Hans U. Simon
Annette Ilgen

Bochum, den 7.10.2008

Beispiele zur Vorlesung

Theoretische Informatik

WS 08/09

Vorbemerkung: Hier findet sich eine Sammlung von Beispielen und Motivationen zur Vorlesung Theoretische Informatik.

2 Reguläre Sprachen

2.1 endliche Automaten

2.1.1 Beispiel deterministischer endlicher Automat (DFA)

$Z = \{z_0, z_1, z_2, z_3, z_4\}$
$\Sigma = \{a, b\}$
$\delta : Z \times \Sigma \to Z$ mit
$\delta(z_0, a) = z_1, \delta(z_0, b) = z_5$
$\delta(z_1, a) = z_1, \delta(z_1, b) = z_2$
$\delta(z_2, a) = z_3, \delta(z_2, b) = z_4$
$\delta(z_3, a) = z_2, \delta(z_3, b) = z_4$
$\delta(z_4, a) = z_3, \delta(z_4, b) = z_2$
$\delta(z_5, a) = z_5, \delta(z_5, b) = z_5$
$S = z_0$ Startzustand
$E = \{z_2, z_4\}$

So wie eine Grammatik eine Sprache erzeugt, wird auch ein Automat genutzt um eine Sprache zu beschreiben. Der Automat erhält als Eingabe ein Wort. Dieses Wort wird Buchstabe für Buchstabe abgearbeitet. Erreicht der Automat nach Abarbeitung des Wortes einen Endzustand, so gilt für das Wort, dass es Element der Sprache ist.

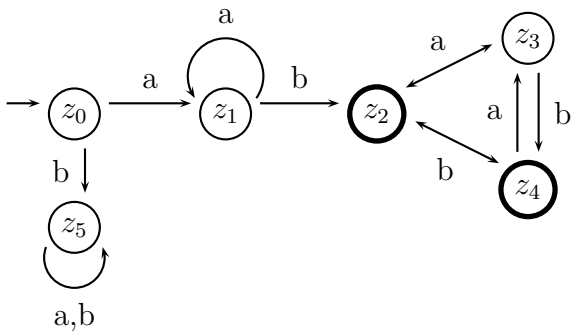
Wenden wir z.B. obigen Automaten auf $w = abb$ an.
Wir beginnen mit dem Startzustand z_0 und dem ersten Buchstaben von w.
$\delta(z_0, a) = z_1$
Wir erhalten z_1 und wenden es auf den zweiten Buchstaben von w an.
$\delta(z_1, b) = z_2$
Wir erhalten z_2 und wenden es auf den dritten und letzten Buchstaben an.
$\delta(z_2, b) = z_4$
Wir erhalten also nach Abarbeitung des Wortes den Zustand $z_4 \in E$ welcher ein Endzustand ist. Somit ist $w = abb \in L$.

Notation: Die Abbildung δ wird im Folgenden in Form einer Tabelle dargestellt, da dies übersichtlicher ist. Die Tabelle zu obigem Beispiel hat die Form:

δ	z_0	z_1	z_2	z_3	z_4	z_5
a	z_1	z_1	z_3	z_2	z_3	z_5
b	z_5	z_2	z_4	z_4	z_2	z_5

2.1.2 Beispiel Zustandsgraph

Dies ist der Zustandsgraph zu dem DFA aus Beispiel 2.1.1



Die Knoten des Graphen sind die Zustände. Für jedes Paar $(z_i, \sigma) \in Z \times \Sigma$ mit $\delta(z_i, \sigma) = z_j$ geht eine Kante von z_i nach z_j mit der Beschriftung σ aus.

Am Zustandsgraphen lässt sich noch einfacher ablesen, ob ein Wort w Element der zugehörigen Sprache des DFA ist. Vom Startzustand ausgehend, folgen wir dem Graphen entlang der Kanten, die der Reihe nach den Buchstaben des zu prüfenden Wortes entsprechen. Kommen wir dabei in einen Endzustand, so ist das Wort w Element der Sprache.

Notation: Der bzw. die Startzustände werden dargestellt indem ein unbeschrifteter freier Pfeil auf sie verweist. Die Endzustände sind mit breitem Rand markiert.

