LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Formale Sprachen und Komplexität
Sommersemester 2019

Reguläre Sprachen: Regularität widerlegen, Satz von Myhill-Nerode, Eigenschaften regulärer Sprachen

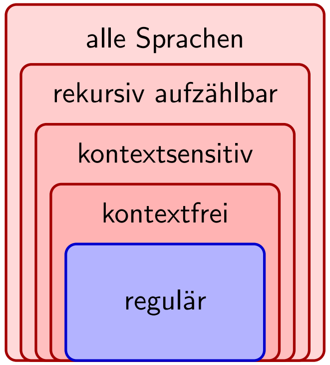
Prof. Dr. David Sabel

LFE Theoretische Informatik

Letzte Änderung der Folien: 4. Juni 2019

Inhaltsübersicht

Motivation zum Pumping-Lemma



Formalismen zur Darstellung von regulären Sprachen:

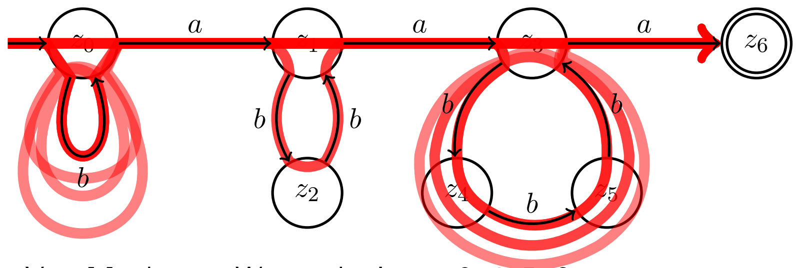
- Endliche Automaten
- Reguläre Ausdrücke
- Reguläre Grammatiken

Wie zeigt man, dass eine formale Sprache **nicht regulär** ist?

$\Rightarrow$ Das Pumping-Lemma ist ein Werkzeug dafür!

Idee des Pumping-Lemmas: Beispiel

Beispiel: DFA M:

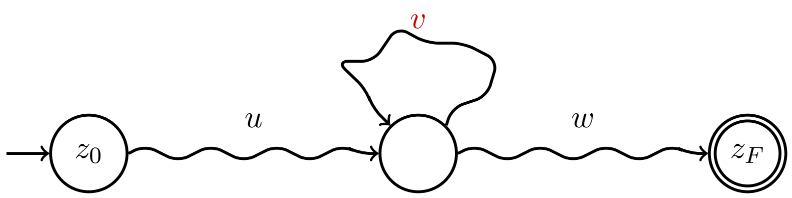


- Von M erkannte Wörter der Länge 3, 4, 5, 6, ...
- Beobachtung 1: Jedes Wort z mit Länge > 3, das M erkennt, **muss** mindestens eine **Schleife durchlaufen**.
- Beobachtung 2: Wenn wir die **Schleife mehrfach durchlaufen**, wird das entsprechende Wort immer noch erkannt, d.h.

Wörter in $L(M)$ mit Länge > 3 können wir **aufpumpen** und verbleiben in der Sprache $L(M)$

Idee des Pumping-Lemmas: Allgemeiner

Gilt das allgemein?



- Wenn ein endlicher Automat n **Zustände** hat, dann müssen akzeptierte Wörter der **Länge** $\geq n$ eine Schleife durchlaufen
- Diese Wörter kann man aufpumpen: uvw, $uvvw$,$uvvvw$, ...
  Allgemein: uv^iw für $i = 0, 1, 2, \ldots$ liegen in der erkannten Sprache

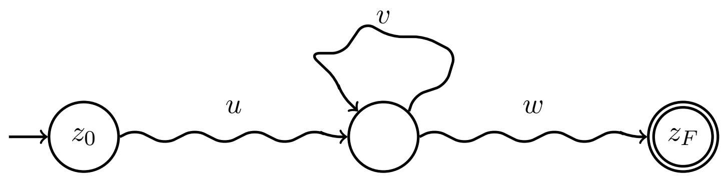
Das Pumping-Lemma

Lemma 4.9.1 (Pumping-Lemma)

Jede reguläre Sprache L hat die folgende Pumping-Eigenschaft:
Es gibt eine Zahl $n \in \mathbb{N}_{>0}$, sodass jedes Wort $z \in L$, welches Mindestlänge n hat (d.h. $|z| \geq n$), als $z = uvw$ geschrieben werden kann, so dass gilt:

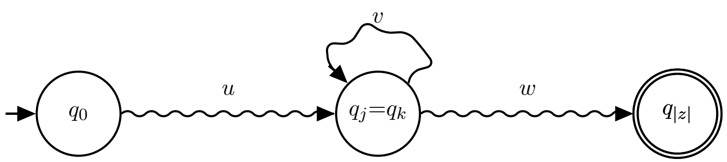
- $|uv| \leq n$
- $|v| \geq 1$
- für alle $i \geq 0$: $uv^iw \in L$.

Die Zahl n nennt man auch die Pumping-Konstante der Sprache L



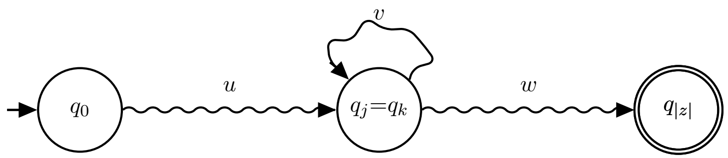
Beweis des Pumping-Lemmas (1)

- Sei $M = (Z, \Sigma, \delta, z_0, E)$ ein DFA, der L akzeptiert mit $|Z| = n$.
- Jeder Lauf für ein $z \in L$ besucht $|z| + 1$ Zustände. Sei $|z| \geq n$.
  Sei $q_0, q_1, \ldots q_{|z|}$ die besuchte Folge mit $q_0 = z_0$ und $q_{|z|} \in E$.
- Da $|Z| = n$, wird spätestens nach Lesen von n Zeichen ein Zustand erneut besucht
- Sei q_k (mit $k \leq n$) der erste Zustand, der bereits besucht wurde:
  D.h. es gibt $j < k$, sodass $q_k = q_j$ und k ist minimal, $z = uvw$ mit



Beweis des Pumping-Lemmas (2)

...
D.h. es gibt $j < k$, sodass $q_k = q_j$ und k ist minimal, $z = uvw$ mit



Wir zeigen nun die drei geforderten Eigenschaften der Zerlegung:

- Aus $j < k$ folgt $|v| \geq 1$.
- Aus $k \leq n$ folgt $|uv| \leq n$.
- Aus $q_j = q_k$ folgt $\widehat{\delta}(q_0, u) = q_j = \widehat{\delta}(q_0, uv) = q_k$ und somit $\widehat{\delta}(q_0, uw) = \widehat{\delta}(q_0, uvw) = q_{|z|} \in E$, d.h. $uv^0w \in L(M)$.
  Sei $i > 0$. Aus $\widehat{\delta}(q_j, v) = q_k = q_j$ folgt $\widehat{\delta}(q_j, v^i) = q_j$ und daher $\widehat{\delta}(uv^iw) = \widehat{\delta}(q_k, v^iw) = \widehat{\delta}(q_j, w) = q_{|z|} \in E$.
  Daher gilt $uv^iw \in L(M)$ für alle $i \in \mathbb{N}$. $\square$

Endliche Sprachen

Zur Erinnerung: Pumping-Eigenschaft

Es gibt eine Zahl $n \in \mathbb{N}_{>0}$, sodass jedes Wort $z \in L$, welches Mindestlänge n hat (d.h. $|z| \geq n$), als $z = uvw$ geschrieben werden kann, so dass gilt:

- $|uv| \leq n$
- $|v| \geq 1$
- für alle $i \geq 0$: $uv^iw \in L$.

