

Formale Systeme

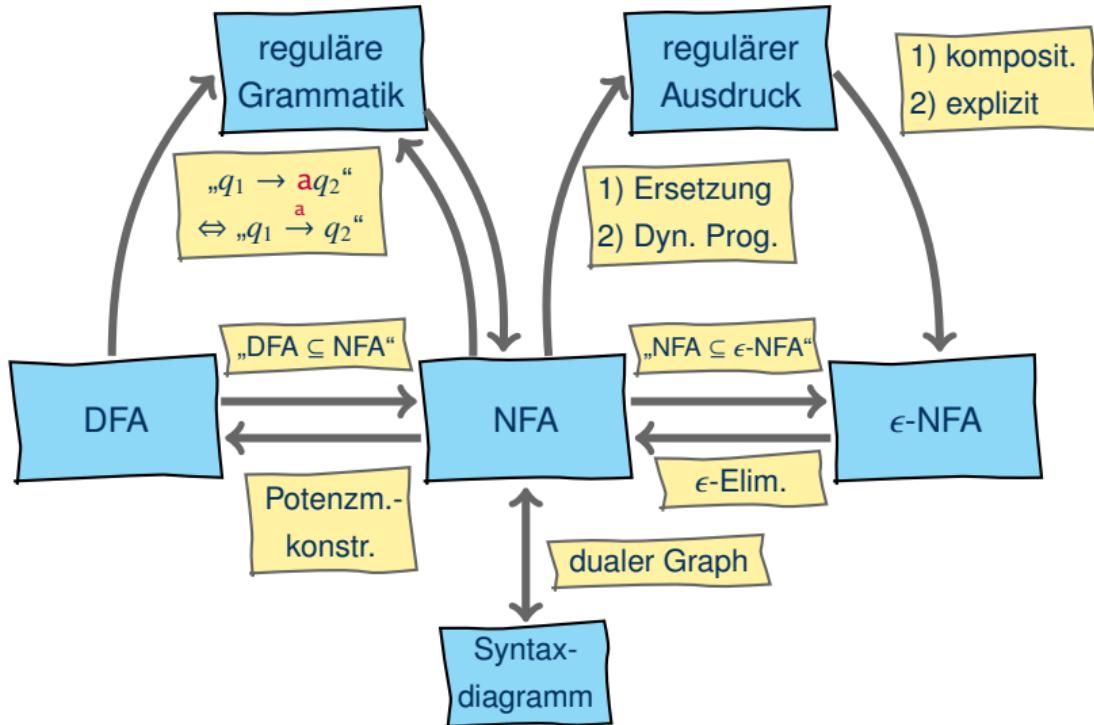
8. Vorlesung: Minimale Automaten

Markus Krötzsch

Professur für Wissensbasierte Systeme

TU Dresden, 10. November 2025

Darstellungen von Typ-3-Sprachen



Minimierung von Automaten

Automaten verkleinern

Wir haben bereits Methoden kennengelernt, um Automaten zu vereinfachen:

- Entfernen von Zuständen, die von keinem Anfangszustand aus erreichbar sind
- Entfernen von Zuständen, von denen aus kein Endzustand erreicht werden kann

Erhalten wir damit den kleinstmöglichen äquivalenten Automaten?

Nein – ein einfaches Gegenbeispiel:

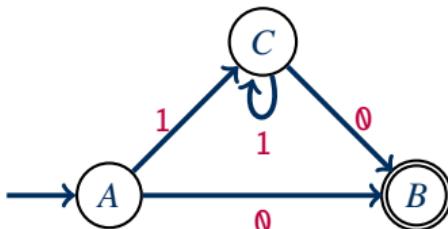
Beispiel: Sei \mathcal{M} ein endlicher Automat, bei dem alle Zustände erreichbar sind und einen Endzustand erreichen können. Der Vereinigungsautomat^a $\mathcal{M} \oplus \mathcal{M}$ akzeptiert die selbe Sprache, hat nur erreichbare Zustände, aber die doppelte Zustandszahl.

^aHierbei müssen die Zustände einer Kopie von \mathcal{M} umbenannt werden.