2.1.3 Beispiel nichtdeterministischer endlicher Automat (NFA)

Der nichtdeterministische Automat erlaubt mehrere Startzustände sowie in einem Schritt mehrere Zustandswechsel. δ wird dabei nun nicht mehr nur auf Z sondern auf die Potenzmenge $\mathcal{P}(Z)$ abgebildet.

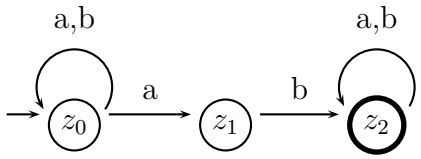
$Z = \{z_0, z_1, z_2, \}$
$\Sigma = \{a, b\}$
$\delta : Z \times \Sigma \to \mathcal{P}(Z)$ mit

δ	z_0	z_1	z_2
a	$\{z_0, z_1\}$	$\varnothing$	$\{z_2\}$
b	$\{z_0\}$	$\{z_2\}$	$\{z_2\}$

$S = \{z_0\}$ Startzustände
$E = \{z_2\}$ Endzustände

2.1.4 Beispiel Zustandsgraph eines NFA

Dies ist der Zustandsgraph zu dem NFA aus Beispiel 2.1.3



Anders als bei einem DFA ist es hier gestattet, dass von einem Zustandsknoten keine oder beliebig viele Pfeile mit der gleichen Beschriftung ausgehen. Im Beispiel NFA gehen von z_0 zwei Pfeile mit Beschrifung a aus. Einer der Pfeile geht auf z_0 selber zurück während der zweite auf z_1 verweist.

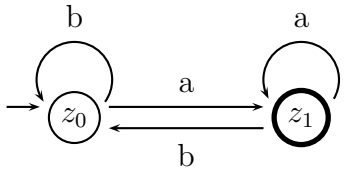
2.1.5 Beispiel Transformation eines DFA in eine Grammatik

Betrachte den DFA
$Z = \{z_0, z_1\}$
$\Sigma = \{a, b\}$
$\delta : Z \times \Sigma \to \mathcal{P}(Z)$ mit

δ	z_0	z_1
a	z_1	z_1
b	z_0	z_0

$S = z_0$ Startzustand
$E = \{z_1\}$ Endzustände

Der Zustandsgraph sieht wie folgt aus.



Der DFA beschreibt die Sprache $L = \{w \in \Sigma^+ | w \text{ endet mit } a\}$.

Gesucht ist nun eine Grammatik die aus dem DFA konstruiert wird und die selbe Sprache beschreibt.

Konstruiere $G = \{V, \Sigma, P, S\}$ wie folgt.
$V = Z$, $\Sigma = \Sigma$
$z_0 =$ Startvariable
P in Regelnotation:

- Für jedes $\delta(z, a_1) = \tilde{z}$ nimm die Regel $z \to a_1\tilde{z}$ auf.

- Für jedes $\delta(z, a_1) = \tilde{z} \in E$ nimm zusätzlich die Regel $z \to a_1$ auf.

erhalte also:

$z_0 \to az_1|bz_0|a$
$z_1 \to az_1|bz_0|a$

Die erhaltene Grammatik beschreibt die selbe Sprache wie der DFA. Jedoch im allgemeinen nicht auf die einfachste Weise, wie dieses Beispiel zeigt. Da die Variablen z_0 und z_1 die gleichen Regeln beschreiben können sie zusammengefasst werden zu einer Regel.

$$z_0 \to az_0|bz_0|a$$

Diese reicht aus um die Grammatik zu beschreiben.

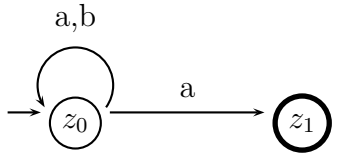
2.1.6 Beispiel Transformation eines NFA in einen DFA

Betrachte den NFA
$Z = \{z_0, z_1\}$, $\Sigma = \{a, b\}$
$\delta : Z \times \Sigma \to Z$ mit

δ	z_0	z_1
a	$\{z_0, z_1\}$	$\varnothing$
b	$\{z_0\}$	$\varnothing$

$S = \{z_0\}$ *Startzustände*
$E = \{z_1\}$ *Endzustände*

Der zugehörige Zustandsgraph sieht wie folgt aus.



