Automatentheorie und ihre Anwendungen
Teil 4: Automaten auf unendlichen Bäumen

Thomas Schneider

1.–15. Juli 2015

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 ⚠

Und nun ...

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."
  $G(e \rightarrow F\neg e)$ ($e \in$ 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)

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.“
$G(e \rightarrow F\neg e)$ oder $GF\neg e$ sind

- zu stark in Verbindung mit universellem Model-Checking
- zu schwach in Verbindung mit existenziellem MC

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 $s_0 \in S_0$

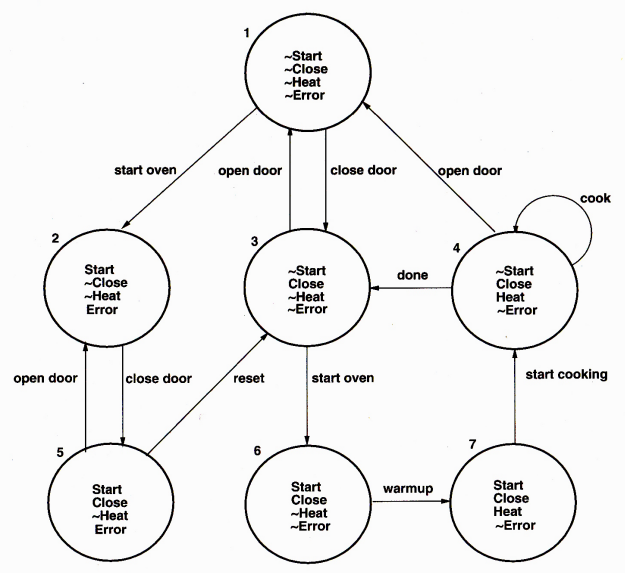
- entsteht durch „Auffalten“ von $\mathcal{S}$ in s_0
- enthält *alle unendlichen* Pfade, die in s_0 starten

  d. h.: jeder Zustand $s \in S$ hat als Kinder
  alle seine Nachfolgerzustände aus $\mathcal{S}$

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

Beispiel: siehe nächste Folie & Tafel

Beispielstruktur Mikrowelle



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

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 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 e. temporalen Operator gefolgt werden
- jeder temporale Operator direkt einem Pfadquantor folgen

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)
$EFA(p \, U \, q)$	✓	

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 1

Erfülltheit von Zustandsformeln in Zuständen $s \in S$

$s \models p$	falls	$p \in \ell(s)$, für alle $p \in$ 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 $\pi \in$ Paths(s)	
$s \models A\psi$	falls	$\pi \models \psi$ für alle $\pi \in$ 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 $i \leqslant k < j$	

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

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 Mikrowelle

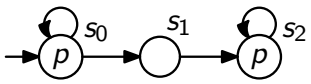
- „Wenn Fehler auftritt, ist er nach endlicher Zeit behoben.“
  $AG(e \to AF\neg e)$ ($e \in$ AV steht für „Error“)
- „Wenn Fehler auftritt, *kann* er nach endl. Z. behoben werden“
  $AG(e \to EF\neg e)$
- „Wenn die Mikrowelle gestartet wird,
  beginnt sie nach endlicher Zeit zu heizen.“
  $AG(s \to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

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

CTL und LTL bezüglich Ausdrucksstärke unvergleichbar!

z. B. sind $AFAGp$ und FGp **nicht äquivalent**:

Betrachte $\mathcal{S}$ wie rechts



- 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 $s_0^\omega \in \text{Paths}(s_0)$

Insbesondere gibt es

- keine zu $AFAGp$ äquivalente LTL-Formel (o. Beweis)
- keine zu FGp äquivalente CTL-Formel (o. Beweis)

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:

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

Komplexität: P-vollständig
(zur Erinnerung: LTL-MC ist PSPACE-vollständig)

Model-Checking für CTL mit Baumautomaten

Automatenbasierte Entscheidungsprozedur für CTL

- basiert auf **alternierenden Baumautomaten**
  (Erweiterung des Begriffs der nichtdeterminist. Baumautomaten)
- hier nicht behandelt

Verwandt:

Automatenbasierte Entscheidungsprozedur für CTL*-**Erfüllbarkeit**

- basiert auf „klassischen“ Rabin-Baumautomaten
- technisch aufwändige Konstruktion
- hier ebenfalls nicht behandelt

Es folgt:

Überblick klassische nichtdeterministische Baumautomaten

Und nun ...