Ein interessanteres Beispiel

Der Vereinigungsautomat ist immer ein NFA. Nichtdeterminismus macht es einfach, nichtminimale Automaten zu finden.

Interessanter sind nichtminimale DFAs:



Dieser DFA hat keine offensichtlich überflüssigen Zustände, aber der folgende kleinere DFA erkennt die gleiche Sprache 1^*0 :



Automaten minimieren?

Wie kann man Automaten weiter minimieren?

Beobachtungen:

- Zur Erkennung von Wörtern muss der Automat nur seinen aktuellen Zustand kennen
- Wichtig ist, wohin man vom aktuellen Zustand aus gelangt, wenn man das restliche Wort einliest
- Es ist nicht relevant, auf welchem Weg man zu diesem Zustand gelangt ist

Idee: Zwei Zustände sind gleichwertig, wenn man ausgehend von beiden Zuständen die selbe Sprache akzeptieren kann

~ Gleichwertige Zustände könnten verschmolzen werden ...

Äquivalenz von Zuständen

Für einen DFA $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, F \rangle$ und einen Zustand $q \in Q$ sei $\mathcal{M}_q = \langle Q, \Sigma, \delta, q, F \rangle$ der abgewandelte DFA mit Startzustand q .

Zwei Zustände $p, q \in Q$ sind \mathcal{M} -äquivalent, in Symbolen $p \sim_{\mathcal{M}} q$, wenn gilt:

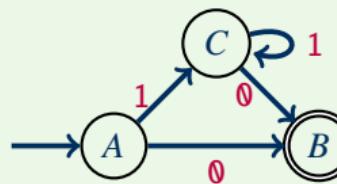
$$\mathbf{L}(\mathcal{M}_p) = \mathbf{L}(\mathcal{M}_q)$$

das heißt wenn für jedes Wort $w \in \Sigma^*$ gilt:

$$\delta(p, w) \in F \text{ genau dann wenn } \delta(q, w) \in F.$$

Wenn der Automat \mathcal{M} klar ist, schreiben wir einfach \sim statt $\sim_{\mathcal{M}}$.

Beispiel:



$$\mathbf{L}(\mathcal{M}_A) = \{1\}^* \{0\} = \mathbf{L}(\mathcal{M}_C)$$

$$\mathbf{L}(\mathcal{M}_B) = \{\epsilon\}$$

$$\leadsto A \sim C$$

Eigenschaften von \sim_M

Definition (kurz): $q \sim_M p$ genau dann wenn $L(M_p) = L(M_q)$.

Damit sehen wir leicht:

- \sim ist reflexiv: $q \sim q$
- \sim ist symmetrisch: wenn $q_1 \sim q_2$ dann $q_2 \sim q_1$
- \sim ist transitiv: wenn $q_1 \sim q_2$ und $q_2 \sim q_3$ dann $q_1 \sim q_3$

(jeweils für alle $q, q_1, q_2, q_3 \in Q$)

Eigenschaft: \sim ist eine Äquivalenzrelation.

Außerdem gilt für alle $a \in \Sigma$

- wenn $q_1 \sim q_2$ dann $\delta(q_1, a) \sim \delta(q_2, a)$, falls diese Übergänge definiert sind
(daher nehmen wir im Folgenden oft eine totale Übergangsfunktion an)

Eigenschaft: \sim ist verträglich mit der Übergangsfunktion.

Notation für Äquivalenzrelationen

Wir verwenden die bei Äquivalenzen üblichen Begriffe:

Wir schreiben $[q]_\sim$ für die \sim -Äquivalenzklasse von q , d.h.

$$[q]_\sim = \{p \in Q \mid q \sim p\}.$$

Für eine Menge $P \subseteq Q$ schreiben wir $P/_\sim$ für den Quotienten von P und \sim :

$$P/_\sim = \{[p]_\sim \mid p \in P\}.$$

(Die Quotientenbildung heißt Faktorisierung; sie entspricht dem „Verschmelzen“ äquivalenter Zustände.)

