University of Applied Sciences
HOCHSCHULE EMDEN·LEER

Theoretische Informatik

Prof. Dr. Juho Mäkiö

07.11. 2018

Themen der Vorlesung insgesamt….

- Endliche Automaten (DEA, NEA und NEA mit epsilon-Übergängen),
- Kellerautomaten,
- reguläre Ausdrücke,
- Transformationen und Minimierung (NEA nach DEA,
- NEA/eps nach NEA,
- regulärer Ausdruck nach NEA/eps),
- reguläre und nicht-reguläre Sprachen,
- Grammatiken und kontextfreie Sprachen.

Ziel des Kurses

- Vermitteln von Grundkonzepten der Theoretischen Informatik.
- Studierenden beherrschen… / können…
  - die grundlegenden Begriffe, Konzepte und Methoden endlicher Automaten und Grammatiken und selbständig Automaten für bestimmte Problemstellungen entwickeln können.
  - die verschiedenen Transformationen,
  - Beweisen der Nicht-Regularität einer Sprache
  - den Zusammenhang zwischen Automaten und Grammatiken

AUTOMATENMINIMIERUNG

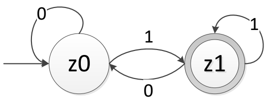
Automatenminimierung

Für eine Sprache L ⊂ Σ* kann es ggfs. mehrere Automaten geben, z.B. A_1, A_2

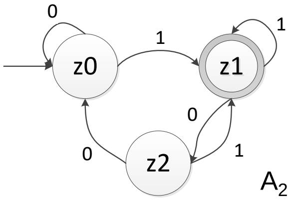
mit $L(A_1) = L(A_2)$

A_1 und A_2 sind dann äquivalent:

$A_1 \equiv A_2$



A_1



A_2

Automatenminimierung

Wann ist ein Automat A_1 besser als ein Automat A_2?

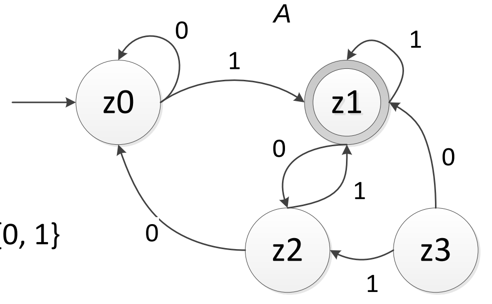
~ schwer zu beantworten

- leichter zu verstehen ?
- weniger Zustände → geeignetes Qualitätsmerkmal

~ bei der Implementierung eventuell besser / Effizienz

Minimalautomat

- A heißt Minimalautomat für L mit L(*A*) = L, falls es keinen Automat *A'* gibt mit L(*A'*) = L, der weniger Zustände als *A* hat.

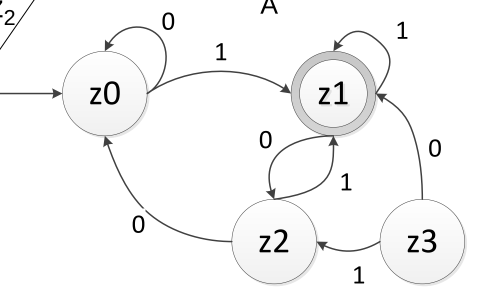


L(*A*) = {ω1 | ω ∈ Σ*} Σ = {0, 1}

Wie könnte man den Automaten *A* verbessern?

Minimalautomat

- Zwei „Verbesserungsansätze"
  - Entfernung von Zuständen, die von z_0 aus nicht erreichbar sind. (also: z_3)
  - Verschmelzung „äquivalenter („gleichwertiger") Zustände": hier : z_0, z_2



Was heisst das?

