}(s)$

9:06

CTL-Semantik

Sei $\mathcal{S} = (S, S_0, R, \ell)$ eine Kripke-Struktur.

Definition 4.1

Erfülltheit von Zustandsformeln in Zuständen $s \in S$

$s \models p$	falls	$p \in \ell(s)$, für alle $p \in \mathrm{AV}$	
$s \models \neg\zeta$	falls	$s \not\models \zeta$	
$s \models \zeta_1 \wedge \zeta_2$	falls	$s \models \zeta_1$ und $s \models \zeta_2$	(analog für $\zeta_1 \vee \zeta_2$)
$s \models E\psi$	falls	$\pi \models \psi$ für ein $\pi \in \mathrm{Paths}(s)$	
$s \models A\psi$	falls	$\pi \models \psi$ für alle $\pi \in \mathrm{Paths}(s)$	

Erfülltheit von Pfadformeln in Pfaden π in $\mathcal{S}$

$\pi \models X\zeta$	falls	$\pi[1] \models \zeta$	(analog für $F\zeta$ und $G\zeta$)
$\pi \models \zeta_1\, U\, \zeta_2$	falls	$\pi[j] \models \zeta_2$ für ein $j \geqslant 0$ und $\pi[k] \models \zeta_1$ für alle k mit $0 \leqslant k < j$	

9:06

CTL-Semantik

Sei $\mathcal{S} = (S, S_0, R, \ell)$ eine Kripke-Struktur.

Definition 4.1

Erfülltheit von Zustandsformeln in Zuständen $s \in S$

$s \models p$	falls	$p \in \ell(s)$, für alle $p \in \mathrm{AV}$	
$s \models \neg\zeta$	falls	$s \not\models \zeta$	
$s \models \zeta_1 \wedge \zeta_2$	falls	$s \models \zeta_1$ und $s \models \zeta_2$	(analog für $\zeta_1 \vee \zeta_2$)
$s \models E\psi$	falls	$\pi \models \psi$ für ein $\pi \in \mathrm{Paths}(s)$	
$s \models A\psi$	falls	$\pi \models \psi$ für alle $\pi \in \mathrm{Paths}(s)$	

Erfülltheit von Pfadformeln in Pfaden π in $\mathcal{S}$

$\pi \models X\zeta$	falls	$\pi[1] \models \zeta$	(analog für $F\zeta$ und $G\zeta$)
$\pi \models \zeta_1 \, U \, \zeta_2$	falls	$\pi[j] \models \zeta_2$ für ein $j \geqslant 0$ und $\pi[k] \models \zeta_1$ für alle k mit $0 \leqslant k < j$	

Schreiben $\quad \mathcal{S} \models \zeta \quad$ falls $\quad s_0 \models \zeta$ für alle $s_0 \in S_0$

2019-02-01

Teil 4: unendliche Bäume
└ *Model-Checking mit CTL*
└ CTL-Semantik

CTL-Semantik
Sei $\mathcal{S} = (S, S_0, R, \ell)$ eine Kripke-Struktur.
Definition 4.1
Erfülltheit von Zustandsformeln in Zuständen $s \in S$
$s \models p$ falls $p \in \ell(s)$, für alle $p \in \mathrm{AV}$
$s \models \neg\zeta$ falls $s \not\models \zeta$
$s \models \zeta_1 \wedge \zeta_2$ falls $s \models \zeta_1$ und $s \models \zeta_2$ (analog für $\zeta_1 \vee \zeta_2$)
$s \models E\psi$ falls $\pi \models \psi$ für ein $\pi \in \mathrm{Paths}(s)$
$s \models A\psi$ falls $\pi \models \psi$ für alle $\pi \in \mathrm{Paths}(s)$
Erfülltheit von Pfadformeln in Pfaden π in $\mathcal{S}$
$\pi \models X\zeta$ falls $\pi[1] \models \zeta$ (analog für $F\zeta$ und $G\zeta$)
$\pi \models \zeta_1 \, U \, \zeta_2$ falls $\pi[j] \models \zeta_2$ für ein $j \geqslant 0$ und $\pi[k] \models \zeta_1$ für alle k mit $0 \leqslant k < j$
Schreiben $\mathcal{S} \models \zeta$ falls $s_0 \models \zeta$ für alle $s_0 \in S_0$

9:06

Zurück zu unseren Beispielen: Spezifikationen in CTL

Beispiel Nebenläufigkeit

- Beide Teilprogramme sind nie zugleich im kritischen Bereich.
  $AG\neg(p_{12} \wedge p_{22})$ ($p_i \in$ AV: „Programmzähler in Zeile i“)

Zurück zu unseren Beispielen: Spezifikationen in CTL
Beispiel Nebenläufigkeit
Beide Teilprogramme sind nie zugleich im kritischen Bereich.
$AG\neg(p_{12} \wedge p_{22})$ ($p_i \in$ AV: „Programmzähler in Zeile i“)

9:10

Zurück zu unseren Beispielen: Spezifikationen in CTL

Beispiel Nebenläufigkeit

- Beide Teilprogramme sind nie zugleich im kritischen Bereich.
  $AG\neg(p_{12} \wedge p_{22})$ ($p_i \in$ AV: „Programmzähler in Zeile i“)
- Jedes Teilprog. kommt beliebig oft in seinen krit. Bereich.
  $AGAFp_{12} \wedge AGAFp_{22}$

Zurück zu unseren Beispielen: Spezifikationen in CTL

Beispiel Nebenläufigkeit

- Beide Teilprogramme sind nie zugleich im kritischen Bereich.
  $AG\neg(p_{12} \wedge p_{22})$ ($p_i \in$ AV: „Programmzähler in Zeile i“)
- Jedes Teilprog. kommt beliebig oft in seinen krit. Bereich.
  $AGAFp_{12} \wedge AGAFp_{22}$

9:10

Zurück zu unseren Beispielen: Spezifikationen in CTL

Beispiel Nebenläufigkeit

- Beide Teilprogramme sind nie zugleich im kritischen Bereich.
  $AG\neg(p_{12} \wedge p_{22})$ ($p_i \in$ AV: „Programmzähler in Zeile i")
- Jedes Teilprog. kommt beliebig oft in seinen krit. Bereich.
  $AGAFp_{12} \wedge AGAFp_{22}$
- Jedes Teilprog. *kann* beliebig oft in seinen kB kommen.
  $AGEFp_{12} \wedge AGEFp_{22}$

Zurück zu unseren Beispielen: Spezifikationen in CTL

Beispiel Nebenläufigkeit

- Beide Teilprogramme sind nie zugleich im kritischen Bereich. $AG\neg(p_{12} \wedge p_{22})$ ($p_i \in$ AV: „Programmzähler in Zeile i")
- Jedes Teilprog. kommt beliebig oft in seinen krit. Bereich. $AGAFp_{12} \wedge AGAFp_{22}$
- Jedes Teilprog. *kann* beliebig oft in seinen kB kommen. $AGEFp_{12} \wedge AGEFp_{22}$

9:10

Zurück zu unseren Beispielen: Spezifikationen in CTL

Beispiel Nebenläufigkeit

- Beide Teilprogramme sind nie zugleich im kritischen Bereich.
  $AG\neg(p_{12} \wedge p_{22})$ ($p_i \in$ AV: „Programmzähler in Zeile i“)
- Jedes Teilprog. kommt beliebig oft in seinen krit. Bereich.
  $AGAFp_{12} \wedge AGAFp_{22}$
- Jedes Teilprog. *kann* beliebig oft in seinen kB kommen.
  $AGEFp_{12} \wedge AGEFp_{22}$

Liveness properties:

$AG\zeta$ besagt: „ζ ist in allen Berechnungen immer wahr“

$AGAF\zeta$ besagt: „ζ ist in allen Berechnungen ∞ oft wahr“

$AGEF\zeta$ besagt: „jede begonnene Berechnung kann so fortgesetzt werden, dass ζ irgendwann wahr wird.“

Zurück zu unseren Beispielen: Spezifikationen in CTL
Beispiel Nebenläufigkeit
Beide Teilprogramme sind nie zugleich im kritischen Bereich. $AG\neg(p_{12} \wedge p_{22})$ ($p_i \in$ AV: „Programmzähler in Zeile i“)
Jedes Teilprog. kommt beliebig oft in seinen krit. Bereich. $AGAFp_{12} \wedge AGAFp_{22}$
Jedes Teilprog. *kann* beliebig oft in seinen kB kommen. $AGEFp_{12} \wedge AGEFp_{22}$
Liveness properties:
$AG\zeta$ besagt: „ζ ist in allen Berechnungen immer wahr“
$AGAF\zeta$ besagt: „ζ ist in allen Berechnungen ∞ oft wahr“
$AGEF\zeta$ besagt: „jede begonnene Berechnung kann so fortgesetzt werden, dass ζ irgendwann wahr wird.“

9:10

Zurück zu unseren Beispielen: Spezifikationen in CTL

Beispiel Mikrowelle

- „Wenn Fehler auftritt, ist er nach endlicher Zeit behoben.“
  $AG(e \to AF\neg e)$ ($e \in$ AV steht für „Error“)

Zurück zu unseren Beispielen: Spezifikationen in CTL
Beispiel Mikrowelle
„Wenn Fehler auftritt, ist er nach endlicher Zeit behoben.“
$AG(e \to AF\neg e)$ ($e \in$ AV steht für „Error“)

9:12

Zurück zu unseren Beispielen: Spezifikationen in CTL

Beispiel Mikrowelle

- „Wenn Fehler auftritt, ist er nach endlicher Zeit behoben.“
  $AG(e \to AF\neg e)$ ($e \in \mathrm{AV}$ steht für „Error“)
- „Wenn Fehler auftritt, *kann* er nach endl. Z. behoben werden“
  $AG(e \to EF\neg e)$

Zurück zu unseren Beispielen: Spezifikationen in CTL
Beispiel Mikrowelle
„Wenn Fehler auftritt, ist er nach endlicher Zeit behoben.“ $AG(e \to AF\neg e)$ ($e \in \mathrm{AV}$ steht für „Error“)
„Wenn Fehler auftritt, *kann* er nach endl. Z. behoben werden“ $AG(e \to EF\neg e)$

9:12

Zurück zu unseren Beispielen: Spezifikationen in CTL

Beispiel Mikrowelle

- „Wenn Fehler auftritt, ist er nach endlicher Zeit behoben.“
  $AG(e \rightarrow AF\neg e)$ ($e \in$ AV steht für „Error“)
- „Wenn Fehler auftritt, *kann* er nach endl. Z. behoben werden“
  $AG(e \rightarrow EF\neg e)$
- „Wenn die Mikrowelle gestartet wird,
  beginnt sie nach endlicher Zeit zu heizen.“
  $AG(s \rightarrow AFh)$ ($s, h \in$ AV stehen für „Start“ bzw. „Heat“)

Zurück zu unseren Beispielen: Spezifikationen in CTL
Beispiel Mikrowelle
„Wenn Fehler auftritt, ist er nach endlicher Zeit behoben.“ $AG(e \rightarrow AF\neg e)$ ($e \in$ AV steht für „Error“)
„Wenn Fehler auftritt, *kann* er nach endl. Z. behoben werden“ $AG(e \rightarrow EF\neg e)$
„Wenn die Mikrowelle gestartet wird, beginnt sie nach endlicher Zeit zu heizen.“ $AG(s \rightarrow AFh)$ ($s, h \in$ AV stehen für „Start“ bzw. „Heat“)

9:12

Zurück zu unseren Beispielen: Spezifikationen in CTL

Beispiel Mikrowelle

- „Wenn Fehler auftritt, ist er nach endlicher Zeit behoben.“
  $AG(e \to AF\neg e)$ ($e \in$ AV steht für „Error“)
- „Wenn Fehler auftritt, *kann* er nach endl. Z. behoben werden“
  $AG(e \to EF\neg e)$
- „Wenn die Mikrowelle gestartet wird,
  beginnt sie nach endlicher Zeit zu heizen.“
  $AG(s \to AFh)$ ($s, h \in$ AV stehen für „Start“ bzw. „Heat“)
- „Wenn die Mikrowelle gestartet wird,
  ist es möglich, dass sie nach endlicher Zeit zu heizen beginnt.“
  $AG(s \to EFh)$

Zurück zu unseren Beispielen: Spezifikationen in CTL
Beispiel Mikrowelle
„Wenn Fehler auftritt, ist er nach endlicher Zeit behoben.“ $AG(e \to AF\neg e)$ ($e \in$ AV steht für „Error“)
„Wenn Fehler auftritt, *kann* er nach endl. Z. behoben werden“ $AG(e \to EF\neg e)$
„Wenn die Mikrowelle gestartet wird, beginnt sie nach endlicher Zeit zu heizen.“ $AG(s \to AFh)$ ($s, h \in$ AV stehen für „Start“ bzw. „Heat“)
„Wenn die Mikrowelle gestartet wird, *ist es möglich*, dass sie nach endlicher Zeit zu heizen beginnt.“ $AG(s \to EFh)$

9:12

Zurück zu unseren Beispielen: Spezifikationen in CTL

Beispiel Mikrowelle

- „Wenn Fehler auftritt, ist er nach endlicher Zeit behoben.“
  $AG(e \rightarrow AF\neg e)$ ($e \in$ AV steht für „Error“)
- „Wenn Fehler auftritt, *kann* er nach endl. Z. behoben werden“
  $AG(e \rightarrow EF\neg e)$
- „Wenn die Mikrowelle gestartet wird,
  beginnt sie nach endlicher Zeit zu heizen.“
  $AG(s \rightarrow AFh)$ ($s, h \in$ AV stehen für „Start“ bzw. „Heat“)
- „Wenn die Mikrowelle gestartet wird,
  ist es möglich, dass sie nach endlicher Zeit zu heizen beginnt.“
  $AG(s \rightarrow EFh)$

Progress properties: $AG(\zeta_1 \rightarrow AF\zeta_2)$, $AG(\zeta_1 \rightarrow EF\zeta_2)$ bedeuten:
Wann immer ζ_1 eintritt, ist nach endlicher Zeit ζ_2 „garantiert“

Zurück zu unseren Beispielen: Spezifikationen in CTL

Beispiel Mikrowelle

- „Wenn Fehler auftritt, ist er nach endlicher Zeit behoben.“ $AG(e \rightarrow AF\neg e)$ ($e \in$ AV steht für „Error“)
- „Wenn Fehler auftritt, *kann* er nach endl. Z. behoben werden“ $AG(e \rightarrow EF\neg e)$
- „Wenn die Mikrowelle gestartet wird, beginnt sie nach endlicher Zeit zu heizen.“ $AG(s \rightarrow AFh)$ ($s, h \in$ AV stehen für „Start“ bzw. „Heat“)
- „Wenn die Mikrowelle gestartet wird, *ist es möglich*, dass sie nach endlicher Zeit zu heizen beginnt.“ $AG(s \rightarrow EFh)$

Progress properties: $AG(\zeta_1 \rightarrow AF\zeta_2)$, $AG(\zeta_1 \rightarrow EF\zeta_2)$ bedeuten: Wann immer ζ_1 eintritt, ist nach endlicher Zeit ζ_2 „garantiert“

9:12

Ausdrucksstärke von CTL versus LTL

Definition 4.2

Seien ζ eine CTL-Zustandsformel und φ eine LTL-Formel.

ζ und φ sind **äquivalent**, geschrieben $\boldsymbol{\zeta \equiv \varphi}$, wenn für alle Kripke-Strukturen $\mathcal{S} = (S, S_0, R, \ell)$ gilt:

$$\mathcal{S} \models \zeta \quad \text{gdw.} \quad \mathcal{S} \models \varphi$$

Zur Erinnerung:

- $\mathcal{S} \models \zeta$, wenn $s_0 \models \zeta$ für **alle** $s_0 \in S_0$
- $\mathcal{S} \models \varphi$, wenn $\pi, 0 \models \varphi$ für **alle** $\pi \in \text{Paths}(s_0)$ und **alle** $s_0 \in S_0$

Ausdrucksstärke von CTL versus LTL

Definition 4.2

Seien ζ eine CTL-Zustandsformel und φ eine LTL-Formel.

ζ und φ sind **äquivalent**, geschrieben $\zeta \equiv \varphi$, wenn für alle Kripke-Strukturen $\mathcal{S} = (S, S_0, R, \ell)$ gilt:

$\mathcal{S} \models \zeta$ gdw. $\mathcal{S} \models \varphi$

Zur Erinnerung:

- $\mathcal{S} \models \zeta$, wenn $s_0 \models \zeta$ für alle $s_0 \in S_0$
- $\mathcal{S} \models \varphi$, wenn $\pi, 0 \models \varphi$ für alle $\pi \in \text{Paths}(s_0)$ und alle $s_0 \in S_0$

9:15: 5 min Pause, dann 2 min für Folie

Ausdrucksstärke von CTL versus LTL

Lemma 4.3

$AFAGp \not\equiv FGp$

Ausdrucksstärke von CTL versus LTL
Lemma 4.3
$AFAGp \not\equiv FGp$

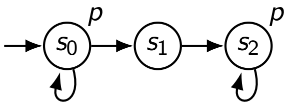
9:22

Ausdrucksstärke von CTL versus LTL

Lemma 4.3

$AFAGp \not\equiv FGp$

Beweis. Betrachte Kripke-Struktur $\mathcal{S}$:



- alle Pfade $\pi \in \text{Paths}(s_0)$ erfüllen FGp

Ausdrucksstärke von CTL versus LTL

Lemma 4.3
$AFAGp \not\equiv FGp$

Beweis. Betrachte Kripke-Struktur $\mathcal{S}$:

- alle Pfade $\pi \in \text{Paths}(s_0)$ erfüllen FGp

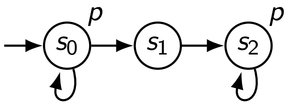
9:22

Ausdrucksstärke von CTL versus LTL

Lemma 4.3

$AFAGp \not\equiv FGp$

Beweis. Betrachte Kripke-Struktur $\mathcal{S}$:



- alle Pfade $\pi \in \text{Paths}(s_0)$ erfüllen FGp
- aber $\mathcal{S} \not\models AFAGp$:

Ausdrucksstärke von CTL versus LTL
Lemma 4.3
$AFAGp \not\equiv FGp$
Beweis. Betrachte Kripke-Struktur $\mathcal{S}$:
alle Pfade $\pi \in \text{Paths}(s_0)$ erfüllen FGp
aber $\mathcal{S} \not\models AFAGp$:

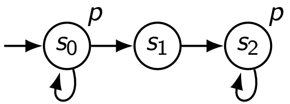
9:22

Ausdrucksstärke von CTL versus LTL

Lemma 4.3

$AFAGp \not\equiv FGp$

Beweis. Betrachte Kripke-Struktur $\mathcal{S}$:

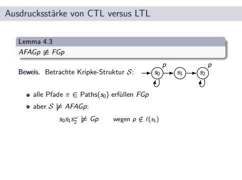


- alle Pfade $\pi \in \text{Paths}(s_0)$ erfüllen FGp
- aber $\mathcal{S} \not\models AFAGp$:

$$s_0 s_1 s_2^\omega \not\models Gp \qquad \text{wegen } p \notin \ell(s_1)$$

2019-02-01

Teil 4: unendliche Bäume
└*Model-Checking mit CTL*
└Ausdrucksstärke von CTL versus LTL

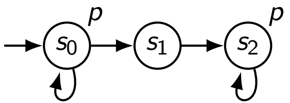


9:22

Ausdrucksstärke von CTL versus LTL

Lemma 4.3

$AFAGp \not\equiv FGp$

Beweis. Betrachte Kripke-Struktur $\mathcal{S}$: 

- alle Pfade $\pi \in \text{Paths}(s_0)$ erfüllen FGp
- aber $\mathcal{S} \not\models AFAGp$:

$$\begin{array}{lll} & s_0 s_1 s_2^\omega \not\models Gp & \text{wegen } p \notin \ell(s_1) \\ \Rightarrow & s_0 \not\models AGp & \text{weil } s_0 s_1 s_2^\omega \in \text{Paths}(s_0) \end{array}$$

Ausdrucksstärke von CTL versus LTL
Lemma 4.3
$AFAGp \not\equiv FGp$
Beweis. Betrachte Kripke-Struktur $\mathcal{S}$:
alle Pfade $\pi \in \text{Paths}(s_0)$ erfüllen FGp
aber $\mathcal{S} \not\models AFAGp$:
$s_0 s_1 s_2^\omega \not\models Gp$ wegen $p \notin \ell(s_1)$
$\Rightarrow$ $s_0 \not\models AGp$ weil $s_0 s_1 s_2^\omega \in \text{Paths}(s_0)$

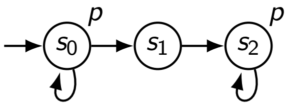
9:22

Ausdrucksstärke von CTL versus LTL

Lemma 4.3

$AFAGp \not\equiv FGp$

Beweis. Betrachte Kripke-Struktur $\mathcal{S}$:



- alle Pfade $\pi \in \text{Paths}(s_0)$ erfüllen FGp
- aber $\mathcal{S} \not\models AFAGp$:

$$
\begin{array}{lll}
 & s_0 s_1 s_2^\omega \not\models Gp & \text{wegen } p \notin \ell(s_1) \\
\Rightarrow & s_0 \not\models AGp & \text{weil } s_0 s_1 s_2^\omega \in \text{Paths}(s_0) \\
\Rightarrow & s_0^\omega \not\models FAGp & \text{weil } s_0^\omega \text{ nur aus } s_0 \text{ besteht}
\end{array}
$$

Ausdrucksstärke von CTL versus LTL

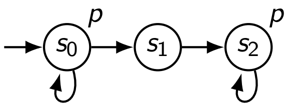
9:22

Ausdrucksstärke von CTL versus LTL

Lemma 4.3

$AFAGp \not\equiv FGp$

Beweis. Betrachte Kripke-Struktur $\mathcal{S}$:



- alle Pfade $\pi \in \text{Paths}(s_0)$ erfüllen FGp
- aber $\mathcal{S} \not\models AFAGp$:

$$
\begin{array}{llll}
 & s_0 s_1 s_2^\omega \not\models Gp & & \text{wegen } p \notin \ell(s_1) \\
\Rightarrow & s_0 \not\models AGp & & \text{weil } s_0 s_1 s_2^\omega \in \text{Paths}(s_0) \\
\Rightarrow & s_0^\omega \not\models FAGp & & \text{weil } s_0^\omega \text{ nur aus } s_0 \text{ besteht} \\
\Rightarrow & s_0 \not\models AFAGp & & \text{weil } s_0^\omega \in \text{Paths}(s_0)
\end{array}
$$

2019-02-01

Teil 4: unendliche Bäume
└─*Model-Checking mit CTL*
└─Ausdrucksstärke von CTL versus LTL

Ausdrucksstärke von CTL versus LTL

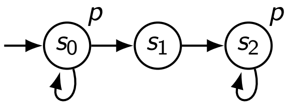
9:22

Ausdrucksstärke von CTL versus LTL

Lemma 4.3

$AFAGp \not\equiv FGp$

Beweis. Betrachte Kripke-Struktur $\mathcal{S}$:



- alle Pfade $\pi \in \text{Paths}(s_0)$ erfüllen FGp
- aber $\mathcal{S} \not\models AFAGp$:

$$\begin{array}{lll} & s_0 s_1 s_2^\omega \not\models Gp & \text{wegen } p \notin \ell(s_1) \\ \Rightarrow & s_0 \not\models AGp & \text{weil } s_0 s_1 s_2^\omega \in \text{Paths}(s_0) \\ \Rightarrow & s_0^\omega \not\models FAGp & \text{weil } s_0^\omega \text{ nur aus } s_0 \text{ besteht} \\ \Rightarrow & s_0 \not\models AFAGp & \text{weil } s_0^\omega \in \text{Paths}(s_0) \end{array}$$

□

Ausdrucksstärke von CTL versus LTL
Lemma 4.3
$AFAGp \not\equiv FGp$
Beweis. Betrachte Kripke-Struktur $\mathcal{S}$:
alle Pfade $\pi \in \text{Paths}(s_0)$ erfüllen FGp
aber $\mathcal{S} \not\models AFAGp$:
$s_0 s_1 s_2^\omega \not\models Gp$ wegen $p \notin \ell(s_1)$
$\Rightarrow s_0 \not\models AGp$ weil $s_0 s_1 s_2^\omega \in \text{Paths}(s_0)$
$\Rightarrow s_0^\omega \not\models FAGp$ weil s_0^ω nur aus s_0 besteht
$\Rightarrow s_0 \not\models AFAGp$ weil $s_0^\omega \in \text{Paths}(s_0)$

9:22

Ausdrucksstärke von CTL versus LTL

Lemma 4.4

Sei ζ eine CTL-Zustandsformel und ζ' die LTL-Formel, die man durch Entfernen aller Pfadquantoren aus ζ erhält. Dann gilt:

$$\zeta \equiv \zeta' \text{ oder es gibt keine zu } \zeta \text{ äquivalente LTL-Formel.}$$

Ohne Beweis. (Clarke, Draghicescu 1988)

Ausdrucksstärke von CTL versus LTL

Lemma 4.4

Sei ζ eine CTL-Zustandsformel und ζ' die LTL-Formel, die man durch Entfernen aller Pfadquantoren aus ζ erhält. Dann gilt:

$\zeta \equiv \zeta'$ oder es gibt keine zu ζ äquivalente LTL-Formel.

Ohne Beweis. (Clarke, Draghicescu 1988)

9:26 bis 9:38

Ausdrucksstärke von CTL versus LTL

Lemma 4.4

Sei ζ eine CTL-Zustandsformel und ζ' die LTL-Formel, die man durch Entfernen aller Pfadquantoren aus ζ erhält. Dann gilt:

$\zeta \equiv \zeta'$ oder es gibt keine zu ζ äquivalente LTL-Formel.

Ohne Beweis. (Clarke, Draghicescu 1988)

Lemma 4.5

(1) Es gibt keine zu *AFAGp* äquivalente LTL-Formel.

(2) Es gibt keine zu *FGp* äquivalente CTL-Zustandsformel.

Ausdrucksstärke von CTL versus LTL

Lemma 4.4
Sei ζ eine CTL-Zustandsformel und ζ' die LTL-Formel, die man durch Entfernen aller Pfadquantoren aus ζ erhält. Dann gilt:
$\zeta \equiv \zeta'$ oder es gibt keine zu ζ äquivalente LTL-Formel.

