

FGI-1 Repetitorium: Logik / Automatentheorie [Tag 2]

Jan Henrik Röwekamp
(8roeweka)

17.07., 18.07. & 19.07.2013

Ablauf – Tag 2

- Wdh.: Prädikatenlogik & Strukturen
- Prädikatenlogische Resolution
 - Formel schließen und bereinigte Pränexform (BPF)
 - Skolemisierung, Unifikation und Resolution
- Warum Automaten?
- Die Chomsky-Hierarchie
- Reguläre Mengen, DFA's und NFA's
- <Pause>
- Der Potenzautomat: Vom NFA zum DFA
- ϵ -FA & Überführung zum NFA (ϵ -frei machen)
- Kleene-Verfahren

Wiederholung: Prädikatenlogik

- Aufsplitten von Aussagensymbolen in:
 - (komplexe) Terme, Funktionen und Prädikate
- Quantoren gebunden an Variablen
 - Allquantor $\forall$ und Existenzquantor $\exists$
- Strukturen S bestehen aus Universum U und Interpretation I
 - Auswertung einer PL-Formel
 - Berechnung eines Wahrheitswertes

Ablauf – Tag 2

- Wdh.: Prädikatenlogik & Strukturen
- **Prädikatenlogische Resolution**
 - Formel schließen und bereinigte Pränexform (BPF)
 - Skolemisierung, Unifikation und Resolution
- Warum Automaten?
- Die Chomsky-Hierarchie
- Reguläre Mengen, DFA's und NFA's
- <Pause>
- Der Potenzautomat: Vom NFA zum DFA
- ϵ -FA & Überführung zum NFA (ϵ -frei machen)
- Kleene-Verfahren

Prädikatenlogische Resolution

- Unerfüllbarkeits-Nachweis unabhängig von Strukturen und Interpretationen.
- 6 Schritte:
 - Formel bereinigen
 - Formel schließen
 - Pränexform erzeugen
 - Skolemisierung
 - Matrix in KNF umformen
 - Unifikation + Resolution

PL-Resolution: Aufgabe (1)

- Seien die folgenden Formeln gegeben:
 - $F = \forall x \neg P(x,x)$
 - $G = \forall x \forall y \forall z ((P(x,y) \wedge P(y,z)) \Rightarrow P(x,z))$
 - $H = \forall x \forall y (P(x,y) \Rightarrow \neg P(y,x))$
- Weisen Sie mittels prädikatenlogischer Resolution nach, dass $\{F, G\} \models H$ gilt!
- Wir erinnern uns:
 - $\{F, G\} \models H$ gdw. $\{F, G\} \cup \{\neg H\} \models$

Bereinigte Pränexform (BPF)

- Jede PL-Formel kann in BPF gebracht werden.
- Eine Formel ist *bereinigt*, wenn:
 - Alle Quantoren unterschiedliche Variablen aufweisen
 - Keine Variable sowohl frei, als auch gebunden vorkommt.
- Eine Formel ist in *Pränexform*, wenn:
 - Alle Quantoren vorne in der Formel stehen.
 - Der Rest der Formel wird in diesem Fall *Matrix* genannt.

Freie und gebundene Variablen

- Variablen können frei (ohne Quantor) oder gebunden an einen Quantor vorkommen.
- Gebundene Variablen können umbenannt werden („Gebundene Umbenennung“)
- Freie Variablen dürfen **nicht** umbenannt werden!
 - Strukturen interpretieren freie Variablen.
Umbenennung führt zu anderer Interpretation!
 - Gebundene Variablen werden über den Quantor ausgewertet, daher ist der Variablenname egal.

Erfüllbarkeitsäquivalenz

- Resolution: Nachweis von Unerfüllbarkeit
- Exakte Äquivalenz wird nicht benötigt zum Nachweis der Erfüllbarkeit. (Zu Äquivalenzen in der Prädikatenlogik siehe: **Satz 10.2!**)
- Wenn die (Un)erfüllbarkeit einer Formel F gleich der einer Formel G ist, sind G und F *erfüllbarkeitsäquivalent*.
- Wird hier ab sofort als „eä“ abgekürzt.
- Einige Schritte der PL-Resolution verletzen die Äquivalenz, *nicht aber die Eä*.

PL-Resolution: Aufgabe (2)

<BPF erzeugen>

- Formel bisher: $\forall x \neg P(x,x) \wedge$
 $\forall x \forall y \forall z ((P(x,y) \wedge P(y,z)) \Rightarrow P(x,z)) \wedge$
 $\neg \forall x \forall y (P(x,y) \Rightarrow \neg P(y,x))$
- 1) Formel bereinigen (geb. Umben.) [**Äq**]:
 $\forall x \neg P(x,x) \wedge$
 $\forall u \forall y \forall z ((P(u,y) \wedge P(y,z)) \Rightarrow P(u,z)) \wedge$
 $\neg \forall v \forall w (P(v,w) \Rightarrow \neg P(w,v))$
- 2) Formel schließen. [**Eä**]
 - Da keine freien Variablen vorhanden sind, ist nichts zu tun. (Andernfalls Bindung an Existenzquantor)

PL-Resolution: Aufgabe (3)