Äquivalente Zustände

- ~ Von z_0 bzw. z_2 aus wird dieselbe Sprache akzeptiert.

$$(z_0, \omega) \rightarrow^* (z, \varepsilon) \quad \text{mit } z \in E$$

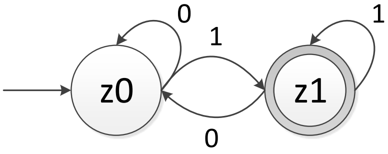
und

$$(z_2, \omega) \rightarrow^* (z, \varepsilon) \quad \text{mit } z \in E$$

z.B. $\omega = 0101$

Damit können z_0 und z_2 zusammengelegt werden:
mit L(A‘) = L(A)

A‘



A‘ ist Minimalautomat

Zwischenzusammenfassung

- Automat: $A = (Z, \Sigma, \delta, z_0, E)$
  - $\delta(z, a) = z'$ <=> $z \to z'$ oder $z > z'$ oder $z \to_a z'$
- Konfiguration: $(z, \omega) \mid z \in Z \wedge \omega \in \Sigma^*$
- Übergangsrelation: $(z, a\omega') \to (z', \omega')$ mit $\delta(z, a) = z'$
  - mehrfache Anwendung von -> : ->*
- Reguläre Sprache:
  - $L(A) = \{\omega \in \Sigma^* \mid$ **$(z_0, \omega) \to^* (z_e, \varepsilon)$**, $z_e \in E\}$ **Wird akzeptiert**
- äquivalente Automaten $A_1, A_2 : A_1 \equiv A_2 \quad L(A_1) = L(A_2)$
- Erreichbare Zustände:
  - $[z]^* \subset Z \; z' \in [z]^* : (z, \omega) \to^* (z', \varepsilon)$ für $\omega \in \Sigma^*$
  - Damit gilt: $z \in [z]^*$ für $\omega = \varepsilon$

Zwischenzusammenfassung

- Fehlerzustand z: $[z]^* \cap E = \emptyset$ (kein „Weg“ zum Endzustand)
- $L(A, z) = \{\omega \in \sum^* \mid (z, \omega) \to^* (z', \varepsilon), z' \in E\}$
  - A akzeptiert ω von Z aus
  - ~ als wenn z Startzustand wäre
- Minimalautomat (reduzierter Automat) A für L: L(A) = L und es gibt keinen Automaten A‘, der weniger Zustände als A hat
- Isomorphe Automaten A_1, A_2:
  - 2 Automaten $A_1 = (Z_1, \sum_1, z_{01}, E_1)$ und $A_2 = (Z_2, \sum_2, z_{02}, E_2)$ heißen isomorph, falls A_1, A_2 bis auf die Bezeichnung der Zustände „identisch“ sind: Es gibt eine bijektive Funktion:
    $f: Z_1 \to Z_2$ mit $f(z_{01}) = f(z_{02})$, $f(E_1) = E_2$
    $z \to_a z' \iff f(z) \to_a f(z')$

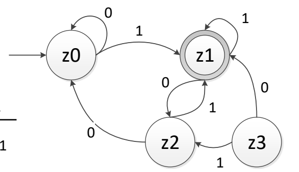
Zwischenzusammenfassung

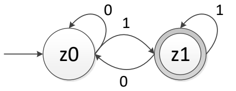
- Äquivalente Zustände:
  - z‘ und z sind äquivalent: z‘ ≡ z, falls L(A, z) = L(A, z‘) gilt. Damit wird von z bzw. z‘ aus dieselbe Sprache akzeptiert.
- Äquivalenzklassen:
  - $[z] = \{z' \in Z \mid z' \equiv z\}$ Menge der zu z äquivalenten Zustände (Äquivalenzrelation)
  - Es gilt: $z \in [z]$ -> s. Algorithmus zur Minimierung

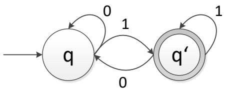
Beispiel

- A = (Z, ∑, δ, z_0, E) : Z = {z_0, z_1, z_2, z_3},
  ∑ = {0, 1} , z_0 , E = {z_1}
  δ(z_0, 1) = z_1 :
  z_0 -> z_1
  z_0 > z_1
  z_0 $->_1$ z_1

