

Theoretische Grundlagen der Informatik

Vorlesung am 30. Oktober 2014

INSTITUT FÜR THEORETISCHE INFORMATIK



Satz:

Sei L eine reguläre Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, so dass für jedes Wort $w \in L$ mit $|w| \geq n$ und jede Darstellung $w = tyx$ mit $|y| = n$ gilt:

Für das Teilwort y existiert eine Darstellung $y = uvz$ mit $v \neq \varepsilon$ bei der auch $tuv^i zx \in L$ ist für alle $i \in \mathbb{N}_0$.

Satz:

Sei L eine reguläre Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, so dass für jedes Wort $w \in L$ mit $|w| \geq n$ und jede Darstellung $w = tyx$ mit $|y| = n$ gilt:

Für das Teilwort y existiert eine Darstellung $y = uvz$ mit $v \neq \varepsilon$ bei der auch $tuv^i zx \in L$ ist für alle $i \in \mathbb{N}_0$.

Beweis:

- Sei L eine reguläre Sprache.
- Sei $\mathcal{A} = (Q, \Sigma, \delta, s, F)$ der deterministische endliche Automat, der L erkennt.
- Setze $n := |Q| + 1$.
- Sei $tyx \in L$ mit $|y| = n$.
- Sei q_0, \dots, q_n die Folge der Zustände, die bei der Abarbeitung von y durchlaufen werden.

Beweis:

- Sei L eine reguläre Sprache.
- Sei $\mathcal{A} = (Q, \Sigma, \delta, s, F)$ der deterministische endliche Automat, der L erkennt.
- Setze $n := |Q| + 1$.
- Sei $tyx \in L$ mit $|y| = n$.
- Sei q_0, \dots, q_n die Folge der Zustände, die bei der Abarbeitung von y durchlaufen werden.
- Es enthält q_0, \dots, q_n mindestens einen Zykel
- Es gibt Zerlegung $y = uvz$ so dass v der Buchstabenfolge entspricht, die beim Durchlaufen des Zyklus abgearbeitet wird.
- Insbesondere ist v nicht leer.
- Dieser Zykel kann dann beliebig oft durchlaufen werden.
- Also ist auch $tuv^i zx$ ein gültiges Wort, das der Automat erkennt.

Satz:

Sei L eine reguläre Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, so dass für jedes Wort $w \in L$ mit $|w| \geq n$ und jede Darstellung $w = tyx$ mit $|y| = n$ gilt:

Für das Teilwort y existiert eine Darstellung $y = uvz$ mit $v \neq \varepsilon$ bei der auch $tuv^i zx \in L$ ist für alle $i \in \mathbb{N}_0$.

- Es enthält q_0, \dots, q_n mindestens einen Zykel
- Es gibt Zerlegung $y = uvz$ so dass v der Buchstabenfolge entspricht, die beim Durchlaufen des Zyklus abgearbeitet wird.
- Insbesondere ist v nicht leer.
- Dieser Zykel kann dann beliebig oft durchlaufen werden.
- Also ist auch $tuv^i zx$ ein gültiges Wort, das der Automat erkennt.

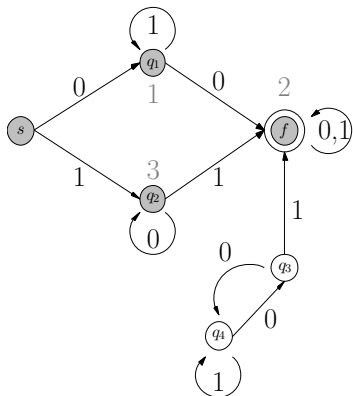
- **Minimierung von Automaten**
- **Äquivalenzklassenautomat**

Frage: Kann man konstruktiv die Anzahl der Zustände eines deterministischen endlichen Automaten erheblich verringern?

Frage: Kann man konstruktiv die Anzahl der Zustände eines deterministischen endlichen Automaten erheblich verringern?

Definition:

Zustände eines (deterministischen) endlichen Automaten, die vom Anfangszustand aus nicht erreichbar sind, heißen **überflüssig**.



- Wir können endliche Automaten als gerichtete Graphen auffassen.
- Die überflüssigen Zustände entsprechen dann den Knoten, zu denen es vom Anfangsknoten aus keinen gerichteten Weg gibt.
- Eine Tiefensuche (**Depth-First Search**, DFS) in dem Graphen liefert damit alle nicht überflüssigen Zustände.