Ohne Beweis. (Clarke, Draghicescu 1988)

Lemma 4.5
(1) Es gibt keine zu *AFAGp* äquivalente LTL-Formel.
(2) Es gibt keine zu *FGp* äquivalente CTL-Zustandsformel.

9:26 bis 9:38

Ausdrucksstärke von CTL versus LTL

Lemma 4.4

Sei ζ eine CTL-Zustandsformel und ζ' die LTL-Formel, die man durch Entfernen aller Pfadquantoren aus ζ erhält. Dann gilt:

$\zeta \equiv \zeta'$ oder es gibt keine zu ζ äquivalente LTL-Formel.

Ohne Beweis. (Clarke, Draghicescu 1988)

Lemma 4.5

(1) Es gibt keine zu *AFAGp* äquivalente LTL-Formel.

(2) Es gibt keine zu *FGp* äquivalente CTL-Zustandsformel.

Beweis.

(1) folgt aus Lemmas 4.3 und 4.4

Ausdrucksstärke von CTL versus LTL

Lemma 4.4
Sei ζ eine CTL-Zustandsformel und ζ' die LTL-Formel, die man durch Entfernen aller Pfadquantoren aus ζ erhält. Dann gilt:
$\zeta \equiv \zeta'$ oder es gibt keine zu ζ äquivalente LTL-Formel.
Ohne Beweis. (Clarke, Draghicescu 1988)
Lemma 4.5
(1) Es gibt keine zu *AFAGp* äquivalente LTL-Formel.
(2) Es gibt keine zu *FGp* äquivalente CTL-Zustandsformel.
Beweis.
(1) folgt aus Lemmas 4.3 und 4.4

9:26 bis 9:38

Ausdrucksstärke von CTL versus LTL

Lemma 4.4

Sei ζ eine CTL-Zustandsformel und ζ' die LTL-Formel, die man durch Entfernen aller Pfadquantoren aus ζ erhält. Dann gilt:

$\zeta \equiv \zeta'$ oder es gibt keine zu ζ äquivalente LTL-Formel.

Ohne Beweis. (Clarke, Draghicescu 1988)

Lemma 4.5

(1) Es gibt keine zu *AFAGp* äquivalente LTL-Formel.

(2) Es gibt keine zu *FGp* äquivalente CTL-Zustandsformel.

Beweis.

(1) folgt aus Lemmas 4.3 und 4.4

(2) siehe Tafel **T 4.3** □

Ausdrucksstärke von CTL versus LTL

Lemma 4.4
Sei ζ eine CTL-Zustandsformel und ζ' die LTL-Formel, die man durch Entfernen aller Pfadquantoren aus ζ erhält. Dann gilt:
$\zeta \equiv \zeta'$ oder es gibt keine zu ζ äquivalente LTL-Formel.
Ohne Beweis. (Clarke, Draghicescu 1988)
Lemma 4.5
(1) Es gibt keine zu *AFAGp* äquivalente LTL-Formel.
(2) Es gibt keine zu *FGp* äquivalente CTL-Zustandsformel.
Beweis.
(1) folgt aus Lemmas 4.3 und 4.4
(2) siehe Tafel T 4.3 □

9:26 bis 9:38

Ausdrucksstärke von CTL versus LTL

Auch **progress properties** sind **nicht** in LTL ausdrückbar:

Lemma 4.6

Sei $\zeta = AG(p \to EFp')$. Es gibt keine zu ζ äquivalente LTL-Formel.

Ausdrucksstärke von CTL versus LTL
Auch **progress properties** sind **nicht** in LTL ausdrückbar:
Lemma 4.6
Sei $\zeta = AG(p \to EFp')$. Es gibt keine zu ζ äquivalente LTL-Formel.

9:38

Ausdrucksstärke von CTL versus LTL

Auch **progress properties** sind **nicht** in LTL ausdrückbar:

Lemma 4.6

Sei $\zeta = AG(p \rightarrow EFp')$. Es gibt keine zu ζ äquivalente LTL-Formel.

Beweis. Angenommen, es gebe LTL-Formel $\varphi \equiv \zeta$.

9:38

Ausdrucksstärke von CTL versus LTL
Auch progress properties sind nicht in LTL ausdrückbar:
Lemma 4.6
Sei $\zeta = AG(p \rightarrow EFp')$. Es gibt keine zu ζ äquivalente LTL-Formel.
Beweis. Angenommen, es gebe LTL-Formel $\varphi \equiv \zeta$.

Ausdrucksstärke von CTL versus LTL

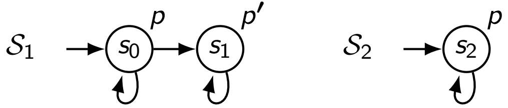
Auch **progress properties** sind **nicht** in LTL ausdrückbar:

Lemma 4.6

Sei $\zeta = AG(p \rightarrow EFp')$. Es gibt keine zu ζ äquivalente LTL-Formel.

Beweis. Angenommen, es gebe LTL-Formel $\varphi \equiv \zeta$.

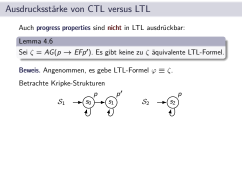
Betrachte Kripke-Strukturen



2019-02-01

Teil 4: unendliche Bäume
└*Model-Checking mit CTL*

└Ausdrucksstärke von CTL versus LTL


Ausdrucksstärke von CTL versus LTL
Auch progress properties sind nicht in LTL ausdrückbar:
Lemma 4.6
Sei $\zeta = AG(p \rightarrow EFp')$. Es gibt keine zu ζ äquivalente LTL-Formel.
Beweis. Angenommen, es gebe LTL-Formel $\varphi \equiv \zeta$.
Betrachte Kripke-Strukturen

9:38

Ausdrucksstärke von CTL versus LTL

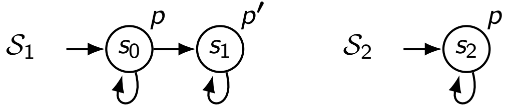
Auch **progress properties** sind **nicht** in LTL ausdrückbar:

Lemma 4.6

Sei $\zeta = AG(p \to EFp')$. Es gibt keine zu ζ äquivalente LTL-Formel.

Beweis. Angenommen, es gebe LTL-Formel $\varphi \equiv \zeta$.

Betrachte Kripke-Strukturen



Dann gilt $\mathcal{S}_1 \models \zeta$.

Ausdrucksstärke von CTL versus LTL
Auch progress properties sind nicht in LTL ausdrückbar:
Lemma 4.6
Sei $\zeta = AG(p \to EFp')$. Es gibt keine zu ζ äquivalente LTL-Formel.
Beweis. Angenommen, es gebe LTL-Formel $\varphi \equiv \zeta$.
Betrachte Kripke-Strukturen
Dann gilt $\mathcal{S}_1 \models \zeta$.

9:38

Ausdrucksstärke von CTL versus LTL

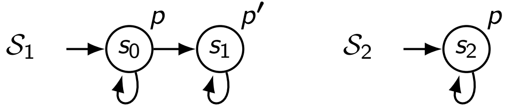
Auch **progress properties** sind **nicht** in LTL ausdrückbar:

Lemma 4.6

Sei $\zeta = AG(p \to EFp')$. Es gibt keine zu ζ äquivalente LTL-Formel.

Beweis. Angenommen, es gebe LTL-Formel $\varphi \equiv \zeta$.

Betrachte Kripke-Strukturen



Dann gilt $\mathcal{S}_1 \models \zeta$.

Also auch $\mathcal{S}_1 \models \varphi$.

Ausdrucksstärke von CTL versus LTL
Auch progress properties sind nicht in LTL ausdrückbar:
Lemma 4.6
Sei $\zeta = AG(p \to EFp')$. Es gibt keine zu ζ äquivalente LTL-Formel.
Beweis. Angenommen, es gebe LTL-Formel $\varphi \equiv \zeta$.
Betrachte Kripke-Strukturen
Dann gilt $\mathcal{S}_1 \models \zeta$.
Also auch $\mathcal{S}_1 \models \varphi$.

9:38

Ausdrucksstärke von CTL versus LTL

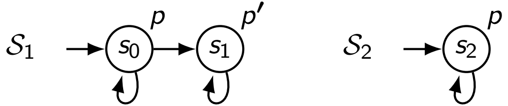
Auch **progress properties** sind **nicht** in LTL ausdrückbar:

Lemma 4.6

Sei $\zeta = AG(p \to EFp')$. Es gibt keine zu ζ äquivalente LTL-Formel.

Beweis. Angenommen, es gebe LTL-Formel $\varphi \equiv \zeta$.

Betrachte Kripke-Strukturen



Dann gilt $\mathcal{S}_1 \models \zeta$.

Also auch $\mathcal{S}_1 \models \varphi$.

Da Paths$(s_2) \subseteq$ Paths(s_0), gilt auch $\mathcal{S}_2 \models \varphi$.

Ausdrucksstärke von CTL versus LTL
Auch progress properties sind nicht in LTL ausdrückbar:
Lemma 4.6
Sei $\zeta = AG(p \to EFp')$. Es gibt keine zu ζ äquivalente LTL-Formel.
Beweis. Angenommen, es gebe LTL-Formel $\varphi \equiv \zeta$.
Betrachte Kripke-Strukturen
Dann gilt $\mathcal{S}_1 \models \zeta$.
Also auch $\mathcal{S}_1 \models \varphi$.
Da Paths$(s_2) \subseteq$ Paths(s_0), gilt auch $\mathcal{S}_2 \models \varphi$.

9:38

Ausdrucksstärke von CTL versus LTL

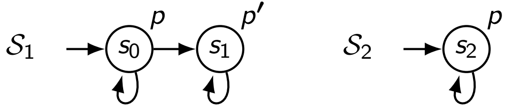
Auch **progress properties** sind **nicht** in LTL ausdrückbar:

Lemma 4.6

Sei $\zeta = AG(p \to EFp')$. Es gibt keine zu ζ äquivalente LTL-Formel.

Beweis. Angenommen, es gebe LTL-Formel $\varphi \equiv \zeta$.

Betrachte Kripke-Strukturen



Dann gilt $\mathcal{S}_1 \models \zeta$.

Also auch $\mathcal{S}_1 \models \varphi$.

Da $\mathrm{Paths}(s_2) \subseteq \mathrm{Paths}(s_0)$, gilt auch $\mathcal{S}_2 \models \varphi$.

Aber offensichtlich $\mathcal{S}_2 \not\models \zeta$. ↯ □

Ausdrucksstärke von CTL versus LTL
Auch progress properties sind nicht in LTL ausdrückbar:
Lemma 4.6
Sei $\zeta = AG(p \to EFp')$. Es gibt keine zu ζ äquivalente LTL-Formel.
Beweis. Angenommen, es gebe LTL-Formel $\varphi \equiv \zeta$.
Betrachte Kripke-Strukturen
Dann gilt $\mathcal{S}_1 \models \zeta$.
Also auch $\mathcal{S}_1 \models \varphi$.
Da $\mathrm{Paths}(s_2) \subseteq \mathrm{Paths}(s_0)$, gilt auch $\mathcal{S}_2 \models \varphi$.
Aber offensichtlich $\mathcal{S}_2 \not\models \zeta$. ↯ □

9:38

Ausdrucksstärke von CTL versus LTL

Erweiterung von LTL und CTL: **CTL***
CTL*: 1986 von E. A. Emerson und J. Y. Halpern (*1953, Inform., Cornell)

9:42

Ausdrucksstärke von CTL versus LTL
Erweiterung von LTL und CTL: **CTL***
CTL*: 1986 von E. A. Emerson und J. Y. Halpern (*1953, Inform., Cornell)

Model-Checking für CTL (Skizze)

Standard-Algorithmus („bottom-up labelling“, ohne Automaten):

Eingabe: Kripke-Str. $\mathcal{S}$, Zust. s_0, CTL-Zustandsformel ζ
Frage: $s_0 \models \zeta$?

Vorgehen:

Model-Checking für CTL (Skizze)
Standard-Algorithmus („bottom-up labelling“, ohne Automaten):
Eingabe: Kripke-Str. $\mathcal{S}$, Zust. s_0, CTL-Zustandsformel ζ
Frage: $s_0 \models \zeta$?
Vorgehen:

9:42 bis 9:53

Model-Checking für CTL (Skizze)

Standard-Algorithmus („bottom-up labelling“, ohne Automaten):

Eingabe: Kripke-Str. $\mathcal{S}$, Zust. s_0, CTL-Zustandsformel ζ
Frage: $s_0 \models \zeta$?

Vorgehen:

- Stelle ζ als Baum dar (Bsp. siehe Tafel) **T 4.4**

Model-Checking für CTL (Skizze)
Standard-Algorithmus („bottom-up labelling“, ohne Automaten):
Eingabe: Kripke-Str. $\mathcal{S}$, Zust. s_0, CTL-Zustandsformel ζ
Frage: $s_0 \models \zeta$?
Vorgehen:
Stelle ζ als Baum dar (Bsp. siehe Tafel) T 4.4

9:42 bis 9:53

Model-Checking für CTL (Skizze)

Standard-Algorithmus („bottom-up labelling“, ohne Automaten):

Eingabe: Kripke-Str. $\mathcal{S}$, Zust. s_0, CTL-Zustandsformel ζ
Frage: $s_0 \models \zeta$?

Vorgehen:

- Stelle ζ als Baum dar (Bsp. siehe Tafel) **T 4.4**
- Gehe Baum von unten nach oben durch
  und markiere Zustände s in $\mathcal{S}$ mit der jeweiligen Teilformel,
  wenn sie in s erfüllt ist **T 4.4 Forts.**

Model-Checking für CTL (Skizze)

Standard-Algorithmus („bottom-up labelling“, ohne Automaten):

Eingabe: Kripke-Str. $\mathcal{S}$, Zust. s_0, CTL-Zustandsformel ζ
Frage: $s_0 \models \zeta$?

Vorgehen:

- Stelle ζ als Baum dar (Bsp. siehe Tafel) **T 4.4**
- Gehe Baum von unten nach oben durch und markiere Zustände s in $\mathcal{S}$ mit der jeweiligen Teilformel, wenn sie in s erfüllt ist **T 4.4 Forts.**

9:42 bis 9:53

Model-Checking für CTL (Skizze)

Standard-Algorithmus („bottom-up labelling“, ohne Automaten):

Eingabe: Kripke-Str. $\mathcal{S}$, Zust. s_0, CTL-Zustandsformel ζ
Frage: $s_0 \models \zeta$?

Vorgehen:

- Stelle ζ als Baum dar (Bsp. siehe Tafel) **T 4.4**
- Gehe Baum von unten nach oben durch
  und markiere Zustände s in $\mathcal{S}$ mit der jeweiligen Teilformel,
  wenn sie in s erfüllt ist **T 4.4 Forts.**
- Akzeptiere gdw. s_0 mit ζ markiert ist

Model-Checking für CTL (Skizze)
Standard-Algorithmus („bottom-up labelling“, ohne Automaten):
Eingabe: Kripke-Str. $\mathcal{S}$, Zust. s_0, CTL-Zustandsformel ζ
Frage: $s_0 \models \zeta$?
Vorgehen:
- Stelle ζ als Baum dar (Bsp. siehe Tafel) **T 4.4**
- Gehe Baum von unten nach oben durch und markiere Zustände s in $\mathcal{S}$ mit der jeweiligen Teilformel, wenn sie in s erfüllt ist **T 4.4 Forts.**
- Akzeptiere gdw. s_0 mit ζ markiert ist

9:42 bis 9:53

Model-Checking für CTL (Skizze)

Standard-Algorithmus („bottom-up labelling“, ohne Automaten):

Eingabe: Kripke-Str. $\mathcal{S}$, Zust. s_0, CTL-Zustandsformel ζ
Frage: $s_0 \models \zeta$?

Vorgehen:

- Stelle ζ als Baum dar (Bsp. siehe Tafel) **T 4.4**
- Gehe Baum von unten nach oben durch
  und markiere Zustände s in $\mathcal{S}$ mit der jeweiligen Teilformel,
  wenn sie in s erfüllt ist **T 4.4 Forts.**
- Akzeptiere gdw. s_0 mit ζ markiert ist

Komplexität: P-vollständig (LTL-MC: **PSpace**-vollständig)

Dafür ist CTL-SAT **ExpTime**-vollständig (LTL-SAT: **PSpace**-vollst.).

Model-Checking für CTL (Skizze)

Standard-Algorithmus („bottom-up labelling“, ohne Automaten):

Eingabe: Kripke-Str. $\mathcal{S}$, Zust. s_0, CTL-Zustandsformel ζ
Frage: $s_0 \models \zeta$?

Vorgehen:

- Stelle ζ als Baum dar (Bsp. siehe Tafel) **T 4.4**
- Gehe Baum von unten nach oben durch
  und markiere Zustände s in $\mathcal{S}$ mit der jeweiligen Teilformel,
  wenn sie in s erfüllt ist **T 4.4 Forts.**
- Akzeptiere gdw. s_0 mit ζ markiert ist

Komplexität: P-vollständig (LTL-MC: **PSpace**-vollständig)

Dafür ist CTL-SAT **ExpTime**-vollständig (LTL-SAT: **PSpace**-vollst.).

9:42 bis 9:53

Model-Checking für CTL mit Baumautomaten

Automatenbasierte Entscheidungsprozedur für CTL

... basiert auf **alternierenden Baumautomaten**

(Erweiterung des Begriffs der nichtdeterminist. Baumautomaten; siehe Teil 5 der Vorlesung)

Model-Checking für CTL mit Baumautomaten
Automatenbasierte Entscheidungsprozedur für CTL
... basiert auf alternierenden Baumautomaten
(Erweiterung des Begriffs der nichtdeterminist. Baumautomaten; siehe Teil 5 der Vorlesung)

9:53 bis 9:55, 5 min Reserve.

CTL*-MC ist PSpace-vollst., CTL*-SAT 2ExpTime-vollst.
Siehe Baier & Katoen S. 430.

Model-Checking für CTL mit Baumautomaten

Automatenbasierte Entscheidungsprozedur für CTL

. . . basiert auf **alternierenden Baumautomaten**

(Erweiterung des Begriffs der nichtdeterminist. Baumautomaten; siehe Teil 5 der Vorlesung)

Verwandt:

Automatenbasierte Entscheidungsprozedur für CTL*-**Erfüllbarkeit**

- basiert auf nichtdeterministischen Rabin-Baumautomaten
- technisch aufwändige Konstruktion
- hier nicht behandelt

Model-Checking für CTL mit Baumautomaten
Automatenbasierte Entscheidungsprozedur für CTL
. . . basiert auf alternierenden Baumautomaten
(Erweiterung des Begriffs der nichtdeterminist. Baumautomaten; siehe Teil 5 der Vorlesung)
Verwandt:
Automatenbasierte Entscheidungsprozedur für CTL*-Erfüllbarkeit
basiert auf nichtdeterministischen Rabin-Baumautomaten
technisch aufwändige Konstruktion
hier nicht behandelt

9:53 bis 9:55, 5 min Reserve.

CTL*-MC ist PSpace-vollst., CTL*-SAT 2ExpTime-vollst.
Siehe Baier & Katoen S. 430.

Model-Checking für CTL mit Baumautomaten

Automatenbasierte Entscheidungsprozedur für CTL

... basiert auf **alternierenden Baumautomaten**

(Erweiterung des Begriffs der nichtdeterminist. Baumautomaten; siehe Teil 5 der Vorlesung)

Verwandt:

Automatenbasierte Entscheidungsprozedur für CTL*-**Erfüllbarkeit**

- basiert auf nichtdeterministischen Rabin-Baumautomaten
- technisch aufwändige Konstruktion
- hier nicht behandelt

Es folgt:

Überblick „klassische“ nichdeterministische Baumautomaten

Model-Checking für CTL mit Baumautomaten

Automatenbasierte Entscheidungsprozedur für CTL
... basiert auf alternierenden Baumautomaten
(Erweiterung des Begriffs der nichtdeterminist. Baumautomaten; siehe Teil 5 der Vorlesung)
Verwandt:
Automatenbasierte Entscheidungsprozedur für CTL*-Erfüllbarkeit
- basiert auf nichtdeterministischen Rabin-Baumautomaten
- technisch aufwändige Konstruktion
- hier nicht behandelt
Es folgt:
Überblick „klassische“ nichdeterministische Baumautomaten

9:53 bis 9:55, 5 min Reserve.

CTL*-MC ist PSpace-vollst., CTL*-SAT 2ExpTime-vollst.
Siehe Baier & Katoen S. 430.

Und nun ...

1. *Model-Checking mit CTL*
2. Automaten auf unendlichen Bäumen
3. Komplementierung

Und nun ...

1. Model-Checking mit CTL
2. Automaten auf unendlichen Bäumen
3. Komplementierung

Baumautomaten: Grundbegriffe

Betrachten **unendlichen vollständigen Binärbaum**

- Positionen: *alle* Wörter aus $\{0, 1\}^*$
- jeder Knoten p hat linkes und rechtes Kind: $p0, p1$
- **Tiefe** von Knoten p: $|p|$
- **Ebene** k: alle Knoten der Tiefe k
- p_2 ist **Nachfolger** von p_1, geschrieben $p_1 \sqsubseteq p_2$, wenn $p_2 = p_1 p$ für ein $p \in \{0, 1\}^*$ **T 4.5**

Baumautomaten: Grundbegriffe
Betrachten unendlichen vollständigen Binärbaum
Positionen: alle Wörter aus $\{0, 1\}^*$
jeder Knoten p hat linkes und rechtes Kind: $p0, p1$
Tiefe von Knoten p: $|p|$
Ebene k: alle Knoten der Tiefe k
p_2 ist Nachfolger von p_1, geschrieben $p_1 \sqsubseteq p_2$, wenn $p_2 = p_1 p$ für ein $p \in \{0, 1\}^*$ T 4.5

16:00

Baumautomaten: Grundbegriffe

Betrachten **unendlichen vollständigen Binärbaum**

- Positionen: *alle* Wörter aus $\{0,1\}^*$
- jeder Knoten p hat linkes und rechtes Kind: $p0, p1$
- **Tiefe** von Knoten p: $|p|$
- **Ebene** k: alle Knoten der Tiefe k
- p_2 ist **Nachfolger** von p_1, geschrieben $p_1 \sqsubseteq p_2$, wenn $p_2 = p_1 p$ für ein $p \in \{0,1\}^*$ **T 4.5**

Pfad: Teilmenge $\pi \subseteq \{0,1\}^*$ mit $\varepsilon \in \pi$ und:

- wenn $p \in \pi$, dann genau eins der Kinder $p0, p1$ in π
- $\forall k$: von allen Knoten der Ebene k ist genau einer in π

T 4.5 Forts.

Baumautomaten: Grundbegriffe

Betrachten **unendlichen vollständigen Binärbaum**

- Positionen: *alle* Wörter aus $\{0,1\}^*$
- jeder Knoten p hat linkes und rechtes Kind: $p0, p1$
- **Tiefe** von Knoten p: $|p|$
- **Ebene** k: alle Knoten der Tiefe k
- p_2 ist **Nachfolger** von p_1, geschrieben $p_1 \sqsubseteq p_2$, wenn $p_2 = p_1 p$ für ein $p \in \{0,1\}^*$ **T 4.5**

Pfad: Teilmenge $\pi \subseteq \{0,1\}^*$ mit $\varepsilon \in \pi$ und:

- wenn $p \in \pi$, dann genau eins der Kinder $p0, p1$ in π
- $\forall k$: von allen Knoten der Ebene k ist genau einer in π

T 4.5 Forts.

16:00

Baumautomaten: Grundbegriffe

Betrachten **unendlichen vollständigen Binärbaum**

- Positionen: *alle* Wörter aus $\{0,1\}^*$
- jeder Knoten p hat linkes und rechtes Kind: $p0, p1$
- **Tiefe** von Knoten p: $|p|$
- **Ebene** k: alle Knoten der Tiefe k
- p_2 ist **Nachfolger** von p_1, geschrieben $p_1 \sqsubseteq p_2$, wenn $p_2 = p_1 p$ für ein $p \in \{0,1\}^*$ **T 4.5**

Pfad: Teilmenge $\pi \subseteq \{0,1\}^*$ mit $\varepsilon \in \pi$ und:

- wenn $p \in \pi$, dann genau eins der Kinder $p0, p1$ in π
- $\forall k$: von allen Knoten der Ebene k ist genau einer in π

T 4.5 Forts.

Σ-Baum t (Alphabet Σ ohne Stelligkeit):

Funktion $t : \{0,1\}^* \to \Sigma$ **T 4.5 Forts.**

Baumautomaten: Grundbegriffe

Betrachten **unendlichen vollständigen Binärbaum**

- Positionen: *alle* Wörter aus $\{0,1\}^*$
- jeder Knoten p hat linkes und rechtes Kind: $p0, p1$
- **Tiefe** von Knoten p: $|p|$
- **Ebene** k: alle Knoten der Tiefe k
- p_2 ist **Nachfolger** von p_1, geschrieben $p_1 \sqsubseteq p_2$, wenn $p_2 = p_1 p$ für ein $p \in \{0,1\}^*$ T 4.5

Pfad: Teilmenge $\pi \subseteq \{0,1\}^*$ mit $\varepsilon \in \pi$ und:

- wenn $p \in \pi$, dann genau eins der Kinder $p0, p1$ in π
- $\forall k$: von allen Knoten der Ebene k ist genau einer in π

T 4.5 Forts.

Σ-Baum t (Alphabet Σ ohne Stelligkeit):

Funktion $t : \{0,1\}^* \to \Sigma$ T 4.5 Forts.