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 $p \in \{0,1\}^*$

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

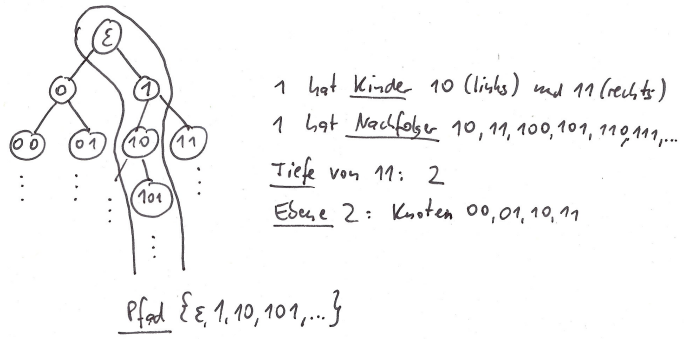
Σ-Baum t (Alphabet Σ ohne Stelligkeit):

Funktion $t : \{0,1\}^* \to \Sigma$

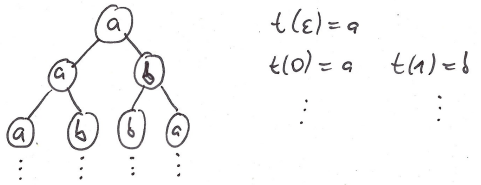
Skizze: siehe Tafel

Skizzen zu den Grundbegriffen

Positionen und Pfade im Binärbaum



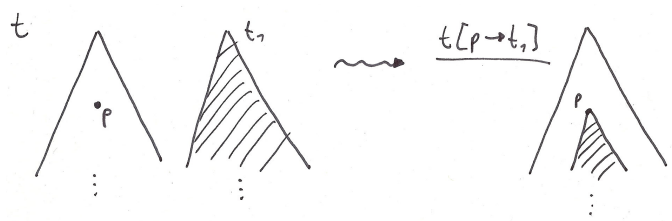
Beispiel-Σ-Baum,
$\Sigma = \{a, b\}$



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:



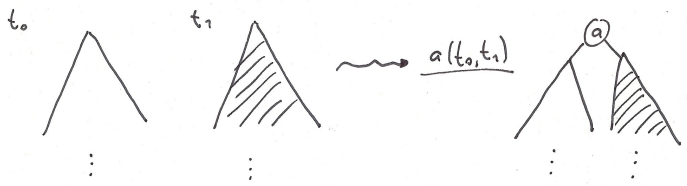
exakte Beschreibung:

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

Baumautomaten: etwas mehr Notation (2)

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

Definition 2

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 **Endzustände** ist.

(entsprechen offenbar Top-down-Automaten)

Muller- und Paritäts-Baumautomaten

Definition 3

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)

Runs auf Baumautomaten

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

Definition 4

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

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 5

- 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

Akzeptanz und erkannte Sprache

... sind wie üblich definiert:

Definition 6

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

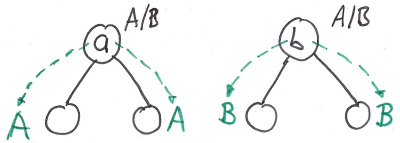
- $\mathcal{A}$ **akzeptiert** t,
  wenn es einen erfolgreichen Run von $\mathcal{A}$ auf t gibt.
- $L_\omega(\mathcal{A}) = \{t \mid \mathcal{A} \text{ akzeptiert } t\}$
- L heißt **Büchi-erkennbar**,
  wenn es einen NBBA $\mathcal{A}$ gibt mit $L_\omega(\mathcal{A}) = L$.
- Analog: **Muller-erkennbar**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



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

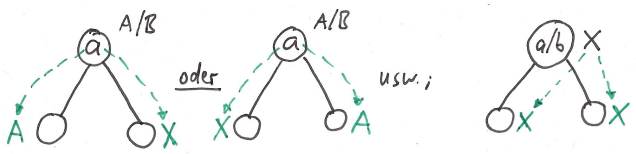
- derselbe NBBA, aber mit $F = \{B\}$
  $L_\omega(\mathcal{A}) = \{t \mid \text{jeder Pfad hat } \infty \text{ viele } b\text{'s}\}$
- derselbe NBBA, aber mit $F = \{A,B\}$
  $L_\omega(\mathcal{A}) = \{t \mid t \text{ ist ein } \Sigma\text{-Baum}\}$