Als prädikatenlogische Formel:

$$\begin{array}{l} \exists n \in \mathbb{N}_{>0}: \\ \quad \forall z \in L: \\ \quad\quad (|z| \geq n \Rightarrow \\ \quad\quad\quad \exists u, v, w:(z{=}uvw \wedge |uv| \leq n \wedge |v| \geq 1 \wedge \forall i \geq 0:(uv^iw \in L))) \end{array}$$

Warum erfüllen **endliche Sprachen** das Pumping-Lemma?

Wähle n größer als die Länge des längsten Worts!

Anwendung des Pumping-Lemmas

- Pumping-Lemma:
  Sprache regulär $\Longrightarrow$ Sprache erfüllt die Pumping-Eigenschaft
- Zeige, dass eine Sprache nicht regulär ist, durch Kontraposition:

Sprache erfüllt **nicht** die Pumping-Eigenschaft
$\Longrightarrow$ Sprache ist **nicht** regulär

Umformung der negierten Pumping-Eigenschaft

$$\begin{array}{ll}
& \neg(\exists n \in \mathbb{N}_{>0}{:}\forall z \in L{:}(|z| \geq n \Rightarrow \exists u,v,w{:}(z{=}uvw \wedge |uv| \leq n \wedge |v| \geq 1 \wedge \forall i \geq 0{:}(uv^iw \in L)))) \\
\longleftrightarrow & \forall n \in \mathbb{N}_{>0}{:}\neg(\forall z \in L{:}(|z| \geq n \Rightarrow \exists u,v,w{:}(z{=}uvw \wedge |uv| \leq n \wedge |v| \geq 1 \wedge \forall i \geq 0{:}(uv^iw \in L)))) \\
\longleftrightarrow & \forall n \in \mathbb{N}_{>0}{:}(\exists z \in L{:}(\neg(|z| \geq n \Rightarrow \exists u,v,w{:}(z{=}uvw \wedge |uv| \leq n \wedge |v| \geq 1 \wedge \forall i \geq 0{:}(uv^iw \in L))))) \\
\longleftrightarrow & \forall n \in \mathbb{N}_{>0}{:}(\exists z \in L{:}(\neg(\neg(|z| \geq n) \vee (\exists u,v,w{:}(z{=}uvw \wedge |uv| \leq n \wedge |v| \geq 1 \wedge \forall i \geq 0{:}(uv^iw \in L)))))) \\
\longleftrightarrow & \forall n \in \mathbb{N}_{>0}{:}(\exists z \in L{:}((|z| \geq n) \wedge \neg(\exists u,v,w{:}(z{=}uvw \wedge |uv| \leq n \wedge |v| \geq 1 \wedge \forall i \geq 0{:}(uv^iw \in L))))) \\
\longleftrightarrow & \forall n \in \mathbb{N}_{>0}{:}(\exists z \in L{:}((|z| \geq n) \wedge (\forall u,v,w{:}(\neg(z{=}uvw \wedge |uv| \leq n \wedge |v| \geq 1 \wedge \forall i \geq 0{:}(uv^iw \in L)))))) \\
\longleftrightarrow & \forall n \in \mathbb{N}_{>0}{:}(\exists z \in L{:}((|z| \geq n) \wedge (\forall u,v,w{:}(\neg(z{=}uvw \wedge |uv| \leq n \wedge |v| \geq 1) \vee \neg(\forall i \geq 0{:}uv^iw \in L))))) \\
\longleftrightarrow & \forall n \in \mathbb{N}_{>0}{:}(\exists z \in L{:}((|z| \geq n) \wedge (\forall u,v,w{:}((z{=}uvw \wedge |uv| \leq n \wedge |v| \geq 1) \Rightarrow \neg(\forall i \geq 0{:}uv^iw \in L))))) \\
\longleftrightarrow & \forall n \in \mathbb{N}_{>0}{:}(\exists z \in L{:}((|z| \geq n) \wedge (\forall u,v,w{:}((z{=}uvw \wedge |uv| \leq n \wedge |v| \geq 1) \Rightarrow \exists i \geq 0{:}uv^iw \notin L))))
\end{array}$$

Formale Sprache L erfüllt nicht die Pumping-Eigenschaft:

Für jede Zahl $n \in \mathbb{N}_{>0}$ gibt es ein Wort $z \in L$ mit $|z| \geq n$, sodass für jede Zerlegung $z = uvw$ mit

- $|uv| \leq n$ und
- $|v| \geq 1$

ein $i \geq 0$ existiert mit $uv^iw \notin L$.

Anwendung des Pumping-Lemmas

Satz

Die Sprache $L = \{a^jb^j \mid j \in \mathbb{N}\}$ ist nicht regulär.

Beweis: Wir zeigen, dass L die Pumping-Eigenschaft nicht erfüllt und schließen mit dem Pumping-Lemma, dass L nicht regulär ist:

- Sei $n \in \mathbb{N}_{>0}$ beliebig. Wir wählen $z \in L$:
  $z = a^nb^n$ (damit ist auch $|z| \geq n$ erfüllt).
- Sei $z = uvw$ eine beliebige Zerlegung von z, sodass $|uv| \leq n$ und $|v| \geq 1$.
- Dann ist $u = a^r, v = a^s$ mit $r + s \leq n$, $s > 0$ und $w = a^tb^n$ mit $r + s + t = n$.
  Daher können wir z.B. $i = 2$ wählen und erhalten
  $uv^iw = uv^2w = a^ra^sa^sa^tb^n = a^{n+s}b^n \notin L$,da $s > 0$. □

Beweise Nicht-Regularität als Spiel

Sei L die formale Sprache.

1. Der **Gegner** wählt die Zahl $n \in \mathbb{N}_{>0}$.
2. **Wir** wählen das Wort $z \in L$ mit $|z| \geq n$.
3. Der **Gegner** wählt Zerlegung $z = uvw$ mit $|uv| \leq n$ und $|v| \geq 1$.
4. **Wir** gewinnen das Spiel, wenn wir ein $i \geq 0$ angeben können, sodass $uv^iw \notin L$.