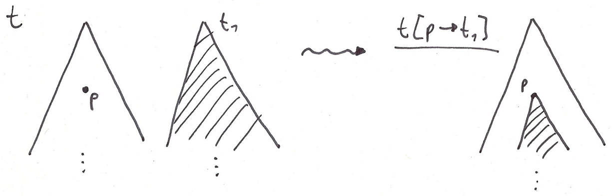
16:00

Baumautomaten: etwas mehr Notation (1)

$\hat{t} = t[p \to t_1]$:

der Baum, den man aus t erhält, wenn man den Teilbaum, der in p wurzelt, durch t_1 ersetzt

Skizze:



exakte Beschreibung:

$$\hat{t}(p') = \begin{cases} t_1(p'') & \text{wenn } p' = pp'' \\ t(p') & \text{wenn } p \not\sqsubseteq p' \end{cases}$$

Baumautomaten: etwas mehr Notation (1)

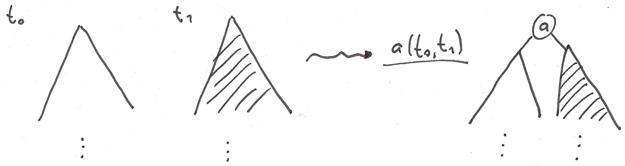
16:08

Baumautomaten: etwas mehr Notation (2)

$\hat{t} = \boldsymbol{a(t_0, t_1)}$:

der Baum mit Wurzel a und Teilbäumen t_0, t_1 in den Wurzelkindern $0, 1$:

Skizze:



exakte Beschreibung:

$$\hat{t}(p) = \begin{cases} a & \text{wenn } p = \varepsilon \\ t_0(p') & \text{wenn } p = 0p' \\ t_1(p') & \text{wenn } p = 1p' \end{cases}$$

Baumautomaten: etwas mehr Notation (2)

16:11

Baumautomaten: Definition

Definition 4.7

Ein **nichtdeterministischer Büchi-Baumautomat (NBBA)** über Σ ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$, wobei

- Q eine endliche nichtleere **Zustandsmenge** ist,
- Σ ein Alphabet ist
- $\Delta \subseteq Q \times \Sigma \times Q \times Q$ die **Überführungsrelation** ist,
- $I \subseteq Q$ die Menge der **Anfangszustände** ist,
- $F \subseteq Q$ die Menge der **akzeptierenden Zustände** ist.

Baumautomaten: Definition

Definition 4.7
Ein nichtdeterministischer Büchi-Baumautomat (NBBA) über Σ ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$, wobei
- Q eine endliche nichtleere Zustandsmenge ist,
- Σ ein Alphabet ist
- $\Delta \subseteq Q \times \Sigma \times Q \times Q$ die Überführungsrelation ist,
- $I \subseteq Q$ die Menge der Anfangszustände ist,
- $F \subseteq Q$ die Menge der akzeptierenden Zustände ist.

16:14

Baumautomaten: Definition

Definition 4.7

Ein **nichtdeterministischer Büchi-Baumautomat (NBBA)** über Σ ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$, wobei

- Q eine endliche nichtleere **Zustandsmenge** ist,
- Σ ein Alphabet ist
- $\Delta \subseteq Q \times \Sigma \times Q \times Q$ die **Überführungsrelation** ist,
- $I \subseteq Q$ die Menge der **Anfangszustände** ist,
- $F \subseteq Q$ die Menge der **akzeptierenden Zustände** ist.

(entsprechen offenbar Top-down-Automaten)

Baumautomaten: Definition

Definition 4.7

Ein nichtdeterministischer Büchi-Baumautomat (NBBA) über Σ ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$, wobei

- Q eine endliche nichtleere Zustandsmenge ist,
- Σ ein Alphabet ist
- $\Delta \subseteq Q \times \Sigma \times Q \times Q$ die Überführungsrelation ist,
- $I \subseteq Q$ die Menge der Anfangszustände ist,
- $F \subseteq Q$ die Menge der akzeptierenden Zustände ist.

(entsprechen offenbar Top-down-Automaten)

16:14

Muller- und Paritäts-Baumautomaten

Definition 4.8

Ein **nichtdeterministischer Muller-Baumautomat (NMBA)** über Σ ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$, wobei

- Q, Σ, Δ, I wie für NBBAs sind
- $\mathcal{F} \subseteq 2^Q$ die **Akzeptanzkomponente** ist

Muller- und Paritäts-Baumautomaten
Definition 4.8
Ein nichtdeterministischer Muller-Baumautomat (NMBA) über Σ ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$, wobei
- Q, Σ, Δ, I wie für NBBAs sind
- $\mathcal{F} \subseteq 2^Q$ die Akzeptanzkomponente ist

16:15

Muller- und Paritäts-Baumautomaten

Definition 4.8

Ein **nichtdeterministischer Muller-Baumautomat (NMBA)** über Σ ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$, wobei

- Q, Σ, Δ, I wie für NBBAs sind
- $\mathcal{F} \subseteq 2^Q$ die **Akzeptanzkomponente** ist

Ein **nichtdeterministischer Paritäts-Baumautomat (NPBA)** über Σ ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \Delta, I, c)$, wobei

- Q, Σ, Δ, I wie für NBBAs sind
- $c : Q \to \mathbb{N}$ die **Akzeptanzkomponente** ist

Muller- und Paritäts-Baumautomaten

Definition 4.8

Ein nichtdeterministischer Muller-Baumautomat (NMBA) über Σ ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$, wobei

- Q, Σ, Δ, I wie für NBBAs sind
- $\mathcal{F} \subseteq 2^Q$ die Akzeptanzkomponente ist

Ein nichtdeterministischer Paritäts-Baumautomat (NPBA) über Σ ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \Delta, I, c)$, wobei

- Q, Σ, Δ, I wie für NBBAs sind
- $c : Q \to \mathbb{N}$ die Akzeptanzkomponente ist

16:15

Muller- und Paritäts-Baumautomaten

Definition 4.8

Ein **nichtdeterministischer Muller-Baumautomat (NMBA)** über Σ ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$, wobei

- Q, Σ, Δ, I wie für NBBAs sind
- $\mathcal{F} \subseteq 2^Q$ die **Akzeptanzkomponente** ist

Ein **nichtdeterministischer Paritäts-Baumautomat (NPBA)** über Σ ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \Delta, I, c)$, wobei

- Q, Σ, Δ, I wie für NBBAs sind
- $c : Q \to \mathbb{N}$ die **Akzeptanzkomponente** ist

(Rabin- und Streett-Baumautomaten wie üblich definiert)

Muller- und Paritäts-Baumautomaten
Definition 4.8
Ein nichtdeterministischer Muller-Baumautomat (NMBA) über Σ ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$, wobei
Q, Σ, Δ, I wie für NBBAs sind
$\mathcal{F} \subseteq 2^Q$ die Akzeptanzkomponente ist
Ein nichtdeterministischer Paritäts-Baumautomat (NPBA) über Σ ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \Delta, I, c)$, wobei
Q, Σ, Δ, I wie für NBBAs sind
$c : Q \to \mathbb{N}$ die Akzeptanzkomponente ist
(Rabin- und Streett-Baumautomaten wie üblich definiert)

16:15

Runs auf Baumautomaten

Run = Markierung der Positionen in $\{0,1\}^*$ mit Zuständen, verträglich mit Anfangszuständen und Überführungsrelation

16:16

Runs auf Baumautomaten

Run = Markierung der Positionen in $\{0,1\}^*$ mit Zuständen, verträglich mit Anfangszuständen und Überführungsrelation

Runs auf Baumautomaten

Run = Markierung der Positionen in $\{0,1\}^*$ mit Zuständen, verträglich mit Anfangszuständen und Überführungsrelation

Definition 4.9

Ein **Run** eines NBBA (NMBA, NPBA) $\mathcal{A}$ auf einem Σ-Baum t ist eine Funktion $r : \{0,1\}^* \to Q$, so dass

- $r(\varepsilon) \in I$;
- für alle $p \in \{0,1\}^*$ gilt: $(r(p), t(p), r(p0), r(p1)) \in \Delta$

Runs auf Baumautomaten

Run = Markierung der Positionen in $\{0,1\}^*$ mit Zuständen, verträglich mit Anfangszuständen und Überführungsrelation

Definition 4.9

Ein Run eines NBBA (NMBA, NPBA) $\mathcal{A}$ auf einem Σ-Baum t ist eine Funktion $r : \{0,1\}^* \to Q$, so dass

- $r(\varepsilon) \in I$;
- für alle $p \in \{0,1\}^*$ gilt: $(r(p), t(p), r(p0), r(p1)) \in \Delta$

16:16

Runs auf Baumautomaten

Run = Markierung der Positionen in $\{0,1\}^*$ mit Zuständen, verträglich mit Anfangszuständen und Überführungsrelation

Definition 4.9

Ein **Run** eines NBBA (NMBA, NPBA) $\mathcal{A}$ auf einem Σ-Baum t ist eine Funktion $r : \{0,1\}^* \to Q$, so dass

- $r(\varepsilon) \in I$;
- für alle $p \in \{0,1\}^*$ gilt: $(r(p), t(p), r(p0), r(p1)) \in \Delta$

Erfolgreicher Run: verträglich mit Akzeptanzkomponente

Runs auf Baumautomaten

Run = Markierung der Positionen in $\{0,1\}^*$ mit Zuständen, verträglich mit Anfangszuständen und Überführungsrelation

Definition 4.9

Ein **Run** eines NBBA (NMBA, NPBA) $\mathcal{A}$ auf einem Σ-Baum t ist eine Funktion $r : \{0,1\}^* \to Q$, so dass

- $r(\varepsilon) \in I$;
- für alle $p \in \{0,1\}^*$ gilt: $(r(p), t(p), r(p0), r(p1)) \in \Delta$

Erfolgreicher Run: verträglich mit Akzeptanzkomponente

16:16

Erfolgreiche Runs

Sei r Run eines NxBAs $\mathcal{A}$ und π ein Pfad

Betrachten wieder **Unendlichkeitsmenge**

$$\text{Inf}(r, \pi) = \{q \in Q \mid r(p) = q \text{ für unendlich viele } p \in \pi\}$$

Erfolgreiche Runs

Sei r Run eines NxBAs $\mathcal{A}$ und π ein Pfad

Betrachten wieder **Unendlichkeitsmenge**

$\text{Inf}(r, \pi) = \{q \in Q \mid r(p) = q \text{ für unendlich viele } p \in \pi\}$

16:18

Parity-Bedingung: in Aufgabe 4 auf Übungsblatt 4 war es max statt min – macht aber keinen Unterschied; ggf. Nummerierung der Zustände „umdrehen“.

Erfolgreiche Runs

Sei r Run eines NxBAs $\mathcal{A}$ und π ein Pfad

Betrachten wieder **Unendlichkeitsmenge**

$$\mathrm{Inf}(r, \pi) = \{q \in Q \mid r(p) = q \text{ für unendlich viele } p \in \pi\}$$

Definition 4.10

- Run r des NBBA $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ ist **erfolgreich**, falls **für alle Pfade** π gilt: $\mathrm{Inf}(r, \pi) \cap F \neq \emptyset$

Erfolgreiche Runs

Sei r Run eines NxBAs $\mathcal{A}$ und π ein Pfad
Betrachten wieder **Unendlichkeitsmenge**

$\mathrm{Inf}(r, \pi) = \{q \in Q \mid r(p) = q \text{ für unendlich viele } p \in \pi\}$

Definition 4.10
- Run r des NBBA $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ ist **erfolgreich**, falls **für alle Pfade** π gilt: $\mathrm{Inf}(r, \pi) \cap F \neq \emptyset$

16:18

Parity-Bedingung: in Aufgabe 4 auf Übungsblatt 4 war es max statt min – macht aber keinen Unterschied; ggf. Nummerierung der Zustände „umdrehen“.

Erfolgreiche Runs

Sei r Run eines NxBAs $\mathcal{A}$ und π ein Pfad

Betrachten wieder **Unendlichkeitsmenge**

$$\mathrm{Inf}(r, \pi) = \{q \in Q \mid r(p) = q \text{ für unendlich viele } p \in \pi\}$$

Definition 4.10

- Run r des NBBA $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ ist **erfolgreich**, falls **für alle Pfade** π gilt: $\mathrm{Inf}(r, \pi) \cap F \neq \emptyset$
- Run r des NMBA $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$ ist **erfolgreich**, falls **für alle Pfade** π gilt: $\mathrm{Inf}(r, \pi) \in \mathcal{F}$

Erfolgreiche Runs

Sei r Run eines NxBAs $\mathcal{A}$ und π ein Pfad
Betrachten wieder Unendlichkeitsmenge
$\mathrm{Inf}(r, \pi) = \{q \in Q \mid r(p) = q \text{ für unendlich viele } p \in \pi\}$
Definition 4.10
Run r des NBBA $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ ist erfolgreich, falls für alle Pfade π gilt: $\mathrm{Inf}(r, \pi) \cap F \neq \emptyset$
Run r des NMBA $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$ ist erfolgreich, falls für alle Pfade π gilt: $\mathrm{Inf}(r, \pi) \in \mathcal{F}$

16:18

Parity-Bedingung: in Aufgabe 4 auf Übungsblatt 4 war es max statt min – macht aber keinen Unterschied; ggf. Nummerierung der Zustände „umdrehen“.

Erfolgreiche Runs

Sei r Run eines NxBAs $\mathcal{A}$ und π ein Pfad

Betrachten wieder **Unendlichkeitsmenge**

$$\mathrm{Inf}(r,\pi) = \{q \in Q \mid r(p) = q \text{ für unendlich viele } p \in \pi\}$$

Definition 4.10

- Run r des NBBA $\mathcal{A} = (Q,\Sigma,\Delta,I,F)$ ist **erfolgreich**, falls **für alle Pfade** π gilt: $\mathrm{Inf}(r,\pi) \cap F \neq \emptyset$
- Run r des NMBA $\mathcal{A} = (Q,\Sigma,\Delta,I,\mathcal{F})$ ist **erfolgreich**, falls **für alle Pfade** π gilt: $\mathrm{Inf}(r,\pi) \in \mathcal{F}$
- Run r des NPBA $\mathcal{A} = (Q,\Sigma,\Delta,I,c)$ ist **erfolgreich**, falls **für alle Pfade** π gilt: $\min\{c(q) \mid q \in \mathrm{Inf}(r,\pi)\}$ ist gerade

16:18

Parity-Bedingung: in Aufgabe 4 auf Übungsblatt 4 war es max statt min – macht aber keinen Unterschied; ggf. Nummerierung der Zustände „umdrehen“.

Akzeptanz und erkannte Sprache

... sind wie üblich definiert:

Definition 4.11

Sei $\mathcal{A}$ ein NBBA, NMBA oder NPBA,
sei t ein Σ-Baum und L eine Menge von Σ-Bäumen.

- $\mathcal{A}$ **akzeptiert** t,
  wenn es einen erfolgreichen Run von $\mathcal{A}$ auf t gibt.
- $\boldsymbol{L_\omega(\mathcal{A})} = \{t \mid \mathcal{A} \text{ akzeptiert } t\}$
- L heißt **Büchi-erkennbar**,
  wenn es einen NBBA $\mathcal{A}$ gibt mit $L_\omega(\mathcal{A}) = L$.
- Analog: **Muller-erkennbar** und **paritäts-erkennbar**

16:20

Akzeptanz und erkannte Sprache

... sind wie üblich definiert:

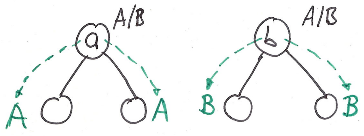
Definition 4.11
Sei $\mathcal{A}$ ein NBBA, NMBA oder NPBA,
sei t ein Σ-Baum und L eine Menge von Σ-Bäumen.
- $\mathcal{A}$ akzeptiert t, wenn es einen erfolgreichen Run von $\mathcal{A}$ auf t gibt.
- $L_\omega(\mathcal{A}) = \{t \mid \mathcal{A} \text{ akzeptiert } t\}$
- L heißt Büchi-erkennbar, wenn es einen NBBA $\mathcal{A}$ gibt mit $L_\omega(\mathcal{A}) = L$.
- Analog: Muller-erkennbar und paritäts-erkennbar

Beispiele (Büchi)

- NBBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = ?$

16:21

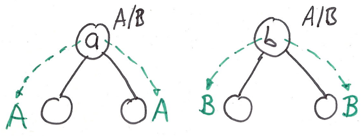
Beispiele (Büchi)
NBBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A\})$ mit
$\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$
Skizze:
$L_\omega(\mathcal{A}) = ?$

Beispiele (Büchi)

- NBBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:



$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } a\text{'s}\}$

Beispiele (Büchi)

- NBBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:

$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } a\text{'s}\}$

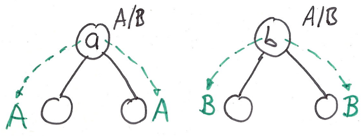
16:21

Beispiele (Büchi)

- NBBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } a\text{'s}\}$

- derselbe NBBA, aber mit $F = \{B\}$
  $L_\omega(\mathcal{A}) = ?$

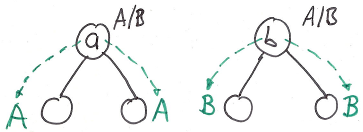
16:21

Beispiele (Büchi)

- NBBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } a\text{'s}\}$

- derselbe NBBA, aber mit $F = \{B\}$
  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } b\text{'s}\}$

Beispiele (Büchi)
NBBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A\})$ mit
$\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$
Skizze:
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } a\text{'s}\}$
derselbe NBBA, aber mit $F = \{B\}$
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } b\text{'s}\}$

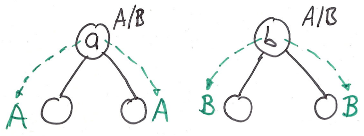
16:21

Beispiele (Büchi)

- NBBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } a\text{'s}\}$

- derselbe NBBA, aber mit $F = \{B\}$
  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } b\text{'s}\}$

- derselbe NBBA, aber mit $F = \{A, B\}$
  $L_\omega(\mathcal{A}) = ?$

Beispiele (Büchi)
NBBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A\})$ mit
$\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$
Skizze:
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } a\text{'s}\}$
derselbe NBBA, aber mit $F = \{B\}$
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } b\text{'s}\}$
derselbe NBBA, aber mit $F = \{A, B\}$
$L_\omega(\mathcal{A}) = ?$

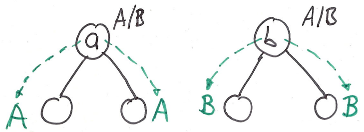
16:21

Beispiele (Büchi)

- NBBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } a\text{'s}\}$

- derselbe NBBA, aber mit $F = \{B\}$

  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } b\text{'s}\}$

- derselbe NBBA, aber mit $F = \{A, B\}$

  $L_\omega(\mathcal{A}) = \{t \mid t \text{ ist ein } \Sigma\text{-Baum}\}$

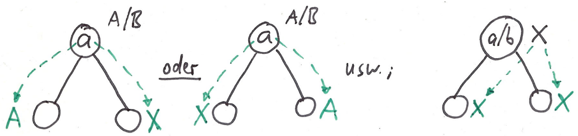
16:21

Beispiele (Büchi)

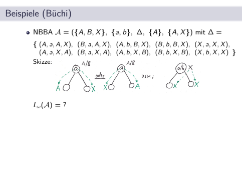
- NBBA $\mathcal{A} = (\{A, B, X\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A, X\})$ mit $\Delta =$

$$\{\ (A,a,A,X),\ (B,a,A,X),\ (A,b,B,X),\ (B,b,B,X),\ (X,a,X,X),\\ (A,a,X,A),\ (B,a,X,A),\ (A,b,X,B),\ (B,b,X,B),\ (X,b,X,X)\ \}$$

Skizze:



$L_\omega(\mathcal{A}) = ?$


Beispiele (Büchi)

16:24

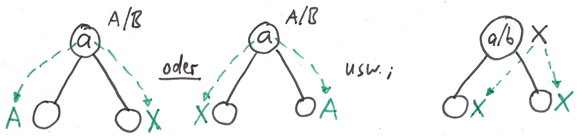
1. Bsp.: lies **spaltenweise!**

Beispiele (Büchi)

- NBBA $\mathcal{A} = (\{A, B, X\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A, X\})$ mit $\Delta =$

$$\{\ (A,a,A,X),\ (B,a,A,X),\ (A,b,B,X),\ (B,b,B,X),\ (X,a,X,X),\ (A,a,X,A),\ (B,a,X,A),\ (A,b,X,B),\ (B,b,X,B),\ (X,b,X,X)\ \}$$

Skizze:



$L_\omega(\mathcal{A}) = \{t \mid t$ hat mind. einen Pfad mit ∞ vielen a's$\}$

16:24

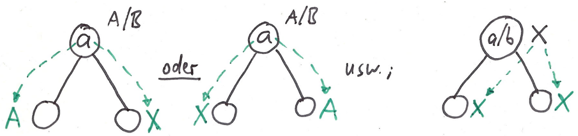
1. Bsp.: lies **spaltenweise!**

Beispiele (Büchi)

- NBBA $\mathcal{A} = (\{A, B, X\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A, X\})$ mit $\Delta =$

  $\{\ (A, a, A, X),\ (B, a, A, X),\ (A, b, B, X),\ (B, b, B, X),\ (X, a, X, X),$
  $(A, a, X, A),\ (B, a, X, A),\ (A, b, X, B),\ (B, b, X, B),\ (X, b, X, X)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid t \text{ hat mind. einen Pfad mit } \infty \text{ vielen } a\text{'s}\}$

- derselbe NBBA, aber mit $F = \{B, X\}$

  $L_\omega(\mathcal{A}) = ?$

16:24

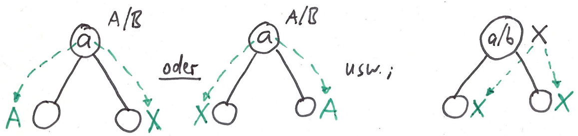
1. Bsp.: lies **spaltenweise!**

Beispiele (Büchi)

- NBBA $\mathcal{A} = (\{A, B, X\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A, X\})$ mit $\Delta =$

  $\{\ (A, a, A, X),\ (B, a, A, X),\ (A, b, B, X),\ (B, b, B, X),\ (X, a, X, X),$
  $(A, a, X, A),\ (B, a, X, A),\ (A, b, X, B),\ (B, b, X, B),\ (X, b, X, X)\ \}$

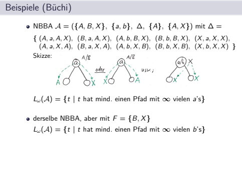
  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid t \text{ hat mind. einen Pfad mit } \infty \text{ vielen } a\text{'s}\}$

- derselbe NBBA, aber mit $F = \{B, X\}$

  $L_\omega(\mathcal{A}) = \{t \mid t \text{ hat mind. einen Pfad mit } \infty \text{ vielen } b\text{'s}\}$



16:24

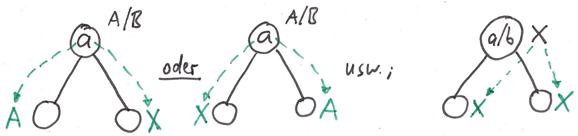
1. Bsp.: lies **spaltenweise!**

Beispiele (Büchi)

- NBBA $\mathcal{A} = (\{A, B, X\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A, X\})$ mit $\Delta =$

  $\{\ (A, a, A, X),\ (B, a, A, X),\ (A, b, B, X),\ (B, b, B, X),\ (X, a, X, X),$
  $(A, a, X, A),\ (B, a, X, A),\ (A, b, X, B),\ (B, b, X, B),\ (X, b, X, X)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid t \text{ hat mind. einen Pfad mit } \infty \text{ vielen } a\text{'s}\}$

- derselbe NBBA, aber mit $F = \{B, X\}$

  $L_\omega(\mathcal{A}) = \{t \mid t \text{ hat mind. einen Pfad mit } \infty \text{ vielen } b\text{'s}\}$

- derselbe NBBA, aber mit $F = \{X\}$

  $L_\omega(\mathcal{A}) = ?$

Beispiele (Büchi)
NBBA $\mathcal{A} = (\{A, B, X\}, \{a, b\}, \Delta, \{A\}, \{A, X\})$ mit $\Delta =$ …
$L_\omega(\mathcal{A}) = \{t \mid t$ hat mind. einen Pfad mit ∞ vielen a's$\}$
derselbe NBBA, aber mit $F = \{B, X\}$
$L_\omega(\mathcal{A}) = \{t \mid t$ hat mind. einen Pfad mit ∞ vielen b's$\}$
derselbe NBBA, aber mit $F = \{X\}$
$L_\omega(\mathcal{A}) = ?$

16:24

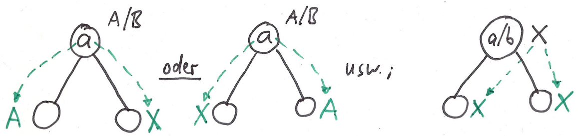
1. Bsp.: lies **spaltenweise!**

Beispiele (Büchi)

- NBBA $\mathcal{A} = (\{A, B, X\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A, X\})$ mit $\Delta =$

  $\{\ (A, a, A, X),\ (B, a, A, X),\ (A, b, B, X),\ (B, b, B, X),\ (X, a, X, X),$
  $(A, a, X, A),\ (B, a, X, A),\ (A, b, X, B),\ (B, b, X, B),\ (X, b, X, X)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid t \text{ hat mind. einen Pfad mit } \infty \text{ vielen } a\text{'s}\}$

- derselbe NBBA, aber mit $F = \{B, X\}$

  $L_\omega(\mathcal{A}) = \{t \mid t \text{ hat mind. einen Pfad mit } \infty \text{ vielen } b\text{'s}\}$

- derselbe NBBA, aber mit $F = \{X\}$

  $L_\omega(\mathcal{A}) = \emptyset$

16:24

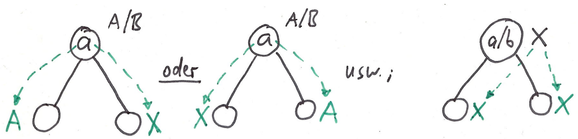
1. Bsp.: lies **spaltenweise!**

Beispiele (Büchi)

- NBBA $\mathcal{A} = (\{A, B, X\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A, X\})$ mit $\Delta =$

  $\{\ (A, a, A, X),\ (B, a, A, X),\ (A, b, B, X),\ (B, b, B, X),\ (X, a, X, X),$
  $(A, a, X, A),\ (B, a, X, A),\ (A, b, X, B),\ (B, b, X, B),\ (X, b, X, X)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid t \text{ hat mind. einen Pfad mit } \infty \text{ vielen } a\text{'s}\}$

- derselbe NBBA, aber mit $F = \{B, X\}$

  $L_\omega(\mathcal{A}) = \{t \mid t \text{ hat mind. einen Pfad mit } \infty \text{ vielen } b\text{'s}\}$

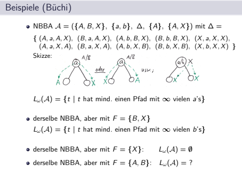
- derselbe NBBA, aber mit $F = \{X\}$: $\quad L_\omega(\mathcal{A}) = \emptyset$

- derselbe NBBA, aber mit $F = \{A, B\}$: $\quad L_\omega(\mathcal{A}) = ?$

2019-02-01

Teil 4: unendliche Bäume
└─Automaten auf unendlichen Bäumen

└─Beispiele (Büchi)



16:24

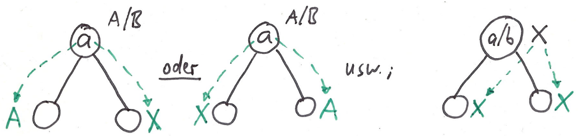
1. Bsp.: lies **spaltenweise!**

Beispiele (Büchi)

- NBBA $\mathcal{A} = (\{A, B, X\},\ \{a, b\},\ \Delta,\ \{A\},\ \{A, X\})$ mit $\Delta =$

  $\{\ (A, a, A, X),\ (B, a, A, X),\ (A, b, B, X),\ (B, b, B, X),\ (X, a, X, X),$
  $\ \ (A, a, X, A),\ (B, a, X, A),\ (A, b, X, B),\ (B, b, X, B),\ (X, b, X, X)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid t \text{ hat mind. einen Pfad mit } \infty \text{ vielen } a\text{'s}\}$

- derselbe NBBA, aber mit $F = \{B, X\}$

  $L_\omega(\mathcal{A}) = \{t \mid t \text{ hat mind. einen Pfad mit } \infty \text{ vielen } b\text{'s}\}$

- derselbe NBBA, aber mit $F = \{X\}$: $\quad L_\omega(\mathcal{A}) = \emptyset$

- derselbe NBBA, aber mit $F = \{A, B\}$: $\quad L_\omega(\mathcal{A}) = \emptyset$

16:24

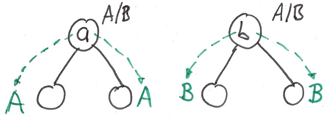
1. Bsp.: lies **spaltenweise!**

Beispiele (Muller)

- NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = ?$

16:28

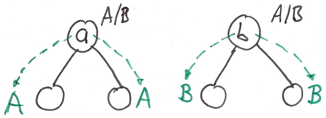
Beispiele (Muller)
NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit
$\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$
Skizze:
$L_\omega(\mathcal{A}) = ?$

Beispiele (Muller)

- NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } b\text{'s}\}$ (!)