<BPF erzeugen>

- 3) Pränexform erzeugen: [Äq]

$$\forall x \neg P(x,x) \wedge$$

$$\forall u \forall y \forall z ((P(u,y) \wedge P(y,z)) \Rightarrow P(u,z)) \wedge$$

$$\neg \forall v \forall w (P(v,w) \Rightarrow \neg P(w,v))$$

$$\equiv \forall x \neg P(x,x) \wedge \forall u \forall y \forall z ((P(u,y) \wedge P(y,z)) \Rightarrow P(u,z)) \wedge$$
$$\underline{\exists v} \underline{\exists w} \underline{\neg} (P(v,w) \Rightarrow \neg P(w,v))$$

$$\equiv \forall x \neg P(x,x) \wedge \underline{\exists v} \underline{\exists w} \forall u \forall y \forall z ($$
$$((P(u,y) \wedge P(y,z)) \Rightarrow P(u,z)) \wedge \neg (P(v,w) \Rightarrow \neg P(w,v)))$$

$$\equiv \underline{\exists v} \underline{\exists w} \underline{\forall u} \underline{\forall y} \underline{\forall z} \forall x (\neg P(x,x) \wedge$$
$$((P(u,y) \wedge P(y,z)) \Rightarrow P(u,z)) \wedge \neg (P(v,w) \Rightarrow \neg P(w,v)))$$

Skolemisierung

- Skolemisierung wird verwendet um Existenzquantoren aus der Formel zu entfernen.
- Grundlegende Idee:
 - Resolution ist ein Unerfüllbarkeits-Test-Verfahren
 - Erhaltung der genauen Äquivalenz der Formel ist nicht unbedingt nötig.
 - Lediglich die Erfüllbarkeit muss erhalten bleiben.
 - Wenn die Formel erfüllbar war, gab es zumindest ein Element des Universums für jeden Existenzquantor, um diesen zu erfüllen.

Skolemisierung

- Ersetzung der existenzquantifizierten Variablen mit einem festen Term (z.B. einer Konstanten).
- Abhängigkeit von Allquantoren **vor** dem Existenzquantor muss berücksichtigt werden:
- $\exists x \dots x \quad \rightarrow \quad \dots a$
- $\forall y \exists x \dots x \quad \rightarrow \quad \forall y \dots f(y)$
- $\forall y \forall z \exists x \dots x \quad \rightarrow \quad \forall y \forall z \dots f(y, z)$
- $\forall y \exists x \forall z \dots x \quad \rightarrow \quad \forall y \forall z \dots f(y)$
- Wichtig: Neue Konstanten- und Funktionssymbole dürfen vorher noch nicht vorkommen.

PL-Resolution: Aufgabe (4)

<4) Skolemisierung [Eä]>

- $\exists v \exists w \forall u \forall y \forall z \forall x (\neg P(x,x) \wedge ((P(u,y) \wedge P(y,z)) \Rightarrow P(u,z)) \wedge \neg(P(v,w) \Rightarrow \neg P(w,v)))$
- Sei I die Matrix der obigen Formel.
- Eä. Formel $\forall u \forall y \forall z \forall x I_{[v/a][w/b]}$:
 $\forall u \forall y \forall z \forall x (\neg P(x,x) \wedge ((P(u,y) \wedge P(y,z)) \Rightarrow P(u,z)) \wedge \neg(P(a, b) \Rightarrow \neg P(b, a)))$

PL-Resolution: Aufgabe (5)

<5) Matrix in KNF bringen [Äq]>

- $$\forall u \forall y \forall z \forall x \left(\neg P(x,x) \wedge ((P(u,y) \wedge P(y,z)) \Rightarrow P(u,z)) \wedge \neg(P(a, b) \Rightarrow \neg P(b, a)) \right)$$
- $$\equiv \forall u \forall y \forall z \forall x \left(\neg P(x,x) \wedge (\neg(P(u,y) \wedge P(y,z)) \vee P(u,z)) \wedge \neg(\neg P(a, b) \vee \neg P(b, a)) \right)$$
- $$\equiv \forall u \forall y \forall z \forall x \left(\neg P(x,x) \wedge (\neg P(u,y) \vee \neg P(y,z) \vee P(u,z)) \wedge P(a, b) \wedge P(b, a) \right)$$
- Klauselmengendarstellung:

$$\{ \{ \neg P(x,x) \}, \{ \neg P(u,y), \neg P(y,z), P(u,z) \}, \{ P(a, b) \}, \{ P(b, a) \} \}$$

Unifikation

- Stand bis jetzt: PL-Formel in BPF, die vorne nur Allquantoren aufweist.
- Resolution testet Unerfüllbarkeit und Allquantoren werden zu „falsch“ ausgewertet, wenn ihr Skopus für wenigstens für ein Element zu „falsch“ wird.
- Idee:
 - Gezielte Suche nach einem solchen Element.
 - Element verwenden, um damit die leere Klausel im letzten Schritt abzuleiten.

Allgemeinster Unifikator

- Die Unifikation kann in mehreren Schritten (mit mehreren Substitutionen) erfolgen:
 - $y \rightarrow f(a)$
 - $y \rightarrow f(x) \rightarrow f(a)$
- Die zweite Unifikation ist „allgemeiner“ als die erste.
- Allgemeinheit: Wenn eine Substitution in 2 Schritte aufgeteilt werden kann, ist der erste Teilschritt *allgemeiner* als die gesamte Subst.
- Gesucht: der/die **allgemeinste(n) Unifikator(en)**