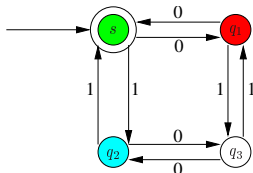
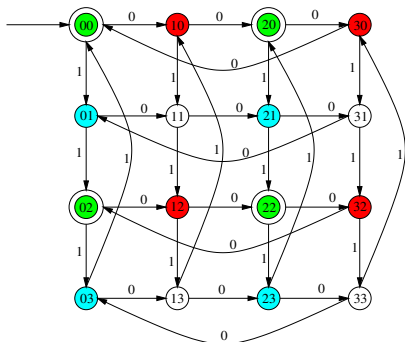
Satz:

Die Menge aller überflüssigen Zustände eines (deterministischen) endlichen Automaten kann in der Zeit $\mathcal{O}(|Q| \cdot |\Sigma|)$ berechnet werden.

Beweis: Wende DFS ab dem Startzustand an. Dies erfordert einen Aufwand proportional zu der Anzahl der Kanten in dem Graphen.

- Ein deterministischer endlicher Automat ohne überflüssige Zustände muss jedoch noch nicht minimal sein.

Beispiel



Beide Automaten akzeptieren die Sprache

$$L = \{w \in \{0, 1\}^* \mid (|w|_0 \bmod 2) = (|w|_1 \bmod 2) = 0\}$$

mit $|w|_a$ = Anzahl der Vorkommen des Zeichens $a \in \Sigma$ in w

- Zwei Zustände haben dasselbe Akzeptanzverhalten, wenn es für das Erreichen eines Endzustandes durch Abarbeiten eines Wortes w unerheblich ist, aus welchem der beiden Zustände wir starten.
- Reduktion der Anzahl der Zustände durch Zusammenlegen der Zustände mit gleichem Akzeptanzverhalten
- Letzten Beispiel: Färbung der Zustände mit gleichem Verhalten durch gleiche Farben

Definition (Äquivalenz):

Zwei Zustände p und q eines deterministischen endlichen Automaten heißen **äquivalent** ($p \equiv q$), wenn für alle Wörter $w \in \Sigma^*$ gilt:

$$\delta(p, w) \in F \iff \delta(q, w) \in F.$$

Offensichtlich ist \equiv eine Äquivalenzrelation. Mit $[p]$ bezeichnen wir die Äquivalenzklasse der zu p äquivalenten Zustände.

- Zwei Zustände haben dasselbe Akzeptanzverhalten, wenn es für das Erreichen eines Endzustandes durch Abarbeiten eines Wortes w unerheblich ist, aus welchem der beiden Zustände wir starten.
- Reduktion der Anzahl der Zustände durch Zusammenlegen der Zustände mit gleichem Akzeptanzverhalten
- Letzten Beispiel: Färbung der Zustände mit gleichem Verhalten durch gleiche Farben

Definition (Äquivalenz):

Zwei Zustände p und q eines deterministischen endlichen Automaten heißen **äquivalent** ($p \equiv q$), wenn für alle Wörter $w \in \Sigma^*$ gilt:

$$\delta(p, w) \in F \iff \delta(q, w) \in F.$$

Offensichtlich ist \equiv eine Äquivalenzrelation. Mit $[p]$ bezeichnen wir die Äquivalenzklasse der zu p äquivalenten Zustände.

Definition (Äquivalenzklassenautomat):

Zu einem DEA $\mathcal{A} = (Q, \Sigma, \delta, s, F)$ definieren wir den Äquivalenzklassenautomaten $\mathcal{A}^{\equiv} = (Q^{\equiv}, \Sigma^{\equiv}, \delta^{\equiv}, s^{\equiv}, F^{\equiv})$ durch:

- $Q^{\equiv} := \{[q] \mid q \in Q\}$
- $\Sigma^{\equiv} := \Sigma$
- $\delta^{\equiv}([q], a) := [\delta(q, a)]$
- $s^{\equiv} := [s]$
- $F^{\equiv} := \{[f] \mid f \in F\}$

Definition (Äquivalenzklassenautomat):

Zu einem DEA $\mathcal{A} = (Q, \Sigma, \delta, s, F)$ definieren wir den Äquivalenzklassenautomaten $\mathcal{A}^{\equiv} = (Q^{\equiv}, \Sigma^{\equiv}, \delta^{\equiv}, s^{\equiv}, F^{\equiv})$ durch:

- $Q^{\equiv} := \{[q] \mid q \in Q\}$
- $\Sigma^{\equiv} := \Sigma$
- $\delta^{\equiv}([q], a) := [\delta(q, a)]$
- $s^{\equiv} := [s]$
- $F^{\equiv} := \{[f] \mid f \in F\}$

Satz:

Der Äquivalenzklassenautomat \mathcal{A}^{\equiv} zu einem deterministischen endlichen Automaten \mathcal{A} ist wohldefiniert.

Satz:

Der Äquivalenzklassenautomat \mathcal{A}^{\equiv} zu einem deterministischen endlichen Automaten \mathcal{A} ist wohldefiniert.