2019-02-01

Teil 4: unendliche Bäume
└Automaten auf unendlichen Bäumen
└Beispiele (Muller)

16:28

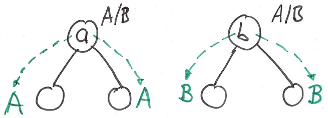
Beispiele (Muller)
NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit
$\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$
Skizze:
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } b\text{'s}\}$ (!)

Beispiele (Muller)

- NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } b\text{'s}\}$ (!)

- derselbe NMBA, aber mit $F = \{\{B\}\}$

  $L_\omega(\mathcal{A}) = ?$

16:28

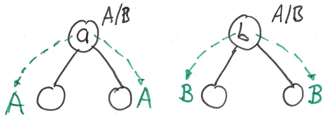
Beispiele (Muller)
NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit
$\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$
Skizze:
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } b\text{'s}\}$ (!)
derselbe NMBA, aber mit $F = \{\{B\}\}$
$L_\omega(\mathcal{A}) = ?$

Beispiele (Muller)

- NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } b\text{'s}\}$ (!)

- derselbe NMBA, aber mit $F = \{\{B\}\}$
  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } a\text{'s}\}$

Beispiele (Muller)
NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit
$\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$
Skizze:
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } b\text{'s}\}$ (!)
derselbe NMBA, aber mit $F = \{\{B\}\}$
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } a\text{'s}\}$

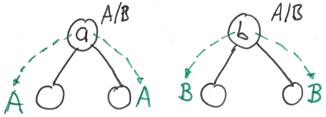
16:28

Beispiele (Muller)

- NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:

  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } b\text{'s}\}$ (!)

- derselbe NMBA, aber mit $F = \{\{B\}\}$
  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } a\text{'s}\}$

- derselbe NMBA, aber mit $F = \{\{A, B\}\}$
  $L_\omega(\mathcal{A}) = ?$

16:28

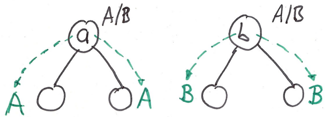
Beispiele (Muller)
NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit
$\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$
Skizze:
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } b\text{'s}\}$ (!)
derselbe NMBA, aber mit $F = \{\{B\}\}$
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } a\text{'s}\}$
derselbe NMBA, aber mit $F = \{\{A, B\}\}$
$L_\omega(\mathcal{A}) = ?$

Beispiele (Muller)

- NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } b\text{'s}\}$ (!)

- derselbe NMBA, aber mit $F = \{\{B\}\}$
  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } a\text{'s}\}$

- derselbe NMBA, aber mit $F = \{\{A, B\}\}$
  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } a\text{'s und } \infty \text{ viele } b\text{'s}\}$

Beispiele (Muller)
NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit
$\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$
Skizze:
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } b\text{'s}\}$ (!)
derselbe NMBA, aber mit $F = \{\{B\}\}$
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } a\text{'s}\}$
derselbe NMBA, aber mit $F = \{\{A, B\}\}$
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } a\text{'s und } \infty \text{ viele } b\text{'s}\}$

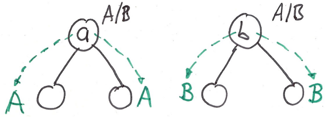
16:28

Beispiele (Muller)

- NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } b\text{'s}\}$ (!)

- derselbe NMBA, aber mit $F = \{\{B\}\}$
  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } a\text{'s}\}$

- derselbe NMBA, aber mit $F = \{\{A, B\}\}$
  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } a\text{'s und } \infty \text{ viele } b\text{'s}\}$

- derselbe NMBA, aber mit $F = \{\{A\}, \{B\}\}$
  $L_\omega(\mathcal{A}) = ?$

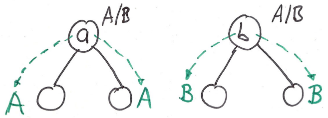
16:28

Beispiele (Muller)

- NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit

  $\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$

  Skizze:



  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } b\text{'s}\}$ (!)

- derselbe NMBA, aber mit $F = \{\{B\}\}$
  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } a\text{'s}\}$

- derselbe NMBA, aber mit $F = \{\{A, B\}\}$
  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } a\text{'s und } \infty \text{ viele } b\text{'s}\}$

- derselbe NMBA, aber mit $F = \{\{A\}, \{B\}\}$
  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endl. viele } b\text{'s } \textit{oder} \text{ endl. viele } a\text{'s}\}$

2019-02-01

Teil 4: unendliche Bäume
└Automaten auf unendlichen Bäumen
└Beispiele (Muller)

16:28

Beispiele (Muller)
NMBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ \{\{A\}\})$ mit
$\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$
Skizze:
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } b\text{'s}\}$ (!)
derselbe NMBA, aber mit $F = \{\{B\}\}$
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } a\text{'s}\}$
derselbe NMBA, aber mit $F = \{\{A, B\}\}$
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } a\text{'s und } \infty \text{ viele } b\text{'s}\}$
derselbe NMBA, aber mit $F = \{\{A\}, \{B\}\}$
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endl. viele } b\text{'s } \textit{oder} \text{ endl. viele } a\text{'s}\}$

Beispiel (Parität)

Zur Erinnerung:

Run r ist erfolgreich, wenn für alle Pfade $\pi \subseteq T$ gilt:

$$\min\{c(q) \mid q \in \mathsf{Inf}(r, \pi)\} \text{ ist gerade}$$

Beispiel (Parität)
Zur Erinnerung:
Run r ist erfolgreich, wenn für alle Pfade $\pi \subseteq T$ gilt:
$\min\{c(q) \mid q \in \mathsf{Inf}(r, \pi)\}$ ist gerade

16:32

Beispiel (Parität)

Zur Erinnerung:

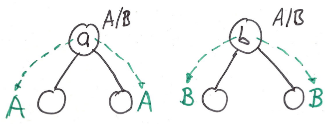
Run r ist erfolgreich, wenn für alle Pfade $\pi \subseteq T$ gilt:

$$\min\{c(q) \mid q \in \mathrm{Inf}(r, \pi)\} \text{ ist gerade}$$

NPBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ c)$ mit

$$\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$$

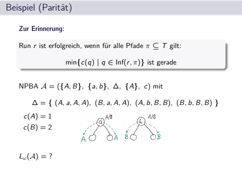
$c(A) = 1$
$c(B) = 2$



$L_\omega(\mathcal{A}) = ?$

Teil 4: unendliche Bäume
└─Automaten auf unendlichen Bäumen
└─Beispiel (Parität)

2019-02-01



16:32

Beispiel (Parität)

Zur Erinnerung:

Run r ist erfolgreich, wenn für alle Pfade $\pi \subseteq T$ gilt:

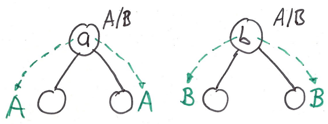
$$\min\{c(q) \mid q \in \mathrm{Inf}(r, \pi)\} \text{ ist gerade}$$

NPBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ c)$ mit

$$\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$$

$c(A) = 1$

$c(B) = 2$



$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } a\text{'s}\}$

2019-02-01

Teil 4: unendliche Bäume
Automaten auf unendlichen Bäumen
Beispiel (Parität)

16:32

Beispiel (Parität)
Zur Erinnerung:
Run r ist erfolgreich, wenn für alle Pfade $\pi \subseteq T$ gilt:
$\min\{c(q) \mid q \in \mathrm{Inf}(r, \pi)\}$ ist gerade
NPBA $\mathcal{A} = (\{A, B\},\ \{a, b\},\ \Delta,\ \{A\},\ c)$ mit
$\Delta = \{\ (A, a, A, A),\ (B, a, A, A),\ (A, b, B, B),\ (B, b, B, B)\ \}$
$c(A) = 1$
$c(B) = 2$
$L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat endlich viele } a\text{'s}\}$

Büchi- versus Muller-Erkennbarkeit

Satz 4.12

1. Jede Büchi-erkennbare Sprache ist Muller-erkennbar.
2. **Nicht jede** Muller-erkennbare Sprache ist Büchi-erkennbar.

Büchi- versus Muller-Erkennbarkeit
Satz 4.12
1. Jede Büchi-erkennbare Sprache ist Muller-erkennbar.
2. Nicht jede Muller-erkennbare Sprache ist Büchi-erkennbar.

16:35

Büchi- versus Muller-Erkennbarkeit

Satz 4.12

1. Jede Büchi-erkennbare Sprache ist Muller-erkennbar.
2. **Nicht jede** Muller-erkennbare Sprache ist Büchi-erkennbar.

Beweis.

1. Wie im letzten Kapitel.

16:35

Büchi- versus Muller-Erkennbarkeit

Satz 4.12
1. Jede Büchi-erkennbare Sprache ist Muller-erkennbar.
2. Nicht jede Muller-erkennbare Sprache ist Büchi-erkennbar.

Beweis.
1. Wie im letzten Kapitel.

Büchi- versus Muller-Erkennbarkeit

Satz 4.12

1. Jede Büchi-erkennbare Sprache ist Muller-erkennbar.
2. **Nicht jede** Muller-erkennbare Sprache ist Büchi-erkennbar.

Beweis.

1. Wie im letzten Kapitel.
2. Betrachten $L = \{t \mid \text{jeder Pfad in } t \text{ hat endlich viele } a\text{'s}\}$

   L ist Muller-erkennbar (siehe Bsp. auf Folie 34)

   Müssen zeigen: L nicht Büchi-erkennbar

16:35

Büchi- versus Muller-Erkennbarkeit

Satz 4.12
1. Jede Büchi-erkennbare Sprache ist Muller-erkennbar.
2. Nicht jede Muller-erkennbare Sprache ist Büchi-erkennbar.

Beweis.
1. Wie im letzten Kapitel.
2. Betrachten $L = \{t \mid \text{jeder Pfad in } t \text{ hat endlich viele } a\text{'s}\}$
L ist Muller-erkennbar (siehe Bsp. auf Folie 34)
Müssen zeigen: L nicht Büchi-erkennbar

Büchi- versus Muller-Erkennbarkeit

Zu zeigen: $L = \{t \mid \text{jeder Pfad in } t \text{ hat endlich viele } a\text{'s}\}$ nicht Büchi-erkennbar.

16:36

Büchi- versus Muller-Erkennbarkeit
Zu zeigen: $L = \{t \mid \text{jeder Pfad in } t \text{ hat endlich viele } a\text{'s}\}$ nicht Büchi-erkennbar.

Büchi- versus Muller-Erkennbarkeit

Zu zeigen: $L = \{t \mid \text{jeder Pfad in } t \text{ hat endlich viele } a\text{'s}\}$ nicht Büchi-erkennbar.

Nehmen an, es gebe NBBA $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ mit $L_\omega(\mathcal{A}) = L$.

Büchi- versus Muller-Erkennbarkeit

Zu zeigen: $L = \{t \mid \text{jeder Pfad in } t \text{ hat endlich viele } a\text{'s}\}$ nicht Büchi-erkennbar.

Nehmen an, es gebe NBBA $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ mit $L_\omega(\mathcal{A}) = L$.

16:36

Büchi- versus Muller-Erkennbarkeit

Zu zeigen: $L = \{t \mid \text{jeder Pfad in } t \text{ hat endlich viele } a\text{'s}\}$
nicht Büchi-erkennbar.

Nehmen an, es gebe NBBA $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ mit $L_\omega(\mathcal{A}) = L$.

O. B. d. A. sei $I = \{q_0\}$. Sei $n := |F|$.

Büchi- versus Muller-Erkennbarkeit

Zu zeigen: $L = \{t \mid \text{jeder Pfad in } t \text{ hat endlich viele } a\text{'s}\}$
nicht Büchi-erkennbar.

Nehmen an, es gebe NBBA $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ mit $L_\omega(\mathcal{A}) = L$.
O. B. d. A. sei $I = \{q_0\}$. Sei $n := |F|$.

16:36

Büchi- versus Muller-Erkennbarkeit

Zu zeigen: $L = \{t \mid \text{jeder Pfad in } t \text{ hat endlich viele } a\text{'s}\}$ nicht Büchi-erkennbar.

Nehmen an, es gebe NBBA $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ mit $L_\omega(\mathcal{A}) = L$.

O. B. d. A. sei $I = \{q_0\}$. Sei $n := |F|$.

Idee:

- Bestimme Baum $t \in L$ mit Run r und Pfad, auf dem zwischen 2 Besuchen *desselben* akzeptierenden Zustandes ein a auftritt
- „Pumpe“ t, r so auf, dass dieser Teilpfad sich ∞ oft wiederholt
- ϟ Neuer Baum wird akzeptiert, aber neuer Pfad hat ∞ viele a's

16:36

Büchi- versus Muller-Erkennbarkeit

Zu zeigen: $L = \{t \mid \text{jeder Pfad in } t \text{ hat endlich viele } a\text{'s}\}$ nicht Büchi-erkennbar.

Nehmen an, es gebe NBBA $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ mit $L_\omega(\mathcal{A}) = L$.
O. B. d. A. sei $I = \{q_0\}$. Sei $n := |F|$.

Idee:
- Bestimme Baum $t \in L$ mit Run r und Pfad, auf dem zwischen 2 Besuchen *desselben* akzeptierenden Zustandes ein a auftritt
- „Pumpe“ t, r so auf, dass dieser Teilpfad sich ∞ oft wiederholt
- ϟ Neuer Baum wird akzeptiert, aber neuer Pfad hat ∞ viele a's

Büchi- versus Muller-Erkennbarkeit

Betrachte Baum $t \in L$ mit $\quad t(p) = a \quad \text{gdw.} \quad p \in \bigcup_{i=1,\dots,n} (1^+0)^i,$

Büchi- versus Muller-Erkennbarkeit
Betrachte Baum $t \in L$ mit $t(p) = a$ gdw. $p \in \bigcup_{i=1,\dots,n} (1^+0)^i$,

16:38 bis 16:50, 5 min Pause

Man beachte: a's tauchen in *beliebiger Tiefe* auf,
aber auf jedem Pfad nur auf den ersten n Linksschritten.

Dadurch hat jeder Pfad nur max. n a's, also endlich viele.

Büchi- versus Muller-Erkennbarkeit

Betrachte Baum $t \in L$ mit $\quad t(p) = a \quad$ gdw. $\quad p \in \bigcup_{i=1,\dots,n} (1^+0)^i$,

d. h. t enthält ein a an allen Positionen, die man erreicht,
wenn man bei der Wurzel startet und bis zu n-mal wie folgt läuft:

- einmal oder mehrmals zum rechten Kind (beliebig oft)
- einmal zum linken Kind („Linksschritt“)

Büchi- versus Muller-Erkennbarkeit
Betrachte Baum $t \in L$ mit $t(p) = a$ gdw. $p \in \bigcup_{i=1,\dots,n} (1^+0)^i$,
d. h. t enthält ein a an allen Positionen, die man erreicht,
wenn man bei der Wurzel startet und bis zu n-mal wie folgt läuft:
einmal oder mehrmals zum rechten Kind (beliebig oft)
einmal zum linken Kind („Linksschritt“)

16:38 bis 16:50, 5 min Pause

Man beachte: a's tauchen in *beliebiger Tiefe* auf,
aber auf jedem Pfad nur auf den ersten n Linksschritten.

Dadurch hat jeder Pfad nur max. n a's, also endlich viele.

Büchi- versus Muller-Erkennbarkeit

Betrachte Baum $t \in L$ mit $\quad t(p) = a \quad$ gdw. $\quad p \in \bigcup_{i=1,\ldots,n} (1^+0)^i$,

d. h. t enthält ein a an allen Positionen, die man erreicht,
wenn man bei der Wurzel startet und bis zu n-mal wie folgt läuft:

- einmal oder mehrmals zum rechten Kind (beliebig oft)
- einmal zum linken Kind („Linksschritt“)

An den übrigen Positionen enthält t ein b.

Büchi- versus Muller-Erkennbarkeit
Betrachte Baum $t \in L$ mit $t(p) = a$ gdw. $p \in \bigcup_{i=1,\ldots,n} (1^+0)^i$,
d. h. t enthält ein a an allen Positionen, die man erreicht,
wenn man bei der Wurzel startet und bis zu n-mal wie folgt läuft:
- einmal oder mehrmals zum rechten Kind (beliebig oft)
- einmal zum linken Kind („Linksschritt“)
An den übrigen Positionen enthält t ein b.

16:38 bis 16:50, 5 min Pause

Man beachte: a’s tauchen in *beliebiger Tiefe* auf,
aber auf jedem Pfad nur auf den ersten n Linksschritten.

Dadurch hat jeder Pfad nur max. n a’s, also endlich viele.

Büchi- versus Muller-Erkennbarkeit

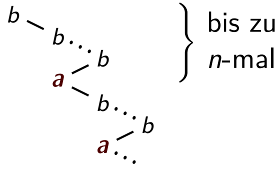
Betrachte Baum $t \in L$ mit $\quad t(p) = a \quad$ gdw. $\quad p \in \bigcup_{i=1,\dots,n} (1^+0)^i$,

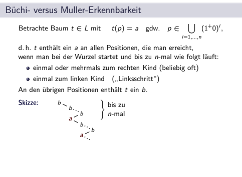
d. h. t enthält ein a an allen Positionen, die man erreicht, wenn man bei der Wurzel startet und bis zu n-mal wie folgt läuft:

- einmal oder mehrmals zum rechten Kind (beliebig oft)
- einmal zum linken Kind („Linksschritt“)

An den übrigen Positionen enthält t ein b.

Skizze:





16:38 bis 16:50, 5 min Pause

Man beachte: a's tauchen in *beliebiger Tiefe* auf, aber auf jedem Pfad nur auf den ersten n Linksschritten.

Dadurch hat jeder Pfad nur max. n a's, also endlich viele.

Büchi- versus Muller-Erkennbarkeit

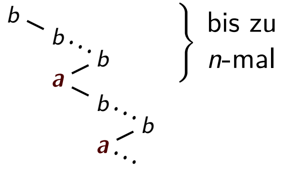
Betrachte Baum $t \in L$ mit $\quad t(p) = a \quad$ gdw. $\quad p \in \bigcup_{i=1,\dots,n} (1^+0)^i$,

d. h. t enthält ein a an allen Positionen, die man erreicht,
wenn man bei der Wurzel startet und bis zu n-mal wie folgt läuft:

- einmal oder mehrmals zum rechten Kind (beliebig oft)
- einmal zum linken Kind („Linksschritt")

An den übrigen Positionen enthält t ein b.

Skizze:



Klar: $t \in L$. Sei r ein erfolgreicher Run.

Büchi- versus Muller-Erkennbarkeit
Betrachte Baum $t \in L$ mit $t(p) = a$ gdw. $p \in \bigcup_{i=1,\dots,n} (1^+0)^i$,
d. h. t enthält ein a an allen Positionen, die man erreicht, wenn man bei der Wurzel startet und bis zu n-mal wie folgt läuft:
einmal oder mehrmals zum rechten Kind (beliebig oft)
einmal zum linken Kind („Linksschritt")
An den übrigen Positionen enthält t ein b.
Skizze: bis zu n-mal
Klar: $t \in L$. Sei r ein erfolgreicher Run.

16:38 bis 16:50, 5 min Pause

Man beachte: a's tauchen in *beliebiger Tiefe* auf,
aber auf jedem Pfad nur auf den ersten n Linksschritten.

Dadurch hat jeder Pfad nur max. n a's, also endlich viele.

Büchi- versus Muller-Erkennbarkeit

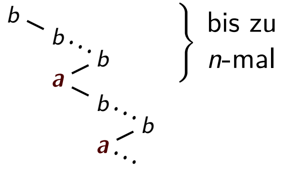
Betrachte Baum $t \in L$ mit $\quad t(p) = a \quad$ gdw. $\quad p \in \bigcup_{i=1,\ldots,n} (1^{+}0)^{i}$,

d. h. t enthält ein a an allen Positionen, die man erreicht,
wenn man bei der Wurzel startet und bis zu n-mal wie folgt läuft:

- einmal oder mehrmals zum rechten Kind (beliebig oft)
- einmal zum linken Kind („Linksschritt")

An den übrigen Positionen enthält t ein b.

Skizze:



Klar: $t \in L$. Sei r ein erfolgreicher Run.

Details des Pumpens: s. Tafel **T 4.6** □

Büchi- versus Muller-Erkennbarkeit

Betrachte Baum $t \in L$ mit $t(p) = a$ gdw. $p \in \bigcup_{i=1,\ldots,n} (1^{+}0)^{i}$,

d. h. t enthält ein a an allen Positionen, die man erreicht, wenn man bei der Wurzel startet und bis zu n-mal wie folgt läuft:

- einmal oder mehrmals zum rechten Kind (beliebig oft)
- einmal zum linken Kind („Linksschritt")

An den übrigen Positionen enthält t ein b.

Skizze:

Klar: $t \in L$. Sei r ein erfolgreicher Run.

Details des Pumpens: s. Tafel T 4.6 □

16:38 bis 16:50, 5 min Pause

Man beachte: a's tauchen in *beliebiger Tiefe* auf,
aber auf jedem Pfad nur auf den ersten n Linksschritten.

Dadurch hat jeder Pfad nur max. n a's, also endlich viele.

Folgerung aus dem Beweis von Satz 4.12

Folgerung 4.13

Die Klasse der Büchi-erkennbaren Baumsprachen ist
nicht abgeschlossen unter ?

Folgerung aus dem Beweis von Satz 4.12
Folgerung 4.13
Die Klasse der Büchi-erkennbaren Baumsprachen ist
nicht abgeschlossen unter ?

16:55

Vor Satz 4.14 sagen: Anfang des letzten Beweises erinnert nicht ohne Grund an Beweis der Ungleichmächtigkeit von DBAs und NBAs

Folgerung aus dem Beweis von Satz 4.12

Folgerung 4.13

Die Klasse der Büchi-erkennbaren Baumsprachen ist **nicht** abgeschlossen unter Komplement.

Folgerung aus dem Beweis von Satz 4.12
Folgerung 4.13
Die Klasse der Büchi-erkennbaren Baumsprachen ist nicht abgeschlossen unter Komplement.

16:55

Vor Satz 4.14 sagen: Anfang des letzten Beweises erinnert nicht ohne Grund an Beweis der Ungleichmächtigkeit von DBAs und NBAs

Folgerung aus dem Beweis von Satz 4.12

Folgerung 4.13

Die Klasse der Büchi-erkennbaren Baumsprachen ist **nicht** abgeschlossen unter Komplement.

Man kann Satz 4.12 stärker formulieren (ohne Beweis):

Satz 4.14

Die Menge der Baumsprachen, die Muller-, aber nicht Büchi-erkennbar sind, ist

$$\{L_\triangle \mid L \text{ ist NBA-erkennbar, aber nicht DBA-erkennbar}\}.$$

($L \subseteq \Sigma^\omega$ ist eine ω-Sprache;
$L_\triangle$ = Menge aller Σ-Bäume, deren Beschriftung entlang *jedes* Pfades in L liegt)

Folgerung aus dem Beweis von Satz 4.12
Folgerung 4.13
Die Klasse der Büchi-erkennbaren Baumsprachen ist nicht abgeschlossen unter Komplement.
Man kann Satz 4.12 stärker formulieren (ohne Beweis):
Satz 4.14
Die Menge der Baumsprachen, die Muller-, aber nicht Büchi-erkennbar sind, ist
{L△ | L ist NBA-erkennbar, aber nicht DBA-erkennbar}.
(L ⊆ Σ^ω ist eine ω-Sprache;
L△ = Menge aller Σ-Bäume, deren Beschriftung entlang jedes Pfades in L liegt)

16:55

Vor Satz 4.14 sagen: Anfang des letzten Beweises erinnert nicht ohne Grund an Beweis der Ungleichmächtigkeit von DBAs und NBAs

Paritäts- versus Muller-Erkennbarkeit

Satz 4.15

1. Jede paritäts-erkennbare Sprache ist Muller-erkennbar.
2. Jede Muller-erkennbare Sprache ist paritäts-erkennbar.

16:57

Paritäts- versus Muller-Erkennbarkeit
Satz 4.15
1. Jede paritäts-erkennbare Sprache ist Muller-erkennbar.
2. Jede Muller-erkennbare Sprache ist paritäts-erkennbar.

Paritäts- versus Muller-Erkennbarkeit

Satz 4.15

1. Jede paritäts-erkennbare Sprache ist Muller-erkennbar.
2. Jede Muller-erkennbare Sprache ist paritäts-erkennbar.

Beweis.

1. Wie im letzten Kapitel.

Paritäts- versus Muller-Erkennbarkeit
Satz 4.15
1. Jede paritäts-erkennbare Sprache ist Muller-erkennbar.
2. Jede Muller-erkennbare Sprache ist paritäts-erkennbar.
Beweis.
1. Wie im letzten Kapitel.

