Theorie der Informatik

9. Reguläre Sprachen II

Malte Helmert Gabriele Röger

Universität Basel

23. März 2015

Pumping Lemma

Pumping Lemma: Motivation

Man kann zeigen, dass eine Sprache regulär ist, indem man eine geeignete Grammatik, endlichen Automaten oder regulären Ausdruck angibt. Wie kann man zeigen, dass eine Sprache nicht regulär ist?

Foto mit freundlicher Genehmigung von imagerymajestic / FreeDigitalPhotos.net

Pumping Lemma: Motivation

Man kann zeigen, dass eine Sprache regulär ist, indem man eine geeignete Grammatik, endlichen Automaten oder regulären Ausdruck angibt. Wie kann man zeigen, dass eine Sprache nicht regulär ist?

- Direkter Beweis, dass keine reguläre Grammatik existiert, die die Sprache erzeugt
  $\rightsquigarrow$ i.A. schwierig

Foto mit freundlicher Genehmigung von imagerymajestic / FreeDigitalPhotos.net

Pumping Lemma: Motivation

Man kann zeigen, dass eine Sprache regulär ist, indem man eine geeignete Grammatik, endlichen Automaten oder regulären Ausdruck angibt. Wie kann man zeigen, dass eine Sprache nicht regulär ist?

- Direkter Beweis, dass keine reguläre Grammatik existiert, die die Sprache erzeugt
  $\rightsquigarrow$ i.A. schwierig
- Pumping Lemma: Notwendige Eigenschaft, die für alle regulären Sprachen gelten muss.

Foto mit freundlicher Genehmigung von imagerymajestic / FreeDigitalPhotos.net

Pumping Lemma

Satz (Pumping Lemma)

Sei L eine reguläre Sprache. Dann gibt es ein $n \in \mathbb{N}$ (eine Pumpingzahl für L), so dass sich alle Wörter $x \in L$ mit $|x| \geq n$ zerlegen lassen in $x = uvw$, so dass folgendes gilt:

1. $|v| \geq 1$,
2. $|uv| \leq n$, *und*
3. *für alle* $i = 0, 1, 2, \ldots$ *gilt:* $uv^i w \in L$.

Pumping Lemma: Beweis

Satz (Pumping Lemma)

Sei L eine reguläre Sprache. Dann gibt es ein $n \in \mathbb{N}$, so dass sich alle Wörter $x \in L$ mit $|x| \geq n$ zerlegen lassen in $x = uvw$, so dass:

1. $|v| \geq 1$,
2. $|uv| \leq n$, *und*
3. *für alle* $i = 0, 1, 2, \ldots$ *gilt:* $uv^i w \in L$.

Pumping Lemma: Beweis

Satz (Pumping Lemma)

Sei L eine reguläre Sprache. Dann gibt es ein $n \in \mathbb{N}$, so dass sich alle Wörter $x \in L$ mit $|x| \geq n$ zerlegen lassen in $x = uvw$, so dass:

1. $|v| \geq 1$,
2. $|uv| \leq n$, *und*
3. *für alle* $i = 0, 1, 2, \ldots$ *gilt:* $uv^i w \in L$.

Beweis.

Für reguläres L existiert DFA $M = (Z, \Sigma, \delta, z_0, E)$ mit $\mathcal{L}(M) = L$.
Wir zeigen, dass $n = |Z|$ die geforderten Eigenschaften hat.

...

Pumping Lemma: Beweis

Satz (Pumping Lemma)

Sei L eine reguläre Sprache. Dann gibt es ein $n \in \mathbb{N}$, so dass sich alle Wörter $x \in L$ mit $|x| \geq n$ zerlegen lassen in $x = uvw$, so dass:

1. $|v| \geq 1$,
2. $|uv| \leq n$, *und*
3. *für alle* $i = 0, 1, 2, \ldots$ *gilt:* $uv^i w \in L$.

Beweis.

Für reguläres L existiert DFA $M = (Z, \Sigma, \delta, z_0, E)$ mit $\mathcal{L}(M) = L$. Wir zeigen, dass $n = |Z|$ die geforderten Eigenschaften hat.

Betrachte beliebiges $x \in \mathcal{L}(M)$ mit Länge $|x| \geq |Z|$. Inklusive dem Startzustand durchläuft M beim Abarbeiten von x insgesamt $|x| + 1$ Zustände. Wegen $|x| \geq |Z|$ muss daher mindestens ein Zustand mehrfach durchlaufen werden. ...

Pumping Lemma: Beweis

Satz (Pumping Lemma)

Sei L eine reguläre Sprache. Dann gibt es ein $n \in \mathbb{N}$, so dass sich alle Wörter $x \in L$ mit $|x| \geq n$ zerlegen lassen in $x = uvw$, so dass:

1. $|v| \geq 1$,
2. $|uv| \leq n$, *und*
3. *für alle* $i = 0, 1, 2, \ldots$ *gilt:* $uv^i w \in L$.

Beweis (Fortsetzung).

Wähle Zerlegung $x = uvw$, so dass M nach Lesen von u und Lesen von uv im gleichen Zustand ist. Offensichtlich können wir die Zerlegung so wählen, dass $|v| \geq 1$ und $|uv| \leq |Z|$ erfüllt sind. ...

Pumping Lemma: Beweis

Satz (Pumping Lemma)

Sei L eine reguläre Sprache. Dann gibt es ein $n \in \mathbb{N}$, so dass sich alle Wörter $x \in L$ mit $|x| \geq n$ zerlegen lassen in $x = uvw$, so dass:

1. $|v| \geq 1$,
2. $|uv| \leq n$, *und*
3. *für alle* $i = 0, 1, 2, \ldots$ *gilt:* $uv^i w \in L$.