Wenn wir das Spiel **für alle Wahlmöglichkeiten des Gegners** gewinnen, dann haben wir die Nichtregularität von L nachgewiesen.

Beispiel

Satz

Die Sprache $L = \{a^p \mid p \text{ ist Primzahl}\}$ ist nicht regulär.

Wir zeigen, dass wir das eben eingeführte Spiel stets gewinnen:

1. Sei $n \in \mathbb{N}_{>0}$ vom Gegner gewählt.
2. Wir wählen $z \in L$ als $z = a^p$ mit p ist die nächste Primzahl, die größer gleich n ist.
3. Der Gegner wählt Zerlegung $u = a^r$, $v = a^s$, $w = a^t$ mit $uvw = a^p$, $|uv| \leq n$, $|v| \geq 1$ (und damit $s \geq 1$).
4. Wir wählen $i = p + 1$. Dann ist $uv^iw \notin L$, denn $uv^iw = a^r(a^s)^{p+1}a^t = a^{r+s\cdot(p+1)+t} = a^{r+s\cdot p+s+t} = a^{s\cdot p+p} = a^{p\cdot(s+1)}$ und für $s \geq 1$ folgt, dass $p \cdot (s+1)$ keine Primzahl sein kann. □

Beispiel

Satz

Die Sprache $L = \{a^n \mid n \text{ ist Quadratzahl}\}$ ist nicht regulär.

1. Sei $n \in \mathbb{N}_{>0}$ vom Gegner gewählt.
2. Wir wählen $z = a^{n^2} \in L$.
3. Sei $z = uvw$ vom Gegner zerlegt, sodass $|uv| \leq n$ und $|v| \geq 1$.
4. Wir wählen $i = 2$, d.h. wir betrachten $uv^2w = a^k$.
   - $1 + n^2 \leq k$ (denn $|v| \geq 1$)
   - $k \leq n^2 + n$ (denn $|uv| \leq n$ und daher $|v| \leq n$).

   Dann kann k jedoch keine Quadratzahl sein, denn $n^2 + n = (n+1) \cdot n < (n+1)^2$.
   Daher gilt $uv^2w \notin L$.
   Das Pumping-Lemma zeigt somit, dass L nicht regulär ist. □

Beispiel

Satz

Die Sprache $L = \{a^{2^n} \mid n \in \mathbb{N}\}$ ist nicht regulär.

Beweis:

- Sei $n \in \mathbb{N}_{>0}$ beliebig.
- Sei $z \in L$ mit $|z| \geq n$ das Wort $z = a^{2^n}$.
- Sei $z = uvw$ mit $|uv| \leq n$ und $|v| = k \geq 1$.
- Dann ist $1 \leq k \leq n$ und $uv^2w = a^{2^n+k}$ und $2^n + k \neq 2^l$ da $2^n + k < 2^{n+1} = 2^n + 2^n$ denn $k \leq n < 2^n$.
  Daher ist $uv^2w \notin L$.

Mit dem Pumping-Lemma folgt, dass L nicht regulär ist. □

Beispiel

Satz

Die Sprache $L = \{w \in \{a,b\}^* \mid w \text{ ist Palindrom}\}$ ist nicht regulär.

- Sei $n \in \mathbb{N}_{>0}$ beliebig.
- Wir wählen $z = a^n b a^n \in L$ als Wort mit Mindestlänge n.
- Sei $z = uvw$ mit $|uv| \leq n$ und $|v| \geq 1$.
- Dann ist $uv^0w = a^k b a^n$ mit $k = n - |v| < n$ kein Palindrom.

Mit dem Pumping-Lemma folgt, dass L nicht regulär ist. □

Pumping-Eigenschaft ist nicht hinreichend

Lemma

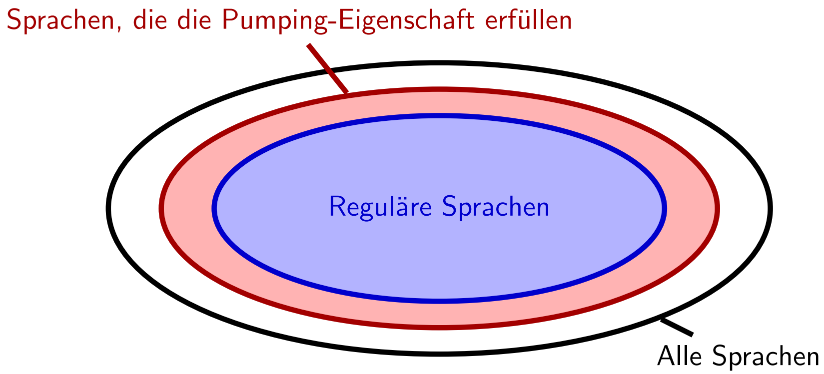
Es gibt Sprachen, die die Pumping-Eigenschaft erfüllen aber nicht regulär sind. Die Sprache $L = \{a^k b^l c^l \mid k, l \in \mathbb{N}\} \cup \{b,c\}^*$ ist eine solche Sprache.

Beweis, dass L die Pumping-Eigenschaft erfüllt:

- Sei $n \geq 1$ beliebig.
- Sei $z \in L$ mit $|z| \geq n$
- Wenn $z \in \{b,c\}^*$, zerlege $z = uvw$ mit $u = \varepsilon$,v das erste Symbol von z und w der $n-1$-Zeichen lange Suffix von z. Offensichtlich gilt $|v| \geq 1$, $|uv| \leq n$ und $uv^iw \in \{b,c\}^* \subseteq L$ für alle $i \in \mathbb{N}$.
- Wenn z von der Form $a^k b^l c^l$ ist und $z \notin \{b,c\}^*$, dann muss $k > 0$ gelten und wir zerlegen $z = uvw$ mit $u = \varepsilon$, $v = a$, $w = a^{k-1} b^l c^l$. Da $|v| = 1, |uv| \leq n$ und $uv^iw = a^{k+i-1} b^l c^l \in L$ für alle $i \in \mathbb{N}$, erfüllt L die Eigenschaften des Pumping-Lemmas.

Beweis, dass L nicht regulär ist folgt **später!**

Mengendiagramm



Wichtige Konsequenz

Das Pumping-Lemma kann <u>nicht</u> verwendet werden, um zu zeigen, dass eine Sprache regulär ist.

Zusammenfassung Pumping-Lemma

- Das Pumping-Lemma formuliert **eine notwendige Bedingung** für **reguläre** Sprachen:

  Sehr informell:
  Wörter einer regulären Sprache können aufgepumpt werden, wenn sie lang genug sind.

- Anwendung:

  L erfüllt die **Pumping-Eigenschaft nicht**
  $\implies$ L **nicht regulär**

- Das Pumping-Lemma gibt **keine hinreichende Bedingung** für reguläre Sprachen, d.h. Regularität kann **nicht** mit dem Pumping-Lemma gezeigt werden.
- Nicht-Regularität widerlegen funktioniert nicht in jedem Fall mit dem Pumping-Lemma!