Konstruiere einen DFA $M = \{\tilde{Z}, \Sigma, \tilde{\delta}, \tilde{z_0}, \tilde{E}\}$ der die selbe Sprache erkennt.

$\tilde{Z} = \mathcal{P}(Z)$

$\tilde{\delta} : \tilde{Z} \times \Sigma \to \tilde{(}Z)$ mit
$\tilde{\delta}(Q, a_1) = \bigcup_{z \in Q} \delta(z, a_1)$

$\tilde{S} = z_0$ Startzustand

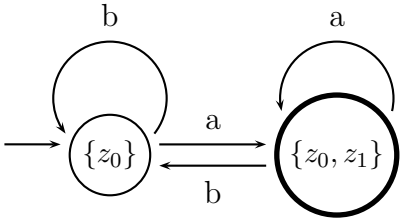
$\tilde{E} = \{Q \subseteq Z | E \cap Q \neq \varnothing\}$ Endzustände

Wir erhalten also:
$\tilde{Z} = \{\varnothing, \{z_0\}, \{z_1\}, \{z_0, z_1\}\}$

$\tilde{\delta}$	$\varnothing$	$\{z_0\}$	$\{z_1\}$	$\{z_0, z_1\}$
a	$\varnothing$	$\{z_0, z_1\}$	$\varnothing$	$\{z_0, z_1\}$
b	$\varnothing$	$\{z_0\}$	$\varnothing$	$\{z_0\}$

$\{z_0\}$ ist Startzustand
$\tilde{E} = \{\{z_1\}, \{z_0, z_1\}\}$

Da die Zustände $\{z_1\}$ und $\varnothing$ vom Startzustand aus nicht zu erreichen sind, können sie weggelassen werden. Wir erhalten für den DFA folgenden Zustandsgraphen:



2.1.7 Beispiel: Transformation einer regulären Grammatik in einen NFA

Betrachte folgende reguläre Grammatik über dem Alphabet $\Sigma = \{a, b, c\}$
$G = \{V, \Sigma, P, S\}$
$V = \{S, T, Y, Z\}$,
$S =$ Startvariable
P in Regelnotation:
$S \to aT$
$T \to aT|bY$
$Y \to aY|bY|cZ|c$
$Z \to aZ|bZ|cZ|a|b|c$

Die Grammatik erzeugt die Sprache $L = \{w \in \Sigma^* |$ a, b und c treten in w in genau dieser Reihenfolge zum ersten mal auf$\}$ Wir suchen nun einen NFA der die selbe Sprache erzeugt. Dieser wird wie folgt konstruiert:

$Z = V \cup \{X\}$
$S = Startzustand$
$E = \{X\}$ da es keine Regel $S \to \epsilon \in P$ gibt, anndernfalls wäre $E = \{X, S\}$
δ wird dabei wie folgt definiert:

- Für jede Regel $A \to aB$ in P nimm B in $\delta(A, a)$ auf.

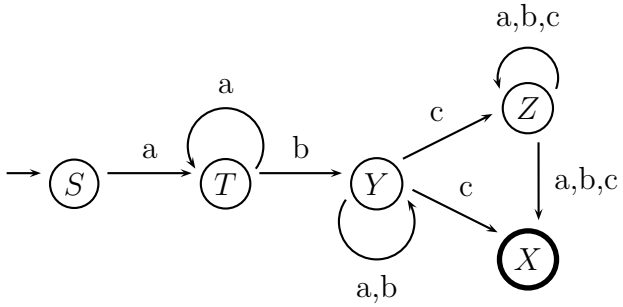
- Für jede Regel $A \to a$ in P nimm X in $\delta(A, a)$ auf.

So erhalten wir
$\delta(S, a) = \{T\}, \delta(S, b) = \varnothing, \delta(S, c) = \varnothing$
$\delta(T, a) = \{T\}, \delta(T, b) = \{Y\}, \delta(T, c) = \varnothing$
$\delta(Y, a) = \{Y\}, \delta(Y, b) = \{Y\}, \delta(Y, c) = \{Z, X\}$
$\delta(Z, a) = \{Z, X\}, \delta(Z, b) = \{Z, X\}, \delta(Z, c) = \{Z, X\}$

In Tabellennotation:

δ	S	T	Y	Z	X
a	$\{T\}$	$\{T\}$	$\{Y\}$	$\{Z, X\}$	$\varnothing$
b	$\varnothing$	$\{Y\}$	$\{Y\}$	$\{Z, X\}$	$\varnothing$
c	$\varnothing$	$\varnothing$	$\{Z, X\}$	$\{Z, X\}$	$\varnothing$

Der zugehörige Zustandsgraph sieht wie folgt aus.