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:



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

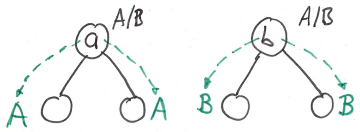
- derselbe NBBA, aber mit $F = \{B, X\}$
  $L_\omega(\mathcal{A}) = \{t \mid t \text{ hat mind. einen Pfad mit } \infty \text{ vielen } b\text{'s}\}$
- derselbe NBBA, aber mit $F = \{X\}$: $\quad L_\omega(\mathcal{A}) = \emptyset$
- derselbe NBBA, aber mit $F = \{A, B\}$

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:

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

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 7

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

Beweis.

1. Wie im letzten Kapitel.
2. Idee:
   - Betrachten $L = \{t \mid \text{jeder Pfad in } t \text{ hat endlich viele } a\text{'s}\}$; nehmen an, L werde von NBBA $\mathcal{A}$ mittels Run r erkannt
   - Bestimme Baum $t \in L$ und Pfad, auf dem zwischen zwei Besuchen *desselben* EZ ein a auftritt
   - „Pumpe“ t, r so auf, dass dieser Teilpfad sich ∞ oft wiederholt
   - ↯ Neuer Baum wird akzeptiert, aber neuer Pfad hat ∞ viele a's

Details: s. Tafel ● □

Folgerung aus dem Beweis von Satz 7

Folgerung 8

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

Man kann Satz 7 stärker formulieren (ohne Beweis):

Satz 9

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

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

($L \subseteq \Sigma^\omega$ ist eine ω-Sprache

L_Δ = Menge aller Σ-Bäume, deren Beschriftung entlang *jedes* Pfades in L liegt)

Paritäts- versus Muller-Erkennbarkeit

Satz 10

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

Beweis.

1. Wie im letzten Kapitel.
2. Idee:
   - Wandeln NMBA $\mathcal{A}$ mit n Zuständen in NPBA $\mathcal{A}'$ um
   - $\mathcal{A}'$ „merkt sich“, in welcher Reihenfolge die n Zustände zuletzt gesehen wurden (Permutation $q_1 \cdots q_n$ von Q)
   - $\mathcal{A}'$ stellt sicher, dass ab einem gewissen Zeitpunkt genau die Endzustände von $\mathcal{A}$ dauerhaft am Ende der Permutation stehen

Details der Konstruktion (1)

Sei $\mathcal{A} = (Q, \Sigma, \Delta, I, \mathcal{F})$ NMBA mit $|Q| = n$.

Konstruieren NPBA $\mathcal{A}' = (Q', \Sigma, \Delta', I', c)$ mit Zuständen

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

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 ●

Details der Konstruktion (2)

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

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

Dann gilt $\text{Inf}(q_1 q_2 q_3 \ldots) = S$ 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 $\underbrace{n - k + 1, \ldots, n}_{\text{letzte } k \text{ Positionen}}$ in perm_i ist S

Beweis der Hilfsaussage: siehe Tafel ●

Details der Konstruktion (3)

Können nun Konstruktion fortsetzen:

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

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

- $(i, a, i', i'') \in \Delta$
- $i'_1 \cdots i'_{n-1}$ 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 $i_1 \cdots i_{n-1} i$ $\Big\}$

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

Beweis der Korrektheit: siehe Tafel ● □

Abschlusseigenschaften

Satz 11

Die Klasse der ...

1. *Büchi-erkennbaren Sprachen ist abgeschlossen unter $\cup$ und $\cap$, aber nicht unter $^-$.*
2. *Muller-erkennbaren Sprachen ist abgeschlossen unter $\cup, \cap, ^-$.*

Beweis:

1. $\cup\,\cap$ wie gehabt; $^-$ siehe Folgerung 8.
2. $\cup\,\cap$ wie gehabt;
   $^-$ siehe nächsten Abschnitt ● □

Und nun ...