16:57

Paritäts- versus Muller-Erkennbarkeit

Satz 4.15

1. Jede paritäts-erkennbare Sprache ist Muller-erkennbar.
2. Jede Muller-erkennbare Sprache ist paritäts-erkennbar.

Beweis.

1. Wie im letzten Kapitel.
2. Sei $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$ NMBA, mit
   $I = \{q_0\}$ und $\mathcal{F} = \{F\}$ (o. B. d. A.) sowie $n := |Q|$.

16:57

Paritäts- versus Muller-Erkennbarkeit
Satz 4.15
1. Jede paritäts-erkennbare Sprache ist Muller-erkennbar.
2. Jede Muller-erkennbare Sprache ist paritäts-erkennbar.
Beweis.
1. Wie im letzten Kapitel.
2. Sei $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$ NMBA, mit $I = \{q_0\}$ und $\mathcal{F} = \{F\}$ (o. B. d. A.) sowie $n := |Q|$.

Paritäts- versus Muller-Erkennbarkeit

Satz 4.15

1. Jede paritäts-erkennbare Sprache ist Muller-erkennbar.
2. Jede Muller-erkennbare Sprache ist paritäts-erkennbar.

Beweis.

1. Wie im letzten Kapitel.
2. Sei $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$ NMBA, mit
   $I = \{q_0\}$ und $\mathcal{F} = \{F\}$ (o. B. d. A.) sowie $n := |Q|$.

   Gesucht: äquivalenter NPBA $\mathcal{A}'$

Paritäts- versus Muller-Erkennbarkeit

Satz 4.15
1. Jede paritäts-erkennbare Sprache ist Muller-erkennbar.
2. Jede Muller-erkennbare Sprache ist paritäts-erkennbar.

Beweis.
1. Wie im letzten Kapitel.
2. Sei $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$ NMBA, mit $I = \{q_0\}$ und $\mathcal{F} = \{F\}$ (o. B. d. A.) sowie $n := |Q|$.
   Gesucht: äquivalenter NPBA $\mathcal{A}'$

16:57

Paritäts- versus Muller-Erkennbarkeit

Satz 4.15

1. Jede paritäts-erkennbare Sprache ist Muller-erkennbar.
2. Jede Muller-erkennbare Sprache ist paritäts-erkennbar.

Beweis.

1. Wie im letzten Kapitel.
2. Sei $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$ NMBA, mit $I = \{q_0\}$ und $\mathcal{F} = \{F\}$ (o. B. d. A.) sowie $n := |Q|$.

   Gesucht: äquivalenter NPBA $\mathcal{A}'$

   Idee: $\mathcal{A}'$ soll ...

   - „sich merken", in welcher Reihenfolge die n Zustände zuletzt gesehen wurden (Permutation $q_1 \cdots q_n$ von Q)

16:57

Paritäts- versus Muller-Erkennbarkeit

Satz 4.15

1. Jede paritäts-erkennbare Sprache ist Muller-erkennbar.
2. Jede Muller-erkennbare Sprache ist paritäts-erkennbar.

Beweis.

1. Wie im letzten Kapitel.
2. Sei $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$ NMBA, mit $I = \{q_0\}$ und $\mathcal{F} = \{F\}$ (o. B. d. A.) sowie $n := |Q|$.

   Gesucht: äquivalenter NPBA $\mathcal{A}'$

   Idee: $\mathcal{A}'$ soll ...

   - „sich merken", in welcher Reihenfolge die n Zustände zuletzt gesehen wurden (Permutation $q_1 \cdots q_n$ von Q)
   - sicherstellen, dass ab einem gewissen Zeitpunkt genau die Zustände aus F dauerhaft am Ende der Permutation stehen

Paritäts- versus Muller-Erkennbarkeit

Satz 4.15
1. Jede paritäts-erkennbare Sprache ist Muller-erkennbar.
2. Jede Muller-erkennbare Sprache ist paritäts-erkennbar.

Beweis.
1. Wie im letzten Kapitel.
2. Sei $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$ NMBA, mit $I = \{q_0\}$ und $\mathcal{F} = \{F\}$ (o. B. d. A.) sowie $n := |Q|$.
Gesucht: äquivalenter NPBA $\mathcal{A}'$
Idee: $\mathcal{A}'$ soll ...
- „sich merken", in welcher Reihenfolge die n Zustände zuletzt gesehen wurden (Permutation $q_1 \cdots q_n$ von Q)
- sicherstellen, dass ab einem gewissen Zeitpunkt genau die Zustände aus F dauerhaft am Ende der Permutation stehen

16:57

Details der Konstruktion (1)

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_0\}, F)$ NMBA mit $|Q| = n$.
Konstruieren NPBA $\mathcal{A}' = (Q', \Sigma, \Delta', I', c)$ mit Zuständen

$$Q' = \{\langle q_1 \cdots q_n,\ \ell\rangle \mid q_1 \cdots q_n \text{ ist Permutation von } Q,\ \ell \in \{1, \dots, n\}\}$$

Idee:

- q_n ist der zuletzt besuchte Zustand auf dem aktuellen Pfad, q_{n-1} der zuletzt besuchte Zustand $\neq q_n$ usw.
- ℓ ist Position von q_n in der vorangehenden Permutation

Skizze: siehe Tafel **T 4.7**

16:59

Details der Konstruktion (1)

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_0\}, F)$ NMBA mit $|Q| = n$.
Konstruieren NPBA $\mathcal{A}' = (Q', \Sigma, \Delta', I', c)$ mit Zuständen

$$Q' = \{\langle q_1 \cdots q_n,\ \ell\rangle \mid q_1 \cdots q_n \text{ ist Permutation von } Q,\ \ell \in \{1, \dots, n\}\}$$

Idee:

- q_n ist der zuletzt besuchte Zustand auf dem aktuellen Pfad, q_{n-1} der zuletzt besuchte Zustand $\neq q_n$ usw.
- ℓ ist Position von q_n in der vorangehenden Permutation

Skizze: siehe Tafel T 4.7

Details der Konstruktion (2)

Zeigen zunächst folgende **Hilfsaussage (HA)** über Zustände von $\mathcal{A}'$

Sei $q_1 q_2 q_3 \dots$ eine Folge von Zuständen aus Q;
sei $s_1 s_2 s_3 \dots$ die zugehörige Folge von Zuständen aus Q'
mit $s_1 = \langle t_1 \cdots t_{n-1} q_1,\ 1 \rangle$ und $s_i = \langle \mathrm{perm}_i, \ell_i \rangle$ für alle $i \geqslant 0$.

Dann gilt $\mathrm{Inf}(q_1 q_2 q_3 \dots) = S$ mit $|S| = k$ gdw.

1. Für endlich viele i ist $\ell_i \leqslant n - k$ und
2. Für unendlich viele i gilt:
   (a) $\ell_i = n - k + 1$ und
   (b) Die Menge der Zustände in den Positionen $\underbrace{n-k+1, \dots, n}_{\text{letzte } k \text{ Positionen}}$ in perm_i ist S

Beweis der Hilfsaussage: siehe Tafel **T 4.8**

Details der Konstruktion (2)
Zeigen zunächst folgende Hilfsaussage (HA) über Zustände von $\mathcal{A}'$
Sei $q_1 q_2 q_3 \dots$ eine Folge von Zuständen aus Q;
sei $s_1 s_2 s_3 \dots$ die zugehörige Folge von Zuständen aus Q'
mit $s_1 = \langle t_1 \cdots t_{n-1} q_1,\ 1 \rangle$ und $s_i = \langle \mathrm{perm}_i, \ell_i \rangle$ für alle $i \geqslant 0$.
Dann gilt $\mathrm{Inf}(q_1 q_2 q_3 \dots) = S$ mit $|S| = k$ gdw.
Für endlich viele i ist $\ell_i \leqslant n - k$ und
Für unendlich viele i gilt:
(a) $\ell_i = n - k + 1$ und
(b) Die Menge der Zustände in den Positionen $\underbrace{n-k+1, \dots, n}_{\text{letzte } k \text{ Positionen}}$ in perm_i ist S
Beweis der Hilfsaussage: siehe Tafel T 4.8

17:06 bis 17:11 Tafelanschrieb nur, wenn viel mehr Zeit!

Details der Konstruktion (3)

Können nun Konstruktion fortsetzen:

$$I' = \Big\{ \langle t_1 \cdots t_{n-1} q_0,\, 1 \rangle \mid t_1 \cdots t_{n-1} \text{ ist Perm. von } Q \setminus \{q_0\} \Big\}$$

Details der Konstruktion (3)
Können nun Konstruktion fortsetzen:
$I' = \Big\{ \langle t_1 \cdots t_{n-1} q_0,\, 1 \rangle \mid t_1 \cdots t_{n-1} \text{ ist Perm. von } Q \setminus \{q_0\} \Big\}$

17:11 bis 17:29; (wenn Zeit knapp, dann ohne Tafelanschrieb)

(!) Wahl der Akzeptanzbedingung wird nur durch Beweis so richtig klar . . .

Details der Konstruktion (3)

Können nun Konstruktion fortsetzen:

$$I' = \Big\{ \langle t_1 \cdots t_{n-1} q_0,\ 1 \rangle \mid t_1 \cdots t_{n-1} \text{ ist Perm. von } Q \setminus \{q_0\} \Big\}$$

$$\Delta' = \Big\{ \Big(\langle i_1 \cdots i_{n-1} i,\ \ell \rangle,\ a,\ \langle i'_1 \cdots i'_{n-1} i',\ \ell' \rangle,\ \langle i''_1 \cdots i''_{n-1} i'',\ \ell'' \rangle \Big) \mid$$

- $(i, a, i', i'') \in \Delta$
- $i'_1 \cdots i'_{n-1}$ entsteht aus $i_1 \cdots i_{n-1} i$ durch Löschen von i'
- $i''_1 \cdots i''_{n-1}$ entsteht aus $i_1 \cdots i_{n-1} i$ durch Löschen von i''
- ℓ' = Position von i' in $i_1 \cdots i_{n-1} i$
- ℓ'' = Position von i'' in $i_1 \cdots i_{n-1} i$ $\Big\}$

Details der Konstruktion (3)
Können nun Konstruktion fortsetzen:
$I' = \{ \langle t_1 \cdots t_{n-1} q_0, 1 \rangle \mid t_1 \cdots t_{n-1}$ ist Perm. von $Q \setminus \{q_0\} \}$
$\Delta' = \{ (\langle i_1 \cdots i_{n-1} i, \ell \rangle, a, \langle i'_1 \cdots i'_{n-1} i', \ell' \rangle, \langle i''_1 \cdots i''_{n-1} i'', \ell'' \rangle) \mid$
$(i, a, i', i'') \in \Delta$
$i'_1 \cdots i'_{n-1}$ entsteht aus $i_1 \cdots i_{n-1} i$ durch Löschen von i'
$i''_1 \cdots i''_{n-1}$ entsteht aus $i_1 \cdots i_{n-1} i$ durch Löschen von i''
ℓ' = Position von i' in $i_1 \cdots i_{n-1} i$
ℓ'' = Position von i'' in $i_1 \cdots i_{n-1} i$ $\}$

17:11 bis 17:29; (wenn Zeit knapp, dann ohne Tafelanschrieb)

(!) Wahl der Akzeptanzbedingung wird nur durch Beweis so richtig klar …

Details der Konstruktion (3)

Können nun Konstruktion fortsetzen:

$$I' = \Big\{ \langle t_1 \cdots t_{n-1} q_0,\ 1 \rangle \mid t_1 \cdots t_{n-1} \text{ ist Perm. von } Q \setminus \{q_0\} \Big\}$$

$$\Delta' = \Big\{ \Big(\langle i_1 \cdots i_{n-1} i,\ \ell \rangle,\ a,\ \langle i'_1 \cdots i'_{n-1} i',\ \ell' \rangle,\ \langle i''_1 \cdots i''_{n-1} i'',\ \ell'' \rangle \Big) \mid$$

- $(i, a, i', i'') \in \Delta$
- $i'_1 \cdots i'_{n-1}$ entsteht aus $i_1 \cdots i_{n-1} i$ durch Löschen von i'
- $i''_1 \cdots i''_{n-1}$ entsteht aus $i_1 \cdots i_{n-1} i$ durch Löschen von i''
- $\ell' =$ Position von i' in $i_1 \cdots i_{n-1} i$
- $\ell'' =$ Position von i'' in $i_1 \cdots i_{n-1} i$ $\Big\}$

$$c(s) = \begin{cases} 2\ell & \text{falls } s = \langle q_1 \cdots q_n,\ \ell \rangle \text{ und } \{q_\ell, \ldots, q_n\} = F \\ 2\ell + 1 & \text{falls } s = \langle q_1 \cdots q_n,\ \ell \rangle \text{ und } \{q_\ell, \ldots, q_n\} \neq F \end{cases}$$

Details der Konstruktion (3)
Können nun Konstruktion fortsetzen:
$I' = \Big\{ \langle t_1 \cdots t_{n-1} q_0,\ 1 \rangle \mid t_1 \cdots t_{n-1} \text{ ist Perm. von } Q \setminus \{q_0\} \Big\}$
$\Delta' = \Big\{ \Big(\langle i_1 \cdots i_{n-1} i,\ \ell \rangle,\ a,\ \langle i'_1 \cdots i'_{n-1} i',\ \ell' \rangle,\ \langle i''_1 \cdots i''_{n-1} i'',\ \ell'' \rangle \Big) \mid$
- $(i, a, i', i'') \in \Delta$
- $i'_1 \cdots i'_{n-1}$ entsteht aus $i_1 \cdots i_{n-1} i$ durch Löschen von i'
- $i''_1 \cdots i''_{n-1}$ entsteht aus $i_1 \cdots i_{n-1} i$ durch Löschen von i''
- $\ell' =$ Position von i' in $i_1 \cdots i_{n-1} i$
- $\ell'' =$ Position von i'' in $i_1 \cdots i_{n-1} i$ $\Big\}$
$c(s) = \begin{cases} 2\ell & \text{falls } s = \langle q_1 \cdots q_n,\ \ell \rangle \text{ und } \{q_\ell, \ldots, q_n\} = F \\ 2\ell + 1 & \text{falls } s = \langle q_1 \cdots q_n,\ \ell \rangle \text{ und } \{q_\ell, \ldots, q_n\} \neq F \end{cases}$

17:11 bis 17:29; (wenn Zeit knapp, dann ohne Tafelanschrieb)

(!) Wahl der Akzeptanzbedingung wird nur durch Beweis so richtig klar ...

Details der Konstruktion (3)

Können nun Konstruktion fortsetzen:

$$I' = \Big\{\langle t_1 \cdots t_{n-1} q_0,\ 1\rangle \mid t_1 \cdots t_{n-1} \text{ ist Perm. von } Q \setminus \{q_0\}\Big\}$$

$$\Delta' = \Big\{\Big(\langle i_1 \cdots i_{n-1} i,\ \ell\rangle,\ a,\ \langle i'_1 \cdots i'_{n-1} i',\ \ell'\rangle,\ \langle i''_1 \cdots i''_{n-1} i'',\ \ell''\rangle\Big) \mid$$

- $(i, a, i', i'') \in \Delta$
- $i'_1 \cdots i'_{n-1}$ entsteht aus $i_1 \cdots i_{n-1} i$ durch Löschen von i'
- $i''_1 \cdots i''_{n-1}$ entsteht aus $i_1 \cdots i_{n-1} i$ durch Löschen von i''
- $\ell' =$ Position von i' in $i_1 \cdots i_{n-1} i$
- $\ell'' =$ Position von i'' in $i_1 \cdots i_{n-1} i$ $\Big\}$

$$c(s) = \begin{cases} 2\ell & \text{falls } s = \langle q_1 \cdots q_n,\ \ell\rangle \text{ und } \{q_\ell, \dots, q_n\} = F \\ 2\ell + 1 & \text{falls } s = \langle q_1 \cdots q_n,\ \ell\rangle \text{ und } \{q_\ell, \dots, q_n\} \neq F \end{cases}$$

Beweis der Korrektheit: siehe Tafel **T 4.9** □

17:11 bis 17:29; (wenn Zeit knapp, dann ohne Tafelanschrieb)

(!) Wahl der Akzeptanzbedingung wird nur durch Beweis so richtig klar ...

Abschlusseigenschaften

Satz 4.16

Die Klasse der ...

1. Büchi-erkennbaren Sprachen ist abgeschlossen unter $\cup$ und $\cap$, aber nicht unter $\overline{}$.
2. Muller-erkennbaren Sprachen ist abgeschlossen unter $\cup, \cap, \overline{}$.

Abschlusseigenschaften

Satz 4.16

Die Klasse der ...

1. Büchi-erkennbaren Sprachen ist abgeschlossen unter $\cup$ und $\cap$, aber nicht unter $\overline{}$.
2. Muller-erkennbaren Sprachen ist abgeschlossen unter $\cup, \cap, \overline{}$.

17:29 bis 17:30 $\rightsquigarrow$ Ende Gelände

Abschlusseigenschaften

Satz 4.16

Die Klasse der ...

1. Büchi-erkennbaren Sprachen ist abgeschlossen unter $\cup$ und $\cap$, aber nicht unter $\overline{}$.
2. Muller-erkennbaren Sprachen ist abgeschlossen unter $\cup, \cap, \overline{}$.

Beweis:

1. $\cup\cap$ wie gehabt; $\overline{}$ siehe Folgerung 4.13.
2. $\cup\cap$ wie gehabt;
   $\overline{}$ siehe nächsten Abschnitt □

Abschlusseigenschaften
Satz 4.16
Die Klasse der ...
Büchi-erkennbaren Sprachen ist abgeschlossen unter ∪ und ∩, aber nicht unter ‾.
Muller-erkennbaren Sprachen ist abgeschlossen unter ∪, ∩, ‾.
Beweis:
∪∩ wie gehabt; ‾ siehe Folgerung 4.13.
∪∩ wie gehabt; ‾ siehe nächsten Abschnitt

17:29 bis 17:30 $\rightsquigarrow$ Ende Gelände

Und nun ...

1. *Model-Checking mit CTL*
2. Automaten auf unendlichen Bäumen
3. Komplementierung

Teil 4: unendliche Bäume
└─Komplementierung
└─Und nun ...
2019-02-01

Und nun ...
Model-Checking mit CTL
Automaten auf unendlichen Bäumen
Komplementierung

Überblick

Ziel dieses Abschnitts:

Lösen Komplementierung mit Hilfe eines bekannten Resultates über Gewinnstrategien in einer bestimmten Art (abstrakter) Spiele

Vorgehen:

- Ordnen jedem NPBA $\mathcal{A}$ und Baum t ein 2-Personen-Spiel $G_{\mathcal{A},t}$ zu (Beschränkung auf NPBAs ist unerheblich, siehe Satz 4.15)
- Dann wird leicht zu sehen sein:
  $\mathcal{A}$ akzeptiert t $\Leftrightarrow$ Spielerin 1 hat Gewinnstrategie in $G_{\mathcal{A},t}$
- Ein Resultat aus der Spieltheorie impliziert:
  In $G_{\mathcal{A},t}$ hat immer **genau eine Spielerin eine Gewinnstrategie, die nicht vom bisherigen Spielverlauf abhängt**
- Konstruieren $\mathcal{A}'$, so dass gilt:
  $\mathcal{A}'$ akzeptiert t $\Leftrightarrow$ **Spielerin 2 hat Gewinnstrategie in $G_{\mathcal{A},t}$**
  Dann folgt $L_\omega(\mathcal{A}') = \overline{L_\omega(\mathcal{A})}$

2019-02-01

Teil 4: unendliche Bäume
└Komplementierung
└Überblick

Überblick

Ziel dieses Abschnitts:
Lösen Komplementierung mit Hilfe eines bekannten Resultates über Gewinnstrategien in einer bestimmten Art (abstrakter) Spiele
Vorgehen:
Ordnen jedem NPBA $\mathcal{A}$ und Baum t ein 2-Personen-Spiel $G_{\mathcal{A},t}$ zu (Beschränkung auf NPBAs ist unerheblich, siehe Satz 4.15)
Dann wird leicht zu sehen sein: $\mathcal{A}$ akzeptiert t $\Leftrightarrow$ Spielerin 1 hat Gewinnstrategie in $G_{\mathcal{A},t}$
Ein Resultat aus der Spieltheorie impliziert: In $G_{\mathcal{A},t}$ hat immer genau eine Spielerin eine Gewinnstrategie, die nicht vom bisherigen Spielverlauf abhängt
Konstruieren $\mathcal{A}'$, so dass gilt: $\mathcal{A}'$ akzeptiert t $\Leftrightarrow$ Spielerin 2 hat Gewinnstrategie in $G_{\mathcal{A},t}$
Dann folgt $L_\omega(\mathcal{A}') = \overline{L_\omega(\mathcal{A})}$

16:00

Intuitive Beschreibung des Spiels $G_{\mathcal{A},t}$

Zwei Spielerinnen **Aut** (Automat), **PF** (Pfadfinderin)

- sind abwechselnd an der Reihe
- bewegen sich pro Runde 1 Schritt im Baum durch Markieren von Positionen mit Zuständen; zu Beginn: (ε, q_I), $q_I \in I$

Intuitive Beschreibung des Spiels $G_{\mathcal{A},t}$
Zwei Spielerinnen **Aut** (Automat), **PF** (Pfadfinderin)
- sind abwechselnd an der Reihe
- bewegen sich pro Runde 1 Schritt im Baum durch Markieren von Positionen mit Zuständen; zu Beginn: (ε, q_I), $q_I \in I$

16:03 bis 16:12

q_I bezeichne ab hier einen beliebigen Anfangszustand;
brauchen „q_0“ später noch anderweitig.

Es wird also ein Baumautomat mit **beliebiger Akzeptanzbedingung** (Büchi, Muller, whatever) zugrunde gelegt.

Intuitive Beschreibung des Spiels $G_{\mathcal{A},t}$

Zwei Spielerinnen **Aut** (Automat), **PF** (Pfadfinderin)

- sind abwechselnd an der Reihe
- bewegen sich pro Runde 1 Schritt im Baum durch Markieren von Positionen mit Zuständen; zu Beginn: (ε, q_I), $q_I \in I$

In jeder Runde wählt

- **Aut** eine Transition, die auf die markierte Position anwendbar ist
- **PF** einen Kindknoten und verschiebt Markierung dorthin

Intuitive Beschreibung des Spiels $G_{\mathcal{A},t}$
Zwei Spielerinnen Aut (Automat), PF (Pfadfinderin)
sind abwechselnd an der Reihe
bewegen sich pro Runde 1 Schritt im Baum durch Markieren von Positionen mit Zuständen; zu Beginn: (ε, q_I), $q_I \in I$
In jeder Runde wählt
Aut eine Transition, die auf die markierte Position anwendbar ist
PF einen Kindknoten und verschiebt Markierung dorthin

16:03 bis 16:12

q_I bezeichne ab hier einen beliebigen Anfangszustand; brauchen „q_0“ später noch anderweitig.

Es wird also ein Baumautomat mit **beliebiger Akzeptanzbedingung** (Büchi, Muller, whatever) zugrunde gelegt.

Intuitive Beschreibung des Spiels $G_{\mathcal{A},t}$

Zwei Spielerinnen **Aut** (Automat), **PF** (Pfadfinderin)

- sind abwechselnd an der Reihe
- bewegen sich pro Runde 1 Schritt im Baum durch Markieren von Positionen mit Zuständen; zu Beginn: (ε, q_I), $q_I \in I$

In jeder Runde wählt

- **Aut** eine Transition, die auf die markierte Position anwendbar ist
- **PF** einen Kindknoten und verschiebt Markierung dorthin

Spiel läuft ∞ lange, erzeugt ∞ Folge $r = q_0 q_1 q_2$ von Zuständen (bestimmt durch die gewählten Transitionen)

Intuitive Beschreibung des Spiels $G_{\mathcal{A},t}$
Zwei Spielerinnen **Aut** (Automat), **PF** (Pfadfinderin)
- sind abwechselnd an der Reihe
- bewegen sich pro Runde 1 Schritt im Baum durch Markieren von Positionen mit Zuständen; zu Beginn: (ε, q_I), $q_I \in I$
In jeder Runde wählt
- **Aut** eine Transition, die auf die markierte Position anwendbar ist
- **PF** einen Kindknoten und verschiebt Markierung dorthin
Spiel läuft ∞ lange, erzeugt ∞ Folge $r = q_0 q_1 q_2$ von Zuständen (bestimmt durch die gewählten Transitionen)

16:03 bis 16:12

q_I bezeichne ab hier einen beliebigen Anfangszustand; brauchen „q_0“ später noch anderweitig.

Es wird also ein Baumautomat mit **beliebiger Akzeptanzbedingung** (Büchi, Muller, whatever) zugrunde gelegt.

Intuitive Beschreibung des Spiels $G_{\mathcal{A},t}$

Zwei Spielerinnen **Aut** (Automat), **PF** (Pfadfinderin)

- sind abwechselnd an der Reihe
- bewegen sich pro Runde 1 Schritt im Baum durch Markieren von Positionen mit Zuständen; zu Beginn: (ε, q_I), $q_I \in I$

In jeder Runde wählt

- **Aut** eine Transition, die auf die markierte Position anwendbar ist
- **PF** einen Kindknoten und verschiebt Markierung dorthin

Spiel läuft ∞ lange, erzeugt ∞ Folge $r = q_0 q_1 q_2$ von Zuständen (bestimmt durch die gewählten Transitionen)

Aut gewinnt, wenn r der Akzeptanzbedingung von $\mathcal{A}$ entspricht; sonst gewinnt **PF**
(d. h. **Aut** versucht, $\mathcal{A}$ zum Akzeptieren zu bringen; **PF** versucht das zu verhindern)

Intuitive Beschreibung des Spiels $G_{\mathcal{A},t}$
Zwei Spielerinnen Aut (Automat), PF (Pfadfinderin)
sind abwechselnd an der Reihe
bewegen sich pro Runde 1 Schritt im Baum durch Markieren von Positionen mit Zuständen; zu Beginn: (ε, q_I), $q_I \in I$
In jeder Runde wählt
Aut eine Transition, die auf die markierte Position anwendbar ist
PF einen Kindknoten und verschiebt Markierung dorthin
Spiel läuft ∞ lange, erzeugt ∞ Folge $r = q_0 q_1 q_2$ von Zuständen (bestimmt durch die gewählten Transitionen)
Aut gewinnt, wenn r der Akzeptanzbedingung von $\mathcal{A}$ entspricht; sonst gewinnt PF
(d. h. Aut versucht, $\mathcal{A}$ zum Akzeptieren zu bringen; PF versucht das zu verhindern)

16:03 bis 16:12

q_I bezeichne ab hier einen beliebigen Anfangszustand; brauchen „q_0“ später noch anderweitig.