Satz:

Der Äquivalenzklassenautomat \mathcal{A}^{\equiv} zu einem deterministischen endlichen Automaten \mathcal{A} ist wohldefiniert.

Beweis: Wir müssen zeigen, dass F^{\equiv} und δ^{\equiv} wohldefiniert sind, der Rest ist klar. Dazu zeigen wir:

- ein Endzustand kann nur zu einem Endzustand äquivalent sein,
- δ führt äquivalente Zustände beim Lesen desselben Symbols wieder in äquivalente Zustände über.

Satz:

Der Äquivalenzklassenautomat \mathcal{A}^{\equiv} zu einem deterministischen endlichen Automaten \mathcal{A} ist wohldefiniert.

- ein Endzustand kann nur zu einem Endzustand äquivalent sein,

Für ε gilt:

$$\delta(p, \varepsilon) \in F \Leftrightarrow \delta(q, \varepsilon) \in F .$$

Es ist

$$\delta(p, \varepsilon), \delta(q, \varepsilon) \in F \text{ genau für } p, q \in F .$$

Also:

$$\text{Falls } p \equiv q, \text{ dann gilt } p, q \in F \text{ oder } p, q \notin F .$$

Also ist F^{\equiv} wohldefiniert.

Satz:

Der Äquivalenzklassenautomat \mathcal{A}^{\equiv} zu einem deterministischen endlichen Automaten \mathcal{A} ist wohldefiniert.

- δ führt äquivalente Zustände beim Lesen desselben Symbols wieder in äquivalente Zustände über.

Sei $p \equiv q$. Dann gilt für alle $w \in \Sigma^*$

$$\delta(q, w) \in F \Leftrightarrow \delta(p, w) \in F$$

Somit gilt nach Definition von \equiv auch für alle $a \in \Sigma$:

$$\delta(\delta(q, a), w) = \delta(q, aw) \in F \Leftrightarrow \delta(p, aw) = \delta(\delta(p, a), w) \in F.$$

Damit folgt $\delta(q, a) \equiv \delta(p, a)$, also ist auch δ^{\equiv} wohldefiniert.

Satz:

Der Äquivalenzklassenautomat \mathcal{A}^{\equiv} zu \mathcal{A} akzeptiert dieselbe Sprache wie \mathcal{A} .

Satz:

Der Äquivalenzklassenautomat \mathcal{A}^{\equiv} zu \mathcal{A} akzeptiert dieselbe Sprache wie \mathcal{A} .

Beweis:

- Sei $w \in \Sigma^*$, $q_0 := s, q_1, \dots, q_n$ die Folge der Zustände, die von \mathcal{A} bei der Abarbeitung von w durchlaufen werden.
- Bei Abarbeitung von w in \mathcal{A}^{\equiv} werden dann die Zustände $[q_0], [q_1], \dots, [q_n]$ durchlaufen.
- \mathcal{A} akzeptiert w genau dann, wenn $q_n \in F$ gilt. \mathcal{A}^{\equiv} akzeptiert w genau dann, wenn $[q_n] \in F^{\equiv}$ gilt.
- Nach Definition von \mathcal{A}^{\equiv} ist $q_n \in F$ genau dann, wenn $[q_n] \in F^{\equiv}$ gilt.

Frage:

Wie konstruiert man \mathcal{A}^\equiv zu \mathcal{A} ? D.h. wie berechnet man alle Äquivalenzklassen zu den Zuständen von \mathcal{A} ?

Beweis der Äquivalenz von zwei Zuständen p scheint aufwendig:
Nach Definition muss nachgewiesen werden, dass für alle $w \in \Sigma^*$ gilt:

$$\delta(p, w) \in F \iff \delta(q, w) \in F.$$

- Es gibt jedoch unendlich viele $w \in \Sigma^*$.
- Es ist einfacher für p und q zu zeigen, dass p nicht äquivalent zu q ist.
- Dafür benötigen wir *nur ein* Wort $w \in \Sigma^*$ mit
 $\delta(p, w) \in F$ aber $\delta(q, w) \notin F$, oder
 $\delta(p, w) \notin F$ aber $\delta(q, w) \in F$.

Notation:

Wir bezeichnen ein solches Wort w als **Zeuge** für die Nichtäquivalenz von p und q und sagen w trennt p und q .

Idee: Teste systematisch Zustandspaare auf Nichtäquivalenz

- Betrachte alle Worte aus Σ^* in aufsteigender Länge.
- Überprüfe für jedes Wort, ob es Zeuge für Nichtäquivalenz von zwei Zuständen ist.

Frage

Wann kann dieses Verfahren abgebrochen werden?