Wie immer gilt:

- Wenn $q_1 \sim q_2$ dann $[q_1]_\sim = [q_2]_\sim$
- Unterschiedliche Äquivalenzklassen sind disjunkt, d.h. $[q_1]_\sim \neq [q_2]_\sim$ impliziert $[q_1]_\sim \cap [q_2]_\sim = \emptyset$
- Die Äquivalenzklassen partitionieren Q , d.h. Q ist die Vereinigung der (disjunkten) Klassen

Der Quotientenautomat

Wir vereinfachen Automaten, indem wir äquivalente Zustände verschmelzen:

Für einen DFA $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, F \rangle$ mit totaler Übergangsfunktion ist der Quotientenautomat \mathcal{M}/\sim gegeben durch $\mathcal{M}/\sim = \langle Q/\sim, \Sigma, \delta_\sim, [q_0]_{\sim_M}, F/\sim \rangle$ wobei gilt:

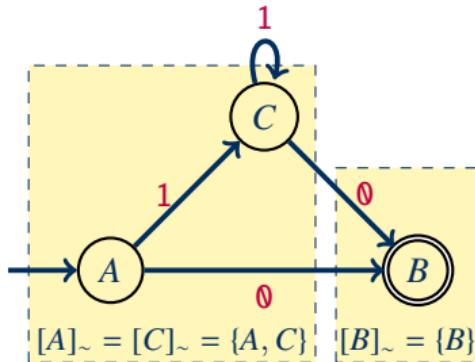
- $Q/\sim = \{[q]_\sim \mid q \in Q\}$
- $\delta_\sim([q]_\sim, a) = [\delta(q, a)]_\sim$
- $F/\sim = \{[q]_\sim \mid q \in F\}$

Diese Definition ergibt Sinn, da gilt:

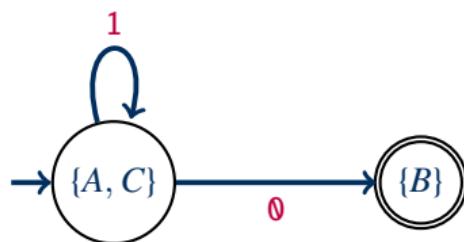
- wenn $[q]_\sim = [p]_\sim$ dann $[\delta(q, a)]_{\sim_M} = [\delta(p, a)]_{\sim_M}$ (Verträglichkeit von \sim und δ ; benötigt totale Übergangsfunktion)
- wenn $[q]_\sim = [p]_\sim$ dann $q \in F$ gdw. $p \in F$ (Übung)

~ Definition unabhängig vom gewählten Repräsentanten von $[q]_\sim$

Beispiel



Es ergibt sich der folgende Quotientenautomat:



Korrektheit Quotientenautomat

Satz: Für jeden totalen DFA \mathcal{M} gilt $\mathbf{L}(\mathcal{M}) = \mathbf{L}(\mathcal{M}/\sim)$.

Beweis: Für alle $w \in \Sigma^*$ gilt:

$$w \in \mathbf{L}(\mathcal{M}) \text{ gdw. } \delta(q_0, w) \in F \quad \text{laut Definition}$$

$$\text{gdw. } [\delta(q_0, w)]_\sim \in F/\sim \quad \text{wie zuvor bemerkt (Übung)}$$

$$\text{gdw. } \delta_\sim([q_0]_\sim, w) \in F/\sim \quad \text{Lemma } \heartsuit$$

$$\text{gdw. } w \in \mathbf{L}(\mathcal{M}/\sim) \quad \text{laut Definition}$$

Lemma \heartsuit : Für beliebige $q \in Q$ und $w \in \Sigma^*$ gilt:

$$[\delta(q, w)]_\sim = \delta_\sim([q]_\sim, w).$$

Beweis durch Induktion über $|w|$ (Übung)

□

Berechnung von \sim_M

Wie kann man \sim_M praktisch ermitteln?

Zuvor bemerkten wir:

- (1) Wenn $q_1 \sim q_2$ dann $q_1 \in F$ gdw. $q_2 \in F$
- (2) Wenn $q_1 \sim q_2$ dann $\delta(q_1, a) \sim \delta(q_2, a)$

Umgekehrt gilt also:

- (1') Wenn $q_1 \in F$ und $q_2 \notin F$ dann $q_1 \not\sim q_2$
- (2') Wenn $\delta(q_1, a) \not\sim \delta(q_2, a)$ dann $q_1 \not\sim q_2$

Tatsächlich ist $\not\sim$ die kleinste Relation, die (1') und (2') erfüllt.