Unifikationsalgorithmus

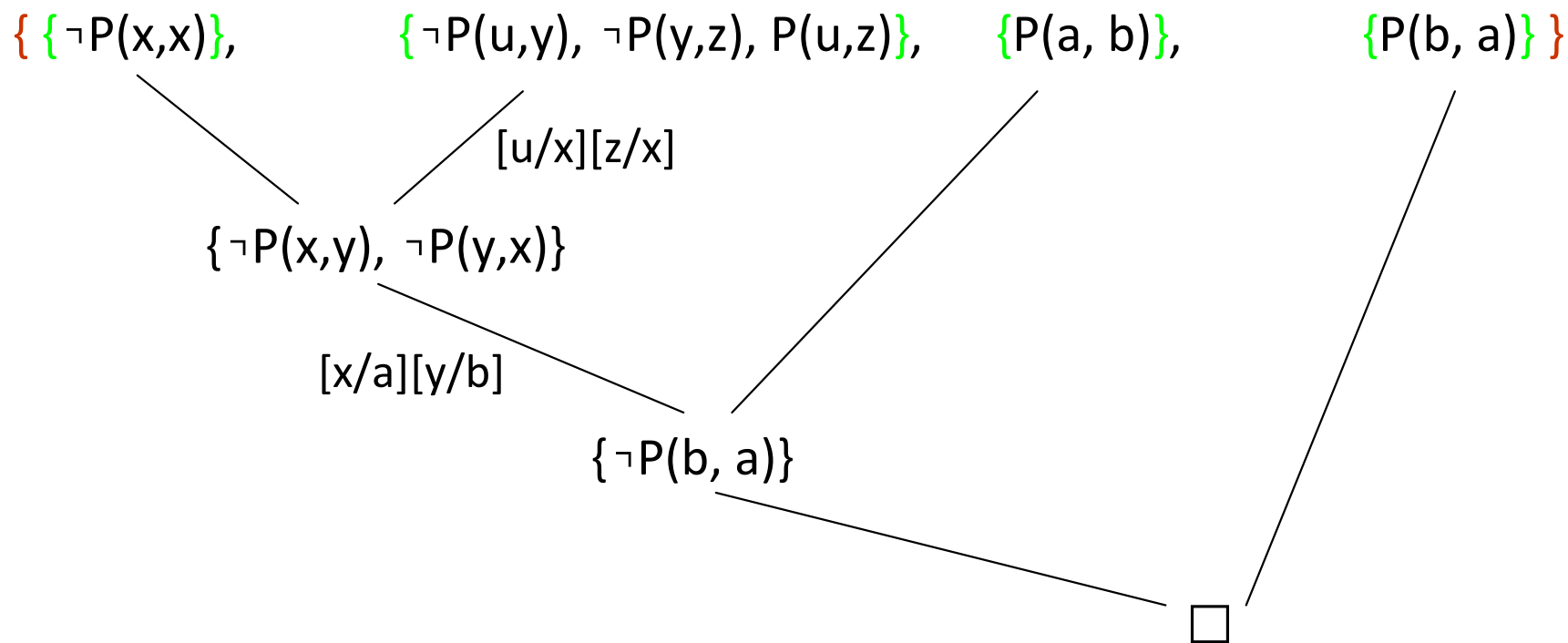
- Ablauf: <Loop> Laufe in allen Klauseln (von links nach rechts) zur ersten Position, in der sich zwei Zeichen unterscheiden.
 - Falls der Unterschied durch einen Junktor oder Prädikat verursacht wird: term.: „nicht unifizerbar“
 - Falls keiner der unt. Terme eine Variable x ist, term.: „nicht unifizierbar“, sonst sei der andere Term t .
 - Falls x in t vorkommt: term.: „nicht unifizierbar“
 - Nimm $[x/t]$ in die Subst. σ auf.
 - Wenn nicht alle Pointer am Ende sind: goto <Loop>
 - Gib σ als **allgemeinsten Unifikator** aus.

Resolution

- Unifikation erfolgt während der Resolution.
- Resolution im Kern wie in der Aussagenlogik.
- Wichtig:
 - Keine Konstanten ersetzen
 - Keine Funktionssymbole ersetzen
 - Stelligkeiten (von Funktionen und Prädikaten) können nicht geändert werden.
 - Prädikatensymbole müssen passen.

PL-Resolution: Aufgabe (6)

<6) Die Resolution selbst>



Resumee:

$\square \in \mathbf{Res}^* \Rightarrow$

$\{\mathbf{F}, \mathbf{G}\} \cup \{\neg \mathbf{H}\} \models \Rightarrow$

$\{\mathbf{F}, \mathbf{G}\} \models \mathbf{H}$

Das wars zu Logik!

Fragen?

Ansonsten auf zum Automatenteil!



PAUSE
(30 Minuten)

Nachtrag: Formeln schließen

- Gegeben: Die Formel $(Q(y) \Rightarrow \forall x P(x))$
- Gesucht: Geschlossene Form
- 1) $(\exists y Q(y) \Rightarrow \forall x P(x)) \equiv \forall y (Q(y) \Rightarrow \forall x P(x))$
oder
- 2) $\exists y (Q(y) \Rightarrow \forall x P(x))$
- Version 1 macht nur die Teilformel $Q(y)$ eä.
- Version 2 hingegen die ganze Formel, wie es beabsichtigt war.
- 1) und 2) sind nicht äq., da das Schließen der Formel so oder so nicht die Äq. erhält!

Ablauf – Tag 2

- Wdh.: Prädikatenlogik & Strukturen
- Prädikatenlogische Resolution
 - Formel schließen und bereinigte Pränexform (BPF)
 - Skolemisierung, Unifikation und Resolution
- **Warum Automaten?**
- Die Chomsky-Hierarchie
- Reguläre Mengen, DFA's und NFA's
- <Pause>
- Der Potenzautomat: Vom NFA zum DFA
- ϵ -FA & Überführung zum NFA (ϵ -frei machen)
- Kleene-Verfahren

Warum Automaten?