Beweis (Fortsetzung).

Das Teilwort v entspricht nach Abarbeitung von u einer Schleife in dem DFA und kann daher beliebig oft durchlaufen werden. Jede anschliessende Fortsetzung mit w endet im selben Endzustand wie die Abarbeitung von x. Daher gilt für alle $i = 0, 1, 2, \ldots$, dass $uv^i w \in \mathcal{L}(M) = L$. □

Pumping Lemma: Verwendung

Verwendung des Pumping Lemma (PL):

Beweis der Nichtregularität

- Wenn L regulär ist, dann gilt PL für L
- Wenn PL für L gilt, kann man nichts über L aussagen.
- Aber: Wenn PL für L nicht gilt, kann L nicht regulär sein.
- D.h.: Wenn es kein $n \in \mathbb{N}$ mit den geforderten Eigenschaften für PL gibt, dann kann L nicht regulär sein.

Pumping Lemma: Beispiel

Beispiel

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

Beweis.

Angenommen L ist regulär. Dann sei p eine Pumpingzahl für L.

Das Wort $x = a^p b^p$ ist in L und hat Länge $\geq p$. Sei $x = uvw$ eine Zerlegung mit den Eigenschaften aus dem PL.

Dann ist auch das Wort $x' = uv^2w$ in L. Da $|uv| \leq p$ besteht uv nur aus Symbolen a und $x' = a^{|u|}a^{2|v|}a^{p-|uv|}b^p = a^{p+|v|}b^p$.

Da $|v| \geq 1$ gilt $p + |v| \neq p$ und damit $x' \notin L$.

Dies ist ein Widerspruch zum PL. $\rightsquigarrow$ L nicht regulär. $\square$

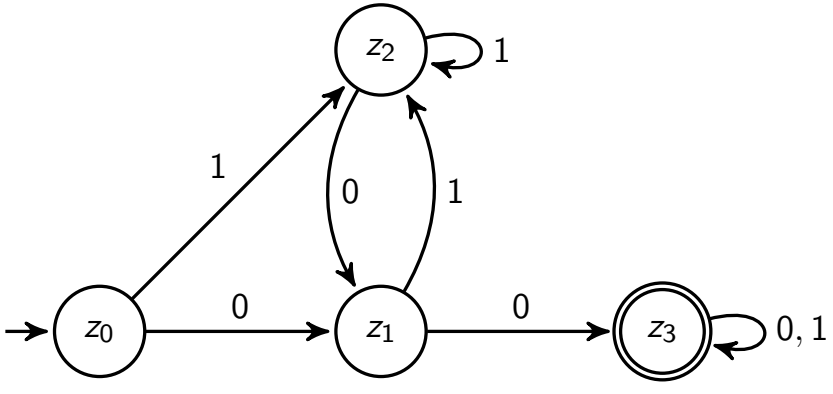
Fragen

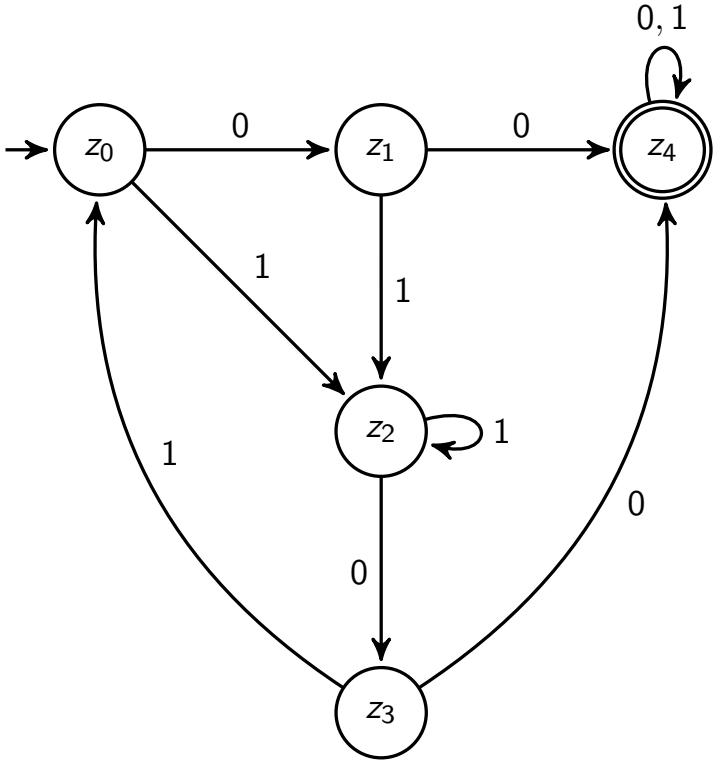
Fragen?

Minimalautomaten

Beispiel

Folgende DFAs akzeptieren die gleiche Sprache:





Frage: Was ist der kleinste DFA, der diese Sprache akzeptiert?

Minimalautomat: Definition

Definition

Ein Minimalautomat für eine reguläre Sprache L ist ein DFA $M = (Z, \Sigma, \delta, z_0, E)$ mit $\mathcal{L}(M) = L$ und einer minimalen Anzahl von Zuständen. Das heisst, es gibt keinen DFA $M' = (Z', \Sigma, \delta', z_0', E')$ mit $\mathcal{L}(M) = \mathcal{L}(M')$ und $|Z'| < |Z|$.