Dabei ist X Endzustand und die Sprache die von dem NFA erzeugt wird ist wieder $L = \{w \in \Sigma^* |\ a, b \text{ und } c \text{ treten in } w \text{ in genau dieser Reihenfolge zum ersten mal auf}\}$.

2.2 Reguläre Ausdrücke

2.2.1 Beispiel: Reguläre Ausdrücke

Sei $\Sigma = \{a, b, c\}$ ein Alphabet. Es folgen einige Beispiele von Sprachen die einfach durch reguläre Ausdrücke dargestelt werden können.

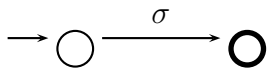
1. $\{w \in \Sigma^* |\ w \text{ beginnt mit } a \text{ und endet mit } b\} = a(a|b|c)^*b$
2. $\{w \in \Sigma^* |\ |w|_a = n\} = ((b|c)^*a)^n(b|c)^*$
3. $\{w \in \Sigma^* |\ |w| = n \text{ und } w_n = a\} = (a|b|c)^{n-1}a$
4. $\{w \in \Sigma^* |\ a, b \text{ und } c \text{ treten in } w \text{ in genau dieser Reihenfolge zum ersten mal auf}\} = aa^*b(a|b)^*c(a|b|c)^*$

2.2.2 Beispiel: vom regulären Ausdruck zum NFA

Zu jedem regulären Ausdruck α, lässt sich ein NFA konstruieren der die Sprache $L(\alpha)$ erzeugt. Wir wollen mit Hilfe von Zustandsgraphen veranschaulichen wie mittels der verschiedenen Synthesen der NFA konstruiert wird. Betrachten wir den Ausdruck

$$\alpha = ((a|c)(ab|ba)^*|b^*(cb|ac)) \tag{1}$$

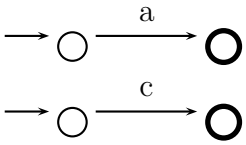
Auf unterster Ebene dieses Ausdrucks finden sich die Terminalzeichen. $L(a) = \{a\}, L(b) = \{b\}, L(c) = \{c\}$. Sie werden erzeugt durch



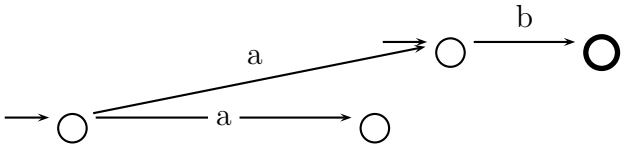
wobei σ für den entsprechenden Buchstaben steht.

Wir zerlegen den Ausdruck in Teile und erstellen im folgenden die einzelnen Zustandsgrapen
$\alpha_1 = (a|c)$
$\alpha_2 = ab$
$\alpha_3 = ba$
$\alpha_4 = (\alpha_2|\alpha_3) = (ab|ba)$
$\alpha_5 = \alpha_4^* = (ab|ba)^*$
$\alpha_6 = \alpha_1 \cdot \alpha_5 = (a|c) \cdot (ab|ba)^*$
$\alpha_7 = b^*$
$\alpha_8 = cb$
$\alpha_9 = ac$
$\alpha_{10} = (\alpha_8|\alpha_9) = (cb|ac)$
$\alpha_{11} = \alpha_7 \cdot \alpha_{10} = b^* \cdot (cb|ac)$ $\alpha = (\alpha_6|\alpha_{11}) = ((a|c)(ab|ba)^*|b^*(cb|ac))$

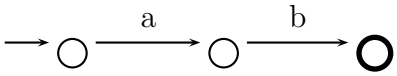
- $\alpha_1 = (a|c)$



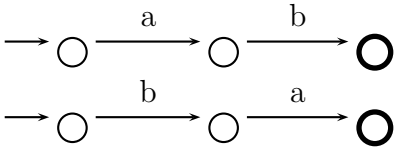
- α_2 und analog $\alpha_3, \alpha_8, \alpha_9$:



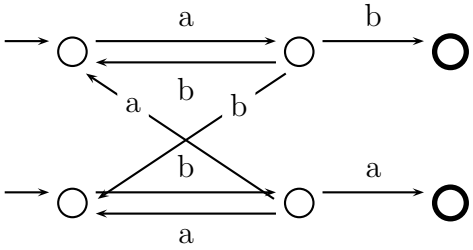
  Der frühere Endzustand wird nun zu einer Sackgasse, daher kann er weggelassen werden, ohne die Sprache zu verändern.