→ Wir können $\not\sim$ (und damit auch \sim) durch rekursive Anwendung der Regeln (1') und (2') berechnen

Algorithmus zur Berechnung von $\sim_{\mathcal{M}}$

Eingabe: DFA $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, F \rangle$

Ausgabe: $\sim_{\mathcal{M}}$

- Initialisiere $\sim := \emptyset$
- (Regel 1) Für jedes Paar von Zuständen $\langle q, p \rangle \in Q \times Q$:
falls $q \in F$ und $p \notin F$, dann „speichere $q \sim p$ “
- (Regel 2) Für jedes Paar $\langle q, p \rangle \in Q \times Q \setminus \sim$ und jedes $a \in \Sigma$:
falls $\delta(q, a) \sim \delta(p, a)$ dann „speichere $q \sim p$ “
- Wiederhole die Anwendung von Regel 2 solange, bis es keine Änderungen mehr gibt
- Das Ergebnis ist $(Q \times Q) \setminus \sim$

Darstellung von \sim im Algorithmus

Die Anweisung „speichere $q \sim p$ “ könnte umgesetzt werden als:

$$\sim := \sim \cup \{\langle q, p \rangle, \langle p, q \rangle\}$$

Es ist aber nicht nötig, alle Paare in \sim einzeln zu speichern:

- \sim ist irreflexiv, d.h. man muss $q \sim q$ nicht betrachten
- \sim ist symmetrisch, d.h. man muss jeweils nur entweder $q \sim p$ oder $p \sim q$ betrachten

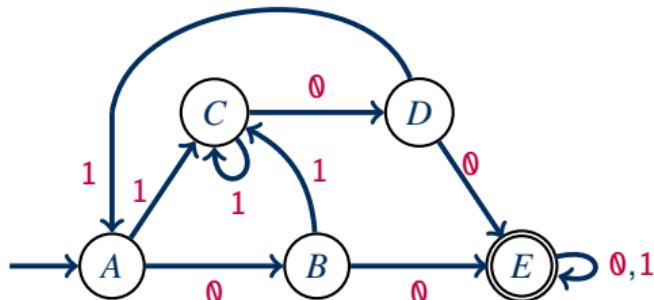
→ Halb-Tabelle genügt zum Eintragen der möglichen Paare

Beispiel: Für einen DFA mit Zuständen
 $Q = \{A, B, C, D, E\}$ genügt eine Tabelle
mit zehn Feldern (statt $5^2 = 25$).

(dazu reihen wir Zustände vertikal in umgekehrter Reihenfolge)

| | A | B | C | D |
|---|---|---|---|---|
| E | | | | |
| D | | | | |
| C | | | | |
| B | | | | |

Beispiel Quotientenautomat



- (1) $q \in F$ und $p \notin F$
impliziert $q \not\sim p$
- (2) $\delta(q, a) \not\sim \delta(p, a)$
impliziert $q \not\sim p$

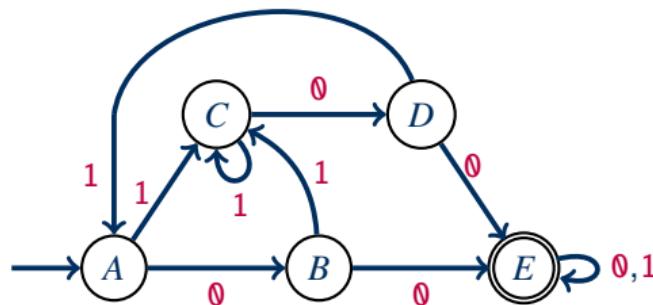
Wir tragen in der Tabelle jeweils die Wörter ein, die $q \not\sim p$ zeigen:

| | A | B | C | D |
|---|------------|------------|------------|------------|
| E | ϵ | ϵ | ϵ | ϵ |
| D | 0 | | 0 | |
| C | | 0 | | |
| B | 0 | | | |