Die Nerode-Relation

Definition (Nerode-Relation $\sim_L$)

Sei L eine formale Sprache über Σ. Die Nerode-Relation $\sim_L \subseteq \Sigma^* \times \Sigma^*$ zu L ist definiert für alle Worte $u, v \in \Sigma^*$ durch:

$$u \sim_L v \iff \forall w \in \Sigma^* : uw \in L \iff vw \in L$$

Informell: $u \sim_L v$, wenn sich ihr Enthaltensein in L gleich verhält bezüglich beliebiger Erweiterung um denselben Suffix.

Beispiel: $L = \{ab^n \mid n \in \mathbb{N}\}$

Gilt $a \sim_L ab$?	✓
Gilt $a \sim_L b$?	$\times$ (z.B. $w = \varepsilon$ oder $w = b^i$)
Gilt $\varepsilon \sim_L a$?	$\times$ (z.B. $w = b^i$ oder $w = ab^i$)
Gilt $aau \sim_L aav$ mit $u, v \in \{a,b\}^*$?	✓
Gilt $ab^i \sim_L ab^j$?	✓
Gilt $aau \sim_L bv$ mit $u, v \in \{a,b\}^*$?	✓

Die Nerode-Relation (2)

Satz

Die Nerode-Relation ist eine Äquivalenzrelation.

Beweis: einfach

Erinnerung: Index einer Äquivalenzrelation:

- Äquivalenzklasse $[u]_{\sim_L} = \{w \mid u \sim_L w\}$
- Der Index einer Äquivalenzrelation ist die Anzahl ihrer disjunkten Äquivalenzklassen $\Sigma^* = [u_1]_{\sim_L} \uplus [u_2]_{\sim_L} \ldots$
- Der Index kann auch unendlich sein.

Beispiel: Index

$L = \{ab^n \mid n \in \mathbb{N}\}$, $Index(\sim_L) =?$

- $[\varepsilon]_{\sim_L} = \{\varepsilon\}$
- $[a]_{\sim_L} = \{a, ab, abb, abbb, \ldots\}$
- $[aa]_{\sim_L} = \{aau \mid u \in \{a,b\}^*\} \cup \{bu \mid u \in \{a,b\}^*\} \cup \{abuav \mid u, v \in \{a,b\}^*\}$
- $\{a,b\}^* = [\varepsilon]_{\sim_L} \uplus [a]_{\sim_L} \uplus [aa]_{\sim_L}$
- Daher $Index(\sim_L) = 3$

Satz von Myhill und Nerode

Theorem (Satz von Myhill und Nerode)

Eine formale Sprache L ist genau dann **regulär**, wenn der **Index** von $\sim_L$ **endlich** ist.

Unabhängig bewiesen von John Myhill (1957) und Anil Nerode (1958)

Damit haben wir eine genaue(!) Charakterisierung der regulären Sprachen

Beweis des Satzes von Myhill und Nerode

Teil 1: Wenn L regulär ist, dann ist der Index von $\sim_L$ endlich.

- Sei L regulär und $M = (Z, \Sigma, \delta, z_0, E)$ ein DFA mit $L(M) = L$.
- Sei $\approx_M \subseteq (\Sigma^* \times \Sigma^*)$ definiert als: $u \approx_M v \iff \widehat{\delta}(z_0, u) = \widehat{\delta}(z_0, v)$
  „$u \approx_M v$, wenn u, v auf M zum gleichen Zustand führen."
- $\approx_M$ ist eine Äquivalenzrelation
- $\mathit{Index}(\approx_M) = |Z|$ (und damit endlich)
- Wir zeigen: $u \approx_M v \implies u \sim_L v$ (d.h. $\approx_M$ verfeinert $\sim_L$)
  Dann gilt auch $\mathit{Index}(\sim_L) \leq \mathit{Index}(\approx_M) = |Z|$ (und damit endlich).
- Sei $u \approx_M v$ und $w \in \Sigma^*$. Dann gilt:

$$\begin{aligned}\widehat{\delta}(q_0, uw) &= \widehat{\delta}(\widehat{\delta}(q_0, u), w) \\ &= \widehat{\delta}(\widehat{\delta}(q_0, v), w) = \widehat{\delta}(q_0, vw)\end{aligned}$$

und damit $uw \in L \iff vw \in L$. Da w beliebig, zeigt dies $u \sim_L v$.

Beweis des Satzes von Myhill und Nerode

Teil 2: Wenn der Index von $\sim_L$ endlich ist, dann ist L regulär.

- Sei der Index von $\sim_L$ endlich.
- $\Sigma^* = [u_1]_{\sim_L} \cup \ldots \cup [u_n]_{\sim_L}$ für $n \in \mathbb{N}_{>0}$
- Konstruiere **Nerode-Automaten**:
  DFA $M = (Z, \Sigma, \delta, [\varepsilon]_{\sim_L}, E)$ mit
  - $Z = \{[u_1]_{\sim_L}, \ldots, [u_n]_{\sim_L}\}$,
  - $\delta([u_i]_{\sim_L}, a) = [u_i a]_{\sim_L}$ für alle $a \in \Sigma$ und
  - $E = \{[u_i]_{\sim_L} \mid i \in \{1, \ldots, n\}, u_i \in L\}$.
- Es gilt: $w \in L(M)$
  g.d.w. $\widehat{\delta}([\varepsilon]_{\sim_L}, w) \in E$
  g.d.w. $[w]_{\sim_L} \subseteq L$
  g.d.w. $w \in L$

  Daher gilt: $L(M) = L$

Anwendung des Satzes von Myhill und Nerode

Da der Satz eine hinreichende und notwendige Bedingung für reguläre Sprachen liefert, könnte man ihn verwenden, um

1. Regularität nachzuweisen: Zeige $\mathit{Index}(\sim_L)$ ist endlich.
   „oft einfacher: gebe endlichen Automat an, der L akzeptiert"
2. Nichtregularität nachzuweisen: Zeige $\mathit{Index}(\sim_L)$ ist unendlich.
   „wird insbesondere dann benutzt, wenn das Pumping-Lemma nicht funktioniert"

Wie zeigt man $\mathit{Index}(\sim_L) = \infty$?

Rezept: Finde unendlich viele Äquivalenzklassen $[u_1]_{\sim_L}, [u_2]_{\sim_L}, \ldots$, die paarweise verschieden sind ($[u_i]_{\sim_L} \cap [u_j]_{\sim_L} = \emptyset$ für $i \neq j$):

- Finde für $i = 1, 2, \ldots$ Worte u_i und w_i, sodass alle u_i paarweise verschieden und alle w_i paarweise verschieden und:

$$u_i w_i \in L \text{ aber } u_j w_i \notin L \text{ für alle } i \neq j.$$

- Dann sind $[u_1]_{\sim_L}, [u_2]_{\sim_L}, [u_3]_{\sim_L}, \ldots$ paarweise disjunkt

Beachte: Es ist **nicht** notwendig, **alle** Äquivalenzklassen der disjunkten Zerlegung von Σ^* zu finden!