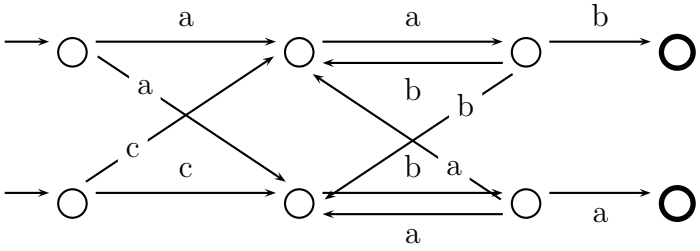
- $\alpha_4 = (ab|ba)$



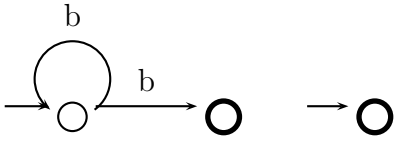
- $\alpha_5 = \alpha_4^*$



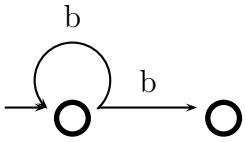
- $\alpha_6 = \alpha_1 \cdot \alpha_5$



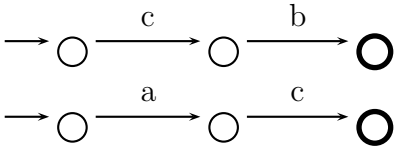
- $\alpha_7 = b^*$:



Die Funktion des zusätzlich zugefügten Knotens ist es zu garantieren, dass auch ϵ erzeugt werden kann. Wir können den Graphen hier vereinfachen indem wir diesen Knoten wieder weglassen, dafür aber den Startknoten in die Menge der Endzustände aufnehmen.

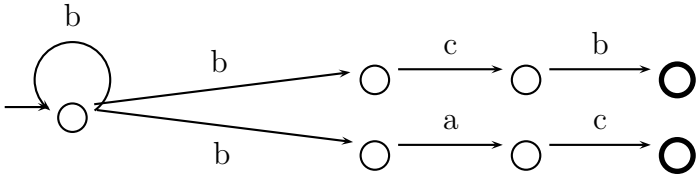


- $\alpha_{10} = (\alpha_8|\alpha_9) = (cb|ac)$:



- $\alpha_{11} = \alpha_7 \cdot \alpha_{10}$:

  Nach entfernen einer Sackgasse erhalten wir



Den NFA zu $L(\alpha) = L((a|c)(ab|ba)^*|b * (cb|ac))$ erhalten wir schließlich indem wir die Zustandsgraphen zu α_6 und α_{11} vereinigen.

2.3 Das Pumping-Lemma

Sei ein DFA M mit vier Zuständen gegeben.
$\Sigma = \{0, 1\}$
$Z = \{z_0, z_1, z_2, z_3\}$
$S = z_0$
$E = \{z_2\}$

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

Da Sprachen die durch einen DFA erzeugt werden regulär sind, lässt sich das Pumping Lemma auf sie anwenden. Es besagt, dass man für alle Wörter $x \in T(M)$ der Länge größer als n eine Zerlegung $x = uvw$ findet, sodass auch $uv^iw \in T(M)$ liegt.

Betrachten wir z.B. das Wort $x = 0100 \in T(M)$ und suchen nun eine Zerlegung in $x = uvw$. Dazu betrachten wir die Zustandsfolge die x erzeugt.

$$z_0 \overset{0}{\to} z_1 \overset{1}{\to} z_0 \overset{0}{\to} z_1 \overset{0}{\to} z_2$$

Da es nur vier Zustände gibt, muss mindestens einer von ihnen in der Zustandsfolge doppelt vorkommen. Hier sind das z_0 und z_1. Wir erhalten somit einen Zyklus im Zustandsgraphen, der natürlich auch iteriert durchlaufen werden kann. Zerlegen wir das Wort z.B. anhand des Zykluses der von z_1 nach z_1 führt so erhalten wir $u = 0, v = 10, w = 0$.