- Gegeben: Eine Obermenge, z.B. die Menge aller 0 und 1 Folgen (binäre Codierung)
- Gesucht: Teilmenge, die bestimmte Eigenschaften erfüllt, bzw. Identifikation der Elemente der Teilmenge in der Obermenge
- Je nach Teilmenge sehr leicht bis sehr schwer bis hin zu unmöglich
- Idee: Abstrakte Automaten mit unterschiedlicher Mächtigkeit

Ablauf – Tag 2

- Wdh.: Prädikatenlogik & Strukturen
- Prädikatenlogische Resolution
 - Formel schließen und bereinigte Pränexform (BPF)
 - Skolemisierung, Unifikation und Resolution
- Warum Automaten?
- **Die Chomsky-Hierarchie**
- Reguläre Mengen, DFA's und NFA's
- <Pause>
- Der Potenzautomat: Vom NFA zum DFA
- ϵ -FA & Überführung zum NFA (ϵ -frei machen)
- Kleene-Verfahren

Die Chomsky-Hierarchie

Überabzählbare Mengen

$\neq$

Abzählbare Mengen

$\supseteq$

Typ 0

Typ 0 -
Grammatik

DTM NTM

Aufzählbare Mengen

$\supseteq$

DTM NTM

Entscheidbare Mengen

$\supseteq$

Typ 1

Kontextsensitive
Grammatik

LBA

Kontextsensitive Mengen

– uvwxy-Theorem

$\supseteq$

PDA

Kontextfreie Mengen

Typ 2

Kontextfreie
Grammatik

DPDA

Deterministisch
Kontextfreie Mengen

– uvw-Theorem

$\supseteq$

Typ 3

Rechts- bzw.
linkslinere
Grammatik

DFA NFA ϵ -FA GFA $\leftarrow$ Reguläre Mengen

Potenzautomat

ϵ -frei machen

Kleene

trivial

Spezialfälle

Ablauf – Tag 2

- Wdh.: Prädikatenlogik & Strukturen
- Prädikatenlogische Resolution
 - Formel schließen und bereinigte Pränexform (BPF)
 - Skolemisierung, Unifikation und Resolution
- Warum Automaten?
- Die Chomsky-Hierarchie
- Reguläre Mengen, DFA's und NFA's
- <Pause>
- Der Potenzautomat: Vom NFA zum DFA
- ϵ -FA & Überführung zum NFA (ϵ -frei machen)
- Kleene-Verfahren

Reguläre Mengen

- Klasse an Mengen
- Lassen sich durch bestimmte Strukturen beschreiben (reguläre Ausdrücke)
- Praxisanwendung: Z.B. Parsing
- Typ 3 Mengen der Chomsky-Hierarchie

Reguläre Ausdrücke

- Beschreiben reguläre Mengen
- 2 Schreibweisen:
 - Reguläre Ausdruck – Schreibweise $(a + bc)^*$
 - (Mengen – Schreibweise) $(\{a\} \cup \{b\} \cdot \{c\})^*$
- Bestehen aus den folgenden Komponenten:
 - Konkatenation: $(a \cdot b)$ beschreibt die Menge $\{ab\}$
 - Auswahl: $(a+b)$ beschreibt die Menge $\{a, b\}$
 - Sternbildung: a^* beschr. die Menge $\{\epsilon, a, aa, aaa, \dots\}$
 - Plusbildung: a^+ beschreibt die Menge $\{a, aa, aaa, \dots\}$

Ablauf – Tag 2

- Wdh.: Prädikatenlogik & Strukturen
- Prädikatenlogische Resolution
 - Formel schließen und bereinigte Pränexform (BPF)
 - Skolemisierung, Unifikation und Resolution
- Warum Automaten?
- Die Chomsky-Hierarchie
- Reguläre Mengen, DFA's und NFA's
- <Pause>
- Der Potenzautomat: Vom NFA zum DFA
- ϵ -FA & Überführung zum NFA (ϵ -frei machen)
- Kleene-Verfahren

Die Chomsky-Hierarchie

Überabzählbare Mengen

$\neq$

Abzählbare Mengen

$\supseteq$

Typ 0

Typ 0 -
Grammatik

DTM NTM

Aufzählbare Mengen

$\supseteq$

DTM NTM

Entscheidbare Mengen

$\supseteq$

Typ 1

Kontextsensitive
Grammatik

LBA

Kontextsensitive Mengen

– uvwxy-Theorem

$\supseteq$

PDA

Kontextfreie Mengen

Typ 2

Kontextfreie
Grammatik

DPDA

Deterministisch
Kontextfreie Mengen

– uvw-Theorem

$\supseteq$

Typ 3

Rechts- bzw.
linkslinere
Grammatik

Potenzautomat

DFA NFA

ϵ -FA

GFA

ϵ -frei machen

Kleene

trivial

Reguläre Mengen

Spezialfälle

DFA/NFA: (Nicht-)deterministische endliche Automaten

- Formal Quintupel (5-Tupel): $A = (Q, \Sigma, \delta, q_0/S_0, F)$
- Q ist eine endliche Menge an Zuständen.
- Σ ein endliches Eingabealphabet
- F ist eine Endzustandsmenge, es gilt $F \subseteq Q$.
- q_0 ist ein Startzustand (DFA) bzw. S_0 eine Startzustandsmenge (NFA).
- δ ist eine Überföhrungsfunktion:
 - DFA: $\delta : Q \times \Sigma \rightarrow Q$
 - NFA: $\delta : Q \times \Sigma \rightarrow 2^Q$

Akzeptierte Sprache von NFAs/DFAs

- <Tafel>
- Initial zusammenhängende DFAs (izDFAs)
 - Jeder Zustand ist vom Startzustand aus erreichbar
- Vollständige DFAs (vDFAs)
 - Aus jedem Zustand ist für jedes Eingabesymbol ein Folgezustand definiert.