Beispiel

Betrachte $L = \{a^n b^n \mid n \in \mathbb{N}_{>0}\}$ und deren Nerode-Relation $\sim_L$.

- Für $u_i = a^i b$ für $i \in \mathbb{N}_{>0}$ gilt:

  $[u_i]_{\sim_L} =$ alle Worte, denen noch $i-1$ b's fehlen, um in L enthalten zu sein

$$\begin{aligned}
[u_1]_{\sim_L} &= \{ab, a^2b^2, \ldots\} = L\\
[u_2]_{\sim_L} &= \{a^2b, a^3b^2, a^4b^3, \ldots\}\\
[u_3]_{\sim_L} &= \{a^3b, a^4b^2, a^5, b^3, \ldots\}\\
&\ldots
\end{aligned}$$

- Sei $w_i = b^{i-1}$ für $i \in \mathbb{N}_{>0}$
- $u_i w_i = a^i b^i \in L$, aber $u_i w_j = a^i b^j \notin L$ für $i \neq j$
- Daher gilt: $[u_i]_{\sim_L} \cap [u_j]_{\sim_L} = \emptyset$ und $\mathit{Index}(\sim_L) = \infty$
- Mit dem Satz von Myhill und Nerode folgt: L ist nicht regulär.

Bemerkung: $a^i \notin [u_i]_{\sim_L}$, denn $a^i ab^{i+1} \in L$ aber $u_i ab^{i+1} \notin L$

Nicht-reguläre Sprache, die die Pumping-Eigenschaft hat

Satz

Die Sprache $L = \{a^k b^l c^l \mid k, l \in \mathbb{N}\} \cup \{b, c\}^*$ ist nicht regulär.

Beweis:

- Betrachte $[u_i]_{\sim_L} = [ab^i c]_{\sim_L}$ für $i \in \mathbb{N}_{>0}$.
- Mit $w_i = c^{i-1}$ gilt:
  $u_i w_i = ab^i c^i \in L$, aber $u_i w_j = ab^i c^j \notin L$ für $i \neq j$.
- Daher sind alle $[u_i]_{\sim_L}$ disjunkt und $\mathit{Index}(\sim_L) = \infty$.
- Mit dem Satz von Myhill und Nerode folgt:
  L ist nicht regulär. □

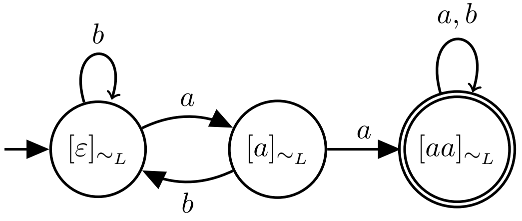
Beispiel: Reguläre Sprache

Sei L Sprache über $\{a, b\}$ mit $L = \{uaav \mid uv \in \{a, b\}^*\}$
Die disjunkten Äquivalenzklassen von $\sim_L$ sind:

- $[\varepsilon]_{\sim_L} =$ alle Worte ohne zwei aufeinanderfolgende a's, die nicht mit a enden
- $[a]_{\sim_L} =$ alle Worte ohne zwei aufeinanderfolgende a's, die mit a enden
- $[aa]_{\sim_L} = L$

Daher folgt $\mathit{Index}(\sim_L) = 3$ und L ist regulär.

Nerode-Automat dazu:



Minimalautomat

Satz

Der Nerode-Automat einer regulären Sprache L ist der sogenannte Minimalautomat, d.h. jeder DFA M', mit $L(M') = L$ hat mindestens so viele Zustände wie der Nerode-Automat. Zudem sind alle minimalen DFAs (bis auf Umbenennung von Zuständen) identisch.

Beweis (Teil 1): (Minimalität):

- Sei $M = (Z, \Sigma, \delta, z_0, E)$ der Nerode-Automat der Sprache L
- Sei $M' = (Z', \Sigma, \delta', z'_0, E')$ ein beliebiger DFA mit $L(M') = L$.
- Der Beweis des Satzes von Myhill und Nerode zeigt:
  1. $\approx_{M'}$ ist eine Verfeinerung von $\sim_L$ ($\approx_{M'} \subseteq \sim_L$)
  2. $\approx_M = \sim_L$

  Daher ist $\mathit{Index}(\approx_{M'}) \geq \mathit{Index}(\approx_M)$ und daher auch $|Z'| = \mathit{Index}(\approx_{M'}) \geq |Z| = \mathit{Index}(\approx_M)$.

Minimalautomat

Beweis (Teil 2): (alle minimalen DFA bis auf Umbenennung identisch):

- Sei $M = (Z, \Sigma, \delta, z_0, E)$ der Nerode-Automat der Sprache L und $M' = (Z', \Sigma, \delta', z_0', E')$ ein minimaler DFA mit $L(M') = L$
- Minimalität impliziert $|Z| = |Z'|$.
- Da $\approx_{M'} \subseteq \sim_L$ gilt: $\approx_{M'} = \sim_L$.
- Da auch $\sim_L = \approx_M$ gilt, folgt:
$$f : Z \to Z' \text{ mit } f([u]_{\approx_M}) = f([u]_{\sim_L}) := [u]_{\approx'_M}$$
ist ein Isomorphismus auf den Zuständen
- Es gilt auch
$$\begin{aligned} f(\delta([u]_{\sim_L}, a)) &= f([ua]_{\sim_L}) \\ &= [ua]_{\approx_{M'}} \\ &= \delta'([u]_{\approx_{M'}}, a) \\ &= \delta'(f([u]_{\sim_L}, a)) \end{aligned}$$
- D.h. die Automaten M und M' verhalten sich (bis auf die Umbenennung f) identisch.

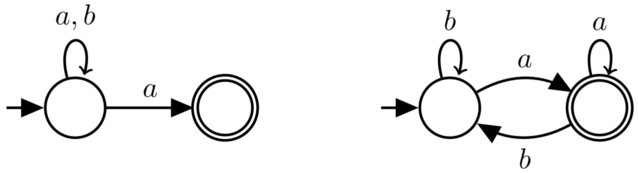
Minimalautomaten: NFAs

Der vorherige Satz zeigt:

Der Nerode-Automat ist eindeutig und minimal.

Für **NFAs** git das im Allgemeinen nicht!

Beispiel:



Beide Automaten erkennen $\{ua \mid u \in \{a,b\}^*\}$ und haben eine minimale Zustandsanzahl, aber sind strukturell verschieden!