	0	1
z_0	z_0	z_1
z_1	z_2	z_1
z_2	z_0	z_1
z_3	z_1	z_2



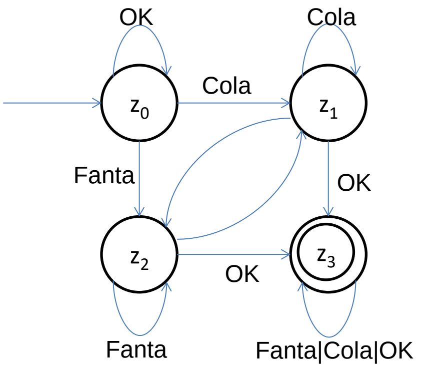




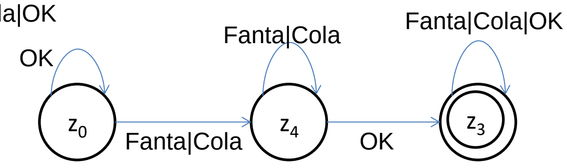
- (z_0, 0101) ~ Konfiguration
  (z_0, 0101) ->* (z_1, Ɛ)
- L(A) = {ω1 | ω ∈ ∑*} (reguläre Sprache)
- A, A‘ und A“ sind äquivalent:
  L(A) = L(A’) = L(A”) = {ω1 | ω ∈ ∑*}
- A‘ ist minimaler Automat
- A‘ und A“ sind isomorph

KONSTRUKTION VON MINIMALAUTOMATEN

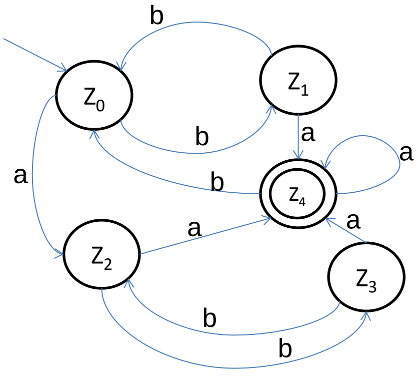
Einführungsbeispiel



Z_0				
Z_1	X			
Z_2	X			
Z_3	X	X	X	
	Z_0	Z_1	Z_2	Z_3



Beispiel 2

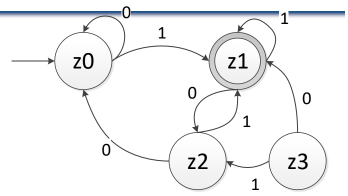


Z_0					
Z_1	X				
Z_2	X	X			
Z_3	X	X			
Z_4	X	X	X	X	
	Z_0	Z_1	Z_2	Z_3	Z_4

Paare: Endzustand-nicht Endzustand

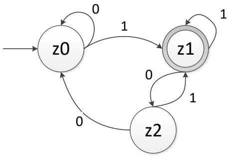
Markiere die Zustandspaare als nicht-äquivalent an, von denen man zum einem bereits markierten Zustandspaar kommt. (Ein Symbol reicht.)

Konstruktion eines minimalen Automaten



Überflüssige Zustände entfernen

z_3 von z_0 nicht erreichbar,
also überflüssig (-> Entfernen)

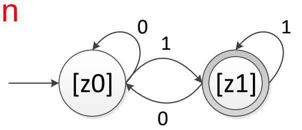


Äquivalente Zustände erkennen:

L(A, z0) = L(A, z2) also z0 ≡ z2

Äquivalente Zustände zusammenfassen

Es entstehen Automaten mit Zuständen, wobei jeder Zustand z eigentlich [z] repräsentiert:



isomorph zu obigem Automaten

2. Verfahren

	0	1
z_0	z_1	z_2
z_1	z_2	z_3

markiert

also (z_0, z_1) markieren

	0	1
z_0	z_1	z_2
z_2	z_1	z_3

markiert

also (z_0, z_2) markieren

	0	1
z_1	z_2	z_3
z_2	z_1	z_3

noch nichts markiert

Alle Möglichkeiten bearbeitet,
also (z_1, z_2) nicht markieren.