Akzeptierte Sprache DFA: Beispiel

- $A = (Q, \Sigma, \delta, q_0, F)$

- $Q = \{q_0, q_1\}$

- $\Sigma = \{a, b\}$

- $F = \{q_1\}$

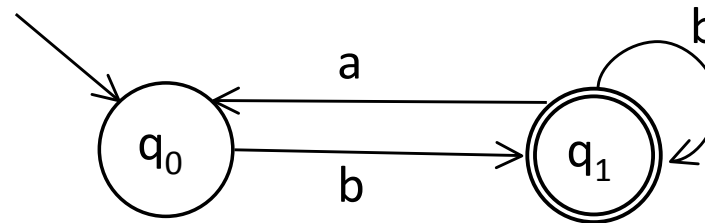
- δ :

- $\delta(q_0, b) = q_1$

- $\delta(q_1, b) = q_1$

- $\delta(q_1, a) = q_0$

- $L(A) = b (b^* + ba)^* = b (b + ba)^*$



Ablauf – Tag 2

- Wdh.: Prädikatenlogik & Strukturen
- Prädikatenlogische Resolution
 - Formel schließen und bereinigte Pränexform (BPF)
 - Skolemisierung, Unifikation und Resolution
- Warum Automaten?
- Die Chomsky-Hierarchie
- Reguläre Mengen, DFA's und NFA's
- <Pause>
- **Der Potenzautomat: Vom NFA zum DFA**
- ϵ -FA & Überführung zum NFA (ϵ -frei machen)
- Kleene-Verfahren

Die Chomsky-Hierarchie

Überabzählbare Mengen

$\neq$

Abzählbare Mengen

$\supseteq$

Typ 0

Typ 0 -
Grammatik

DTM NTM

Aufzählbare Mengen

$\supseteq$

DTM NTM

Entscheidbare Mengen

$\supseteq$

Typ 1

Kontextsensitive
Grammatik

LBA

Kontextsensitive Mengen

– uvwxy-Theorem

$\supseteq$

PDA

Kontextfreie Mengen

Typ 2

Kontextfreie
Grammatik

DPDA

Deterministisch
Kontextfreie Mengen

– uvw-Theorem

$\supseteq$

Typ 3

Rechts- bzw.
linkslinere
Grammatik

Potenzautomat

DFA

NFA

ϵ -FA

GFA

trivial

Reguläre Mengen

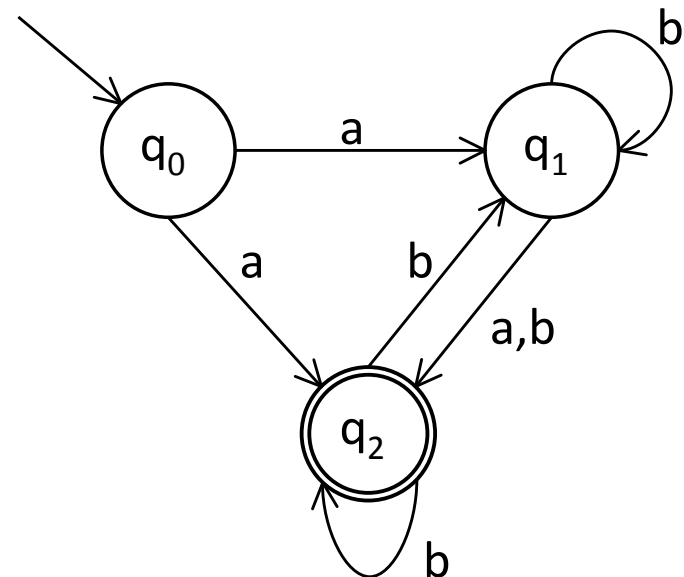
Spezialfälle

ϵ -frei machen

Kleene

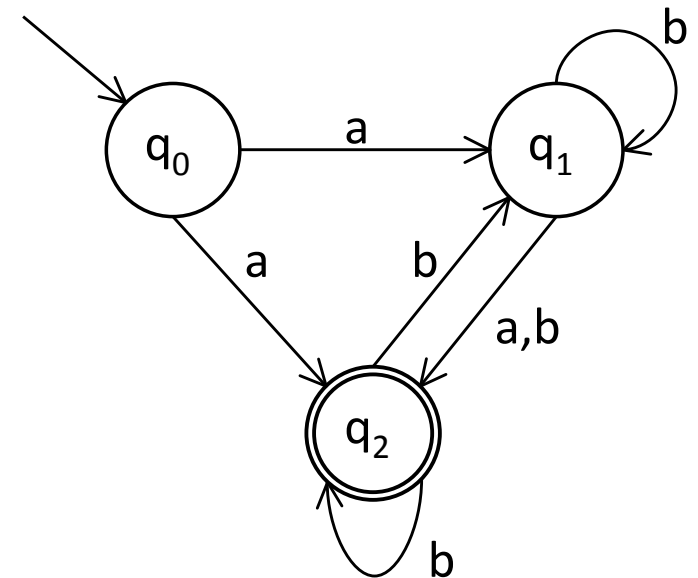
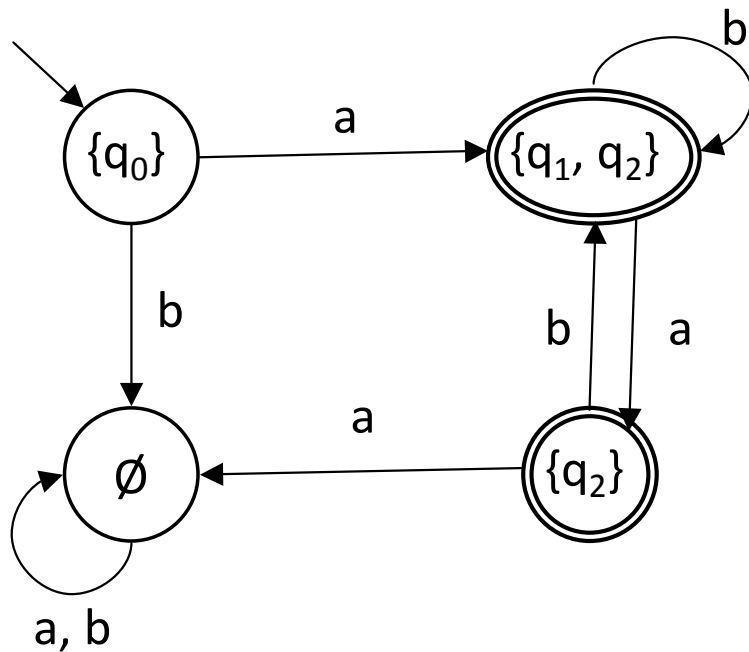
Potenzautomat