Äquivalenzklassenautomat

Definition (Äquivalenzklassenautomat)

Sei $M = (Z, \Sigma, \delta, z_0, E)$ ein DFA. Wir nennen zwei Zustände $z, z' \in Z$ äquivalent und schreiben $z \equiv z'$ falls gilt:

$$\text{für alle } w \in \Sigma^* : \widehat{\delta}(z, w) \in E \iff \widehat{\delta}(z', w) \in E.$$

Der Äquivalenzklassenautomat zu M ist der DFA $M' = (Z', \Sigma, \delta', z_0', E')$ mit

$$\begin{aligned} Z &= \{[z]_\equiv \mid z \in Z\}, \\ z_0' &= [z_0]_\equiv, \\ E' &= \{[z]_\equiv \mid z \in E\} \text{ und} \\ \delta'([z]_\equiv, a) &= [\delta(z, a)]_\equiv. \end{aligned}$$

Äquivalenzklassenautomat (2)

Der Automat ist wohldefiniert:

- E' ist wohldefiniert:
Aus $z \equiv z'$ folgt, dass $z \in E \iff z' \in E$ gilt (dies folgt mit $w = \varepsilon$ aus $z \equiv z'$, denn $\widehat{\delta}(z, w) \in E \iff \widehat{\delta}(z', w) \in E$)
- $\delta([z]_\equiv, a)$ ist eindeutig für alle $[z]_\equiv \in Z', a \in \Sigma$,
d.h aus $z \equiv z'$ folgt $\delta(z, a) \equiv \delta(z', a)$:

  Sei $z \equiv z'$, dann gilt $\widehat{\delta}(z, w) \in E \iff \widehat{\delta}(z', w) \in E$ und damit auch $\widehat{\delta}(z, aw) \in E \iff \widehat{\delta}(z', aw) \in E$ und damit auch $\widehat{\delta}(\delta(z, a), w) \in E \iff \widehat{\delta}(\delta(z', a), w) \in E$. D. h. es gilt dann auch $\delta(z, a) \equiv \delta(z', a)$.

Äquivalenzklassenautomat: Korrektheit und Minimalität

Satz

Sei $M = (Z, \Sigma, \delta, z_0, E)$ ein DFA und $M' = (Z', \Sigma, \delta', z_0', E')$ der Äquivalenzklassenautomat zu M. Dann gilt

1. $L(M) = L(M')$
2. Falls alle Zustände in Z vom Startzustand z_0 erreichbar sind, dann ist M' minimal.

Beweis (Teil 1):

Sei $w \in \Sigma^*$. Dann gilt:

- M durchläuft die Zustandsfolge $q_0, \ldots, q_{|w|}$ entlang w und akzeptiert w genau dann, wenn $q_{|w|} \in E$ gilt.
- M' durchläuft die Zustandsfolge $[q_0]_\equiv, \ldots, [q_{|w|}]_\equiv$ und akzeptiert w genau dann, wenn $[q_{|w|}]_\equiv \in E'$ gilt.

Da per Definition $[q_{|w|}]_\equiv \in E'$ genau dann gilt, wenn $q_{|w|} \in E$ gilt, folgt, dass M und M' dieselben Worte akzeptieren.

Äquivalenzklassenautomat: Korrektheit und Minimalität(2)

Satz

Sei $M = (Z, \Sigma, \delta, z_0, E)$ ein DFA und $M' = (Z', \Sigma, \delta', z_0', E')$ der Äquivalenzklassenautomat zu M. Dann gilt

1. $L(M) = L(M')$
2. Falls alle Zustände in Z vom Startzustand z_0 erreichbar sind, dann ist M' minimal.

Beweis (Teil 2):

- Da alle $z \in Z$ erreichbar, sind alle $z' \in Z'$ erreichbar.
- Für Minimalität genügt es zu zeigen: $|Z'| \leq Index(\sim_L)$
- Zeige für jede Äquivalenzklasse $[v]_{\sim_L}$:
  für alle $u \in [v]_{\sim_L}$ ist $\widehat{\delta}(z_0', u)$ derselbe Zustand
  (dann gibt es nicht mehr als $Index(\sim_L)$ erreichbare Zustände.)
- Da alle $z' \in Z'$ erreichbar, gilt damit $|Z'| = Index(\sim_L)$

Äquivalenzklassenautomat: Korrektheit und Minimalität(3)

Satz

Sei $M = (Z, \Sigma, \delta, z_0, E)$ ein DFA und $M' = (Z', \Sigma, \delta', z_0', E')$ der Äquivalenzklassenautomat zu M. Dann gilt

1. $L(M) = L(M')$
2. Falls alle Zustände in Z vom Startzustand z_0 erreichbar sind, dann ist M' minimal.

Beweis (Teil 2, Forts.):

- Seien $u, u' \in [v]_{\sim_L}$, d. h. $u \sim_L u'$. Dann gilt
  $\forall w \in \Sigma^* : uw \in L \iff u'w \in L$,
  woraus folgt: $\forall w \in \Sigma^* : \widehat{\delta'}(z_0', uw) \in E' \iff \widehat{\delta'}(z_0', u'w) \in E'$
  woraus folgt: $\forall w \in \Sigma^* : \widehat{\delta'}(\widehat{\delta'}(z_0', u), w) \in E' \iff \widehat{\delta'}(\widehat{\delta'}(z_0', u'), w) \in E'$
  woraus folgt: $\widehat{\delta'}(z_0', u) \equiv \widehat{\delta'}(z_0', u')$
  woraus folgt: $[\widehat{\delta'}(z_0', u)]_\equiv = [\widehat{\delta'}(z_0', u')]_\equiv$.

Zustandsminimierung von DFAs

Idee:

- Berechne äquivalente Zustände (bezüglich $\equiv$)
- Bilde Äquivalenzklassenautomat, indem äquivalente Zustände verschmolzen werden.

Berechnung äquivalenter Zustände

Ideen

- Markiere Paare von Zuständen, die verschieden sein müssen (initial alle $\{z, z'\}$ mit $z \in E$, $z' \notin E$).
- Vervollständige das Markieren, durch Untersuchen von Übergängen:
  - Wenn $\{z, z'\}$ noch nicht markiert: Prüfe für jedes $a \in \Sigma$, ob die beiden Nachfolger $\{\delta(z, a), \delta(z', a)\}$ markiert sind.
  - Falls ja, dann markiere $\{z, z'\}$.
  - Wiederhole, bis sich nichts mehr ändert.
- Alle am Ende unmarkierten Paare sind äquivalente Zustände..

Algorithmus 3: Berechnung aller äquivalenten Zustände

Eingabe: DFA $M = (Z, \Sigma, \delta, z_0, E)$, der keine unerreichbaren Zustände hat
Ausgabe: Zustandspaare $\{z, z'\}$ mit $z \neq z'$ für die gilt $z \equiv z'$

```
Beginn
  stelle Tabelle T aller Zustandspaare {z, z'} mit z ≠ z' und z, z' ∈ Z auf;
  markiere alle Paare {z, z'} in T mit z ∈ E und z' ∉ E;
  wiederhole
    für jedes unmarkierte Paar {z, z'} in T tue
      für jedes a ∈ Σ tue
        wenn {δ(z, a), δ(z', a)} in T markiert ist dann
          markiere {z, z'} in T;
        Ende
      Ende
    Ende
  bis sich T nicht mehr verändert;
  return {{z, z'} | {z, z'} ist nicht markiert in T}
Ende
```


Korrektheit

Satz

Sei $M = (Z, \Sigma, \delta, z_0, E)$ ein DFA, der keine unerreichbaren Zustände hat. Algorithmus 3 berechnet äquivalente Zustandspaare von M und es gibt keine weitere äquivalenten Paare.