Minimalautomat: Definition

Definition

Ein Minimalautomat für eine reguläre Sprache L ist ein DFA $M = (Z, \Sigma, \delta, z_0, E)$ mit $\mathcal{L}(M) = L$ und einer minimalen Anzahl von Zuständen. Das heisst, es gibt keinen DFA $M' = (Z', \Sigma, \delta', z_0', E')$ mit $\mathcal{L}(M) = \mathcal{L}(M')$ und $|Z'| < |Z|$.

Wie findet man einen Minimalautomaten?

Idee:

- Beginne mit irgendeinem DFA, der die Sprache akzeptiert.
- Verschmelze Zustände, von denen aus die gleichen Wörter in Endzustand führen.

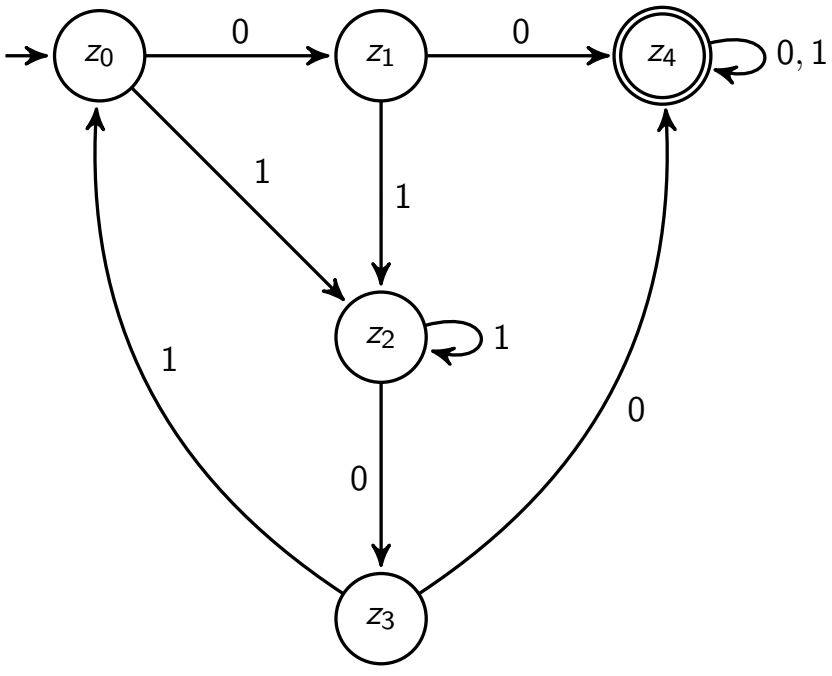
Minimalautomat: Algorithmus

Eingabe: DFA M
(ohne vom Startzustand aus unerreichbare Zustände)

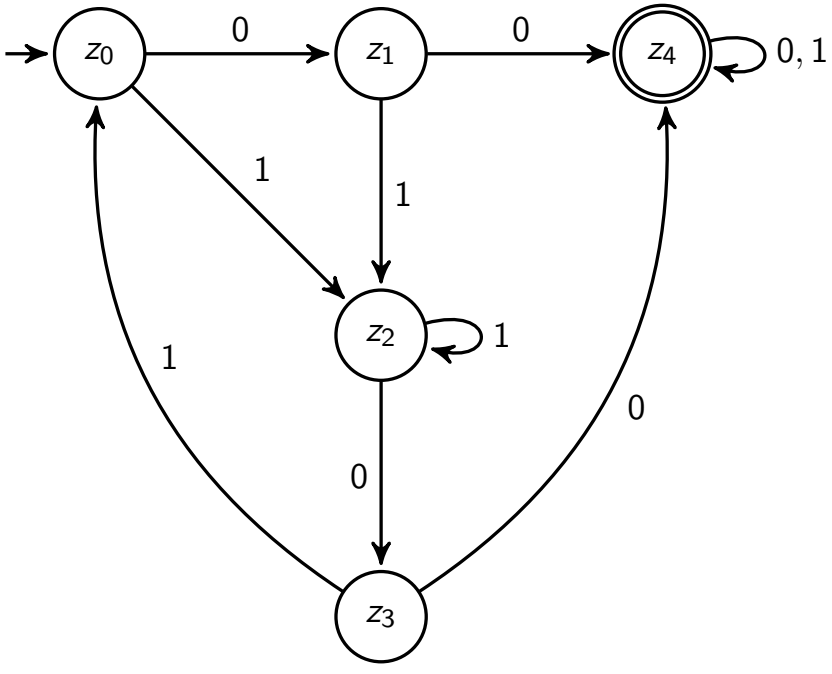
Ausgabe: Angabe, welche Zustände man verschmelzen muss, um äquivalenten Minimalautomaten zu erhalten

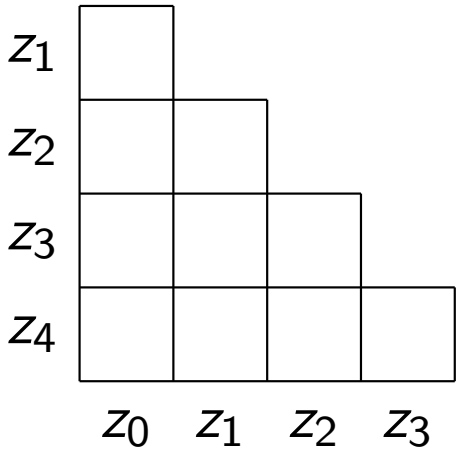
1. Stelle Tabelle aller Zustandspaare $\{z, z'\}$ mit $z \neq z'$ auf.
2. Markiere alle Paare $\{z, z'\}$ mit $z \in E$ und $z' \notin E$.
3. Falls für ein unmarkiertes Paar $\{z, z'\}$ gilt, dass $\{\delta(z, a), \delta(z', a)\}$ für ein $a \in \Sigma$ bereits markiert ist, markiere auch $\{z, z'\}$.
4. Wiederhole den letzten Schritt, bis sich keine Änderung mehr ergibt.
5. Alle jetzt noch unmarkierten Paare können jeweils zu einem Zustand verschmolzen werden.