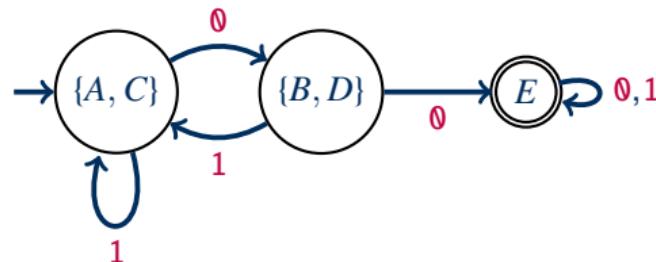
Weitere Abarbeitung von Regel (2) führt
nicht mehr zu Änderungen

$$\sim = \{\langle B, D \rangle, \langle D, B \rangle, \langle A, C \rangle, \langle C, A \rangle\} \cup \{\langle q, q \rangle \mid q \in Q\}$$

Beispiel Quotientenautomat



Quotientenautomat:



Reduktion von Automaten

Wir können das bisher gezeigte zusammenfassen:

Sei M ein DFA mit totaler Übergangsfunktion. Der reduzierte Automat M_r ergibt sich durch folgende Schritte:

- (1) Entferne alle unerreichbaren Zustände aus M
- (2) Berechne den Quotientenautomaten

Dieses Verfahren erzeugt den gewünschten minimalen DFA:

Satz: M_r ist bezüglich der Zustandsmenge der minimale DFA mit totaler Übergangsfunktion, der die Sprache $L(M)$ erkennt.

Zudem stellt sich heraus, dass dieser minimale DFA eindeutig ist:

Satz: Alle minimalen DFA mit totaler Übergangsfunktion, die $L(M)$ erkennen, sind bis auf Umbenennung von Zuständen gleich (sie sind isomorph). Daher hängt M_r nur von $L(M)$ ab, nicht von M .

Satz: M_r ist bezüglich der Zustandsmenge der minimale DFA mit totaler Übergangs-funktion, der die Sprache $L(M)$ erkennt.

Beweisplan:

1. M_r erkennt $L(M)$: Dies folgt aus der Korrektheit der Quotientenbildung bei Automaten
2. M_r ist minimal für diese Eigenschaft: Wir werden dies in mehreren Schritten zeigen:
 - Wir konstruieren einen weiteren minimalen Automaten M_L direkt aus L
 - Wir zeigen, dass M_L und M_r bis auf Umbenennung von Zuständen gleich sind

Damit ist auch die behauptete Eindeutigkeit gezeigt.

Die Nerode-Rechtskongruenz

Für eine Sprache $L \subseteq \Sigma^*$ ist die **Nerode-Rechtskongruenz** \simeq_L wie folgt definiert. Für Wörter $u, v \in \Sigma^*$ sei $u \simeq_L v$ wenn gilt:

für alle $w \in \Sigma^*$ gilt $uw \in L$ genau dann wenn $vw \in L$.

Wenn L klar ist, dann schreiben wir einfach \simeq statt \simeq_L

Anders gesagt: zwei Wörter v und u sind kongruent, wenn man in einem Wort das Präfix v gegen u vertauschen kann, ohne dass dies den Status des Worts bezüglich L verändert

Dies kann mit der Idee der Zustandsäquivalenz verglichen werden:

(Rückblick) Für Zustände $p, q \in Q$ sei $p \sim q$ wenn gilt:

für alle $w \in \Sigma^*$ gilt $\delta(p, w) \in F$ genau dann wenn $\delta(q, w) \in F$.

Eigenschaften von \simeq

Definition (kurz): $u \simeq_L v$ wenn für alle $w \in \Sigma^*$ gilt: $uw \in L$ genau dann wenn $vw \in L$.

Damit sehen wir leicht:

- \simeq ist reflexiv: $u \simeq u$
- \simeq ist symmetrisch: wenn $u \simeq v$ dann $v \simeq u$
- \simeq ist transitiv: wenn $u \simeq v$ und $v \simeq w$ dann $u \simeq w$

(jeweils für alle $u, v, w \in \Sigma^*$)

Eigenschaft: \simeq ist eine Äquivalenzrelation.