Beweis (Teil 1): Wird $\{z, z'\}$ markiert, dann gilt $z \not\equiv z'$.

- Wir zeigen: Für jedes markierte Paar $\{z, z'\}$ gibt es Wort w mit $\neg(\widehat{\delta}(z, w) \in E \iff \widehat{\delta}(z', w) \in E)$
- Induktion über Anzahl Schleifeniterationen bis $\{z, z'\}$ markiert wird.
- Basis: 0 Iterationen, $\{z, z'\}$ wird vor der Schleife markiert, $w = \varepsilon$ erfüllt Behauptung.
- Schritt: Mehr als 0 Iterationen. Dann wird $\{z, z'\}$ markiert, weil es $a \in \Sigma$ und ein markiertes Paar $\{q, q'\}$ gibt mit $\delta(z, a) = q$ und $\delta(z', a) = q'$. Induktionsannahme liefert Wort w' mit $\neg(\widehat{\delta}(q, w') \in E \iff \widehat{\delta}(q', w') \in E)$.
- Mit $w = aw'$ folgt: $\neg(\widehat{\delta}(z, w) \in E \iff \widehat{\delta}(z', w) \in E)$.

Korrektheit (2)

Satz

Sei $M = (Z, \Sigma, \delta, z_0, E)$ ein DFA, der keine unerreichbaren Zustände hat. Algorithmus 3 berechnet äquivalente Zustandspaare von M und es gibt keine weitere äquivalenten Paare.

Beweis (Teil 2): Wenn $z \not\equiv z'$, dann markiert Alg. 3 das Paar $\{z, z'\}$

- Beweis durch Widerspruch. Annahme es gibt Paare $z \not\equiv z'$, die Alg. 3 nicht markiert. Wähle Paar $\{z, z'\}$, für welches es ein minimal langes Wort w gibt mit $\neg(\widehat{\delta}(z, w) \in E \iff \widehat{\delta}(z', w) \in E)$.
- Wenn $w = \varepsilon$, dann wird $\{z, z'\}$ vor Schleife markiert. Widerspruch!
- Wenn $w = aw'$ mit $a \in \Sigma$, dann gilt: Wenn $\{\delta(z, a), \delta(z', a)\}$ vom Algorithmus markiert wird, dann auch $\{z, z'\}$. Daher: $\{\delta(z, a), \delta(z', a)\}$ wird nicht markiert. Aber dann gilt für w': $\neg\left(\widehat{\delta}(\delta(z, a), w') \in E \iff \widehat{\delta}(\delta(z', a'), w') \in E\right)$ und $|w'| < |w|$. Widerspruch zur Minimalität!

Laufzeit

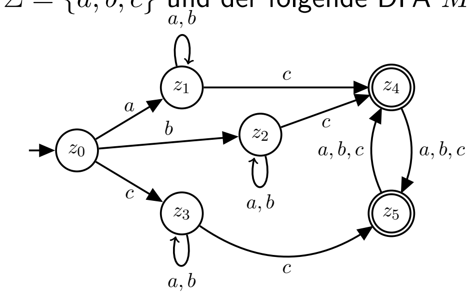
- Darstellung der Tabelle T:
  zweidimensionales Array der Größe $|Z| \times |Z|$
- Ermöglicht konstanten Zugriff auf Markierungen
- Pro Durchlauf der Schleife: $O(|Z|^2|\Sigma|)$
- Anzahl der Durchläufe ist durch $|Z|^2$ begrenzt, da es nur $|Z|^2$ Paare gibt und mindestens 1 Paar pro Durchlauf markiert wird
- Restliche Schritte: Konstante Laufzeit
- Daher: Algorithmus 3 kann in Zeit $O(|Z|^4 \cdot |\Sigma|)$ implementiert werden
- Tatsächlich gibt es effizientere Implementierungen: $O(|\Sigma| \cdot |Z| \log |Z|)$

Algorithmus 4: Minimierung von DFAs

Eingabe: DFA $M = (Z, \Sigma, \delta, z_0, E)$
Ausgabe: Minimaler DFA M' mit $L(M) = L(M')$
Beginn
  entferne Zustände aus M, die nicht vom Startzustand aus erreichbar sind;
  berechne äquivalente Zustände mit Algorithmus 3;
  erzeuge den Äquivalenzklassenautomat, indem die berechneten äquivalenten Zustände vereinigt werden;
Ende

Beispiel zur Minimierung

Sei $\Sigma = \{a, b, c\}$ und der folgende DFA M gegeben:



z_1	X				
z_2	X				
z_3	X				
z_4	X	X	X	X	
z_5	X	X	X	X	
	z_0	z_1	z_2	z_3	z_4

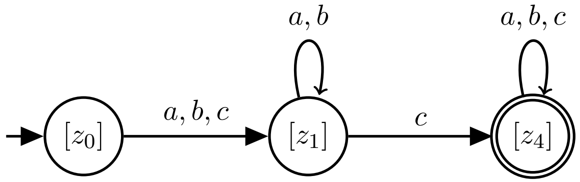
- alle Zustände sind von z_0 aus erreichbar
- äquivalente Zustände berechnen: Tabelle T erstellen
- Initiales Markieren: $\{z, z'\}$ mit $z \in \{z_0, z_1, z_2, z_3\}$ und $z' \in \{z_4, z_5\}$
- $\{z_0, z_1\}$, da $\{\delta(z_0, c), \delta(z_1, c)\} = \{z_3, z_4\}$ bereits markiert ist,
- $\{z_0, z_2\}$, da $\{\delta(z_0, c), \delta(z_2, c)\} = \{z_3, z_4\}$ bereits markiert ist, und
- $\{z_0, z_3\}$, da $\{\delta(z_0, c), \delta(z_3, c)\} = \{z_3, z_5\}$ bereits markiert

Beispiel zur Minimierung (2)

Ergibt $z_1 \equiv z_3$, $z_1 \equiv z_2$, $z_2 \equiv z_3$, $z_4 \equiv z_5$ und daher die Äquivalenzklassen

$$[z_0]_\equiv = \{z_0\},\ [z_1]_\equiv = \{z_1, z_2, z_3\},\ z_{4\equiv} = \{z_4, z_5\}$$

und den Minimalautomaten



Abschlusseigenschaften

Satz
Die regulären Sprachen sind abgeschlossen bezüglich Vereinigung, Produkt und Kleeneschem Abschluss.