Ergebnis:

	z_0	z_1	z_2	z_3
z_0	≡			
z_1	X	≡		
z_2	X	≡	≡	
z_3	X	X	X	≡

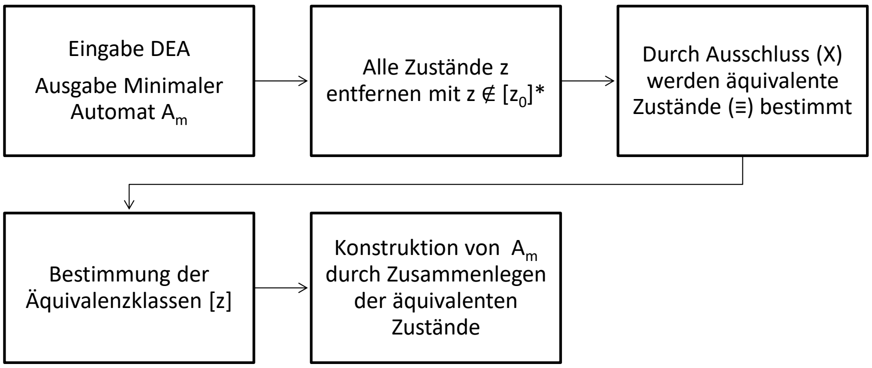
Vorgang wiederholen, bis alle Paare auf eine mögliche Nicht-Äquivalenz untersucht wurden.

Ergebnis: (z_1, z_2) ist nicht-nicht-äquivalent, also äquivalent.

Bildung der Äquivalenzklassen

$[z_0] = \{z_0\}$,
$[z_1] = \{z_1, z_2\} = [z_2]$,
$[z_3] = \{z_3\}$

Algorithmus zur Minimierung eines DEA

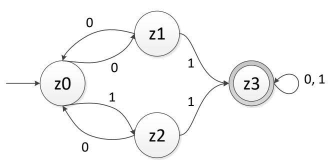


Beispiel

Gesucht minimaler Automat.
≡ ≙ äquivalent
X ≙ nicht äquivalent

	z_0	z_1	z_2	z_3
z_0	≡			
z_1		≡		
z_2			≡	
z_3	X	X	X	≡

1



2

	0	1
z_0	z_1	z_2
z_1	z_0	z_3

bereits markiert
=> (z_0, z_1) markieren

	0	1
z_0	z_1	z_2
z_2	z_0	z_3

bereits markiert
=> (z_0, z_2) markieren

	0	1
z_1	z_0	z_3
z_2	z_0	z_3

nichts markiert

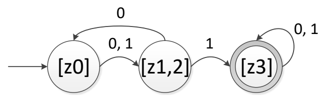
3

	z_0	z_1	z_2	z_3
z_0	≡			
z_1	X	≡		
z_2	X	≡	≡	
z_3	X	X	X	≡

4

$[z_0] = \{z_0\}$
$[z_1] = [z_2] = \{z_1, z_2\} = [z_{1,2}]$
$[z_3] = \{z_3\}$

5

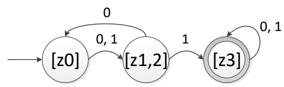


Beispiel

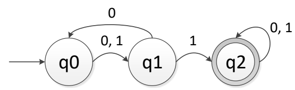
$\overline{A}$=($\overline{Z}$, Σ, $\overline{\delta}$, $\overline{z_0}$, $\overline{E}$)

$\overline{Z}$ = {[z_0], [$z_{1,2}$], [z_3]} Σ = {0, 1}

	0	1
[z_0]	[$z_{1,2}$]	[$z_{1,2}$]
[$z_{1,2}$]	[z_0]	[z_3]
[$z_{3/E}$]	[z_3]	[z_3]