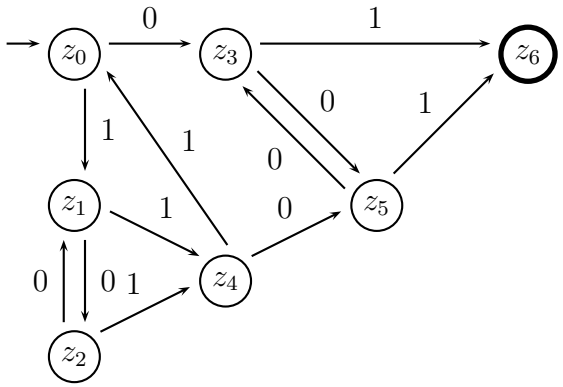
Es gilt nun, dass $uv^i w = 0(10)^i 0 \in L$ für alle $i \geq 0$.

2.4 Minimierung eines DFA

Folgender DFA über dem Alphabet $\Sigma = \{0, 1\}$ soll minimiert werden.
$Z = \{z_0, z_1, z_2, z_3, z_4, z_5, z_6, z_7\}$
$S = z_0$
$E = \{z_6\}$

δ	z_0	z_1	z_2	z_3	z_4	z_5	z_6	z_7
0	z_3	z_2	z_1	z_5	z_5	z_3	z_7	z_7
1	z_1	z_4	z_4	z_6	z_0	z_6	z_7	z_7

Im Zustandsgraphen wurde die Sackgasse z_7 der Übersicht halber weggelassen.



Um den Minimalautomaten zu erstellen suchen wir die Äquivalenzklassen der Zustände.

Schritt 1 Markiere alle Paare (z_i, z_j) ,mit $z_i \in E$ und $z_j \notin E$.

Schritt 2 Markiere alle Paare $(z_i, z_j), z_i \neq z_j$ für die es ein markiertes Paar $(\tilde{z}_i, \tilde{z}_j)$ und ein $a \in \Sigma$ gibt mit $\delta(z_i, a) = \tilde{z}_i$ und $\delta(z_j, a) = \tilde{z}_j$.

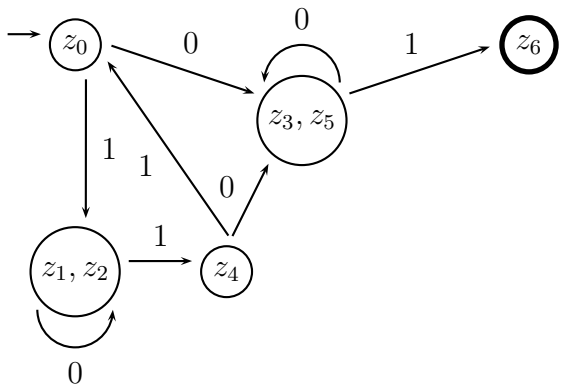
Schritt 3,4,5 Gehe vor wie in Schritt 2 bis keine weiteren Knoten mehr markiert werden.

	z_0	z_1	z_2	z_3	z_4	z_5	z_6
z_0							
z_1	$*^3$						
z_2	$*^3$	-					
z_3	$*^2$	$*^2$	$*^2$				
z_4	$*^3$	$*^3$	$*^4$	$*^2$			
z_5	$*^2$	$*^2$	$*^2$	-	$*^2$		
z_6	$*^1$	$*^1$	$*^1$	$*^1$	$*^1$	$*^1$	

Notation: Der Exponent gibt an in welchem Schritt der Knoten markiert wurde.

Alle nicht markiertern Paare sind äquivalent und können zu einem Knoten zusammengefasst werden. Es werden also z_1 und z_2 zu einem Knoten zusammengefasst und z_3 und z_5. Dabei werden alle Kanten die vorher auf einen der beiden Knoten gerichtet waren nun auf den zusammengefassten Knoten gerichtet. Alle Kanten die von einem der beiden Knoten ausgingen gehen nun vom zusammengefassten Knoten aus, dabei werden Kanten mit gleicher Beschriftung die auf den selben Knoten führen zusammengefasst.