- NFA's lassen sich in (exponentiell größere) DFA's überführen, ohne dabei die akzeptierte Sprache zu ändern.
- Aufgabe: Wandeln Sie den folgenden NFA B in einen DFA um:
- Welche Sprache akzeptiert der NFA B?



Potenzautomat: Lösung

- $L(B) = a + ab^*(a+b) (b^*bb^*(a+b))^*$



Ablauf – Tag 2

- Wdh.: Prädikatenlogik & Strukturen
- Prädikatenlogische Resolution
 - Formel schließen und bereinigte Pränexform (BPF)
 - Skolemisierung, Unifikation und Resolution
- Warum Automaten?
- Die Chomsky-Hierarchie
- Reguläre Mengen, DFA's und NFA's
- <Pause>
- Der Potenzautomat: Vom NFA zum DFA
- ϵ -FA & Überführung zum NFA (ϵ -frei machen)
- Kleene-Verfahren

Die Chomsky-Hierarchie

Überabzählbare Mengen

$\neq$

Abzählbare Mengen

$\supseteq$

Typ 0

Typ 0 -
Grammatik

DTM NTM

Aufzählbare Mengen

$\supseteq$

DTM NTM

Entscheidbare Mengen

$\supseteq$

Typ 1

Kontextsensitive
Grammatik

LBA

Kontextsensitive Mengen

– uvwxy-Theorem

$\supseteq$

PDA

Kontextfreie Mengen

Typ 2

Kontextfreie
Grammatik

DPDA

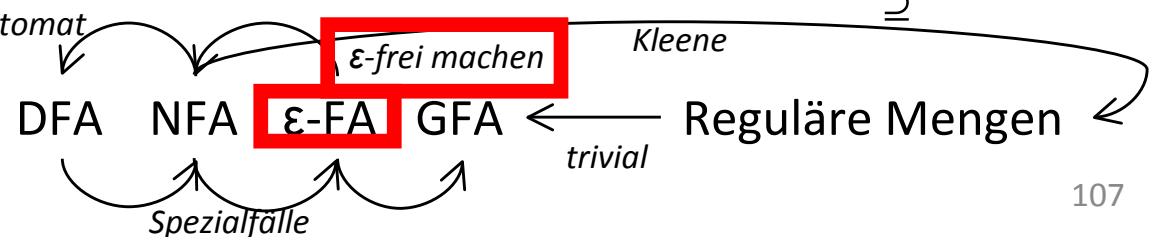
Deterministisch
Kontextfreie Mengen

– uvw-Theorem

$\supseteq$

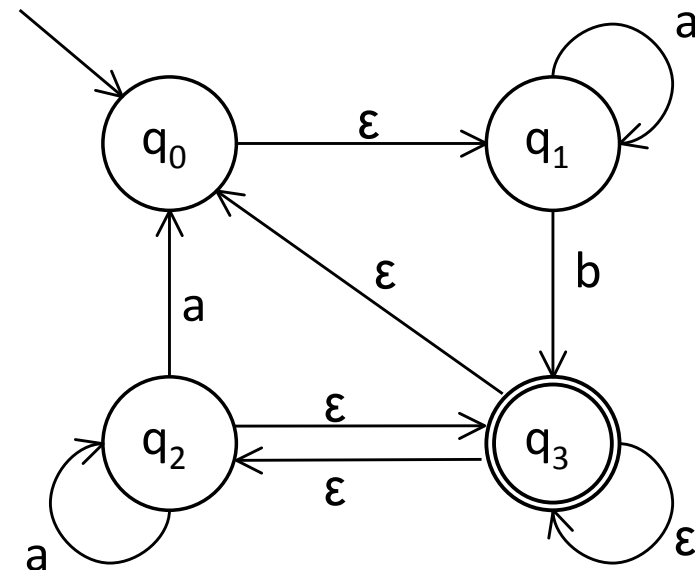
Typ 3

Rechts- bzw.
linkslin
Grammatik



ϵ -FA & ϵ -frei machen

- ϵ -FA: Endlicher Automat, der das leere Wort an Kanten erlaubt.
- Kann zu gleich großem NFA umgebaut werden.
- Aufgabe: Wandeln Sie den folgenden ϵ -FA in einen NFA um:



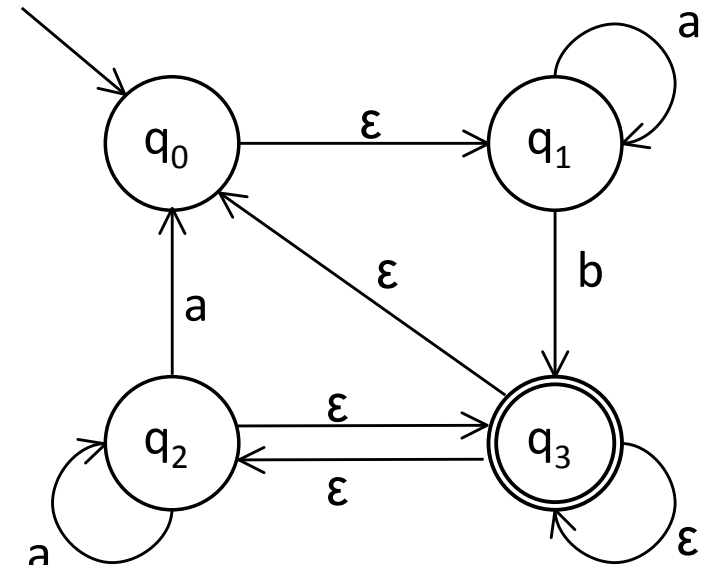
ϵ -frei machen: Lösung

R_0	q_0	q_1	q_2	q_3
q_0	1			
q_1		1		
q_2			1	
q_3				1

R_1	q_0	q_1	q_2	q_3
q_0		1		
q_1				
q_2				1
q_3	1		1	1

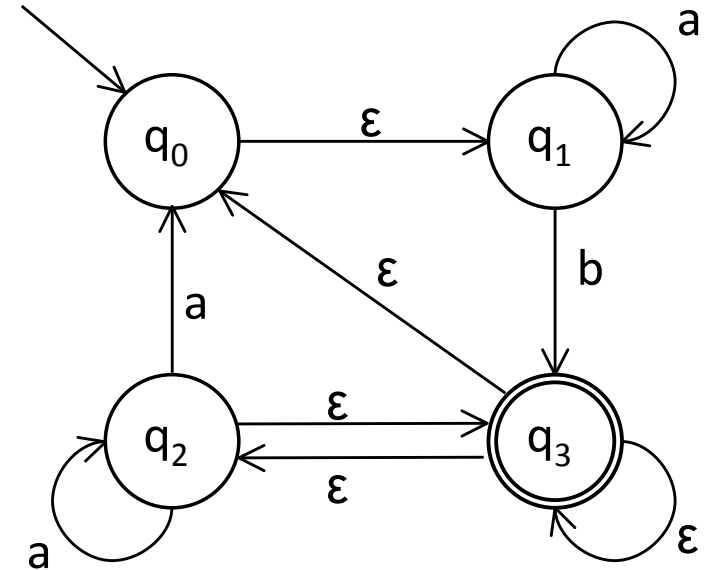
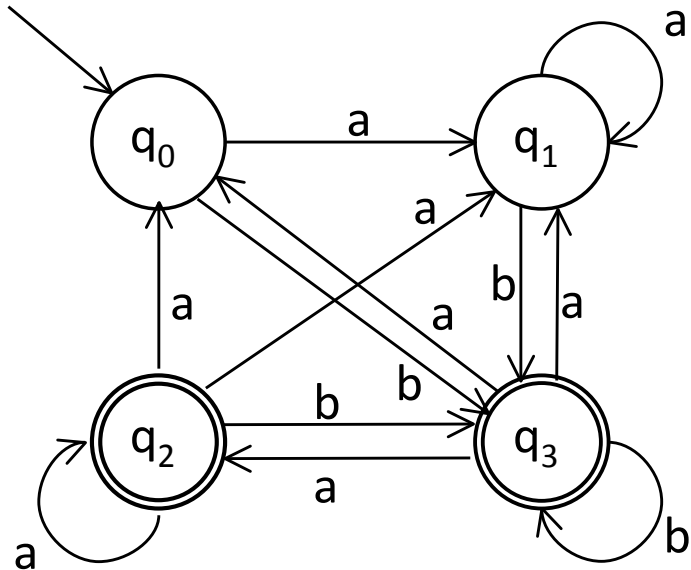
R_2	q_0	q_1	q_2	q_3
q_0				
q_1				
q_2	1		1	1
q_3	1	1	1	1

R_3	q_0	q_1	q_2	q_3
q_0				
q_1				
q_2	1	1	1	1
q_3	1	1	1	1



R	q_0	q_1	q_2	q_3
q_0	1	1		
q_1		1		
q_2	1	1	1	1
q_3	1	1	1	1

ϵ -frei machen: Lösung



R	q ₀	q ₁	q ₂	q ₃
q ₀	1	1		
q ₁		1		
q ₂	1	1	1	1
q ₃	1	1	1	1

Ablauf – Tag 2

- Wdh.: Prädikatenlogik & Strukturen
- Prädikatenlogische Resolution
 - Formel schließen und bereinigte Pränexform (BPF)
 - Skolemisierung, Unifikation und Resolution
- Warum Automaten?
- Die Chomsky-Hierarchie
- Reguläre Mengen, DFA's und NFA's
- <Pause>
- Der Potenzautomat: Vom NFA zum DFA
- ε -FA & Überführung zum NFA (ε -frei machen)
- Kleene-Verfahren