Minimalautomat: Beispiel

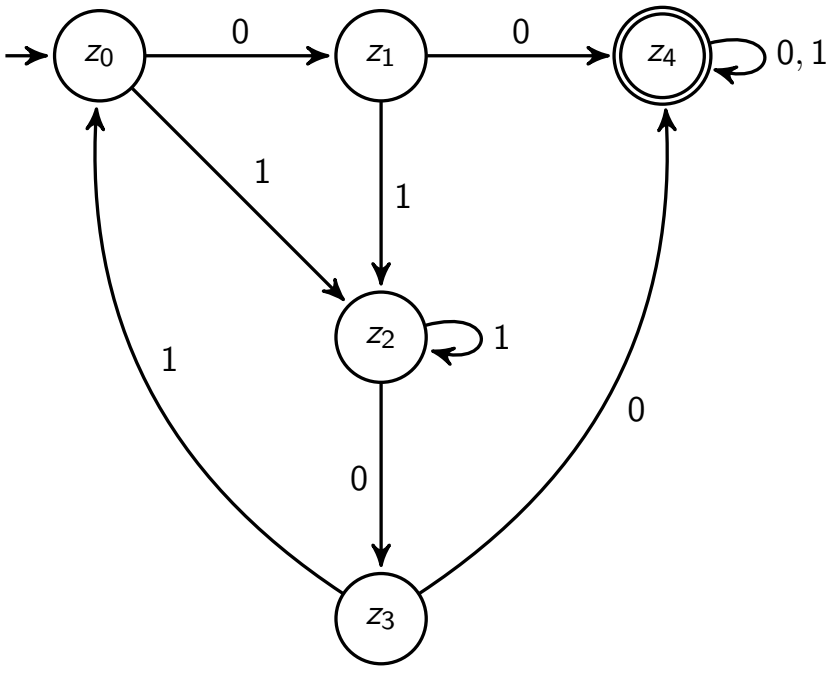


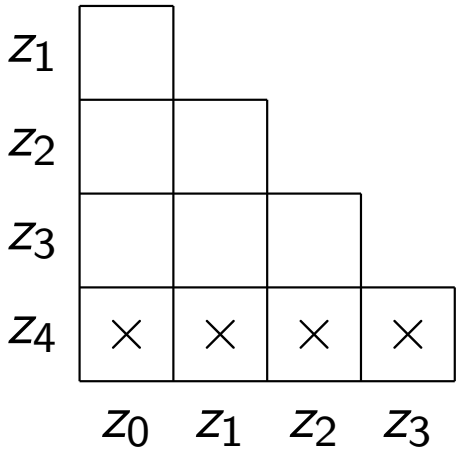
Minimalautomat: Beispiel



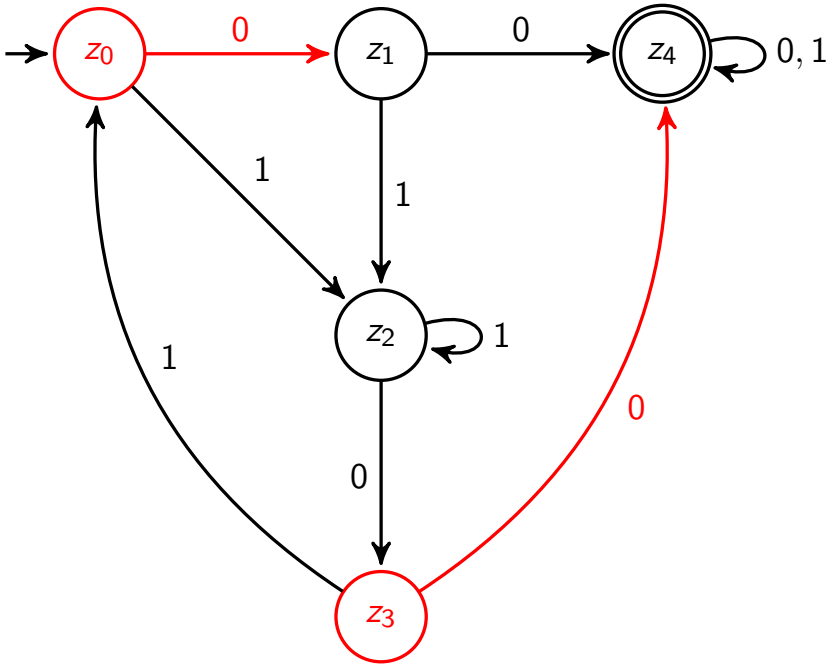


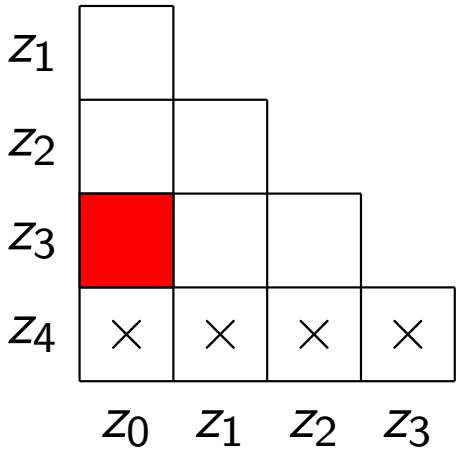
Minimalautomat: Beispiel



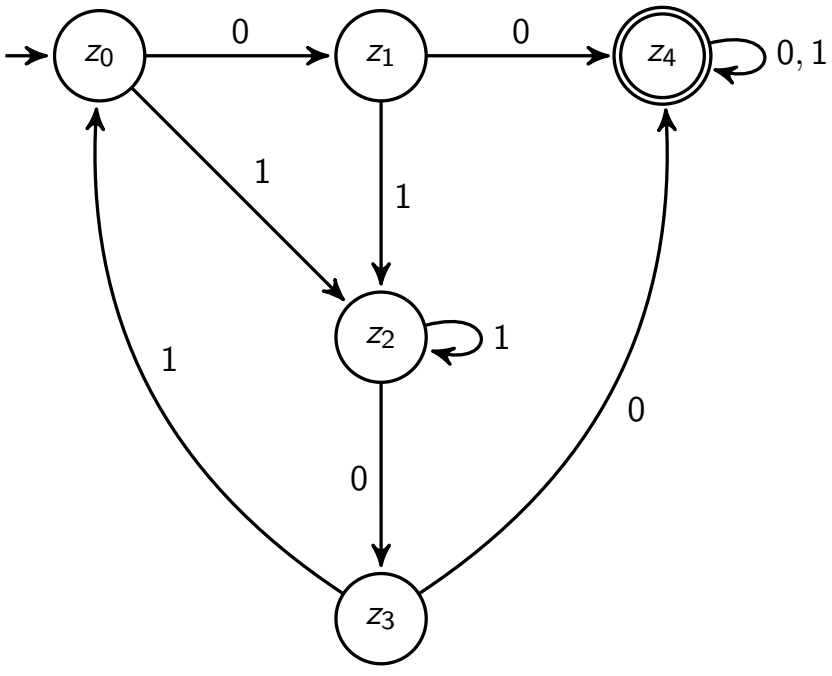


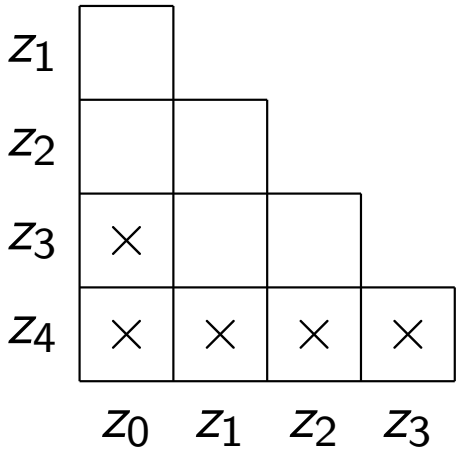