Es wird also ein Baumautomat mit **beliebiger Akzeptanzbedingung** (Büchi, Muller, whatever) zugrunde gelegt.

Intuitive Beschreibung des Spiels $G_{\mathcal{A},t}$

Zwei Spielerinnen **Aut** (Automat), **PF** (Pfadfinderin)

- sind abwechselnd an der Reihe
- bewegen sich pro Runde 1 Schritt im Baum durch Markieren von Positionen mit Zuständen; zu Beginn: (ε, q_I), $q_I \in I$

In jeder Runde wählt

- **Aut** eine Transition, die auf die markierte Position anwendbar ist
- **PF** einen Kindknoten und verschiebt Markierung dorthin

Spiel läuft ∞ lange, erzeugt ∞ Folge $r = q_0 q_1 q_2$ von Zuständen (bestimmt durch die gewählten Transitionen)

Aut gewinnt, wenn r der Akzeptanzbedingung von $\mathcal{A}$ entspricht; sonst gewinnt **PF**
(d. h. **Aut** versucht, $\mathcal{A}$ zum Akzeptieren zu bringen; **PF** versucht das zu verhindern)

Skizze: s. Tafel **T 4.10**

Intuitive Beschreibung des Spiels $G_{\mathcal{A},t}$
Zwei Spielerinnen Aut (Automat), PF (Pfadfinderin)
sind abwechselnd an der Reihe
bewegen sich pro Runde 1 Schritt im Baum durch Markieren von Positionen mit Zuständen; zu Beginn: (ε, q_I), $q_I \in I$
In jeder Runde wählt
Aut eine Transition, die auf die markierte Position anwendbar ist
PF einen Kindknoten und verschiebt Markierung dorthin
Spiel läuft ∞ lange, erzeugt ∞ Folge $r = q_0 q_1 q_2$ von Zuständen (bestimmt durch die gewählten Transitionen)
Aut gewinnt, wenn r der Akzeptanzbedingung von $\mathcal{A}$ entspricht; sonst gewinnt PF
(d. h. Aut versucht, $\mathcal{A}$ zum Akzeptieren zu bringen; PF versucht das zu verhindern)
Skizze: s. Tafel T 4.10

16:03 bis 16:12

q_I bezeichne ab hier einen beliebigen Anfangszustand; brauchen „q_0“ später noch anderweitig.

Es wird also ein Baumautomat mit **beliebiger Akzeptanzbedingung** (Büchi, Muller, whatever) zugrunde gelegt.

Genaue Beschreibung des Spiels $G_{\mathcal{A},t}$

Spiel ist ein unendlicher Graph

- Knoten sind die **Spielpositionen**:
  - für **Aut**: $\{(p,q) \mid p \in \{0,1\}^*,\ q \in Q\}$ (Positionen im Baum)
  - für **PF**: $\{(q, t(p), q_0, q_1) \in \Delta \mid p \in \{0,1\}^*\}$ (Transitionen)

Genaue Beschreibung des Spiels $G_{\mathcal{A},t}$

Spiel ist ein unendlicher Graph

- Knoten sind die Spielpositionen:
  - für Aut: $\{(p,q) \mid p \in \{0,1\}^*,\ q \in Q\}$ (Positionen im Baum)
  - für PF: $\{(q, t(p), q_0, q_1) \in \Delta \mid p \in \{0,1\}^*\}$ (Transitionen)

16:12

Achtung: Ab jetzt bezeichnet p Positionen im Baum, nicht mehr Zustände!

Genaue Beschreibung des Spiels $G_{\mathcal{A},t}$

Spiel ist ein unendlicher Graph

- Knoten sind die **Spielpositionen**:
  - für **Aut**: $\{(p,q) \mid p \in \{0,1\}^*,\ q \in Q\}$ (Positionen im Baum)
  - für **PF**: $\{(q,t(p),q_0,q_1) \in \Delta \mid p \in \{0,1\}^*\}$ (Transitionen)
- Kanten sind die möglichen **Spielzüge**:
  - $(p,q) \rightarrow (q,t(p),q_0,q_1)$
  - $(q,t(p),q_0,q_1) \rightarrow (p0,q_0)$
 $\searrow (p1,q_1)$

Genaue Beschreibung des Spiels $G_{\mathcal{A},t}$
Spiel ist ein unendlicher Graph
Knoten sind die Spielpositionen:
für Aut: $\{(p,q) \mid p \in \{0,1\}^*,\ q \in Q\}$ (Positionen im Baum)
für PF: $\{(q,t(p),q_0,q_1) \in \Delta \mid p \in \{0,1\}^*\}$ (Transitionen)
Kanten sind die möglichen Spielzüge:
$(p,q) \rightarrow (q,t(p),q_0,q_1)$
$(q,t(p),q_0,q_1) \rightarrow (p0,q_0)$ $\searrow (p1,q_1)$

16:12

Achtung: Ab jetzt bezeichnet p Positionen im Baum, nicht mehr Zustände!

Genaue Beschreibung des Spiels $G_{\mathcal{A},t}$

Spiel ist ein unendlicher Graph

- Knoten sind die **Spielpositionen**:
  - für **Aut**: $\{(p, q) \mid p \in \{0,1\}^*,\ q \in Q\}$ (Positionen im Baum)
  - für **PF**: $\{(q, t(p), q_0, q_1) \in \Delta \mid p \in \{0,1\}^*\}$ (Transitionen)
- Kanten sind die möglichen **Spielzüge**:
  - $(p, q) \to (q, t(p), q_0, q_1)$
  - $(q, t(p), q_0, q_1) \to (p0, q_0)$
    $\searrow (p1, q_1)$
- Startknoten: (ε, q_I) für $q_I \in I$ (o. B. d. A. $I = \{q_I\}$)

16:12

Achtung: Ab jetzt bezeichnet p Positionen im Baum, nicht mehr Zustände!

Genaue Beschreibung des Spiels $G_{\mathcal{A},t}$

Spiel ist ein unendlicher Graph

- Knoten sind die **Spielpositionen**:
  - für **Aut**: $\{(p,q) \mid p \in \{0,1\}^*,\ q \in Q\}$ (Positionen im Baum)
  - für **PF**: $\{(q, t(p), q_0, q_1) \in \Delta \mid p \in \{0,1\}^*\}$ (Transitionen)
- Kanten sind die möglichen **Spielzüge**:
  - $(p,q) \rightarrow (q, t(p), q_0, q_1)$
  - $(q, t(p), q_0, q_1) \rightarrow (p0, q_0)$
    $\searrow (p1, q_1)$
- Startknoten: (ε, q_I) für $q_I \in I$ (o. B. d. A. $I = \{q_I\}$)

Jede mögliche ∞ Folge von Spielzügen entspricht
einem ∞ Pfad im Spielbaum $G_{\mathcal{A},t}$

Genaue Beschreibung des Spiels $G_{\mathcal{A},t}$
Spiel ist ein unendlicher Graph
Knoten sind die Spielpositionen: für Aut: $\{(p,q) \mid p \in \{0,1\}^*,\ q \in Q\}$ (Positionen im Baum); für PF: $\{(q, t(p), q_0, q_1) \in \Delta \mid p \in \{0,1\}^*\}$ (Transitionen)
Kanten sind die möglichen Spielzüge: $(p,q) \rightarrow (q, t(p), q_0, q_1)$; $(q, t(p), q_0, q_1) \rightarrow (p0, q_0)$ $\searrow (p1, q_1)$
Startknoten: (ε, q_I) für $q_I \in I$ (o. B. d. A. $I = \{q_I\}$)
Jede mögliche ∞ Folge von Spielzügen entspricht einem ∞ Pfad im Spielbaum $G_{\mathcal{A},t}$

16:12

Achtung: Ab jetzt bezeichnet p Positionen im Baum, nicht mehr Zustände!

Genaue Beschreibung des Spiels $G_{\mathcal{A},t}$

Spiel ist ein unendlicher Graph

- Knoten sind die **Spielpositionen**:
  - für **Aut**: $\{(p,q) \mid p \in \{0,1\}^*,\ q \in Q\}$ (Positionen im Baum)
  - für **PF**: $\{(q,t(p),q_0,q_1) \in \Delta \mid p \in \{0,1\}^*\}$ (Transitionen)
- Kanten sind die möglichen **Spielzüge**:
  - $(p,q) \to (q,t(p),q_0,q_1)$
  - $(q,t(p),q_0,q_1) \to (p0,q_0)$ $\searrow (p1,q_1)$
- Startknoten: (ε,q_I) für $q_I \in I$ (o. B. d. A. $I = \{q_I\}$)

Jede mögliche ∞ Folge von Spielzügen entspricht einem ∞ Pfad im Spielbaum $G_{\mathcal{A},t}$

Knoten v' **erreichbar** von Knoten v:
es gibt endliche Folge von Spielzügen von v nach v'

Genaue Beschreibung des Spiels $G_{\mathcal{A},t}$

Spiel ist ein unendlicher Graph

- Knoten sind die Spielpositionen:
  - für Aut: $\{(p,q) \mid p \in \{0,1\}^*,\ q \in Q\}$ (Positionen im Baum)
  - für PF: $\{(q,t(p),q_0,q_1) \in \Delta \mid p \in \{0,1\}^*\}$ (Transitionen)
- Kanten sind die möglichen Spielzüge:
  - $(p,q) \to (q,t(p),q_0,q_1)$
  - $(q,t(p),q_0,q_1) \to (p0,q_0)$ $\searrow (p1,q_1)$
- Startknoten: (ε,q_I) für $q_I \in I$ (o. B. d. A. $I = \{q_I\}$)

Jede mögliche ∞ Folge von Spielzügen entspricht einem ∞ Pfad im Spielbaum $G_{\mathcal{A},t}$

Knoten v' erreichbar von Knoten v:
es gibt endliche Folge von Spielzügen von v nach v'

16:12

Achtung: Ab jetzt bezeichnet p Positionen im Baum, nicht mehr Zustände!

Spielstrategien

Strategie ab Spielposition v für Spielerin $X \in \{\mathbf{Aut}, \mathbf{PF}\}$:

Funktion, die jeder Zugfolge $v \dots v'$ mit v' Spielposition für X einen in v' möglichen Zug zuweist
(legt fest, welchen Zug X in jeder von v aus erreichbaren Spielposition macht)

Spielstrategien

Strategie ab Spielposition v für Spielerin $X \in \{\mathbf{Aut}, \mathbf{PF}\}$:
Funktion, die jeder Zugfolge $v \dots v'$ mit v' Spielposition für X einen in v' möglichen Zug zuweist
(legt fest, welchen Zug X in jeder von v aus erreichbaren Spielposition macht)

16:17 bis 16:26

Spielstrategien

Strategie ab Spielposition v für Spielerin $X \in \{\textbf{Aut}, \textbf{PF}\}$:

Funktion, die jeder Zugfolge $v \dots v'$ mit v' Spielposition für X einen in v' möglichen Zug zuweist
(legt fest, welchen Zug X in jeder von v aus erreichbaren Spielposition macht)

Gewinnstrategie für Spielerin $X \in \{\textbf{Aut}, \textbf{PF}\}$:

Strategie, die sicherstellt, dass X gewinnt,
unabhängig von den Zügen der Gegenspielerin **T 4.11**

Spielstrategien

Strategie ab Spielposition v für Spielerin $X \in \{\textbf{Aut}, \textbf{PF}\}$:
Funktion, die jeder Zugfolge $v \dots v'$ mit v' Spielposition für X einen in v' möglichen Zug zuweist
(legt fest, welchen Zug X in jeder von v aus erreichbaren Spielposition macht)

Gewinnstrategie für Spielerin $X \in \{\textbf{Aut}, \textbf{PF}\}$:
Strategie, die sicherstellt, dass X gewinnt,
unabhängig von den Zügen der Gegenspielerin T 4.11

16:17 bis 16:26

Spielstrategien

Strategie ab Spielposition v für Spielerin $X \in \{\mathbf{Aut}, \mathbf{PF}\}$:

Funktion, die jeder Zugfolge $v \dots v'$ mit v' Spielposition für X einen in v' möglichen Zug zuweist
(legt fest, welchen Zug X in jeder von v aus erreichbaren Spielposition macht)

Gewinnstrategie für Spielerin $X \in \{\mathbf{Aut}, \mathbf{PF}\}$:

Strategie, die sicherstellt, dass X gewinnt,
unabhängig von den Zügen der Gegenspielerin **T 4.11**

gedächtnislose Strategie:

Strategie, die nur von v' abhängt, nicht von den vorigen Positionen

Spielstrategien

Strategie ab Spielposition v für Spielerin $X \in \{\mathbf{Aut}, \mathbf{PF}\}$:
Funktion, die jeder Zugfolge $v \dots v'$ mit v' Spielposition für X einen in v' möglichen Zug zuweist
(legt fest, welchen Zug X in jeder von v aus erreichbaren Spielposition macht)

Gewinnstrategie für Spielerin $X \in \{\mathbf{Aut}, \mathbf{PF}\}$:
Strategie, die sicherstellt, dass X gewinnt,
unabhängig von den Zügen der Gegenspielerin **T 4.11**

gedächtnislose Strategie:
Strategie, die nur von v' abhängt, nicht von den vorigen Positionen

16:17 bis 16:26

Akzeptanz mittels Gewinnstrategien

Lemma 4.17

Seien $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum.
Dann gilt:

$t \in L_\omega(\mathcal{A}) \;\Leftrightarrow\;$ **Aut** hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)

Akzeptanz mittels Gewinnstrategien

Lemma 4.17

Seien $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum.
Dann gilt:
$t \in L_\omega(\mathcal{A}) \;\Leftrightarrow\;$ **Aut** hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)

16:26

Akzeptanz mittels Gewinnstrategien

Lemma 4.17

Seien $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum.
Dann gilt:

$t \in L_\omega(\mathcal{A}) \Leftrightarrow$ **Aut** hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)

Beweis:

Konstruiere Gewinnstrategie direkt aus einem erfolgreichen Run und umgekehrt

Akzeptanz mittels Gewinnstrategien

Lemma 4.17
Seien $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum.
Dann gilt:
$t \in L_\omega(\mathcal{A}) \Leftrightarrow$ **Aut** hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)

Beweis:
Konstruiere Gewinnstrategie direkt aus einem erfolgreichen Run und umgekehrt

16:26

Akzeptanz mittels Gewinnstrategien

„$t \in L_\omega(\mathcal{A}) \;\Rightarrow\;$ Aut hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)“

Gelte $t \in L_\omega(\mathcal{A})$ und sei r erfolgreicher Run von $\mathcal{A}$ auf t.
Konstruiere Gewinnstrategie für **Aut** wie folgt aus r.

2019-02-01

Teil 4: unendliche Bäume
└─Komplementierung
└─Akzeptanz mittels Gewinnstrategien

Akzeptanz mittels Gewinnstrategien

„$t \in L_\omega(\mathcal{A}) \Rightarrow$ Aut hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)“

Gelte $t \in L_\omega(\mathcal{A})$ und sei r erfolgreicher Run von $\mathcal{A}$ auf t.
Konstruiere Gewinnstrategie für **Aut** wie folgt aus r.

16:28 bis 16:38

Akzeptanz mittels Gewinnstrategien

„$t \in L_\omega(\mathcal{A}) \;\Rightarrow\;$ Aut hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)“

Gelte $t \in L_\omega(\mathcal{A})$ und sei r erfolgreicher Run von $\mathcal{A}$ auf t.
Konstruiere Gewinnstrategie für **Aut** wie folgt aus r.

- in Startposition (ε, q_I) wähle $(r(\varepsilon), t(\varepsilon), r(0), r(1))$

Akzeptanz mittels Gewinnstrategien

„$t \in L_\omega(\mathcal{A}) \Rightarrow$ Aut hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)“

Gelte $t \in L_\omega(\mathcal{A})$ und sei r erfolgreicher Run von $\mathcal{A}$ auf t.
Konstruiere Gewinnstrategie für **Aut** wie folgt aus r.

- in Startposition (ε, q_I) wähle $(r(\varepsilon), t(\varepsilon), r(0), r(1))$

16:28 bis 16:38

Akzeptanz mittels Gewinnstrategien

„$t \in L_\omega(\mathcal{A}) \;\Rightarrow\;$ Aut hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)“

Gelte $t \in L_\omega(\mathcal{A})$ und sei r erfolgreicher Run von $\mathcal{A}$ auf t.
Konstruiere Gewinnstrategie für **Aut** wie folgt aus r.

- in Startposition (ε, q_I) wähle $(r(\varepsilon), t(\varepsilon), r(0), r(1))$
- in allen anderen Spielpos. (p, q) wähle $(q, t(p), r(p0), r(p1))$

Akzeptanz mittels Gewinnstrategien

„$t \in L_\omega(\mathcal{A}) \Rightarrow$ Aut hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)“

Gelte $t \in L_\omega(\mathcal{A})$ und sei r erfolgreicher Run von $\mathcal{A}$ auf t.
Konstruiere Gewinnstrategie für **Aut** wie folgt aus r.

- in Startposition (ε, q_I) wähle $(r(\varepsilon), t(\varepsilon), r(0), r(1))$
- in allen anderen Spielpos. (p, q) wähle $(q, t(p), r(p0), r(p1))$

16:28 bis 16:38

Akzeptanz mittels Gewinnstrategien

„$t \in L_\omega(\mathcal{A}) \;\Rightarrow\;$ Aut hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)"

Gelte $t \in L_\omega(\mathcal{A})$ und sei r erfolgreicher Run von $\mathcal{A}$ auf t.
Konstruiere Gewinnstrategie für **Aut** wie folgt aus r.

- in Startposition (ε, q_I) wähle $(r(\varepsilon), t(\varepsilon), r(0), r(1))$
- in allen anderen Spielpos. (p, q) wähle $(q, t(p), r(p0), r(p1))$

Wenn **Aut** diese Strategie befolgt, dann entspricht die im Spiel erzeugte Zustandsmenge einem Pfad in r.

Akzeptanz mittels Gewinnstrategien

„$t \in L_\omega(\mathcal{A}) \Rightarrow$ Aut hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)"

Gelte $t \in L_\omega(\mathcal{A})$ und sei r erfolgreicher Run von $\mathcal{A}$ auf t.
Konstruiere Gewinnstrategie für **Aut** wie folgt aus r.

- in Startposition (ε, q_I) wähle $(r(\varepsilon), t(\varepsilon), r(0), r(1))$
- in allen anderen Spielpos. (p, q) wähle $(q, t(p), r(p0), r(p1))$

Wenn **Aut** diese Strategie befolgt, dann entspricht die im Spiel erzeugte Zustandsmenge einem Pfad in r.

16:28 bis 16:38

Akzeptanz mittels Gewinnstrategien

„$t \in L_\omega(\mathcal{A}) \;\Rightarrow\;$ Aut hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)"

Gelte $t \in L_\omega(\mathcal{A})$ und sei r erfolgreicher Run von $\mathcal{A}$ auf t.
Konstruiere Gewinnstrategie für **Aut** wie folgt aus r.

- in Startposition (ε, q_I) wähle $(r(\varepsilon), t(\varepsilon), r(0), r(1))$
- in allen anderen Spielpos. (p, q) wähle $(q, t(p), r(p0), r(p1))$

Wenn **Aut** diese Strategie befolgt, dann entspricht die im Spiel erzeugte Zustandsmenge einem Pfad in r.

Da r erfolgreich, gewinnt **Aut** nach Definition von $G_{\mathcal{A},t}$.

Akzeptanz mittels Gewinnstrategien

„$t \in L_\omega(\mathcal{A}) \;\Rightarrow\;$ Aut hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)"

Gelte $t \in L_\omega(\mathcal{A})$ und sei r erfolgreicher Run von $\mathcal{A}$ auf t.
Konstruiere Gewinnstrategie für **Aut** wie folgt aus r.

- in Startposition (ε, q_I) wähle $(r(\varepsilon), t(\varepsilon), r(0), r(1))$
- in allen anderen Spielpos. (p, q) wähle $(q, t(p), r(p0), r(p1))$

Wenn **Aut** diese Strategie befolgt, dann entspricht die im Spiel erzeugte Zustandsmenge einem Pfad in r.

Da r erfolgreich, gewinnt **Aut** nach Definition von $G_{\mathcal{A},t}$.

16:28 bis 16:38

Akzeptanz mittels Gewinnstrategien

„$t \in L_\omega(\mathcal{A}) \Rightarrow$ Aut hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)"

Gelte $t \in L_\omega(\mathcal{A})$ und sei r erfolgreicher Run von $\mathcal{A}$ auf t.
Konstruiere Gewinnstrategie für **Aut** wie folgt aus r.

- in Startposition (ε, q_I) wähle $(r(\varepsilon), t(\varepsilon), r(0), r(1))$
- in allen anderen Spielpos. (p, q) wähle $(q, t(p), r(p0), r(p1))$

Wenn **Aut** diese Strategie befolgt, dann entspricht die im Spiel erzeugte Zustandsmenge einem Pfad in r.

Da r erfolgreich, gewinnt **Aut** nach Definition von $G_{\mathcal{A},t}$.

„Aut hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position $(\varepsilon, q_I) \Rightarrow t \in L_\omega(\mathcal{A})$"

T 4.12 □

Teil 4: unendliche Bäume
└Komplementierung

2019-02-01

└Akzeptanz mittels Gewinnstrategien

Akzeptanz mittels Gewinnstrategien

„$t \in L_\omega(\mathcal{A}) \Rightarrow$ Aut hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)"

Gelte $t \in L_\omega(\mathcal{A})$ und sei r erfolgreicher Run von $\mathcal{A}$ auf t.
Konstruiere Gewinnstrategie für **Aut** wie folgt aus r.

- in Startposition (ε, q_I) wähle $(r(\varepsilon), t(\varepsilon), r(0), r(1))$
- in allen anderen Spielpos. (p, q) wähle $(q, t(p), r(p0), r(p1))$

Wenn **Aut** diese Strategie befolgt, dann entspricht die im Spiel erzeugte Zustandsmenge einem Pfad in r.

Da r erfolgreich, gewinnt **Aut** nach Definition von $G_{\mathcal{A},t}$.

„Aut hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position $(\varepsilon, q_I) \Rightarrow t \in L_\omega(\mathcal{A})$"

T 4.12 □

16:28 bis 16:38

Determiniertheit von Paritätsspielen

Klassisches Resultat aus der Spieltheorie, hier nicht bewiesen:

Satz 4.18 (Emerson & Jutla 1991, Mostowski 1991)

Alle Paritätsspiele sind **gedächtnislos determiniert**:
genau eine der Spielerinnen hat eine gedächtnislose Gewinnstrategie.

„Paritätsspiel" bezeichnet dabei 2-Personen-Spiele, die

- auf Graphen gespielt werden,
  deren Knoten mit natürlichen Zahlen markiert sind;
- als Gewinnbedingung für unendliche Spielverläufe
  die Paritätsbedingung verwenden.

Für alle $\mathcal{A}$ und t ist $G_{\mathcal{A},t}$ ein Paritätsspiel.

Determiniertheit von Paritätsspielen

Klassisches Resultat aus der Spieltheorie, hier nicht bewiesen:

Satz 4.18 (Emerson & Jutla 1991, Mostowski 1991)

Alle Paritätsspiele sind gedächtnislos determiniert:
genau eine der Spielerinnen hat eine gedächtnislose Gewinnstrategie.

„Paritätsspiel" bezeichnet dabei 2-Personen-Spiele, die

- auf Graphen gespielt werden,
  deren Knoten mit natürlichen Zahlen markiert sind;
- als Gewinnbedingung für unendliche Spielverläufe
  die Paritätsbedingung verwenden.

Für alle $\mathcal{A}$ und t ist $G_{\mathcal{A},t}$ ein Paritätsspiel.

16:38

Determiniertheit von Paritätsspielen

Folgerung aus Satz 4.18:

Folgerung 4.19

Seien $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum.

Dann gibt es für jede Spielposition v in $G_{\mathcal{A},t}$ — und insbesondere für (ε, q_I) — eine gedächtnislose Gewinnstrategie für **Aut** oder **PF**.

Determiniertheit von Paritätsspielen

Folgerung aus Satz 4.18:

Folgerung 4.19

Seien $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum.

Dann gibt es für jede Spielposition v in $G_{\mathcal{A},t}$ — und insbesondere für (ε, q_I) — eine gedächtnislose Gewinnstrategie für **Aut** oder **PF**.

16:39 bis 16:41, 5min Pause

Determiniertheit von Paritätsspielen

Folgerung aus Satz 4.18:

Folgerung 4.19

Seien $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum.

Dann gibt es für jede Spielposition v in $G_{\mathcal{A},t}$ — und insbesondere für (ε, q_I) — eine gedächtnislose Gewinnstrategie für **Aut** oder **PF**.

Folgerung 4.20 (aus Lemma 4.17 und Folgerung 4.19)

$t \in \overline{L_\omega(\mathcal{A})} \;\Leftrightarrow\;$ **PF** hat gedächtnislose GS ab (ε, q_I) in $G_{\mathcal{A},t}$

Ziel: konstruieren NPBA, um deren Existenz zu testen

Determiniertheit von Paritätsspielen

Folgerung aus Satz 4.18:

Folgerung 4.19

Seien $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum.

Dann gibt es für jede Spielposition v in $G_{\mathcal{A},t}$ — und insbesondere für (ε, q_I) — eine gedächtnislose Gewinnstrategie für **Aut** oder **PF**.