Überblick

Ziel dieses Abschnitts:

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

Vorgehen:

- Ordnen jedem Baumautomaten $\mathcal{A}$ und Baum t ein Zweipersonenspiel $G_{\mathcal{A},t}$ zu
- Dann wird leicht zu sehen sein:
  $\mathcal{A}$ akzeptiert t $\Leftrightarrow$ Spieler 1 hat Gewinnstrategie in $G_{\mathcal{A},t}$
- Ein Resultat aus der Spieltheorie impliziert:
  In $G_{\mathcal{A},t}$ hat immer *genau ein* Spieler eine Gewinnstrategie, *die nicht vom bisherigen Spielverlauf abhängt*
- Konstruieren $\mathcal{A}'$, so dass gilt:
  $\mathcal{A}'$ akzeptiert t $\Leftrightarrow$ Spieler 2 hat Gewinnstrategie in $G_{\mathcal{A},t}$
  Dann folgt $L_\omega(\mathcal{A}') = \overline{L_\omega(\mathcal{A})}$

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

Zwei Spieler **Aut** (Automat), **PF** (Pfadfinder)

- sind abwechselnd an der Reihe
- bewegen sich in jeder Runde einen Schritt im Baum durch Markieren von Positionen mit Zuständen; zu Beginn: (ε, q_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 ●

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**:
  - $(p, q) \to (q', t(p'), q_0, q_1)$, wenn $p = p'$ und $q = q'$
  - $(q, t(p), q_0, q_1) \to (p', q')$, wenn $q' = q_i$ und $p' = pi$ für ein i
- Startknoten: (ε, q_I) für $q_I \in I$ (o. B. d. A. $I = \{q_I\}$)

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

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

Spielstrategien

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

Funktion, die jeder Zugfolge $v \ldots v'$ 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 Spieler $X \in \{\textbf{Aut}, \textbf{PF}\}$:

Strategie, die sicherstellt, dass X gewinnt, *unabhängig* von den Zügen des Gegenspielers

gedächtnislose Strategie:

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

Akzeptanz mittels Gewinnstrategien

Lemma 12

Seien $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, Acc)$ ein NxBA 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

Details: s. Tafel ● □

Determiniertheit von Paritätsspielen

Klassisches Resultat aus der Spieltheorie, hier nicht bewiesen:

Satz 13 (Emerson & Jutla 1991, Mostowski 1991)

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

Folgerung 14

Seien $\mathcal{A} = (Q, \Sigma, \Delta, \{q_I\}, Acc)$ ein NxBA 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 15 (aus Lemma 12 und Folgerung 14)

$t \in \overline{L_\omega(\mathcal{A})} \Leftrightarrow$ **PF** *hat gedächtnislose GS ab (ε, q_I) in $G_{\mathcal{A},t}$*

Ziel: konstruieren NPBA, um deren Existenz zu testen

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)**

Gewinnbaum für t:
ein F-Baum, der eine Gewinnstrategie für **PF** in $G_{\mathcal{A},t}$ kodiert

Folgerung 16 (aus Folgerung 15)

$t \in \overline{L_\omega(\mathcal{A})} \Leftrightarrow$ *es gibt einen Gewinnbaum für t*

Neues Ziel: konstruieren NPBA, um Existenz Gewinnbaum zu testen

Existenz von Gewinnbäumen (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* 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 Gewinnbaum für t

$\mathcal{A}'$ arbeitet auf Wörtern der folgenden Form, mit $\pi = \pi_1\pi_2\pi_3\ldots$:

$$\Big\langle s(\varepsilon), t(\varepsilon), \pi_1 \Big\rangle \; \Big\langle s(\pi_1), t(\pi_1), \pi_2 \Big\rangle \; \Big\langle s(\pi_1\pi_2), t(\pi_1\pi_2), \pi_3 \Big\rangle \; \ldots$$

Sei $L_{s,t}$ die Sprache aller dieser Wörter

Beispiel: s. Tafel •

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)
- $\Delta' = \Big\{ \Big(q,\ \langle f, a, i\rangle,\ q'_i\Big) \;\Big|\; \langle f, a, i\rangle \in \Sigma',\ i \in \{0,1\},$
  $\text{es gibt } \delta = (q, a, q'_0, q'_1) \in \Delta \text{ mit } f(\delta) = i \Big\}$

$\mathcal{A}'$ prüft für *jeden möglichen* Zug von **Aut**, ob **Aut** gewinnen kann

Lemma 17

s ist ein Gewinnbaum für t $\Leftrightarrow$ $L_{s,t} \cap L_\omega(\mathcal{A}') = \emptyset$

Beweis: s. Tafel • □

Konstruktion des Komplementautomaten für $\mathcal{A}$ (1)

Gesucht: (siehe Folgerung 16)
NPBA $\mathcal{B}$, der t akzeptiert gdw. es einen Gewinnbaum für t gibt