Minimalautomat: Beispiel



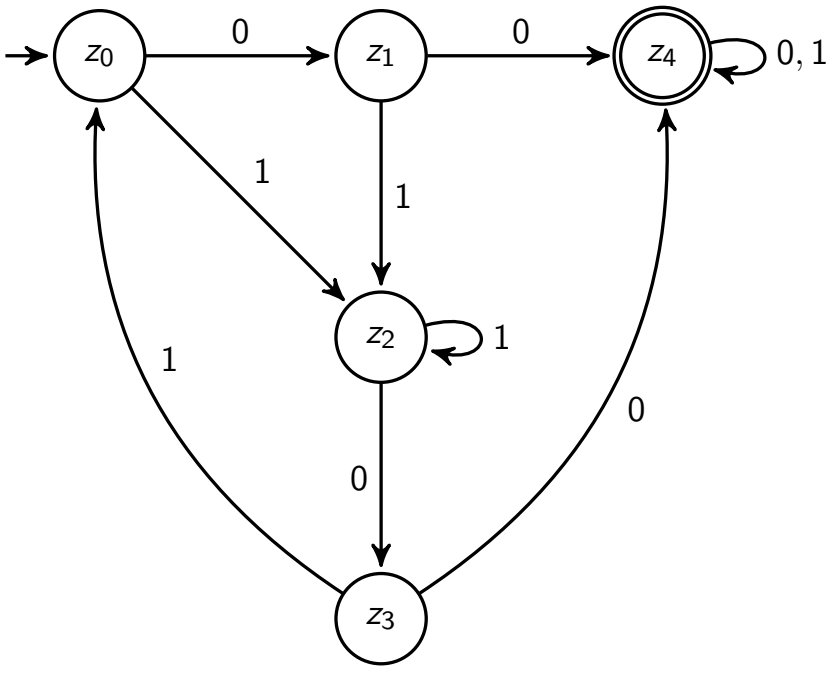


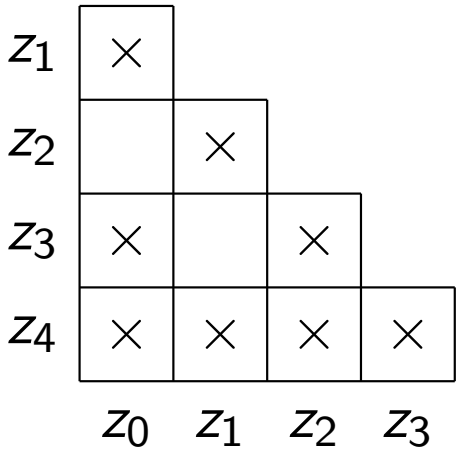
Minimalautomat: Beispiel



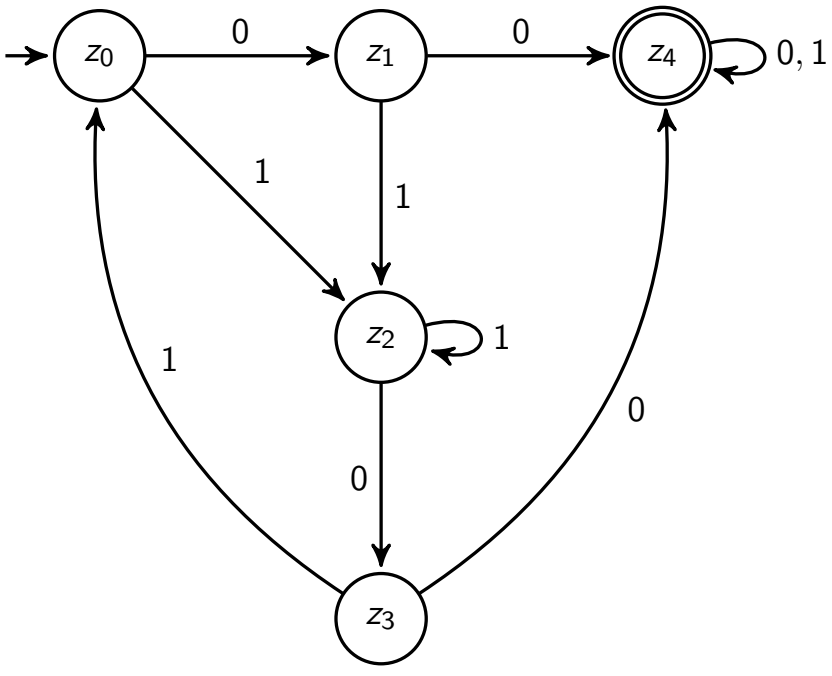


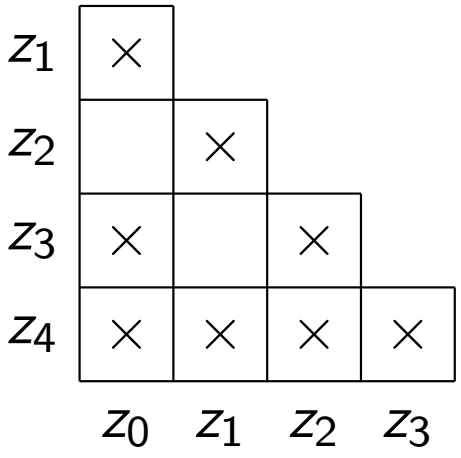
Minimalautomat: Beispiel





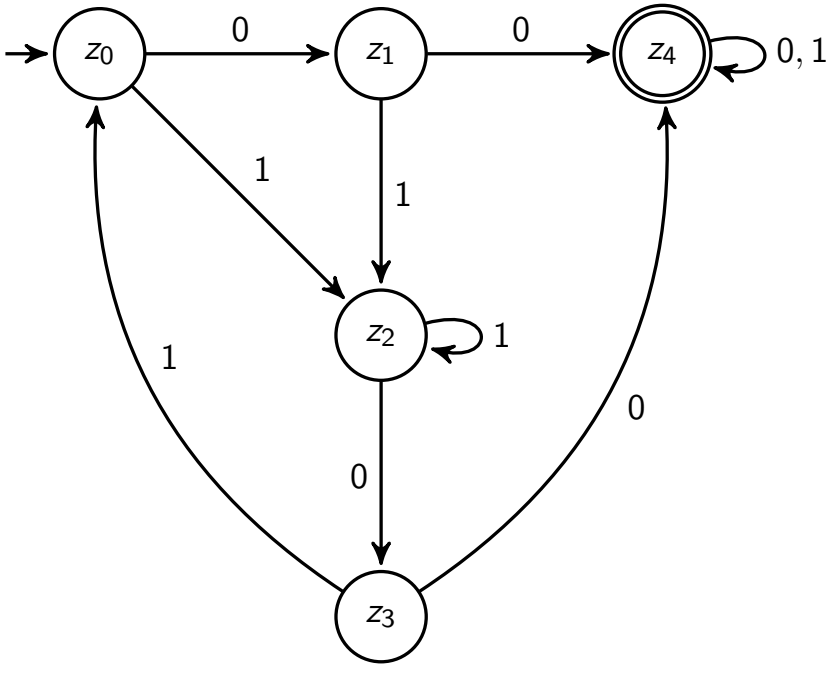