Folgerung 4.20 (aus Lemma 4.17 und Folgerung 4.19)

$t \in \overline{L_\omega(\mathcal{A})} \;\Leftrightarrow\;$ **PF** hat gedächtnislose GS ab (ε, q_I) in $G_{\mathcal{A},t}$

Ziel: konstruieren NPBA, um deren Existenz zu testen

16:39 bis 16:41, 5min Pause

Gewinnbäume

Betrachten gedächtnislose Gewinnstrategien für **PF** als Menge von Funktionen

$$f_p : \Delta \to \{0,1\} \qquad \text{für jede Baumposition } p \in \{0,1\}^*$$

Idee: f_p weist jeder Transition, die **Aut** in Baumposition p wählt, einen Spielzug (Richtung 0/1) zu

Gewinnbäume

Betrachten gedächtnislose Gewinnstrategien für **PF** als Menge von Funktionen

$f_p : \Delta \to \{0,1\}$ für jede Baumposition $p \in \{0,1\}^*$

Idee: f_p weist jeder Transition, die **Aut** in Baumposition p wählt, einen Spielzug (Richtung 0/1) zu

16:46 bis 16:49

Gewinnbäume

Betrachten gedächtnislose Gewinnstrategien für **PF** als Menge von Funktionen

$$f_p : \Delta \to \{0, 1\} \qquad \text{für jede Baumposition } p \in \{0, 1\}^*$$

Idee: f_p weist jeder Transition, die **Aut** in Baumposition p wählt, einen Spielzug (Richtung 0/1) zu

- Sei F die Menge dieser Funktionen
- Ordnen die f_p in einem ***F*-Baum** s an (Strategiebaum)

Gewinnbäume

Betrachten gedächtnislose Gewinnstrategien für **PF** als Menge von Funktionen

$f_p : \Delta \to \{0, 1\}$ für jede Baumposition $p \in \{0, 1\}^*$

Idee: f_p weist jeder Transition, die **Aut** in Baumposition p wählt, einen Spielzug (Richtung 0/1) zu

- Sei F die Menge dieser Funktionen
- Ordnen die f_p in einem ***F*-Baum** s an (Strategiebaum)

16:46 bis 16:49

Gewinnbäume

Betrachten gedächtnislose Gewinnstrategien für **PF** als Menge von Funktionen

$$f_p : \Delta \to \{0,1\} \qquad \text{für jede Baumposition } p \in \{0,1\}^*$$

Idee: f_p weist jeder Transition, die **Aut** in Baumposition p wählt, einen Spielzug (Richtung 0/1) zu

- Sei F die Menge dieser Funktionen
- Ordnen die f_p in einem **F-Baum** s an (Strategiebaum)

PF-Gewinnbaum für t:
ein F-Baum, der eine Gewinnstrategie für **PF** in $G_{\mathcal{A},t}$ kodiert

16:46 bis 16:49

Gewinnbäume
Betrachten gedächtnislose Gewinnstrategien für **PF** als Menge von Funktionen
$f_p : \Delta \to \{0,1\}$ für jede Baumposition $p \in \{0,1\}^*$
Idee: f_p weist jeder Transition, die **Aut** in Baumposition p wählt, einen Spielzug (Richtung 0/1) zu
- Sei F die Menge dieser Funktionen
- Ordnen die f_p in einem **F-Baum** s an (Strategiebaum)
PF-Gewinnbaum für t:
ein F-Baum, der eine Gewinnstrategie für **PF** in $G_{\mathcal{A},t}$ kodiert

Gewinnbäume

Betrachten gedächtnislose Gewinnstrategien für **PF** als Menge von Funktionen

$$f_p : \Delta \to \{0,1\} \qquad \text{für jede Baumposition } p \in \{0,1\}^*$$

Idee: f_p weist jeder Transition, die **Aut** in Baumposition p wählt, einen Spielzug (Richtung 0/1) zu

- Sei F die Menge dieser Funktionen
- Ordnen die f_p in einem ***F*-Baum** s an (Strategiebaum)

PF-Gewinnbaum für t:
ein F-Baum, der eine Gewinnstrategie für **PF** in $G_{\mathcal{A},t}$ kodiert

Folgerung 4.21 (aus Folgerung 4.20)

$t \in \overline{L_\omega(\mathcal{A})} \;\Leftrightarrow\;$ es gibt einen **PF**-Gewinnbaum für t

Neues Ziel: bauen NPBA, um Existenz **PF**-Gewinnbaum zu testen

Gewinnbäume

Betrachten gedächtnislose Gewinnstrategien für PF als Menge von Funktionen

$f_p : \Delta \to \{0,1\}$ für jede Baumposition $p \in \{0,1\}^*$

Idee: f_p weist jeder Transition, die Aut in Baumposition p wählt, einen Spielzug (Richtung 0/1) zu

- Sei F die Menge dieser Funktionen
- Ordnen die f_p in einem F-Baum s an (Strategiebaum)

PF-Gewinnbaum für t:
ein F-Baum, der eine Gewinnstrategie für PF in $G_{\mathcal{A},t}$ kodiert

Folgerung 4.21 (aus Folgerung 4.20)

$t \in \overline{L_\omega(\mathcal{A})} \Leftrightarrow$ es gibt einen PF-Gewinnbaum für t

Neues Ziel: bauen NPBA, um Existenz PF-Gewinnbaum zu testen

16:46 bis 16:49

Existenz von PF-Gewinnbäumen (**PF**-GB)

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum

Zwischenziel: Prüfen, ob gegebener F-Baum s **kein** **PF**-GB ist

16:49 bis 16:58

Existenz von PF-Gewinnbäumen (**PF**-GB)
Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum
Zwischenziel: Prüfen, ob gegebener F-Baum s **kein** **PF**-GB ist

Existenz von PF-Gewinnbäumen (**PF**-GB)

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum

Zwischenziel: Prüfen, ob gegebener F-Baum s **kein** **PF**-GB ist

Idee:

- Benutzen NPA $\mathcal{A}'$ **(ω-Wortautomat)**
- $\mathcal{A}'$ prüft für jeden Pfad π in t und jeden möglichen Spielzug von **Aut** separat, ob Akzeptanzbedingung von $\mathcal{A}$ erfüllt ist

$\rightsquigarrow$ $\mathcal{A}'$ akzeptiert $\geqslant 1$ Pfad $\Leftrightarrow$ s ist kein **PF**-Gewinnbaum für t

Existenz von PF-Gewinnbäumen (**PF**-GB)

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum

Zwischenziel: Prüfen, ob gegebener F-Baum s kein PF-GB ist

Idee:

- Benutzen NPA $\mathcal{A}'$ (ω-Wortautomat)
- $\mathcal{A}'$ prüft für jeden Pfad π in t und jeden möglichen Spielzug von Aut separat, ob Akzeptanzbedingung von $\mathcal{A}$ erfüllt ist

$\rightsquigarrow$ $\mathcal{A}'$ akzeptiert $\geqslant 1$ Pfad $\Leftrightarrow$ s ist kein PF-Gewinnbaum für t

16:49 bis 16:58

Existenz von PF-Gewinnbäumen (**PF**-GB)

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum

Zwischenziel: Prüfen, ob gegebener F-Baum s **kein** **PF**-GB ist

Idee:

- Benutzen NPA $\mathcal{A}'$ **(ω-Wortautomat)**
- $\mathcal{A}'$ prüft für jeden Pfad π in t und jeden möglichen Spielzug von **Aut** separat, ob Akzeptanzbedingung von $\mathcal{A}$ erfüllt ist

$\leadsto$ $\mathcal{A}'$ akzeptiert $\geqslant 1$ Pfad $\Leftrightarrow$ s ist kein **PF**-Gewinnbaum für t

Sei $\pi \in \{0,1\}^\omega$ ein Pfad mit $\pi = \pi_1\pi_2\pi_3\ldots$

$\mathcal{A}'$ arbeitet auf Wörtern der folgenden Form:

$$\left\langle s(\varepsilon), t(\varepsilon), \pi_1 \right\rangle \left\langle s(\pi_1), t(\pi_1), \pi_2 \right\rangle \left\langle s(\pi_1\pi_2), t(\pi_1\pi_2), \pi_3 \right\rangle \ldots$$

Sei $\boldsymbol{L_{s,t}}$ die Sprache aller dieser Wörter

16:49 bis 16:58

Existenz von PF-Gewinnbäumen (PF-GB)
Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum
Zwischenziel: Prüfen, ob gegebener F-Baum s kein PF-GB ist
Idee:
Benutzen NPA $\mathcal{A}'$ (ω-Wortautomat)
$\mathcal{A}'$ prüft für jeden Pfad π in t und jeden möglichen Spielzug von Aut separat, ob Akzeptanzbedingung von $\mathcal{A}$ erfüllt ist
$\leadsto$ $\mathcal{A}'$ akzeptiert $\geqslant 1$ Pfad $\Leftrightarrow$ s ist kein PF-Gewinnbaum für t
Sei $\pi \in \{0,1\}^\omega$ ein Pfad mit $\pi = \pi_1\pi_2\pi_3\ldots$
$\mathcal{A}'$ arbeitet auf Wörtern der folgenden Form:
$\left\langle s(\varepsilon), t(\varepsilon), \pi_1 \right\rangle \left\langle s(\pi_1), t(\pi_1), \pi_2 \right\rangle \left\langle s(\pi_1\pi_2), t(\pi_1\pi_2), \pi_3 \right\rangle \ldots$
Sei $L_{s,t}$ die Sprache aller dieser Wörter

Existenz von PF-Gewinnbäumen (**PF**-GB)

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum

Zwischenziel: Prüfen, ob gegebener F-Baum s **kein PF**-GB ist

Idee:

- Benutzen NPA $\mathcal{A}'$ **(ω-Wortautomat)**
- $\mathcal{A}'$ prüft für jeden Pfad π in t und jeden möglichen Spielzug von **Aut** separat, ob Akzeptanzbedingung von $\mathcal{A}$ erfüllt ist

$\rightsquigarrow$ $\mathcal{A}'$ akzeptiert $\geqslant 1$ Pfad $\Leftrightarrow$ s ist kein **PF**-Gewinnbaum für t

Sei $\pi \in \{0,1\}^\omega$ ein Pfad mit $\pi = \pi_1\pi_2\pi_3\ldots$

$\mathcal{A}'$ arbeitet auf Wörtern der folgenden Form:

$$\big\langle s(\varepsilon), t(\varepsilon), \pi_1 \big\rangle \ \big\langle s(\pi_1), t(\pi_1), \pi_2 \big\rangle \ \big\langle s(\pi_1\pi_2), t(\pi_1\pi_2), \pi_3 \big\rangle \ \ldots$$

Sei $\boldsymbol{L_{s,t}}$ die Sprache aller dieser Wörter

Beispiel: s. Tafel **T 4.13**

Existenz von PF-Gewinnbäumen (PF-GB)
Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum
Zwischenziel: Prüfen, ob gegebener F-Baum s kein PF-GB ist
Idee:
- Benutzen NPA $\mathcal{A}'$ (ω-Wortautomat)
- $\mathcal{A}'$ prüft für jeden Pfad π in t und jeden möglichen Spielzug von Aut separat, ob Akzeptanzbedingung von $\mathcal{A}$ erfüllt ist
$\rightsquigarrow$ $\mathcal{A}'$ akzeptiert $\geqslant 1$ Pfad $\Leftrightarrow$ s ist kein PF-Gewinnbaum für t
Sei $\pi \in \{0,1\}^\omega$ ein Pfad mit $\pi = \pi_1\pi_2\pi_3\ldots$
$\mathcal{A}'$ arbeitet auf Wörtern der folgenden Form:
$\langle s(\varepsilon), t(\varepsilon), \pi_1 \rangle \ \langle s(\pi_1), t(\pi_1), \pi_2 \rangle \ \langle s(\pi_1\pi_2), t(\pi_1\pi_2), \pi_3 \rangle \ \ldots$
Sei $L_{s,t}$ die Sprache aller dieser Wörter
Beispiel: s. Tafel T 4.13

16:49 bis 16:58

Konstruktion des Wortautomaten für Gewinnbäume

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum
Konstruieren NPA $\mathcal{A}' = (Q, \Sigma', \Delta', \{q_I\}, c)$ wie folgt:

Konstruktion des Wortautomaten für Gewinnbäume
Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum
Konstruieren NPA $\mathcal{A}' = (Q, \Sigma', \Delta', \{q_I\}, c)$ wie folgt:

16:58 $\rightarrow$ mit Beweis bis 17:30

Achtung: $f \in F$ bedeutet „Funktion aus der Menge aller Funktionen (Gewinnstrategien)“, nicht „akzeptierender Zustand“!

Konstruktion des Wortautomaten für Gewinnbäume

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum
Konstruieren NPA $\mathcal{A}' = (Q, \Sigma', \Delta', \{q_I\}, c)$ wie folgt:

- $\Sigma' = \left\{ \langle f, a, i \rangle \;\middle|\; f \in F,\; a \in \Sigma,\; i \in \{0, 1\} \right\}$

Konstruktion des Wortautomaten für Gewinnbäume
Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum
Konstruieren NPA $\mathcal{A}' = (Q, \Sigma', \Delta', \{q_I\}, c)$ wie folgt:
- $\Sigma' = \left\{ \langle f, a, i \rangle \;\middle|\; f \in F,\; a \in \Sigma,\; i \in \{0, 1\} \right\}$

16:58 $\to$ mit Beweis bis 17:30

Achtung: $f \in F$ bedeutet „Funktion aus der Menge aller Funktionen (Gewinnstrategien)“, nicht „akzeptierender Zustand“!

Konstruktion des Wortautomaten für Gewinnbäume

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum

Konstruieren NPA $\mathcal{A}' = (Q, \Sigma', \Delta', \{q_I\}, c)$ wie folgt:

- $\Sigma' = \Big\{ \langle f, a, i \rangle \;\Big|\; f \in F,\ a \in \Sigma,\ i \in \{0, 1\} \Big\}$
- Q, c wie in $\mathcal{A}$ (wollen Akzeptanz von $\mathcal{A}$ prüfen)

Konstruktion des Wortautomaten für Gewinnbäume

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum

Konstruieren NPA $\mathcal{A}' = (Q, \Sigma', \Delta', \{q_I\}, c)$ wie folgt:

- $\Sigma' = \Big\{ \langle f, a, i \rangle \;\Big|\; f \in F,\ a \in \Sigma,\ i \in \{0, 1\} \Big\}$
- Q, c wie in $\mathcal{A}$ (wollen Akzeptanz von $\mathcal{A}$ prüfen)

16:58 $\rightarrow$ mit Beweis bis 17:30

Achtung: $f \in F$ bedeutet „Funktion aus der Menge aller Funktionen (Gewinnstrategien)“, nicht „akzeptierender Zustand“!

Konstruktion des Wortautomaten für Gewinnbäume

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum

Konstruieren NPA $\mathcal{A}' = (Q, \Sigma', \Delta', \{q_I\}, c)$ wie folgt:

- $\Sigma' = \Big\{\langle f, a, i\rangle \;\Big|\; f \in F,\ a \in \Sigma,\ i \in \{0,1\}\Big\}$
- Q, c wie in $\mathcal{A}$ (wollen Akzeptanz von $\mathcal{A}$ prüfen)
- $\Delta' = \Big\{\big(q,\ \langle f, a, i\rangle,\ q'_i\big) \;\Big|\; \langle f, a, i\rangle \in \Sigma',\ i \in \{0,1\},$
  $\quad$ es gibt $\delta = (q, a, q'_0, q'_1) \in \Delta$ mit $f(\delta) = i\Big\}$

Konstruktion des Wortautomaten für Gewinnbäume

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum
Konstruieren NPA $\mathcal{A}' = (Q, \Sigma', \Delta', \{q_I\}, c)$ wie folgt:
- $\Sigma' = \Big\{\langle f, a, i\rangle \;\Big|\; f \in F,\ a \in \Sigma,\ i \in \{0,1\}\Big\}$
- Q, c wie in $\mathcal{A}$ (wollen Akzeptanz von $\mathcal{A}$ prüfen)
- $\Delta' = \Big\{\big(q,\ \langle f, a, i\rangle,\ q'_i\big) \;\Big|\; \langle f, a, i\rangle \in \Sigma',\ i \in \{0,1\},$ es gibt $\delta = (q, a, q'_0, q'_1) \in \Delta$ mit $f(\delta) = i\Big\}$

16:58 $\rightarrow$ mit Beweis bis 17:30

Achtung: $f \in F$ bedeutet „Funktion aus der Menge aller Funktionen (Gewinnstrategien)“, nicht „akzeptierender Zustand“!

Konstruktion des Wortautomaten für Gewinnbäume

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum
Konstruieren NPA $\mathcal{A}' = (Q, \Sigma', \Delta', \{q_I\}, c)$ wie folgt:

- $\Sigma' = \left\{ \langle f, a, i \rangle \;\middle|\; f \in F,\ a \in \Sigma,\ i \in \{0, 1\} \right\}$
- Q, c wie in $\mathcal{A}$ (wollen Akzeptanz von $\mathcal{A}$ prüfen)
- $\Delta' = \left\{ \left(q,\ \langle f, a, i \rangle,\ q'_i\right) \;\middle|\; \langle f, a, i \rangle \in \Sigma',\ i \in \{0, 1\},\right.$ $\left.\text{es gibt } \delta = (q, a, q'_0, q'_1) \in \Delta \text{ mit } f(\delta) = i \right\}$

$\mathcal{A}'$ prüft für *jeden möglichen* Zug von **Aut**, ob **Aut** gewinnen kann

Konstruktion des Wortautomaten für Gewinnbäume
Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum
Konstruieren NPA $\mathcal{A}' = (Q, \Sigma', \Delta', \{q_I\}, c)$ wie folgt:
$\Sigma' = \left\{ \langle f, a, i \rangle \;\middle|\; f \in F,\ a \in \Sigma,\ i \in \{0, 1\} \right\}$
Q, c wie in $\mathcal{A}$ (wollen Akzeptanz von $\mathcal{A}$ prüfen)
$\Delta' = \left\{ \left(q,\ \langle f, a, i \rangle,\ q'_i\right) \;\middle|\; \langle f, a, i \rangle \in \Sigma',\ i \in \{0, 1\},\ \text{es gibt } \delta = (q, a, q'_0, q'_1) \in \Delta \text{ mit } f(\delta) = i \right\}$
$\mathcal{A}'$ prüft für *jeden möglichen* Zug von **Aut**, ob **Aut** gewinnen kann

16:58 $\rightarrow$ mit Beweis bis 17:30

Achtung: $f \in F$ bedeutet „Funktion aus der Menge aller Funktionen (Gewinnstrategien)“, nicht „akzeptierender Zustand“!

Konstruktion des Wortautomaten für Gewinnbäume

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum
Konstruieren NPA $\mathcal{A}' = (Q, \Sigma', \Delta', \{q_I\}, c)$ wie folgt:

- $\Sigma' = \Big\{\langle f, a, i\rangle \;\Big|\; f \in F,\ a \in \Sigma,\ i \in \{0,1\}\Big\}$
- Q, c wie in $\mathcal{A}$ (wollen Akzeptanz von $\mathcal{A}$ prüfen)
- $\Delta' = \Big\{\big(q,\ \langle f, a, i\rangle,\ q_i'\big) \;\Big|\; \langle f, a, i\rangle \in \Sigma',\ i \in \{0,1\},$ es gibt $\delta = (q, a, q_0', q_1') \in \Delta$ mit $f(\delta) = i\Big\}$

$\mathcal{A}'$ prüft für *jeden möglichen* Zug von **Aut**, ob **Aut** gewinnen kann

Lemma 4.22

s ist ein **PF**-Gewinnbaum für t $\Leftrightarrow$ $L_{s,t} \cap L_\omega(\mathcal{A}') = \emptyset$

Konstruktion des Wortautomaten für Gewinnbäume
Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum
Konstruieren NPA $\mathcal{A}' = (Q, \Sigma', \Delta', \{q_I\}, c)$ wie folgt:
$\Sigma' = \Big\{\langle f, a, i\rangle \;\Big|\; f \in F,\ a \in \Sigma,\ i \in \{0,1\}\Big\}$
Q, c wie in $\mathcal{A}$ (wollen Akzeptanz von $\mathcal{A}$ prüfen)
$\Delta' = \Big\{\big(q,\ \langle f, a, i\rangle,\ q_i'\big) \;\Big|\; \langle f, a, i\rangle \in \Sigma',\ i \in \{0,1\},$ es gibt $\delta = (q, a, q_0', q_1') \in \Delta$ mit $f(\delta) = i\Big\}$
$\mathcal{A}'$ prüft für *jeden möglichen* Zug von **Aut**, ob **Aut** gewinnen kann
Lemma 4.22
s ist ein **PF**-Gewinnbaum für t $\Leftrightarrow$ $L_{s,t} \cap L_\omega(\mathcal{A}') = \emptyset$

16:58 $\rightarrow$ mit Beweis bis 17:30

Achtung: $f \in F$ bedeutet „Funktion aus der Menge aller Funktionen (Gewinnstrategien)“, nicht „akzeptierender Zustand“!

Konstruktion des Wortautomaten für Gewinnbäume

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum

Konstruieren NPA $\mathcal{A}' = (Q, \Sigma', \Delta', \{q_I\}, c)$ wie folgt:

- $\Sigma' = \left\{\langle f, a, i\rangle \;\middle|\; f \in F,\ a \in \Sigma,\ i \in \{0,1\}\right\}$
- Q, c wie in $\mathcal{A}$ (wollen Akzeptanz von $\mathcal{A}$ prüfen)
- $\Delta' = \Big\{\big(q,\ \langle f, a, i\rangle,\ q_i'\big) \;\Big|\; \langle f, a, i\rangle \in \Sigma',\ i \in \{0,1\},$ es gibt $\delta = (q, a, q_0', q_1') \in \Delta$ mit $f(\delta) = i\Big\}$

$\mathcal{A}'$ prüft für *jeden möglichen* Zug von **Aut**, ob **Aut** gewinnen kann

Lemma 4.22

s ist ein **PF**-Gewinnbaum für $t \;\Leftrightarrow\; L_{s,t} \cap L_\omega(\mathcal{A}') = \emptyset$

Beweis: s. Tafel **T 4.14** □

2019-02-01

Teil 4: unendliche Bäume
└─Komplementierung
└─Konstruktion des Wortautomaten für Gewinnbäume

Konstruktion des Wortautomaten für Gewinnbäume

Sei $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$ ein NPBA und t ein Σ-Baum
Konstruieren NPA $\mathcal{A}' = (Q, \Sigma', \Delta', \{q_I\}, c)$ wie folgt:
- $\Sigma' = \left\{\langle f, a, i\rangle \;\middle|\; f \in F,\ a \in \Sigma,\ i \in \{0,1\}\right\}$
- Q, c wie in $\mathcal{A}$ (wollen Akzeptanz von $\mathcal{A}$ prüfen)
- $\Delta' = \Big\{\big(q,\ \langle f, a, i\rangle,\ q_i'\big) \;\Big|\; \langle f, a, i\rangle \in \Sigma',\ i \in \{0,1\},$ es gibt $\delta = (q, a, q_0', q_1') \in \Delta$ mit $f(\delta) = i\Big\}$

$\mathcal{A}'$ prüft für *jeden möglichen* Zug von **Aut**, ob **Aut** gewinnen kann

Lemma 4.22
s ist ein **PF**-Gewinnbaum für $t \;\Leftrightarrow\; L_{s,t} \cap L_\omega(\mathcal{A}') = \emptyset$

Beweis: s. Tafel T 4.14 □

16:58 $\to$ mit Beweis bis 17:30

Achtung: $f \in F$ bedeutet „Funktion aus der Menge aller Funktionen (Gewinnstrategien)“, nicht „akzeptierender Zustand“!

Komplementierung: Was bisher geschah

- Gegeben: **NPBA** $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$
- Ordnen $\mathcal{A}$ und jedem **Eingabebaum** t ein 2-Pers.-Spiel $G_{\mathcal{A},t}$ zu
- Spielerin **Aut** wählt Transition für aktuelle Position in t;
  PF wählt Kindsposition ($\rightsquigarrow$ gibt schrittweise Pfad vor)
- **Aut** gewinnt, wenn gespielter Pfad c entspricht

Lemma 4.17

$t \in L_\omega(\mathcal{A}) \;\Leftrightarrow\;$ **Aut** hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)

Mittels Resultat aus der Spieltheorie folgt:

Folgerung 4.20

$t \in \overline{L_\omega(\mathcal{A})} \;\Leftrightarrow\;$ **PF** hat **gedächtnislose** GS ab (ε, q_I) in $G_{\mathcal{A},t}$

Komplementierung: Was bisher geschah

- Gegeben: **NPBA** $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, c)$
- Ordnen $\mathcal{A}$ und jedem **Eingabebaum** t ein 2-Pers.-Spiel $G_{\mathcal{A},t}$ zu
- Spielerin **Aut** wählt Transition für aktuelle Position in t; **PF** wählt Kindsposition ($\rightsquigarrow$ gibt schrittweise Pfad vor)
- **Aut** gewinnt, wenn gespielter Pfad c entspricht

Lemma 4.17
$t \in L_\omega(\mathcal{A}) \Leftrightarrow$ **Aut** hat Gewinnstrategie in $G_{\mathcal{A},t}$ ab Position (ε, q_I)

Mittels Resultat aus der Spieltheorie folgt:

Folgerung 4.20
$t \in \overline{L_\omega(\mathcal{A})} \Leftrightarrow$ **PF** hat **gedächtnislose** GS ab (ε, q_I) in $G_{\mathcal{A},t}$

8:30

Komplementierung: Was bisher geschah ◀◀

- Betrachten gedächtnislose Gewinnstrategie für **PF** als Menge F von Funktionen

$$f_p : \Delta \to \{0,1\} \qquad \text{für jede Baumposition } p \in \{0,1\}^*$$

- Ordnen die f_p in ***F*-Baum *s*** an
- **PF-Gewinnbaum:** F-Baum für eine Gewinnstrategie von **PF**

Dann folgt sofort:

Folgerung 4.21

$t \in \overline{L_\omega(\mathcal{A})} \;\Leftrightarrow\;$ es gibt einen **PF**-Gewinnbaum für t

Komplementierung: Was bisher geschah ◀◀
- Betrachten gedächtnislose Gewinnstrategie für **PF** als Menge F von Funktionen
$f_p : \Delta \to \{0,1\}$ für jede Baumposition $p \in \{0,1\}^*$
- Ordnen die f_p in ***F*-Baum *s*** an
- **PF-Gewinnbaum:** F-Baum für eine Gewinnstrategie von **PF**
Dann folgt sofort:
Folgerung 4.21
$t \in \overline{L_\omega(\mathcal{A})} \Leftrightarrow$ es gibt einen **PF**-Gewinnbaum für t

8:32

F-Baum auch „Strategiebaum“;
$F \mathrel{\hat{=}}$ alle f_p

Komplementierung: Was bisher geschah ◀◀

Konstruieren NPA $\mathcal{A}'$ (ω-Wortautomat!), um **PF**-Gewinnbäume zu erkennen:

- Eingabewörter haben die Form

$$\left\langle s(\varepsilon), t(\varepsilon), \pi_1 \right\rangle \left\langle s(\pi_1), t(\pi_1), \pi_2 \right\rangle \left\langle s(\pi_1\pi_2), t(\pi_1\pi_2), \pi_3 \right\rangle \ \ldots$$

- Sei $L_{s,t}$ die Menge aller solcher Wörter

Konstruktion von $\mathcal{A}'$ stellt sicher:

Lemma 4.22

s ist ein **PF**-Gewinnbaum für $t \iff L_{s,t} \cap L_\omega(\mathcal{A}') = \emptyset$

Komplementierung: Was bisher geschah ◀◀
Konstruieren NPA $\mathcal{A}'$ (ω-Wortautomat!), um **PF**-Gewinnbäume zu erkennen:
Eingabewörter haben die Form
$\left\langle s(\varepsilon), t(\varepsilon), \pi_1 \right\rangle \left\langle s(\pi_1), t(\pi_1), \pi_2 \right\rangle \left\langle s(\pi_1\pi_2), t(\pi_1\pi_2), \pi_3 \right\rangle \ldots$
Sei $L_{s,t}$ die Menge aller solcher Wörter
Konstruktion von $\mathcal{A}'$ stellt sicher:
Lemma 4.22
s ist ein **PF**-Gewinnbaum für $t \iff L_{s,t} \cap L_\omega(\mathcal{A}') = \emptyset$

8:34

Konstruktion des Komplementautomaten für $\mathcal{A}$

Gesucht: (siehe Folgerung 4.21)

NPBA $\mathcal{B}$, der t akzeptiert gdw. es einen **PF**-Gewinnbaum für t gibt

Konstruktion des Komplementautomaten für $\mathcal{A}$
Gesucht: (siehe Folgerung 4.21)
NPBA $\mathcal{B}$, der t akzeptiert gdw. es einen PF-Gewinnbaum für t gibt

8:36 bis 8:40

„$L_{s,t} \subseteq \overline{L_\omega(\mathcal{A}')}$“: ist äquivalent zu „$L_{s,t} \cap L_\omega(\mathcal{A}') = \emptyset$“ aus L. 4.22

Zu Schritt 1:
NPA $\to$ NMA $\to^*$ NBA —Safra$\to^*$ DRA $\to$ DMA $\to^*$ DPA
*Resultate aus Vorlesung, exp. Blowup
Es gibt effizientere direkte Konstruktion NPA $\to$ DPA (sage am Ende was dazu).