- Sei $w = aw'$ ein kürzester Zeuge für $p \neq q$.
- Dann ist w' Zeuge für $p' := \delta(p, a) \neq \delta(q, a) =: q'$.
- Wenn es für $p' \neq q'$ einen kürzeren Zeugen w'' gäbe, so wäre aw'' ein kürzerer Zeuge für $p \neq q$ als w .
- Dies ist aber ein Widerspruch dazu, dass w ein kürzester Zeuge ist.
- **Fazit:** Wenn wir alle Wörter aus Σ^* in der Reihenfolge ihrer Länge darauf testen, ob sie Zeuge sind, und für eine bestimmte Länge kein Zeuge mehr für eine Nichtäquivalenz auftritt, so kann das Verfahren abgebrochen werden.

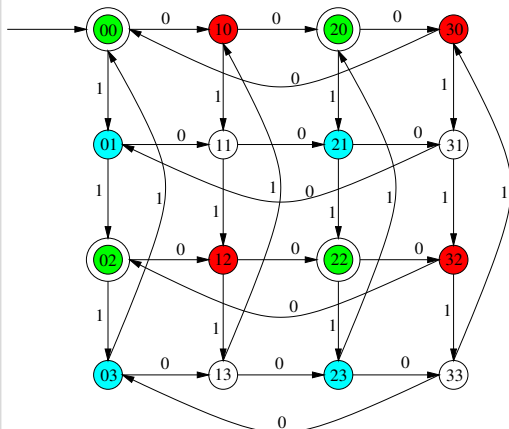
Vorgehensweise für die Konstruktion von \mathcal{A}^{\equiv} aus \mathcal{A}

- Betrachte alle Zustandspaare und zunächst ε ,
- dann alle Elemente aus Σ ,
- dann alle Wörter der Länge 2 aus Σ^* ,
- u.s.w.

Zunächst betrachte alle Zustände als eine Klasse.

- Dann trennt ε die Zustände aus F von denen aus $Q \setminus F$.
- Danach testen wir nur noch Paare von Zuständen aus F beziehungsweise $Q \setminus F$.
- Durch mindestens ein Wort der Länge 1 wird entweder F oder $Q \setminus F$ weiter getrennt, oder das Verfahren ist beendet.
- Dies wird iterativ so weitergeführt mit Wörtern wachsender Länge.

Beispiel zur Vorgehensweise



Vorläufige Äquivalenzklassen

vorher

$\{00, 01, 02, 03, 10, 11, 12, 13, 20, 21, 22, 23, 30, 31, 32, 33\}$

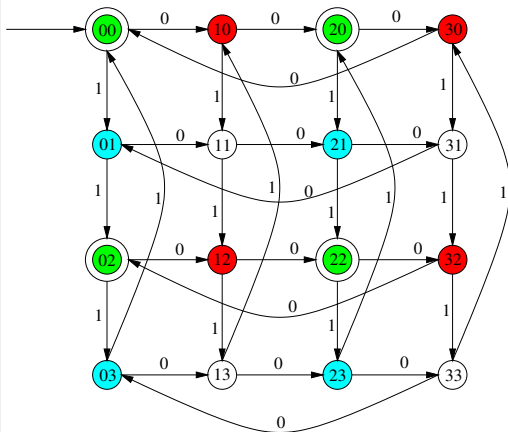
nachher

$\{00, 02, 20, 22\}$

$\{01, 03, 10, 11, 12, 13, 21, 23, 30, 31, 32, 33\}$

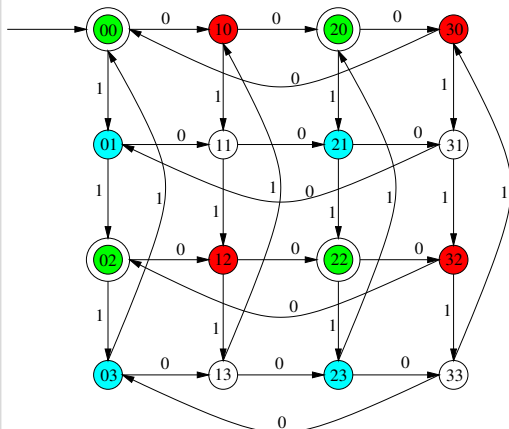
ε trennt $\underbrace{\{00, 02, 20, 22\}}_{\text{grün}}$ von $\{01, 03, 10, 11, 12, 13, 21, 23, 30, 31, 32, 33\}$

Beispiel zur Vorgehensweise



Die Wörter 00, 01, 10, 11 trennen keine Zustandspare mehr.

Beispiel zur Vorgehensweise



Äquivalenzklassen

$\{00, 02, 20, 22\}$
 $\{10, 30, 12, 32\}$
 $\{01, 03, 21, 23\}$
 $\{11, 13, 31, 33\}$

Fazit: Die Äquivalenzklassen der Zustände sind:
 $s = [00]$, $q_1 = [01]$, $q_2 = [10]$ und $q_3 = [11]$.