Beweis:
Vereinigung, Produkt und Kleenescher Abschluss werden durch reguläre Ausdrücke erzeugt und sind daher reguläre Sprachen:

- Benutze reguläre Ausdrücke α_1, α_2 mit $L(\alpha_i) = L_i$.
- $(\alpha_1|\alpha_2)$ erzeugt $L(\alpha_1|\alpha_2) = L(\alpha_1) \cup L(\alpha_2) = L_1 \cup L_2$,
- $\alpha_1\alpha_2$ erzeugt $L(\alpha_1\alpha_2) = L(\alpha_1)L(\alpha_2) = L_1L_2$ und
- $(\alpha_1)^*$ erzeugt $L(\alpha_1)^* = L_1^*$.

Abschlusseigenschaften (2)

Satz
Die regulären Sprachen sind abgeschlossen bezüglich Komplementbildung.

Beweis:

- Sei $M = (Z, \Sigma, \delta, z_0, E)$ ein DFA der L akzeptiert.
- Dann akzeptiert $\overline{M} = (Z, \Sigma, \delta, z_0, Z \setminus E)$ die Sprache $\overline{L}$ (d. h. das Komplement von L):
  Offensichtlich gilt $(\widehat{\delta}(z_0, w) \in E) \iff \neg(\widehat{\delta}(z_0, w) \in Z \setminus E)$
- Daher ist $\overline{L}$ regulär.

Abschlusseigenschaften (3)

Satz
Die regulären Sprachen sind abgeschlossen bezüglich Schnitt.

Beweis:

- Folgt aus $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$ und da reguläre Sprachen abgeschlossen bez. Vereinigung und Komplementbildung.

Alternativ:

- Seien $M_1 = (Z_1, \Sigma, \delta_1, z_{01}, E_1)$ und $M_2 = (Z_2, \Sigma, \delta_2, z_{02}, E_2)$ DFAs, die $L_1 = L(M_1)$ und $L_2 = L(M_2)$ akzeptieren.
- Produktautomat von M_1 und M_2 ist der DFA
  $M = (Z_1 \times Z_2, \Sigma, \delta, (z_{01}, z_{02}), E_1 \times E_2)$ mit
  $\delta((z, z'), a) = (\delta_1(z, a), \delta_2(z', a))$ für alle $a \in \Sigma$ und $(z, z') \in Z_1 \times Z_2$.
- M akzeptiert $L_1 \cap L_2$, denn es gilt:
  $\widehat{\delta}((z_{0,1}, z_{0,2}), w) \in (E_1, E_2) \iff \Big(\widehat{\delta_1}(z_{0,1}, w) \in E_1 \wedge \widehat{\delta_2}(z_{0,2}, w) \in E_2\Big)$.

Abschluseigenschaften zusammengefasst

Theorem (Abschlusseigenschaften der regulären Sprachen)
Die regulären Sprachen sind abgeschlossen bezüglich Vereingung, Schnitt, Komplementbildung, Produkt und Kleeneschem Abschluss.

Nichtregularität zeigen mit Abschlusseigenschaften (1)

Satz

Die Sprache $L = \{a^n \mid n \text{ ist keine Primzahl}\}$ ist nicht regulär.

Beweis durch Widerspruch:

- Annahme: L ist regulär.
- Dann ist $\overline{L}$ auch regulär.
- $\overline{L} = \{a\}^* \setminus L = \{a^n \mid n \text{ ist eine Primzahl}\}$ ist nicht regulär (bereits gezeigt)
- Widerspruch! Daher ist L nicht regulär.

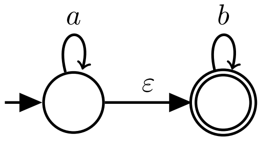
Nichtregularität zeigen mit Abschlusseigenschaften (2)

Satz

Die Sprache $L = \{w \in \{a, b\}^* \mid \#_a(w) = \#_b(w)\}$ ist nicht regulär.

Beweis durch Widerspruch:

- Annahme: L ist regulär.
- Sprache $L' = \{a^n b^m \mid n, m \in \mathbb{N}\}$ ist regulär, da der folgende NFA mit ε-Übergängen L' erkennt:



- Da L und L' regulär sind, ist auch $L \cap L'$ regulär.
- $L \cap L' = \{a^j b^j \mid j \in \mathbb{N}\}$ ist aber nicht regulär (wie bereits gezeigt)
- Widerspruch! D.h. L ist nicht regulär.

Entscheidbarkeitsresultate: Wortproblem

- Bereits gezeigt: Das Wortproblem für Typ 1,2,3-Sprachen ist entscheidbar.
- Für DFAs ist das Wortproblem in Linearzeit in der Länge des Wortes lösbar, denn die Berechnung von $\widehat{\delta}(z_0, w)$ braucht für einen DFA nur $|w|$ Schritte.

Entscheidbarkeitsresultate: Leerheitsproblem

Satz

Das Leerheitsproblem für reguläre Sprachen ist entscheidbar.

Beweis:

- Sei L reguläre Sprache und sei M ein DFA mit $L(M) = L$.
- Dann gilt $L = \emptyset$ genau dann, wenn es keinen Pfad vom Startzustand zu einem Endzustand in M gibt.
- Dies kann man leicht mit einer Tiefensuche auf dem Zustandsgraph von M überprüfen.

Endlichkeitsproblem für reguläre Sprachen

Satz
Das Endlichkeitsproblem für reguläre Sprachen ist entscheidbar.

Beweis:

- Sei L regulär und M ein DFA mit $L(M) = L$
- Es gilt $|L| < \infty$ g.d.w. es keinen Pfad vom Startzustand zu einem Endzustand gibt, der eine Schleife enthält.
- Prüfe dies mit einer Tiefensuche auf dem Zustandsgraph von M.

Schnittproblem für reguläre Sprachen

Satz
Das Schnittproblem für reguläre Sprachen ist entscheidbar.

Beweis:

- Seien L_1, L_2 reguläre Sprachen.
- Seien M_1, M_2 DFAs mit $L(M_i) = L_i$.
- Berechne den Produktautomaten M mit $L(M) = L_1 \cap L_2$
- Prüfe das Leerheitsproblem für $L(M)$.

Äquivalenzproblem für reguläre Sprachen

Satz
Das Äquivalenzproblem für reguläre Sprachen ist entscheidbar.

Beweis:

- Seien L_1, L_2 reguläre Sprachen.
- Seien M_1, M_2 DFAs mit $L(M_i) = L_i$.
- Berechne die Minimalautomaten von M_1 und M_2
- Prüfe die Minimalautomaten auf Isomorphie.

Zusammenfassung

- Pumping-Lemma
  (Pumping-Eigenschaft notwendig, nicht hinreichend!)
- Der Satz von Myhill-Nerode:
  L regulär g.d.w. $Index(\sim_L) < \infty$
- Nicht-Regularität nachweisen:
  Pumping-Lemma oder $Index(\sim_L) = \infty$
- Äquivalenzklassenautomat und Minimierung von DFAs
- Abschlusseigenschaften regulärer Sprachen
- Entscheidbarkeitsresultate für reguläre Sprachen