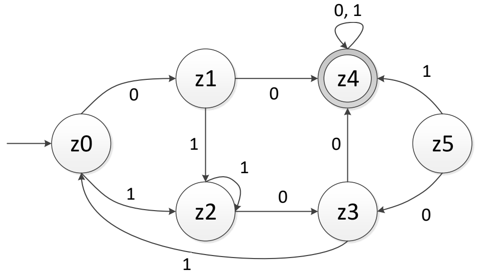


isomorph



Ein zu $\overline{A}$ isomorpher Automat ist: L(A) = L($\overline{A}$) = L(A')

Übung



Konstruieren Sie den minimalen Automaten $\overline{A}$!

Algorithmus zur Bestimmung des minimalen Automaten

Algorithmus Not-Equivalent (NEQ)

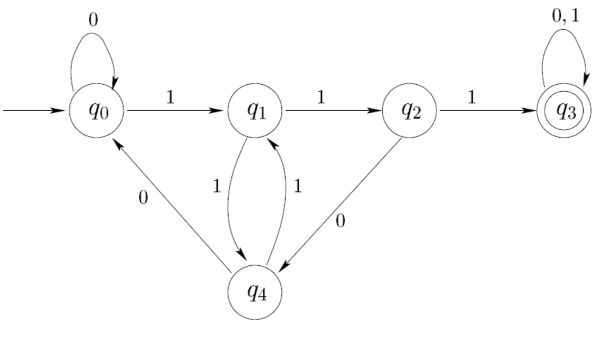
Bei Eingabe $A = (Q, \Sigma, \delta, q_0, F)$:

1. Markiere alle Paare $\{p, q\}$ von Zuständen, von denen genau einer ein akzeptierender Zustand ist.

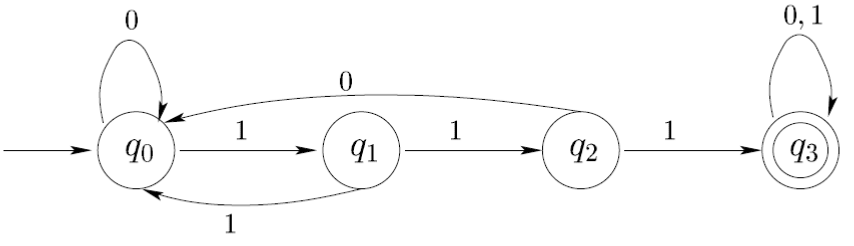
2. Solange es noch ein Paar $\{p, q\}$ von Zuständen p, q und ein Symbol $a \in \Sigma$ gibt, so dass das Paar $\{\delta(p, a), \delta(q, a)\}$ bereits markiert ist, markiere $\{p, q\}$.

Übung

- Konstruieren Sie den minimalen Automaten A‘ zum A

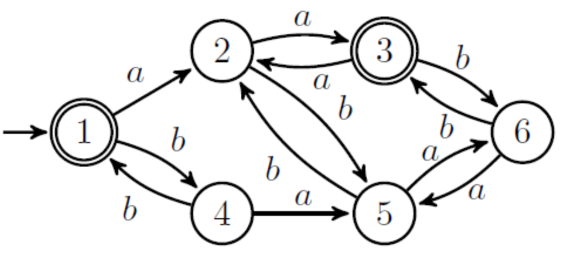


Lösung zur vorherigen Aufgabe

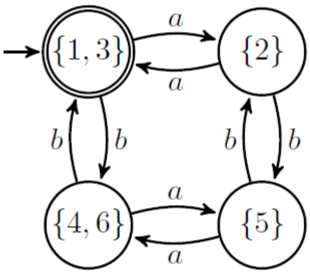


Eine Klausuraufgabe

Minimieren Sie M



Lösung



Yes we can…