Der resultierende Zustandsgraph sieht wie folgt aus.



2.5 Produktautomat

Gegeben seien zwei DFA $M_1 = \{Z_1, \delta_1, S_1, E_1\}$ und $M_2 = \{Z_2, \delta_2, S_2, E_2\}$ wie folgt.

$\Sigma = \{0, 1\}$ $Z_1 = \{z_0, z_1, z_2\}$
$S_1 = z_0$
$E_1 = \{z_1\}$

δ_1	z_0	z_1
0	z_1	z_1
1	z_0	z_0

$L_1 := T(M_1) = (0|1)^*0 =$ alle Wörter die auf 0 enden.

$\Sigma = \{0, 1\}$ $Z_2 = \{t_0, t_1, t_2\}$
$S_2 = t_0$
$E_2 = \{t_2\}$

δ_2	t_0	t_1	t_2
0	t_2	t_1	t_1
1	t_0	t_1	t_1

$L_2 := T(M_2) = 1^*0 =$ alle Wörter aus beliebig vielen Einsen gefolgt von einer Null.

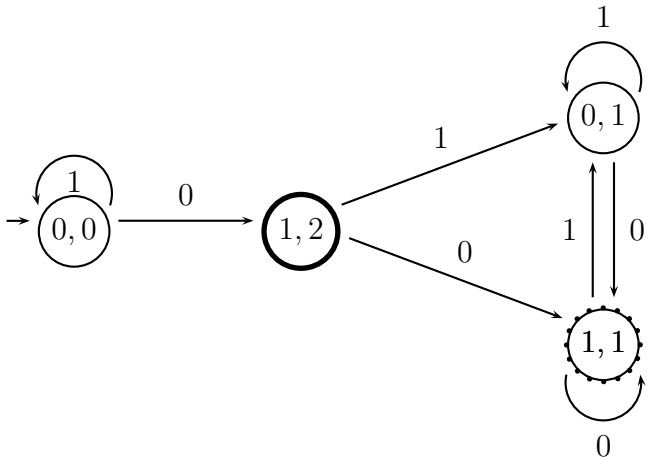
Es soll nun der Produktautomat $M = \{Z, \delta, S, E\}$ bestimmt werden. $Z = Z_1 \times Z_2$
$\delta((z_i, t_j), a) = (\delta_1(z_i, a), \delta_2(t_j, a))$

δ	(z_0,t_0)	(z_0,t_1)	(z_0,t_2)	(z_1,t_0)	(z_1,t_1)	(z_1,t_2)
0	(z_1,t_2)	(z_1,t_1)	(z_1,t_1)	(z_1,t_2)	(z_1,t_1)	(z_1,t_1)
1	(z_0,t_0)	(z_0,t_1)	(z_0,t_1)	(z_0,t_0)	(z_0,t_1)	(z_0,t_1)

$S = S_1 \times S_2 = (z_0, t_0)$

Davon wie wir E wählen hängt nun ab für welche Sprache wir den Produktautomaten erhalten wollen.
Um $T(M_1) \cap T(M_2)$ zu erhalten wähle $E = E_1 \times E_2 = (z_1, t_2)$.
Um $T(M_1) \cup T(M_2)$ zu erhalten wähle $E = (E_1 \times Z_2) \cup (Z_1 \times E_2) = \{(z_1,t_0), (z_1,t_1), (z_1,t_2), (z_0,t_2)\}$.

Wir betrachten unser Ergebnis.
Die Zustände (z_1, t_0) und (z_0, t_2) können weggelassen werden, da man durch keinen anderen Zustand auf sie gelangen kann. Wir erhalten folgenden Zustandsgraphen.



Notation: gepunktet dargestellte Knoten sind nur im Fall $T(M_1) \cup T(M_2)$ auch Endzustände.

Ist $E = (z_1, t_2)$ so erhalten wir die Sprache $L_2 = 1^*0 = L_1 \cap L_2$.
Ist $E = \{(z_1,t_1), (z_1,t_2)\}$ so erhalten wir die Sprache $(0|1)^*0 = L_1 = L_1 \cup L_2$.