Außerdem gilt für alle $w \in \Sigma^*$

- wenn $u \simeq v$ dann $uw \simeq vw$

Eigenschaft: \simeq ist verträglich mit der Konkatenation von rechts.

Dies rechtfertigt die Bezeichnung Rechtskongruenz.

Beispiel

Die Sprache $\mathbf{L} = \{a\}^* \{b\}^*$ hat die folgenden Nerode-Äquivalenzklassen:

- $[\epsilon]_{\sim} = \{a\}^*$: für jedes $v \in [\epsilon]_{\sim}$ ist $vw \in \mathbf{L}$ gdw. $w \in \mathbf{L}$
- $[b]_{\sim} = \{a\}^* \{b\}^+$: für jedes $v \in [b]_{\sim}$ ist $vw \in \mathbf{L}$ gdw. $w \in \{b\}^*$
- $[ba]_{\sim} = \Sigma^* \setminus \mathbf{L}$: für jedes $v \in [ba]_{\sim}$ ist $vw \notin \mathbf{L}$ für alle $w \in \Sigma^*$

Die endliche Sprache $\mathbf{L} = \{a, ab, ba\}$ hat die folgenden Nerode-Äquivalenzklassen:

- $[\epsilon]_{\sim} = \{\epsilon\}$: $\epsilon w \in \mathbf{L}$ gdw. $w \in \mathbf{L}$
- $[a]_{\sim} = \{a\}$: $aw \in \mathbf{L}$ gdw. $w \in \{\epsilon, b\}$
- $[b]_{\sim} = \{b\}$: $bw \in \mathbf{L}$ gdw. $w = a$
- $[ab]_{\sim} = \{ab, ba\}$: für jedes $v \in [ab]_{\sim}$ ist $vw \in \mathbf{L}$ gdw. $w = \epsilon$
- $[bb]_{\sim} = \Sigma^* \setminus \{\epsilon, a, b, ab, ba\}$: für jedes $v \in [bb]_{\sim}$ ist $vw \notin \mathbf{L}$ für alle $w \in \Sigma^*$

Beispiel (2)

Die Sprache $L = \{a^n b^n \mid n \geq 0\}$ hat die folgenden Nerode-Äquivalenzklassen:

- $[\epsilon]_{\sim} = \{\epsilon\}$: $\epsilon w \in L$ gdw. $w \in L$
- $[a]_{\sim} = \{a\}$: $a w \in L$ gdw. $w \in \{a^n b^{n+1} \mid n \geq 0\}$
- $[aa]_{\sim} = \{aa\}$: $aa w \in L$ gdw. $w \in \{a^n b^{n+2} \mid n \geq 0\}$
- $[aaa]_{\sim} = \{aaa\}$: $aaa w \in L$ gdw. $w \in \{a^n b^{n+3} \mid n \geq 0\}$
- ... unendlich viele Äquivalenzklassen $[a^n]_{\sim} = \{a^n\}$

Es gibt weitere Formen von Äquivalenzklassen, z.B. $[aab] = \{a^{n+1} b^n \mid n \geq 0\}$.

↪ $L = \{a^n b^n \mid n \geq 0\}$ hat unendlich viele Nerode-Äquivalenzklassen

\simeq und reguläre Sprachen

Wir werden zeigen, dass jede reguläre Sprache endlich viele \simeq -Äquivalenzklassen hat.

Es gilt sogar noch etwas stärkeres:

Satz (Myhill & Nerode): Eine Sprache L ist genau dann regulär, wenn \simeq_L endlich viele Äquivalenzklassen hat.

Beweis: Siehe nächste Vorlesung.

Zusammenfassung und Ausblick

Im **Quotientenautomaten** werden äquivalente Zustände verschmolzen

Äquivalente Zustände in einem (totalen) DFA können rekursiv ermittelt werden

Der **Satz von Myhill und Nerode** charakterisiert reguläre Sprachen

Offene Fragen:

- Wie geht es weiter mit dem Beweis der Eindeutigkeit des Minimalautomaten?
- Wie aufwändig sind die verschiedenen Konstruktionen auf regulären Sprachen?
- Welche Sprachen sind nicht regulär?