Zu Schritt 2:
$\mathcal{A}''$ braucht ja als Eingabe Tripel, 1. Komponente $s(\cdot)$ aus Strategie.
Diese Info wird schrittweise geraten.

Konstruktion des Komplementautomaten für $\mathcal{A}$

Gesucht: (siehe Folgerung 4.21)

NPBA $\mathcal{B}$, der t akzeptiert gdw. es einen **PF**-Gewinnbaum für t gibt

Wegen Lemma 4.22 muss $\mathcal{B}$ akzeptieren gdw. $L_{s,t} \subseteq \overline{L_\omega(\mathcal{A}')}$

Gesucht: (siehe Folgerung 4.21)
NPBA $\mathcal{B}$, der t akzeptiert gdw. es einen PF-Gewinnbaum für t gibt
Wegen Lemma 4.22 muss $\mathcal{B}$ akzeptieren gdw. $L_{s,t} \subseteq \overline{L_\omega(\mathcal{A}')}$

8:36 bis 8:40

„$L_{s,t} \subseteq \overline{L_\omega(\mathcal{A}')}$“: ist äquivalent zu „$L_{s,t} \cap L_\omega(\mathcal{A}') = \emptyset$“ aus L. 4.22

Zu Schritt 1:
NPA $\rightarrow$ NMA $\rightarrow^*$ NBA —Safra$\rightarrow^*$ DRA $\rightarrow$ DMA $\rightarrow^*$ DPA
*Resultate aus Vorlesung, exp. Blowup
Es gibt effizientere direkte Konstruktion NPA $\rightarrow$ DPA (sage am Ende was dazu).

Zu Schritt 2:
$\mathcal{A}''$ braucht ja als Eingabe Tripel, 1. Komponente $s(\cdot)$ aus Strategie.
Diese Info wird schrittweise geraten.

Konstruktion des Komplementautomaten für $\mathcal{A}$

Gesucht: (siehe Folgerung 4.21)

NPBA $\mathcal{B}$, der t akzeptiert gdw. es einen **PF**-Gewinnbaum für t gibt

Wegen Lemma 4.22 muss $\mathcal{B}$ akzeptieren gdw. $L_{s,t} \subseteq \overline{L_\omega(\mathcal{A}')}$

Konstruktion von $\mathcal{B}$ in 2 Schritten:

Schritt 1

- Sei $\mathcal{A}'' = (Q'', \Sigma', \Delta'', q_I'', c'')$ der **D**PA mit $L_\omega(\mathcal{A}'') = \overline{L_\omega(\mathcal{A}')}$
- $\mathcal{A}''$ ist **deterministisch:** Safra-Konstruktion (+ Umwandlung zwischen den Automatentypen)

Konstruktion des Komplementautomaten für $\mathcal{A}$

Gesucht: (siehe Folgerung 4.21)
NPBA $\mathcal{B}$, der t akzeptiert gdw. es einen PF-Gewinnbaum für t gibt

Wegen Lemma 4.22 muss $\mathcal{B}$ akzeptieren gdw. $L_{s,t} \subseteq \overline{L_\omega(\mathcal{A}')}$

Konstruktion von $\mathcal{B}$ in 2 Schritten:
Schritt 1
- Sei $\mathcal{A}'' = (Q'', \Sigma', \Delta'', q_I'', c'')$ der DPA mit $L_\omega(\mathcal{A}'') = \overline{L_\omega(\mathcal{A}')}$
- $\mathcal{A}''$ ist deterministisch: Safra-Konstruktion (+ Umwandlung zwischen den Automatentypen)

8:36 bis 8:40

„$L_{s,t} \subseteq \overline{L_\omega(\mathcal{A}')}$“: ist äquivalent zu „$L_{s,t} \cap L_\omega(\mathcal{A}') = \emptyset$“ aus L. 4.22

Zu Schritt 1:
NPA → NMA →* NBA —Safra→* DRA → DMA →* DPA
*Resultate aus Vorlesung, exp. Blowup
Es gibt effizientere direkte Konstruktion NPA → DPA (sage am Ende was dazu).

Zu Schritt 2:
$\mathcal{A}''$ braucht ja als Eingabe Tripel, 1. Komponente $s(\cdot)$ aus Strategie.
Diese Info wird schrittweise geraten.

Konstruktion des Komplementautomaten für $\mathcal{A}$

Gesucht: (siehe Folgerung 4.21)

NPBA $\mathcal{B}$, der t akzeptiert gdw. es einen **PF**-Gewinnbaum für t gibt

Wegen Lemma 4.22 muss $\mathcal{B}$ akzeptieren gdw. $L_{s,t} \subseteq \overline{L_\omega(\mathcal{A}')}$

Konstruktion von $\mathcal{B}$ in 2 Schritten:

Schritt 1

- Sei $\mathcal{A}'' = (Q'', \Sigma', \Delta'', q''_I, c'')$ der **D**PA mit $L_\omega(\mathcal{A}'') = \overline{L_\omega(\mathcal{A}')}$
- $\mathcal{A}''$ ist **deterministisch:** Safra-Konstruktion
  (+ Umwandlung zwischen den Automatentypen)

Schritt 2

$\mathcal{B}$ soll auf jedem Pfad von t

- $\mathcal{A}''$ laufen lassen
- und „parallel“ dazu eine Strategie für **PF** raten

Konstruktion des Komplementautomaten für $\mathcal{A}$
Gesucht: (siehe Folgerung 4.21)
NPBA $\mathcal{B}$, der t akzeptiert gdw. es einen PF-Gewinnbaum für t gibt
Wegen Lemma 4.22 muss $\mathcal{B}$ akzeptieren gdw. $L_{s,t} \subseteq \overline{L_\omega(\mathcal{A}')}$
Konstruktion von $\mathcal{B}$ in 2 Schritten:
Schritt 1
Sei $\mathcal{A}'' = (Q'', \Sigma', \Delta'', q''_I, c'')$ der DPA mit $L_\omega(\mathcal{A}'') = \overline{L_\omega(\mathcal{A}')}$
$\mathcal{A}''$ ist deterministisch: Safra-Konstruktion
(+ Umwandlung zwischen den Automatentypen)
Schritt 2
$\mathcal{B}$ soll auf jedem Pfad von t
$\mathcal{A}''$ laufen lassen
und „parallel“ dazu eine Strategie für PF raten

8:36 bis 8:40

„$L_{s,t} \subseteq \overline{L_\omega(\mathcal{A}')}$“: ist äquivalent zu „$L_{s,t} \cap L_\omega(\mathcal{A}') = \emptyset$“ aus L. 4.22

Zu Schritt 1:
NPA $\to$ NMA $\to^*$ NBA —Safra$\to^*$ DRA $\to$ DMA $\to^*$ DPA
*Resultate aus Vorlesung, exp. Blowup
Es gibt effizientere direkte Konstruktion NPA $\to$ DPA (sage am Ende was dazu).

Zu Schritt 2:
$\mathcal{A}''$ braucht ja als Eingabe Tripel, 1. Komponente $s(\cdot)$ aus Strategie.
Diese Info wird schrittweise geraten.

Konstruktion des Komplementautomaten für $\mathcal{A}$

Idee: NPBA $\mathcal{B}$ soll $\mathcal{A}''$ auf **jedem** Pfad simulieren, indem $\mathcal{B}$

- s rät (also pro Position p ein f_p)
- sich ansonsten wie $\mathcal{A}''$ verhält,
  (also pro Position die Folgezustände q_0, q_1 gemäß Δ'' setzt)

$\mathcal{A}''$ deterministisch $\Rightarrow$ Zustand pro Position p eindeutig bestimmt

Konstruktion des Komplementautomaten für $\mathcal{A}$
Idee: NPBA $\mathcal{B}$ soll $\mathcal{A}''$ auf jedem Pfad simulieren, indem $\mathcal{B}$
- s rät (also pro Position p ein f_p)
- sich ansonsten wie $\mathcal{A}''$ verhält, (also pro Position die Folgezustände q_0, q_1 gemäß Δ'' setzt)
$\mathcal{A}''$ deterministisch $\Rightarrow$ Zustand pro Position p eindeutig bestimmt

8:40 $\rightarrow$ bis 9:10

DPA $\mathcal{A}''$ „bezeugt“, dass s *kein* **PF**-Gewinnbaum für t ist

„$\mathcal{A}''$ deterministisch …“: obwohl ∞ viele Pfade durch p gehen!

Konstruktion des Komplementautomaten für $\mathcal{A}$

Idee: NPBA $\mathcal{B}$ soll $\mathcal{A}''$ auf **jedem** Pfad simulieren, indem $\mathcal{B}$

- s rät (also pro Position p ein f_p)
- sich ansonsten wie $\mathcal{A}''$ verhält,
  (also pro Position die Folgezustände q_0, q_1 gemäß Δ'' setzt)

$\mathcal{A}''$ deterministisch $\Rightarrow$ Zustand pro Position p eindeutig bestimmt

Konstruktion von $\mathcal{B} = (Q'', \Sigma, \Delta^{\text{neu}}, q_I'', c'')$:

- Q'', q_I'', c'' werden von $\mathcal{A}''$ übernommen
- $\Delta^{\text{neu}} = \Big\{(q, a, q_0, q_1) \;\Big|\; \text{es gibt } f \in F \text{ mit } \big(q, \langle f, a, i\rangle, q_i\big) \in \Delta'' \text{ für } i = 0,1\Big\}$

Konstruktion des Komplementautomaten für $\mathcal{A}$

Idee: NPBA $\mathcal{B}$ soll $\mathcal{A}''$ auf jedem Pfad simulieren, indem $\mathcal{B}$
- s rät (also pro Position p ein f_p)
- sich ansonsten wie $\mathcal{A}''$ verhält, (also pro Position die Folgezustände q_0, q_1 gemäß Δ'' setzt)

$\mathcal{A}''$ deterministisch $\Rightarrow$ Zustand pro Position p eindeutig bestimmt

Konstruktion von $\mathcal{B} = (Q'', \Sigma, \Delta^{\text{neu}}, q_I'', c'')$:
- Q'', q_I'', c'' werden von $\mathcal{A}''$ übernommen
- $\Delta^{\text{neu}} = \Big\{(q, a, q_0, q_1) \;\Big|\; \text{es gibt } f \in F \text{ mit } \big(q, \langle f, a, i\rangle, q_i\big) \in \Delta'' \text{ für } i = 0,1\Big\}$

8:40 $\longrightarrow$ bis 9:10

DPA $\mathcal{A}''$ „bezeugt", dass s *kein* **PF**-Gewinnbaum für t ist

„$\mathcal{A}''$ deterministisch ...": obwohl ∞ viele Pfade durch p gehen!

$f \in F$: Für *jede mögliche* Funktion $f : \Delta \to \{0,1\}$ („... s rät"!)
und zugehörigen Übergang in Δ'', ein neuer Übergang

Konstruktion des Komplementautomaten für $\mathcal{A}$

Idee: NPBA $\mathcal{B}$ soll $\mathcal{A}''$ auf **jedem** Pfad simulieren, indem $\mathcal{B}$

- s rät (also pro Position p ein f_p)
- sich ansonsten wie $\mathcal{A}''$ verhält,
  (also pro Position die Folgezustände q_0, q_1 gemäß Δ'' setzt)

$\mathcal{A}''$ deterministisch $\Rightarrow$ Zustand pro Position p eindeutig bestimmt

Konstruktion von $\mathcal{B} = (Q'', \Sigma, \Delta^{\text{neu}}, q_I'', c'')$:

- Q'', q_I'', c'' werden von $\mathcal{A}''$ übernommen
- $\Delta^{\text{neu}} = \left\{(q, a, q_0, q_1) \;\middle|\; \text{es gibt } f \in F \text{ mit } \big(q, \langle f, a, i\rangle, q_i\big) \in \Delta'' \text{ für } i = 0, 1\right\}$

Lemma 4.23

$t \in L_\omega(\mathcal{B})$ gdw. es gibt F-Baum s mit $L_{s,t} \subseteq L_\omega(\mathcal{A}'')$

Konstruktion des Komplementautomaten für $\mathcal{A}$

Idee: NPBA $\mathcal{B}$ soll $\mathcal{A}''$ auf jedem Pfad simulieren, indem $\mathcal{B}$
- s rät (also pro Position p ein f_p)
- sich ansonsten wie $\mathcal{A}''$ verhält, (also pro Position die Folgezustände q_0, q_1 gemäß Δ'' setzt)

$\mathcal{A}''$ deterministisch $\Rightarrow$ Zustand pro Position p eindeutig bestimmt

Konstruktion von $\mathcal{B} = (Q'', \Sigma, \Delta^{\text{neu}}, q_I'', c'')$:
- Q'', q_I'', c'' werden von $\mathcal{A}''$ übernommen
- $\Delta^{\text{neu}} = \left\{(q, a, q_0, q_1) \;\middle|\; \text{es gibt } f \in F \text{ mit } \big(q, \langle f, a, i\rangle, q_i\big) \in \Delta'' \text{ für } i = 0, 1\right\}$

Lemma 4.23
$t \in L_\omega(\mathcal{B})$ gdw. es gibt F-Baum s mit $L_{s,t} \subseteq L_\omega(\mathcal{A}'')$

8:40 $\longrightarrow$ bis 9:10

DPA $\mathcal{A}''$ „bezeugt“, dass s *kein* **PF**-Gewinnbaum für t ist

„$\mathcal{A}''$ deterministisch …“: obwohl ∞ viele Pfade durch p gehen!

$f \in F$: Für *jede mögliche* Funktion $f : \Delta \to \{0, 1\}$ („… s rät“!) und zugehörigen Übergang in Δ'', ein neuer Übergang

Lemma ankündigen mit: „es bleibt zu zeigen …“

Konstruktion des Komplementautomaten für $\mathcal{A}$

Idee: NPBA $\mathcal{B}$ soll $\mathcal{A}''$ auf **jedem** Pfad simulieren, indem $\mathcal{B}$

- s rät (also pro Position p ein f_p)
- sich ansonsten wie $\mathcal{A}''$ verhält,
  (also pro Position die Folgezustände q_0, q_1 gemäß Δ'' setzt)

$\mathcal{A}''$ deterministisch $\Rightarrow$ Zustand pro Position p eindeutig bestimmt

Konstruktion von $\mathcal{B} = (Q'', \Sigma, \Delta^{\text{neu}}, q_I'', c'')$:

- Q'', q_I'', c'' werden von $\mathcal{A}''$ übernommen
- $\Delta^{\text{neu}} = \Big\{(q, a, q_0, q_1) \;\Big|\; \text{es gibt } f \in F \text{ mit } \big(q, \langle f, a, i\rangle, q_i\big) \in \Delta'' \text{ für } i = 0, 1\Big\}$

Lemma 4.23

$t \in L_\omega(\mathcal{B})$ gdw. es gibt F-Baum s mit $L_{s,t} \subseteq L_\omega(\mathcal{A}'')$

Beweis: siehe Tafel **T 4.15** □

Konstruktion des Komplementautomaten für $\mathcal{A}$
Idee: NPBA $\mathcal{B}$ soll $\mathcal{A}''$ auf jedem Pfad simulieren, indem $\mathcal{B}$ … Lemma 4.23 … Beweis: siehe Tafel T 4.15 □

8:40 $\rightarrow$ bis 9:10

DPA $\mathcal{A}''$ „bezeugt“, dass s *kein* **PF**-Gewinnbaum für t ist

„$\mathcal{A}''$ deterministisch …“: obwohl ∞ viele Pfade durch p gehen!

$f \in F$: Für *jede mögliche* Funktion $f : \Delta \to \{0, 1\}$ („… s rät“!) und zugehörigen Übergang in Δ'', ein neuer Übergang

Lemma ankündigen mit: „es bleibt zu zeigen …“

. . . Es darf aufgeatmet werden . . . ☺

. . . Es darf aufgeatmet werden . . . ☺

9:10

Jetzt haben wir alles Technische geschafft und können das Hauptresultat sehr einfach beweisen.

Danach haben wir uns eine Pause verdient. :)

Das Resultat

Satz 4.24 (Rabin 1969)

Für jeden NPBA $\mathcal{A}$ gibt es einen NPBA $\mathcal{B}$ mit $L_\omega(\mathcal{B}) = \overline{L_\omega(\mathcal{A})}$.

Das Resultat
Satz 4.24 (Rabin 1969)
Für jeden NPBA $\mathcal{A}$ gibt es einen NPBA $\mathcal{B}$ mit $L_\omega(\mathcal{B}) = \overline{L_\omega(\mathcal{A})}$.

9:10 bis 9:13

Jetzt müssen wir nur noch alle Lemmas zusammentun und erhalten das Hauptresultat.

„Konstr. $\mathcal{A}''$“: ist Komplementärautomat zu $\mathcal{A}'$

TODO: Wenn Zeit ist, Beispiel für die Konstruktion vorführen (siehe Notizen in Sammlung nach T4.15, zweites=einfacheres Bsp.)

Das Resultat

Satz 4.24 (Rabin 1969)

Für jeden NPBA $\mathcal{A}$ gibt es einen NPBA $\mathcal{B}$ mit $L_\omega(\mathcal{B}) = \overline{L_\omega(\mathcal{A})}$.

Beweis:

Für den bisher konstruierten NPBA $\mathcal{B}$ gilt:

$$
\begin{array}{rll}
\boldsymbol{t \in L_\omega(\mathcal{B})} & \text{gdw. } \exists s \,.\, L_{s,t} \subseteq L_\omega(\mathcal{A}'') & \text{(Lemma 4.23)} \\
& \text{gdw. } \exists s \,.\, L_{s,t} \subseteq \overline{L_\omega(\mathcal{A}')} & \text{(Konstr. } \mathcal{A}''\text{)} \\
& \text{gdw. } \exists s \,.\, L_{s,t} \cap L_\omega(\mathcal{A}') = \emptyset & \text{(Mengenlehre)} \\
& \text{gdw. } \exists\, \textbf{PF}\text{-Gewinnbaum } s \text{ für } t & \text{(Lemma 4.22)} \\
& \text{gdw. } \boldsymbol{t \in \overline{L_\omega(\mathcal{A})}} & \text{(Folg. 4.21)}
\end{array}
$$

□

Das Resultat
Satz 4.24 (Rabin 1969)
Für jeden NPBA $\mathcal{A}$ gibt es einen NPBA $\mathcal{B}$ mit $L_\omega(\mathcal{B}) = \overline{L_\omega(\mathcal{A})}$.
Beweis:
Für den bisher konstruierten NPBA $\mathcal{B}$ gilt:
$t \in L_\omega(\mathcal{B})$ gdw. $\exists s \,.\, L_{s,t} \subseteq L_\omega(\mathcal{A}'')$ (Lemma 4.23)
gdw. $\exists s \,.\, L_{s,t} \subseteq \overline{L_\omega(\mathcal{A}')}$ (Konstr. $\mathcal{A}''$)
gdw. $\exists s \,.\, L_{s,t} \cap L_\omega(\mathcal{A}') = \emptyset$ (Mengenlehre)
gdw. $\exists$ PF-Gewinnbaum s für t (Lemma 4.22)
gdw. $t \in \overline{L_\omega(\mathcal{A})}$ (Folg. 4.21)
□

9:10 bis 9:13

Jetzt müssen wir nur noch alle Lemmas zusammentun und erhalten das Hauptresultat.

„Konstr. $\mathcal{A}''$“: ist Komplementärautomat zu $\mathcal{A}'$

TODO: Wenn Zeit ist, Beispiel für die Konstruktion vorführen
(siehe Notizen in Sammlung nach T4.15, zweites=einfacheres Bsp.)

Bemerkungen zur Komplexität der Konstruktion

Sei $n = |Q|$ (Anzahl der Zustände des NBPA $\mathcal{A}$).

Dann hat der NPA $\mathcal{A}'$ dieselben n Zustände.

DPA $\mathcal{A}''$ kann so konstruiert werden, dass $|Q''| \in O(2^{n \log n})$.

$\rightsquigarrow$ NBPA $\mathcal{B}$ hat $O(2^{n \log n})$ Zustände.

Bemerkungen zur Komplexität der Konstruktion
Sei $n = |Q|$ (Anzahl der Zustände des NBPA $\mathcal{A}$).
Dann hat der NPA $\mathcal{A}'$ dieselben n Zustände.
DPA $\mathcal{A}''$ kann so konstruiert werden, dass $|Q''| \in O(2^{n \log n})$.
$\rightsquigarrow$ NBPA $\mathcal{B}$ hat $O(2^{n \log n})$ Zustände.

9:13; auf Literatur verweisen, dann fertig? Oder noch Alternierung beginnen?

Punkt 3 folgt **nicht** aus den Resultaten dieser Vorlesung.

Naives Vorgehen:
NPA $\to$ NMA $\to^*$ NBA —Safra$\to^*$ DRA $\to$ DMA $\to^*$ DPA
*Resultate aus Vorlesung, exp. Blowup
Es gibt offenbar eine effizientere direkte Konstruktion NPA $\to$ DPA

Literatur für diesen Teil (Basis, 1)

- E. Grädel, W. Thomas, T. Wilke (Hrsg.).
  Automata, Logics, and Infinite Games.
  LNCS 2500, Springer, 2002, S. 43–60.
  Kapitel 6–9 über Paritätsspiele und Baumautomaten.
  `http://www.cs.tau.ac.il/~rabinoa/Lncs2500.zip`
  Auch erhältlich auf Anfrage in der BB Mathematik im MZH:
  `19h inf 001 k/100-2500`

- Meghyn Bienvenu.
  Automata on Infinite Words and Trees.
  Vorlesungsskript, Uni Bremen, WS 2009/10.
  Kapitel 4.
  `http://www.informatik.uni-bremen.de/tdki/lehre/ws09/automata/automata-notes.pdf`

Literatur für diesen Teil (Basis, 1)

E. Grädel, W. Thomas, T. Wilke (Hrsg.).
Automata, Logics, and Infinite Games.
LNCS 2500, Springer, 2002, S. 43–60.
Kapitel 6–9 über Paritätsspiele und Baumautomaten.
http://www.cs.tau.ac.il/~rabinoa/Lncs2500.zip
Auch erhältlich auf Anfrage in der BB Mathematik im MZH:
19h inf 001 k/100-2500

Meghyn Bienvenu.
Automata on Infinite Words and Trees.
Vorlesungsskript, Uni Bremen, WS 2009/10.
Kapitel 4.
http://www.informatik.uni-bremen.de/tdki/lehre/ws09/automata/automata-notes.pdf

Literatur für diesen Teil (Basis, 2)

- Christel Baier, Joost-Pieter Katoen.
  Principles of Model Checking.
  MIT Press 2008.
  Abschnitt 6 „Computation Tree Logic“.
  SUB, Zentrale: `a inf 440 ver/782`, `a inf 440 ver/782a`

- Edmund M. Clarke, Orna Grumberg, Doron A. Peled.
  Model Checking.
  MIT Press 1999.
  Abschnitt 3 „Temporal Logics“,
  Abschnitt 4 „Model Checking“.
  SUB, Zentrale: `a inf 440 ver/780(6)`, `a inf 440 ver/780(6)a`

Literatur für diesen Teil (Basis, 2)
Christel Baier, Joost-Pieter Katoen.
Principles of Model Checking.
MIT Press 2008.
Abschnitt 6 „Computation Tree Logic“.
SUB, Zentrale: a inf 440 ver/782, a inf 440 ver/782a
Edmund M. Clarke, Orna Grumberg, Doron A. Peled.
Model Checking.
MIT Press 1999.
Abschnitt 3 „Temporal Logics“,
Abschnitt 4 „Model Checking“.
SUB, Zentrale: a inf 440 ver/780(6), a inf 440 ver/780(6)a

Literatur für diesen Teil (weiterführend)

Edmund M. Clarke, I. A. Draghicescu
Expressibility Results for Linear-Time and Branching-Time Logics
REX Workshop 1988, S. 428–437.
https://doi.org/10.1007/BFb0013029

2019-02-01
Teil 4: unendliche Bäume
└─Literatur für diesen Teil (weiterführend)

Literatur für diesen Teil (weiterführend)
Edmund M. Clarke, I. A. Draghicescu
Expressibility Results for Linear-Time and Branching-Time Logics
REX Workshop 1988, S. 428–437.
https://doi.org/10.1007/BFb0013029

Teil 4: unendliche Bäume