Minimalautomat: Beispiel

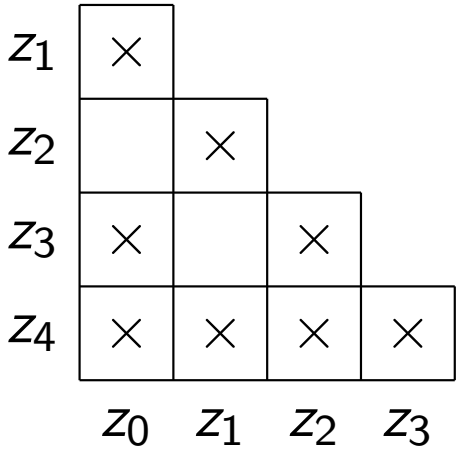




Zustände z_0, z_2 und z_1, z_3 können jeweils zu einem Zustand verschmolzen werden.

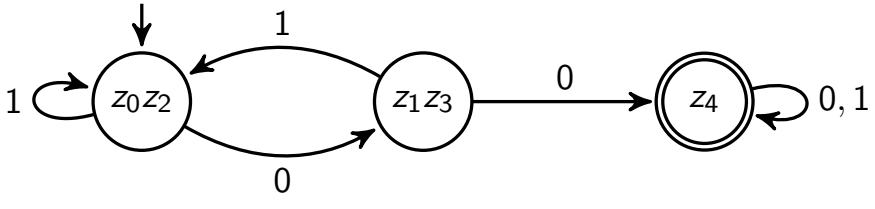
Minimalautomat: Beispiel





Zustände z_0, z_2 und z_1, z_3 können jeweils zu einem Zustand verschmolzen werden.