Wegen Lemma 17 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

Konstruktion des Komplementautomaten für $\mathcal{A}$ (2)

Sei nun der NPBA $\mathcal{B} = (Q'', \Sigma, \Delta^{\text{neu}}, q_I'', c'')$ wie folgt definiert:

- Q'', q_I'', c'' werden von $\mathcal{A}''$ übernommen
- $\Delta^{\text{neu}} = \Big\{(q, a, q_0, q_1) \;\Big|\; \text{es gibt } f \in F \text{ mit } \big(q, \langle f, a, i\rangle, q_i\big) \in \Delta'' \text{ für } i = 0, 1\Big\}$

Es bleibt zu zeigen:

Lemma 18

$L_\omega(\mathcal{B}) = \overline{L_\omega(\mathcal{A})}$

Beweis: siehe Tafel ● □

. . . Es darf aufgeatmet werden . . . ☺

Das Resultat

Satz 19 (Rabin 1969)

Für jeden NPBA $\mathcal{A}$ gibt es einen NPBA $\mathcal{B}$ mit $L_\omega(\mathcal{B}) = \overline{L_\omega(\mathcal{A})}$.

Beweis: Direkte Konsequenz aus Folg. 16 und Lemmas 17, 18 □

Bemerkungen zur Komplexität der Konstruktion

- Sei $n = |Q|$ (Anzahl der Zustände von $\mathcal{A}$)
- Dann hat $\mathcal{A}'$ dieselben n Zustände
- $\mathcal{A}''$ kann so konstruiert werden, dass $|Q''| \in O(2^{n \log n})$

$\rightsquigarrow$ $\mathcal{B}$ hat $O(2^{n \log n})$ Zustände

Abschließende Bemerkungen

Zusammenfassung

- NBBA sind schwächer als NMBA und NPBA
- Büchi-, Muller- und paritäts-erkennbare Sprachen sind abgeschlossen unter $\cup, \cap$ (leicht)
- Büchi-erkennbare Sprachen sind *nicht* abgeschlossen unter $^-$
- Muller- und paritäts-erkennbare Sprachen *sind* abgeschlossen unter $^-$ (anspruchsvoll, Spieltheorie)
- Komplementierung ist zentral im Beweis der Entscheidbarkeit der monadischen Logik zweiter Stufe (Kapitel 12 in LNCS 2500)

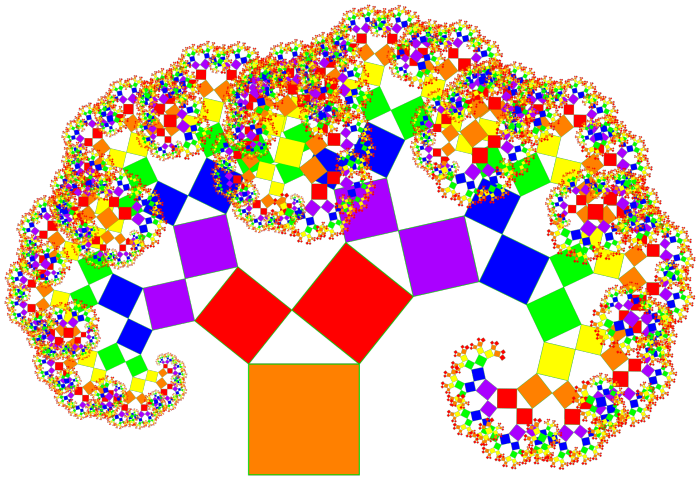
Abschließende Bemerkungen

Ausblick

- Leerheitsproblem für NPBA ist entscheidbar;
  beste bekannte obere Schranke: UP $\cap$ coUP ($\subseteq$ **NP**)
  (Kapitel 8 in LNCS 2500)
- Verallgemeinerung des Nichtdeterminismus:
  alternierende Baumautomaten (Kapitel 9 in LNCS 2500)

Das war . . .

. . . das Kapitel über Automaten auf unendlichen Bäumen.



Pythagoras-Baum. Quelle: Wikipedia, User Gjacquenot (Lizenz CC BY-SA 3.0)

Was jetzt noch fehlt . . .

- kurze Nachbesprechung Vorlesungsevaluation
- Hinweise zu Fachgesprächen/mündl. Prüfung

Vielen Dank für eure Teilnahme!

Literatur für diesen Teil (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 (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