Die Chomsky-Hierarchie

Überabzählbare Mengen

$\neq$

Abzählbare Mengen

$\supseteq$

Typ 0

Typ 0 -
Grammatik

DTM NTM

Aufzählbare Mengen

$\supseteq$

DTM NTM

Entscheidbare Mengen

$\supseteq$

Typ 1

Kontextsensitive
Grammatik

LBA

Kontextsensitive Mengen

– uvwxy-Theorem

$\supseteq$

PDA

Kontextfreie Mengen

Typ 2

Kontextfreie
Grammatik

DPDA

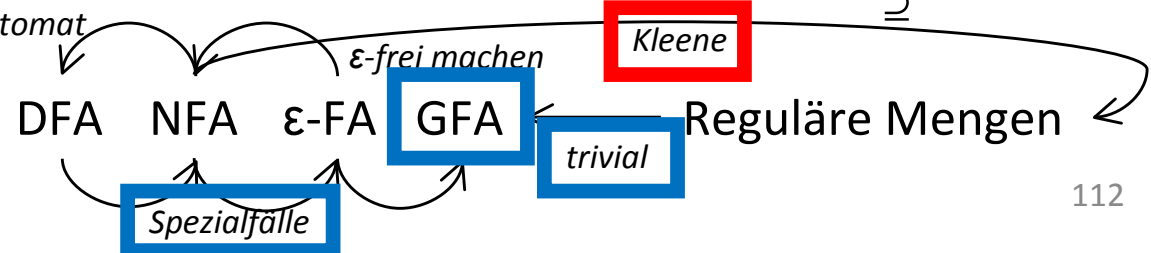
Deterministisch
Kontextfreie Mengen

– uvw-Theorem

$\supseteq$

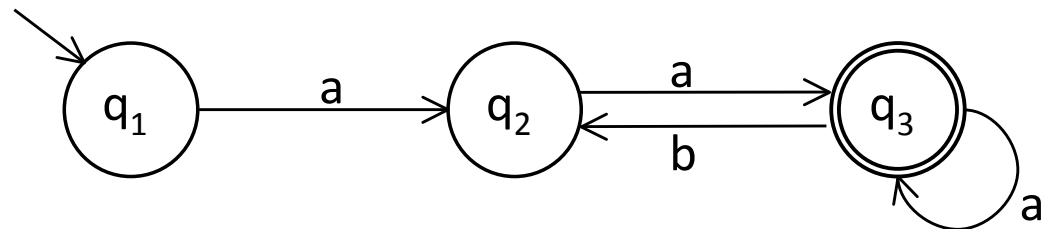
Typ 3

Rechts- bzw.
linkslinere
Grammatik

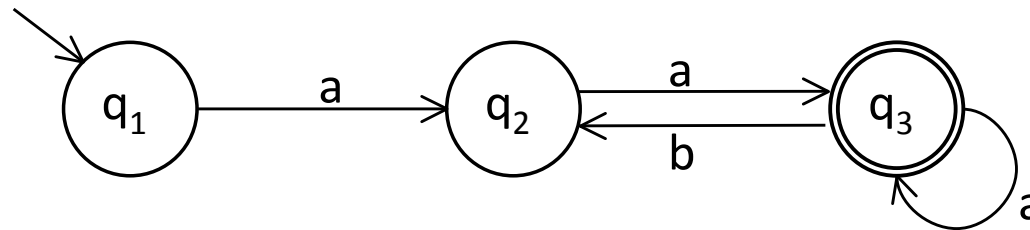


Kleene-Verfahren

- Rekursives Verfahren zum Ableiten des regulären Ausdrucks aus einem Automaten
- Exponentiell aufwändig!
- Aufgabe: Leiten Sie den zugehörigen regulären Ausdruck aus diesem Automaten ab:



Kleene-Verfahren: Anwendung



- Wir leiten aus den Übergängen ab:

$$R^0_{1,1} = \{\epsilon\}$$

$$R^0_{2,1} = \emptyset$$

$$R^0_{3,1} = \emptyset$$

$$R^0_{1,2} = \{a\}$$

$$R^0_{2,2} = \{\epsilon\}$$

$$R^0_{3,2} = \{b\}$$

$$R^0_{1,3} = \emptyset$$

$$R^0_{2,3} = \{a\}$$

$$R^0_{3,3} = \{a, \epsilon\}$$

Kleene-Verfahren: Lösung

$$\begin{aligned} R^1_{1,3} &= R^0_{1,3} \cup R^0_{1,1} (R^0_{1,1})^* R^0_{1,3} \\ &= (R^0_{1,1})^* R^0_{1,3} = \{\varepsilon\}^* \emptyset = \emptyset \end{aligned}$$

$$\begin{aligned} R^1_{1,2} &= R^0_{1,2} \cup R^0_{1,1} (R^0_{1,1})^* R^0_{1,2} \\ &= (R^0_{1,1})^* R^0_{1,2} = \{\varepsilon\}^* \{a\} = \{a\} \end{aligned}$$

$$R^1_{2,2} = R^0_{2,2} \cup R^0_{2,1} \dots = \{\varepsilon\} \cup \emptyset = \{\varepsilon\}$$

$$R^1_{2,3} = R^0_{2,3} \cup R^0_{2,1} \dots = \{a\} \cup \emptyset = \{a\}$$

$$R^1_{3,3} = R^0_{3,3} \cup R^0_{3,1} \dots = \{a, \varepsilon\} \cup \emptyset = \{a, \varepsilon\}$$

$$R^1_{3,2} = R^0_{3,2} \cup R^0_{3,1} \dots = \{b\} \cup \emptyset = \{b\}$$

Kleene-Verfahren: Lösung

$$\begin{aligned} R^3_{1,3} &= R^2_{1,3} \cup R^2_{1,3} (R^2_{3,3})^* R^2_{3,3} \\ &= R^2_{1,3} \cup R^2_{1,3} (R^2_{3,3})^+ \\ &= R^2_{1,3} (\{\epsilon\} \cup (R^2_{3,3})^+) = R^2_{1,3} (R^2_{3,3})^* \\ &= (R^1_{1,3} \cup R^1_{1,2} (R^1_{2,2})^* R^1_{2,3}) (R^2_{3,3})^* \\ &= (\emptyset \cup \{a\} \{\epsilon\}^* \{a\}) (R^2_{3,3})^* \\ &= \{a\} \{a\} (R^1_{3,3} \cup R^1_{3,2} (R^1_{2,2})^* R^1_{2,3})^* \\ &= \{a\} \{a\} (\{a, \epsilon\} \cup \{b\} \{\epsilon\}^* \{a\})^* \\ &= \{a\} \{a\} (\{a\} \cup \{b\} \{a\})^* \simeq \mathbf{aa (a + ba)^*} \end{aligned}$$

Das wars für heute!

Bis morgen zum Finale! 😊