Ergebnis:



Eindeutigkeit von Minimalautomaten

Satz

Alle Minimalautomaten für eine Sprache L sind bis auf Isomorphie (d.h. Umbenennnen der Zustände) identisch.

Ohne Beweis.

Fragen

Fragen?

Abschlusseigenschaften

Abschlusseigenschaften

Wie kann man
reguläre Sprachen kombinieren,
so dass man wieder eine reguläre
Sprache erhält?

Foto mit freundlicher Genehmigung von stockimages / FreeDigitalPhotos.net

Abschlusseigenschaften: Operationen

Seien L und L' reguläre Sprachen über Σ bzw. Σ'.

Wir betrachten folgende Operationen:

- Vereinigung $L \cup L' = \{w \mid w \in L \text{ oder } w \in L'\}$ über $\Sigma \cup \Sigma'$
- Schnitt $L \cap L' = \{w \mid w \in L \text{ und } w \in L'\}$ über $\Sigma \cap \Sigma'$
- Komplement $\bar{L} = \{w \mid w \notin L\}$ über Σ
- Produkt $LL' = \{uv \mid u \in L \text{ und } v \in L'\}$ über $\Sigma \cup \Sigma'$
  Spezialfall: $L^n = L^{n-1}L$, wobei $L^0 = \{\varepsilon\}$
- Stern $L^* = \bigcup_{k \geq 0} L^k$ über Σ

Abschlusseigenschaften

Definition (Abgeschlossenheit)

Sei $\mathcal{K}$ eine Klasse von Sprachen. Dann ist $\mathcal{K}$ abgeschlossen unter Vereinigung (Schnitt, Komplement, Produkt, Stern), falls aus $L \in \mathcal{K}$ und $L' \in \mathcal{K}$ folgt, dass $L \cup L' \in \mathcal{K}$ (bzw. $L \cap L' \in \mathcal{K}$, $\bar{L} \in \mathcal{K}$, $LL' \in \mathcal{K}$, $L^* \in \mathcal{K}$).

Abschlusseigenschaften

Definition (Abgeschlossenheit)

Sei $\mathcal{K}$ eine Klasse von Sprachen. Dann ist $\mathcal{K}$ abgeschlossen unter Vereinigung (Schnitt, Komplement, Produkt, Stern), falls aus $L \in \mathcal{K}$ und $L' \in \mathcal{K}$ folgt, dass $L \cup L' \in \mathcal{K}$ (bzw. $L \cap L' \in \mathcal{K}$, $\bar{L} \in \mathcal{K}$, $LL' \in \mathcal{K}$, $L^* \in \mathcal{K}$).

Satz

Die regulären Sprachen sind abgeschlossen unter:

- *Vereinigung*
- *Schnitt*
- *Komplement*
- *Produkt*
- *Stern*

Abschlusseigenschaften

Beweis.

Abschluss unter Vereinigung, Produkt und Stern folgt daraus, dass für reguläre Ausdrücke α und β auch $(\alpha|\beta)$, $(\alpha\beta)$ und (α^*) reguläre Ausdrücke sind.

Abschlusseigenschaften

Beweis.

Abschluss unter Vereinigung, Produkt und Stern folgt daraus, dass für reguläre Ausdrücke α und β auch $(\alpha|\beta)$, $(\alpha\beta)$ und (α^*) reguläre Ausdrücke sind.

Komplement: Sei $M = (Z, \Sigma, \delta, z_0, E)$ ein DFA mit $\mathcal{L}(M) = L$. Dann ist $M' = (Z, \Sigma, \delta, z_0, Z \setminus E)$ ein DFA mit $\mathcal{L}(M') = \bar{L}$.
akzeptiert.

Abschlusseigenschaften

Beweis.

Abschluss unter Vereinigung, Produkt und Stern folgt daraus, dass für reguläre Ausdrücke α und β auch $(\alpha|\beta)$, $(\alpha\beta)$ und (α^*) reguläre Ausdrücke sind.

Komplement: Sei $M = (Z, \Sigma, \delta, z_0, E)$ ein DFA mit $\mathcal{L}(M) = L$. Dann ist $M' = (Z, \Sigma, \delta, z_0, Z \setminus E)$ ein DFA mit $\mathcal{L}(M') = \bar{L}$. akzeptiert.

Schnitt: Seien $M_1 = (Z_1, \Sigma_1, \delta_1, z_{01}, E_1)$ und $M_2 = (Z_2, \Sigma_2, \delta_2, z_{02}, E_2)$ DFAs. Der Kreuzproduktautomat

$$M = (Z_1 \times Z_2, \Sigma_1 \cap \Sigma_2, \delta, (z_{01}, z_{02}), E_1 \times E_2)$$

$$\text{mit } \delta((z_1, z_2), a) = (\delta_1(z_1, a), \delta_2(z_2, a))$$

akzeptiert $\mathcal{L}(M) = \mathcal{L}(M_1) \cap \mathcal{L}(M_2)$. □

Fragen

Fragen?

Entscheidbarkeit

Entscheidbarkeit

Definition (Entscheidbarkeit (intuitiv))

Eine Sprache L über Σ heisst entscheidbar, wenn es einen Algorithmus gibt, der bei jeder Eingabe eines Wortes $w \in \Sigma^*$ terminiert und ausgibt, ob $w \in L$ oder nicht.

Probleme als Sprachen

Beispiel (Leerheitsproblem)

Das Leerheitsproblem $\mathrm{P}_\emptyset$ ist folgendes Problem:

Gegeben: Reguläre Grammatik G
Gefragt: Ist $\mathcal{L}(G) = \emptyset$?

Oder alternativ als Sprache formuliert:

$$\mathrm{P}_\emptyset = \{w \mid w \text{ kodiert eine Grammatik } G \text{ mit } \mathcal{L}(G) = \emptyset\}$$

Gegeben-Gefragt-Notation vs. Definition als Menge

Definition (neues Problem P)

Gegeben: Instanz $\mathcal{I}$

Gefragt: Hat $\mathcal{I}$ eine bestimmte Eigenschaft?

Gegeben-Gefragt-Notation vs. Definition als Menge

Definition (neues Problem P)

Gegeben: Instanz $\mathcal{I}$
Gefragt: Hat $\mathcal{I}$ eine bestimmte Eigenschaft?

entspricht der Definition

Definition (neues Problem P)

Das Problem P ist die Sprache
$\mathrm{P} = \{w \mid w \text{ kodiert eine Instanz } \mathcal{I} \text{ mit der gefragten Eigenschaft}\}$.

Entscheidbarkeit: Wortproblem

Satz (Wortproblem ist entscheidbar)

Das Wortproblem $\mathrm{P}_{\in}$ für reguläre Sprachen ist:

Gegeben: *Reguläre Grammatik G mit Terminalalphabet Σ und Wort $w \in \Sigma^*$*

Gefragt: *Ist $w \in \mathcal{L}(G)$?*

Das Wortproblem ist entscheidbar.

Beweis.

Konstruiere einen DFA M mit $\mathcal{L}(M) = \mathcal{L}(G)$ (die Beweise aus dem vorherigen Kapitel beschreiben ein geeignetes Verfahren). Simuliere M auf Eingabe w. Die Simulation endet nach $|w|$ Schritten. Der DFA M ist dann in einem Endzustand gdw. $w \in \mathcal{L}(G)$. □

Entscheidbarkeit: Leerheitsproblem

Satz (Leerheitsproblem ist entscheidbar)

Das Leerheitsproblem $\mathrm{P}_\emptyset$ für reguläre Sprachen ist:

Gegeben: *Reguläre Grammatik G*
Gefragt: *Ist $\mathcal{L}(G) = \emptyset$?*

Das Leerheitsproblem ist entscheidbar.

Beweis.

Konstruiere einen DFA M mit $\mathcal{L}(M) = \mathcal{L}(G)$. Im Zustandsdiagramm von M gibt es keinen Pfad vom Startzustand zu einem Endzustand gdw. $\mathcal{L}(G) = \emptyset$. □

Entscheidbarkeit: Endlichkeitsproblem

Satz (Endlichkeitsproblem ist entscheidbar)

Das Endlichkeitsproblem P_∞ für reguläre Sprachen ist:

Gegeben: *Reguläre Grammatik G*
Gefragt: *Ist $|\mathcal{L}(G)| < \infty$?*

Das Endlichkeitsproblem ist entscheidbar.

Beweis.

Konstruiere einen DFA M mit $\mathcal{L}(M) = \mathcal{L}(G)$. Im Zustandsdiagramm von M gibt es einen Zyklus, der vom Startzustand aus erreichbar ist und von dem aus ein Endzustand erreichbar ist, gdw. $|\mathcal{L}(G)| = \infty$. □

Entscheidbarkeit: Schnittproblem

Satz (Schnittproblem ist entscheidbar)

Das Schnittproblem $\mathrm{P}_\cap$ für reguläre Sprachen ist:

Gegeben: *Reguläre Grammatiken G und G'*
Gefragt: *Ist $\mathcal{L}(G) \cap \mathcal{L}(G') = \emptyset$?*

Das Schnittproblem ist entscheidbar.

Beweis.

Da die regulären Sprachen unter Schnitt abgeschlossen sind, konstruieren wir (z.B. über den Zwischenschritt eines Kreuzproduktautomaten) eine Grammatik G'' mit $\mathcal{L}(G'') = \mathcal{L}(G) \cap \mathcal{L}(G')$ und verwenden das Entscheidungsverfahren für das Leerheitsproblem $\mathrm{P}_\emptyset$. □

Entscheidbarkeit: Äquivalenzproblem

Satz (Äquivalenzproblem ist entscheidbar)

Das Äquivalenzproblem $\mathrm{P}_=$ für reguläre Sprachen ist:

Gegeben: *Reguläre Grammatiken G und G'*
Gefragt: *Ist $\mathcal{L}(G) = \mathcal{L}(G')$?*

Das Äquivalenzproblem ist entscheidbar.

Beweis.

Es gilt allgemein für Sprachen L und L', dass

$$L = L' \text{ gdw. } (L \cap \bar{L}') \cup (\bar{L} \cap L') = \emptyset.$$

Die regulären Sprachen sind unter Schnitt, Vereinigung und Komplement abgeschlossen. Wir können eine entsprechende Grammatik konstruieren und das Entscheidungsverfahren für das Leerheitsproblem $\mathrm{P}_\emptyset$ verwenden. □

Fragen

Fragen?

Zusammenfassung

Zusammenfassung

- Mit dem Pumping Lemma kann man zeigen, dass eine Sprache nicht regulär ist.
- Minimalautomaten sind kleinste mögliche DFAs für eine Sprache und für jede Sprache eindeutig.
- Die regulären Sprachen sind unter allen gängigen Operationen abgeschlossen.
- Die gängigen Probleme sind für die regulären Sprachen entscheidbar.